Smarter. Greener. Together.

Industrial Automation Headquarters

Delta Electronics, Inc.
Taoyuan Technology Center
No.18, Xinglong Rd., Taoyuan District,
Taoyuan City 33068, Taiwan
TEL: 886-3-362-6301 / FAX: 886-3-371-6301

Asia

Delta Electronics (Shanghai) Co., Ltd.
No.182 Minyu Rd., Pudong Shanghai, P.R.C.
Post code : 201209
TEL: 86-21-6872-3988 / FAX: 86-21-6872-3996
Customer Service: 400-820-9595

Delta Electronics (Japan), Inc.
Tokyo Office
Industrial Automation Sales Department
2-1-14 Shibadaimon, Minato-ku
Tokyo, Japan 105-0012
TEL: 81-3-5733-1155 / FAX: 81-3-5733-1255

Delta Electronics (Korea), Inc.
Seoul Office
1511, 219, Gasan Digital 1-Ro., Geumcheon-gu,
Seoul, 08501 South Korea
TEL: 82-2-515-5305 / FAX: 82-2-515-5302

Delta Energy Systems (Singapore) Pte Ltd.
4 Kaki Bukit Avenue 1, #05-04, Singapore 417939
TEL: 65-6747-5155 / FAX: 65-6744-9228

Delta Electronics (India) Pvt. Ltd.
Plot No.43, Sector 35, HSIIDC Gurgaon,
PIN 122001, Haryana, India
TEL: 91-124-4874900 / FAX : 91-124-4874945

Delta Electronics (Thailand) PCL.
909 Soi 9, Moo 4, Bangpoo Industrial Estate (E.P.Z),
Pattana 1 Rd., T.Phraksa, A.Muang,
Samutprakarn 10280, Thailand
TEL: 66-2709-2800 / FAX : 662-709-2827

Delta Electronics (Australia) Pty Ltd.
Unit 20-21/45 Normanby Rd., Notting Hill Vic 3168, Australia
TEL: 61-3-9543-3720

Americas

Delta Electronics (Americas) Ltd.
Raleigh Office
P.O. Box 12173, 5101 Davis Drive,
Research Triangle Park, NC 27709, U.S.A.
TEL: 1-919-767-3813 / FAX: 1-919-767-3969

Delta Greentech (Brasil) S/A
São Paulo Office
Rua Itapeva, 26 – 3˚ Andar - Bela Vista
CEP: 01332-000 – São Paulo – SP - Brasil
TEL: 55-11-3530-8643 / 55-11-3530-8640

Delta Electronics International Mexico S.A. de C.V.
Mexico Office
Gustavo Baz No. 309 Edificio E PB 103
Colonia La Loma, CP 54060
Tlalnepantla, Estado de México
TEL: 52-55-3603-9200

EMEA

Headquarters: Delta Electronics (Netherlands) B.V.
Sales: Sales.IA.EMEA@deltaww.com
Marketing: Marketing.IA.EMEA@deltaww.com
Technical Support: iatechnicalsupport@deltaww.com
Customer Support: Customer-Support@deltaww.com
Service: Service.IA.emea@deltaww.com
TEL: +31(0)40 800 3900

BENELUX: Delta Electronics (Netherlands) B.V.
De Witbogt 20, 5652 AG Eindhoven, The Netherlands
Mail: Sales.IA.Benelux@deltaww.com
TEL: +31(0)40 800 3900

DACH: Delta Electronics (Netherlands) B.V.
Coesterweg 45, D-59494 Soest, Germany
Mail: Sales.IA.DACH@deltaww.com
TEL: +49(0)2921 987 0

France: Delta Electronics (France) S.A.
ZI du bois Challand 2, 15 rue des Pyrénées,
Lisses, 91090 Evry Cedex, France
Mail: Sales.IA.FR@deltaww.com
TEL: +33(0)1 69 77 82 60

Iberia: Delta Electronics Solutions (Spain) S.L.U
Ctra. De Villaverde a Vallecas, 265 1º Dcha Ed.
Hormigueras – P.I. de Vallecas 28031 Madrid
TEL: +34(0)91 223 74 20

Carrer Llacuna 166, 08018 Barcelona, Spain
Mail: Sales.IA.Iberia@deltaww.com

Italy: Delta Electronics (Italy) S.r.l.
Via Meda 2–22060 Novedrate(CO)
Piazza Grazioli 18 00186 Roma Italy
Mail: Sales.IA.Italy@deltaww.com
TEL: +39 039 8900365

Russia: Delta Energy System LLC
Vereyskaya Plaza II, office 112 Vereyskaya str.
17 121357 Moscow Russia
Mail: Sales.IA.RU@deltaww.com
TEL: +7 495 644 3240

Turkey: Delta Greentech Elektronik San. Ltd. Sti. (Turkey)
Şerifali Mah. Hendem Cad. Kule Sok. No:16-A
34775 Ümraniye – İstanbul
Mail: Sales.IA.Turkey@deltaww.com
TEL: + 90 216 499 9910

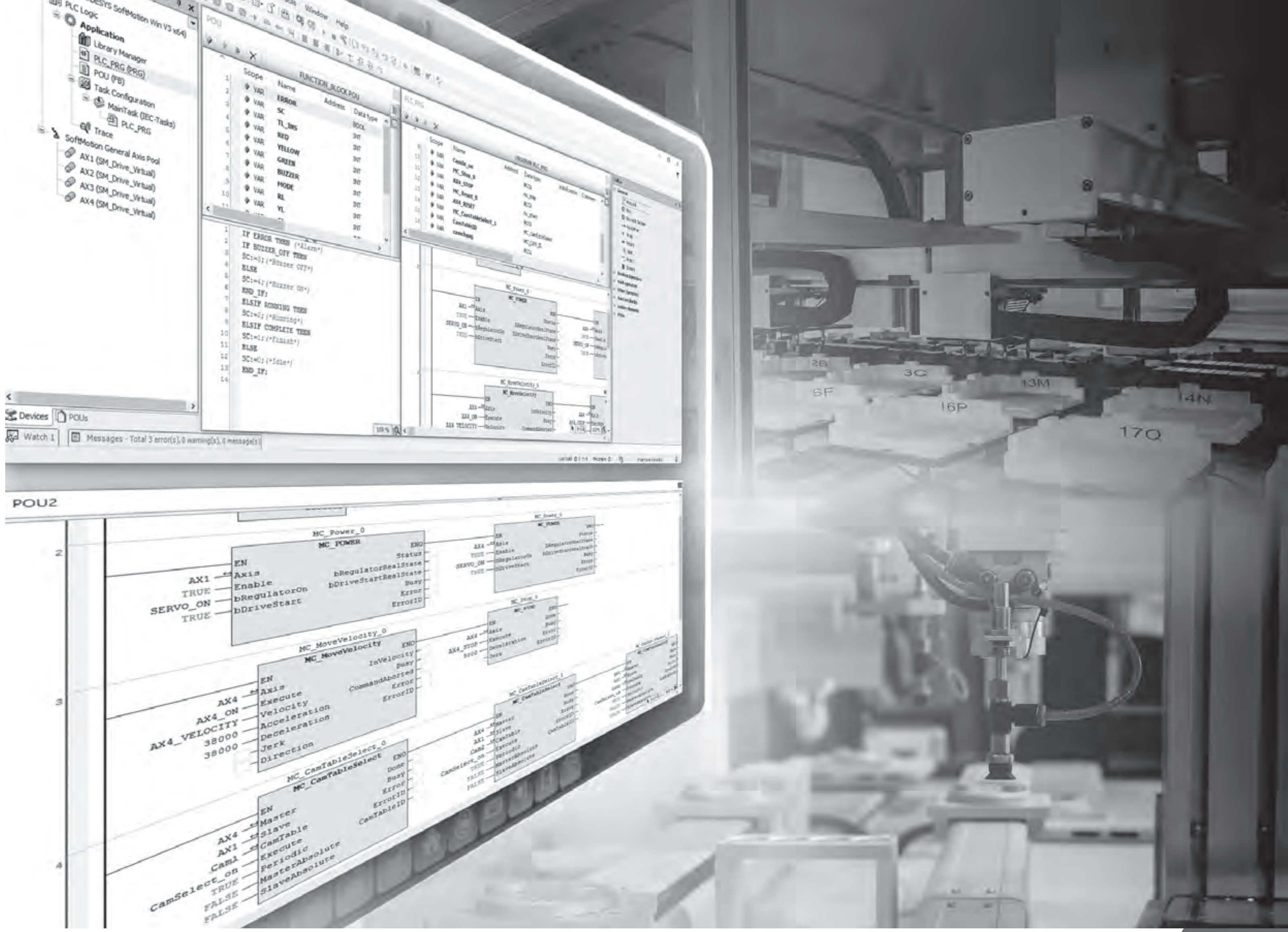
GCC: Delta Energy Systems AG (Dubai BR)
P.O. Box 185668, Gate 7, 3rd Floor, Hamarain Centre
Dubai, United Arab Emirates
Mail: Sales.IA.MEA@deltaww.com
TEL: +971(0)4 2690148

Egypt + North Africa: Delta Electronics
Unit 318, 3rd Floor, Trivium Business Complex, North 90 street,
New Cairo, Cairo, Egypt
Mail: Sales.IA.MEA@deltaww.com

*We reserve the right to change the information in this catalogue without prior notice.

AX-0326920-01
2020/11/09

AX Series Motion Controller Manual



AX Series
Motion Controller Manual

www.deltaww.com

Smarter. Greener. Together.

AX Series Motion Controller Manual

Revision History

Version	Revision	Date
1st	The first version was published.	2020/11/09

AX Series Motion Controller Manual

Table of Contents

Preface

Chapter 1 Introduction to Motion Control

Chapter 2 Motion Control Instructions

Appendix A

Preface

Table of Contents

P.1 Introduction

Thank you for purchasing AX series Motion Controller with our advanced motion control system. Delta’s AX series motion controller based on COSESYS intergrates the control function of PLCs and motion controllers into one platform.

This manual introduces PLCOpen standard motion control instructions and Delta-defined instructions including single-axis, multi-axes instructions and motion control applications.

Please ensure that you fully understand the configuration and operations of the AX series motion control system, and use the AX series Motion Controller CPU correctly.

P.1.1 Applicable Products

This manual relates to the following products

- **AX-3 series/ AX-8 series**

P.1.2 Associated Manuals

The related manuals of the AX Motion Controller series are composed of the following.

1. **DIADesigner-AX User Manual**
   Contents include the use of DIADesigner-AX, the programming languages (ladder diagrams, sequential function charts, function block diagrams, and structured texts), the concept of POUs and Task, and the operation of motion control programming.

2. **AX-3 Series Operational Manual**
   It introduces basic knowledge of motion control structure, software/hardware setup, quick start of Software operations, devices to be used, motion control operations, troubleshooting, Input/ Output modules, modules of temperature measurement and etc.

3. **AX-8 Series Operational Manual**
   It introduces basic knowledge of motion control structure, software/hardware setup, quick start of Software operations, devices to be used, motion control operations, troubleshooting, Input/ Output modules, modules of temperature measurement and etc.

1

Chapter 1 Introduction to Motion Control

Table of Contents

1.1 Introductions of Motion Control

This manual introduces the elements for motion control programming including devices, symbols and motion control instructions.

Motion control instructions are defined as function blocks (FB) and are used in the program for performing a variety of motion control purposes.The motion control (MC) instructions are developed based on the specifications of PLCopen* motion control function blocks. In addition to the PLCopen-based instructions, Codesys also provides Delta-defined function blocks for users to achieve complete motion control applications.

This section gives an overview of the motion control instructions for both PLCopen-based function blocks and Delta-defined function blocks. PLCopen defines the program and function block interfaces so as to achieve a standardized motion control programming environment for the languages specified in IEC61131-3. Using PLCopen-based instructions together with Delta-defined instructions reduces the costs for training and support.

Before using the instructions, please be sure that you understand the devices, symbols and the function of instructions sufficiently.

You can also refer to the Appendices for a quick reference of the motion control instruction list and error codes.

***Note:**

PLCopen is an organization promoting industrial control based on IEC61131-3, which is an international standard widely adopted for PLC programming. For more information regarding PLCopen, check the official website at: http://www.plcopen.org/

1.1.1 Basic Knowledge on Motion Control Instructions

Using motion control instructions requires the basic knowledge of motion control defined in the specifications of PLCopen motion control function blocks. This section provides an overview of these specifications.

- **Name of Motion Control Instructions**

  PLCopen-based motion control instructions begin with “MC_”, while Delta-defined function block instructions begin with “DFB_”.

Type	Description
MC_	PLCopen-based motion control instructions
DMC_	Delta-defined function block instructions*

  ***Note:** Delta-defined function block instructions (DFB) include Delta-defined motion control function blocks and other administrative/non-administrative function blocks which are applicable for AH Motion series CPUs. Therefore you can look up a function block (FB) in this manual.

- **Types of Motion Control Related Instructions**

  Different categories of motion control instructions are divided by functions such as single-axis motion instructions. Refer to Ch2 Motion Control Instructions for more details.

- **Execution of a Function Block**

  Function block instructions generally include two types of inputs for execution: Execute and Enable. When the instruction is executed or enabled, the outputs of the function block can indicate the status. The basic outputs include Busy, Done, CommandAborted, and Error. For detailed information of inputs and outputs of each function block, refer to Ch2 Motion Control Instructions.

- **Error Handling**

  Information regarding error codes, indicators and troubleshooting are set out in Appendice A for a quick reference.

- **Re-execution of a Function Block**

  Re-execution of a function block refers to triggering Execute again after resetting it.You can change the input values and trigger Execute again while the function block is during operation (in busy status).Such output status would remain unchanged (in busy status) which also means the previous executed instruction would be aborted by Aborting of buffer mode.

- **Multi-execution of Multiple Motion Control Instructions**

  Multi-execution of motion control instructions refers to that multiple instructions on the same axis are executed in the same task execution period. The pattern of multi-execution is defined by the input variable BufferMode which is specified to blend the two motions. Therefore, the instruction at the back will determine the behavior of the previous instruction according to BufferMode. Please refer to AX-3 Series Operation Manual for more details.

- **Buffer Modes**

  Some motion instructions have an input called BufferMode.You can execute a different instruction instance during axis motion when the values for BufferMode are specified.This input decides whether the instruction executes immediately (non-buffered mode) or it waits till current motion instruction sets its status outputs (Done/InVelocity/InPosition, etc.)

BufferMode determines the behavior to combine the axis motions for this instruction and the previous instruction. When the instruction is executed;

- The selected buffer mode is valid if the previous instruction is executing.
- The selected buffer mode is invalid if the axis is in Standstill state.

- The following Buffer Modes are supported.

Buffer Mode	Function
0：Aborting	Aborts the ongoing motion. The next instruction takes effect immediately.
1：Buffered	Automatically executes the next instruction after the ongoing motion is completed.
2：BlendingLow	Takes the lower target velocity as the transit velocity between the current instruction and the buffered instruction. (The transit velocity is the velocity that the current instruction uses as the transit point.)
3：BlendingPrevious	Takes the target velocity of the current instruction as the transit velocity.
4：BlendingNext	Takes the target velocity of the buffered instruction as the transit velocity.
5：BlendingHigh	Takes the higher target velocity as the transit velocity between the current instruction and the buffered instruction.

Please refer to AX-3 Series Operation Manual for more details of buffer mode.

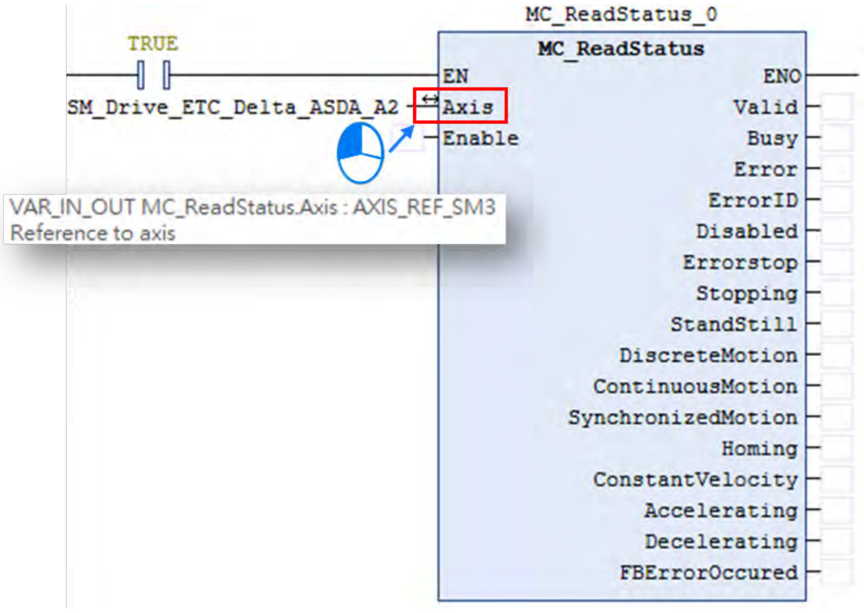
- **Structure Applicable for Motion Control**

  In PLCopen technical standard, the information and parameters required for configuring motion control on axis are defined in a Structure.

  For AX Motion CPUs, a Structure is a data type applicable to group the data elements together, which is easier for users to specify proper parameters.

  For AX Motion CPUs, the applicable Structure is as below:

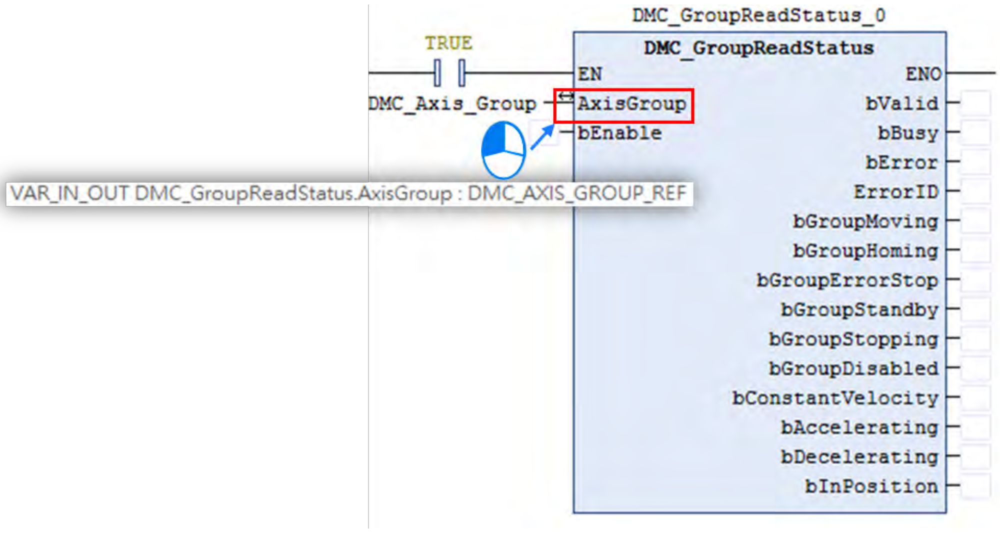
■ Single-axis Function Block



VAR_IN_OUT MC_ReadStatus.Axis : AXIS_REF_SM3
Reference to axis

For single-axis FB, the applicable Structure is AXIS_REF_SM3。

■ Function Block for Axis Group



VAR_IN_OUT DMC_GroupReadStatus.AxisGroup : DMC_AXIS_GROUP_REF

For AxisGroup FB, the applicable Structure is DMC_AXIS_GROUP_REF。

Note: For more details, please find Appendix A.2 Data Types: Enumeration and Structure.

1.2 Categories of Motion Control Instructions

For AX Motion CPUs, motion control instructions are divided into two categories based on PLCOpen.

Categories	Type	Function Group	Description
Single-axis motion control instructions	Motion	Positioning on single axis	“SMC”: Motion instructions “MC_”: PLCopen motion control instructions “DMC_”: Delta-defined motion control instructions
		Velocity control on single axis	
		Torque control on single axis	
		Synchronized control on single axis	
	Administrative	Administrative functions on single axis	
Multi-axis motion control instructions Motion modules	Motion	Multi-axis coordinated control	Performing coordinated movement of an axis group
	Administrative	Administrative functions on multi axis	Controlling, monitoring or resetting axis group status.

MEMO

Chapter2 Motion Control Instruction

Table of Contents

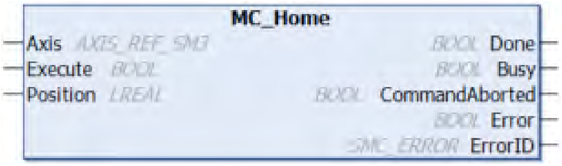
2.1 Motion Control Instructions

Motion control instructions are generally used to control motors on performing specific movement after the specified instruction being executed. The function blocks used in this chapter are from the library “SM3_Basic” and able to operate synchronously with drives. As a result, synchronous axis type should be selected in axis settings. For more details about configuration related to synchronous axes, please refer to section 7.4 in AX-3 Series Operational Manual.

2.1.1 MC_Home

MC_Home controls the axis to perform the homing operation.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Home	
		MC_Home
		Axis AXIS_REF_SM3
		Execute BOOL
		Position LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_Home_instance (
			Axis :=,
			Execute:=,
			Position:=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting value (Default value)	Timing for updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
Position	Specifies the set position. (Unit: user unit)	LREAL	Negative number, positive number, or 0 (0)	When Execute shifts to True and Busy is False.

- **Outputs**

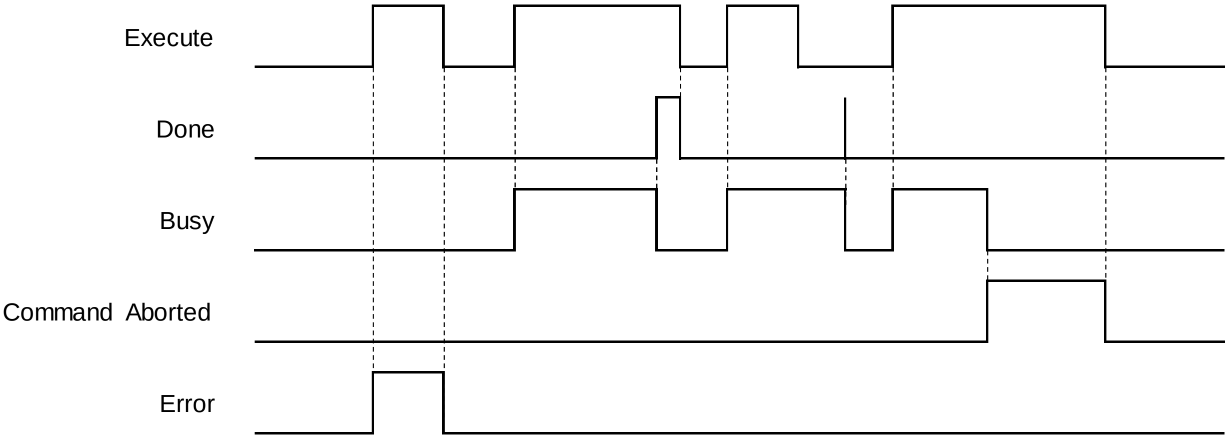
Name	Function	Data Type	Output range (Default value)
Done	True when homing is completed.	BOOL	True/False(False)
Busy	True when the instruction is executed	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR: Enumeration(Enum)

Outputs Update Timing

Name	Timing for shifting to True	Timing for shifting to False
Done	● When the homing is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done shifts to True. ● When Error shifts to True. ● When CommandAborted shifts to True.
CommandAborted	● When this instruction is aborted by another instruction. ● When this instruction is aborted by MC_Stop.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.(Error code is recorded)	● When Error code is cleared.
ErrorID		

Timing Diagram



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timing for Updating
Axis	Reference to axis..	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute is rising edge triggered and Busy is False.

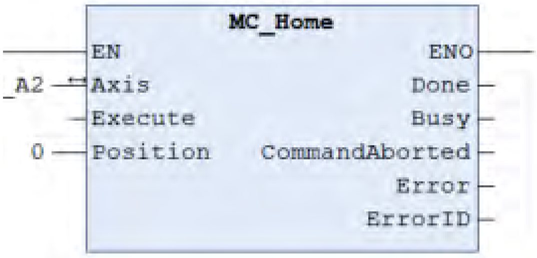
***Note**: AXIS_REF_SM3(FB): Each function block has this in-out pin for the activation of function block.

- **Function**
  - When Execute changes to True, the homing operation starts to be performed on the axis specified in Axis.
  - Position is to specify the set position for homing.
  - In case the MC_Home command is interrupted by MC_Stop and

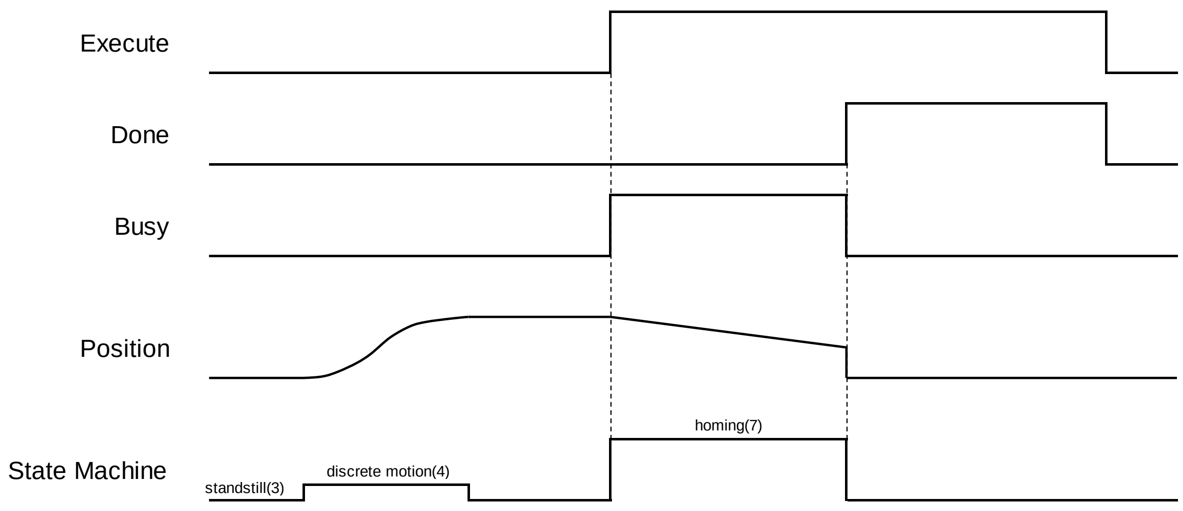
    xWaitForHaltWhenStopInterruptsHome is TRUE, MC_Stop has to wait till the driver reaches velocity zero before setting Done. Instead, if xWaitForHaltWhenStopInterruptsHome is FALSE, .Done will shift to true once MC_Stop interrupts MC_Home.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - Example 1: Explain on how to perform homing by using MC_Home and setting homing mode on parameter configuration page with the software DIAdesigner-AX.



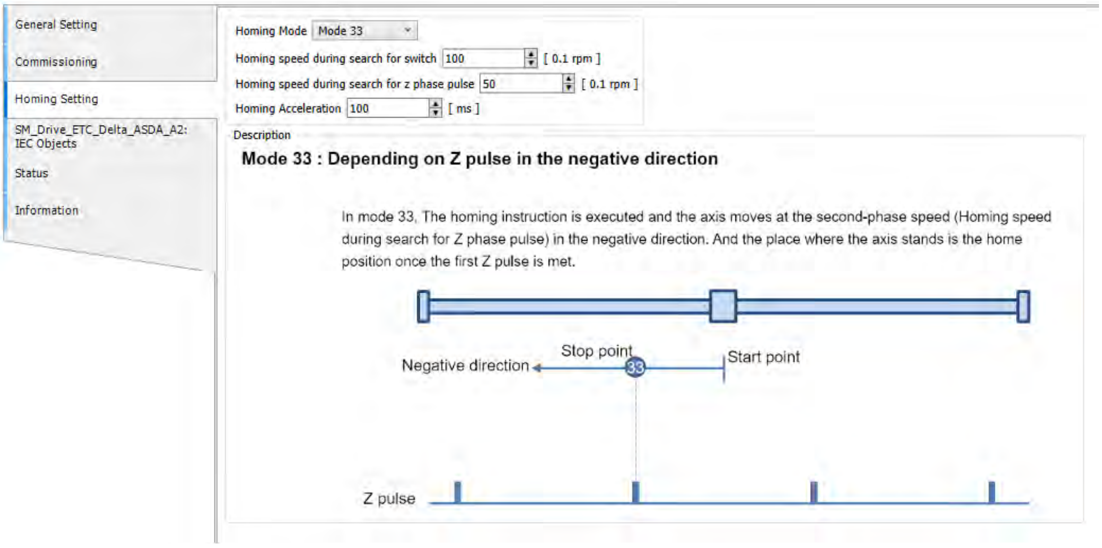
- **Timing Diagram**



- Use MC_Home with the following parameters:

Homing mode	Speed during search for switch	Speed during search for zero	Homing acceleration
33	100	50	100

- The above parameters can be configured on the Homing Setting page:



Homing Mode: Mode 33

Homing speed during search for switch: 100 [0.1 rpm]

Homing speed during search for z phase pulse: 50 [0.1 rpm]

Homing Acceleration: 100 [ms]

Description

Mode 33 : Depending on Z pulse in the negative direction

In mode 33, The homing instruction is executed and the axis moves at the second-phase speed (Homing speed during search for Z phase pulse) in the negative direction. And the place where the axis stands is the home position once the first Z pulse is met.

After MC_Home being executed, the axis would move in the negative direction till finds Z phase pulse. And the place where the axis stands is the home position once the first Z pulse is met.

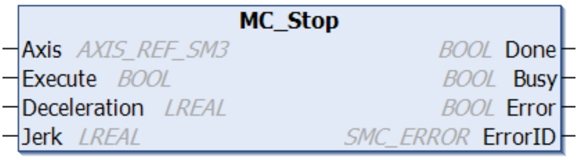
- **Supported Products**

- AX-308E、AX-8

2.1.2 MC_Stop

MC_Stop decelerates an axis to a stop.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Stop	 MC_Stop
		Axis AXIS_REF_SM3
		Execute BOOL
		Deceleration LREAL
		Jerk LREAL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_Stop_instance (
			Axis :=,
			Execute :=,
			Deceleration :=,
			Jerk :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timing for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
Deceleration	Deceleration rate (Unit: user unit/s2)*	LREAL	Positive number or 0(0)	When Execute is rising edge triggered, the rate would be updated.
Jerk	Jerk value (Unit: user unit/s3)*	LREAL	Positive number or 0(0)	When Execute is rising edge triggered, the value would be updated.

- **Outputs**

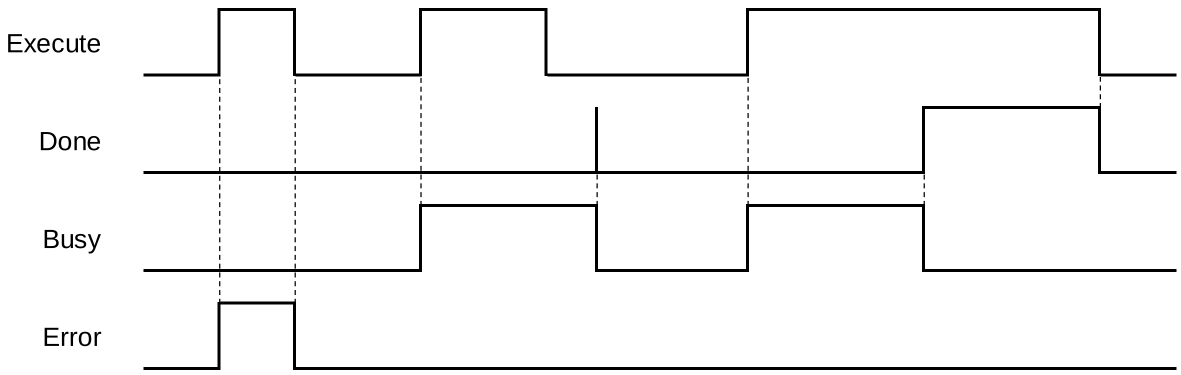
Name	Function	Data Type	Output Range (Default Value)
Done	True when zero velocity is reached.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR: Enumeration(Enum)

- **Output Updating Time**

Name	Timing for shifting to True	Timing for shifting to False
Done	● True when the axis decelerates to a stop and reaches zero velocity.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● True when Execute shifts to True.	● When Done shifts to True. ● When Error shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timing for Updating
Axis	Reference to axis..	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute is rising edge triggered and Busy is False.

***Note**: AXIS_REF_SM3(FB): Each function block has this in-out pin for the activation of function block.

- **Function**

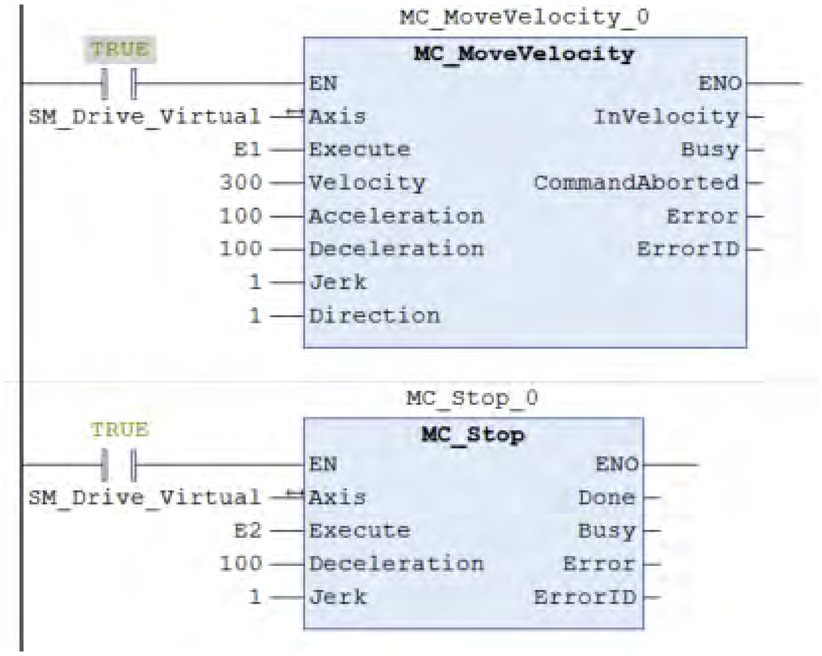
- You can specify the deceleration rate when decelerating the moving axis to a stop. Also, State Machine will be stopping.
- When MC_Power is False during deceleration, the motor is in Free Run state.
- The Done output is set to True when axis has reached velocity zero. At the same time, the input Execute changes to False, while State Machine in stopping state changes to standstill.

- **Troubleshooting**

  ■ If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

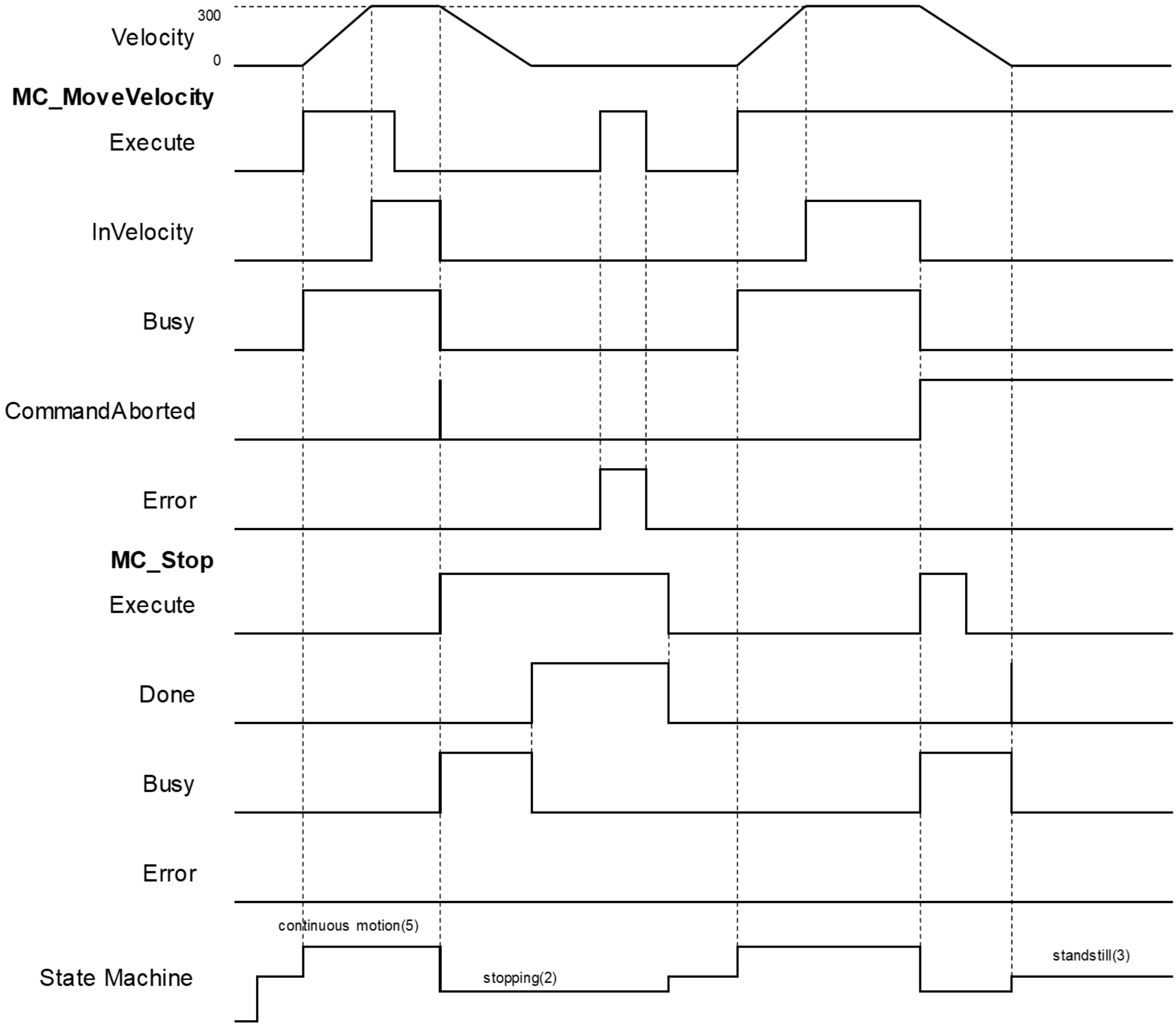
  ■ The example below shows the behavior and position tracking when MC_Stop is executed after MC_MoveVelocity.



MC_MoveVelocity_0
MC_MoveVelocity
TRUE — EN / ENO
SM_Drive_Virtual — Axis / InVelocity
E1 — Execute / Busy
300 — Velocity / CommandAborted
100 — Acceleration / Error
100 — Deceleration / ErrorID
1 — Jerk
1 — Direction

MC_Stop_0
MC_Stop
TRUE — EN / ENO
SM_Drive_Virtual — Axis / Done
E2 — Execute / Busy
100 — Deceleration / Error
1 — Jerk / ErrorID

- **Timing Diagram**



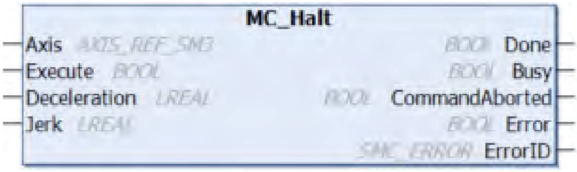
- When Execute of MC_Stop changes to True, it triggers CommandAboted of MC_MoveVelocity at the same time and the motion controller starts to decelerate the axis to a stop. The Axis state is moved to the “Stopping”.
- When the axis reaches zero velocity, the Done output will change to True. Execute is still True so the axis state remains in the state “Stopping”. After the stop is finished and Execute is false, the axis will change to Standstill.
- In case MC_MoveVelocity is executed again while the axis state is “Stopping”, an error will be reported. (Error Code: SMC_AXIS_NOT_READY_FOR_MOTION)

- **Supported Products**
  - AX-308E、AX-8

2.1.3 MC_Halt

MC_Halt commands a controlled axis motion stop.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Halt	 MC_Halt
		Axis AXIS_REF_SM3
		Execute BOOL
		Deceleration LREAL
		Jerk LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_Halt_instance (
			Axis :=,
			Execute :=,
			Deceleration :=,
			Jerk :=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timing for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
Deceleration	Deceleration rate. (Unit: user unit/s2)	LREAL	Positive number or 0 (0)	When Execute shifts to True, the rate will be updated.
Jerk	Jerk value. (Unit: user unit/s3)	LREAL	Positive number or 0 (0)	When Execute is rising edge triggered, the value would be updated.

- **Outputs**

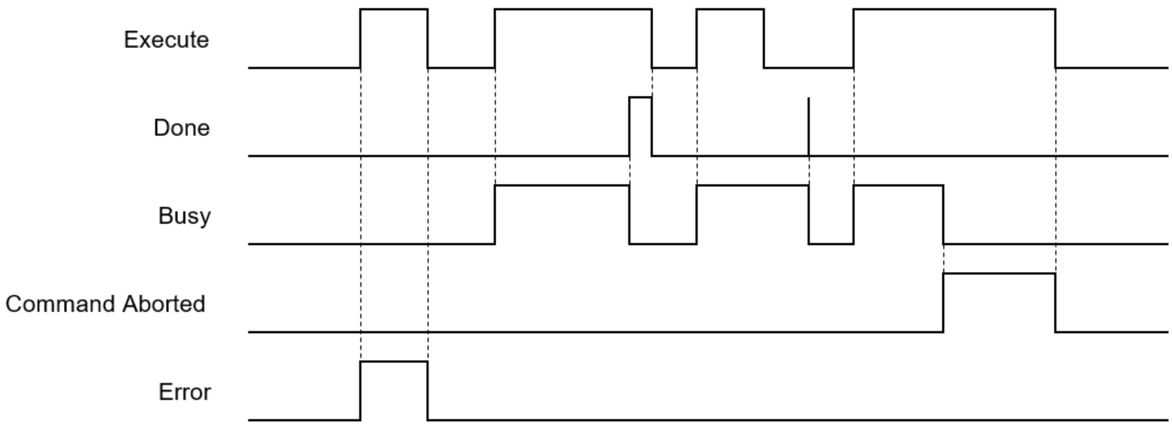
Name	Functioin	Data Type	Output Range (Default Value)
Done	True when zero velocity is reached.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR *	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR: Enumeration(Enum)

- **Output Updating Time**

Name	Timing for shifting to True	Timing for shifting to False
Done	● When the axis decelerates to a stop and reaches zero velocity.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute shifts to True.	● When Done shifts to True. ● When Error shifts to True. ● When CommandAborted shifts to True.
CommandAborted	● When this instruction is aborted because of other function blocks.	● When Execute shifts from True to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

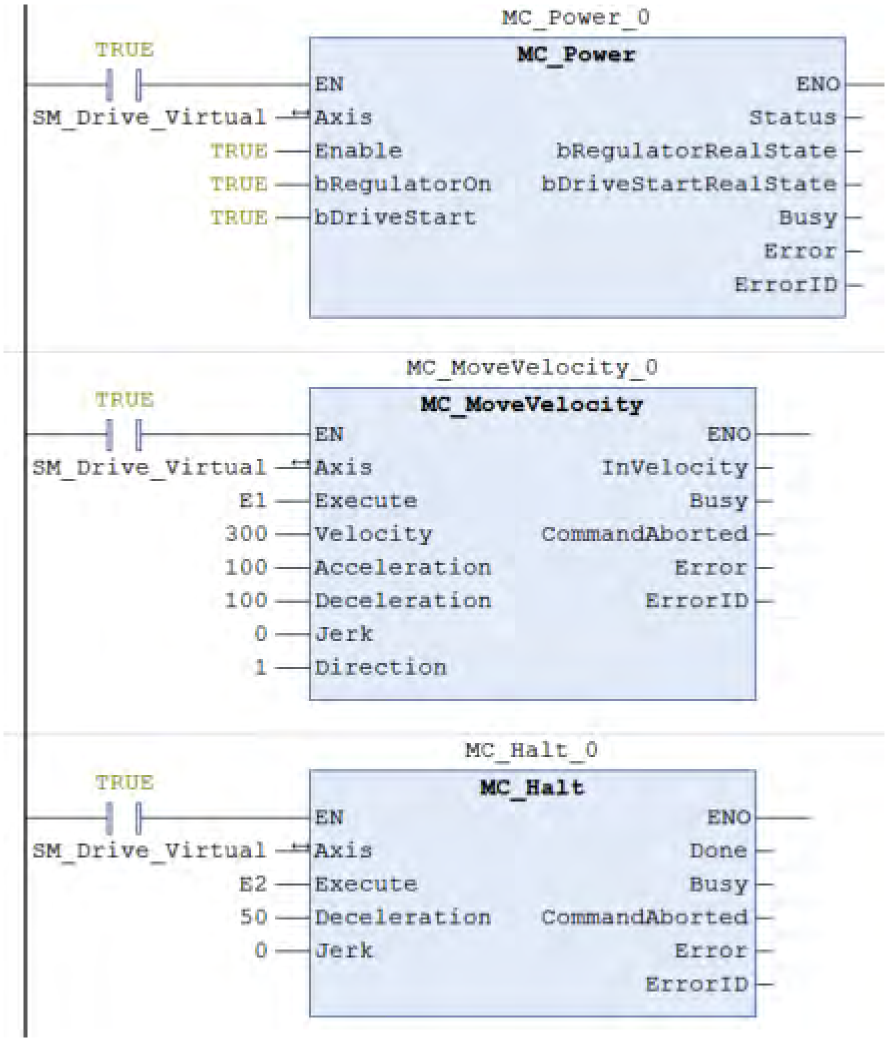
Name	Function	Data Type	Setting value (Default value)	Timing for updating
Axis	Reference to axis..	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute is rising edge triggered and Busy is False

***Note:** AXIS_REF_SM3(FB): Each function block has this in-out pin for the activation of function block.

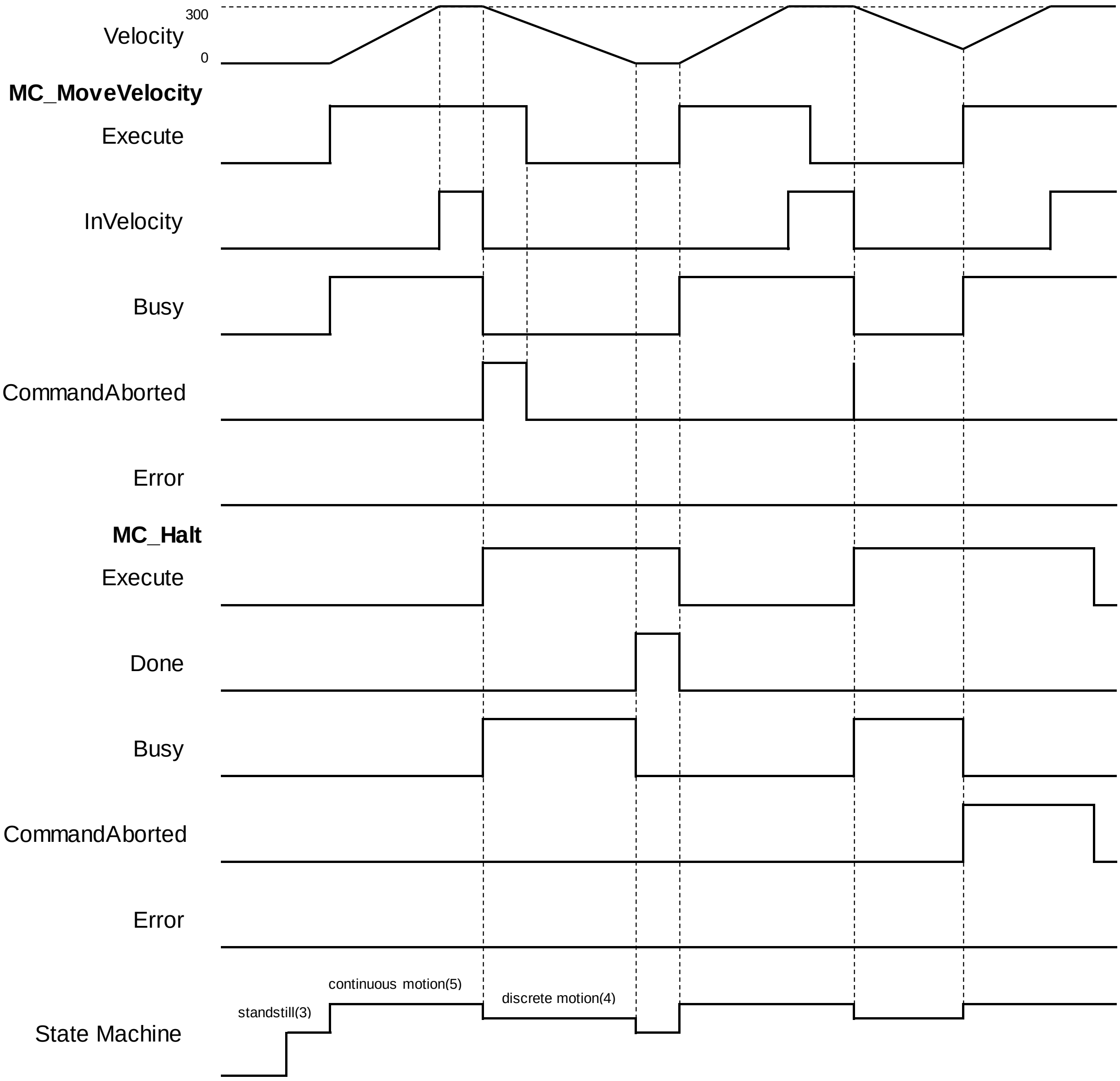
- **Function**
  - Any next motion command can be executed when MC_halt is in Standstill mode (opposite to MC_Stop, which cannot be interrupted by other motion FBs.).
  - When MC_Halt is executed, the axis will enter the discrete_motion state. Once the axis reaches zero, the axis state would transfer to Standstill.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below shows the behavior and position tracking when MC_Halt is executed after MC_MoveVelocity.
  - The MC_Halt stops MC_MoveVelocity if there is no another instruction executed before the axis enters “Standstill” state.l” 。
  - If MC_MoveVelocity executes again during the deceleration, it will abort MC_Halt immediately and accelerate again without entering “Standstill” state. This re-execution behavior is allowed for MC_Halt but not allowed iMC_Stop.



- **Timing Diagram**



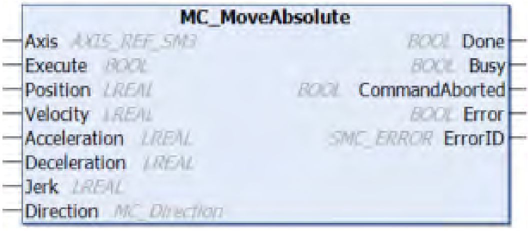
- ◆ When Execute of MC_Halt changes to True, it triggers CommandAborted of MC_MoveVelocity at the same time and the motion controller starts to decelerate the axis to a stop. The Axis state is moved to the “DiscreteMotion”.
- ◆ When the axis reaches zero velocity, Done will change to True. The axis state will transfer to “Standstill”..
- ◆ In case MC_Halt is not decelerating the axis to zero velocity and Execution is True, the Execute input of MC_MoveVelocity will change to True again and stop MC_Halt. Which CommandAboted will change to True with the axis state transferred from discrete_motion to continuous_motion.

- **Supported Products**
  - AX-308E、AX-8

2.1.4 MC_MoveAbsolute

MC_MoveAbsolute controls the axis to move to the specified absolute target position at a specified behavior.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_MoveAbsolute	 MC_MoveAbsolute
		Axis AXIS_REF_SM3
		Execute BOOL
		Position LREAL
		Velocity LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		Direction MC_Direction
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_MoveAbsolute_instance(
			Axis :=,
			Execute :=,
			Position :=,
			Velocity :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Direction :=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting value (Default value)	Timing for updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
Position	Absolute target position (Unit: user unit)	LREAL	Negative number, positive number or 0 (0)	When Execute shifts to True and Busy is False.
Velocity	Target velocity (Unit: user unit/s)	LREAL	Positive number or 0 (0)	When Execute shifts to True and Busy is False.
Acceleration	Acceleration rate (Unit: user unit/s^2)	LREAL	Positive number or 0 (0)	When Execute shifts to True and Busy is False.
Deceleration	Deceleration rate (Unit: user unit/s^2)	LREAL	Positive number or 0 (0)	When Execute shifts to True and Busy is False.
Jerk	Jerk value (Unit: user unit/s^3)	LREAL	Positive number or 0 (0)	When Execute shifts to True and Busy is False.
Direction	Rotation direction	MC_Direction*	3:fastest 2:current 1:positive 0:shortest -1:negative (shortest)	When Execute shifts to True and Busy is False.

***Note:** MC_Direction: Enumeration (Enum)

- **Outputs**

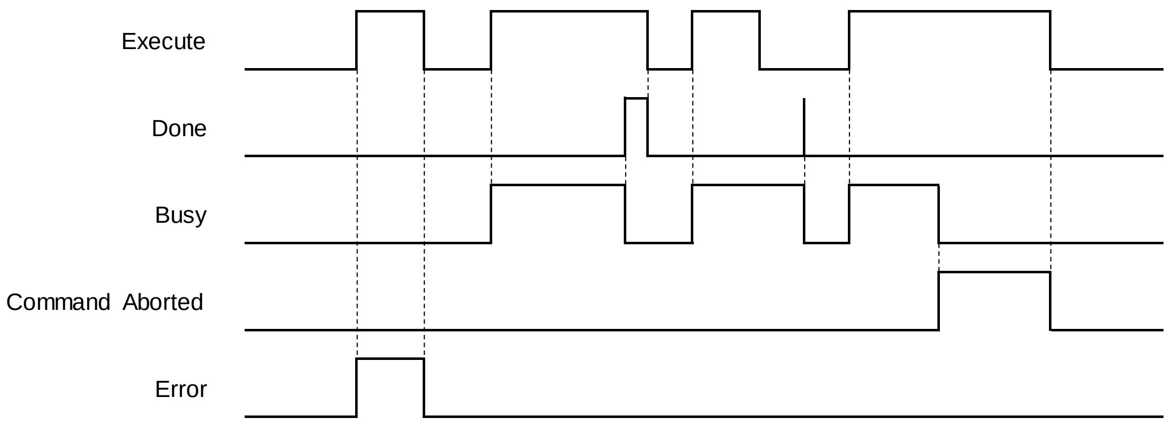
Name	Function	Data Type	Output Range (Default Value)
Done	True when absolute target position is reached.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the axis is being controlled	BOOL	True/False(False)
Error	True if an error occurs	BOOL	True/False(False)
ErrorID	Record the error code when the error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR: Enumeration (Enum)

- **Output Updating Time**

Name	Timing for Shifting to True	Timing for Shifting to False
Done	● When the absolute positioning is completed.	● When Execute shifts to False ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done shifts to True. ● When Error shifts to True. ● When CommandAborted shifts to True.
CommandAborted	● When this instruction is aborted by another function block. ● When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute shifts to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**

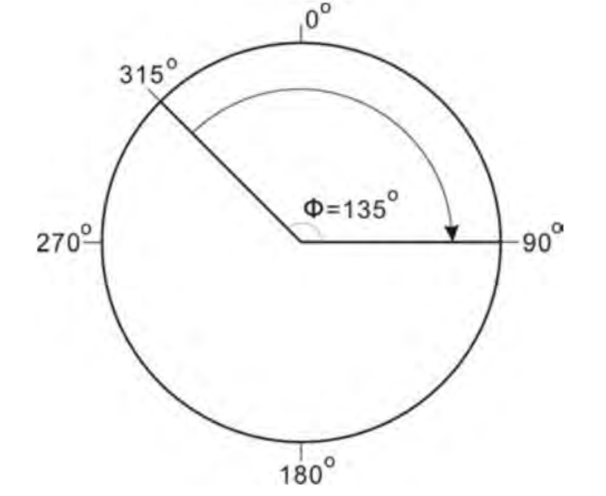
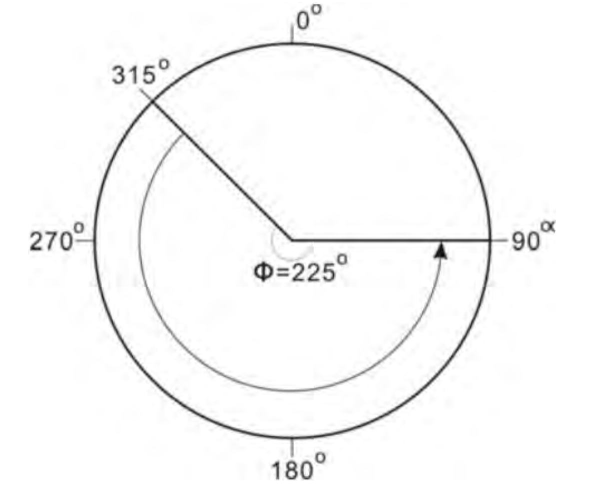


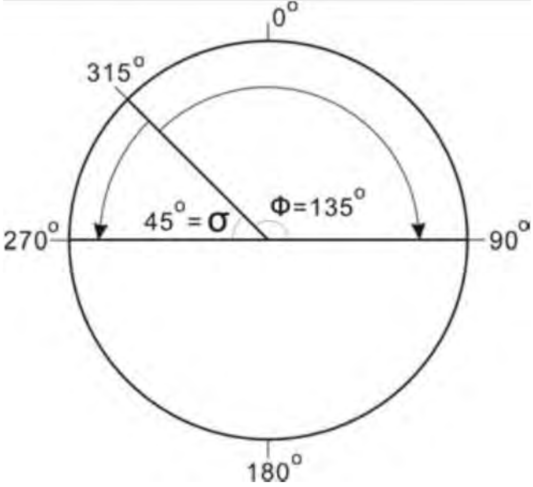
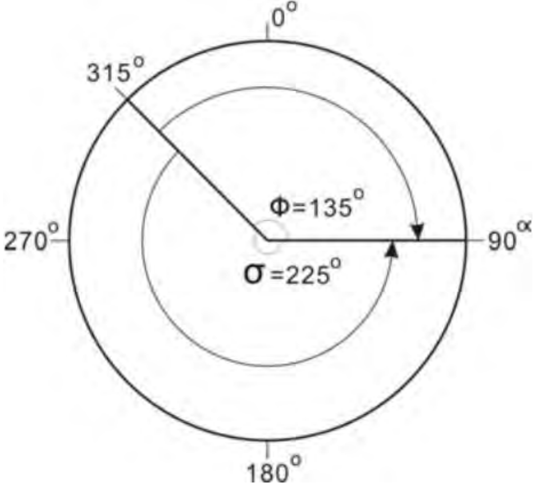
- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis..	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute is rising edge triggered and Busy is False.

***Note:** AXIS_REF_SM3(FB): Each function block has this in-out pin for the activation of function block.

- **Function**
  - **Direction**
    - Direction is used to define the rotation of servo axis and is effective only for modulo/rotary axis.
    - When the direction value is different, the motion direction and the travel distance of the rotary axis will be different as follows. Suppose the output unit of the physical device is “degree”, the motion direction of the rotary axis is illustrated as follows:

Direction: 1 (Positive direction) Current position: 315° Target position: 90° Movement angle: 135°	Direction: -1 (Negative direction) Current position: 315° Target position: 90° Movement angle: 225°
	

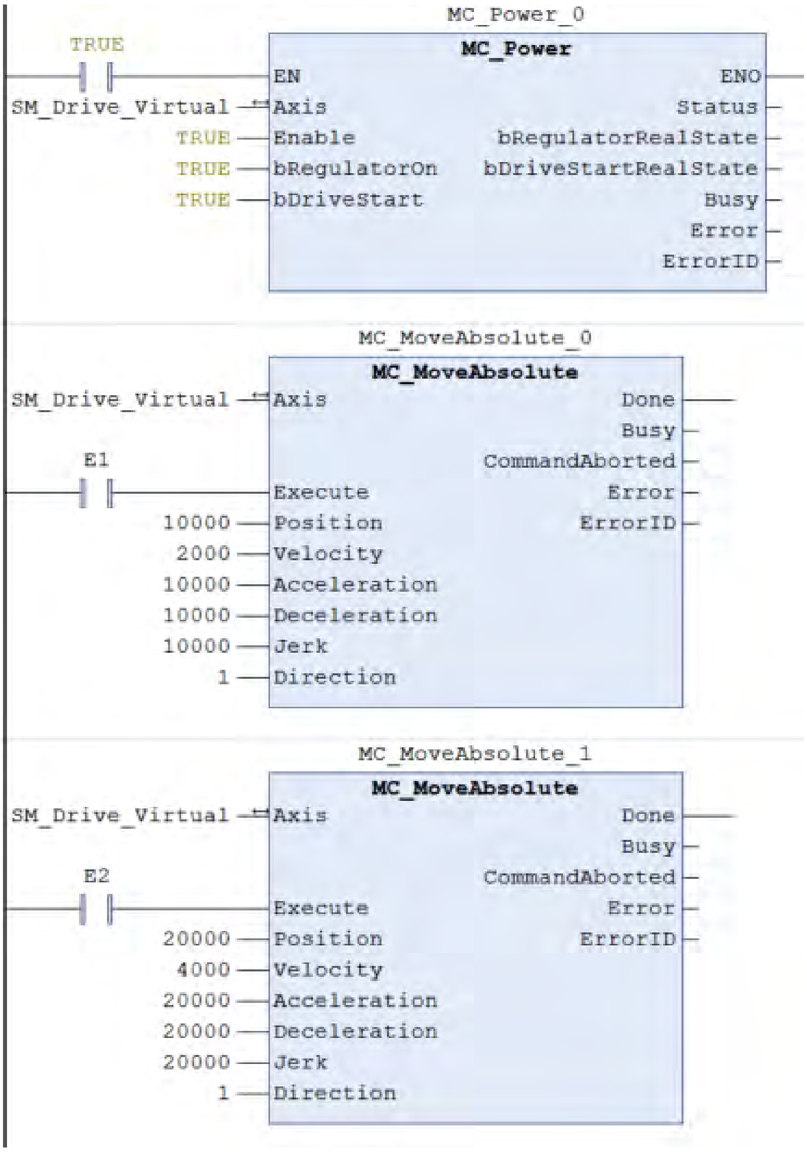
Direction:0(shortest) · 3(fastest) Current position: 315° Target position: 90° Movement angle: 135°	Direction:0(shortest) · 3(fastest) Current position: 315° Target position: 270° Movement angle: 45°
	
Direction:2(current) Rotary axis status: Moving in the negative direction before the function block is executed. Current position: 315° Target position: 90° Movement angle: 225°	Direction:2(current) Rotary axis status: motionless or moving in the positive direction before the function block is executed. Current position: 315° Target position: 90° Movement angle: 135°
	

- **Troubleshooting**

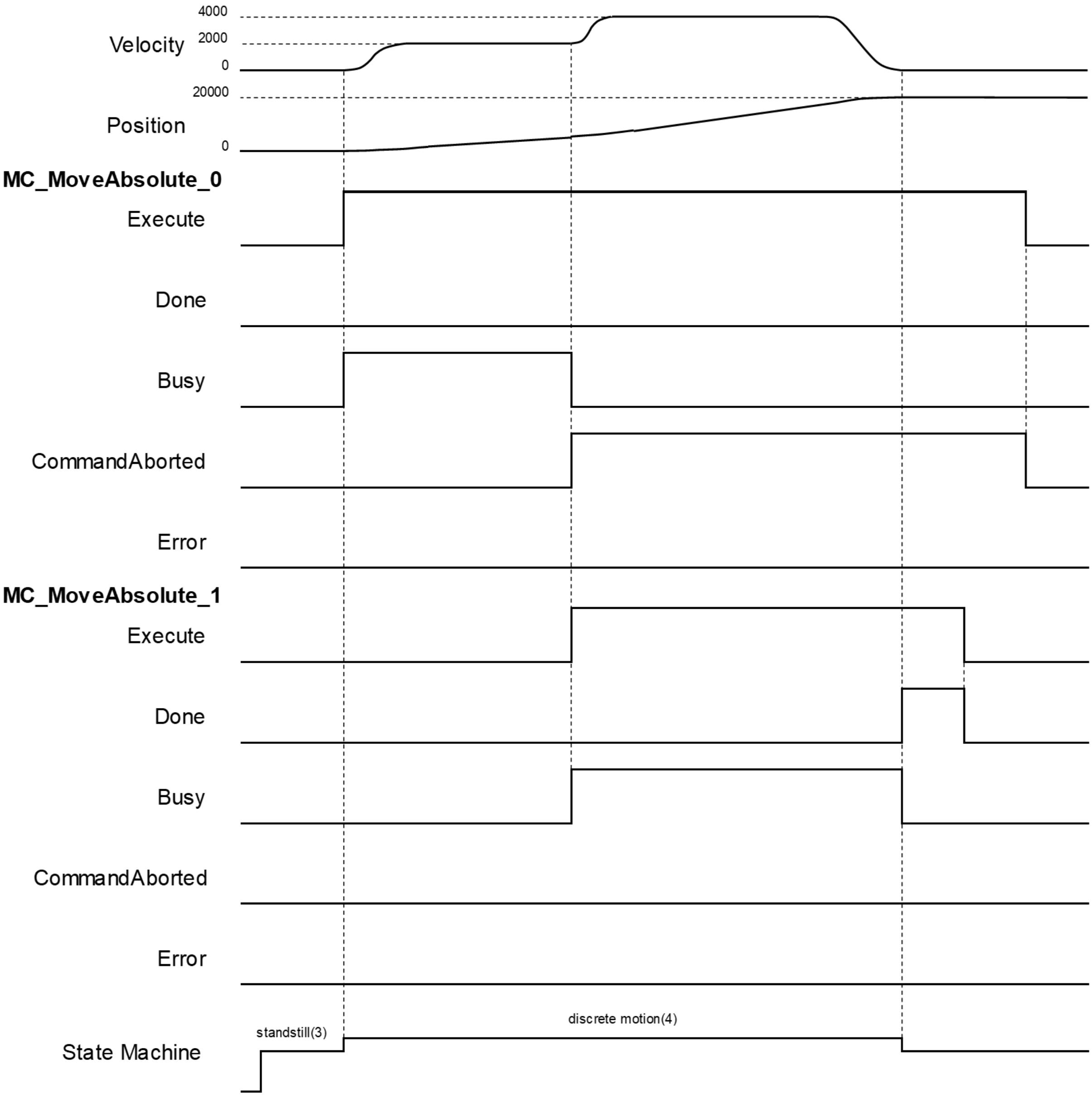
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

  ■ The example below describes the behavior of 2 MC_MoveAbsolute instructions which are connected with each other.



- **Timing Diagram**



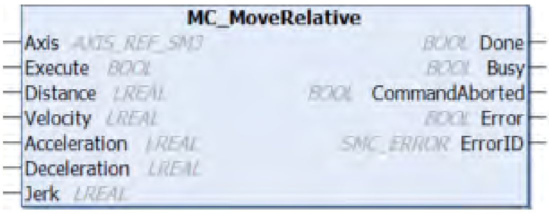
- If Execute is True when MC_MoveAbsolute_0 block is running, the axis will move towards the target position. Once the Execute input of MC_MoveAbsolute_1 changes to True, the execution of MC_MoveAbsolute_0 block would be aborted, which CommandAborted turns True. The final position will be 20,000.
- When MC_MoveAbsolute_1 block is executed, the axis would move towards the absolute target position according to MC_MoveAbsolute_1 parameters.
- When axis reaches the absolue position 20000 set by MC_MoveAbsolute_1, the Done input of MC_MoveAbsolute_1 would turn True as Busy changing to False.
- In case Execute of MC_MoveAbsolute_1 switches to False, the Done output would also change to False state.

- **Supported Products**
  - AX-308E、AX-8

2.1.5 MC_MoveRelative

MC_MoveRelative controls the axis to move a specified relative distance with a specified behavior.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_MoveRelative	 MC_MoveRelative
		Axis AXIS_REF_SM3
		Execute BOOL
		Distance LREAL
		Velocity LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_MoveRelative_instance(
			Axis :=,
			Execute :=,
			Distance :=,
			Velocity :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
Distance	Relative distance to be moved. (Unit: user unit)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
Velocity	Target velocity. (Unit: user unit/s)	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.
Acceleration	Acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	Jerk value. (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.

- **Outputs**

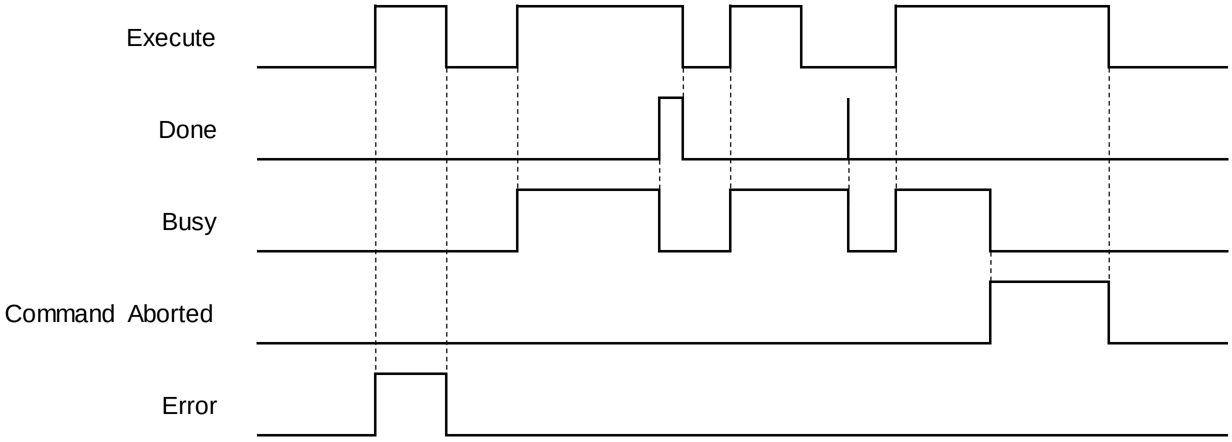
Name	Function	Data Type	Output Range (Default Value)
Done	True when relative distance is completed.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration (Enum)

■ Output Updating Time

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the relative positioning is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done changes to True. ● When Error changes to True. ● When CommandAborted shifts to True.
CommandAborted	● When this instruction is aborted by another function block. When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

■ Timing Diagram



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

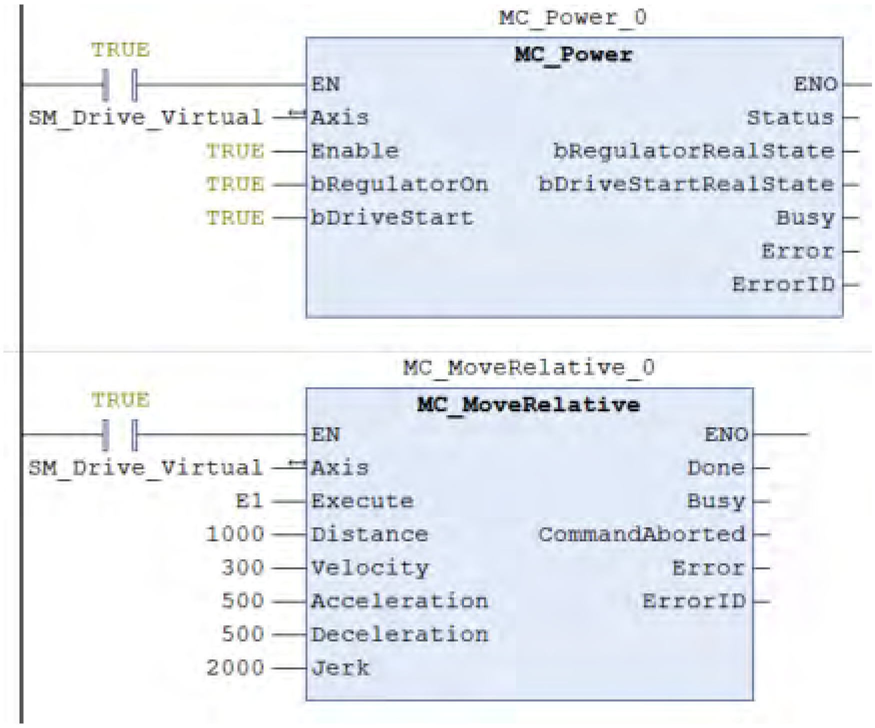
The instruction performs relative positioning with specified target velocity (Velocity), acceleration rate (Acceleration), deceleration rate (Deceleration) and Jerk value (Jerk) when execute changes to True.

- **Troubleshooting**

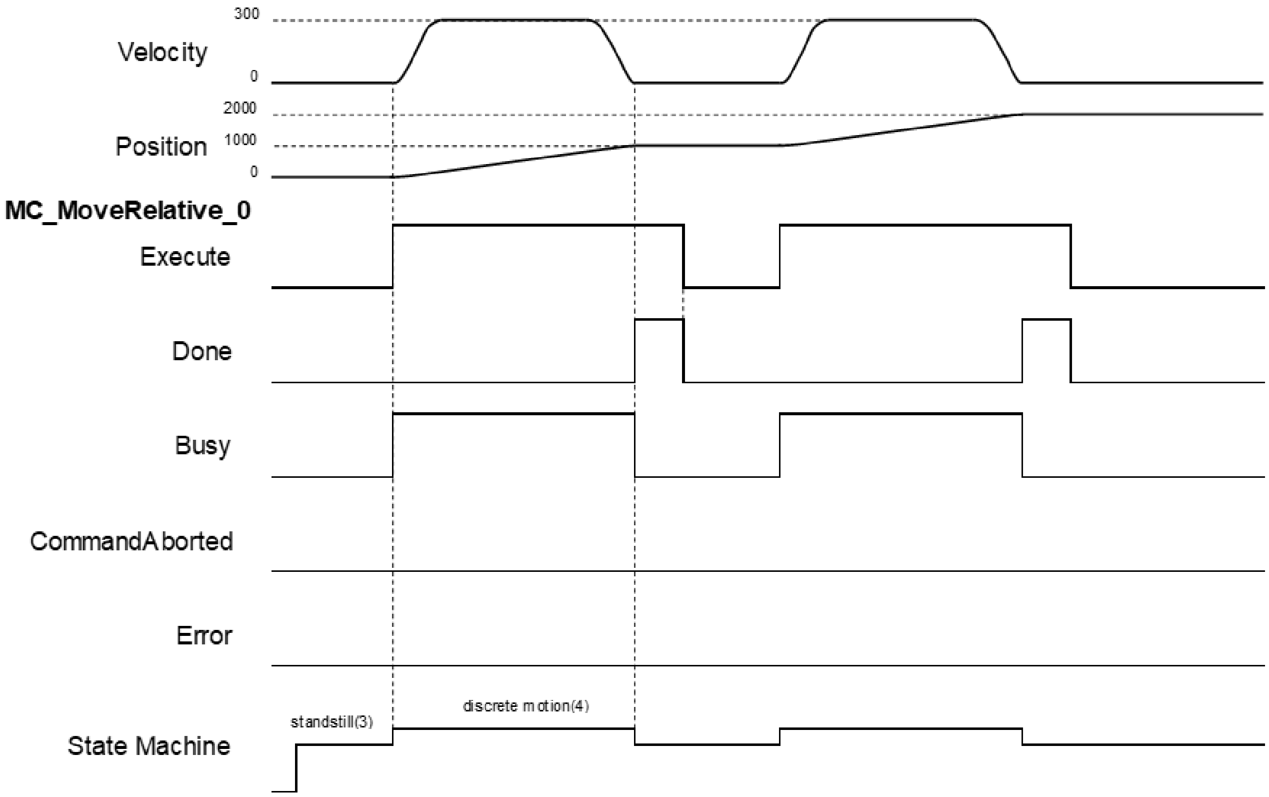
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

  - The example below describes the behavior of the MC_MoveRelative instruction.



- **Timing Diagram**



- ◆ When Execute changes to True, MC_MoveRelative drives the axis to the target position.During movement, Busy is True in the state of Discrete motion.
- ◆ When the axis moved the specified relative distance (1,000), Done changes to True, and Busy changes to False.
- ◆ When Execute changes to False, Done changes to False too.
- ◆ When Execute changes to True again, the instruction will be executed again to drive the axis to the target position and reaching the position of 2,000.

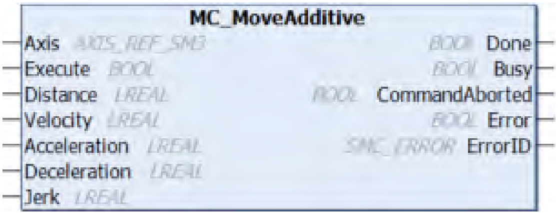
- **Supported Products**

- ■ AX-308E、AX-8

2.1.6 MC_MoveAdditive

MC_MoveAdditive controls the axis to move an additional distance at a given speed and acceleration.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_MoveAdditive	 MC_MoveAdditive
		Axis AXIS_REF_SM3
		Execute BOOL
		Distance LREAL
		Velocity LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_MoveAdditive_instance (
			Axis :=,
			Execute :=,
			Distance :=,
			Velocity :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
Distance	Relative distance to be moved. (Unit: user unit)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
Velocity	Target velocity. (Unit: user unit/s)	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.
Acceleration	Acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	Jerk value. (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.

- **Outputs**

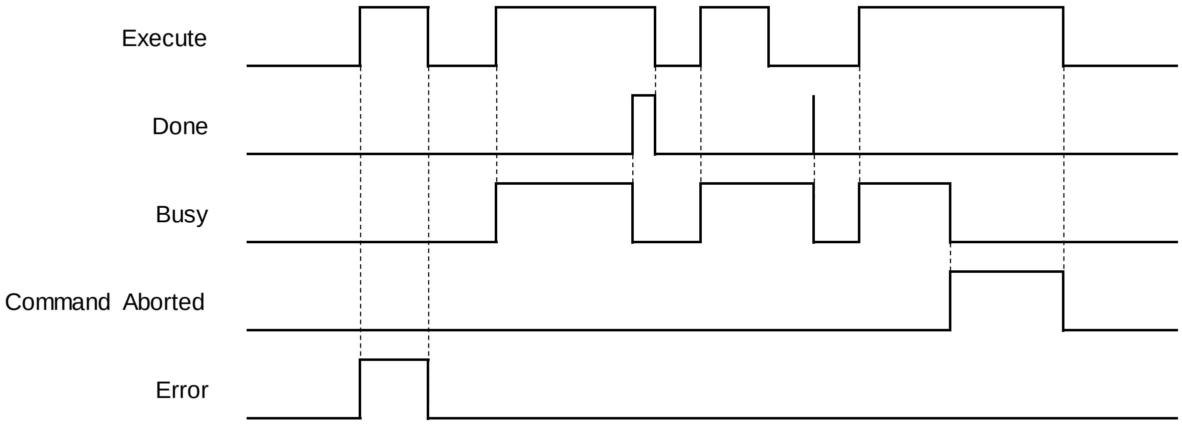
Name	Function	Data Type	Output Range (Default Value)
Done	True when additive distance is completed.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Timing**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● True when the additive positioning is completed.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Busy	● True when Execute changes to True.	● When Done changes to True. ● When Error changes to True. ● When CommandAborted changes to ● True.
CommandAborted	● When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

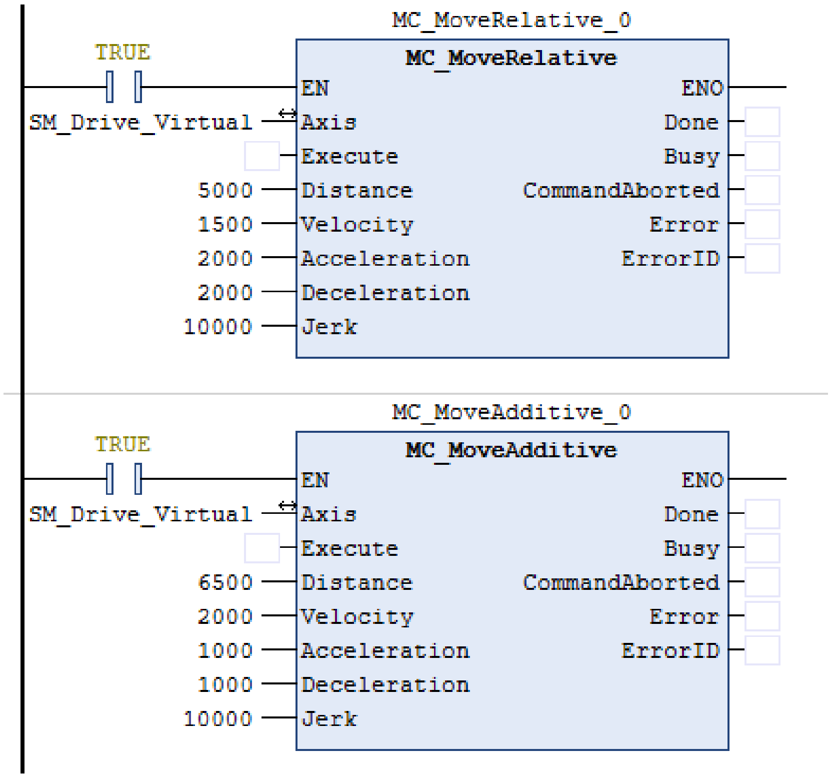
- **Function**

  - MC_MoveAdditive executes the instruction based on user-defined parameters to drive the specific axis to move an additional distance.
  - When MC_MoveAdditive executes alone, the behavior would be identical to a MC_MoveRelative.
  - In case the previous instruction is on-going, an additional distance will be added again for the re-execution of MC_MoveAdditive instruction.

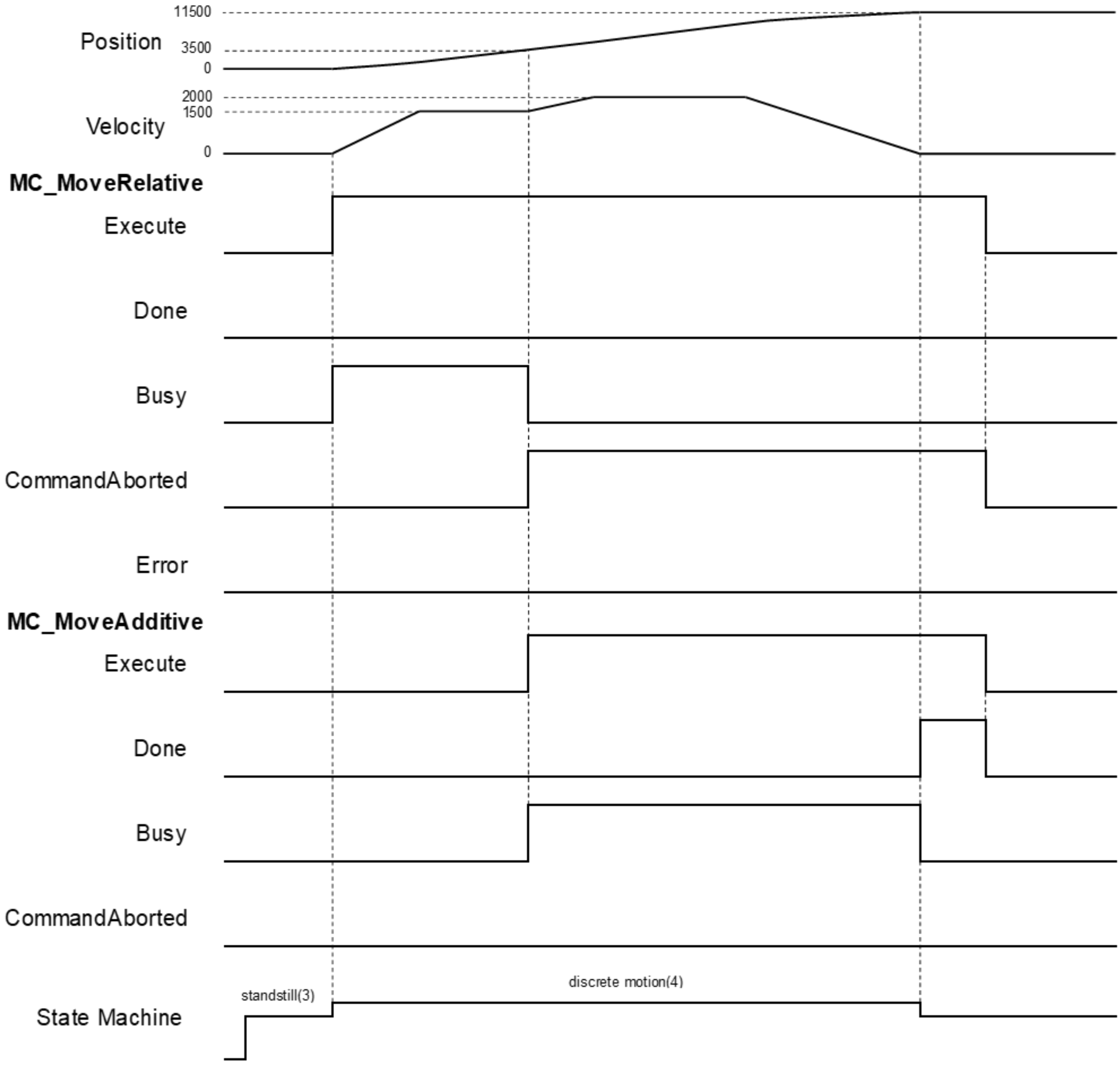
- **Troubleshooting**

  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below describes the behavior of MC_MoveRelative and MoveAdditive instructions which are executed in a series.



- **Timing Diagram**



- ◆ When Execute changes to True, MC_MoveRelative drives the axis to the target position. After Execute changes to True at the position 3500, the MC_MoveRelative instruction would be aborted and CommandAborted changes to True. At the same time, the axis remains in Discrete motion state.
- ◆ Meanwhile, the MC_MoveAdditive instruction is executed and adds a relative distance of 6,500 to the previous commanded position 5,000, and results the new commanded position 11,500.
- ◆ When the axis reaches 13,500, Done changes to True.

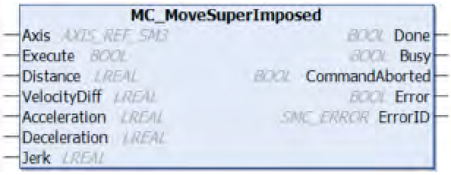
- **Supported Products**

  - ■ AX-308E、AX-8

2.1.7 MC_MoveSuperImposed

MC_MoveSuperimposed controls the axis to move a relative superimposed distance at a specified behavior while the axis is moving.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_MoveSuperImposed	 MC_MoveSuperImposed
		Axis AXIS_REF_SM3
		Execute BOOL
		Distance LREAL
		VelocityDiff LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_MoveSuperImposed _instance (
			Axis :=,
			Execute :=,
			Distance :=,
			VelocityDiff :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
Distance	Additional relative distance to be moved. (Unit: user unit)	LREAL	Negative number, positive number or 0 (0)	When Execute shifts to True and Busy is False.
VelocityDiff	Additional target velocity (Unit: user unit/s)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Acceleration	Additional acceleration rate (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	Additional deceleration rate (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	Additional jerk value (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.

- **Outputs**

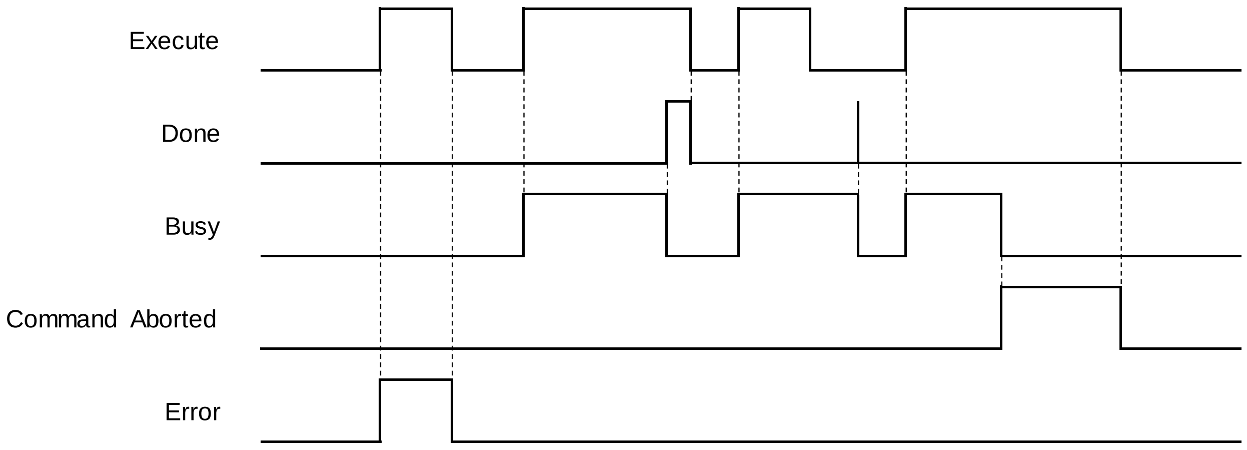
Name	Function	Data Type	Output Range (Default Value)
Done	True when the superimposed movement is completed.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the superimposed distance is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done changes to True. ● When Error changes to True ● When Commandaborted shifts to True
CommandAborted	● When one instruction is aborted by another instruction with the Buffer Mode set to Aborting. ● When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

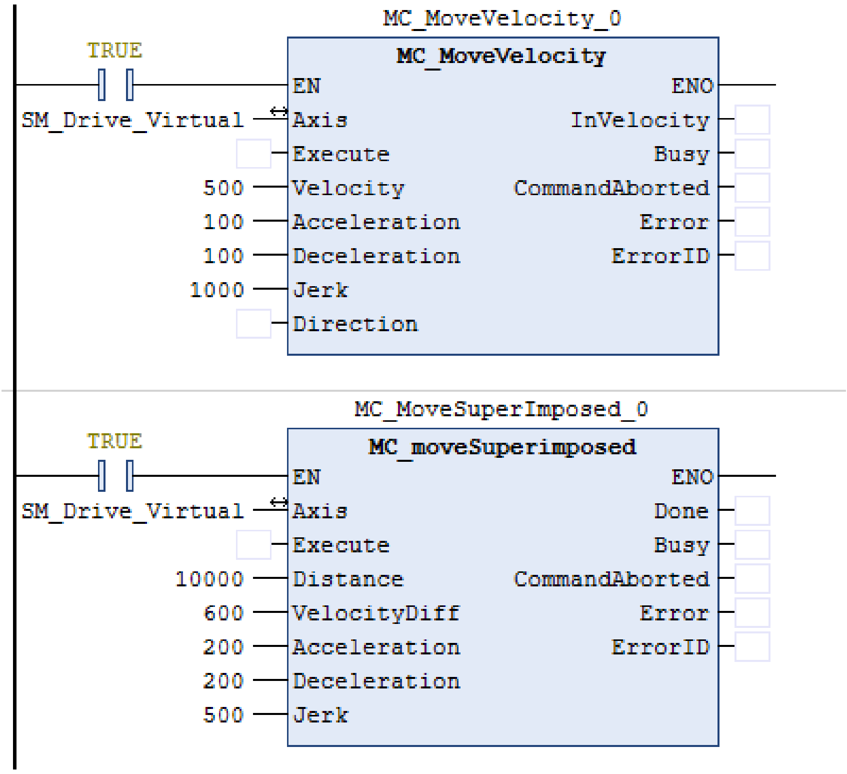
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

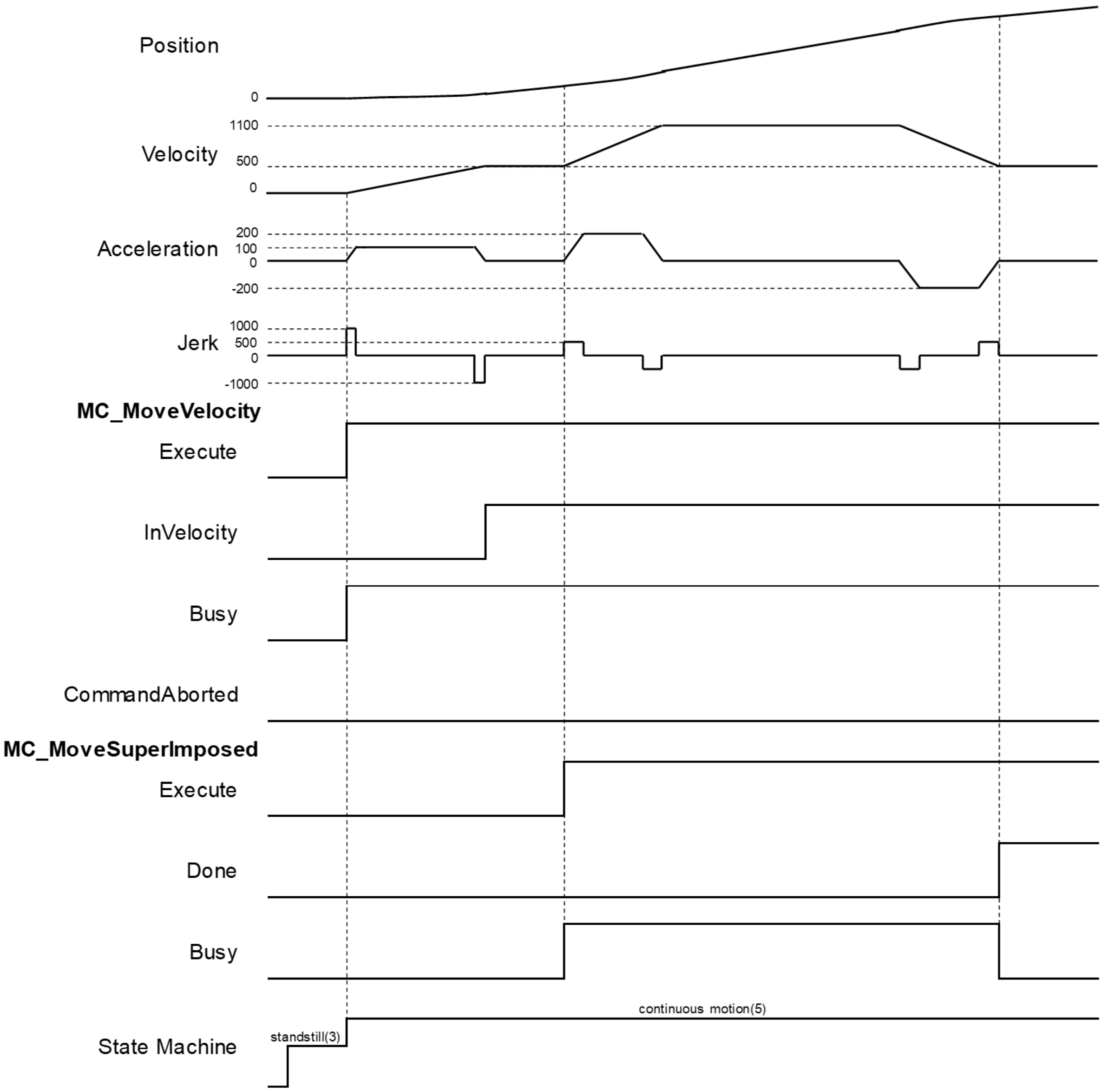
- **Function**
  - The input values of VelocityDiff、Distance、*Acceleration*、*Deceleration* and *Jerk* are superimposed on the on-going motion of the previous instruction.
  - If executing MC_MoveSuperImposed block in Standstill state, the function would be identical to MC_MoveRelative.
  - MC_MoveSuperImposed can be aborted by other function blocks.
  - An error will occur when MC_MoveSuperImposed is repeatedly executed on the same axis.
  - If changing the input values during the execution of MC_MoveSuperImposed or re-execute the function block before the instruction finished, the axis will react according to the new superimposed values and instruction, which are the sum of the previous instruction and MC_MoveSuperimposed instruction. When the superimposed distance is reached, the axis will resume the operation of the previous instruction until the superimposed total distance is reached.
  - MC_MoveSuperimposed and the function block, which is previously executed, would be aborted if a new function block is started while MC_MoveSuperimposed is superimposed on other function blocks.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below describes the behavior of MC_MoveSuperImposed applied to MC_MoveVelocity.



- **Timing Diagram**



- When Execute of MC_MoveVelocity changes to True, the specific axis starts to move towards the target velocity (500) at the constant speed.

- When Execute of MC_MoveSuperImposed changes to True, the MC_MoveSuperImposed instruction starts and applies the additional values(velocity, distance, acceleration, deceleration and jerk) to the axis and the axis performs a superimposed motion path. Since VelocityDiff is set as 600 and the target superimposed distance is far enough, the velocity will be superimposed to 1100(500 + 600).

- When the execution of MC_MoveSuperImposed is finished, Done will turn True and MC_MoveVelocity will keep going.

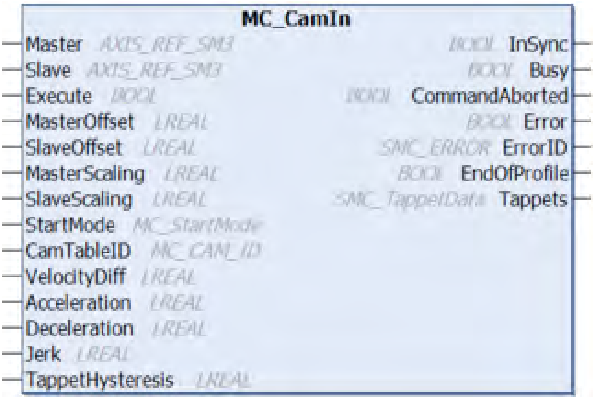
- **Supported Products**

- AX-308E、AX-8

2.1.8 MC_CamIn

MC_CamIn performs cam operation.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_CamIn	 MC_CamIn
		Master AXIS_REF_SM3
		Slave AXIS_REF_SM3
		Execute BOOL
		MasterOffset LREAL
		SlaveOffset LREAL
		MasterScaling LREAL
		SlaveScaling LREAL
		StartMode MC_StartMode
		CamTableID MC_CAM_ID
		VelocityDiff LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		TappetHysteresis LREAL
		BOOL InSync
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
		BOOL EndOfProfile
		SMC_TappetData Tappets
			MC_CamIn_instance (
			Master :=,
			Slave :=,
			Execute :=,
			MasterOffset :=,
			SlaveOffset :=,
			MasterScaling :=,
			SlaveScalling :=,
			StarMode :=,
			CamTableID :=,
			VelocityDiff :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			TappetHysteresis :=,
			InSync =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>,
			EndOfProfile =>,
			Tappets =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
MasterOffset	Shifts the position of the master axis by the specified offset value. (Unit: user unit)	LREAL	Negative number, positive number or 0 (0)	When Execute shifts to True and Busy is False.
SlaveOffset	Shifts the displacement of the slave axis by the specified offset value. (Unit: user unit)	LREAL	Negative number, positive number or 0 (0)	When Execute shifts to True and Busy is False.
MasterScaling	Scales the master axis up and down with the specified factor.	LREAL	Negative number, positive number or 0 (0)	When Execute shifts to True and Busy is False.
SlaveScaling	Scales the slave axis up and down with the specified factor.	LREAL	Negative number, positive number or 0 (0)	When Execute shifts to True and Busy is False.
StartMode	Specifies the engagement behavior of the slave axis	MC_StartMode	0:absolute 1:relative	When Execute shifts to True and Busy is False.

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
			2:ramp_in 3:ramp_in_pos 4:ramp_in_neg (absolute)	
CamTableID	Cam table identifier, which is from output of CamTableSelect.	MC_ CAM_ID	MC_CAM_ID*	When Execute shifts to True and Busy is False.
VelocityDiff	Maximum velocity difference under ramp_in mode. (Unit: user unit/s)	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.
Acceleration	The acceleration rate under ramp_in mode. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	The deceleration rate under ramp_in mode.. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	The jerk value under ramp_in mode. (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
TappetHysteresis	The hysteresis rate of tappet.	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.

***Note:** MC_CAM_ID(Struct): Cam table variables, from output of MC_CAMTableSelect, are input to MC_CamIn.

Name	Function	Data Type	Setting Value (Default Value)
pCT	Internal information stored in the cam table	POINTER TO BYTE	Positive number or 0(0)
Periodic	Periodic mode	BOOL	True/False (True)
MasterAbsolute	MasterAbsolute mode	BOOL	True/False (True)
SlaveAbsolute	SlaveAbsolute mode	BOOL	True/False (True)
StartMaster	The start master axis position in the cam table	LREAL	Negative number, positive number or 0 (0)
EndMaster	The end master axis position in the cam table	LREAL	Negative number, positive number or 0 (0)
StartSlave	The start slave axis position in the cam table	LREAL	Negative number, positive number or 0 (0)
EndSlave	The end slave axis position in the cam table	LREAL	Negative number, positive number or 0(0)

Name	Function	Data Type	Setting Value (Default Value)
byCompatibilityMode	Compatibility mode	BYTE	Positive number or 0(0)

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
InSync	True when the specified master/slave cam operation is synchronized.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when this instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*[1]	SMC_ERROR(SMC_NO_ERROR)
EndOfProfile	True when the end point of the cam profile is completed.	BOOL	True/False(False)
Tappets	Can be used with the function block of SMC_GetTappetValue.	SMC_TappetData*[2]	SMC_TappetData

Note:

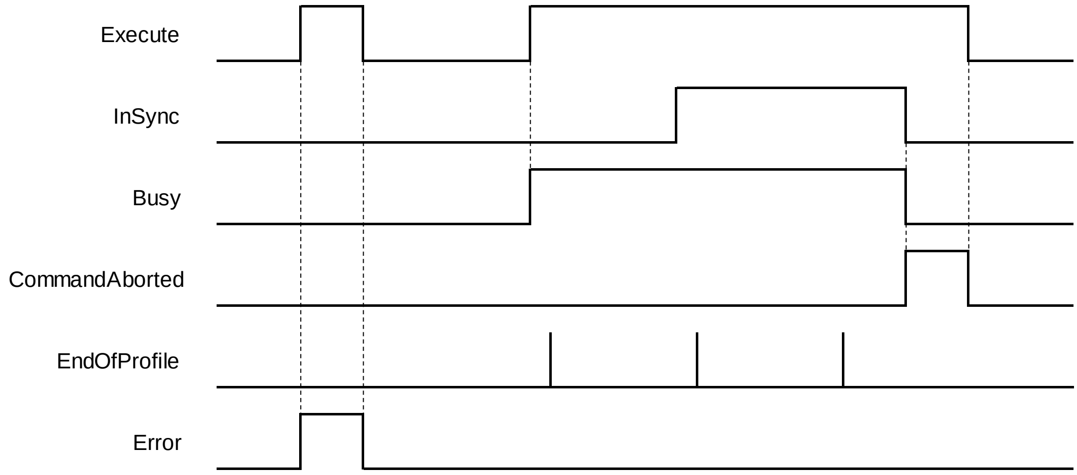
1. SMC_ERROR: Enumeration(Enum)
2. SMC_TappetData: Structure(Struct)

Name	Function	Data Type	Output Range (Default Value)
ctt	Tappet action active when axis passes tappets in the specified direction (positive or negative).	SMC_CAMTAPPETTYPE	0:TAPPET_pos (Pass in positive direction) 1:TAPPET_all (No specific direction) 2:TAPPET_neg (Pass in negative direction) (TAPPET_pos)
cta	The action activated when axis passes tappets.	SMC_CAMTAPPETACTION	0:TAPPETACTION_on(Switch ON) 1:TAPPETACTION_off(Switch OFF) 2:TAPPETACTION_inv (Inverts) 3:TAPPETACTION_time (Switches on after a delay for a certain time period.) (TAPPETACTION_on)
dwDelay	Specify the delay time for switching ON under TAPPETACTION_time mode.	DWORD	Positive number or 0(0)
dwDuration	Specify the time duration for which the tappet is switched to ON under TAPPETACTION_time mode.	DWORD	Positive number or 0(0)
iGroupID	Track ID of tappets	INT	Positive number, negative number or 0(0)
x	Master position where tappet is switched.	LREAL	Positive number, negative number or 0(0)
dwActive	Internal variable	DWORD	Positive number or 0(0)

Output Updating Time

Name	Timimg for Shifting to True	Timimg for Shifting to False
InSync	● When the synchronization between master and slave axis is completed.	● When a falling edge is detected at input *Execute*. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When an instruction is being executed.	● When a rising edge is detected at output CommandAborted. ● When a rising edge is detected at output *Error*.
CommandAborted	● When MC_CamOut is executed. ● When one instruction is aborted by another instruction. ● When a function block instruction is aborted by MC_Stop.	● When a falling edge is detected at input *Execute*. ● If Execute is False and *CommandAborted* shifts to True, *CommandAborted* will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When a falling edge is detected at input Execute. (Error codes are cleared.)
ErrorID		
EndOfProfile	● Cyclic end of the cam profile	● Shift to True for only one period and immediately shift to False if MC_CamTableSelect Periodic is 1(period). ● Shift to False when a falling edge is detected at input Execute and MC_CamTableSelect Periodic is 0.

Timing Diagram



- ◆ When *Execute* shifts from FALSE to TRUE and Busy is TRUE, *InSync* shifts from False to True as soon as the synchronization between master and slave axis is completed. When coming to the end of CAM cycle, *EndOfProfile* shifts from FALSE to TRUE for only one period, then switch back to FALSE. Once the coupling of master and slave axis is deactivated, such as executing MC_CamOut, *CommandAborted shifts from* FALSE to TRUE, while both *InSync* and *Busy* shifts from TRUE to FALSE. Then, *CommandAborted* will shift from TRUE to FALSE as well as *Execute*.

- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Master	Reference to the master axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
Slave	Reference to the slave axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

  - **Relationship between master axis position and slave axis position**

    - The cam relationship which is planned in the software is the position relationship between the master axis and slave axis. The ”position” mentioned here is the cam phase of the master axis / slave axis instead of the actual axis position. If the cam relationship which is planned is seen as the function CAM as below, the input of the function CAM is the master axis cam phase and the output is the slave axis cam phase. The formula is shown as below.

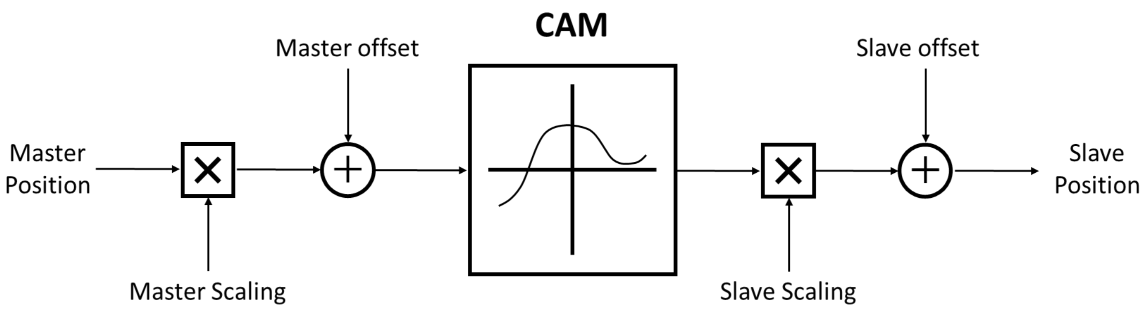
      y = CAM(x)

      x:The master axis cam phase

      y:The slave axis cam phase

    - The cam phase comes from the axis positions and there is a conversion between them. The conversion between the axis position and cam phase is related with the *MasterAbsolute*, *SlaveAbsolute*, *MasterOffset*, *SlaveOffset*, *MasterScaling*, and *SlaveScaling*.

    - The slave axis follows the master axis to make the synchronous cam motion by using the MC_CamIn instruction. In the synchronous cam motion, the corresponding relationship between the master axis position and slave axis position is based on the pre-planned cam relationship (the cam curve or cam table). The process in which the slave axis position is calculated through the master axis position is illustrated as follows.



- The following formula is generated from the figure above.:

Position_Slave=SlaveScaling × CAM(MasterScaling × MasterPosition+MasterOffset)+ SlaveOffset

It can be seen that when master axis is in absolute mode, master position is the remainder of the current master position divided by modulo; When master axis is in relative mode, master position is the start point position (usually 0) of master axis in the corresponding cam curve.

- **Relation between StartMode and MasterAbsolute/ SlaveAbsolute of CamTableSelect**

  - Absolute mode (StartMode = 0): The slave current position is not involved in the Cam calculation as the cam synchronization starts, However, a jump can be caused if the current position of slave axis and its start position from the Cam are not the same

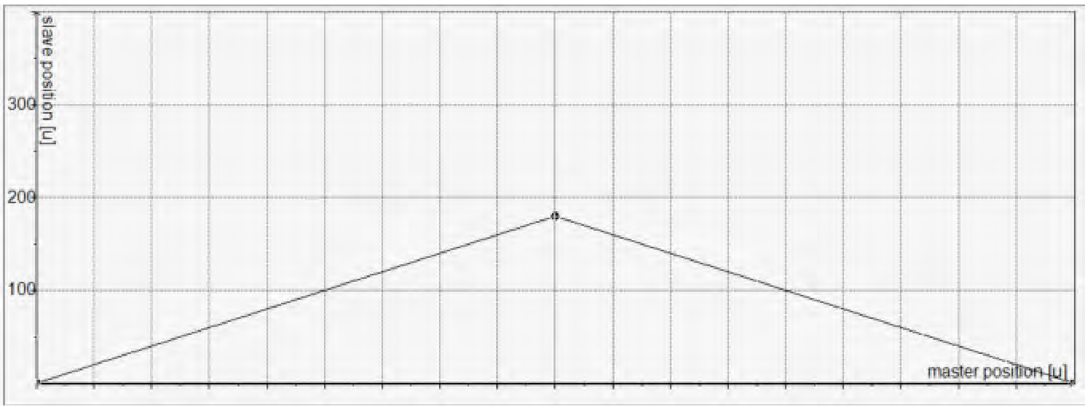
  - Relative mode (StartMode = 1): Cam changes according to the current position of slave axis, which the slave axis position would be added to the slave current position for calcution. If the slave setpoint position according to the cam is not at the start point 0, a jump may occur.

  - Ramp mode (StartMode = 2, 3, 4): Add a motion curve for compensation according to *VelocityDif, Acceleration, Deceleration, and Jerk,* so as to prevent a jump in cam when coupling starts.

MC_CamTableSelect.MasterAbsolute	Master axis
absolute	Absolute mode
relative	Relative mode

MC_CamIn.StartMode	MC_CamTableSelect.SlaveAbsolute	Slave axis
absolute	True	Absolute mode
absolute	False	Relative mode
relative	True	Relative mode
relative	False	Relative mode
ramp_in	True	Ramp in Absolute mode
ramp_in	False	Ramp in Relative mode
ramp_in_pos	True	Ramp in positive Absolute mode
ramp_in_pos	False	Ramp in positive Relative mode
ramp_in_neg	True	Ramp in negative Absolute mode
ramp_in_neg	False	Ramp in negative Relative mode

- Cam table

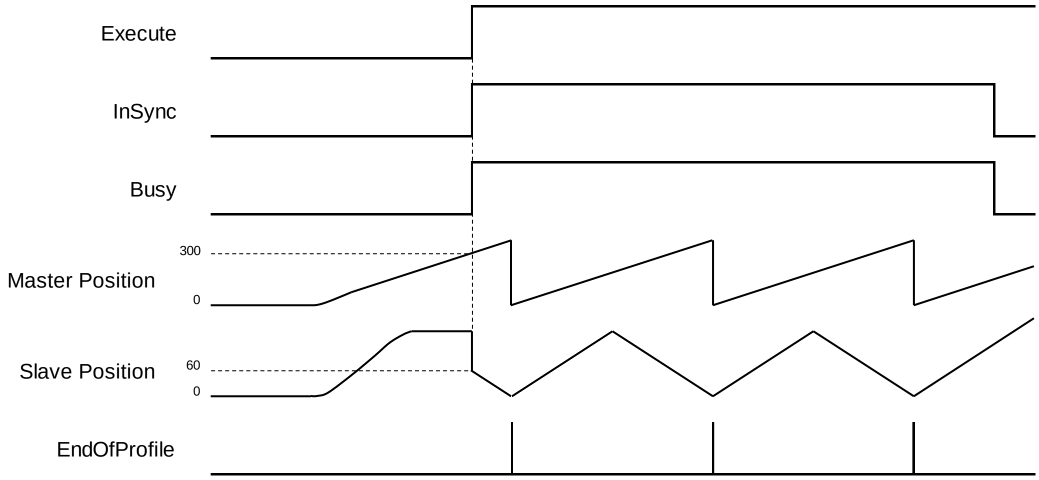


1. Absolute mode on master axis(MasterAbsolute = true)

   Absolute mode on slave axis (SlaveAbsolute = true)

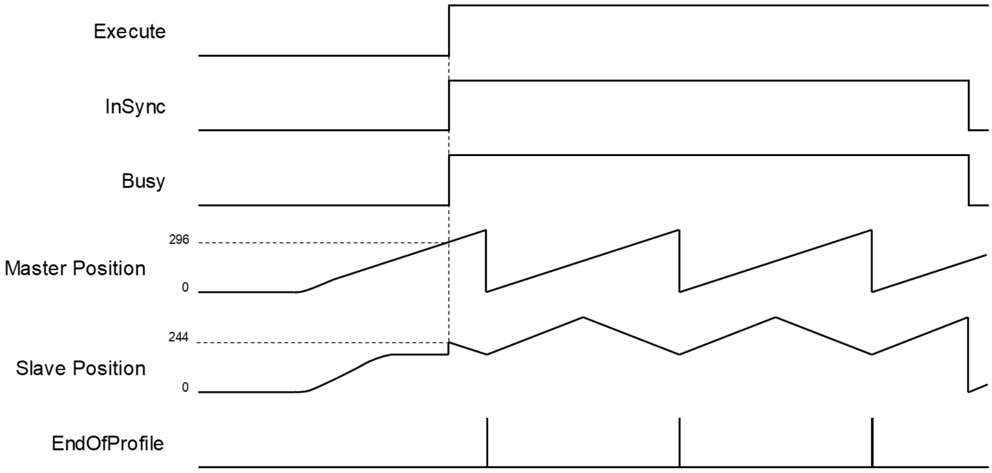
1.1 Absolute mode(StartMode = 0)

Coupling positions for master and slave axis, respectively, are master current position and the slave position from the cam table.



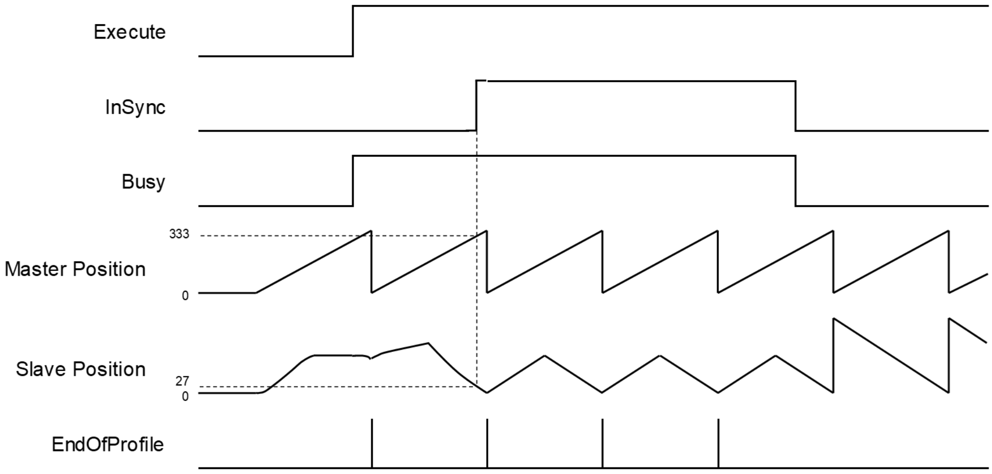
1.2 Absolute mode (StartMode = 1)

Coupling positions for master and slave axis, respectively, are master current position and the slave position on the cam table plus slave current position (180+64 = 244). In addition, a jump would occur if the start point of master axis is not same as the start position on cam table.



1.3 Ramp in mode (StartMode=2)

Coupling positions for master and slave axis, respectively, are master current position and the slave position added with a motion curve for compensation, which is configured via VelocituDiff, Acceleration and Deceleration settings, for the purpose of preventing a jump while ramping in.



1.4 Ramp in positive, Ramp in negative (StartMode = 3, 4)

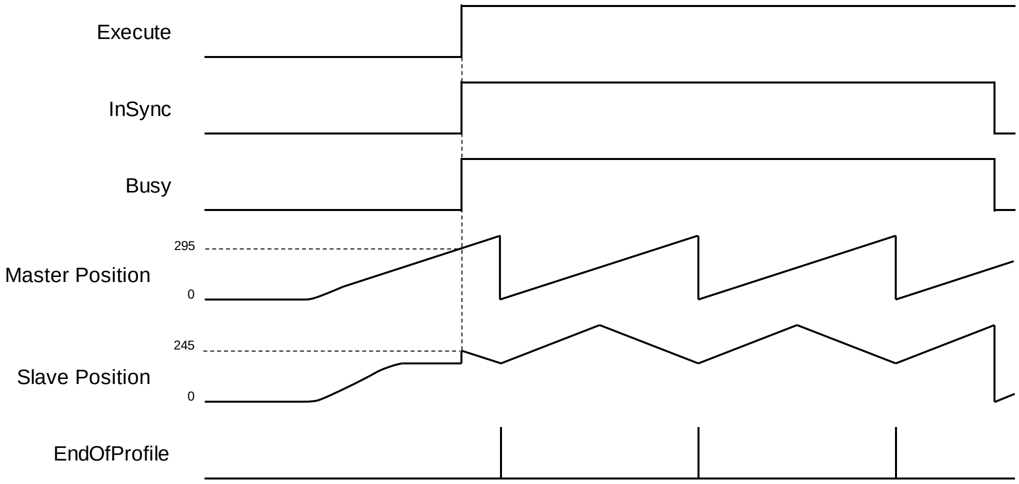
With a rotary/ modulo slave axis, ramp_in_pos compensates only in the positive direction and ramp_in_neg in the negative direction. For linear slaves the compensation direction is generated automatically with ramp_in_pos, ramp_in_neg, and ramp_in mode which also means these three modes are under the same execution condition.

2. Absolute mode on master axis (MasterAbsolute = true),

   Relative mode on slave axis (SlaveAbsolute = false)

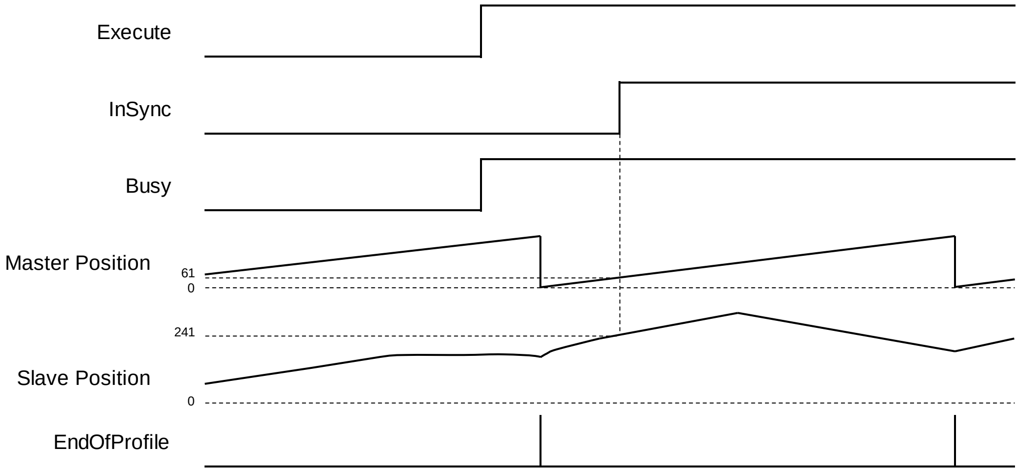
2.1 Absolute/ Relative mode (StartMode = 0, 1)

The defined positions of master and slave axis when cam is engaged, respectively, are master current position and the slave position from the cam table added on the slave current position (180+65 = 245). In addition, a jump would occur if the start point of master axis is not same as the start position on cam table



2.2 Ramp in mode (StartMode = 2)

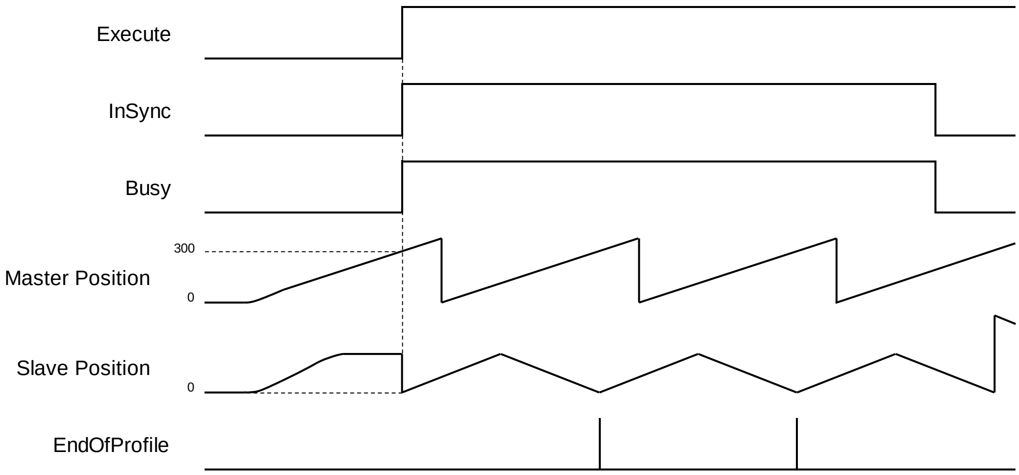
The defined positions of master and slave axis when cam is engaged, respectively, are master current position and the slave position added with a motion curve for compensation, which is configured via VelocituDiff, Acceleration and Deceleration settings, for the purpose of preventing a jump while ramping in. The slave coupling position would be the position on the cam table plus slave current position (61 + 180 = 241).



3. Master absolute mode (MasterAbsolute = false)/ Slave absolute mode (SlaveAbsolute = true)

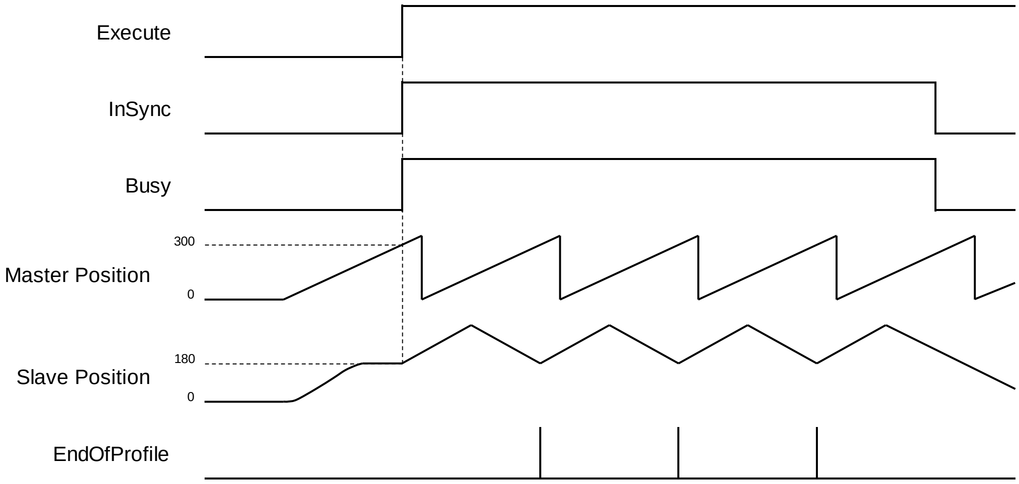
3.1 Absolute mode (StartMode = 0)

Master-slave coupling position: The current position of master axis would be the start position as well as the zero position on the cam table. The corresponding position of the slave axis on the cam table should also be zero, while the slave position (coupling) is zero under absolute mode.



3.2 Relative mode (StartMode = 1)

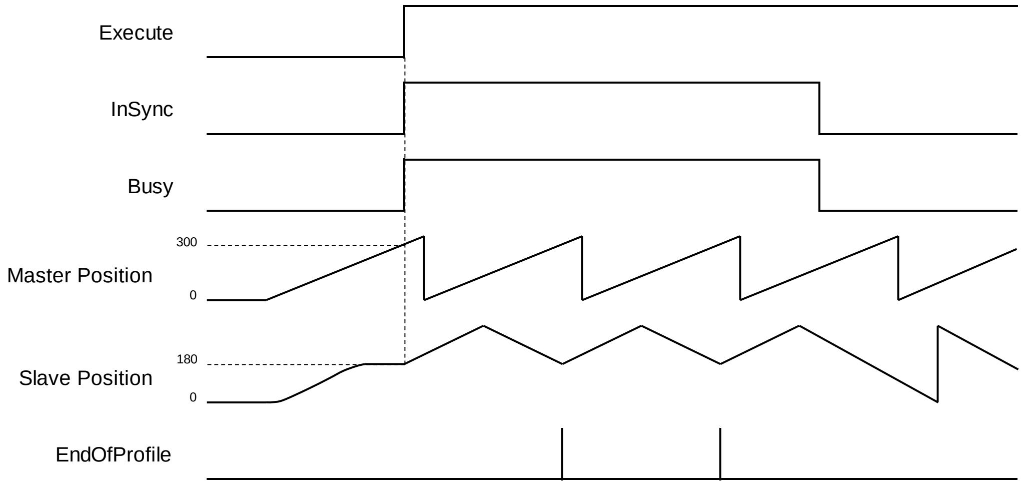
Master-slave coupling position: The current position of master axis would be the start position as well as the zero position on the cam table. The corresponding position of the slave axis on the cam table should also be zero, while the slave position (coupling) under relative mode should be the sum of corresponding position and slave current position (0+180=180).



4. Master relative mode (MasterAbsolute = false)/ Slave relative mode (SlaveAbsolute = false)

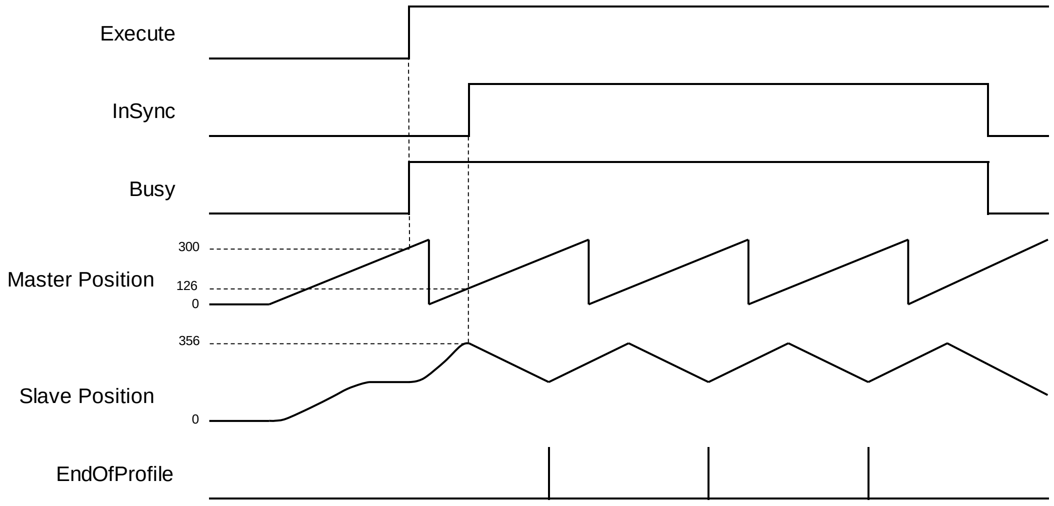
4.1 Abosolute/ Relative mode (StartMode = 0, 1)

Master-slave coupling position: The current position of master axis would be the start position as well as the zero position on the cam table. The corresponding position of the slave axis on the cam table should also be zero, while the slave position (coupling) under relative mode should be the sum of corresponding position and slave current position (0+180=180).



4.2 Ramp in mode (StartMode = 2)

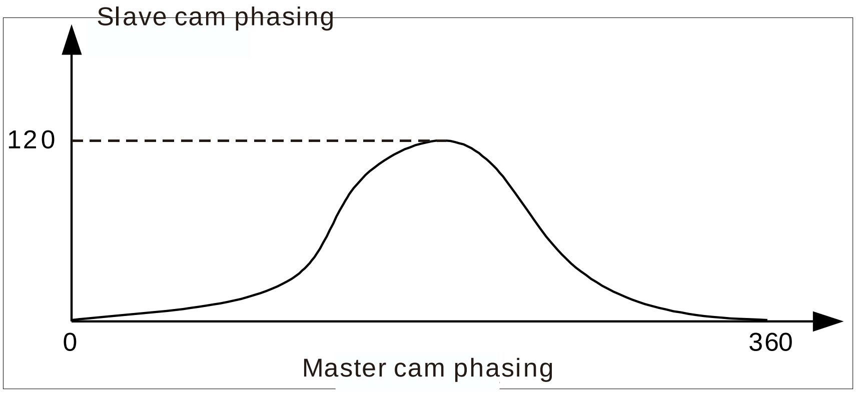
Master-slave coupling position: The current position of master axis would be the start position as well as the zero position on the cam table. A compensating curve is added to the slave position according to the settings of VelocityDiff, Acceleration and Deceleration to avoid jumps.



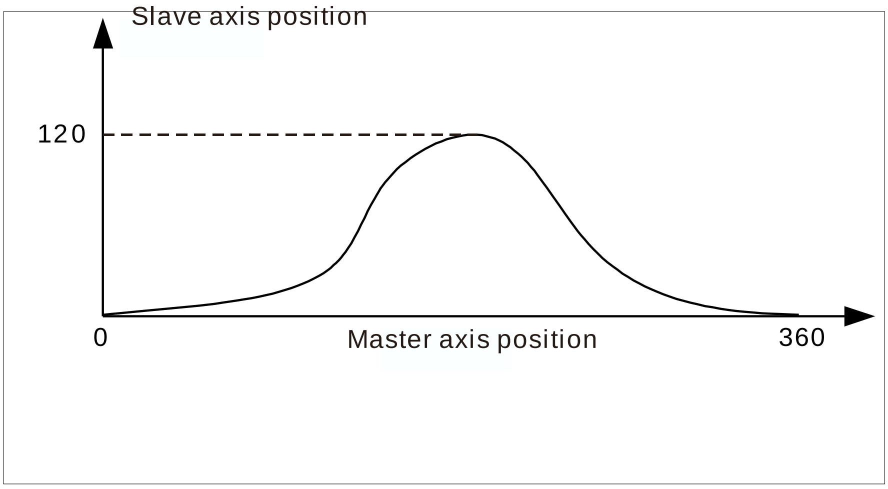
- **Offsets and scaling (MasterOffset/MasterScaling/SlaveOffset/Slavescaling)**

  - Position offsets and scaling can be performed by modifying the parameters based on the preplanned cam curve. For example, you can specify the scaling factor to adjust phase and offset between master and slave in cam table with only one cam curve needed for a processed product, which comes in multiple sizes, so as to switch between different sizes of the product during production. In addition, offsets and scaling factors of master and slave axis can be configured respectively.

  - Offsets and scaling between master and slave axis determine the actual operation for cam profile, which is demonstrated in the following example. The preplanned cam profile curve is shown below.

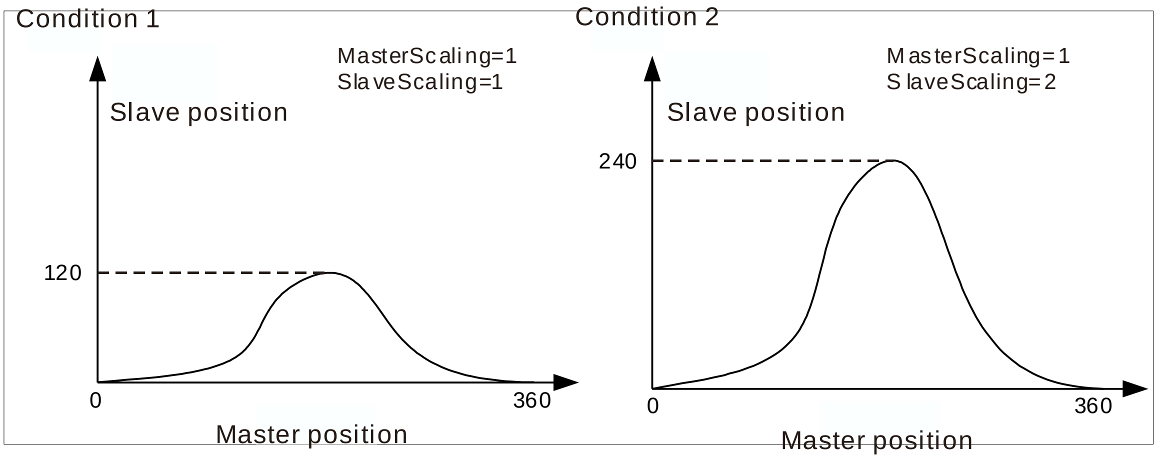


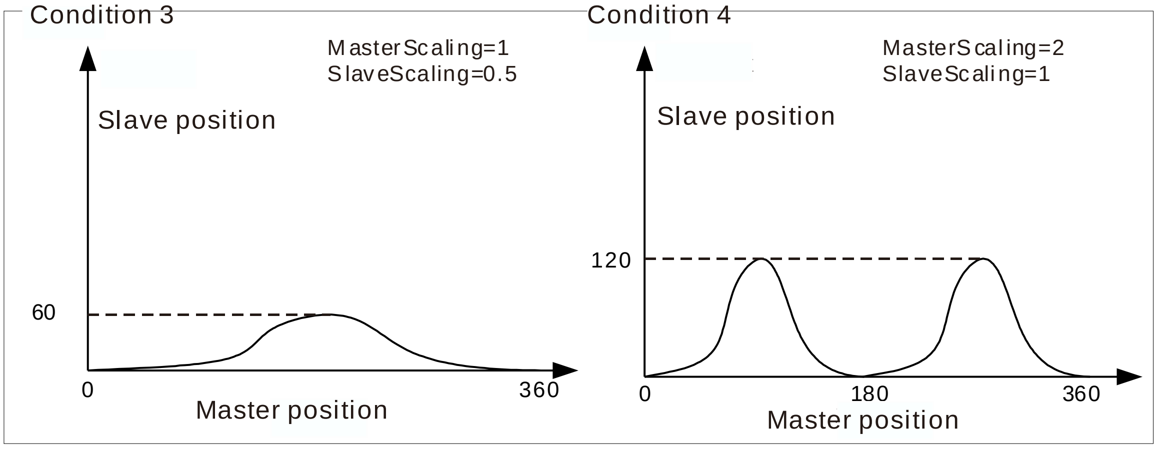
  - If the master and slave axis are under absolute mode, the start position of master and slave axis would be zero while performing coupling action. Without any offset and scaling (default setting), the relationship between the actual positions of master and slave axis are shown below.

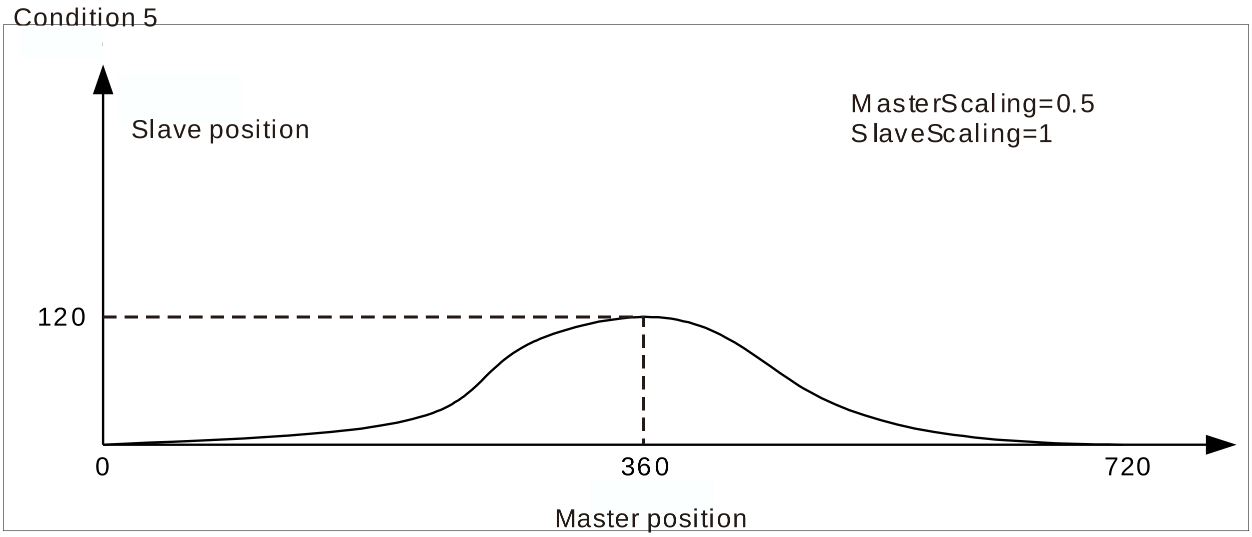


- When the position offsets and scaling are not set as default, impacts on the relationship between the actual positions of master and slave axis are shown below.

1. When MasteOffset= 0, the impact of MasterScaling and SlaveScaling on the actual cam profile.







Condition 1: When MasterScaling and SlaveScaling are set to 1 with no offsets, the actual cam profile would be same as preplanned.

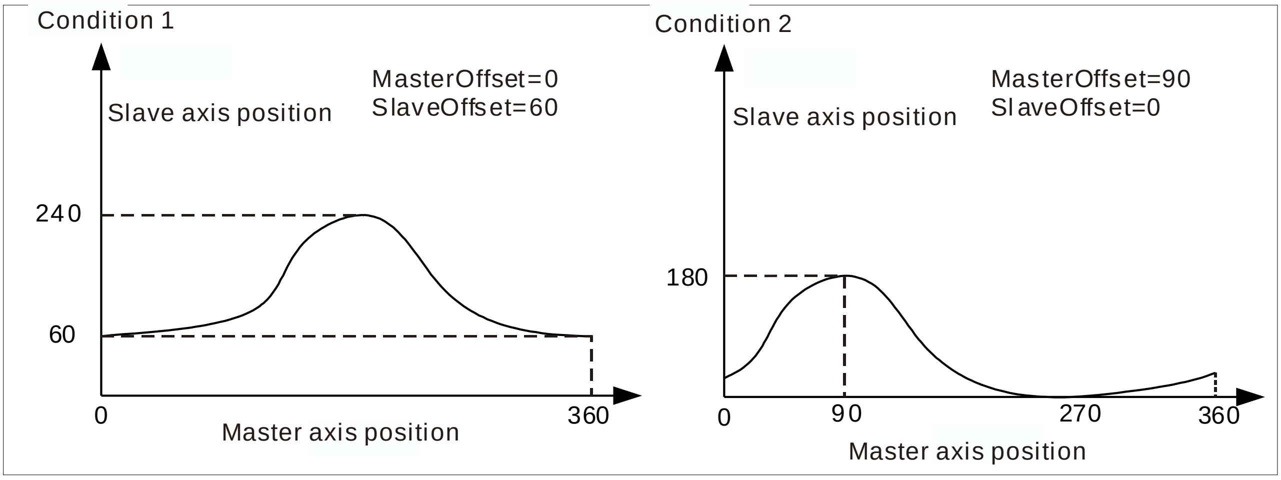
Condition 2: When MasterScaling=1 and SlaveScaling=2 with no offsets, the slave position would be two times more than the preplanned.

Condition 3: When MasterScaling=1 and SlaveScaling=0.5 with no offsets, the slave position would be half of the preplanned.

Condition 4: When MasterScaling=2 and SlaveScaling=1 with no offsets, the position of master axis is doubled compared with the preplanned position relative to the slave axis. From the angle of cam phasing, the master phasing is twice the pre-planned phasing, which the cam master cycle changes from 360 to 180 and the cam slave phasing remains unchanged.

Condition 5: When MasterScaling=0.5 and SlaveScaling=1 with no offsets, the position of master axis would be half of the preplanned position relative to the slave axis. From the angle of cam phasing, the master phasing is half the pre-planned phasing, which the cam master cycle changes from 360 to 720 and the cam slave phasing remains unchanged.

2. When MasteScaling= 0, the impact of MasterOffset and SlaveOffset on the actual operation of cam profile.



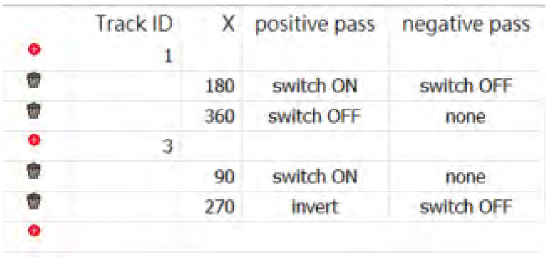
Condition 1: When MasterScaling=1, SlaveScaling=1, MasterOffset=0 and SlaveOffset=60, the slave position relative to the master position would be added with 60 based on the preplanned position. For example, the master position 180 corresponds to the slave position 180 in a planned cam relationship which the corresponding slave axis position should be 240(240=180+60) during the actual execution.

Condition 2: When MasterScaling=1, SlaveScaling=1, MasterOffset=90 and SlaveOffset=0, the master position relative to the slave position would be added with 90 based on the preplanned position. For example, the master position 180 corresponds to the slave position 180 in a planned cam relationship, which the master axis position 90 should correspond to the slave axis position 180(180=90+90) during the actual execution.

- **Period mode**
  - ◆ Use Periodic of MC_CamTableSelect to control the period mode. Under non-periodic mode, EndOfProfile remains to be TRUE after executing for one period. Meanwhile, the slave axis stops moving, but still in sync. The status of slave axis stays under synchronized_motion.
  - ◆ At the same time, Execute changes to False, while OutputsInSync, Busy and EndOfProfile of MC_CamIn remains to be TRUE.

- **Tappet table***

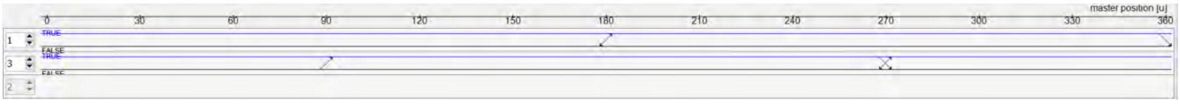
  - Use Tappet table to set tappets in Cam and read the status of tappets with SMC_GetTappetValue, which can also be modified according to the settings in Tappet table and the direction when CAM master passing the tappets.



	Track ID	X	positive pass	negative pass
●	1			
		180	switch ON	switch OFF
		360	switch OFF	none
●	3			
		90	switch ON	none
		270	invert	switch OFF
●				

***Note:** At the same position, a maximum of three tappets can be set on the Tappet table.

  - You can configure several tappets for each track ID on the Tappet table, then view the relationship between tappets and the master axis. While moving the points on Tappets page, the setting parameters on Tappet table page would be changed simultaneously.



- **TappetHysteresis**

  - Set hysteresis intervals of tappet to avoid vibration in axes and encoders, which may lead to wrong switching actions. The specified axis position must exceeds the interval so the next action will be executed. The unit for Hysteresis is user-defined.

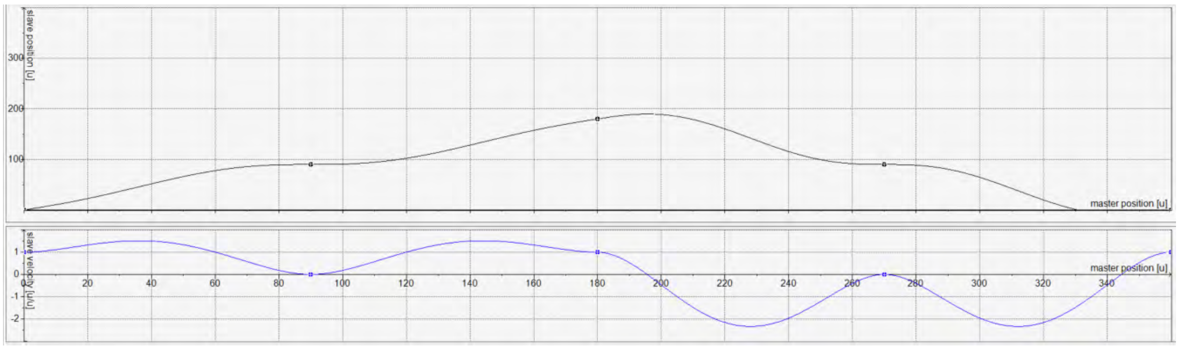
- **Troubleshooting**

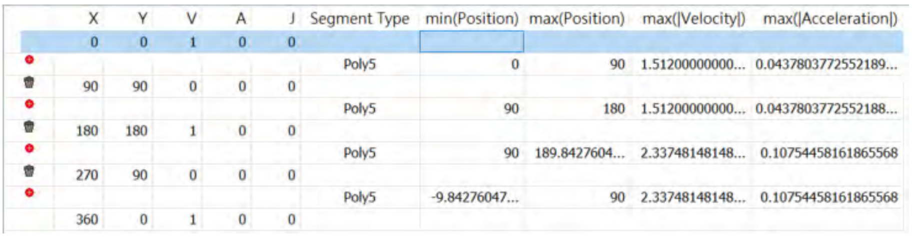
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

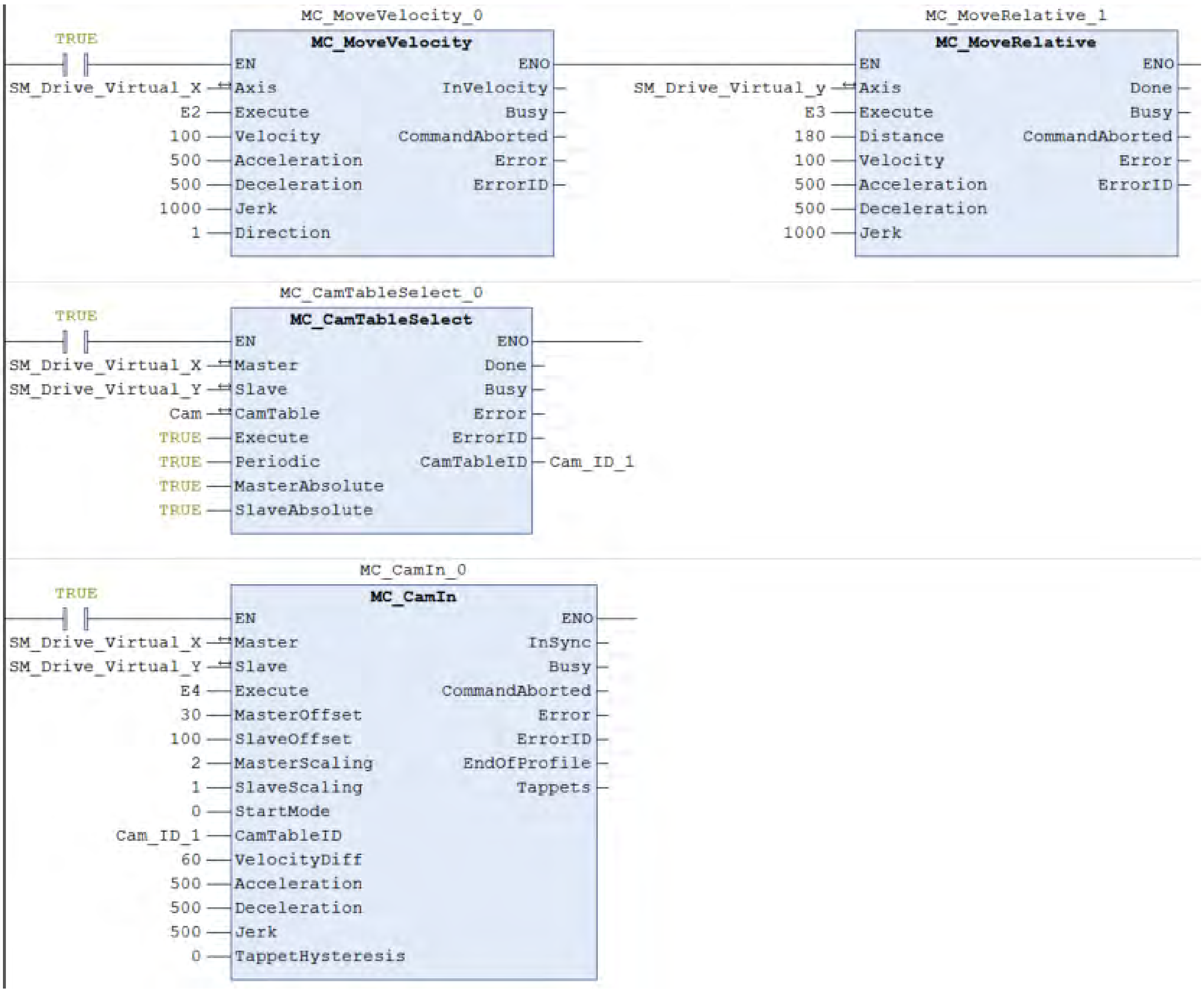
  - **Example1:** Demonstrate the execution result after cam parameters relating to MC_CamInbeing being configured. Both the master and slave axis are rotary axes in this example.

    - Cam curve planning:





	X	Y	V	A	J	Segment Type	min(Position)	max(Position)	max(|Velocity|)	max(|Acceleration|)
	0	0	1	0	0					
●						Poly5	0	90	1.51200000000...	0.0437803772552189...
🗑	90	90	0	0	0					
●						Poly5	90	180	1.51200000000...	0.0437803772552188...
🗑	180	180	1	0	0					
●						Poly5	90	189.8427604...	2.33748148148...	0.10754458161865568
🗑	270	90	0	0	0					
●						Poly5	-9.84276047...	90	2.33748148148...	0.10754458161865568
	360	0	1	0	0					



MC_MoveVelocity_0

MC_MoveVelocity

TRUE — EN; ENO

SM_Drive_Virtual_X — Axis; InVelocity

E2 — Execute; Busy

100 — Velocity; CommandAborted

500 — Acceleration; Error

500 — Deceleration; ErrorID

1000 — Jerk

1 — Direction

MC_MoveRelative_1

MC_MoveRelative

EN; ENO

SM_Drive_Virtual_y — Axis; Done

E3 — Execute; Busy

180 — Distance; CommandAborted

100 — Velocity; Error

500 — Acceleration; ErrorID

500 — Deceleration

1000 — Jerk

MC_CamTableSelect_0

MC_CamTableSelect

TRUE — EN; ENO

SM_Drive_Virtual_X — Master; Done

SM_Drive_Virtual_Y — Slave; Busy

Cam — CamTable; Error

TRUE — Execute; ErrorID

TRUE — Periodic; CamTableID — Cam_ID_1

TRUE — MasterAbsolute

TRUE — SlaveAbsolute

MC_CamIn_0

MC_CamIn

TRUE — EN; ENO

SM_Drive_Virtual_X — Master; InSync

SM_Drive_Virtual_Y — Slave; Busy

E4 — Execute; CommandAborted

30 — MasterOffset; Error

100 — SlaveOffset; ErrorID

2 — MasterScaling; EndOfProfile

1 — SlaveScaling; Tappets

0 — StartMode

Cam_ID_1 — CamTableID

60 — VelocityDiff

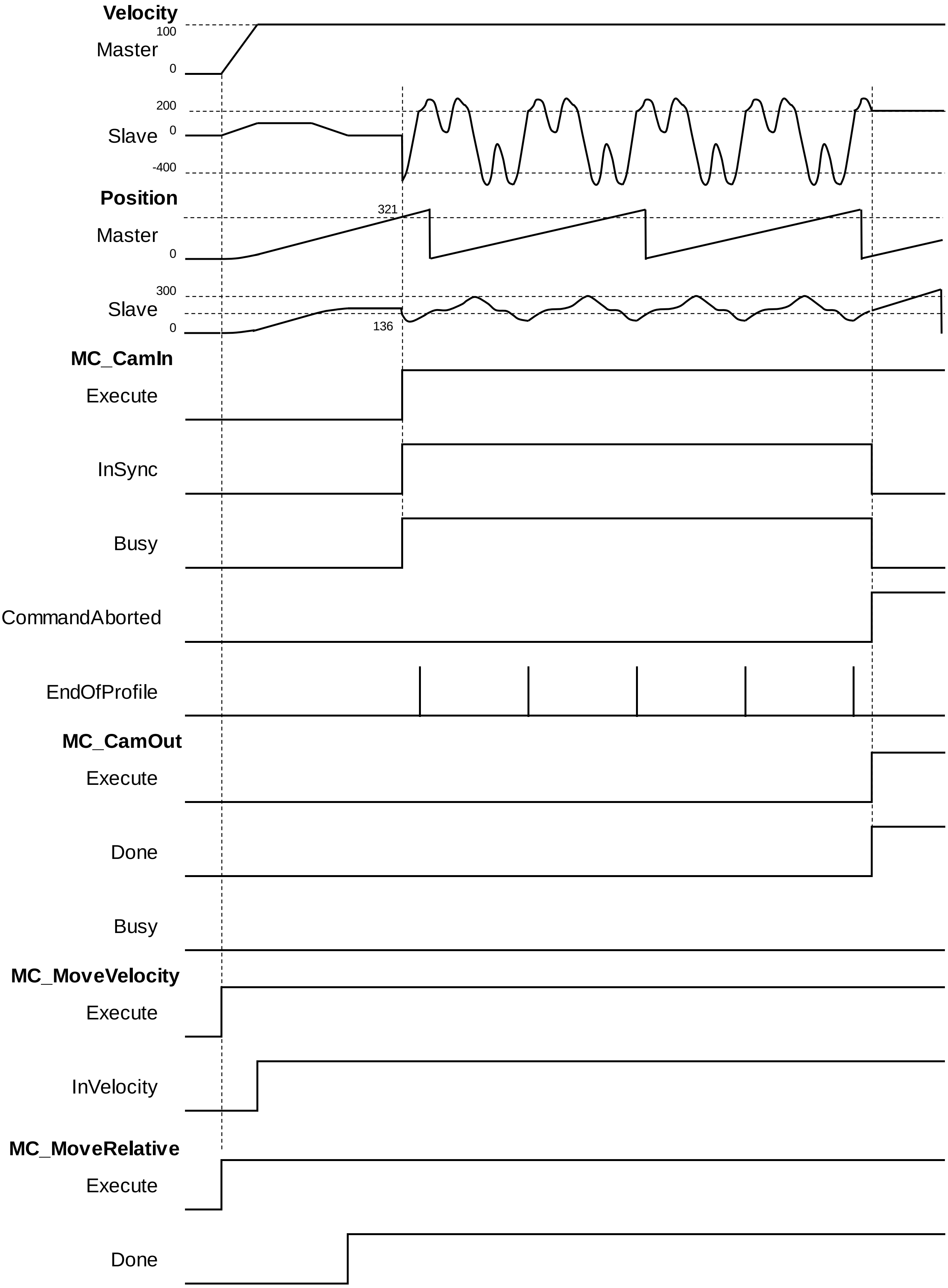
500 — Acceleration

500 — Deceleration

500 — Jerk

0 — TappetHysteresis

- Timing Diagram



- The calculation for axis position and coupling position on cam coordinate:

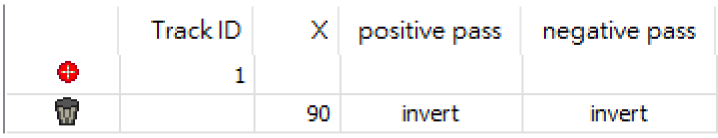
Position_Slave = SlaveScaling×CAM(MasterScaling×MasterPosition + MasterOffset)+SlaveOffset

Slave coupling position = 1 × CAM(2 × 321(master position while executing CamIn)+ 30)+ 100
= 1 × CAM(672)+ 100 = 1 × CAM(312)+ 100
= 36 + 100
= 136

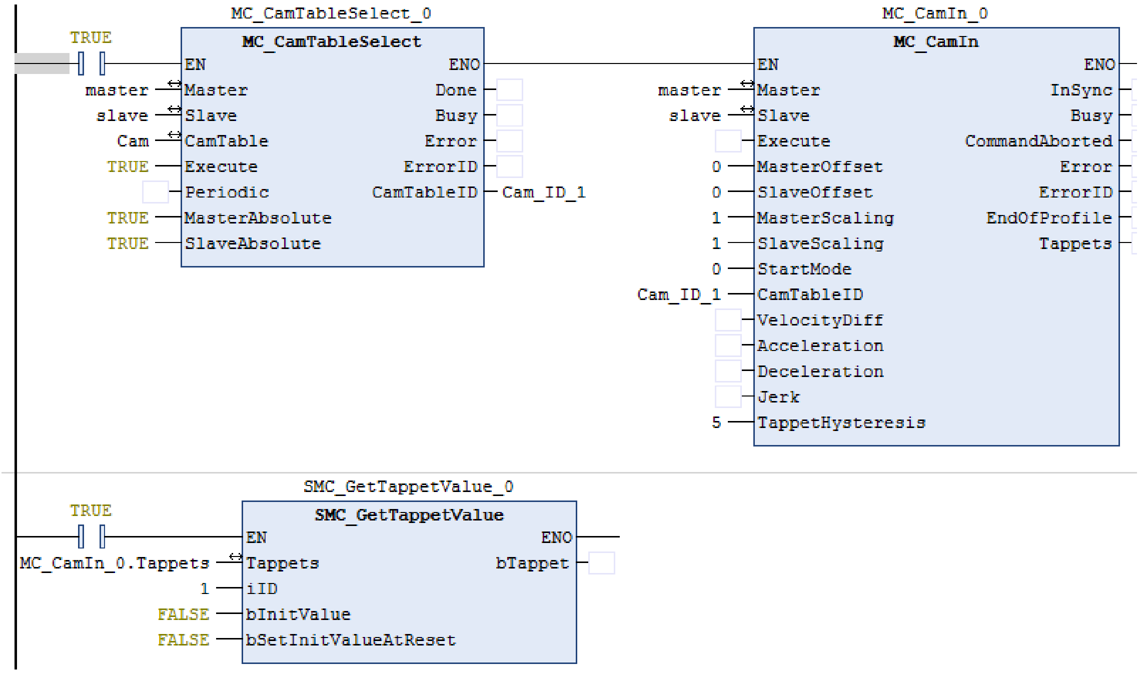
When coupling is completed, the master position would be at 321 and the slave position is at 136.

- **Example2:** The operation of tappet after configuring *TappetHysteresis* as the following example demonstrates.

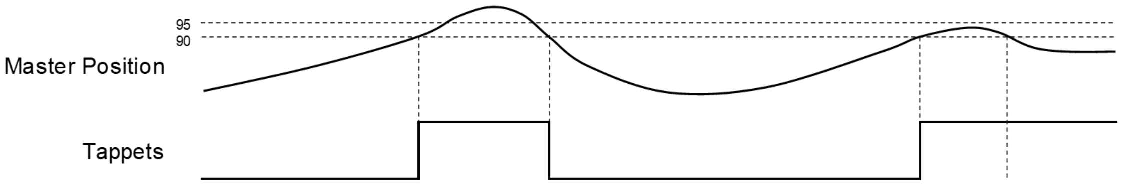
- Tappets



	Track ID	X	positive pass	negative pass
⊕	1			
🗑		90	invert	invert



- Timing Diagram



1. The tappets switch to ON when the master axis passes position 90. Master keeps moving forward until its position exceeds the hysteresis interval and the axis performs reversing. Then the master axis passes position 90 again and exceeds the hysteresis interval, which would make tappets switch to OFF.

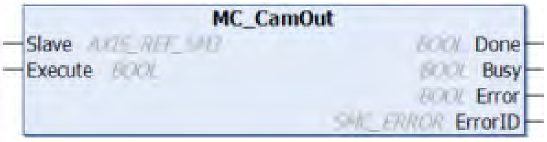
2. The tappets switch to ON when the master axis passes position 90. The master axis keeps moving forward and performs reversing without exceeding the hysteresis interval. Therefore, the tappets will not switch to OFF when the master position passes 90 once again.

- **Supported Products**
  - AX-308E、AX-8

2.1.9 MC_CamOut

MC_CamOut deactivates the coupling between master and slave axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_CamOut	 MC_CamOut
		Slave AXIS_REF_SM3
		Execute BOOL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_CamOut_instance (
			Slave :=,
			Execute :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-

- **Outputs**

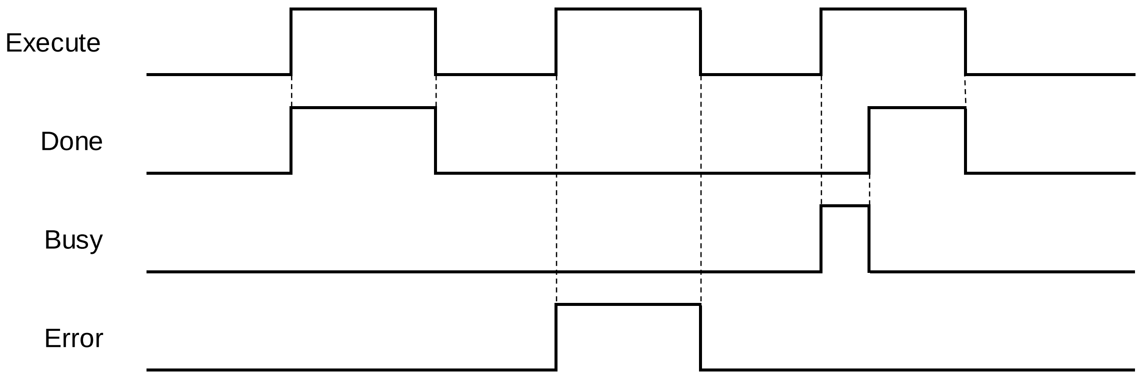
Name	Function	Data Type	Output Range (Default Value)
Done	TRUE when the decoupling task between master and slave is completed.	BOOL	True/False(False)
Busy	TRUE when the instruction is executed.	BOOL	True/False(False)
Error	TRUE when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the instruction CamOut is completed.	● When a falling edge is detected at input Execute. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When an instruction is being executed.	● When a rising edge is detected at output *Error* and *Done*.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When a falling edge is detected at input Execute. (Error codes are cleared.)
ErrorID		

- **Timing Diagram**



- **In-Outs**

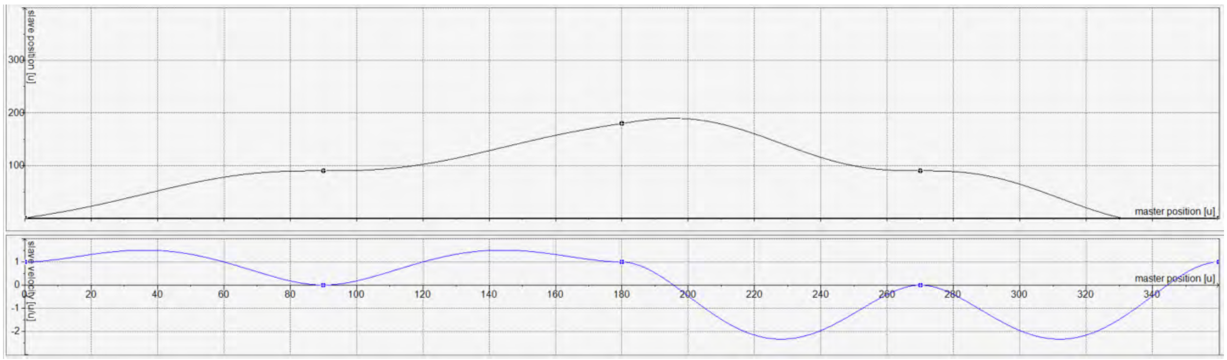
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Slave	Reference to the slave axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

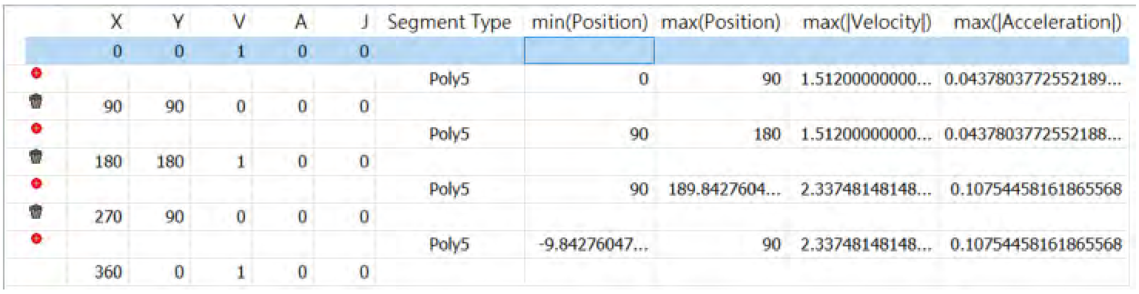
***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - When the slave axis is decoupled from the master axis by MC_CamOut, it moves with the sustained velocity and the slave state is under ContinuousMotion. (irrelevant to the velocity of slave axis)
  - If the synchronization between master and slave axis is not established while executing MC_CamOut. An error of SMC_AXIS_NOT_READY_FOR_MOTION(34) will be reported.
  - The axis state still remains continuous_motion, even though the slave axis is desynchronized at standstill with velocity 0.

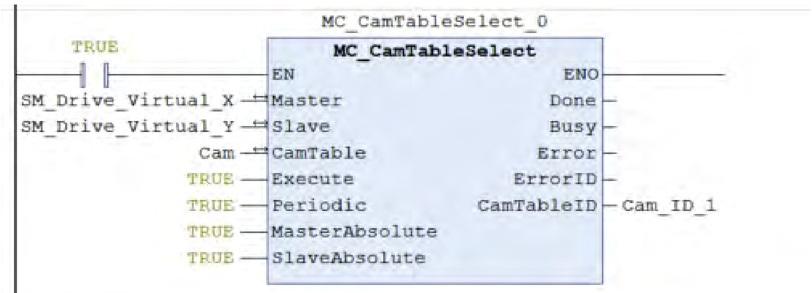
- **Troubleshooting**
  - ■ If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - ■ The following example gives the operation result of MC_CamOut. The master and slave axis in this example are rotary axes.
  - ■ Cam curve planning

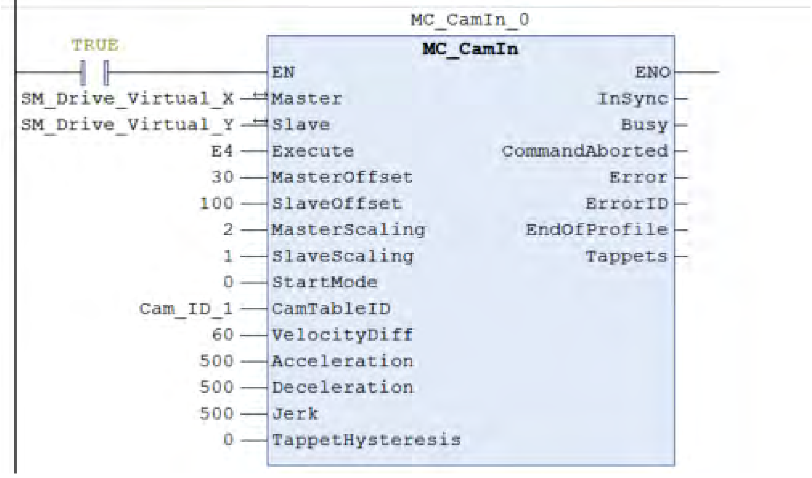




	X	Y	V	A	J	Segment Type	min(Position)	max(Position)	max(|Velocity|)	max(|Acceleration|)
	0	0	1	0	0					
●						Poly5	0	90	1.51200000000...	0.0437803772552189...
🗑	90	90	0	0	0					
●						Poly5	90	180	1.51200000000...	0.0437803772552188...
🗑	180	180	1	0	0					
●						Poly5	90	189.8427604...	2.33748148148...	0.10754458161865568
🗑	270	90	0	0	0					
●						Poly5	-9.84276047...	90	2.33748148148...	0.10754458161865568
	360	0	1	0	0					

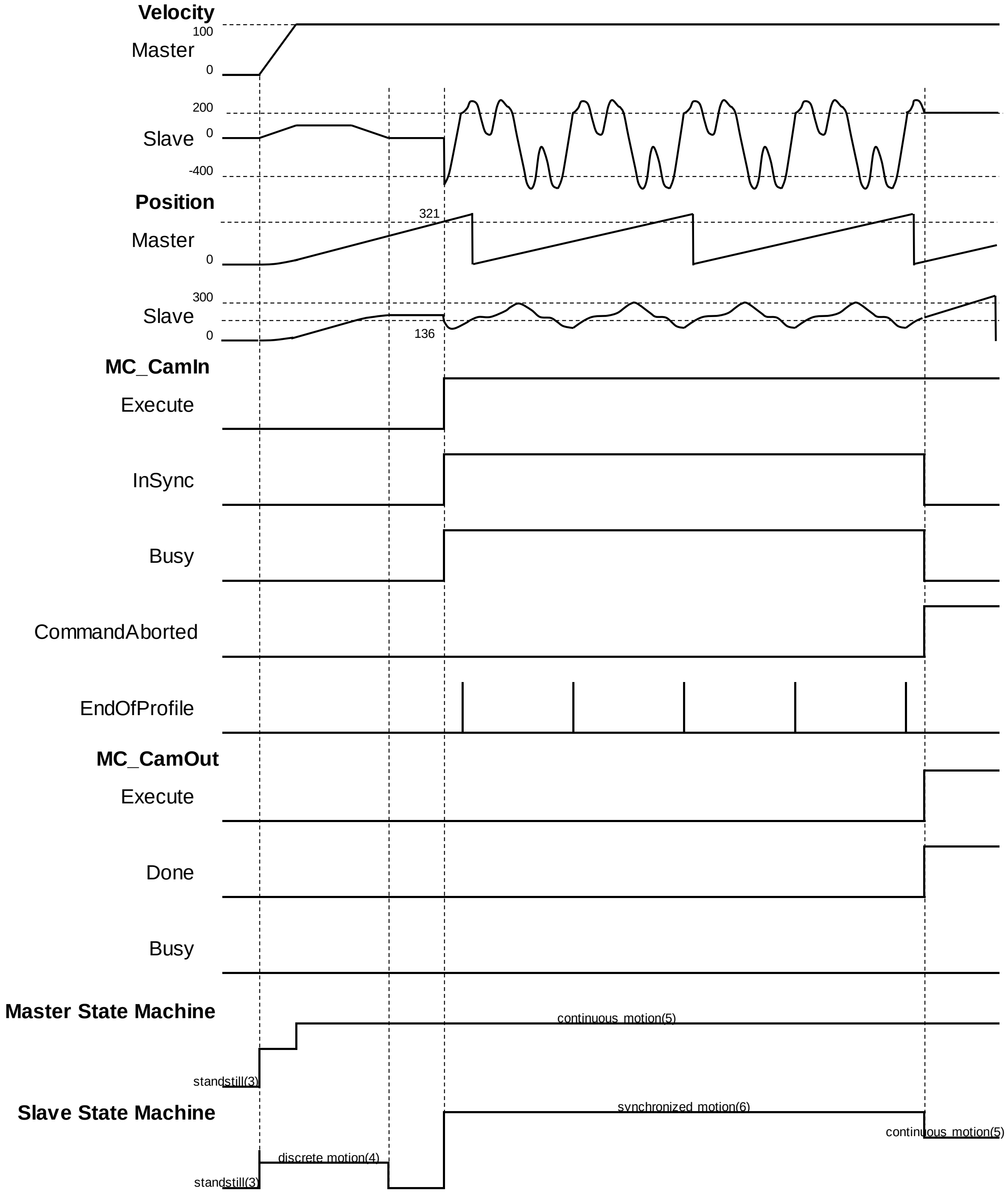


```
                              MC_CamTableSelect_0
     TRUE               MC_CamTableSelect
  ---| |-------------EN                     ENO---------
SM_Drive_Virtual_X ---Master               Done -
SM_Drive_Virtual_Y ---Slave                Busy -
               Cam ---CamTable            Error -
              TRUE ---Execute           ErrorID -
              TRUE ---Periodic      CamTableID - Cam_ID_1
              TRUE ---MasterAbsolute
              TRUE ---SlaveAbsolute
```



```
                                   MC_CamIn_0
     TRUE                     MC_CamIn
  ---| |-------------EN                              ENO---
SM_Drive_Virtual_X ---Master                     InSync -
SM_Drive_Virtual_Y ---Slave                        Busy -
                E4 ---Execute            CommandAborted -
                30 ---MasterOffset                Error -
               100 ---SlaveOffset               ErrorID -
                 2 ---MasterScaling        EndOfProfile -
                 1 ---SlaveScaling              Tappets -
                 0 ---StartMode
          Cam_ID_1 ---CamTableID
                60 ---VelocityDiff
               500 ---Acceleration
               500 ---Deceleration
               500 ---Jerk
                 0 ---TappetHysteresis
```

- Timing Diagram



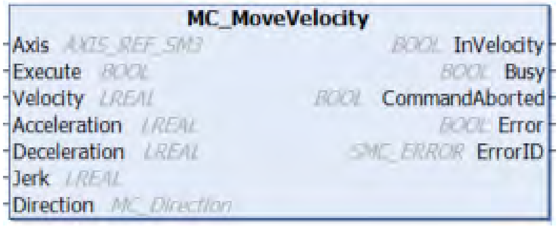
- ◆ The master-slave coupling is deactivated when MC_CamOut is executed. At the same time, MC_CamIn is aborted and a falling edge is detected at *CommandAborted.*
- ◆ The slave axis continues to move at the current speed after being decoupled from the master axis, while the axis state changes to continuous_motion.

- **Supported Products**
  - AX-308E、AX-8

2.1.10 MC_MoveVelocity

MC_MoveVelocity performs velocity control on an axis in the position mode with a specified behavior and an average velocity.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_MoveVelocity	 MC_MoveVelocity
		Axis AXIS_REF_SM3
		Execute BOOL
		Velocity LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		Direction MC_Direction
		BOOL InVelocity
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_MoveVelocity_instance (
			Axis :=,
			Execute :=,
			Velocity :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Direction :=,
			InVelocity =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
Velocity	Target velocity. (Unit: user unit/s)	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.
Acceleration	Acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	Jerk value. (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Direction	Specifies the direction for servo motor rotation.	MC_ Direction *	3:fastest 2:current 1:positive 0:shortest -1:negative (current)	When Execute shifts to True and Busy is False.

***Note:** MC_Direction:Enumeration(Enum)

- **Outputs**

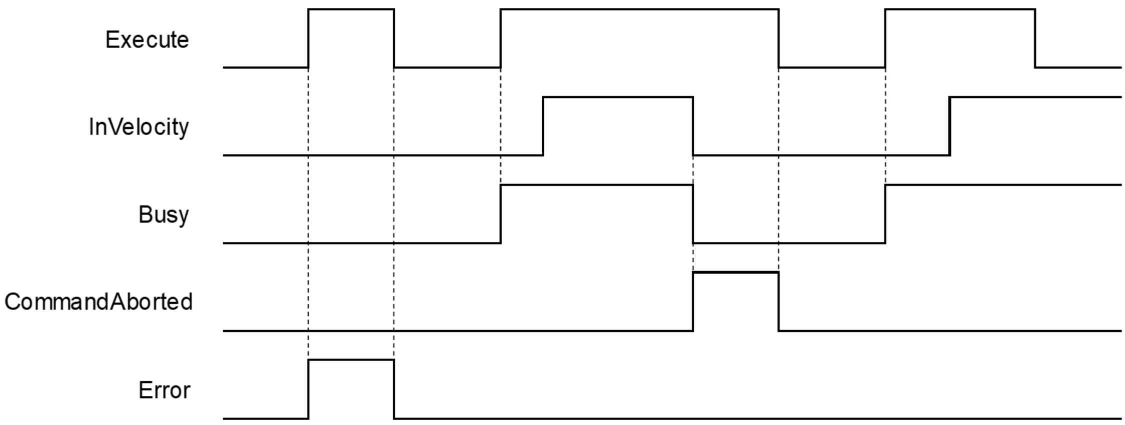
Name	Function	Data Type	Output Range (Default Value)
InVelocity	When the specified target velocity is reached.	BOOL	True/False(False)
Busy	When Execute shifts to True.	BOOL	True/False(False)
CommandAborted	True when this instruction is aborted	BOOL	True/False(False)
Error	True when an error occurs	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
InVelocity	● True when the specified target velocity is reached.	● When CommandAborted shifts to True ● When CommandAborted shifts to True and the target velocity is changed.
Busy	● When Execute shifts to True.	● When Error shifts to True. ● When CommandAborted shifts to True.
CommandAborted	● When this instruction is aborted by another instruction. ● When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

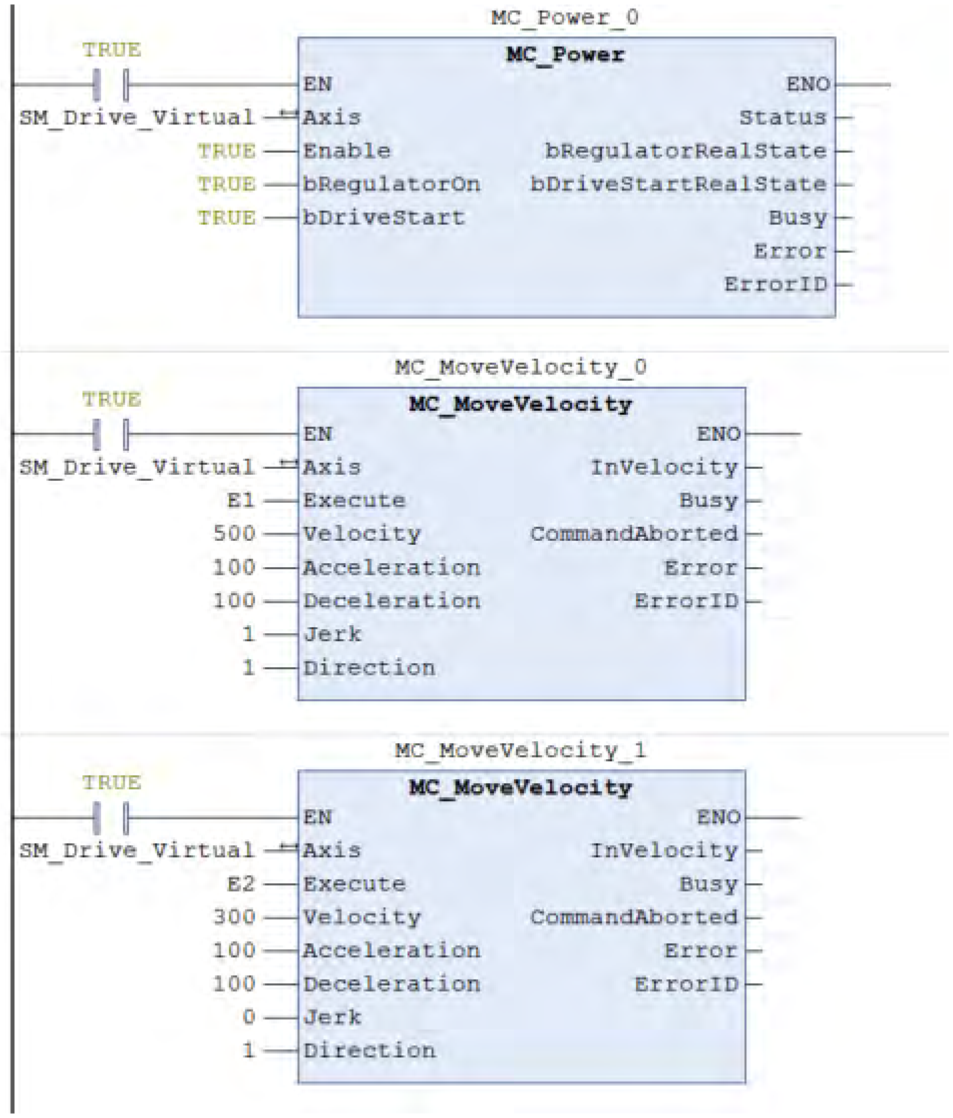
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

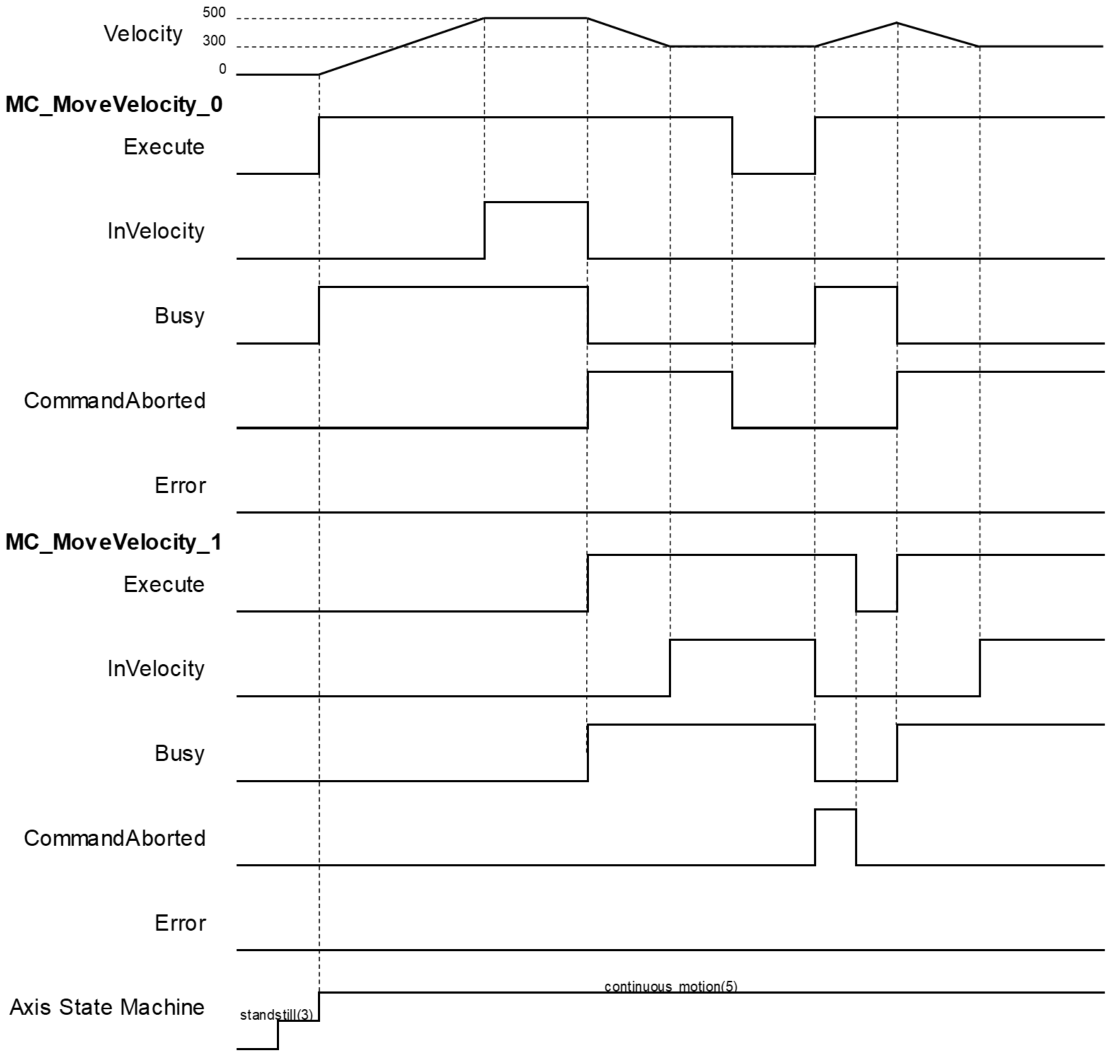
- **Function**
  - The instruction performs speed control with specified target velocity (Velocity), acceleration rate (Acceleration), deceleration rate (Deceleration) and Jerk value (Jerk) when execute changes to True.
  - Users can execute another motion instruction to abort the ongoing motion of MC_MoveVelocity.
  - When aborted by other instructions, OutputsInVelocity would be False and Outputs*CommandAborted is True.*
  - When *Execute* of MC_MoveVelocity changes to True, the axis starts to move at the target velocity. Even if *Execute* turns False, the execution of the function block would not be affected.
  - When ExecuteInputs of MC_MoveVelocity is retriggered and a new target velocity is specified, the axis will change the velocity to the requested velocity.
  - In case the Execute pin changes to False after the function block is executed, *InVelocity* of MC_MoveVelocity will turn True when the target velocity is reached. *InVelocity* will remain as True, until being aborted by other instructions.
  - *InVelocity* remains as True when MC_MoveVelocity reaches the target velocity. Even if the velocity being changed by MC_MoveSuperimposed, the movement of *InVelocity* would not be affected.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below describes the behavior of two MC_MoveVelocity.



■ Timing Diagram



- ◆ When Execute changes to True, the first MC_MoveVelocity controls the axis to reach the specified target velocity 500. When it reaches 500, InVelocity changes to True.
- ◆ If Execute of MC_MoveVelocity_1 changes to True, InVelocity will change to False and CommanAborted will change to True while MC_MoveVelocity_0 is aborted.
- ◆ MC_MoveVelocity_1 will decelerate the axis to the velocity 300. When 300 is reached, InVelocity will change to True and remain in this status as long as the velocity is not changed.
- ◆ When Execute of MC_MoveVelocity_0 changes to False, CommanAborted will change to False.
- ◆ If MC_MoveVelocity_0 is restarted by Execute, which changes to True, the axis will abort MC_MoveVelocity_1 and accelerate toward the velocity 500.
- ◆ Before the axis reaches the target velocity of MC_MoveVelocity_0, Execute of MC_MoveVelocity_1 will again turn False to True and aborts MC_MoveVelocity_0. In this case, the axis decelerates again without reaching the target velocity.

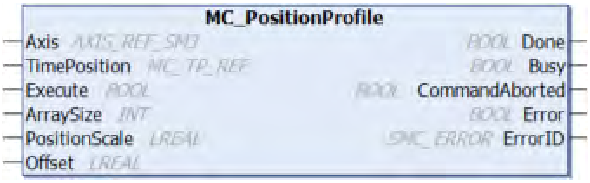
● Supported Products

■ AX-308E、AX-8

2.1.11 MC_PositionProfile

MC_PositionProfile commands a time-position locked motion profile.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_PositionProfile		MC_PositionProfile_instance (
			Axis:=,
			TimePosition:=,
			Execute :=,
			ArraySize:=,
			PositionScale:=,
			Offset:=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
ArraySize	Number of motion profile arrays	INT	Positive number or 0(0)	When Execute shifts to True and Busy is False.
PositionScale	Overall scale factor in value.	LREAL	Negative number, positive number or 0(1)	When Execute shifts to True and Busy is False.
Offset	Overall profile offset in value (Unit: user unit/s)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.

- **Outputs**

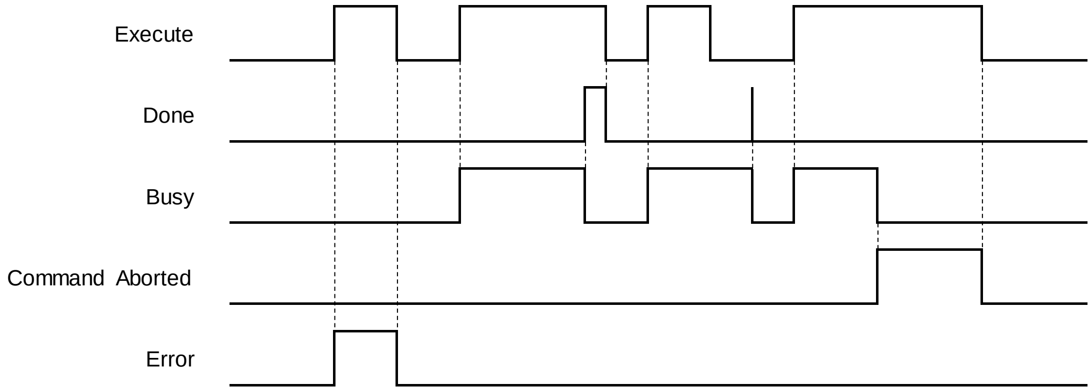
Name	Function	Data Type	Output Range (Default Value)
Done	True when execution of path planning is finished.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the execution of path planning is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done changes to True. ● When Error changes to True. ● When Commandaborted shifts to True.
CommandAborted	● When this instruction is aborted by another function block.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*[1]	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
TimePosition	Time and position during execution.	MC_TP_REF*[2]	MC_TP_REF	When Execute shifts to True and Busy is False.

***Note:**

1. AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.
2. MC_TP_REF: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
Number_of_pairs	There’s no need to set this variable, which has been replaced by InputsArraySize.	INT	-
IsAbsolute	Set the mode of position.	BOOL	True: Abosolue mode False: Relative mode (True)
MC_TP_Array	Time and position data during execution of instruction.	ARRAY [1..100] OF SMC_TP	SMC_TP*

***Note:** SMC_TP: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
delta_time	Period of time between position points	TIME	Positive number or 0(TIME#0ms)
position	Position of the position point	LREAL	Negative number, positive number or 0(0)

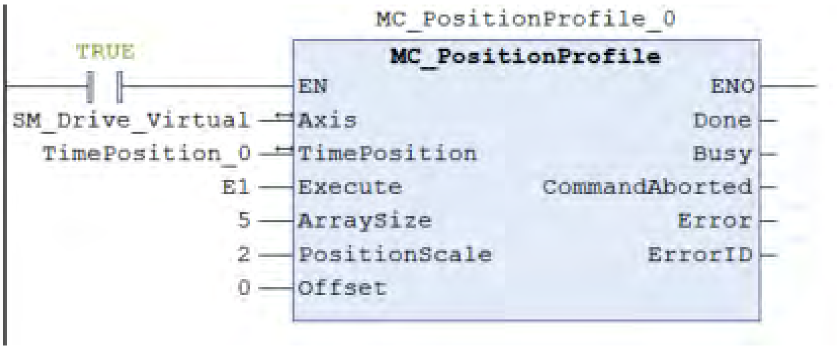
- **Function**

- MC_PositionProfile carries out motion profile with time and position according to the user-defined data in TimePosition variables, the state is Discrete Motion during the movement.

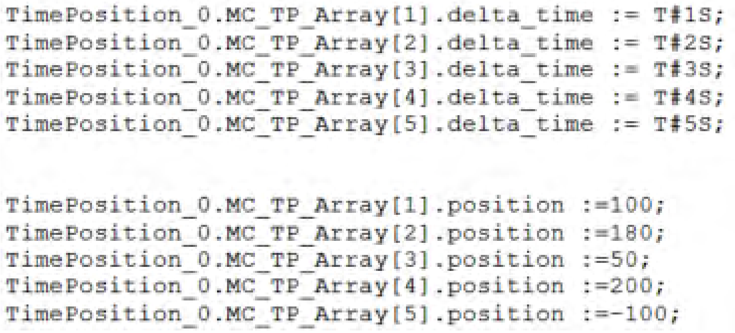
- MC_MoveSuperimposed would not be able to function while MC_PositionProfile is being used.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below describes the behavior of MC_PositionProfile instruction.

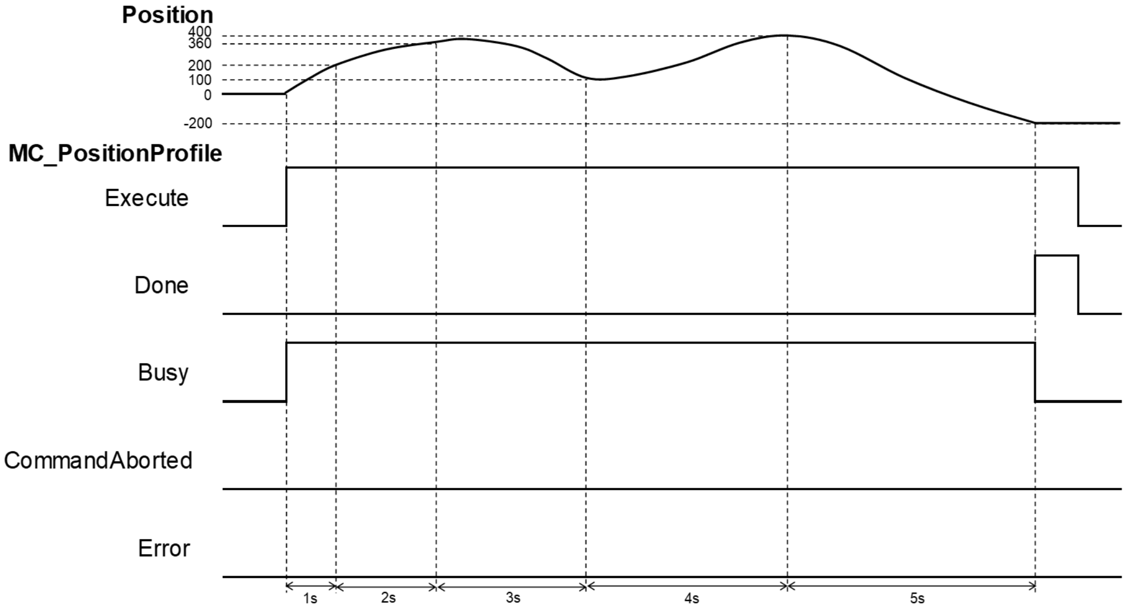


Configure for the curve of motion profile:


```
TimePosition_0.MC_TP_Array[1].delta_time := T#1S;
TimePosition_0.MC_TP_Array[2].delta_time := T#2S;
TimePosition_0.MC_TP_Array[3].delta_time := T#3S;
TimePosition_0.MC_TP_Array[4].delta_time := T#4S;
TimePosition_0.MC_TP_Array[5].delta_time := T#5S;


TimePosition_0.MC_TP_Array[1].position :=100;
TimePosition_0.MC_TP_Array[2].position :=180;
TimePosition_0.MC_TP_Array[3].position :=50;
TimePosition_0.MC_TP_Array[4].position :=200;
TimePosition_0.MC_TP_Array[5].position :=-100;
```

- **Timing Diagram**



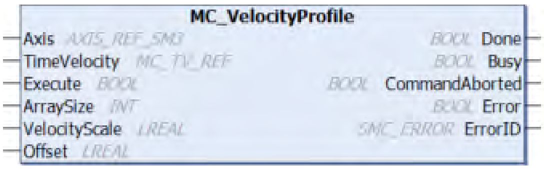
- When *Execute* of MC_PositionProfile is raised, the target axis moves along the curve, which is generated by the settings of delta_time and position in TimePosition.
- The setting value of *IsAbsolute* is True, MC_PositionProfile plans motion curves in the mode of Abosolue position.
- Motion curve of MC_PositionProfile is generated according to the time-position data in TimePosition. As a result of PositionScale=2, the position will be 200 after one second of execution, position 300 after two seconds and so on. When executes after 5 second, the position should be -200.

- **Supported Products**
  - AX-308E、AX-8

2.1.12 MC_VelocityProfile

Similar to MC_PositionProfile, MC_VelocityProfile also commands a time-position locked motion profile; however, its position points are defined by the velocity variables in MC_TV_REF.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_VelocityProfile	 MC_VelocityProfile
		Axis AXIS_REF_SM3
		TimeVelocity MC_TV_REF
		Execute BOOL
		ArraySize INT
		VelocityScale LREAL
		Offset LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_VelocityProfile_instance (
			Axis:=,
			TimeVelocity:=,
			Execute :=,
			ArraySize:=,
			VelocityScale:=,
			Offset:=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
ArraySize	Number of motion profile arrays	INT	Positive number or 0(0)	When Execute shifts to True and Busy is False.
VelocityScale	Overall scale factor in value.	LREAL	Negative number, positive number or 0(1)	When Execute shifts to True and Busy is False.
Offset	Overall profile offset in value (Unit: user unit/s)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.

- **Outputs**

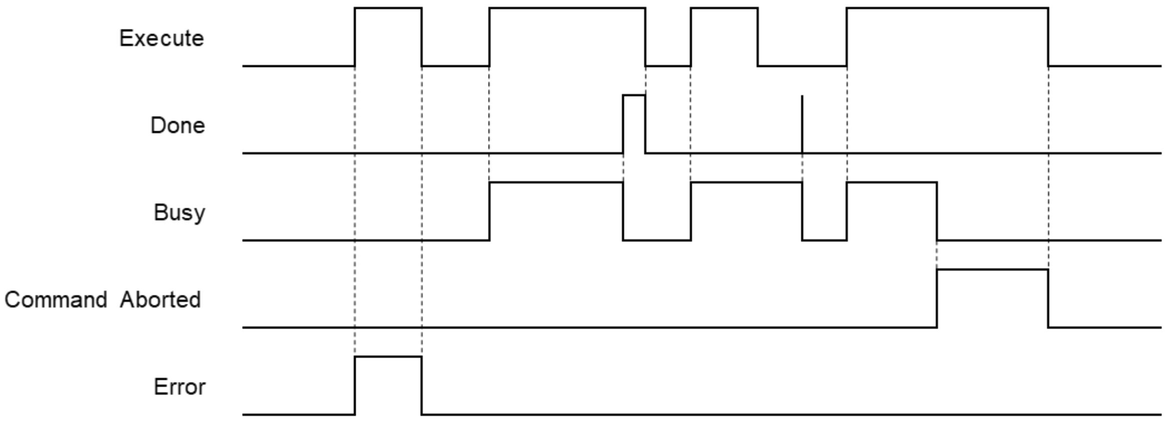
Name	Function	Data Type	Output Range (Default Value)
Done	True when execution of path planning is finished.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the execution of path planning is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done changes to True. ● When Error changes to True. ● When Commandaborted shifts to True.
CommandAborted	● When this instruction is aborted by another function block.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*1	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
TimeVelocity	Time and velocity during execution.	MC_TV_REF*2	MC_TV_REF	When Execute shifts to True and Busy is False.

***Note:**

1. AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.
2. MC_TV_REF: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
Number_of_pairs	There's no need to set this variable, which has been replaced by InputsArraySize.	INT	-
IsAbsolute	Set the mode of velocity.	BOOL	True: Absolute mode False: Relative mode (True)
MC_TV_Array	Time and velocity data during execution of instruction.	ARRAY [1..100] OF SMC_TV	SMC_TV*

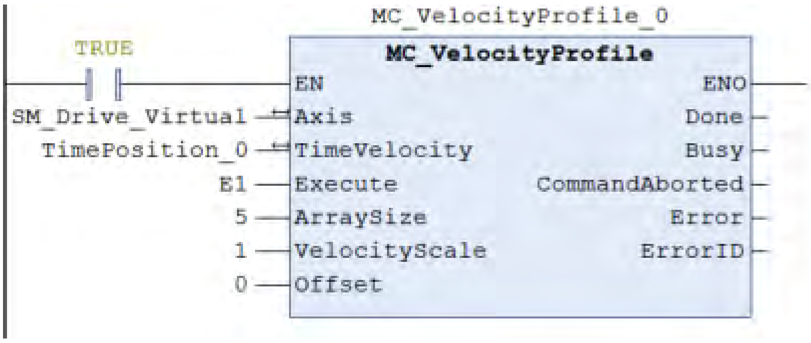
***Note:** SMC_TV: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
delta_time	Period of time between position points	TIME	Positive number or 0(TIME#0ms)
velocity	Velocity of the position point	LREAL	Negative number, positive number or 0(0)

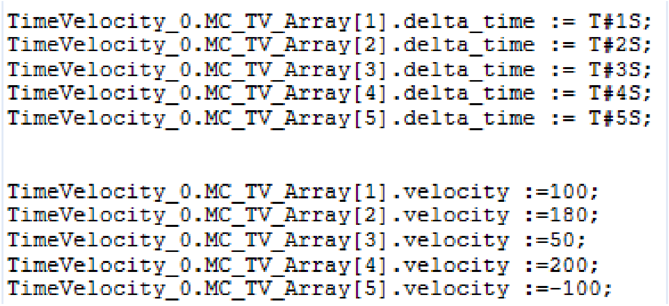
- **Function**
  - MC_VelocityProfile carries out motion profile with time and velocity according to the user-defined data in TimeVelocity variables, the state is Continuous Motion during the movement.
  - MC_VelocityProfile would not be able to function while MC_PositionProfile is being used.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below ddescribes the behavior of MC_VelocityProfile.



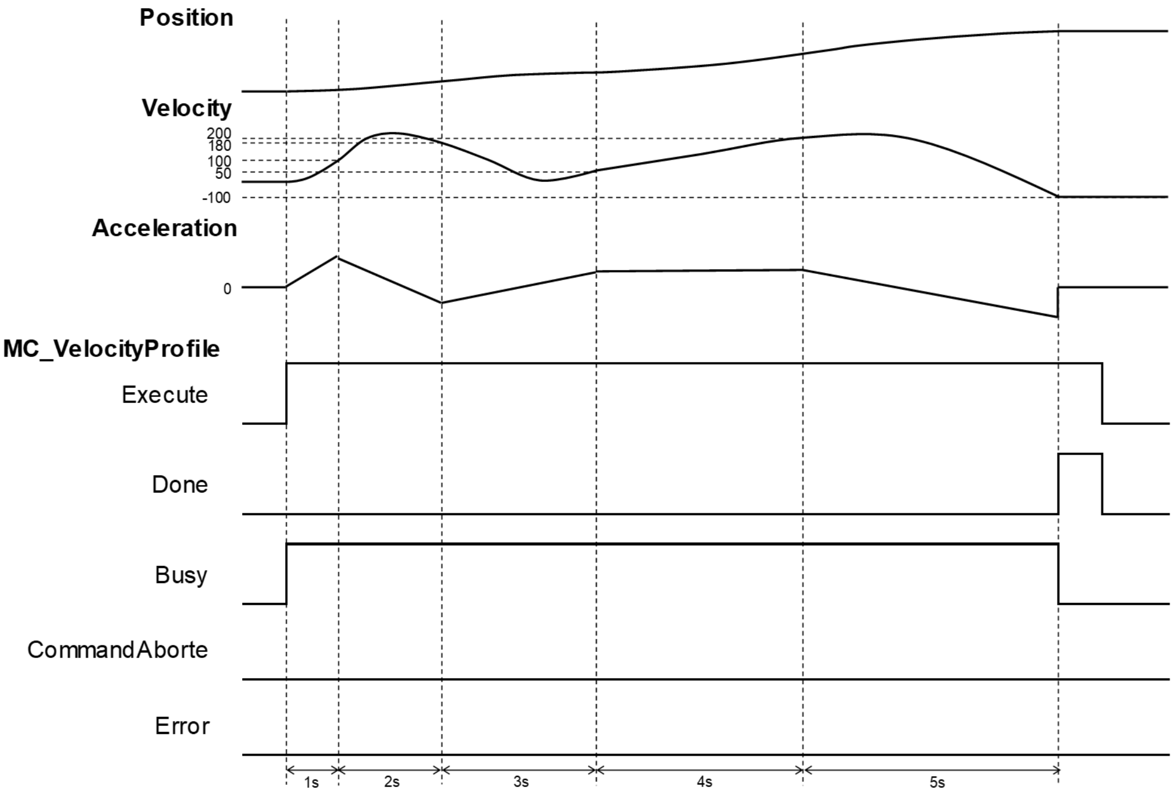
Configure for the curve of motion profile:



```
TimeVelocity_0.MC_TV_Array[1].delta_time := T#1S;
TimeVelocity_0.MC_TV_Array[2].delta_time := T#2S;
TimeVelocity_0.MC_TV_Array[3].delta_time := T#3S;
TimeVelocity_0.MC_TV_Array[4].delta_time := T#4S;
TimeVelocity_0.MC_TV_Array[5].delta_time := T#5S;


TimeVelocity_0.MC_TV_Array[1].velocity :=100;
TimeVelocity_0.MC_TV_Array[2].velocity :=180;
TimeVelocity_0.MC_TV_Array[3].velocity :=50;
TimeVelocity_0.MC_TV_Array[4].velocity :=200;
TimeVelocity_0.MC_TV_Array[5].velocity :=-100;
```

- **Timing Diagram**



◆ When Execute of MC_VelocityProfile is raised, the target axis moves along the curve, which is generated by the settings of delta_time and velocity in TimeVelocity.

◆ The setting value of IsAbsolute is True, MC_VelocityProfile plans motion curves in Abosolue mode.

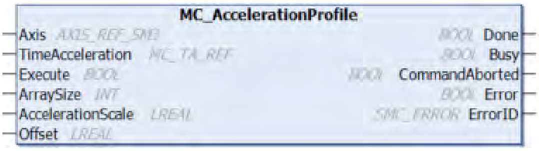
◆ Since Motion curve of MC_VelocityProfile is generated according to the time-velocity data in TimeVelocity, the velocity will be 100 after one second of execution, position 180 after two seconds and so on. When executes after 5 second, the position should be -100.

- **Supported Products**
  - AX-308E、AX-8

2.1.13 MC_AccelerationProfile

Similar to MC_PositionProfile, MC_AccelerationProfile also commands a time-position locked motion profile; however, its position points are defined by acceleration variables in MC_TV_REF.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_AccelerationProfile	 MC_AccelerationProfile
		Axis AXIS_REF_SM3
		TimeAcceleration MC_TA_REF
		Execute BOOL
		ArraySize INT
		AccelerationScale LREAL
		Offset LREAL
		BOOL Done
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_AccelerationProfile _instance (
			Axis:=,
			TimeAcceleration:=,
			Execute :=,
			ArraySize:=,
			AccelerationScale:=,
			Offset:=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
ArraySize	Number of motion profile arrays	INT	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
AccelerationScale	Overall scale factor in value.	LREAL	Negative number, positive number or 0(1)	When Execute shifts to True and Busy is False.
Offset	Overall profile offset in value (Unit: user unit/s)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
Done	True when execution of path planning is finished.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)

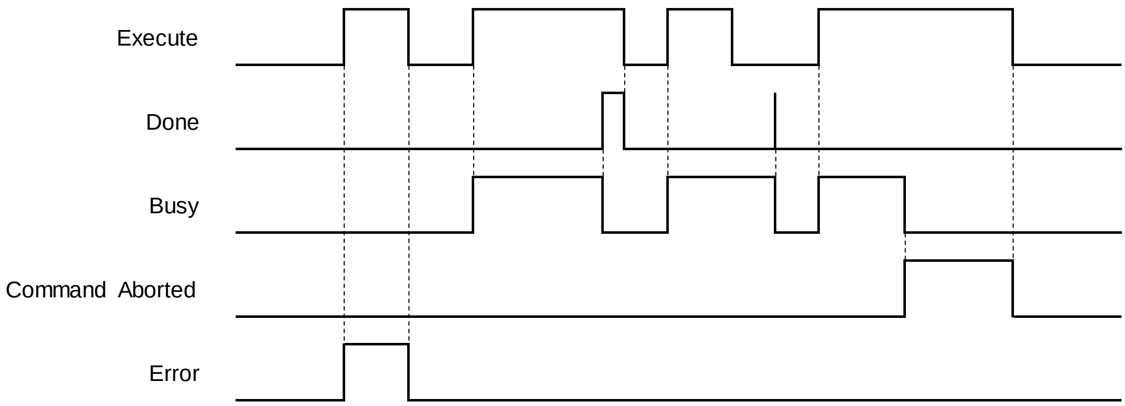
Name	Function	Data Type	Output Range (Default Value)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the execution of path planning is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When Execute changes to True.	● When Done changes to True. ● When Error changes to True. ● When Commandaborted shifts to True.
CommandAborted	● When this instruction is aborted by another function block.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

■ **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*[1]	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
TimeAcceleration	Time and acceleration during execution.	MC_TA_REF*[2]	MC_T_REF	When Execute shifts to True and Busy is False.

***Note:**

1. AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.
2. MC_TA_REF: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
Number_of_pairs	There's no need to set this variable, which has been replaced by InputsArraySize.	INT	-
IsAbsolute	Set the mode of acceleration.	BOOL	True: Absolute mode False: Relative mode (True)
MC_TA_Array	Time and acceleration data during execution of instruction.	ARRAY [1..100] OF SMC_TA	SMC_TA*

***Note:** SMC_TA: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
delta_time	Period of time between position points	TIME	Positive number or 0(TIME#0ms)
acceleration	Acceleration of the position point	LREAL	Negative number, positive number or 0(0)

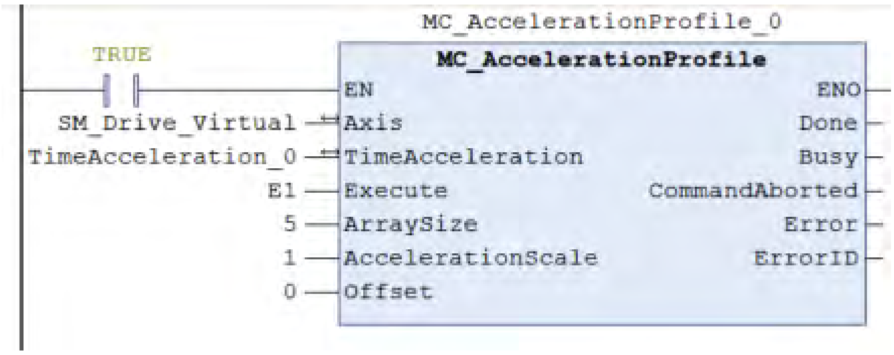
- **Function**

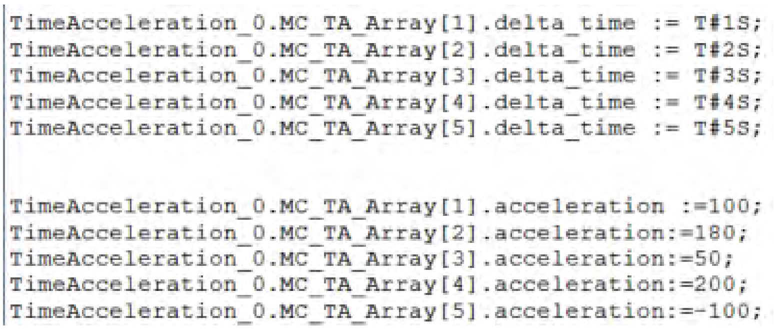
■ MC_AccelerationProfile carries out motion profile with time and acceleration according to the user-defined data in TimeAcceleration variables, the state is Continuous Motion during the movement.

■ MC_MoveSuperimposed cannot add a specific distance to an existing motion of MC_AccelerationProfile.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The example below describes the behavior of MC_AccelerationProfile.

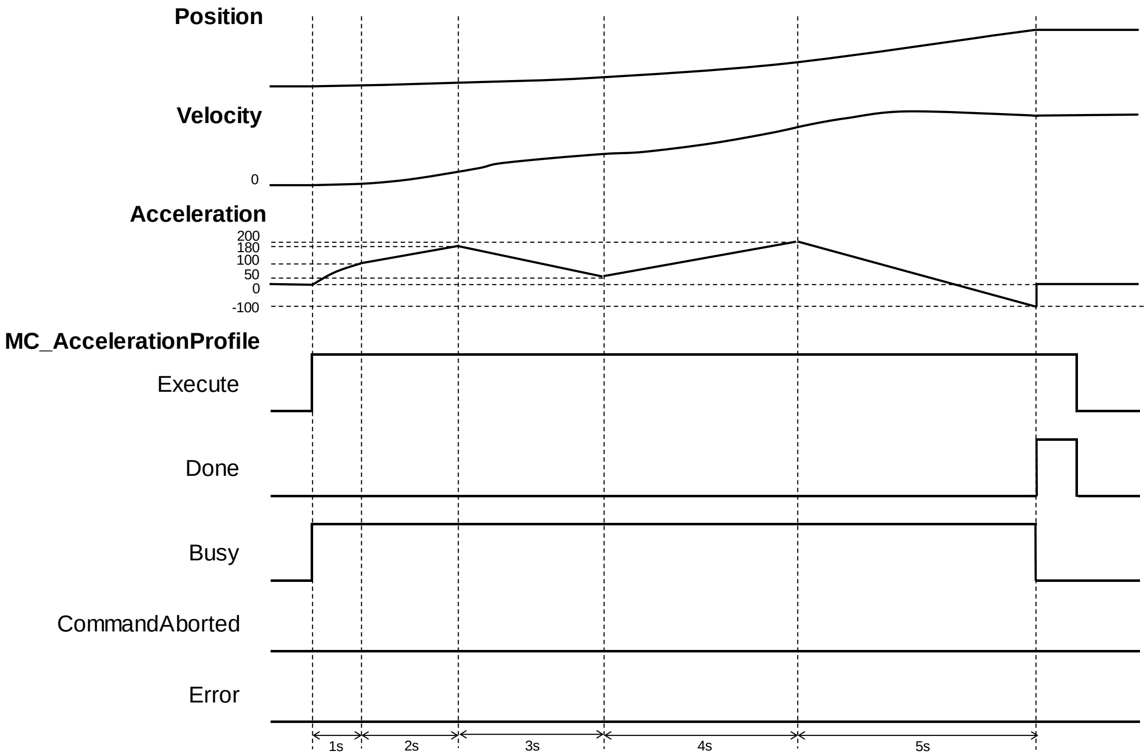




```
TimeAcceleration_0.MC_TA_Array[1].delta_time := T#1S;
TimeAcceleration_0.MC_TA_Array[2].delta_time := T#2S;
TimeAcceleration_0.MC_TA_Array[3].delta_time := T#3S;
TimeAcceleration_0.MC_TA_Array[4].delta_time := T#4S;
TimeAcceleration_0.MC_TA_Array[5].delta_time := T#5S;


TimeAcceleration_0.MC_TA_Array[1].acceleration :=100;
TimeAcceleration_0.MC_TA_Array[2].acceleration:=180;
TimeAcceleration_0.MC_TA_Array[3].acceleration:=50;
TimeAcceleration_0.MC_TA_Array[4].acceleration:=200;
TimeAcceleration_0.MC_TA_Array[5].acceleration:=-100;
```

- **Timing Diagram**



◆ When Execute of MC_AccelerationProfile is raised, the target axis moves along the curve, which is generated by the settings of delta_time and Acceleration in TimeAcceleration.

◆ The setting value of IsAbsolute is True, MC_AccelerationProfile plans motion curves in Abosolue mode.

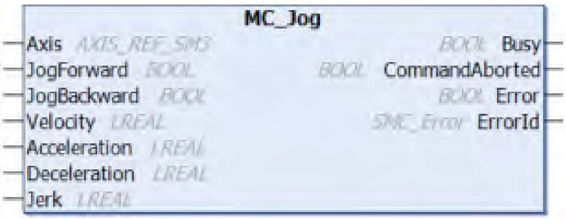
◆ Since Motion curve of MC_AccelerationProfile is generated according to the time-acceleration data in TimeAcceleration, the velocity will be 100 after one second of execution, position 180 after two seconds and so on. When executes after 5 seconds, the position should be -100.

- **Supported Products**
  - AX-308E、AX-8

2.1.14 MC_Jog

MC_Jog enables an axis to be moved forward or backward.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Jog	 MC_Jog
		Axis AXIS_REF_SM3
		JogForward BOOL
		JogBackward BOOL
		Velocity LREAL
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_Error ErrorId
			MC_Jog_instance (
			Axis :=,
			JogForward:=,
			JogBackward:=,
			Velocity :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
JogForward	JogForward changes from FALSE to TRUE.	BOOL	True/False (False)	-
JogBackward	JogBackward changes from FALSE to TRUE	BOOL	True/False (False)	-
Velocity	The target velocity (Unit: user unit/s)	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.
Acceleration	Acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	Jerk value. (Unit: user unit/s3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

* **Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Busy	● True when JogForward or JogBackward shifts to True.	● When Error shifts to True. ● When CommandAborted shifts to True. ● When status is standstill and there’s no complementarity between the status of JogForward and JogBackward.
CommandAborted	● True when the instruction is aborted.	● When JogForward and JogBackward shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When JogForward and JogBackward shift to False.(Error code is cleared)
ErrorID		

- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When JogForward or JogBackward shift to True.

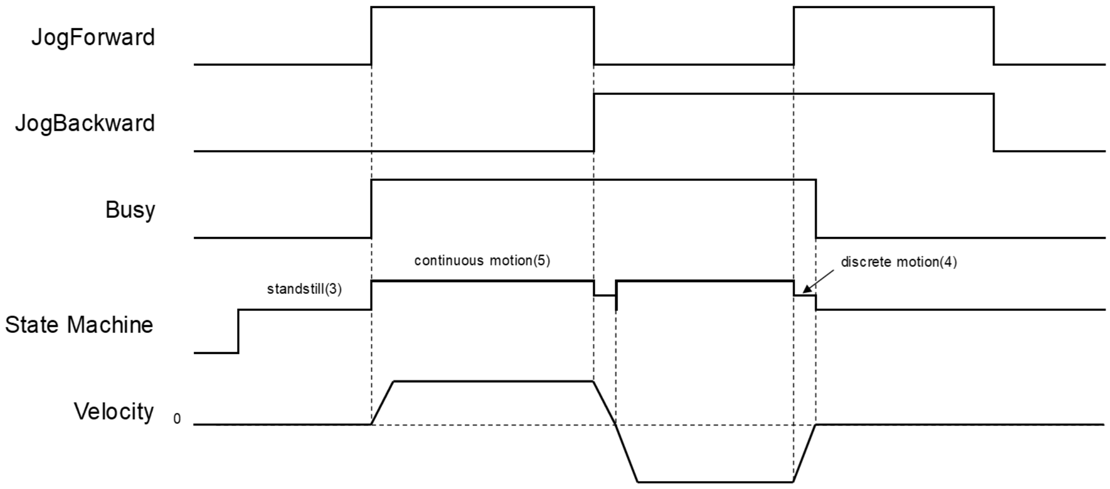
* **Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

- MC_Jog starts a continuous movement with the specified Velocity, while *JogForward* and *JogBackward* determine on the direction of axis movement (negative or positive).

JogForward	JogBackward	Movement
False	False	Motionless
True	False	Jog in positive direction
False	True	Jog in negative direction
True	True	Motionless

- When changing the direction, MC_Jog will reread the Input parameter settings ro perform Jog operation.



- The following description is with reference to the above figure:
    - When *JogFoward* is True and *JogBackward* is False, the axis starts to move in positive direction, which the status is continuous_motion.
    - When *JogFoward* is False and *JogBackward* is True, the axis starts to move in negative direction, which the status is discrete_motion.
    - When decelerating to reach 0 velocity, the status will shift to Standstill, then the axis starts to accelerate in negative direction and change the status to continuous_motion; at the same time, Busy is still True.
    - When both *JogForward* and *JogBackward* areTrue, the axis will accelerate/ decelerate to reach 0 velocity, the status would be discrete_motion.
    - When both JogForward and JogBackward are True, the axis will stop moving and the status would be Standstill with Busy output changing from Busy to False.

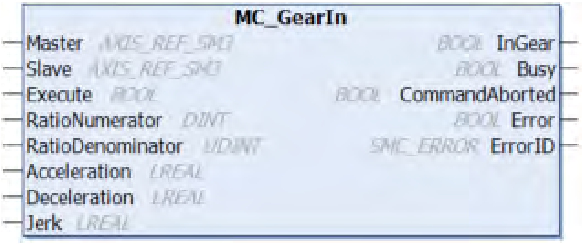
- **Troubleshooting**
    - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
    - AX-308E、AX-8

2.1.15 MC_GearIn

The function block MC_GearIn activates a linear master-slave coupling.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_GearIn	 MC_GearIn
		Master AXIS_REF_SM3
		Slave AXIS_REF_SM3
		Execute BOOL
		RatioNumerator DINT
		RatioDenominator UDINT
		Acceleration LREAL
		Deceleration LREAL
		Jerk LREAL
		BOOL InGear
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_GearIn_instance (
			Master :=,
			Slave :=,
			Execute :=,
			RatioNumerator :=,
			RatioDenominator :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			InGear =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>
			);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
RatioNumerator	Gear ratio numerator* between master and slave axis.	DINT	Negative number, positive number or 0(1)	When Execute shifts to True and Busy is False.
RatioDenominator	Gear ratio denominato* between master and slave axis.	UDINT	Positive number(1)	When Execute shifts to True and Busy is False.
Acceleration	Acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	Jerk value. (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.

***Note:** A negative gear ratio will make the master and slave axis move in an opposite direction.

- **Outputs**

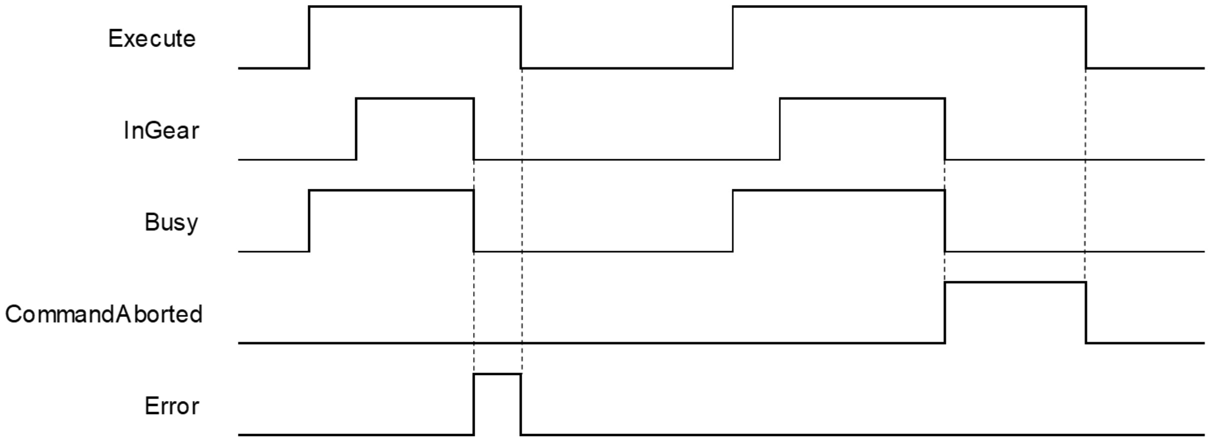
Name	Function	Data Type	Output Range (Default Value)
InGear	True if the coupling was successful.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR: Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
InGear	● When the slave axis reaches the target velocity and the coupling is successful.	● When *Error* shifts to True. ● When CommandAborted shifts to True. ● When the gear ratio changes.
Busy	● When the coupling is processed.	● When Error shifts to True. ● When CommandAborted shifts to True.
CommandAborted	● When MC_GearOut is executed. ● True when the instruction is aborted by another function block. ● When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**

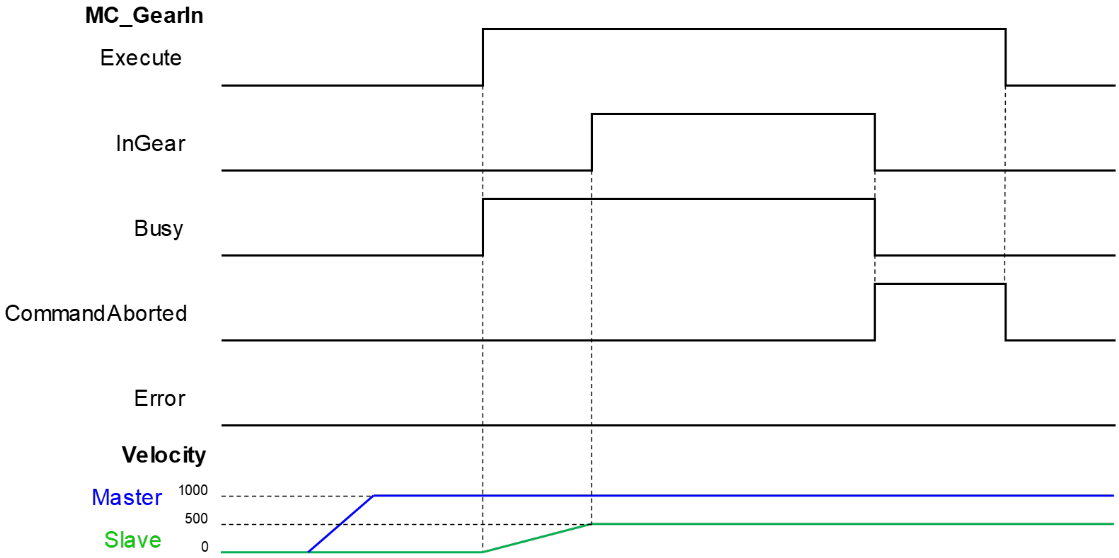


- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Master	Specifies the master axis number.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Execute* shifts to True and Busy is False.
Slave	Specifies the slave axis number.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Execute* shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

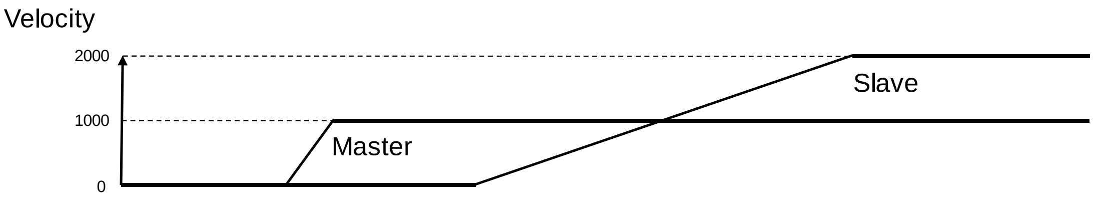
- **Function**
  - In MC_GearIn, slave axis will follow master axis to move at the Set Position.
  - When Execute is True, The target speed of slave axis is the velocity of master axis times gear ratio (Velocity * RatioNumerator / RatioDenominator)



- After the gear relation is established, slave axis will follow master axis to move at the given proportional relationship to accomplish the synchronized control of master and slave axis. Master and slave axis could be real or virtual axis or the external encoder master axis.

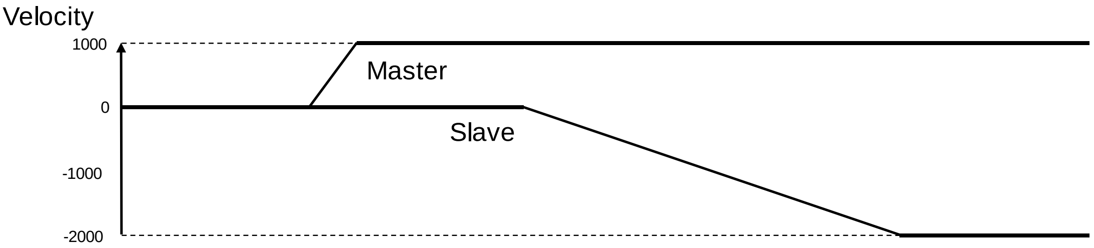
  - RatioNumerator、RatioDenominator

    - When the value of gear ratio is positive, the master and slave axis move in same direction.



Ratio Numerator: Ratio Denominator = 2:1

    - When the value of gear ratio is negative, the master and slave axis move in opposite direction.



Ratio Numerator: Ratio Denominator = -2:1

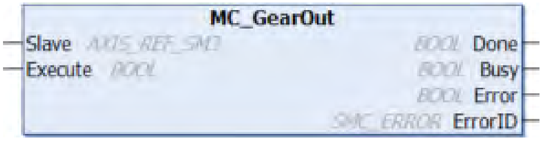
- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - AX-308E、AX-8

2.1.16 MC_GearOut

MC_GearOut disconnects the gear relation (velocity) between master and slave axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_GearOut	 MC_GearOut
		Slave AXIS_REF_SM3
		Execute BOOL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_GearOut_instance (
			Slave :=,
			Execute :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-

- **Outputs**

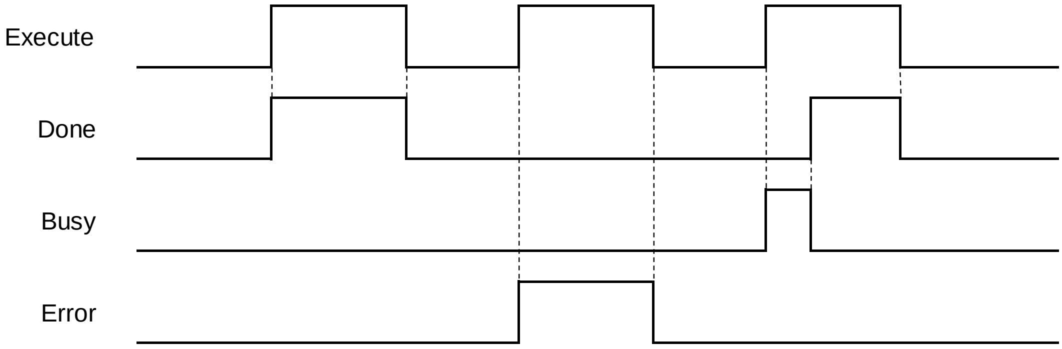
Name	Function	Data Type	Output Range (Default Value)
Done	True when the gear disconnection is completed.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the gear disconnection is completed.	● When Execute changes to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False. ● When *Error* shifts to True.
Busy	● When Execute changes to True.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.(Error code is recorded)	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**

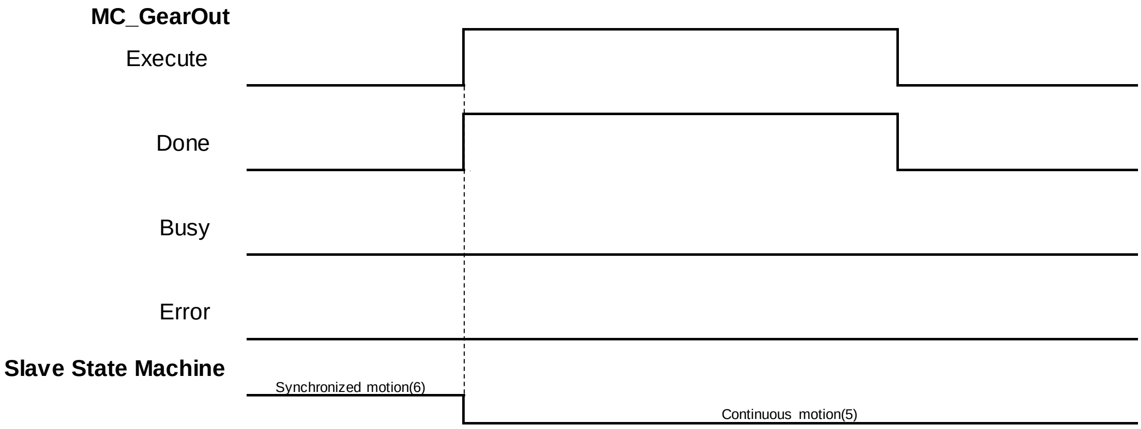


- **In-Outs**

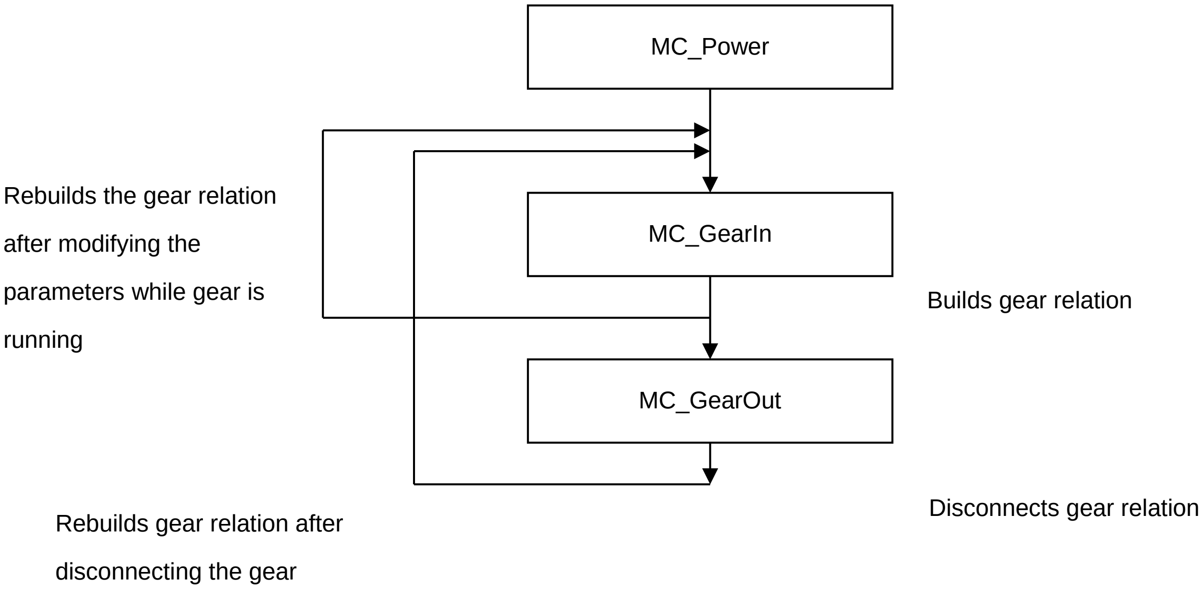
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Slave	Slave axis number	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - After the gear relationship is disconnected, the slave axis will keep moving at the speed where the gear relationship is disconnected. The axis will be in ContinuousMotion (it has nothing to do with the axis velocity)

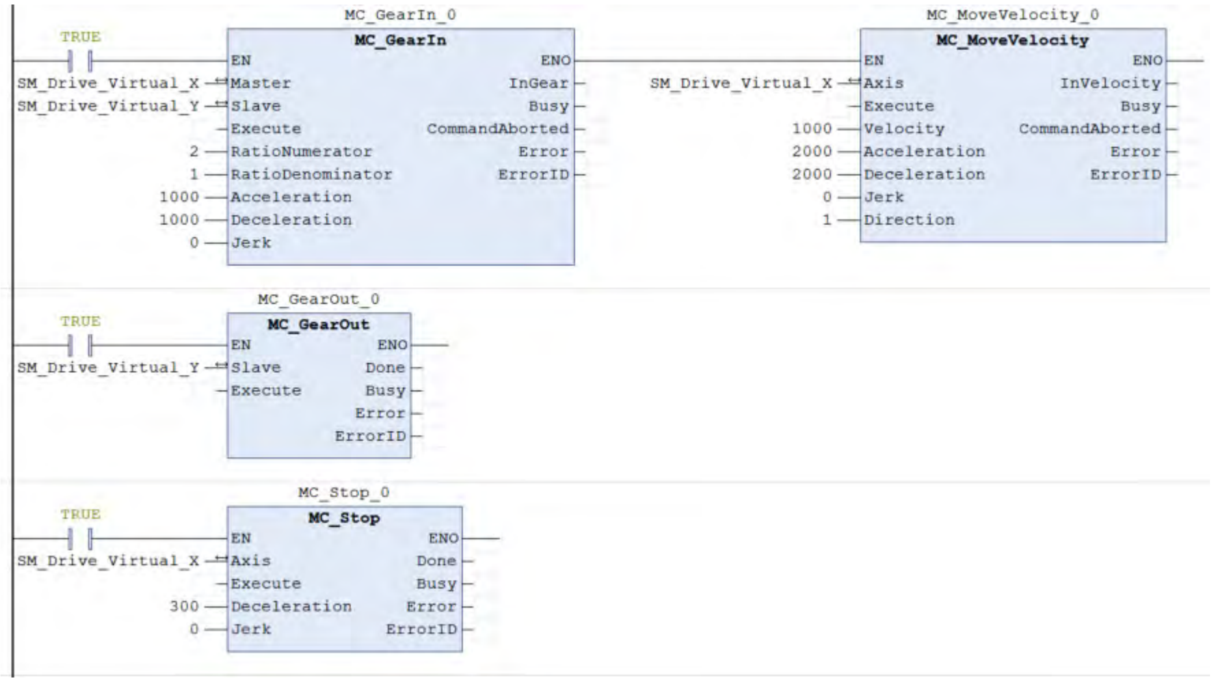


  - When the slave axis is out of sync and the velocity is zer, the status will be continuous_motion and remain unchanged.
  - The sequence for execution of the instructions related to electronic gear.

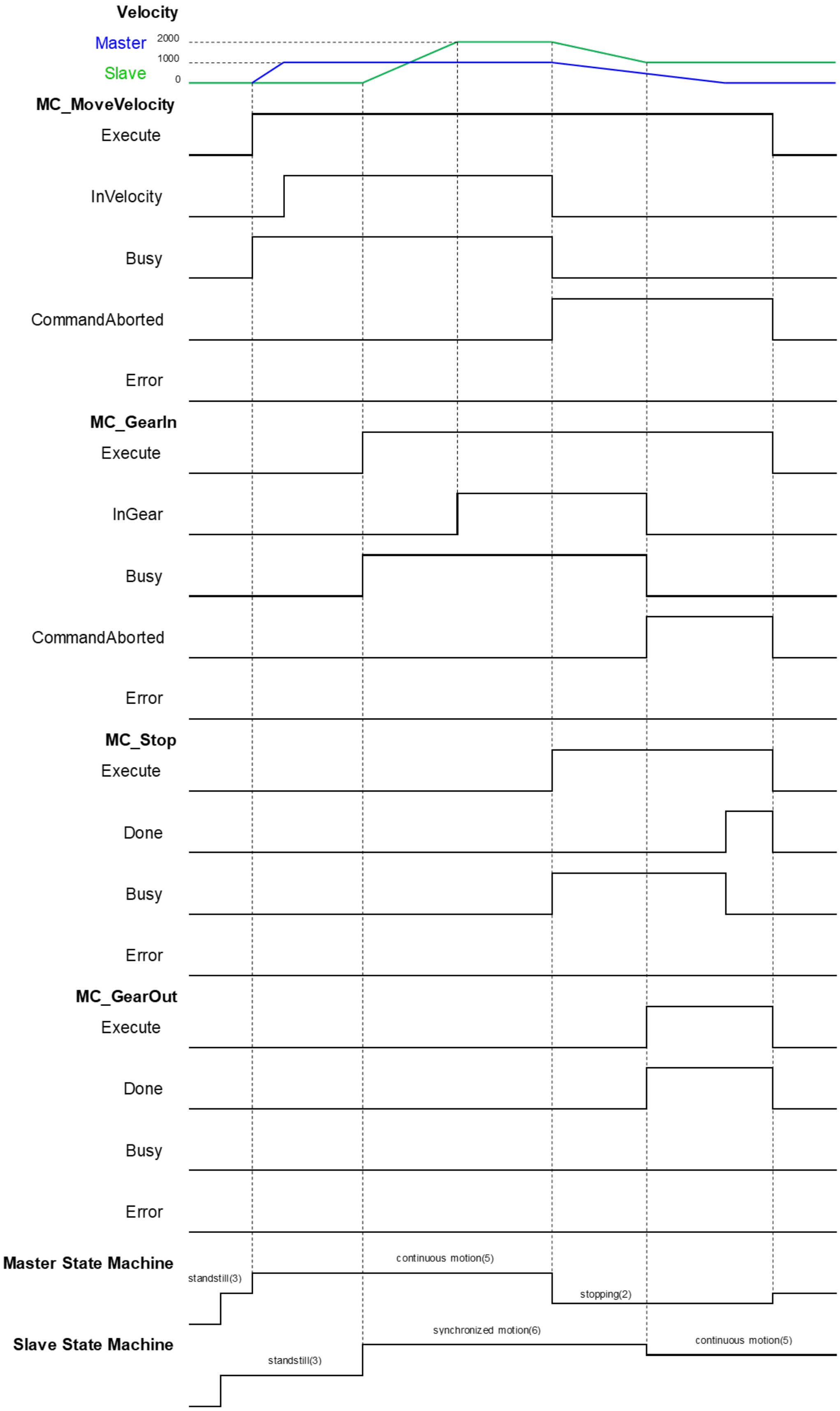

MC_Power
MC_GearIn
MC_GearOut
Rebuilds the gear relation after modifying the parameters while gear is running
Builds gear relation
Disconnects gear relation
Rebuilds gear relation after disconnecting the gear

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - ■ The following example describes the corresponding motion state throughout the gear operation via gear-related instructions. 。



■ Timing Diagram



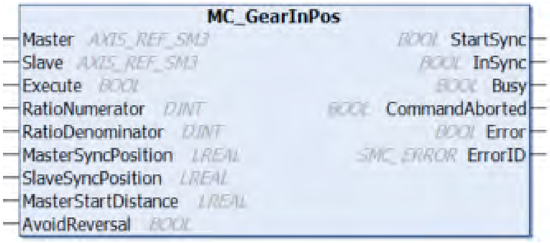
- ◆ When Execute of MC_MoveVelocity changes to True, master axis starts to move.。
- ◆ When M1 Execute of MC_GearIn changes to True, the slave axis starts to catch the master axis.。
- ◆ If the velocity of salve axis reaches doubled master axis' velocity (RatioNumerator: RatioDenominator = 2:1), *InGear* of MC_GearIn will change to True. After the master axis is synchronized with slave axis, the state of slave axis shifts to Synchronized Motion.
- ◆ When Execute of MC_Stop changes to True, the master axis starts to decelerate. At the same time, the slave axis also decelerates based on the gear ratio.
- ◆ In the process of the MC_Stop execution, when Execute of MC_GearOut changes to True, master and slave axis will be under asynchronous status and slave axis will keep moving at the speed (Continuous Motion state) when the gear relation is decoupled.

- **Supported Products**
  - ■ AX-308E、AX-8

2.1.17 MC_GearInPos

MC_GearInPos commands position synchronization which is achieved over a defined region of travel for both master and slave axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_GearInPos	 MC_GearInPos
		Master AXIS_REF_SM3
		Slave AXIS_REF_SM3
		Execute BOOL
		RatioNumerator DINT
		RatioDenominator DINT
		MasterSyncPosition LREAL
		SlaveSyncPosition LREAL
		MasterStartDistance LREAL
		AvoidReversal BOOL
		BOOL StartSync
		BOOL InSync
		BOOL Busy
		BOOL CommandAborted
		BOOL Error
		SMC_ERROR ErrorID
			MC_GearInPos_instance (
			Master :=,
			Slave :=,
			Execute :=,
			RatioNumerator :=,
			RatioDenominator :=,
			MasterSyncPosition :=,
			SlaveSyncPosition :=,
			MasterStartDistance :=,
			AvoidReversal :=,
			StartSync =>,
			InSync =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
RatioNumerator	Gear ratio numerator* between master and slave axis.	DINT	Negative number, positive number or 0(1)	When Execute shifts to True and Busy is False.
RatioDenominator	Gear ratio denominato* between master and slave axis.	UDINT	Positive number or 0(1)	When Execute shifts to True and Busy is False.
MasterSyncPosition	Master Position at which the axes are synchronized.	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
SlaveSyncPosition	Slave Position at which the axes are synchronized.	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
MasterStartDistance	Master Distance for synchronization procedure.	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
AvoidReversal	Reversal is not allowed.	BOOL	True/False(False)	When Execute shifts to True and Busy is False.

***Note:** A negative gear ratio will make the master and slave axis move in an opposite direction.

- **Outputs**

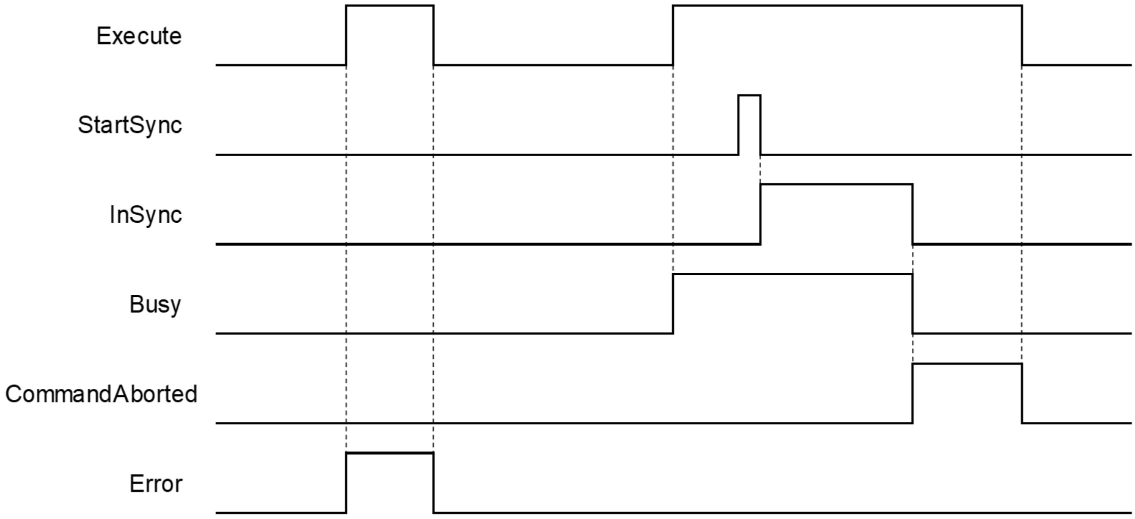
Name	Function	Data Type	Output Range (Default Value)
StartSync	True when the synchronization starts.	BOOL	True/False(False)
InSync	True when the synchronization is ongoing.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

■ **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
StartSync	● When master axis travels to the start position.	● When *InSync* shifts to True. ● When *Error* shifts to True.
InSync	● When the synchronization between master and slave axis is completed.	● When *CommandAborted* shifts to True. ● When *Error* shifts to True.
Busy	● After the synchronization begins.	● When *CommandAborted* shifts to True. ● When *Error* shifts to True.
CommandAborted	● When MC_GearOut is executed. ● True when the instruction is aborted by another function block. ● When this instruction is aborted because of the execution of MC_Stop instruction.	● When Execute changes to False. ● If Execute is False and CommandAborted shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

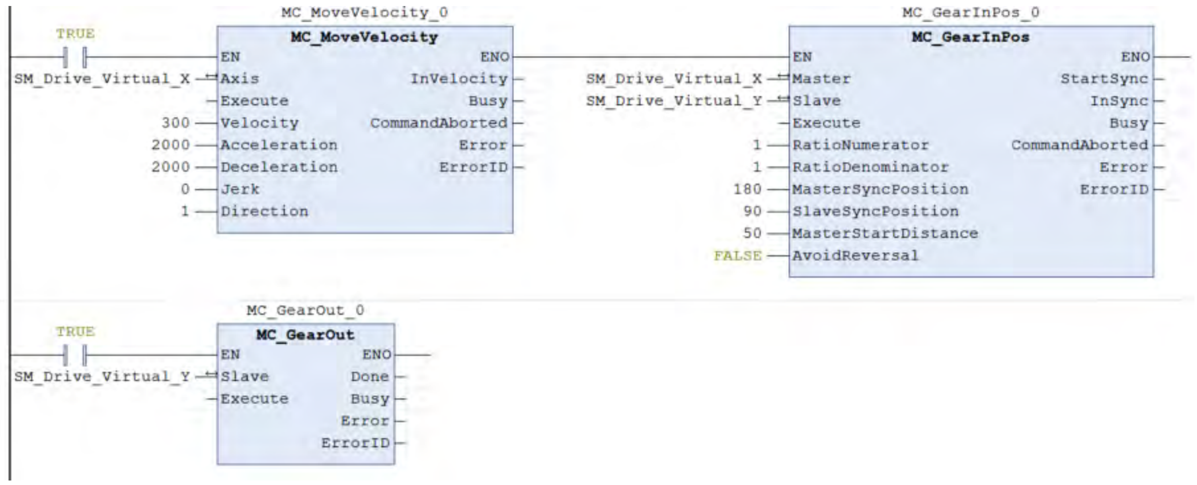
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Master	Specifies the master axis number.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Execute* shifts to True and Busy is False.
Slave	Specifies the slave axis number.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Execute* shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

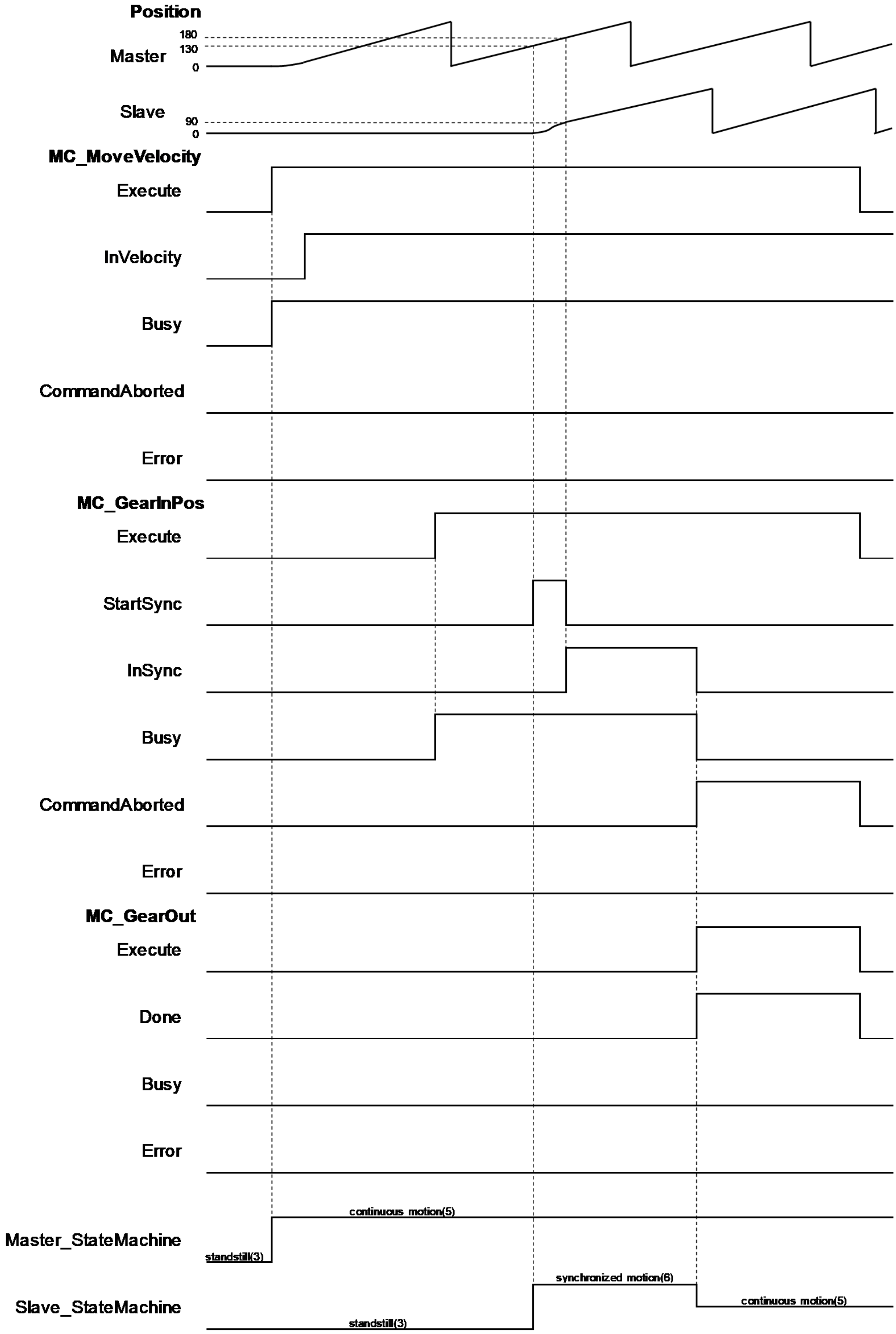
- **Function**
  - Position where the master axis executes StartSync = MasterSyncPosition – MasterStartDistance。
  - It's a must to ensure an appropriate parameter settings of sync position if both master and slave axis are working under Finite mode. Supposed the master and slave axis moving in the positive direction, if the master axis position missed the StartSync position, the gear would not be able to run normally. Therefore, it is suggested to set the master and slave axis operating under Modulo mode.
  - During the progress of synchronization between the master and slave axis, MC_GearInPos begins to plan the motion path of slave axis automatically with gear ratio based on parameters of the position where master axis executes StartSync, MasterSyncPosition and SlaveSyncPosition. After synchgronizing complete, slave axis will start to move by following master axis.
  - When MasterStartDistance = 0 or being negative, CAM motion will be completed immediately.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The following example describes the corresponding motion state throughout the gear operation via MC_GearInPos-related instructions.



- **Timing Diagram**



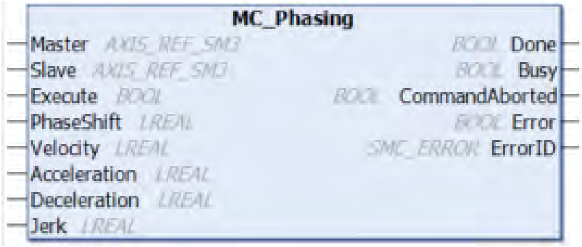
- When *Execute of* MC_MoveVelocity shifts to True, the master axis starts to move.
- When *Execute of* MC_GearInPos shifts to True, it’s waiting for master axis to reach StartSync position.
- When StartSync position is reached, Outputs*StartSync* of MC_GearInPos shifts to True. At the same time, a motion curve is planed for slave axis to move, which the axis would enter Synchronized Motion state.
- When both master and slave axis reach the synchronization position, Outputs*InSync* of MC_GearInPos shifts to True and Outputs*StartSync* changes to False.
- When *Execute* of MC_GearOut shifts to True, the master and slave axis move asynchronously, entering Continuous Motion state.

- **Supported Products**
  - AX-308E、AX-8

2.1.18 MC_Phasing

MC_Phasing specifies the phase shift value between the master and slave axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Phasing		MC_Phasing_instance (
			Master :=,
			Slave :=,
			Execute :=,
			PhaseShift :=,
			Velocity :=,
			Acceleration :=,
			Deceleration :=,
			Jerk :=,
			Done =>,
			Busy =>,
			CommandAborted =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
PhaseShift	Phase shift amount between master and slave axis*	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
Velocity	The max velocity of the phase shift amount (Unit: user unit/s)	LREAL	Positive number or 0(0)	When Execute shifts to True and Busy is False.
Acceleration	The max acceleration of the phase shift amount (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Deceleration	The max deceleration of the phase shift amount (Unit: user unit/s^2)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.
Jerk	The max Jerk value of the phase shift amount (Unit: user unit/s^3)	LREAL	Positive number(0)	When Execute shifts to True and Busy is False.

* **Note:** In case of positive values, the slave axis is behind the master axis. Conversely, the slave axis is ahead of the master axis when the value is negative.

Outputs

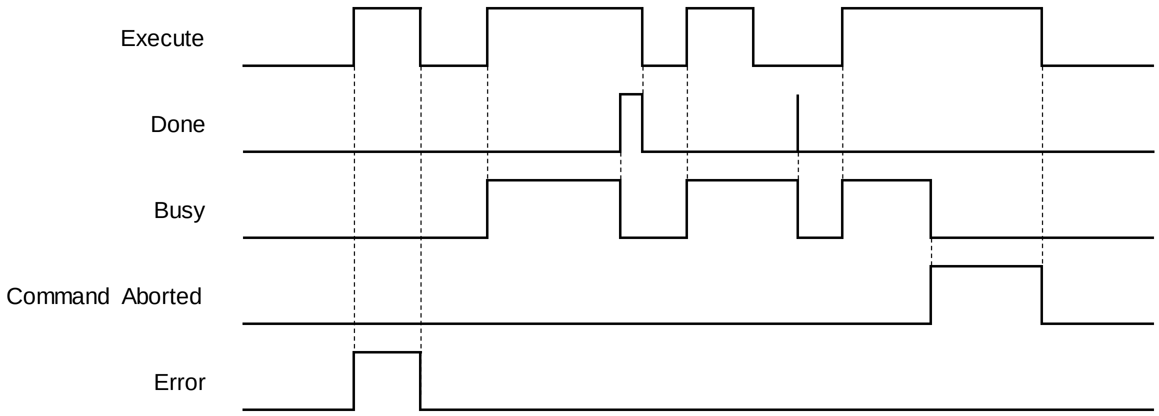
Name	Function	Data Type	Output Range (Default Value)
Done	True when phasing operation is completed.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
CommandAborted	True when the instruction is aborted.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

* **Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the phasing operation is completed.	● When Execute changes to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When the phasing operation is executed.	● When *Error* shifts to True. ● When *CommandAborted* shifts to True.
CommandAborted	● When this instruction is aborted by another instruction with the Buffer Mode set to Aborting. ● When this instruction is aborted because of the execution of MC_Stop instruction.	● When *Execute* changes to False. ● If *Execute* is False and *CommandAborted* shifts to True, it will be True for only one period and immediately shift to False.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When *Execute* shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**

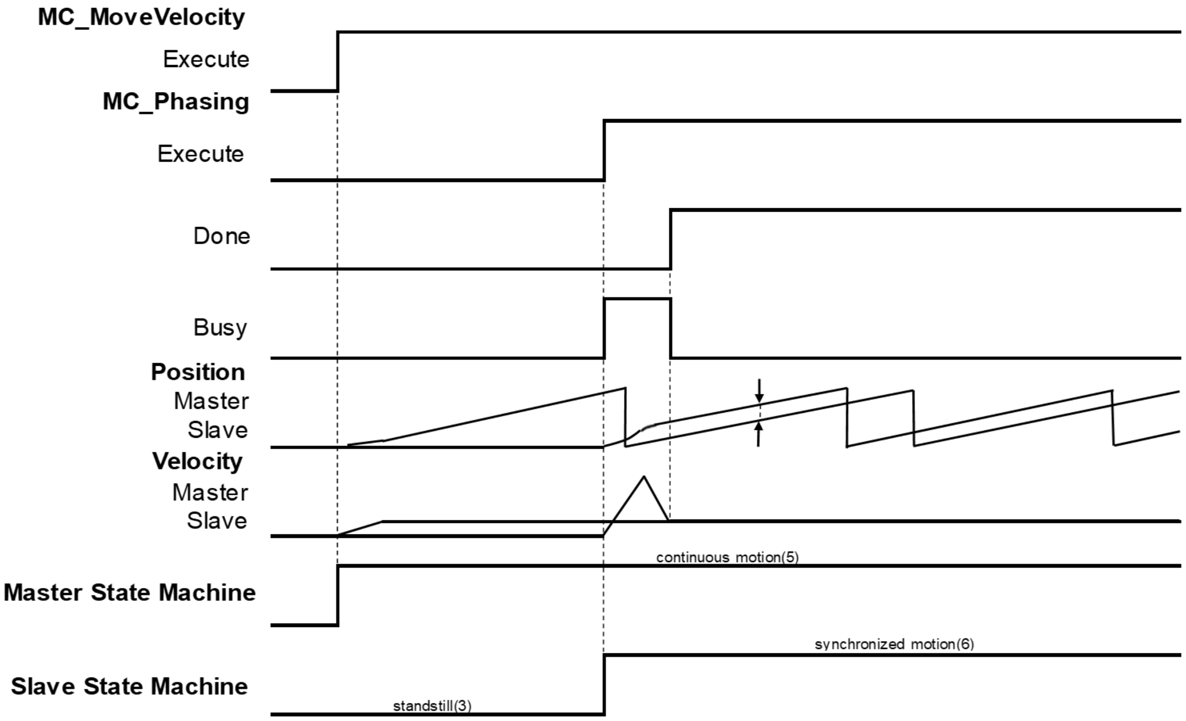


- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Master	Specifies the master axis number.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
Slave	Specifies the slave axis number.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

* **Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - When *Execute* shifts to True and the master-slave axis relation is established, the slave axis will shift the phase by planning a smooth curve. In case of a positive PhaseShift value, the slave axis is behind the master axis. Conversely, the slave axis is ahead of the master axis when the value is negative.
  - The position of master axis remains unchanged while MC_Phasing acts on the slave axis

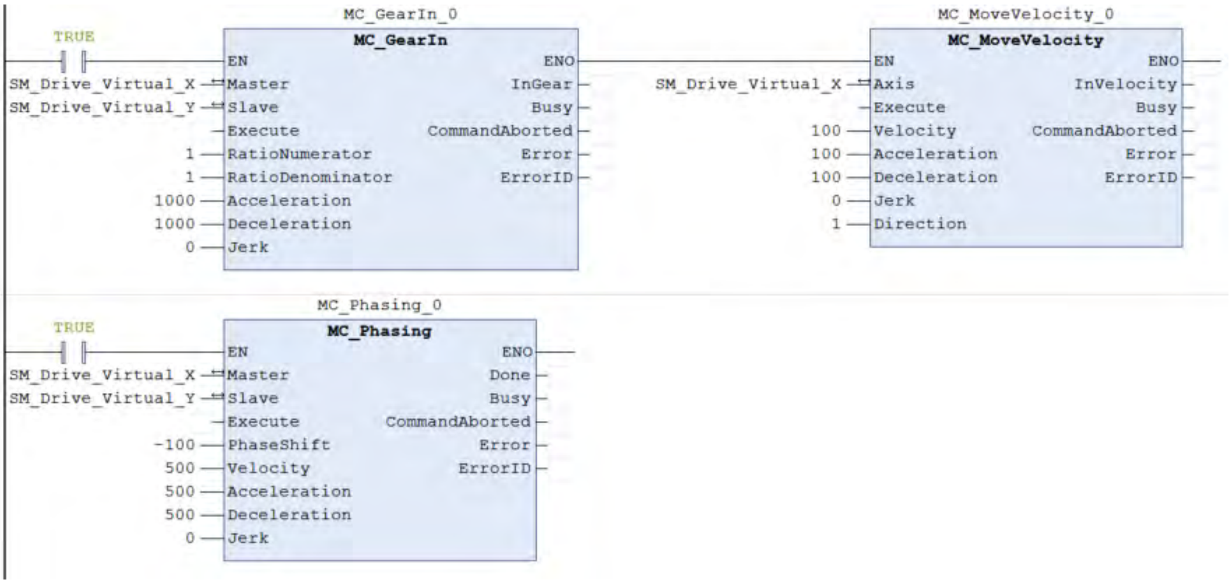


- MC_Phasing can be used when the state is not under Synchronized motion.
    - When MC_Phasing is executed, the state of slave axis will remain as Synchronized motion.
    - When executes MC_Phasing before establishing gear relationship between the master and slave axis, the slave axis will be directly synchronized with the master axis and both move based on the gear raio which is 1:1.
    - When the slave axis executes MC_Phasing, it can be aborted by other single-axis function blocks and the the synchronous relationship will be disconnected.

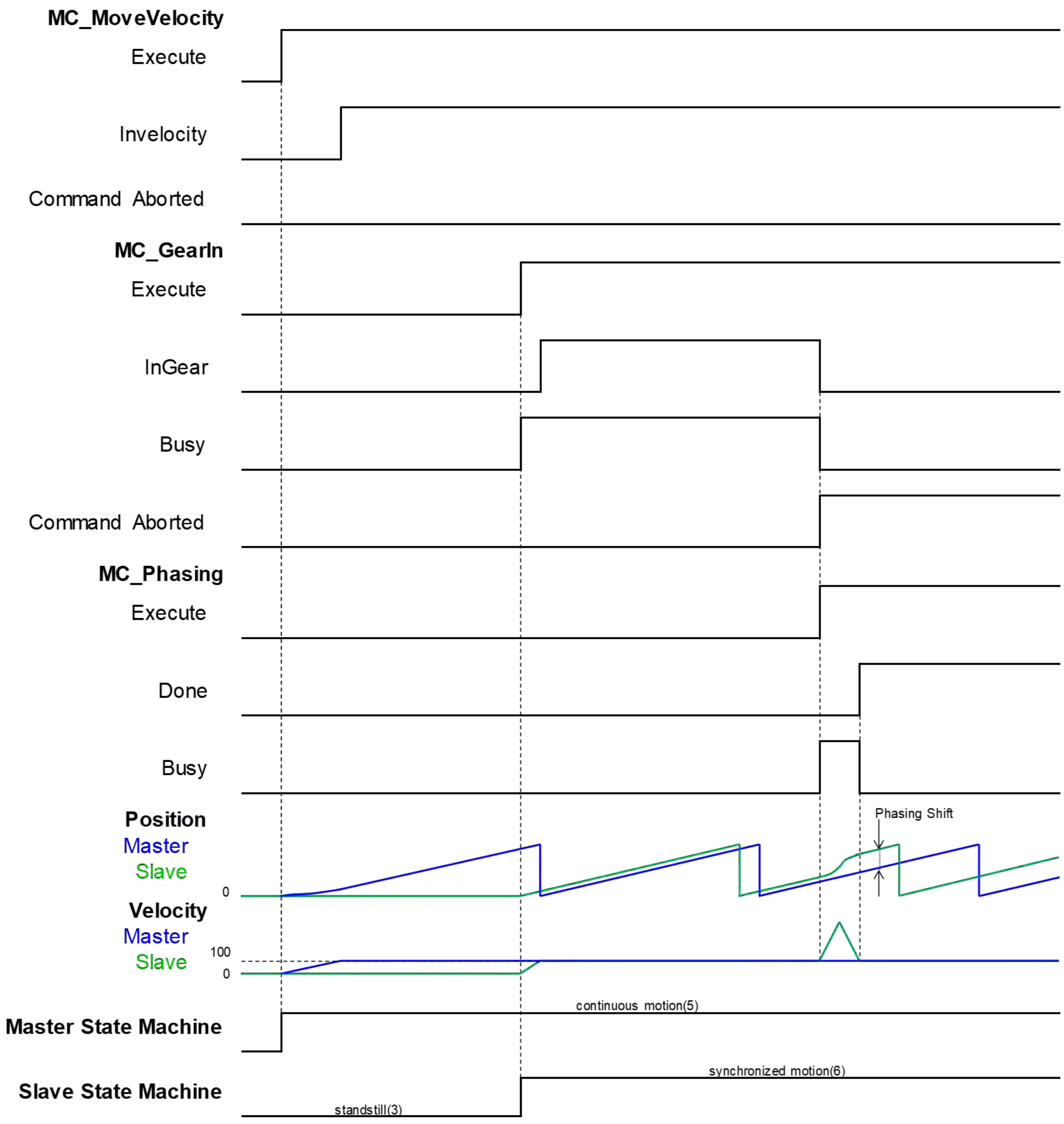
- **Troubleshooting**
    - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

  - The following example describes the corresponding motion state and path throughout the gear operation.



- **Timing Diagram**



- Execute MC_MoveVelocity to make the master axis run at a constant speed, then execute MC_GearIn to establish gear relationship between the master and slave axis.
- When *Execute* of MC_Phasing shifts to True, the relationship between the two axes will be disconnected. MC_Phasing shifts the phase of the slave axis by the specified PhaseShift value.
- When the slave axis reaches the specified value, Done of MC_Phasing shifts to True and the Busy output is reset.

- **Supported Products**
  - AX-308E、AX-8

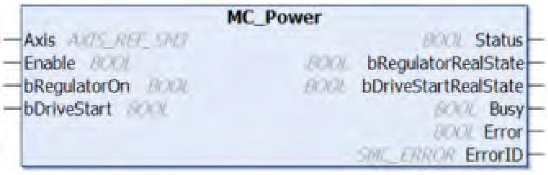
2.2 Administrative Motion Control Instructions

Administrative motion control instructions refer to the actions of configuring corresponding setting and retrieving relating information made for drivers, which would not cause actual displacement of motors. The function blocks used in this chapter are from the library “SM3_Basic” and able to operate synchronously with drives. As a result, synchronous axis type should be selected in axis settings. For more details about configuration related to synchronous axes, please refer to section 7.4 in AX-3 Series Operational Manual.

2.2.1 MC_Power

MC_Power enables or disables the specific axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Power	 MC_Power
		Axis AXIS_REF_SM3
		Enable BOOL
		bRegulatorOn BOOL
		bDriveStart BOOL
		BOOL Status
		BOOL bRegulatorRealState
		BOOL bDriveStartRealState
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_Power_instance (
			Axis :=,
			Enable :=,
			bRegulatorOn:=,
			bDriveStart :=,
			Status =>,
			bRegulatorRealState =>,
			bDriveStartRealState =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
bRegulatorOn	Enables the power	BOOL	True/False(False)	Only when *Enable*=True.
bDriveStart	Controls the QuickStop mechanism.	BOOL	True/False(False)	Only when *Enable*=True.

- **Outputs**

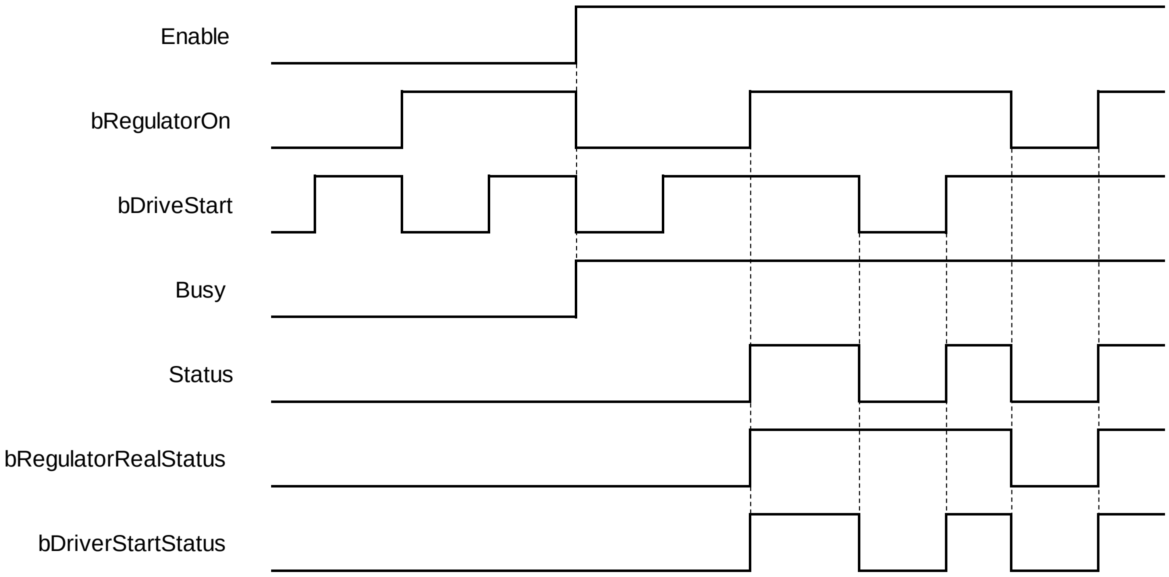
Name	Function	Data Type	Output Range (Default Value)
Status	The specific axis is ready to be moved by the function blocks.	BOOL	True/False(False)
bRegulatorRealState	The power is turned ON.	BOOL	True/False(False)
bDriveStartRealState	Quick stop function is applicable on the device.	BOOL	True/False(False)
Busy	Function block is operating.	BOOL	True/False(False)
Error	Errors occur in function block.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Status	● When Enable is true upon detection of rising edge for both *bRegulatorRealState* and *bDriveStartRealState*.	● When Enable is true, as well as *bRegulatorRealState* or *bDriveStartRealState* shifting to False. ● When Error shifts to True.
bRegulatorRealState	● When Enable is true upon detection of rising edge for *bRegulatorRealState.*	● When Enable is true, as well as bRegulatorRealState being falling edge. ● When Error shifts to True.
bDriveStartRealState	● When Enable is true, as well as both *bRegulatorRealState* and *bDriveStartRealState* being True.	● When Enable is true, as well as bRegulatorRealState or bDriveStartRealState shifting to False. ● When Error shifts to True.
Busy	● When Enable shifts to true	● When Enable shifts to False. ● When Error shifts to True.
Error	● When errors occur in the executing conditions or the input values.	● When errors are moved.
ErrorID		

- **Timing Diagram**

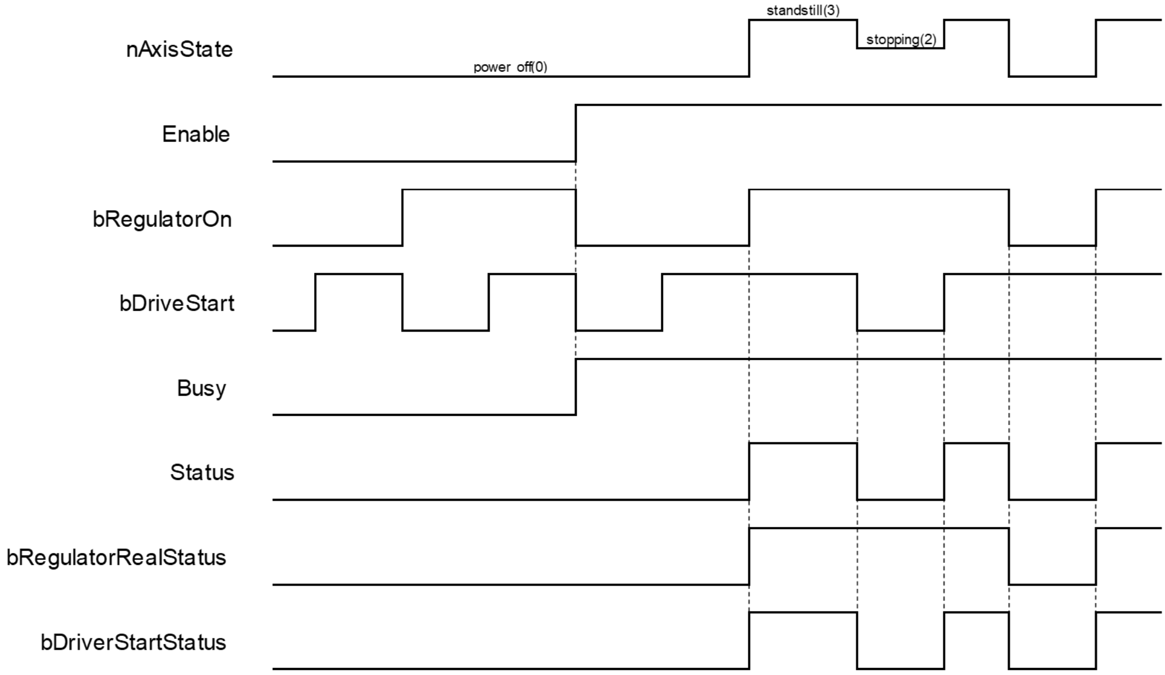


- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Enable* shifts to True.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

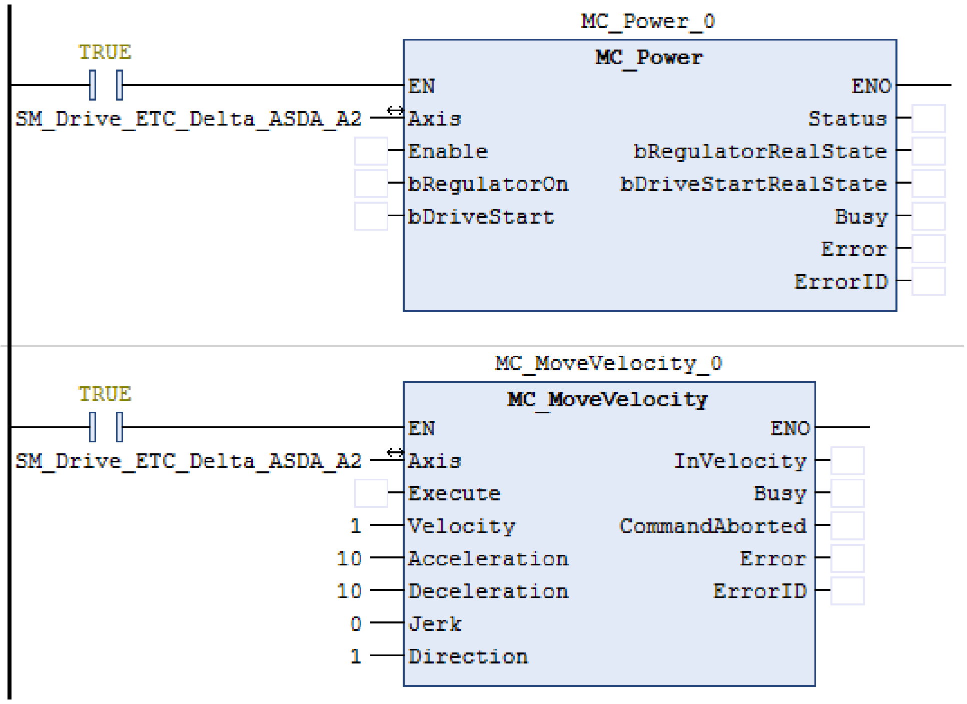
- **Function**
  - When Enable is True, bRegulatorOn and bDriveStart are effective.
  - When Enable, bRegulatorOn and bDriveStart are True, Status shifts to True and nAxisState shifts to standstill.
  - When Enable and bRegulatorOn are True, set DriveStart to be False which nAxisState(axis status) shifts to Stopping。
  - When Enable and bDriveStart are True, set RegulatorOn to be False which nAxisState(axis status) directly shifts to Disabled.
  - Timing Diagram



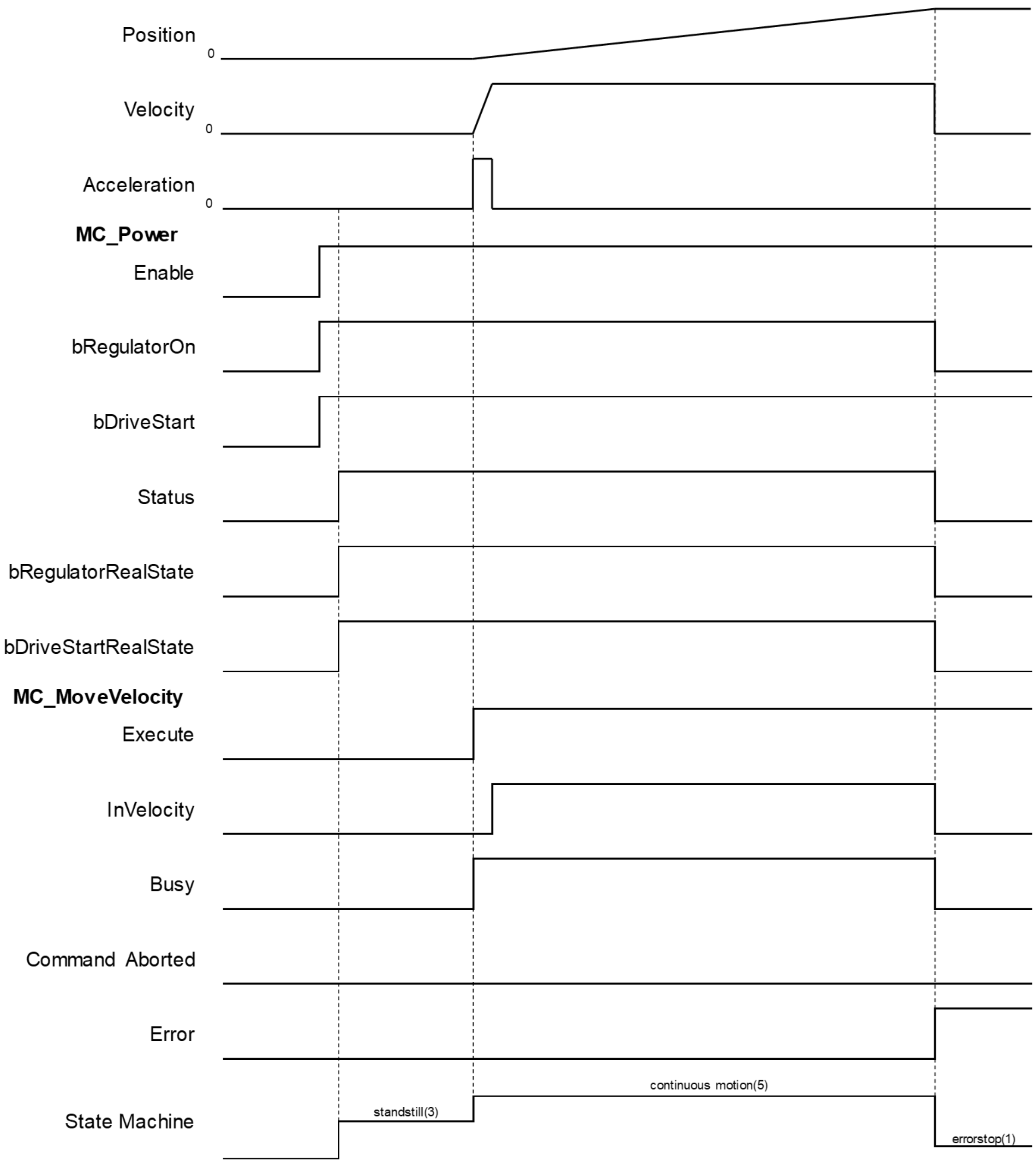
- **Troubleshooting**
  - If an error occurs while executing the instruction or the axis is in Errorstop state, Error shifts to True and the axis will be decelerated to a stop. You can refer to ErrorID (Error Code) to address the problem.

-

- **Example:**
  - Example1: The following example explains the movement of a moving axis when *bRegulatorOn* of MC_Power shifts to False.



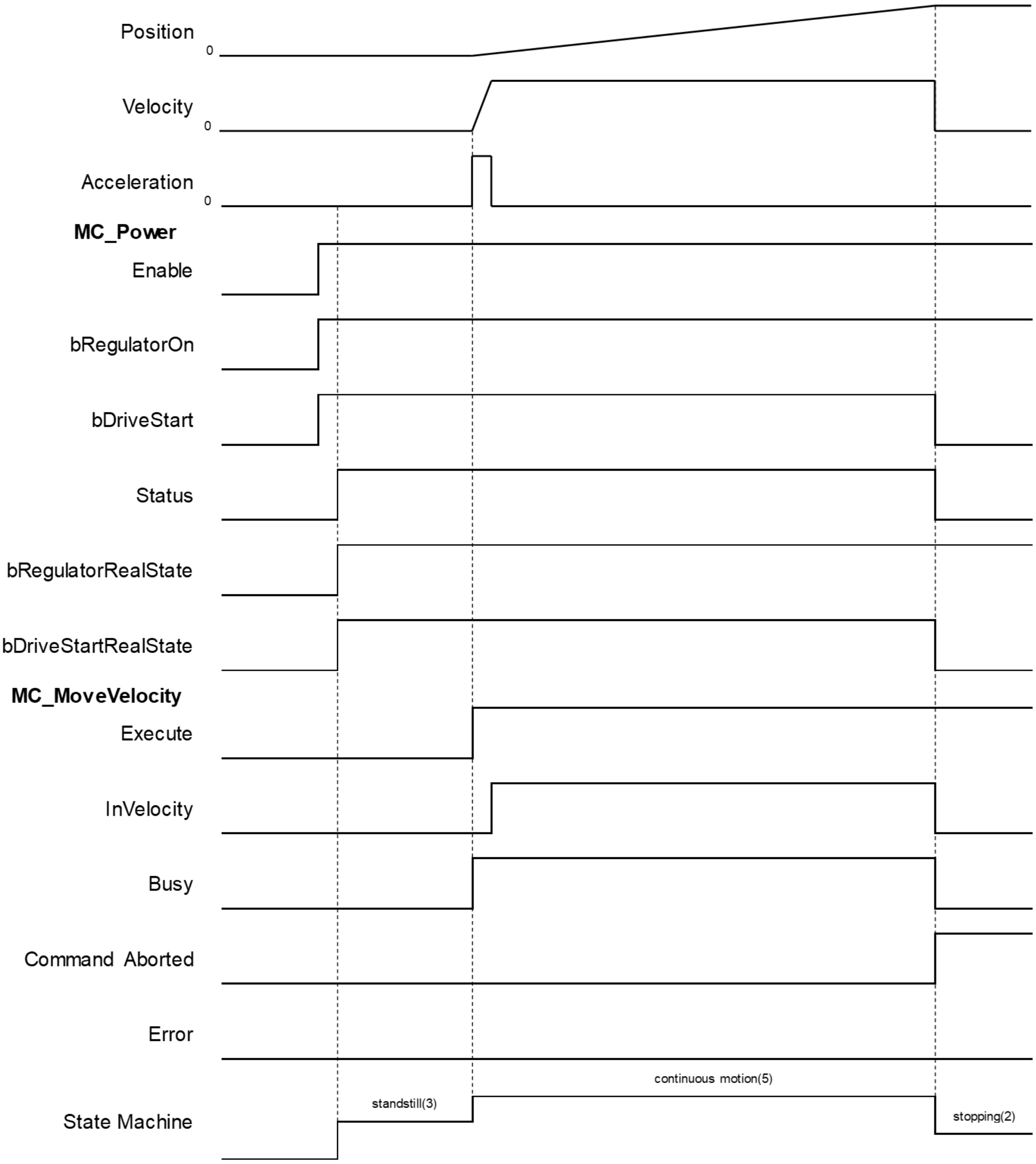
- Timing Diagram



- Shift *bRegulatorOn* of MC_Power from True to False while the axis is moving so as to stop the axis immediately.
- At the same time an error of SMC_REGULATOR_OR_START_NOT_SET(20) occurs in MC_MoveVelocity and the axis state directly switches from continuous_motion to errorstop.

- Example2: Continue with the example 1, the following example explains the movement of a moving axis when *bDriveStart* of MC_Powers shifts to False.

- Timing Diagram



- ◆ Shift *bDriveStart* of MC_Power from True to False while the axis is moving. Such action would stop the axis immediately.

- ◆ *CommandAbort* shifts to True and aborts MC_MoveVelocity, while ther axis state swiches from continuous_motion to stopping.

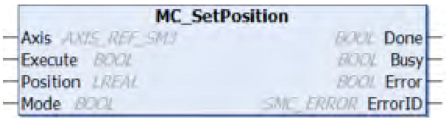
- **Supported Products**

- AX-308E、AX-8

2.2.2 MC_SetPosition

MC_SetPositionn changes the current position by shifting the coordinates of an axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_SetPosition	 MC_SetPosition
		Axis AXIS_REF_SM3
		Execute BOOL
		Position LREAL
		Mode BOOL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_SetPosition_instance(
			Axis :=,
			Execute :=,
			Position :=,
			Mode :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (True)	-
Position	Axis position (User-defined unit)	LREAL	Positive number, negative number or 0(0)	When Execute shifts to True and Busy is False.
Mode	Specify relative position (True) or absolute position(False)	BOOL	True/False (False)	When Execute shifts to True and Busy is False.

- **Outputs**

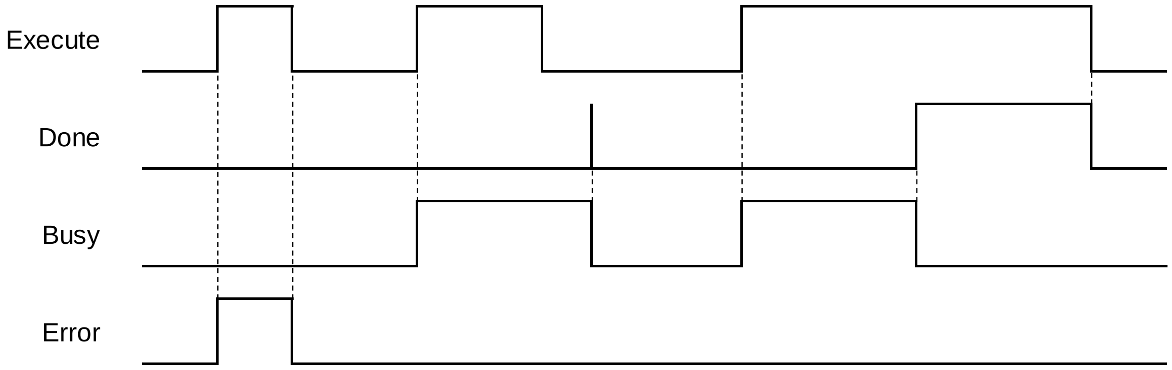
Name	Function	Data Type	Output Range (Default Value)
Done	True when finishes coordinate modification.	BOOL	True/False(False)
Busy	True when the instruction is executing.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the modification to coordinate is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When *Execute* is rising edge triggered.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

- When configure the parameters of axis position via MC_Position, there's no displacement made by axis itself but instead only the coordinate system moves.

- To avoid possible position jumps occur to the slave axis, you should avoid executing MC_SetPosition to the synchronized master axis, or discontinuous jumps in velocity would exists in the slave axis.

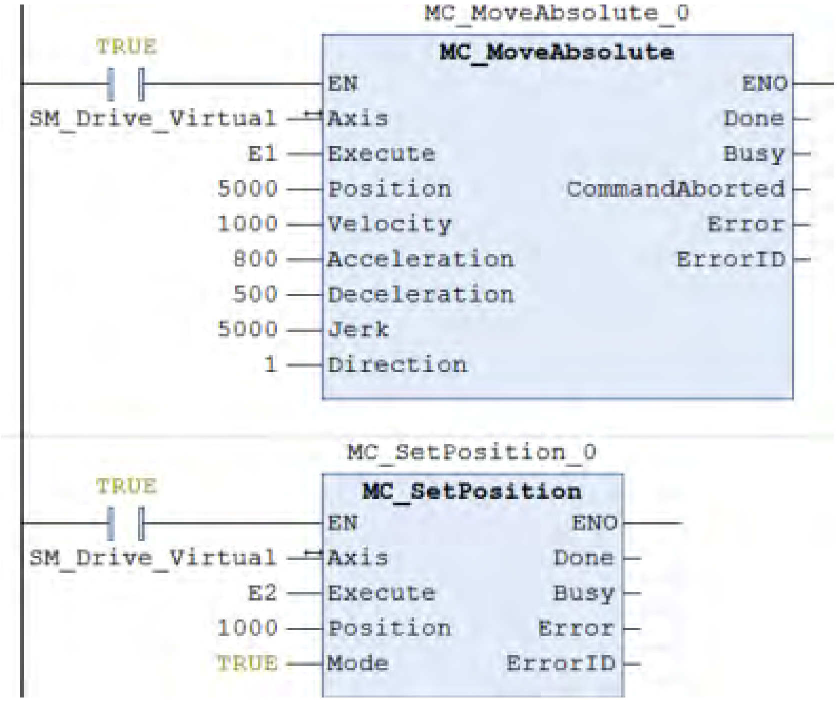
- The value of Position would be added directly to the current position under relative mode, which would be the new location coordinate. For the absolute position, the value of Position would be set to the current location coordinate.

- **Troubleshooting**

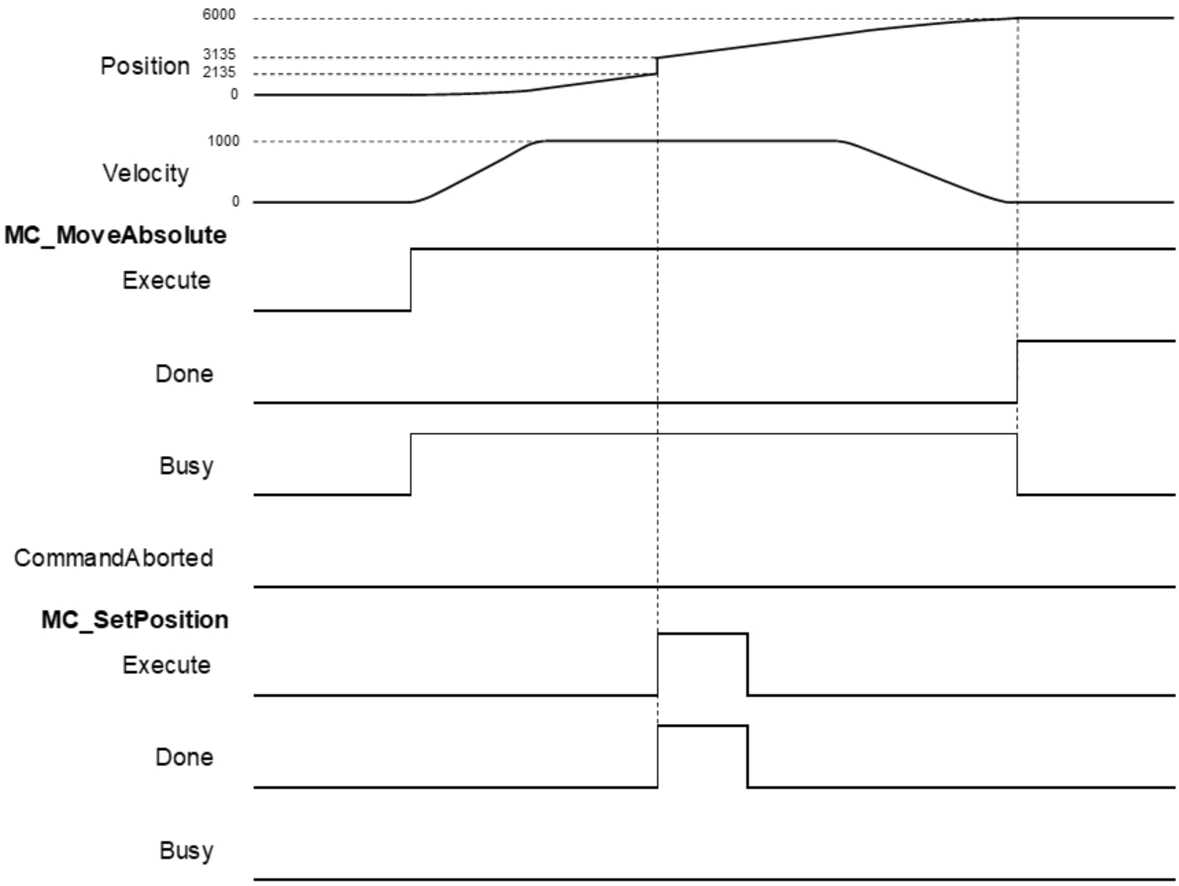
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

  - Example1: Demonstrate the condition of executing MC_MoveAbsolute while using MC_SetPosition to change the coordinate system under relative mode.

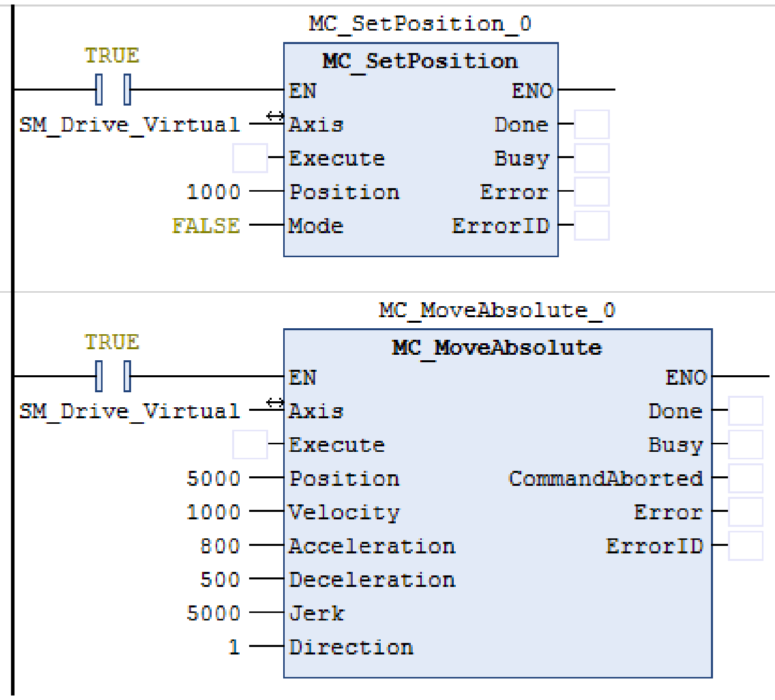


- Timing Diagram

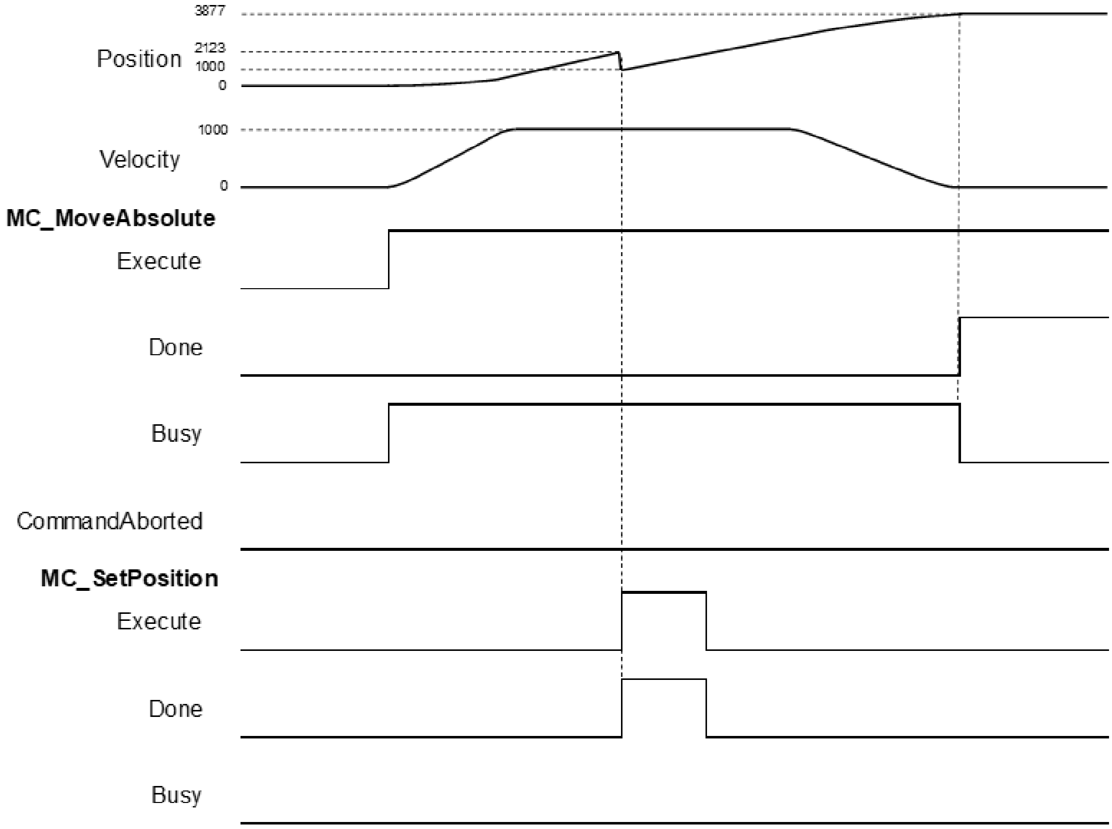


- After MC_MoveAbsolute is executed, use MC_SetPosition to define a new coordinate system under relative mode.
- When detect a rising edge of Execute of MC_SetPosition, the new location of the axis would be 3135 after adjusting the coordinate system, which the axis was originally located at 2135.
- Done is rising edge when the execution of MC_MoveAbsolute is completed and the current position would be at 6000. At this time, MC_MoveAbsolute still moves to 5000 on the old coordinate system, which would become 6000 after the coordinate adjustiment.

- Example2: Demonstrate the condition of executing MC_MoveAbsolute while using MC_SetPosition to change the coordinate system under absolue mode.

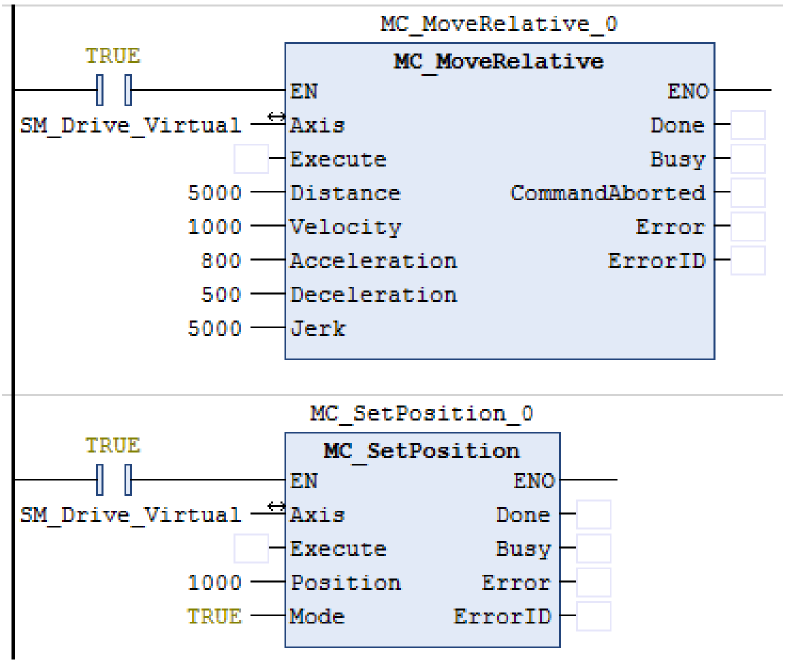


- Timing Diagram



- After MC_MoveAbsolute is executed, use MC_SetPosition to define a new coordinate system under absolute mode.
- When detect a rising edge of Execute of MC_SetPosition, the new location of the axis would be 1000 after adjusting the coordinate system by moving a distance of 1123, which the axis was originally located at 2123.
- Done is rising edge when the execution of MC_MoveAbsolute is completed and the current position would be at 3877(5000 - 1123). At this time, MC_MoveAbsolute still moves to 5000 on the old coordinate system, which would become 3877 after the coordinate adjustment.

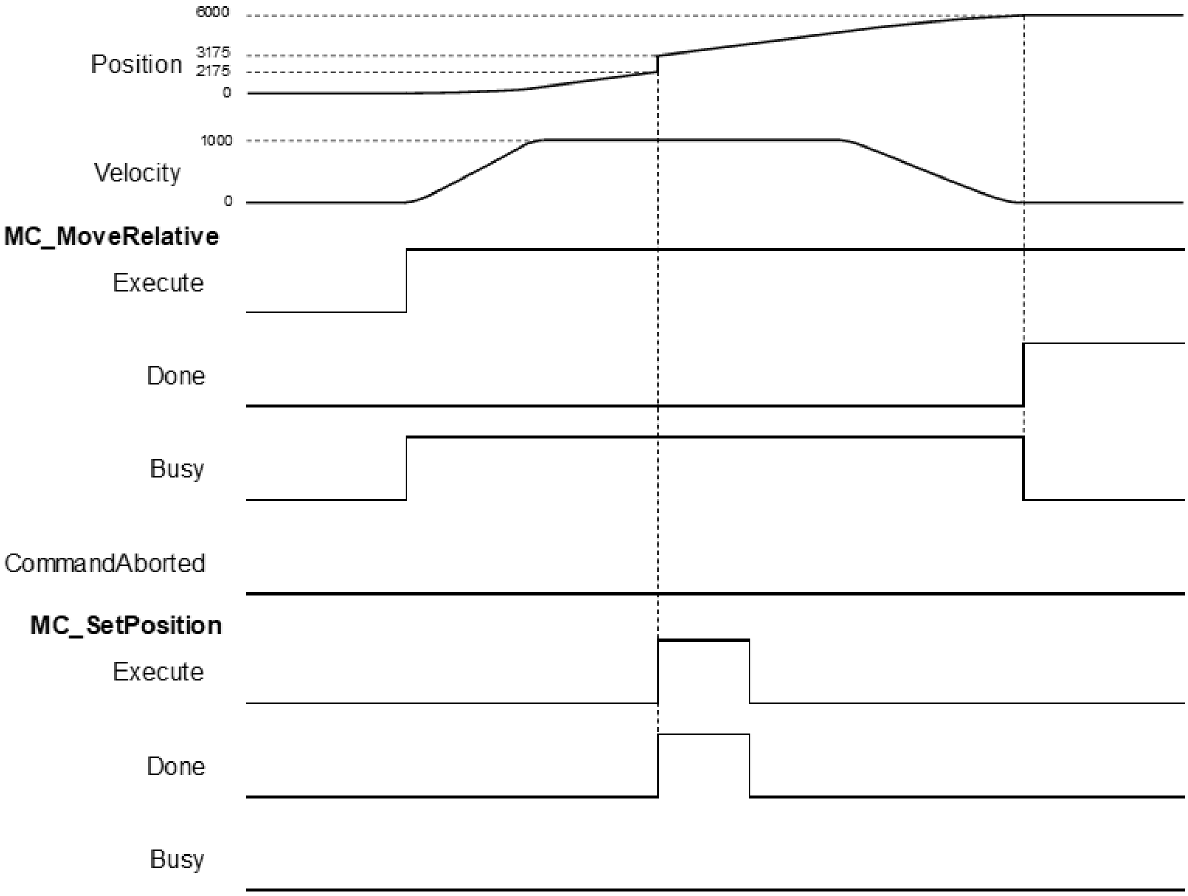
- Example3: Demonstrate the condition of executing MC_MoveRelative while using MC_SetPosition to change the coordinate system under relative mode.



MC_MoveRelative_0
MC_MoveRelative
TRUE — EN; ENO
SM_Drive_Virtual — Axis; Done
Execute; Busy
5000 — Distance; CommandAborted
1000 — Velocity; Error
800 — Acceleration; ErrorID
500 — Deceleration
5000 — Jerk

MC_SetPosition_0
MC_SetPosition
TRUE — EN; ENO
SM_Drive_Virtual — Axis; Done
Execute; Busy
1000 — Position; Error
TRUE — Mode; ErrorID

- Timing Diagram



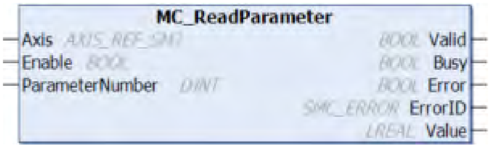
- After MC_MoveRelative is executed, use MC_SetPosition to define a new coordinate system under relative mode.
- When detect a rising edge of Execute of MC_SetPosition, the new location of the axis would be 3175 after adjusting the coordinate system, which the axis was originally located at 2175.
- Done is rising edge when the execution of MC_MoveAbsolute is completed and the current position would be at 6000. At this time, MC_MoveAbsolute still moves to 5000 on the old coordinate system, which would become 6000 after the coordinate adjustiment.

- **Supported Products**
  - AX-308E、AX-8

2.2.3 MC_ReadParameter

MC_ReadParameter reads a value of a specific axis parameter.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadParameter	 MC_ReadParameter
		Axis AXIS_REF_SM3
		Enable BOOL
		ParameterNumber DINT
		BOOL Valid
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		LREAL Value
			MC_ReadParameter_instance (
			Axis :=,
			Enable :=,
			ParameterNumber :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			Value =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False (False)	-
ParameterNumber	Number of the parameter to be read.	DINT	Positive number, negative number or 0(0)	When *Enable* is detected to be rising edge.

- **Outputs**

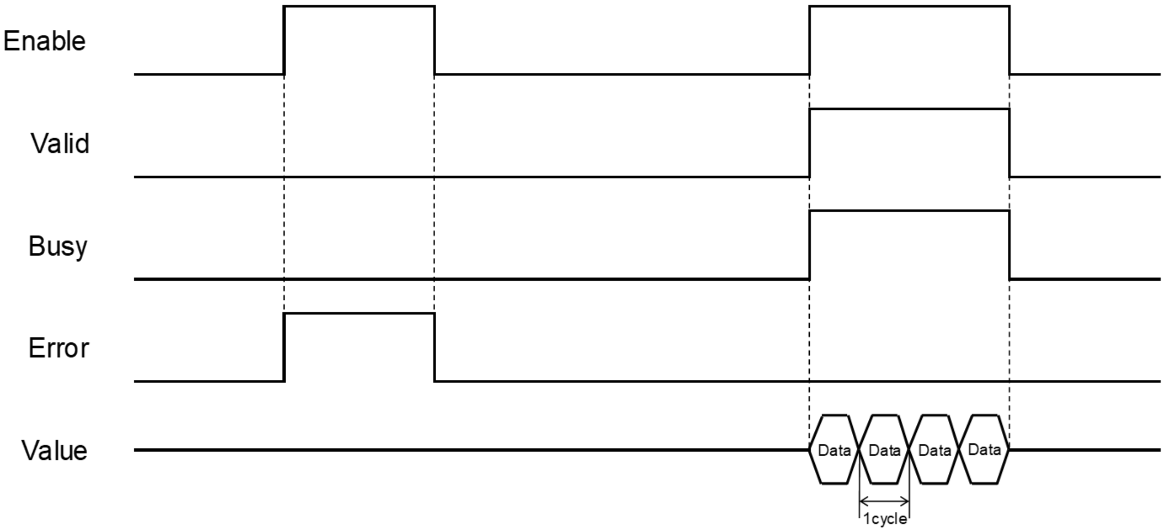
Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Error codes.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
Value	Value of the parameter to read.	LREAL*2	Positive number, negative number or 0(0)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
Value	● When *Valid* is True and there're ongoing updates.	● When Valid is False and stop updating.

- **Timing Diagram**



***Note:** Data = parameter's value. 1 cycle = one task period

- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Enable* is rising edge.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

  ■ How to use MC_ReadParameter to read the parameter values of desired object with its parameter number in EtherCAT Object Dictionary:

  ◆ Use SHL instruction to move the data length of desired object to the left for 24 bits.

  ◆ Use SHL instruction to move the index of desired object to the left for 8 bits.

  ◆ The input ParameterNumber must contains the data length the index and the subindex. Please refer to the following formula:

  ParameterNumber := - DWORD_TO_DINT(SHL(TO_DWORD(data length of object dictionary)，24) + SHL(TO_DWORD(index of object dictionary)，8) + subindex of object dictionary

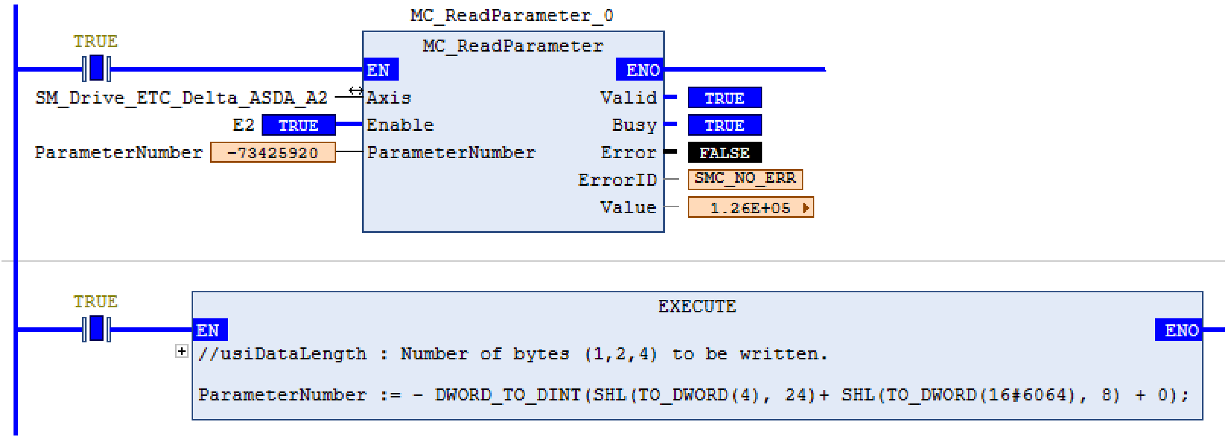
  ■ To read axis parameters, you'll need to enter the parameter number of AXIS_REF_SM3(FB) to ParameterNumber input.

- **Troubleshooting**

  ■ If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

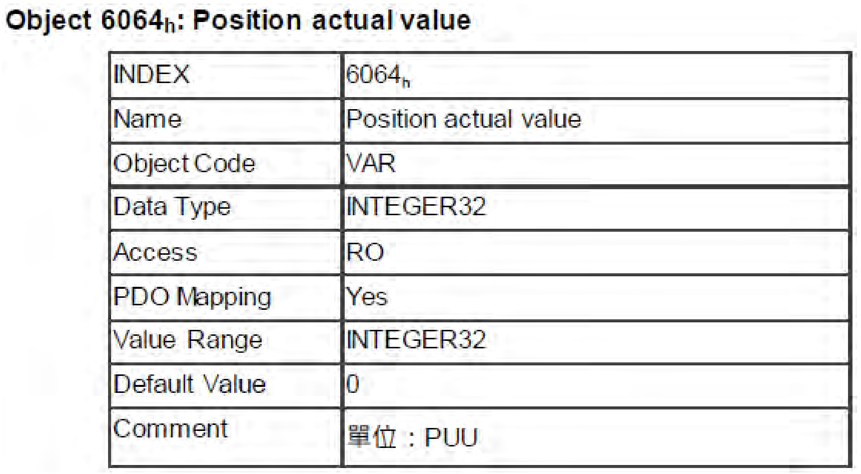
  ■ This example demonstrates how to use MC_ReadParameter to read the value of object 0x6064(current position returned by motor) in the drive.



```
//usiDataLength : Number of bytes (1,2,4) to be written.
ParameterNumber := - DWORD_TO_DINT(SHL(TO_DWORD(4), 24)+ SHL(TO_DWORD(16#6064), 8) + 0);
```

  ■ Input the data length, index, and subindex of the object to the above formula and you'll get the ParameterNumber, which should be entered to the ParameterNumber input. After that, every time when the FB MC_ReadParameter is executed, it will visit the object dictionary specified by the drive and return the values.

- The following figure shows the information related to the parameters of object 0x6064.



Object 6064$_h$: Position actual value

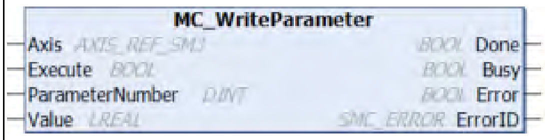
INDEX	6064$_h$
Name	Position actual value
Object Code	VAR
Data Type	INTEGER32
Access	RO
PDO Mapping	Yes
Value Range	INTEGER32
Default Value	0
Comment	單位：PUU

- **Supported Products**
  - AX-308E、AX-8

2.2.4 MC_WriteParameter

MC_WriteParameter writes a value to a specific parameter.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_WriteParameter	 MC_WriteParameter
		Axis AXIS_REF_SM3
		Execute BOOL
		ParameterNumber DINT
		Value LREAL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_WriteParameter_instance(
			Axis :=,
			Execute :=,
			ParameterNumber :=,
			Value :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	Execute the function block.	BOOL	True/False (False)	-
ParameterNumber	Number of the parameter to be written.	DINT	Positive number, negative number or 0(0)	When Execute shifts to True and Busy is False.
Value	Value to be written to the parameter.	LREAL	Positive number, negative number or 0(0)	When Execute shifts to True and Busy is False.

- **Outputs**

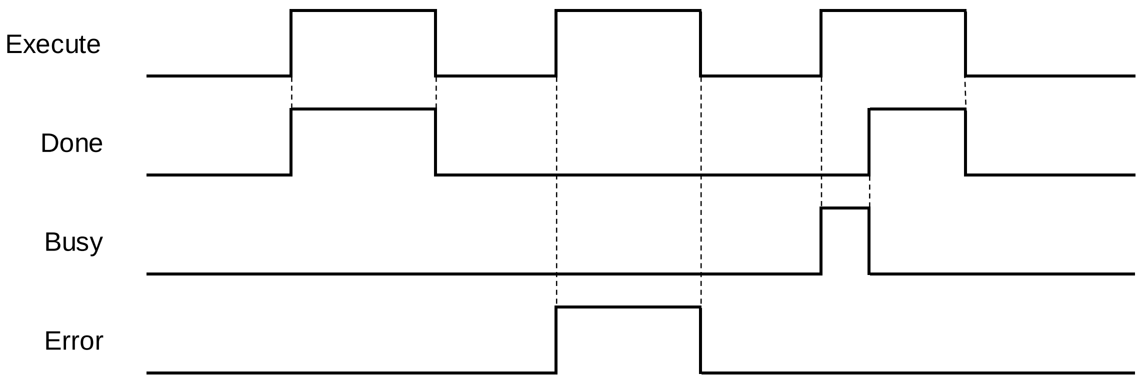
Name	Function	Data Type	Output Range (Default Value)
Done	True if the value is written successfully.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the value is written successfully.	● When *Execute* shifts from True to False.
Busy	● When *Execute* is rising edge triggered. ● When the value is being written to the parameter.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - How to use MC_ReadParameter to read the parameter values of desired object with its parameter number in EtherCAT Object Dictionary:
    - Use SHL instruction to move the data length of desired object to the left for 24 bits.
    - Use SHL instruction to move the index of desired object to the left for 8 bits.
    - The input ParameterNumber must contains the data length the index and the subindex. Please refer to the following formula:

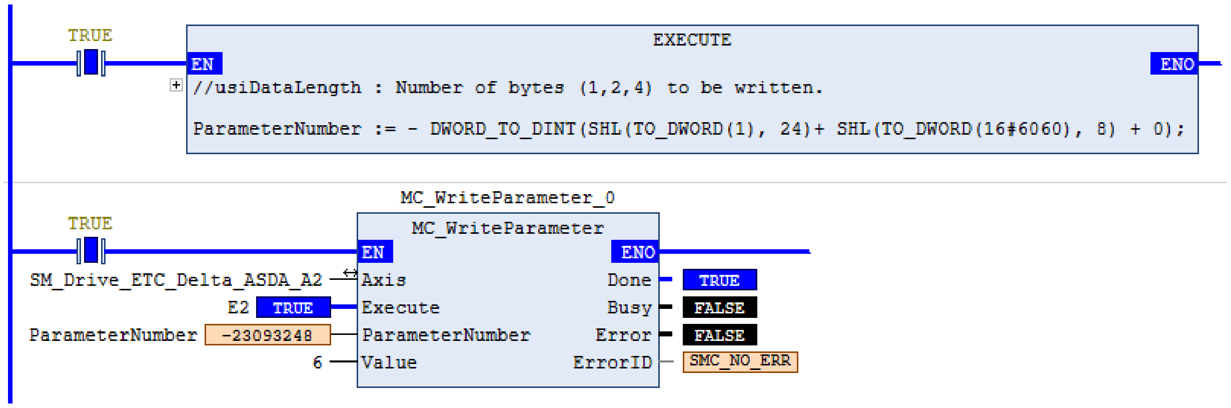
  ParameterNumber := - DWORD_TO_DINT(SHL(TO_DWORD(data length of object dictionary)，24) + SHL(TO_DWORD(index of object dictionary)，8) + subindex of object dictionary

  - To write the value to the parameter, you'll need to input the parameter number of AXIS_REF_SM3(FB) to ParameterNumber.
  - Write parameter values to the input fSetPosition by using MC_WriteParameter while the axis is moving. The value of fSetPosition is changed for only one task cycle time in EtherCAT, then fSetPosition resumes its original planned motion curve to move.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**

■ This example demonstrates how to use MC_WriteParameter to write the value to the object 0x6060(operation mode) in the drive.

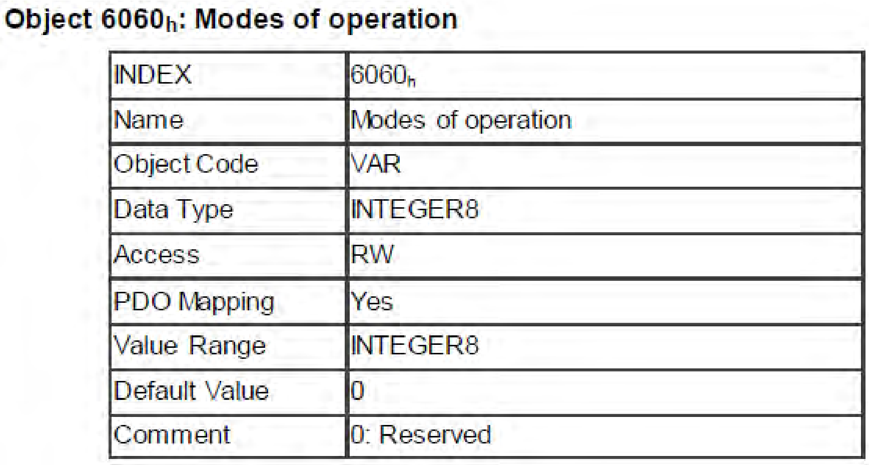


```
TRUE                EXECUTE
EN                                                                                  ENO
//usiDataLength : Number of bytes (1,2,4) to be written.
ParameterNumber := - DWORD_TO_DINT(SHL(TO_DWORD(1), 24)+ SHL(TO_DWORD(16#6060), 8) + 0);

                              MC_WriteParameter_0
TRUE                          MC_WriteParameter
                              EN                        ENO
SM_Drive_ETC_Delta_ASDA_A2 ↔  Axis                     Done  TRUE
E2 TRUE                       Execute                  Busy  FALSE
ParameterNumber -23093248     ParameterNumber         Error  FALSE
6                             Value                 ErrorID  SMC_NO_ERR
```

■ Input the data length, index, and subindex of the object to the above formula and you'll get the ParameterNumber, which should be entered to the ParameterNumber input. After the value is written to MC_WriteParameter succesfully, the control mode of the drive will change to 6.

■ The following figure shows the information related to the parameters of object 0x6060



Object 6060$_h$: Modes of operation

INDEX	6060$_h$
Name	Modes of operation
Object Code	VAR
Data Type	INTEGER8
Access	RW
PDO Mapping	Yes
Value Range	INTEGER8
Default Value	0
Comment	0: Reserved

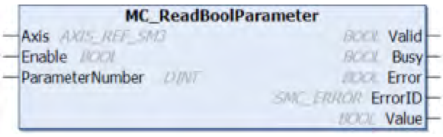
- **Supported Products**

■ AX-308E、AX-8

2.2.5 MC_ReadBoolParameter

MC_ReadBoolParameter reads the value of a specific Boolean parameter.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadBoolParameter	 MC_ReadBoolParameter
		Axis AXIS_REF_SM3
		Enable BOOL
		ParameterNumber DINT
		BOOL Valid
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		BOOL Value
			MC_ReadBoolParameter_instance(
			Axis :=,
			Enable :=,
			ParameterNumber :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			Value =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False(False)	-
ParameterNumber	Number of the specific Boolean parameter	DINT	Positive number, negative number or 0(0)	When *Enable* shifts from False to True.

- **Outputs**

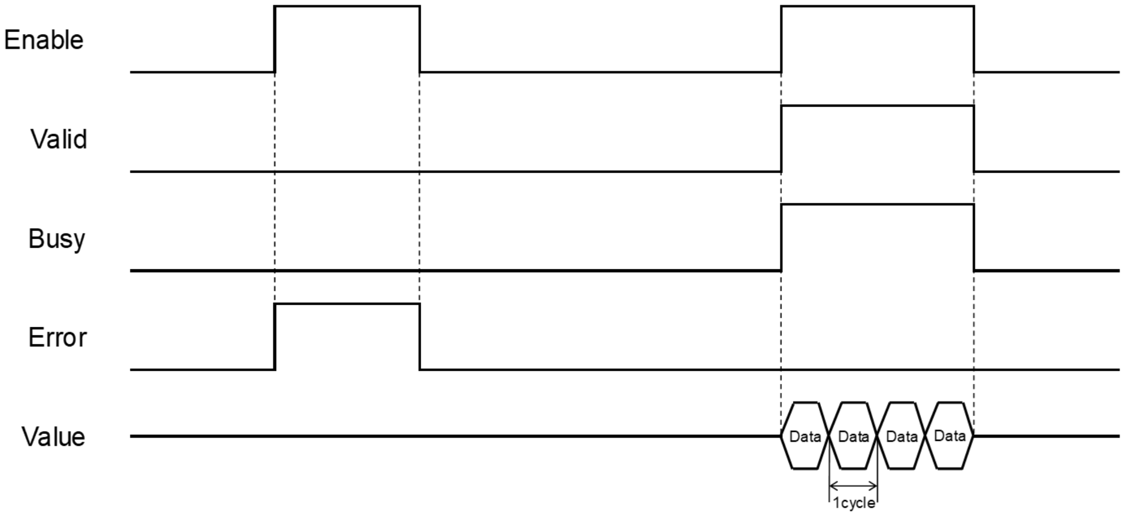
Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Error codes.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
Value	Value of the parameter to read.	BOOL	True/False(False)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
Value	● When *Valid* is True and there're ongoing updates.	● When Valid is False and stop updating.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Enable* shifts to True.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

  - How to use MC_ReadBoolParameter to read the parameter values of desired object with its parameter number in EtherCAT Object Dictionary:

    - Use SHL instruction to move the data length of desired object to the left for 24 bits.
    - Use SHL instruction to move the index of desired object to the left for 8 bits.

- The input ParameterNumber must contains the data length the index and the subindex. Please refer to the following formula:

  ParameterNumber := - DWORD_TO_DINT(SHL(TO_DWORD(data length of object dictionary)，24) + SHL(TO_DWORD(index of object dictionary)，8) + subindex of object dictionary

- For operation example, please refer to the example in MC_ReadParameter.
- To read axis parameters, you′ ll need to enter the parameter number of AXIS_REF_SM3(FB) to ParameterNumber input.

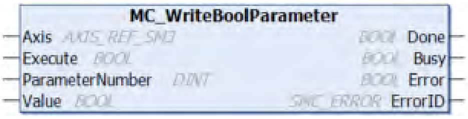
- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - AX-308E、AX-8

2.2.6 MC_WriteBoolParameter

MC_WriteBoolParameter writes a Boolean value to a specific parameter.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_WriteBool Parameter	 MC_WriteBoolParameter
		Axis AXIS_REF_SM3
		Execute BOOL
		ParameterNumber DINT
		Value BOOL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_WriteBoolParameter_instance(
			Axis :=,
			Execute :=,
			ParameterNumber :=,
			Value :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	Execute the function block.	BOOL	True/False (False)	-
ParameterNumber	Number of the parameter to be written.	DINT	Positive number, negative number or 0(0)	When Execute shifts to True and Busy is False.
Value	Boolean value to be written to the parameter.	BOOL	True/False (False)	When Execute shifts to True and Busy is False.

- **Outputs**

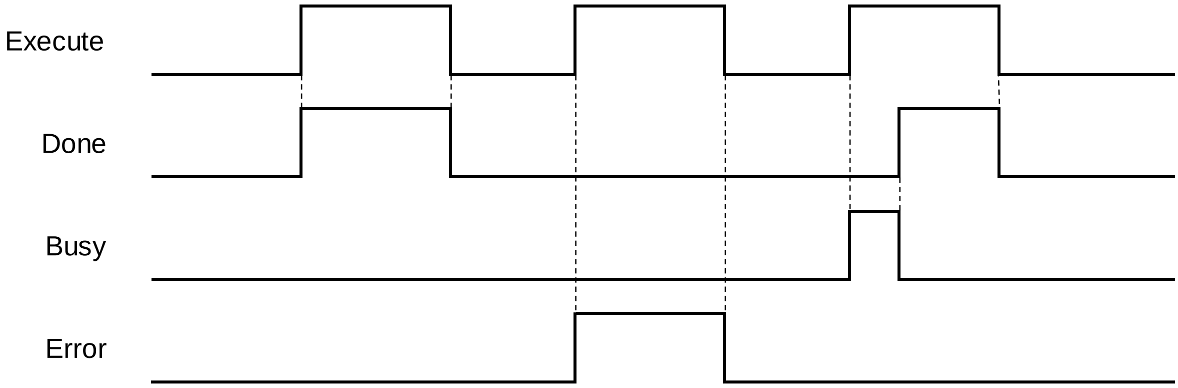
Name	Function	Data Type	Output Range (Default Value)
Done	True if the boolean value is written successfully.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

■ **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the value is written successfully.	● When *Execute* shifts from True to False.
Busy	● When *Execute* is rising edge triggered. ● When the value is being written to the parameter.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

■ **Timing Diagram**



● **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

● **Function**

■ How to use MC_WriteBoolParameter to write the parameter value to the desired object with its parameter number in EtherCAT Object Dictionary:

◆ Use SHL instruction to move the data length of desired object to the left for 24 bits.

◆ Use SHL instruction to move the index of desired object to the left for 8 bits.

The input ParameterNumber must contains the data length the index and the subindex. Please refer to the following formula:

ParameterNumber := - DWORD_TO_DINT(SHL(TO_DWORD(data length of object dictionary)，24) + SHL(TO_DWORD(index of object dictionary)，8) + subindex of object dictionary

- To write the value to the parameter, you’ll need to enter the parameter number of AXIS_REF_SM3(FB) to ParameterNumber input.
- For operation example, please refer to the example in MC_WriteParameter.

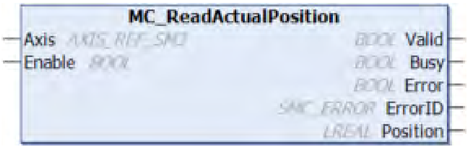
- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - AX-308E、AX-8

2.2.7 MC_ReadActualPosition

MC_ReadActualPosition reads the current axis position.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadActualPosition		MC_ReadActualPosition_instance(
			Axis :=,
			Enable :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			Position =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False (False)	-

- **Outputs**

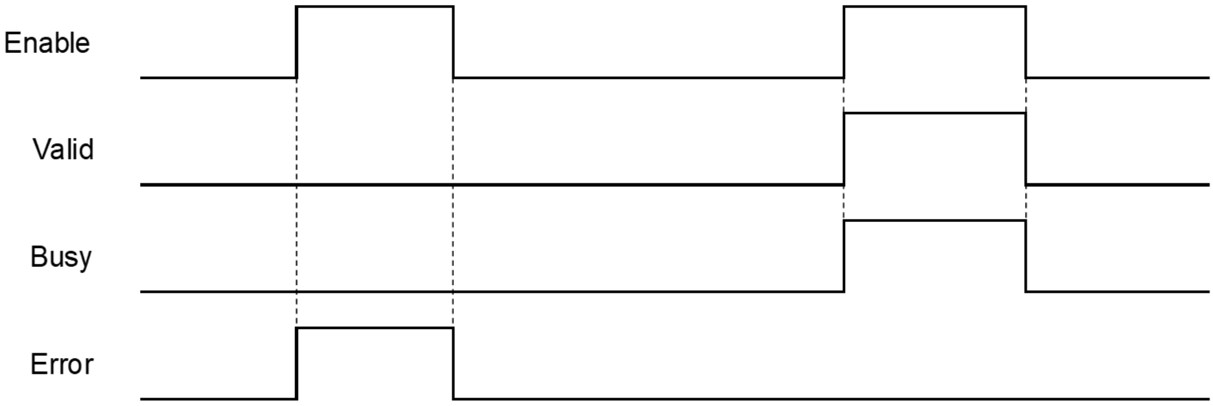
Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
Position	The current position of axis	LREAL	Positive number, negative number or 0(0)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
Position	● When *Valid* is True and there're ongoing updates.	● When Valid is False and stop updating.

- **Timing Diagram**



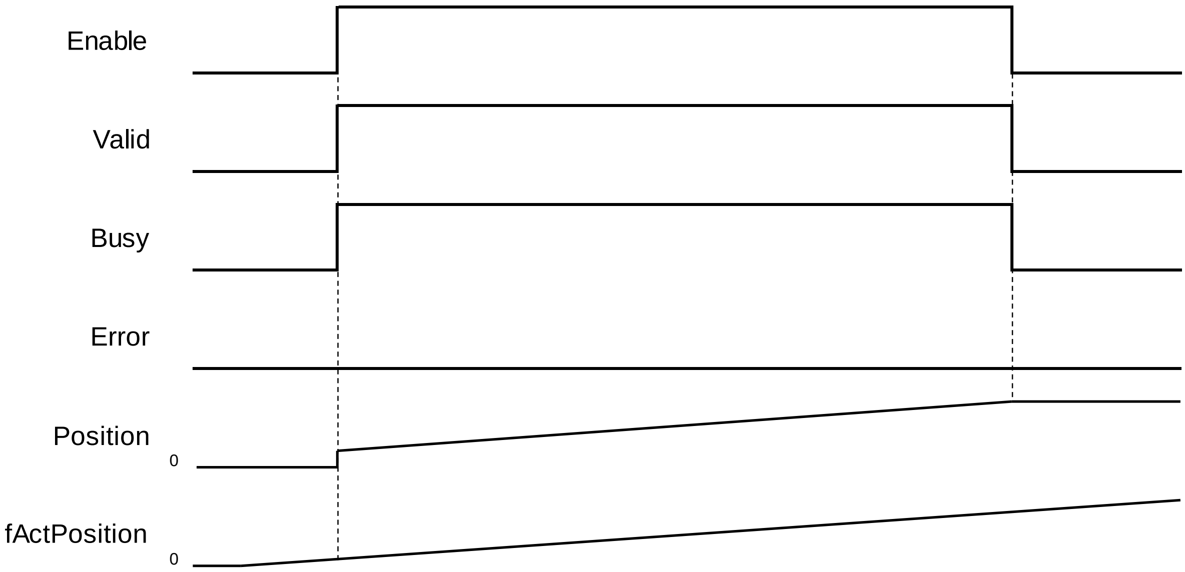
- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Enable shifts to True.

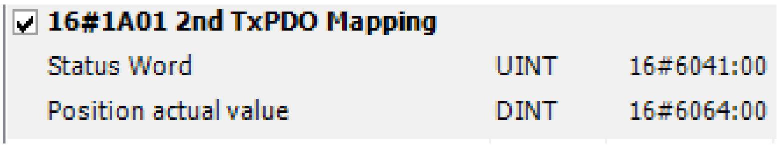
***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**

  - The value read from Position of MC_ReadActualPosition is value of fActPosition in AXIS_REF_SM3.



  - While using MC_ReadActualPosition, OD 0x6064(Actual position) must be mapping to TxPDO so as to read the actual position of the servo. If not, the values read by the function block would be 0.



☑ 16#1A01 2nd TxPDO Mapping

Status Word	UINT	16#6041:00
Position actual value	DINT	16#6064:00

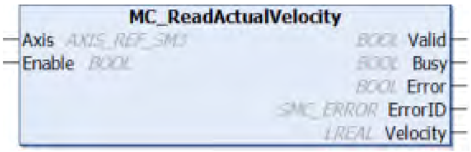
- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - AX-308E、AX-8

2.2.8 MC_ReadActualVelocity

MC_ReadActualVelocity reads the actual axis velocity value.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadActual Velocity	 MC_ReadActualVelocity
		Axis
		AXIS_REF_SM3
		Enable
		BOOL
		BOOL Valid
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		LREAL Velocity
			MC_ReadActualVelocity_instance(
			Axis :=,
			Enable :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			Velocity =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False (False)	-

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
Velocity	The current velocity of axis.	LREAL	Positive number, negative number or 0(0)

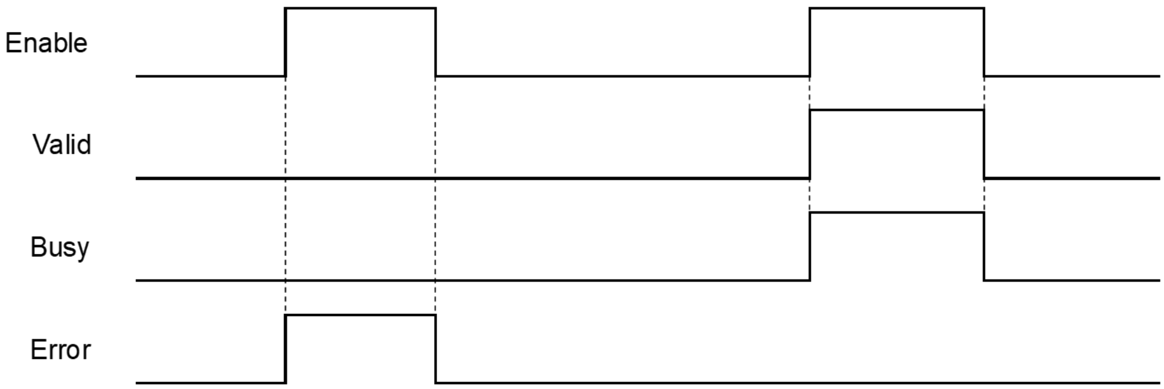
***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

Name	Timimg for Shifting to True	Timimg for Shifting to False
Velocity	● When *Valid* is True and there're ongoing updates.	● When Valid is False and stop updating.

- **Timing Diagram**

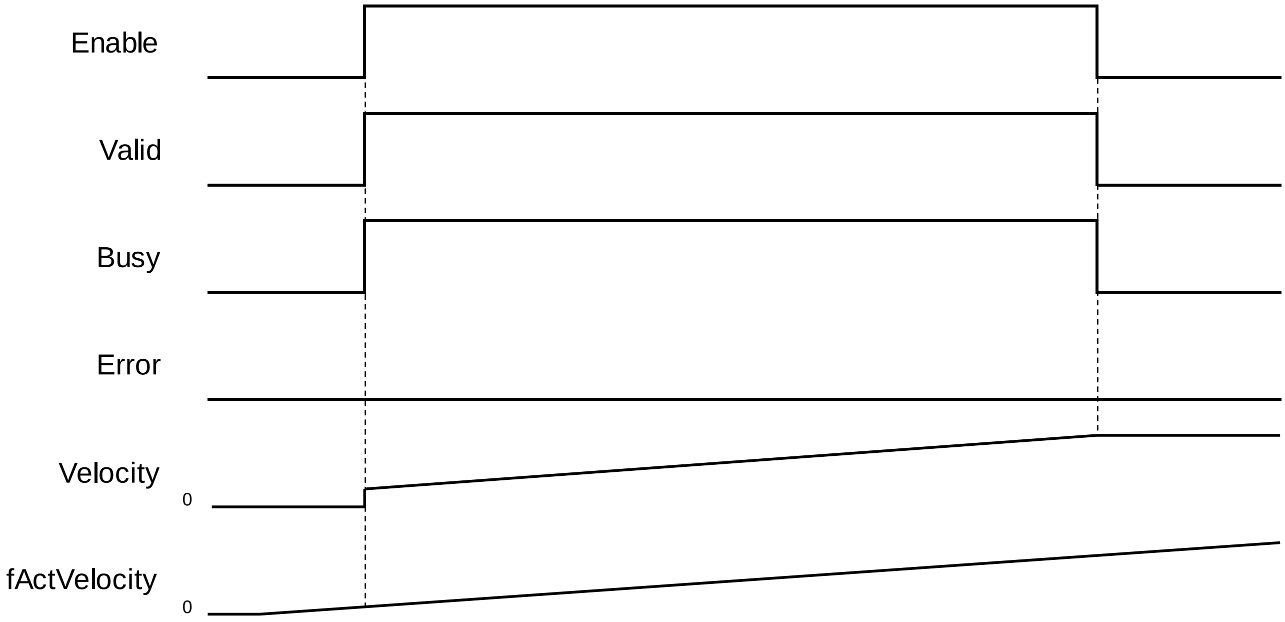


- **In-Outs**

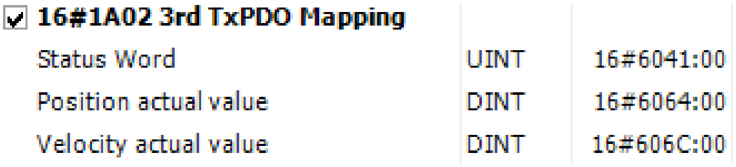
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Enable shifts to True.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - ■ The velocity value read by MC_ReadActualVelocity is the value of fActVelocity in AXIS_REF_SM3.



  - ■ While using MC_ReadActualVelocity, OD 0x606C (Actual velocity) must be mapping to TxPDO so as to read the actual velocity of the servo.



☑ **16#1A02 3rd TxPDO Mapping**

Status Word	UINT	16#6041:00
Position actual value	DINT	16#6064:00
Velocity actual value	DINT	16#606C:00

  - ■ If 0x606C is not mapping to TxPDO, the actual velocity of the servo would be calculated based on OD 0x6064(Actual position).

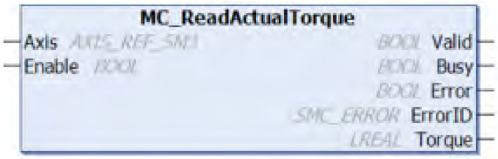
- **Troubleshooting**
  - ■ If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - ■ AX-308E、AX-8

2.2.9 MC_ReadActualTorque

MC_ReadActualTorque reads the actual torque value of axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadActualTorque	 MC_ReadActualTorque
		Axis AXIS_REF_SM3
		Enable BOOL
		BOOL Valid
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		LREAL Torque
			MC_ReadActualTorque_instance (
			Axis :=,
			Enable :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			Torque =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False (False)	-

- **Outputs**

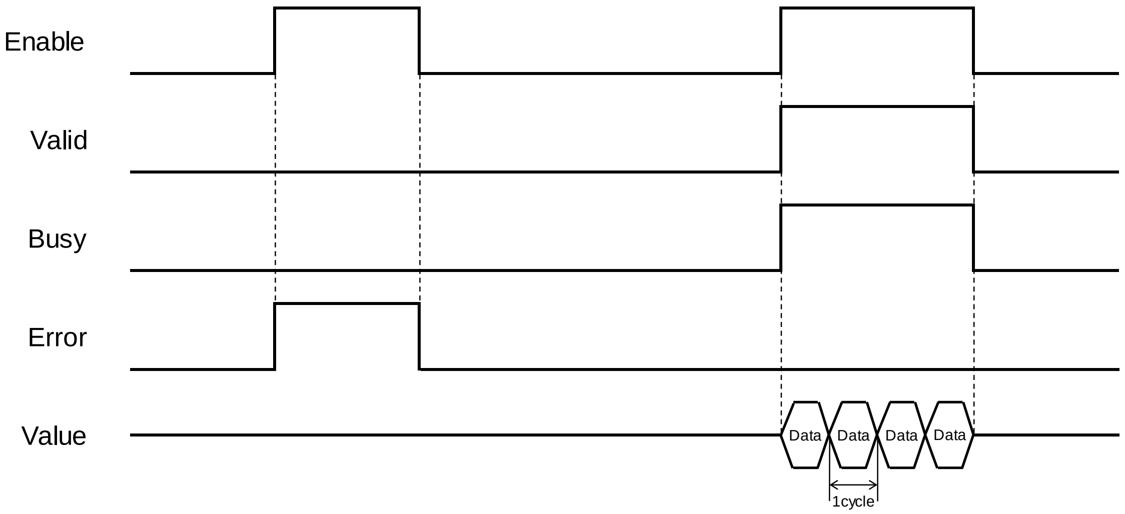
Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
Torque	The actual torque of axis.	LREAL	Positive number or 0(0)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
Torque	● When *Valid* is True and there're ongoing updates.	● When Valid is False and stop updating.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Enable* shifts to True.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - The torque value read by MC_ReadActualTorque is the value of fActTorque in AXIS_REF_SM3.
  - While using MC_ReadActualTorque, OD 0x6077 (Torque actual value) must be mapping to TxPDO so as to read the actual torque of the servo.

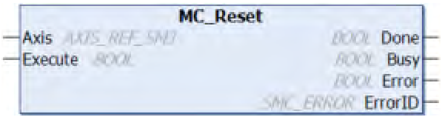
- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - AX-308E、AX-8

2.2.10 MC_Reset

MC_Reset clears axis-related errors so that the error memory is available for new error messages.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_Reset	 MC_Reset
		Axis AXIS_REF_SM3
		Execute BOOL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
			MC_Reset_instance (
			Axis :=,
			Execute :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-

- **Outputs**

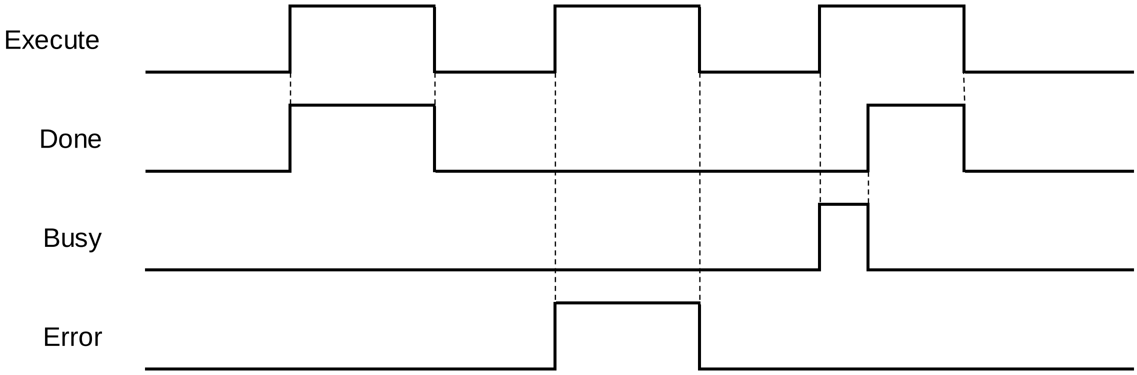
Name	Function	Data Type	Output Range (Default Value)
Done	Errors are cleared and the status changes to Standstill or Disabled.	BOOL	True/False(False)
Busy	True when the instruction is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When complete clearing axis-related errors.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When *Execute* is rising edge triggered.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**

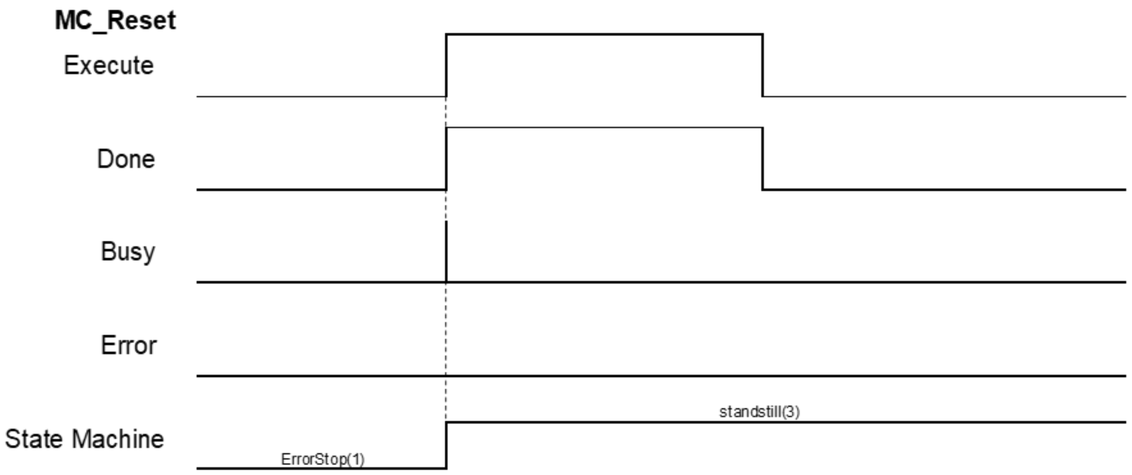


- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When Execute shifts to True and Busy is False.

***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Function**
  - The function block MC_Reset can change the error status of axis back to normal. When Enable of MC_Power is True, the axis status changes from Errorstop to Standstill. When Enable of MC_Power is False, the axis status changes from Errorstop to Disabled.



  - After errors being reported by the servo controller, users can use MC_Reset to clear them and then the axis state will return to Standstill.
  - If not able to use MC_Reset to clear the axis errors, such as communication error, SMC_R_ERROR_NOT_RESETTABLE 122 (Error could not be reset.) will be reported by MC_Reset.

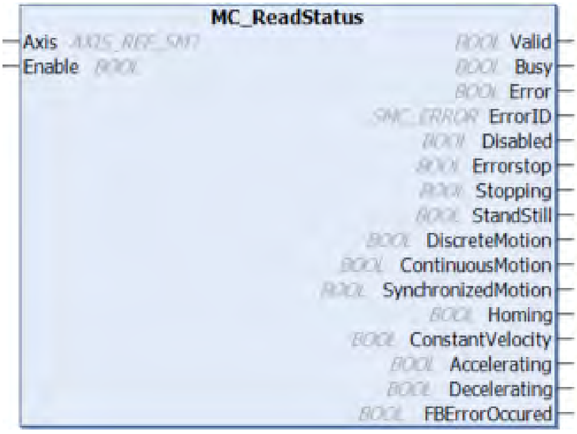
- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Supported Products**
  - AX-308E、AX-8

2.2.11 MC_ReadStatus

MC_ReadStatus reads ther status of the specified axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadStatus	 MC_ReadStatus
		Axis AXIS_REF_SM3
		Enable BOOL
		BOOL Valid
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		BOOL Disabled
		BOOL Errorstop
		BOOL Stopping
		BOOL StandStill
		BOOL DiscreteMotion
		BOOL ContinuousMotion
		BOOL SynchronizedMotion
		BOOL Homing
		BOOL ConstantVelocity
		BOOL Accelerating
		BOOL Decelerating
		BOOL FBErrorOccured
			MC_ReadStatus_instance (
			Axis :=,
			Enable :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			Disabled=>,
			Errorstop=>,
			Stopping=>,
			StandStill=>,
			DiscreteMotion=>,
			ContinuousMotion=>,
			SynchronizedMotion=>,
			Homing=>,
			ConstantVelocity=>,
			Accelerating=>,
			Decelerating=>,
			FBErrorOccured=>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False (False)	-

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*1	SMC_ERROR(SMC_NO_ERROR)
Disabled	Please refer to SMC_AXIS_STATE*2 for axis state descriptions.	BOOL	True/False(False)
Errorstop		BOOL	True/False(False)

Name	Function	Data Type	Output Range (Default Value)
Stopping	Please refer to SMC_AXIS_STATE*2 for axis state descriptions.	BOOL	True/False(False)
StandStill		BOOL	True/False(False)
DiscreteMotion		BOOL	True/False(False)
ContinuousMotion		BOOL	True/False(False)
SynchronizedMotion		BOOL	True/False(False)
Homing		BOOL	True/False(False)
ConstantVelocity	True when the axis moves at a constant speed.	BOOL	True/False(False)
Accelerating	True when the axis accelerates.	BOOL	True/False(False)
Decelerating	True when the axis decelerates.	BOOL	True/False(False)
FBErrorOccured	True when an error occurs.	BOOL	True/False(False)

***Note:**

1. SMC_ERROR:Enumeration(Enum)
2. SMC_AXIS_STATE:Enumeration(Enum)

■ **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
Disabled	● When the axis is in Disabled state.	● When the axis is not in Disabled state.
Errorstop	● When the axis is in Errorstop state.	● When the axis is not in Errorstop state.
Stopping	● When the axis is in Stopping state.	● When the axis is not in Stopping state.
StandStill	● When the axis is in StandStill state.	● When the axis is not in StandStill state.
DiscreteMotion	● When the axis is in Discrete Motion state.	● When the axis is not in Discrete Motion state.
ContinuousMotion	● When the axis is in Continuous Motion state.	● When the axis is not in Continuous Motion state.
SynchronizedMotion	● When the axis is in Synchronized Motion state.	● When the axis is not in Synchronized Motion state.
Homing	● When the axis is in Homing state.	● When the axis is not in Homing state.
ConstantVelocity	● When the axis moves at a constant speed.	● When the axis moves at a non-constant speed.
Accelerating	● When the axis moves with acceleration.	● When the axis moves without acceleration.
Decelerating	● When the axis moves with deceleration.	● When the axis moves without deceleration.
FBErrorOccured	● When errors exist.	● When errors are cleared.

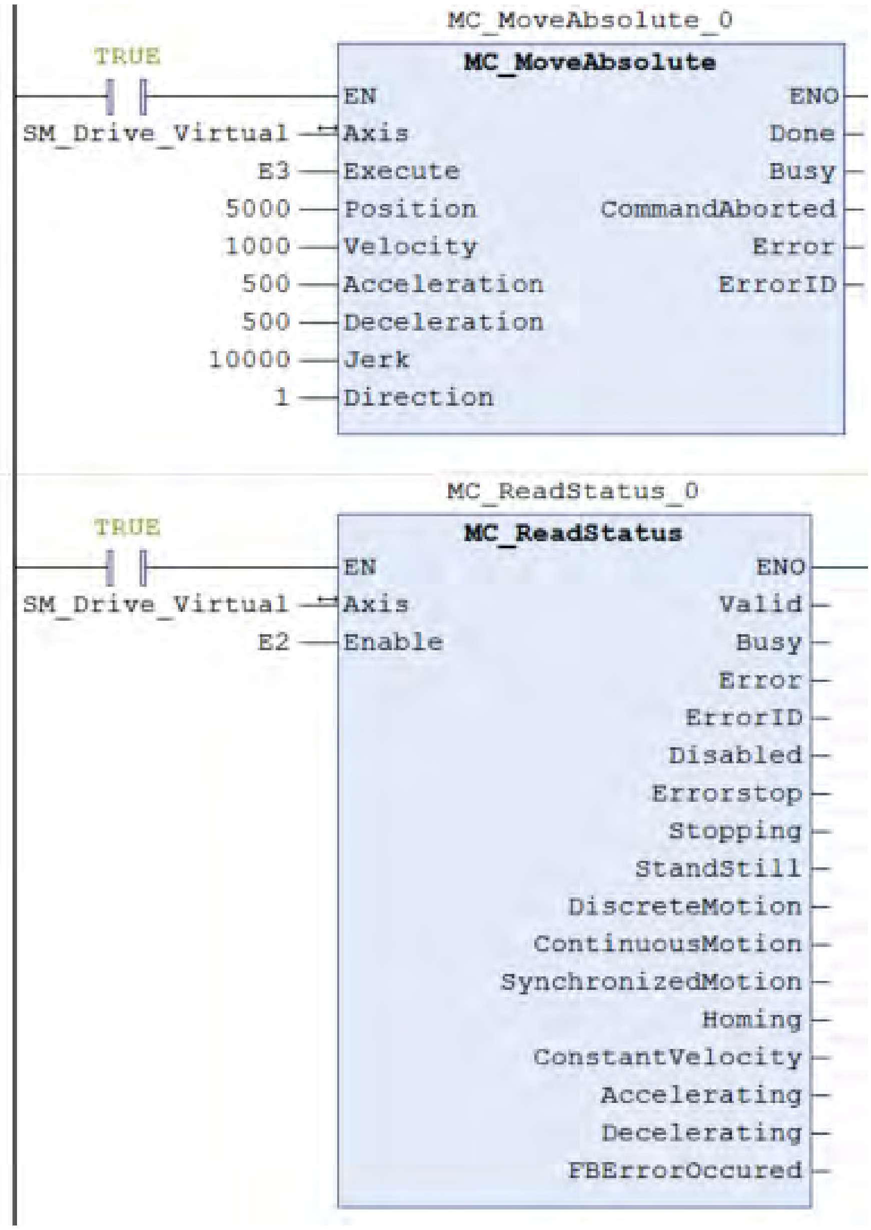
- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Enable* shifts to True.

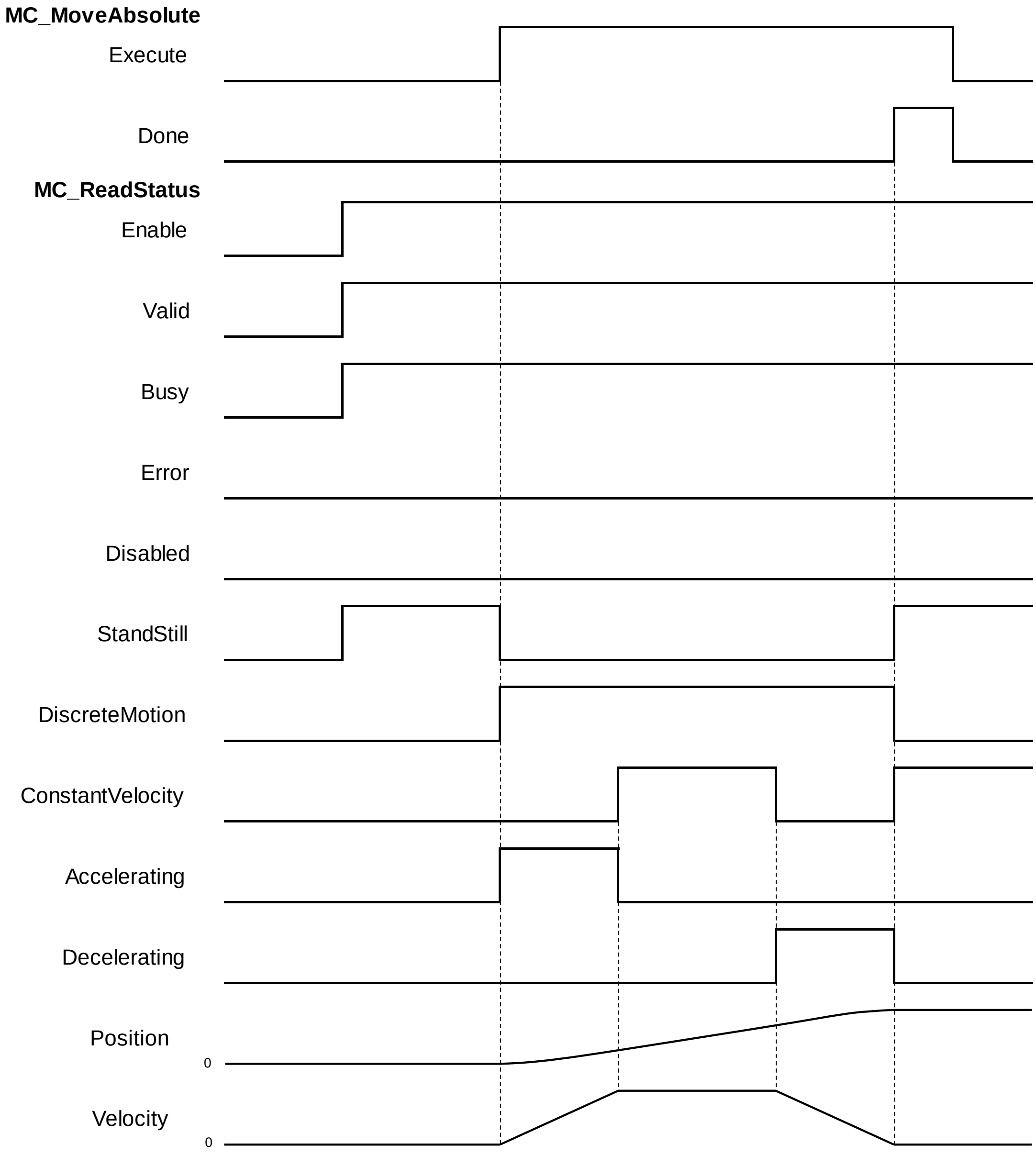
***Note:** AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

- **Troubleshooting**
  - ■ If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - This example demonstrates using MC_ReadStatus to read axis status while executing MC_MoveAbsolute.



- Timing Diagram



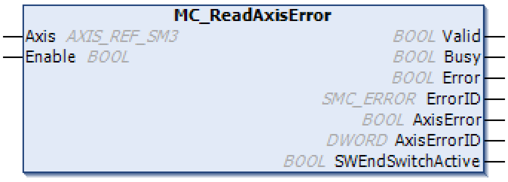
- ◆ After MC_MoveAbsolute being executed, axis state shifts from Standstill to Discrete_motion. At the same time, the axis begins to accelerate and Outputs*Accelerating* shifts to True.
- ◆ When axis velocity reaches the setting in MC_MoveAbsolute, the axis moves at constant speed. Meanwhile, the output of ConstantVelocity shifts to True and Outputs*Accelerating* shifts to False. Upon moving close to the target position, the axis starts decelerating, which *Decelerating* shifts to True and *ConstantVelocity* shifts to False.
- ◆ Done of MC_MoveAbsolute shifts to True when the target position is reached. Output status shifts from *Discretemotion* to *Standstil.*

- **Supported Products**
  - ■ AX-308E、AX-8

2.2.12 MC_ReadAxisError

MC_ReadAxisError reads the error information of axis.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_ReadAxisError	 MC_ReadAxisError
		Axis AXIS_REF_SM3
		Enable BOOL
		BOOL Valid
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		BOOL AxisError
		DWORD AxisErrorID
		BOOL SWEndSwitchActive
			MC_ReadAxisError_instance (
			Axis :=,
			Enable :=,
			Valid =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			AxisError =>,
			AxisErrorID =>,
			SWEndSwitchActive =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	Execute the function block.	BOOL	True/False (False)	-

- **Outputs**

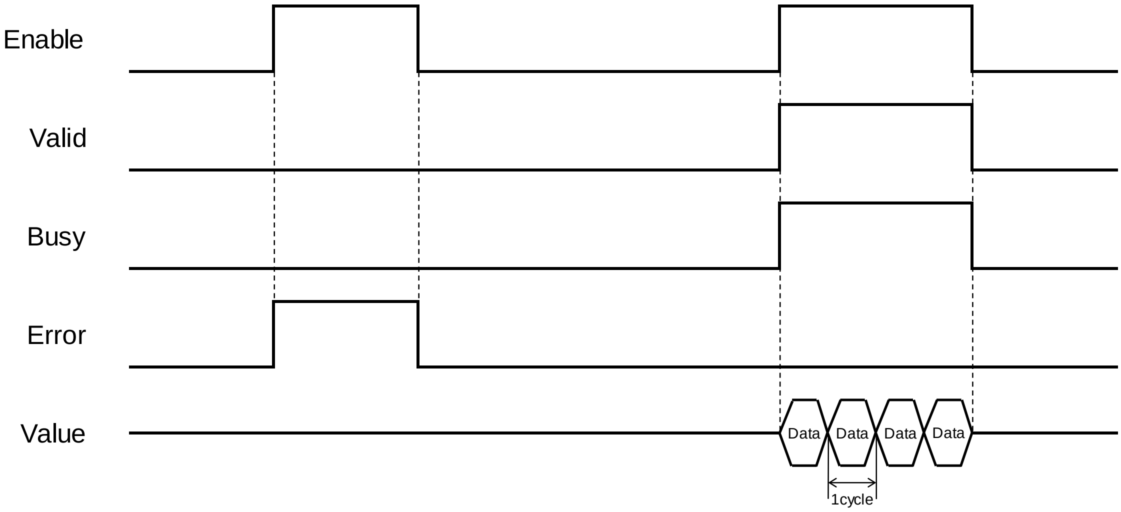
Name	Function	Data Type	Output Range (Default Value)
Valid	True when the parameter to read exists and can be further processed.	BOOL	True/False(False)
Busy	True when the function block is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
AxisError	True if an error occurs in the axis.	BOOL	True/False(False)
AxisErrorID	Error codes specified by the vender.	DWORD	Positive number or 0(0)
SWEndSwitchActive	True when the axis exceeds the software limit.	BOOL	True/False(False)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Valid	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Busy	● When *Enable* is rising edge triggered. ● When the parameter to read exists.	● When *Enable* shifts from True to False. ● When *Error* is rising edge.
Error	● When an error occurs in executing conditions or input values.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
AxisError	● When an error occurs in the axis.	● When the error is removed.
AxisErrorID		
SWEndSwitchActive	● When the axis exceeds the software limit.	● When executes MC_Reset.

- **Timing Diagram**



- **In-Outs**

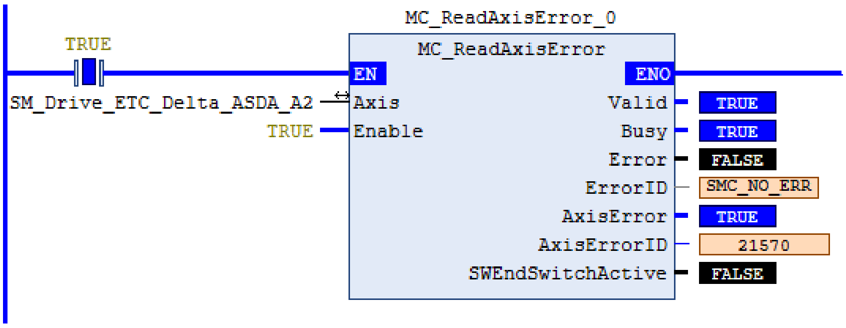
Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *Enable* shifts to True.

***Note:** AXIS_REF_SM3(FB): Each function block has this in-out pin for the activation of function block.

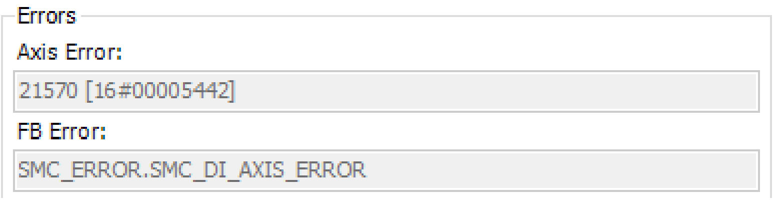
- **Function**
  - Outputs*SWEndSwitchActive* will shift to True once the axis reaches the software limit.
  - AxisErrorID displays the error codes of the servo motor itself. Take ASDA-A2-E for example, when error codes appear in the display on the servo panel, MC_ReadAxisError requests the servo for its error code by giving Error Code(0x603F) and the servo's error code would be displayed on the monitoring screen of axis.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The following example gives the status read by MC_ReadAxisError when the servo reaches hardware limit.



```
                                        MC_ReadAxisError_0
      TRUE                            MC_ReadAxisError
                                  EN                        ENO
SM_Drive_ETC_Delta_ASDA_A2 ──     Axis                    Valid ─ TRUE
                      TRUE ──     Enable                   Busy ─ TRUE
                                                          Error ─ FALSE
                                                        ErrorID ─ SMC_NO_ERR
                                                      AxisError ─ TRUE
                                                    AxisErrorID ─ 21570
                                              SWEndSwitchActive ─ FALSE
```



Errors

Axis Error:

21570 [16#00005442]

FB Error:

SMC_ERROR.SMC_DI_AXIS_ERROR

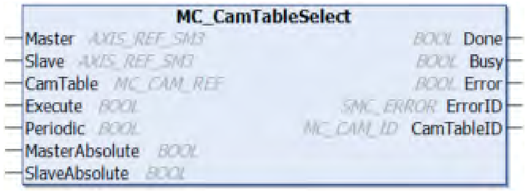
When ASDA-A2-E servo touches the positive hardware limit, "AL015" would be displayed on the servo panel. Meanwhile, use MC_ReadAxisError to read the corresponding error code. The error code for AL015 is 0x5442(Please refer to ASDA-A2-E user manual.) AxisErrorID is used to display the error code, which would also be displayed simultaneously on the monitoring screen of axis.

- **Supported Products**
  - AX-308E、AX-8

2.2.13 MC_CamTableSelect

MC_CamTableSelect selects the cam table for use with MC_CamIn.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_CamTableSelect		MC_CamTableSelect_instance (
			Master :=,
			Slave :=,
			CamTable :=,
			Execute :=,
			Periodic :=,
			MasterAbsolute :=,
			SlaveAbsoulte :=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			CamTableID =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input *Execute* starts the function block.	BOOL	True/False (False)	-
Periodic	Periodic mode	BOOL	True/False (True)	When Execute shifts to True and Busy is False.
MasterAbsolute	MasterAbsolute mode	BOOL	True/False (True)	When Execute shifts to True and Busy is False.
SlaveAbsoulte	SlaveAbsoulte mode	BOOL	True/False (True)	When Execute shifts to True and Busy is False.

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
Done	True when the instruction is completed.	BOOL	True/False(False)
Busy	True when the instruction is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*[1]	SMC_ERROR(SMC_NO_ERROR)

Name	Function	Data Type	Output Range (Default Value)
CamTableID	Create CAM_ID for use by CamTableID of MC_CamIn.	MC_CAM_ID*[2]	MC_CAM_ID

*** Note:**

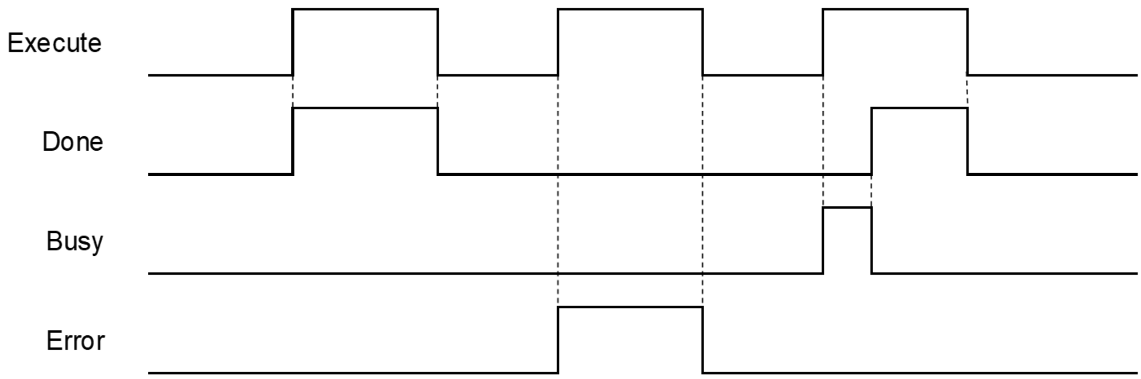
1. SMC_ERROR:Enumeration (Enum)
2. MC_CAM_ID: Structure (Struct)

Name	Function	Data Type	Output Range (Default Value)
pCT	The internal information described by the cam table.	POINTER TO BYTE	Positive number or 0(0)
Periodic	Periodic mode	BOOL	True/False(True)
MasterAbsolute	MasterAbsolute mode	BOOL	True/False(True)
SlaveAbsolute	SlaveAbsolute mode	BOOL	True/False(True)
StartMaster	The master start position of the cam table.	LREAL	Positive number, negative number or 0(0)
EndMaster	The master end position of the cam table.	LREAL	Positive number, negative number or 0(0)
StartSlave	The slave start position of the cam table.	LREAL	Positive number, negative number or 0(0)
EndSlave	The slave end position of the cam table.	LREAL	Positive number, negative number or 0(0)
byCompatibilityMode	Compatibility Mode	BYTE	Positive number or 0(0)

■ **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When CamTableSelect is completed.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When the instruction is being executed.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Master	Reference to master axis	AXIS_REF_SM3*[1]	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
Slave	Reference to slave axis	AXIS_REF_SM3*[1]	AXIS_REF_SM3	When Execute shifts to True and Busy is False.
CamTable	Specified cam table	MC_CAM_REF*[2]	MC_CAM_REF	When Execute shifts to True and Busy is False.

***Note:**

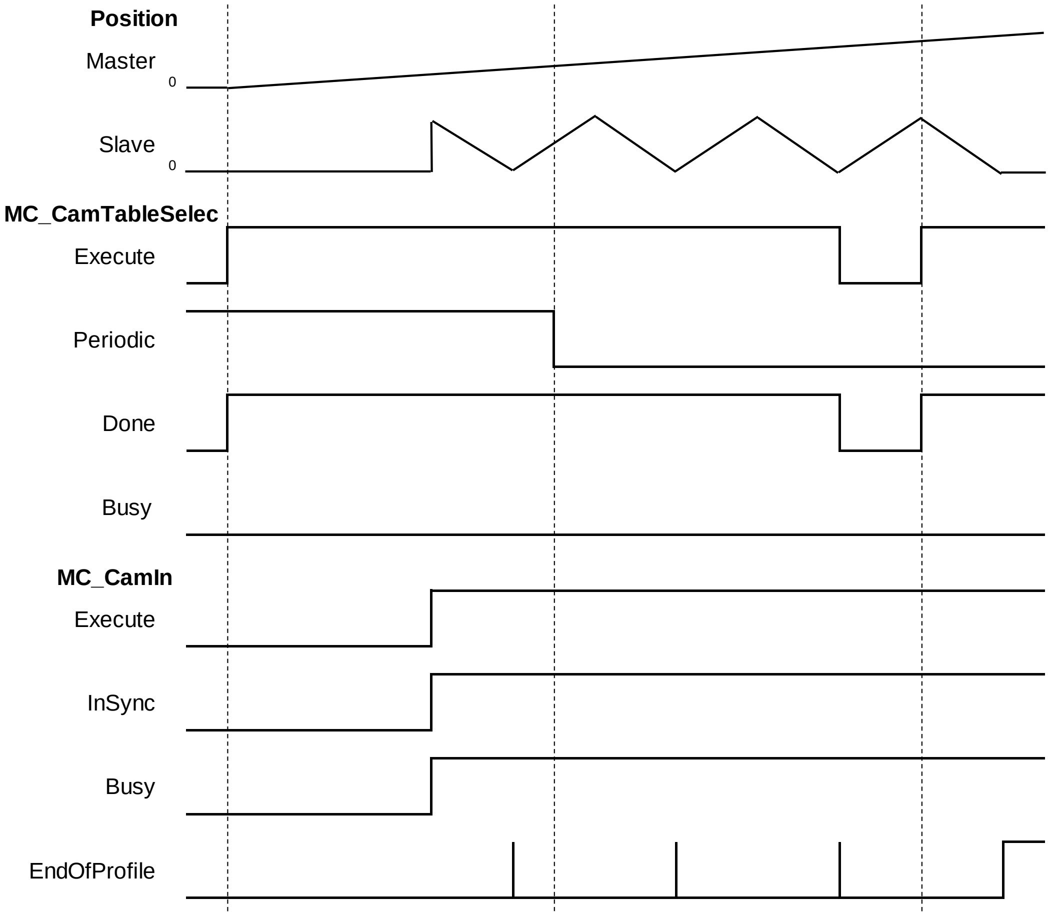
1. AXIS_REF_SM3(FB): Each function block has this in-out pin for the activation of function block.
2. MC_CAM_REF(FB): This data structure is used as reference to a cam table specified by users.

- **Function**
  - Use MC_CamTableSelect to select the cam table for operation.
  - Set Execute to be True so as to execute the specified or refreshed cam table. When Done shifts to True, CamTableID is effective.
  - After the master-slave synchronization is completed, the modification of MC_CamTableSelect parameters can cause changes in the cam behavior.
    - After changes the variables of CamTable, the mode of cam behavior will be effective immediately.
    - The function block must be reboot after changing Periodic mode.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - The following example explains the impact on cam after changing Periodic mode.

- Timing Diagram



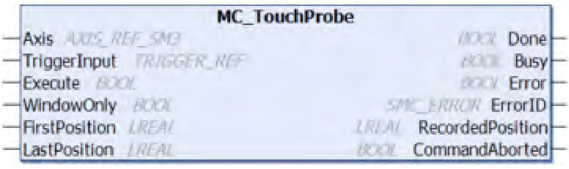
- To change the periodic mode, *Periodic* of MC_CamTable would be pulled down and shift to False, while the slave axis remains its periodicity.
- After rebooting MC_CamTable, the slave axis enters non-periodic mode. As soon the last period movement performed by the slave axis is completed, *EndOfProfile* will shift to True and remain unchanged.

- **Supported Products**
  - AX-308E、AX-8

2.2.14 MC_TouchProbe

MC_TouchProbe records an axis position at the time when a trigger event occurs.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_TouchProbe	 MC_TouchProbe
		Axis AXIS_REF_SM3
		TriggerInput TRIGGER_REF
		Execute BOOL
		WindowOnly BOOL
		FirstPosition LREAL
		LastPosition LREAL
		BOOL Done
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		LREAL RecordedPosition
		BOOL CommandAborted
			MC_TouchProbe_instance (
			Axis:=,
			TriggerInput:=,
			Execute :=,
			WindowOnly:=,
			FirstPosition:=,
			LastPosition:=,
			Done =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			RecordedPosition =>,
			CommandAborted =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (False)	-
WindowOnly	Activate the scope setting of Window.	BOOL	True/False(False)	When Execute shifts to True and Busy is False.
FirstPosition	Define the start position of the Window mask. (User-defined unit)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.
LastPosition	Define the last position of the Window mask. (User-defined unit)	LREAL	Negative number, positive number or 0(0)	When Execute shifts to True and Busy is False.

- **Outputs**

Name	Function	Data Type	Output Range (Default Value)
Done	If the trigger signal is True and the axis position has been recorded.	BOOL	True/False(False)
Busy	True when the instruction is being executed.	BOOL	True/False(False)
Error	True when an error occurs.	BOOL	True/False(False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

Name	Function	Data Type	Output Range (Default Value)
RecordedPosition	Display the axis position recorded at the time of trigger signal being True.	LREAL	LREAL(0)
CommandAborted	True when the instruction is aborted by MC_AbortTrigger.	BOOL	True/False(False)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the trigger singal is True and the axis position has been recorded.	● When Execute shifts from True to False. ● If Execute is False and Done shifts to True, Done will be True for only one period and immediately shift to False.
Busy	● When *Execute* is rising edge triggered.	● When *Done* shifts to True. ● When *Error* shifts to True. ● When *CommandAborted* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.	● When Execute shifts from True to False.(Error code is cleared)
ErrorID		
CommandAborted	● When the function block is aborted by MC_AbortTrigger.	● When Execute shifts from True to False. ● If Execute is False and CommandAborted shifts to True, CommandAborted will be True for only one period and immediately shift to False.

- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*[1]	AXIS_REF_SM3	-
TriggerInput	Trigger signal	TRIGGER_REF*[2]	TRIGGER_REF(-1)	When Execute shifts to True and Busy is False.

***Note:**

1. AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.
2. TRIGGER_REF: Structure (STRUCT)

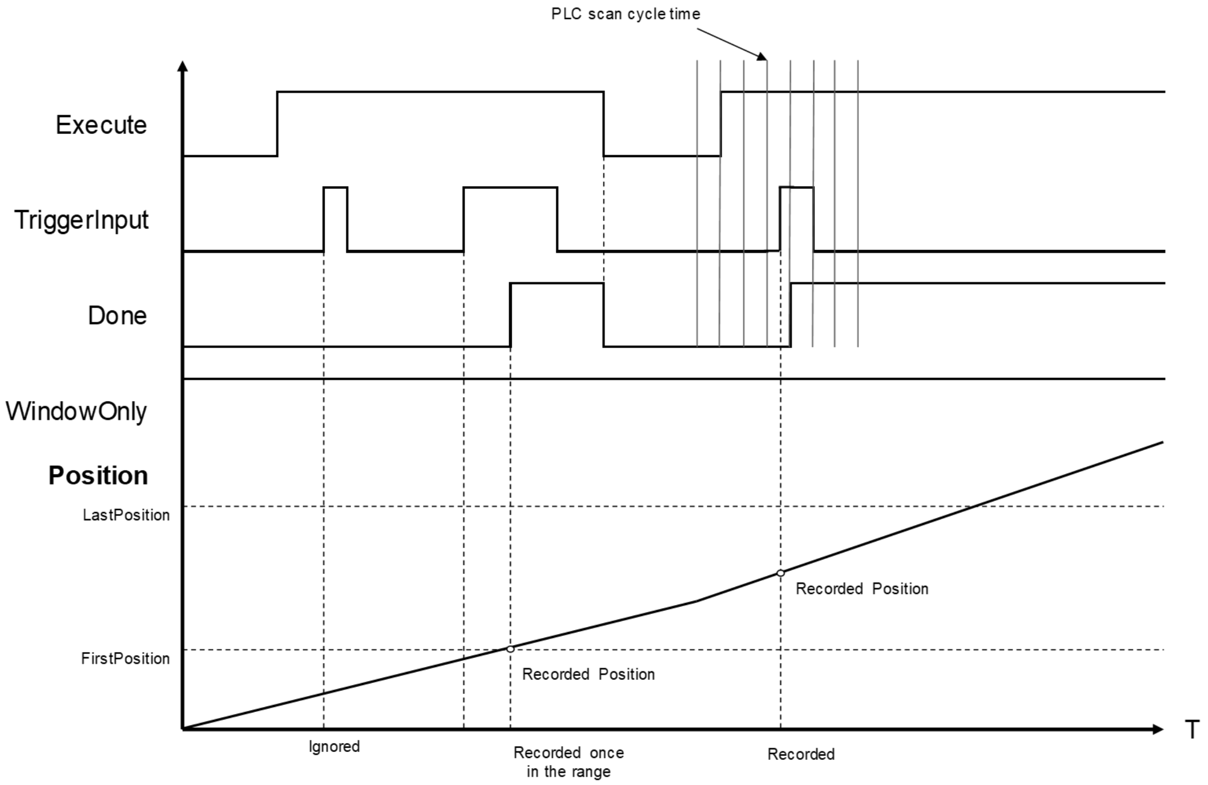
Name	Function	Data Type	Setting range (Default value)
iTriggerNumber	Trigger channel	INT	0:Touch Probe 1, rising edge 1:Touch Probe 1, falling edge 2:Touch Probe 2, rising edge 3:Touch Probe 2, falling edge (-1)
bFastLatching	Trigger signal	BOOL	True: Latching is done in drive False: Latching is done in motion controller (True)
bInput	Trigger signal when bFastLatching=FALSE	BOOL	Trigger signal
bActive	Validity of trigger signal	BOOL	True: Valid (False)

- **Function**
  - Drive mode
    - While using real axes, *bFastLatching* must be set to True (latching in drive) and configure *iTriggerNumber.* (The setting value can not be set as default “-1” or there’s ll be an error occurred in the function block.)

      When *Execute* is True, the function block writes values to 0x60B8(Touch Probe Function) based on the setting of *iTriggerNumber* to open the corresponding Trigger channel.
    - If Inputs*Execute* of MC_TouchProbe is True, only the first position value of trigger signal would be captured and the following signal would be ignored, even when bit1 of 0x60B8 is set to 1 to create multiple triggers.
    - Under drive mode, RecordedPosition reads the values in 0x60BA(Position value positive edge) and then convert with the gear ratio.

  - Controller mode
    - bFastLatching must be set to False and the trigger signal changes to be controlled by bInput.
    - RecordedPosition records the command position and the current command position when bInput triggers signals successfully.

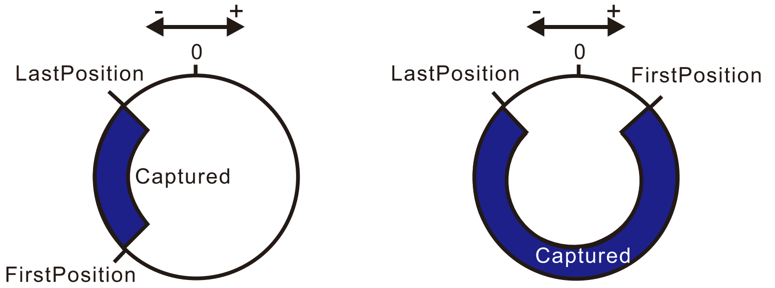
  - The operation of MC_TouhcProbe with window mask function is demonstrated as below:



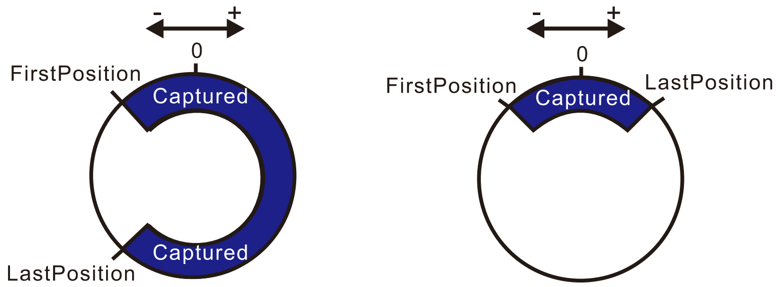
- At the first activation of the trigger input signal, the signal is not accepted because the axis position hasn't reach the specified window mask section.
- When the axis position enters the window mask section, the second activation of the trigger input signal is accepted, and after a period Done chnages to True.
- Time is needed until the touch probe operation is actually activated. The touch probe operation is not possibllyto be activated immediately after WindowOnly shifts to True.
- If the window mask is too small, the touch probe operation is not possble. The effective range for the window mask depends on EtherCAT communications and the performance of encoder input or the servo drive.
- In case that the servo drive does not support the window mask function, an error of SMC_TP_COULDNT_SET_WINDOW(401) will be reported by the function block. (Delta ASDA-A2-E has not yet supported *WindowsOnly* function.)

- Window Mask setting
  - You can observe the results of different window mask settings when the instruction is used for rotary/modulo axes as below. The difference is resulted from the set values between FirstPosition and LastPosition.

1. FirstPosition < LastPosition

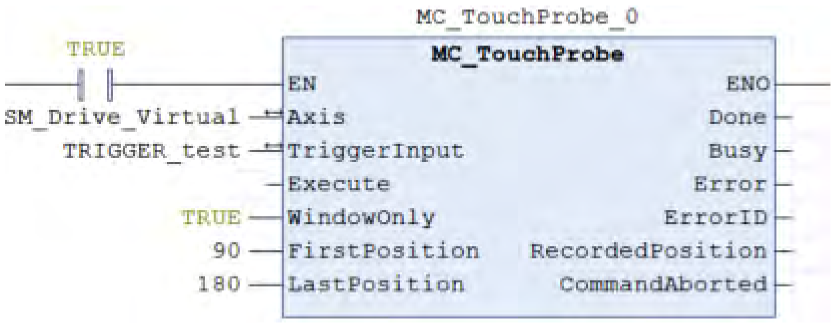


2. FirstPosition > LastPosition

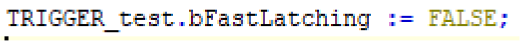


- Troubleshooting
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- Example
  - Example1: demonstrate the operation result of using MC_TouchProbe under controller mode.

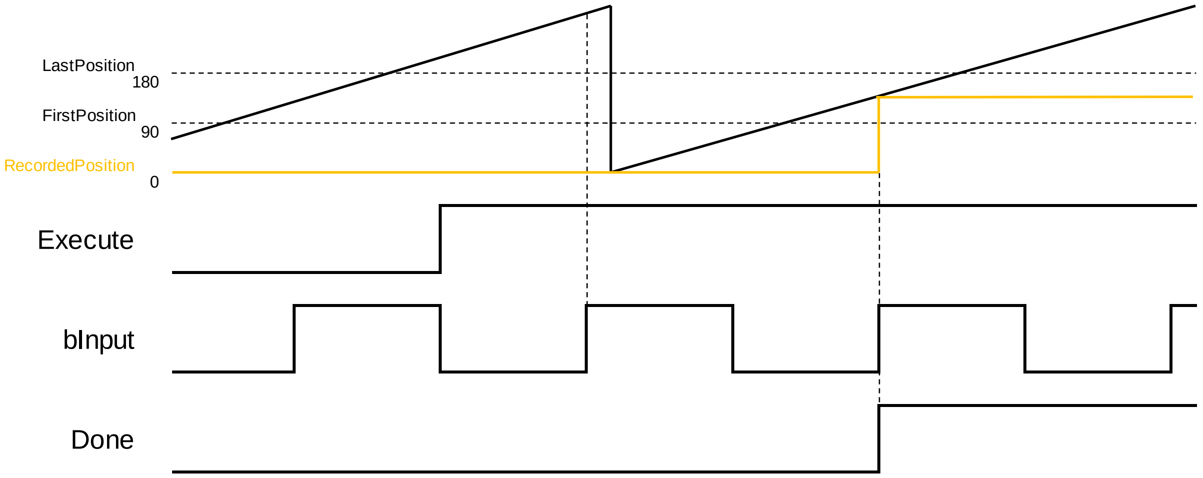


```
                        MC_TouchProbe_0
         TRUE       ┌──────────────────────────────────┐
  ───┤ ├────────────┤EN     MC_TouchProbe         ENO ├───
SM_Drive_Virtual ─⇆│Axis                          Done ├─
    TRIGGER_test ─⇆│TriggerInput                  Busy ├─
                  ─│Execute                      Error ├─
            TRUE ──│WindowOnly                 ErrorID ├─
              90 ──│FirstPosition     RecordedPosition ├─
             180 ──│LastPosition        CommandAborted ├─
                    └──────────────────────────────────┘
```



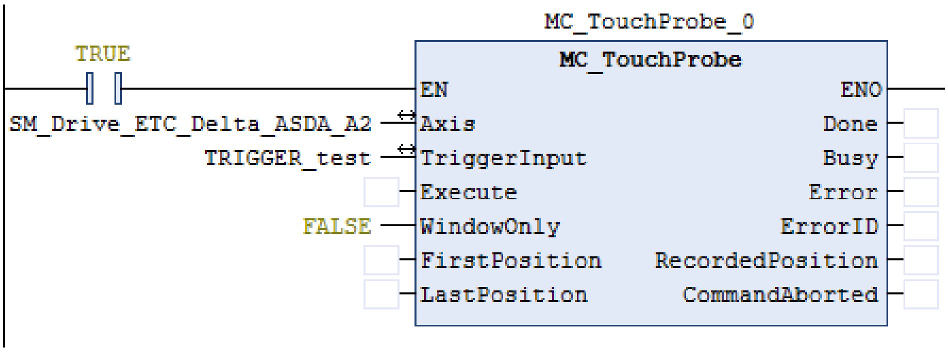
```
TRIGGER_test.bFastLatching := FALSE;
```

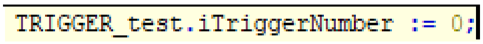
- Timing Diagram



- When Execute of MC_TouchProbe changes to True, it starts to capture the signal. Under controller mode, bInput would be the trigger signal.
- At the first activation of the trigger input signal, the axis position is not recorded because the axis position hasn't reached the specified window mask section. The axis has entered window mask section when trigger for the second time, therefore the position would be recorded in the output RecordedPosition.

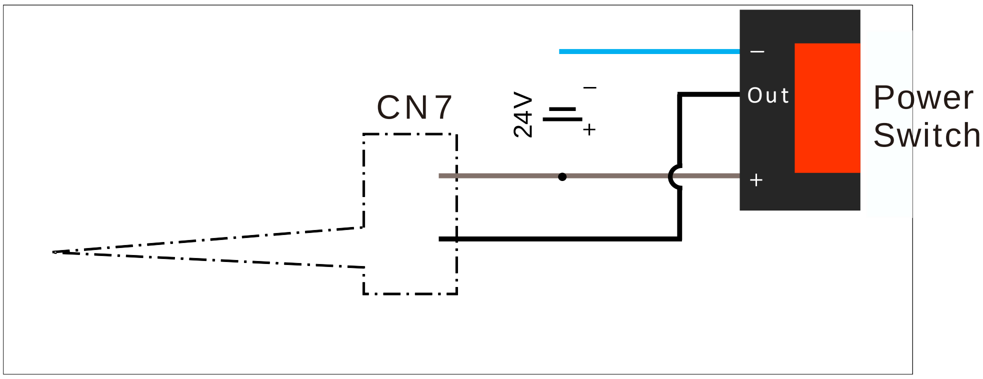
- Example2: Explain how MC_TouchProbe takes the drive as the trigger signal, which demonstrates with ASDA-A2-E as the drive.



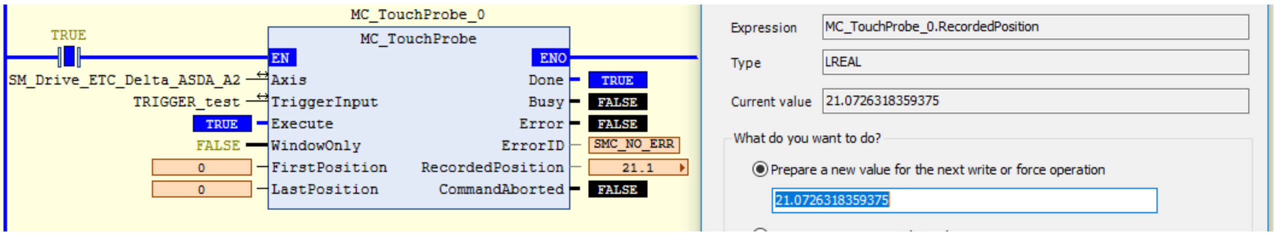


```
TRIGGER_test.iTriggerNumber := 0;
```

- Wiring diagram



- Trigger signal is from DI13 of CNY extension DI connector. You can start the configuration with the diagram above.
- Trigger channel must be specified by the function block. The following example demonstrates with rising edge trigger.



Input value	Input	Output	Output value
SM_Drive_ETC_Delta_ASDA_A2	Axis	Done	TRUE
TRIGGER_test	TriggerInput	Busy	FALSE
TRUE	Execute	Error	FALSE
FALSE	WindowOnly	ErrorID	SMC_NO_ERR
0	FirstPosition	RecordedPosition	21.1
0	LastPosition	CommandAborted	FALSE

Expression: MC_TouchProbe_0.RecordedPosition

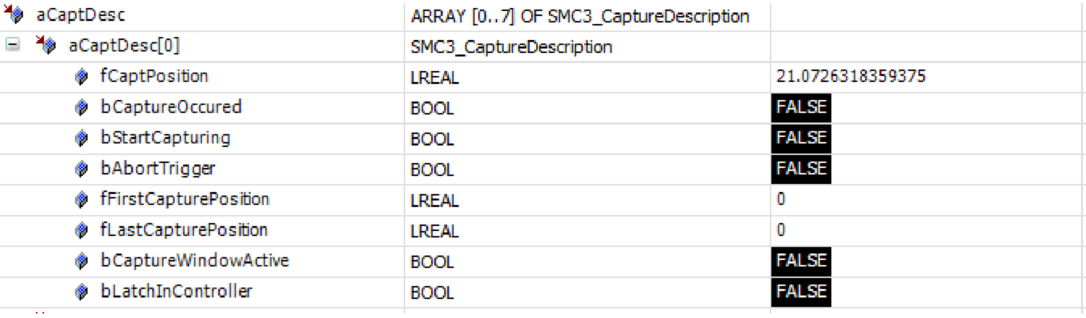
Type: LREAL

Current value: 21.0726318359375

What do you want to do?

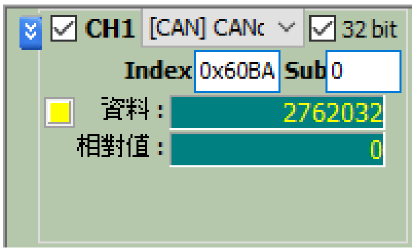
● Prepare a new value for the next write or force operation

21.0726318359375



aCaptDesc	ARRAY [0..7] OF SMC3_CaptureDescription	
⊟ aCaptDesc[0]	SMC3_CaptureDescription	
fCaptPosition	LREAL	21.0726318359375
bCaptureOccured	BOOL	FALSE
bStartCapturing	BOOL	FALSE
bAbortTrigger	BOOL	FALSE
fFirstCapturePosition	LREAL	0
fLastCapturePosition	LREAL	0
bCaptureWindowActive	BOOL	FALSE
bLatchInController	BOOL	FALSE

- When the signal on DI13 of the servo is triggered, MC_TouchProbeOutputsDone would be True. At the same time, MC_TouchProbe reads the value stored in the object 0x60BA(Touch Probe Pos1 Pos Value). After being converted with the gear ratio, the value would be stored in the axis parameter fCaptPosition, which would be output by RecordedPosition.

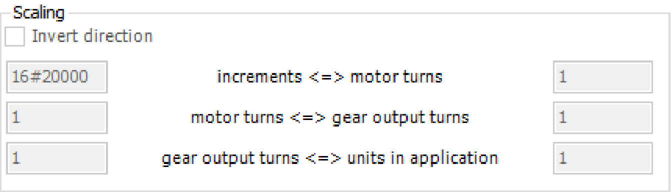


☑ CH1 [CAN] CANc ☑ 32 bit

Index 0x60BA Sub 0

資料： 2762032

相對值： 0



Scaling

□ Invert direction

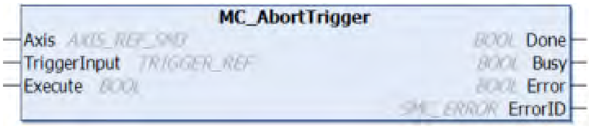
16#20000	increments <=> motor turns	1
1	motor turns <=> gear output turns	1
1	gear output turns <=> units in application	1

- As a result of the gear ratio being set to 0x20000:1, when the drive is rising edge triggered, the value in 0x60BA must be divided with 0x20000. The signal is triggered by the 2762032 index pulses; therefore, the position is recorded at 21.0726318359375(2762032 / 131072).

- **Supported Products**
  - AX-308E、AX-8

2.2.15 MC_AbortTrigger

MC_AbortTrigger aborts the instruction MC_TouchProbe which are intended to capture trigger events.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic Expression</th><th>ST language</th></tr>
<tr><td>FB</td><td>MC_AbortTrigger</td><td>
</td><td>MC_AbortTrigger_instance (
Axis :=,
TriggerInput :=,
Execute:=,
Done =>,
Busy =>,
Error =>,
ErrorID =>);</td></tr>
</table>

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Execute	A rising edge of the input Execute starts the function block.	BOOL	True/False (True)	-

- **Outputs**

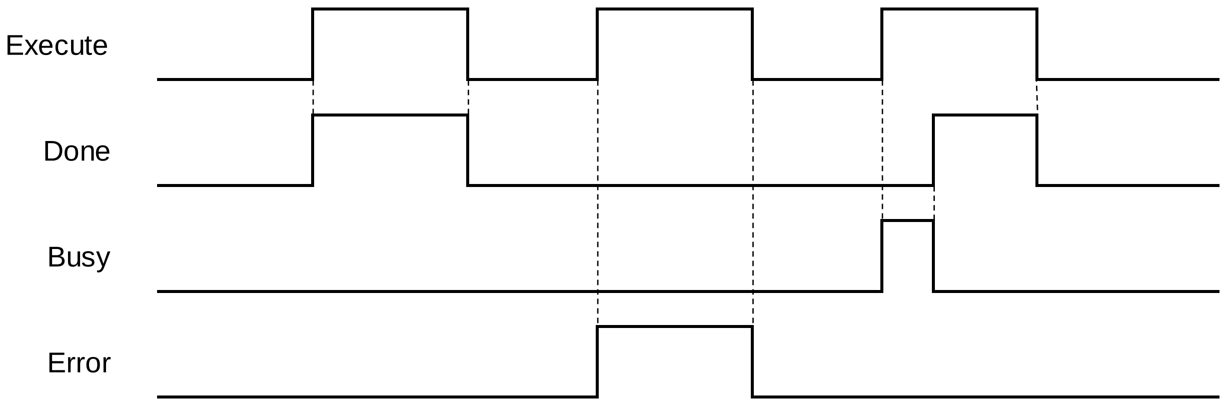
Name	Function	Data Type	Output Range (Default Value)
Done	True when trigger event is aborted	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
Error	True if an error occurs	BOOL	True/False(False)
ErrorID	Indicates the error code when the error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
Done	● When the capture operation is stopped.	● When *Execute* shifts from True to False. ● If *Execute* is False and *Done* shifts to True, it will be True for only one period and immediately shift to False.
Busy	● When *Execute* shifts to True.	● When *Done* shifts to True. ● When *Error* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.(Error code is recorded)	● When *Execute* shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3*1	AXIS_REF_SM3	When *Execute* shifts to True and *Busy* is False.
TriggerInput	Specifies the reference to the source of the trigger signal.	TRIGGER_REF*2	TRIGGER_REF	When *Execute* shifts to True and *Busy* is False.

***Note:**

1. AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.

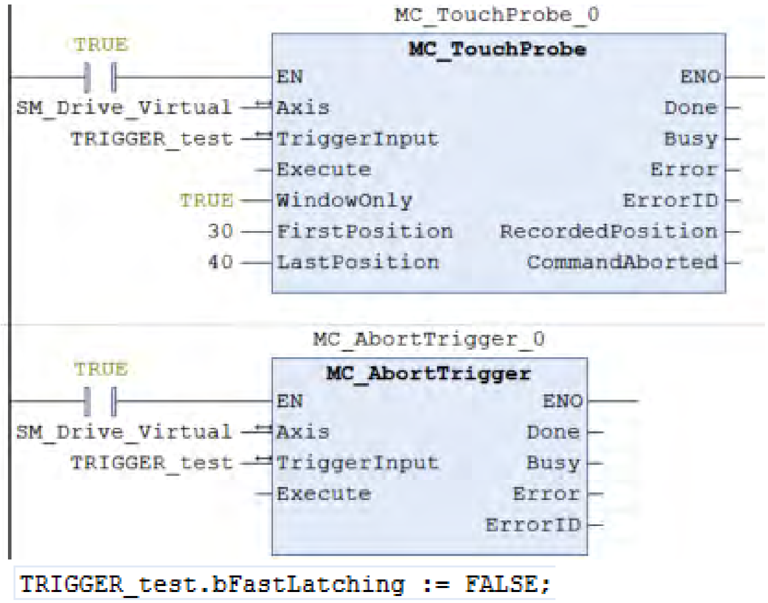
2. TRIGGER_REF: Structure(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
iTriggerNumber	Trigger channel	INT	0:Touch Probe 1, rising edge 1:Touch Probe 1, falling edge 2:Touch Probe 2, rising edge 3:Touch Probe 2, falling edge (-1)
bFastLatching	Trigger signal	BOOL	True: Latching is done in drive False: Latching is done in motion controller (True)
bInput	Trigger signal when bFastLatching=FALSE	BOOL	Trigger signal
bActive	Validity of trigger signal	BOOL	True: Valid (False)

- **Function**
  - ■ You can cancel the touch probe operation by using MC_AbortTrigger.
  - ■ By setting Axis and TriggerInput for this instruction you can define the touch probe operation to abort.

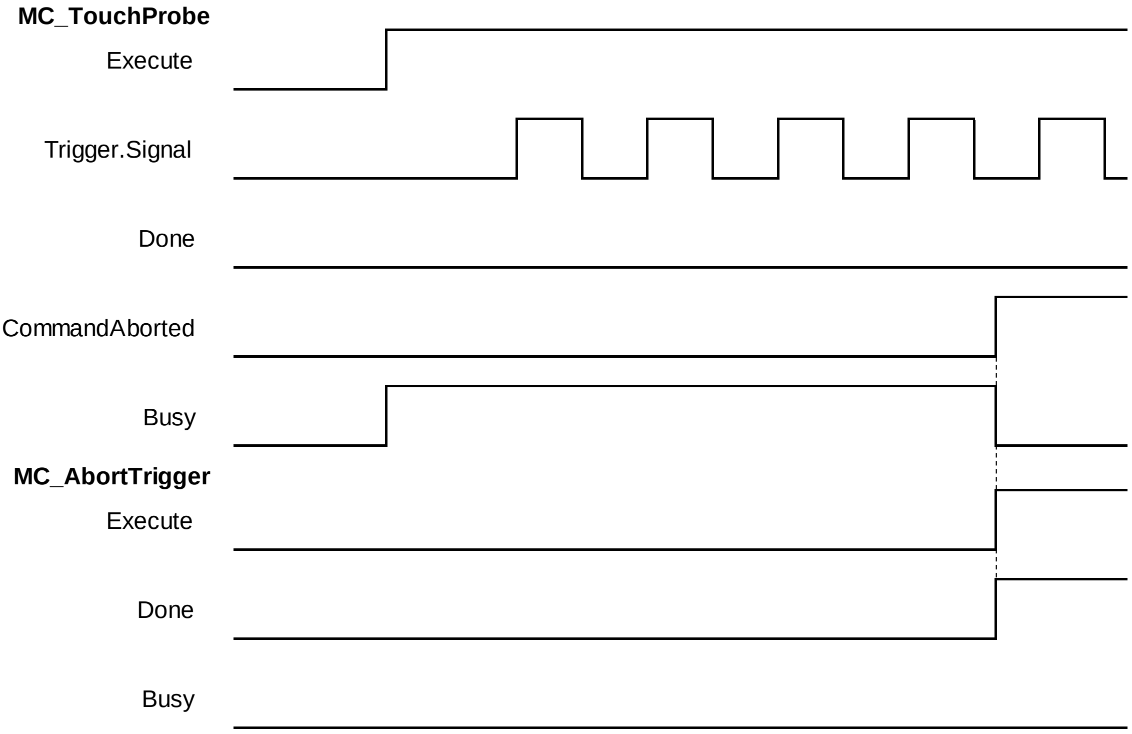
- **Troubleshooting**
  - ■ If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

- **Example**
  - ■ This example demonstrates the relating operation of the combination of MC_AbortTrigger and MC_TouchProbe.



```
TRIGGER_test.bFastLatching := FALSE;
```

- Timing Diagram



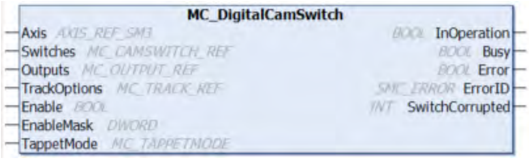
- When a rising edge is detected on *Execute* of MC_AbortTrigger, *CommandAborted* of MC_TouchProbe shifts to True.
- If a rising edge is detected on *Execute* of MC_AbortTrigger when *Done* of MC_TouchProbe shifts to True, an error of SMC_AT_TRIGGERNOTOCCUPIED(410) will be reported by MC_AbortTrigger.

- **Supported Products**
  - AX-308E、AX-8

2.2.16 MC_DigitalCamSwitch

MC_DigitalCamSwitch uses the axis position to control a switch of a digital output.

FB/FC	Instruction	Graphic Expression	ST language

FB	MC_DigitalCam Switch	 MC_DigitalCamSwitch
		Axis AXIS_REF_SM3
		Switches MC_CAMSWITCH_REF
		Outputs MC_OUTPUT_REF
		TrackOptions MC_TRACK_REF
		Enable BOOL
		EnableMask DWORD
		TappetMode MC_TAPPETMODE
		BOOL InOperation
		BOOL Busy
		BOOL Error
		SMC_ERROR ErrorID
		INT SwitchCorrupted
			MC_DigitalCamSwitch_instance(
			Axis :=,
			Switches :=,
			Outputs:=,
			TrackOptions:=,
			Enable:=,
			EnableMask:=,
			TappetMode:=,
			InOperation =>,
			Busy =>,
			Error =>,
			ErrorID =>,
			SwitchCorrupted =>);

- **Inputs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Enable	A rising edge of the input Execute starts the function block.	BOOL	True/False(False)	-
EnableMask	Enable the different tracks.	DWORD	Positive number or 0(16#FFFFFFFF)	When *Enable* shifts to True.
TappetMode	Define the positions for the position-defined calculation of the tappets.	MC_TAPPETMODE*	0:tp_mode_auto 1:tp_mode_demandposition 2:tp_mode_actualposition (tp_mode_auto)	When *Enable* shifts to True.

***Note:** MC_TAPPETMODE:Enumeration(ENUM)

- **Outputs**

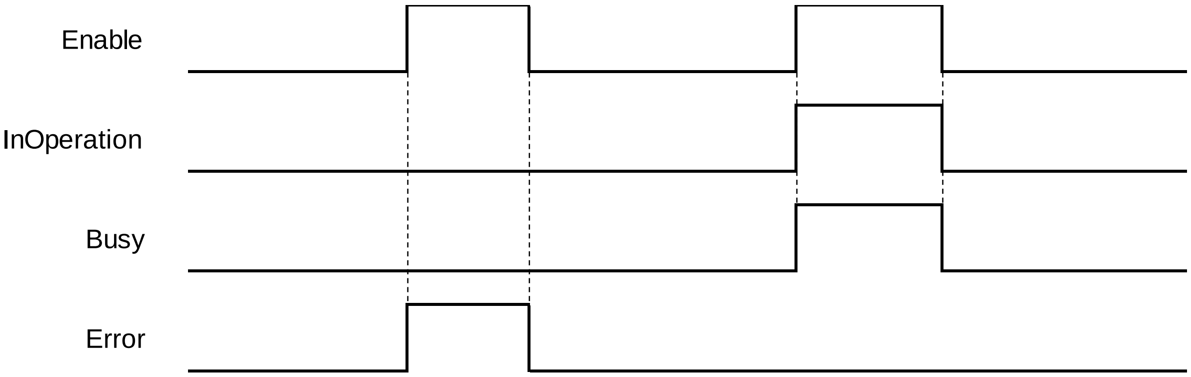
Name	Function	Data Type	Output Range (Default Value)
InOperation	True when the track and instruction is activated.	BOOL	True/False(False)
Busy	True when the instruction is executed.	BOOL	True/False(False)
Error	True if an error occurs	BOOL	True/False(False)
ErrorID	Indicates the error code when the error occurs. Refer to Appendices for error code descriptions.	SMC_ERROR*	SMC_ERROR(SMC_NO_ERROR)
SwitchCorrupted	When the switch action is operated abnormally, the output value will not be -1.	INT	Positive number, negative number or 0 (-1)

***Note:** SMC_ERROR:Enumeration(Enum)

- **Output Updating Time**

Name	Timimg for Shifting to True	Timimg for Shifting to False
InOperation	● When the track and instruction is activated.	● When *Enable* shifts to False.
Busy	● When *Execute* shifts to True.	● When *Enable* shifts to False. ● When *Errorv* shifts to True.
Error	● When an error occurs in the execution conditions or input values for the instruction.(Error code is recorded)	● When *Execute* shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data Type	Setting Value (Default Value)	Timimg for Updating
Axis	Reference to axis.	AXIS_REF_SM3[*1]	AXIS_REF_SM3	When Enable shifts to True.
Switches	Switch-related parameters.	MC_CAMSWITCH_REF[*2]	MC_CAMSWITCH_REF	When Enable shifts to True.
Outputs	Output signals of track	MC_OUTPUT_REF	ARRAY [1..32] OF BOOL(False)	When Enable shifts to True.
TrackOptions	Compensation and Hysteresis parameters for the cam track.	MC_TRACK_REF	ARRAY [1..32] OF MC_TRACK_TR[*3]	When Enable shifts to True.

***Note:**

1. AXIS_REF_SM3(FB):Each function block has this in-out pin for the activation of function block.
2. MC_CAMSWITCH_REF: Structure(STRUCT)

Name	Function	Data Type	Setting Value (Default Value)
NoOfSwitches	Specify the number of switches.	BYTE	Positive number or 0(0)
CamSwitchPtr	Point to an array of the type MC_CAMSWITCH_TR	POINTER TO MC_CAMSWITCH_TR*	POINTER TO MC_CAMSWITCH_TR(0)

***Note:** MC_CAMSWITCH_TR: Structure (STRUCT)

Name	Function	Data Type	Setting Value (Default Value)
TrackNumber	Specify the track for the operation of tappets.	INT	Positive number, negative number or 0(0)
FirstOnPosition	Switch ON when the axis passes.	LREAL	POINTER TO MC_CAMSWITCH_TR(0)
LastOnPosition	Switch OFF when the axis passes.	LREAL	Positive number, negative number or 0(0)
AxisDirection	The switch is active only when the axis is moving in the specified direction.	INT	Positive number, negative number or 0(0)
CamSwitchMode	Switch mode	INT	Positive number, negative number or 0(0)
Duration	How long the switch is on.	TIME	Positive number or 0(0)
bOn	Internal variables	BOOL	True/False(False)
CounterOff	Internal variables	INT	Positive number or 0(0)

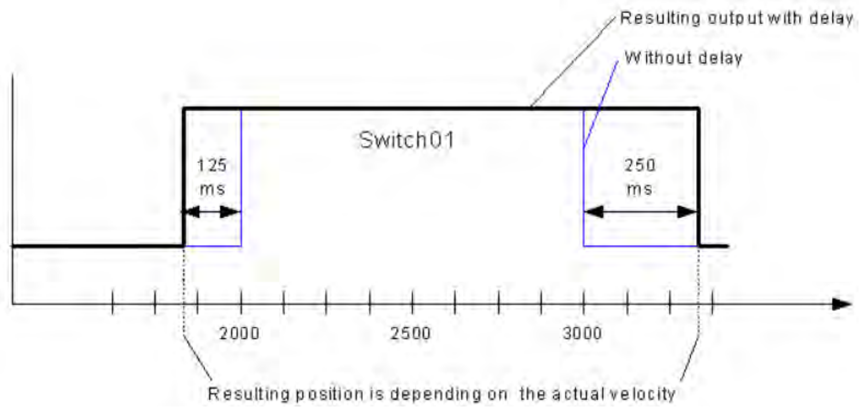
3. MC_TRACK_TR: Stucture(STRUCT)。

Name	Function	Data Type	Setting Value (Default Value)
OnCompensation	Compensation time with which the switch is turned on. (Unit: Sec.)	LREAL	Positive number, negative number or 0(0)
OffCompensation	Compensation time with which the switch is turned off. (Unit: Sec.)	LREAL	Positive number, negative number or 0(0)
Hysteresis	Hysteresis interval	LREAL	Positive number, negative number or 0(0)

- **Function**

  - EnableMask is a 32 bits of bool type parameter, used to enable different tracks. With the concept of the least significant bit representing the first track, the input value would be 16#FFFFFFFB to disable the third track.
  - MC_CAMSWITCH_REF defines switches for digital cam. NoOfSwitches calculates the number of switching positions. CamSwitchPtr is a pointer on an array of type MC_CAMSWITCH_TR.
  - MC_CAMSWITCH_TR specifies the positions of tappets.
    - TrackNumber specifies the output number.
    - FirstOnPosition specifies the switch-on position of the output.
    - LastOnPosition specifies the switch-off position of the output (when CamSwitchMode = 0).
    - AxisDirection = 0: Output is switched in both directions. AxisDirection = 1: Only positive direction. AxisDirection = 2: Only negative direction.
    - Switch is OFF at LastOnPosition when CamSwitchMode = 0. Switch remains ON for a time set (Duration) and then changes to OFF when CamSwitchMode = 1.
    - Duration: Period of time for which the tappet output stays TRUE in case of CamSwitchMode = 1.

- MC_TRACK_REF is the Structure for managing the tracks, which contains OnCompensation, OffCompensation and Hysteresis.
  - OnCompensation is set for the delay of switch-on. If the input value is positive, switching to ON would be delayed, while an early switch-on can be set with a negative input value. The time is given in seconds. For example, if OnCompensation is set to 0.01, switching to ON would be delayed for 0.1 second.
  - OffCompensation is set for the delay of switch-off. If the input value is positive, switching to OFF would be delayed, while an early switch-off can be set with a negative input value.

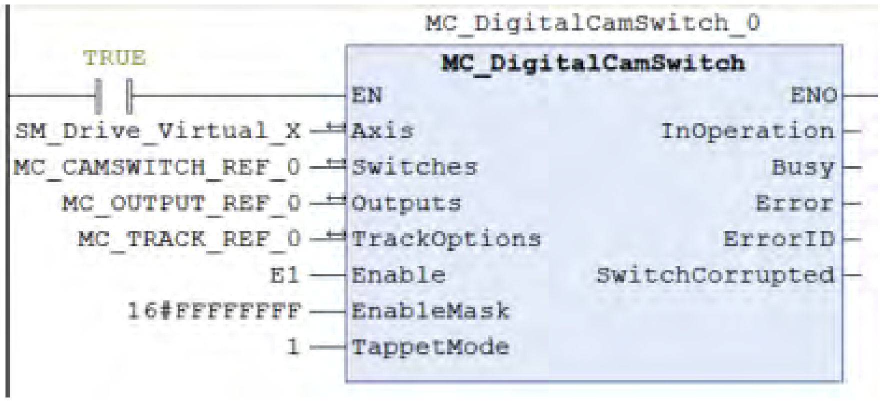

Resulting output with delay
Without delay
Switch01
125 ms
250 ms
2000
2500
3000
Resulting position is depending on the actual velocity

  - The interval for Hysteresis is set to avoid switching errors and the specified axis position must exceeds the interval, so the switch would continue with the next action. The unit of Hysteresis is user-defined.
- Output would be switched to ON with all AxisDirection settings as long as the axis position is inside of the range.
- Multiple switch modes are allowed to be set in a single Track.

- **Troubleshooting**
  - If an error occurs during the execution of the instruction, Error will change to True. You can refer to ErrorID (Error Code) to address the problem.

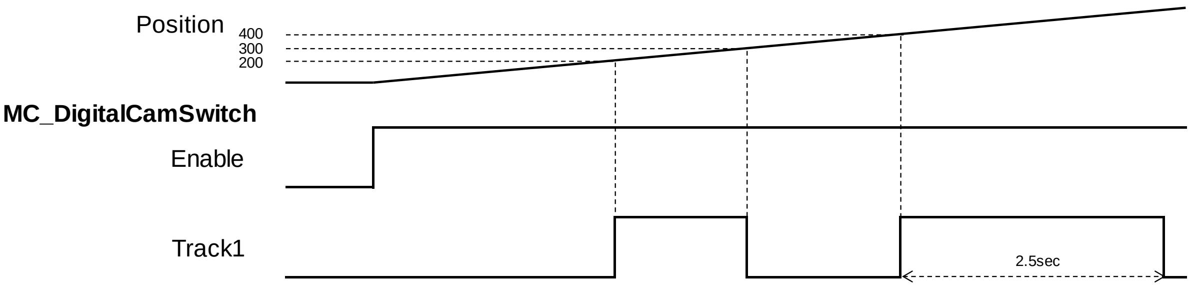
- **Example**
  - Example1: Demonstrates using 2 switches in the same Track in the following example.

Parameter	Type	Switch1	Switch2
TrackNumber	INT	1	1
FirstOnPosition [u]	REAL	200	400
LastOnPosition [u]	REAL	300	-
AxisDirection	INT	0=Both	0=Both
CamSwitchMode	INT	0=Position	1=TIME
Duration	TIME	-	2500ms



```
                              MC_DigitalCamSwitch_0
       TRUE                   MC_DigitalCamSwitch
                              EN                              ENO
SM_Drive_Virtual_X            Axis                    InOperation
MC_CAMSWITCH_REF_0            Switches                       Busy
  MC_OUTPUT_REF_0             Outputs                       Error
   MC_TRACK_REF_0             TrackOptions                ErrorID
               E1             Enable             SwitchCorrupted
      16#FFFFFFFF             EnableMask
                1             TappetMode
```

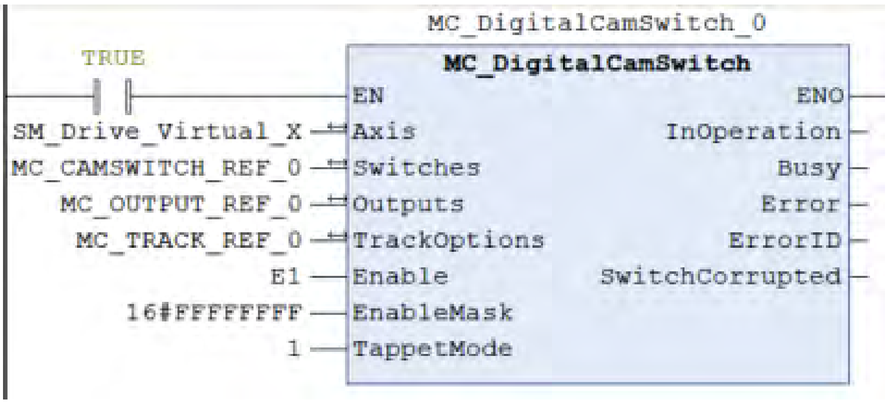
- Timing Diagram



- When the axis reaches the position 200, Switch1 on Track1 would be turned ON till the axis reaches the position 300, then changes to OFF.
- Switch1 would be turned ON again when the position 400 is reached, and lasting for 2.5 seconds, then changes to OFF.

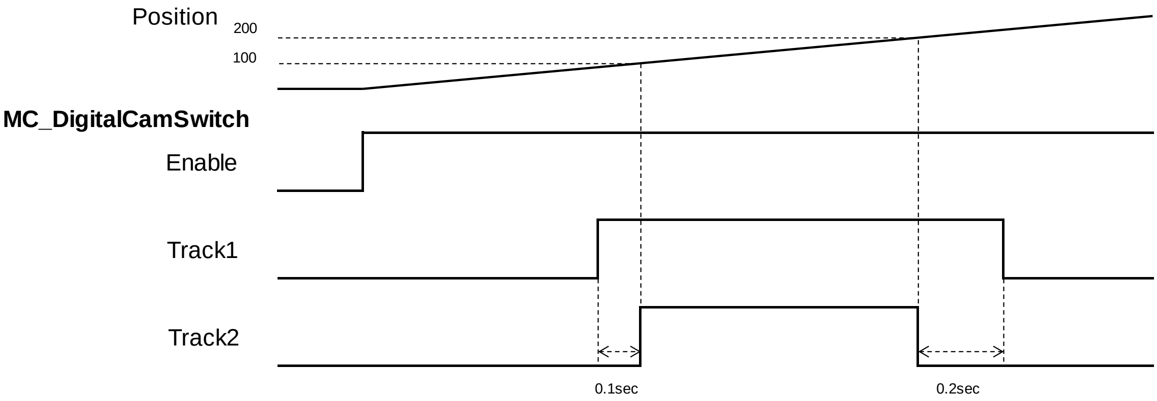
- Example2: The operation result of OnCompensation/OffCompensation is given in the following example.

Parameter	Type	Switch1	Switch2
TrackNumber	INT	1	2
FirstOnPosition [u]	REAL	100	100
LastOnPosition [u]	REAL	200	200
AxisDirection	INT	0=Both	0=Both
CamSwitchMode	INT	0=Position	0=Position
Duration	TIME	-	-
OnCompensation	LREAL	- 0.1	0
OffCompensation	LREAL	0.2	0



```
MC_DigitalCamSwitch_0
MC_DigitalCamSwitch
TRUE                  — EN                        ENO —
SM_Drive_Virtual_X    — Axis              InOperation —
MC_CAMSWITCH_REF_0    — Switches                 Busy —
MC_OUTPUT_REF_0       — Outputs                 Error —
MC_TRACK_REF_0        — TrackOptions          ErrorID —
E1                    — Enable        SwitchCorrupted —
16#FFFFFFFF           — EnableMask
1                     — TappetMode
```

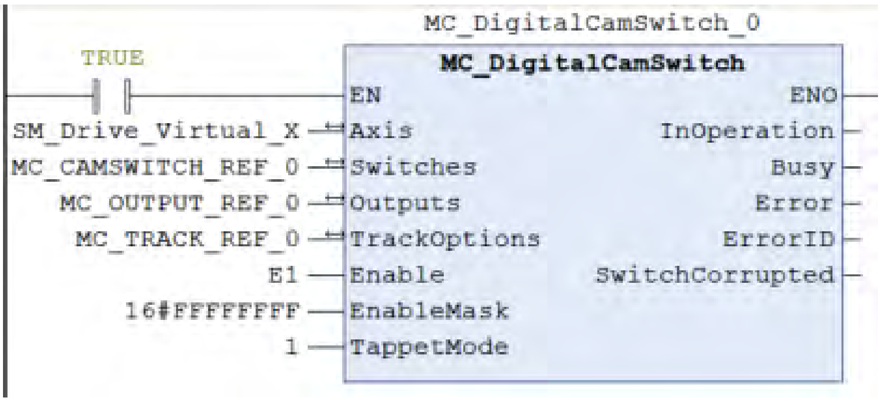
- Timing Diagram



◆ Once the position 100 is reached, Switch1 on Track1 and Switch2 on Track2 are both turned ON and would be turned OFF when position 200 is reached. The switch-on of Switch 1 is advanced for 0.1 second while OnCompensation = -0.1. By setting 0.2 to OffCompensation, Switch 1 is delayed for 0.2 second.

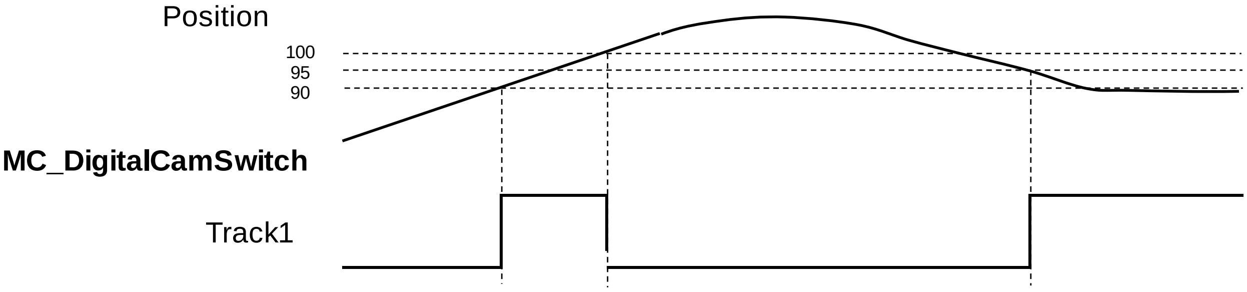
- Example3: The operation result of *Hysteresis* is given in the following example.

Parameter	Type	Switch1
TrackNumber	INT	1
FirstOnPosition [u]	REAL	90
LastOnPosition [u]	REAL	95
AxisDirection	INT	0=Both
CamSwitchMode	INT	0=Position
Duration	TIME	-
Hysteresis	LREAL	10



```
                                  MC_DigitalCamSwitch_0
           TRUE               MC_DigitalCamSwitch
     ───┤ ├─────────────────EN                      ENO─
SM_Drive_Virtual_X ─Axis                     InOperation─
MC_CAMSWITCH_REF_0 ─Switches                        Busy─
  MC_OUTPUT_REF_0 ─Outputs                         Error─
   MC_TRACK_REF_0 ─TrackOptions                  ErrorID─
               E1 ─Enable               SwitchCorrupted─
      16#FFFFFFFF ─EnableMask
                1 ─TappetMode
```

- Timing Diagram



- The FirstOnPosition and LastOnPosition of Switch 1 on Track1 are set to 90 and 95 respectively with Hysteresis set to 10, which means the switch would be turned off after the axis position passing the interval (80~100).
- Track 1 is switched to ON when the axis reaches position 90 and not able to be switched to OFF at position 95 until the axis passes the hysteresis interval.
- When the axis moves reversely to position 95, the switch would be turned ON again and remains, for the reason that the axis position stays within the hysteresis interval (105~85).

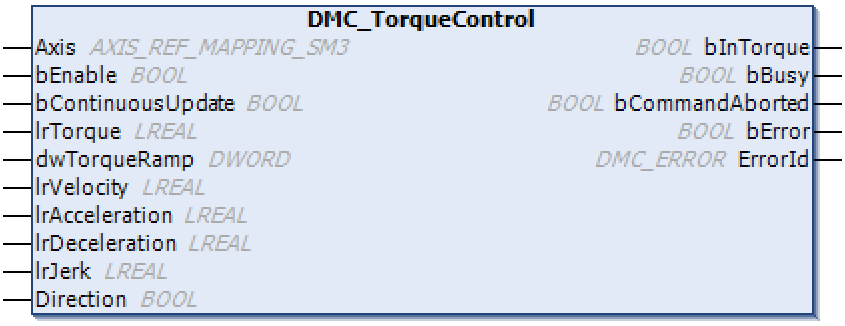
- **Supported Products**
  - AX-308E、AX-8

2.3 Delta Motion Control Instructions

2.3.1 Motion Control Instructions

2.3.1.1 DMC_TorqueControl

DMC_TorqueControl controls the torque according to the torque control mode of the servo drive.

FB/FC	Instruction	Graphic expression
FB	DMC_TorqueControl	 DMC_TorqueControl Axis AXIS_REF_MAPPING_SM3 — BOOL bInTorque bEnable BOOL — BOOL bBusy bContinuousUpdate BOOL — BOOL bCommandAborted lrTorque LREAL — BOOL bError dwTorqueRamp DWORD — DMC_ERROR ErrorId lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL lrJerk LREAL Direction BOOL

ST expression

DMC_TorqueControl_instance(
Axis :=,
bEnable :=,
bContinuousUpdate :=,
lrTorque :=,
dwTorqueRamp :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
lrJerk :=,
Direction :=,
bInTorque =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *bEnable* changes from False to True.	BOOL	True/False (False)	-
bContinuousUpdate	The target torque can be updated continuously when *bContinuousUpdate* is True.	BOOL	True/False (False)	When *bEnable* shifts to True and *Busy* is False.

Name	Function	Data type	Setting value (Default value)	Timing for updating
lrTorque	Specify the target torque. (Unit: N.m)	LREAL	Negative number, positive number, 0 (0)	When *bEnable* shifts to True and *Busy* is False.
dwTorqueRamp	Specify the change rate of the torque (Unit: ms) *	DWORD	Positive number (0)	When *bEnable* shifts to True and *Busy* is False.
lrVelocity	Specify the maximum velocity.	LREAL	Positive number (0)	When *bEnable* shifts to True and *Busy* is False.
lrAcceleration	Reserved	LREAL	-	-
lrDeceleration	Reserved	LREAL	-	-
lrJerk	Reserved	LREAL	-	-
Direction	Reserved	BOOL	-	-

***Note:**

Take ASDA-A2 for example here with the unit: μs (microsecond). For other models, refer to 0x6087 in their object dictionaries.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bInTorque	True when the target torque is reached.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

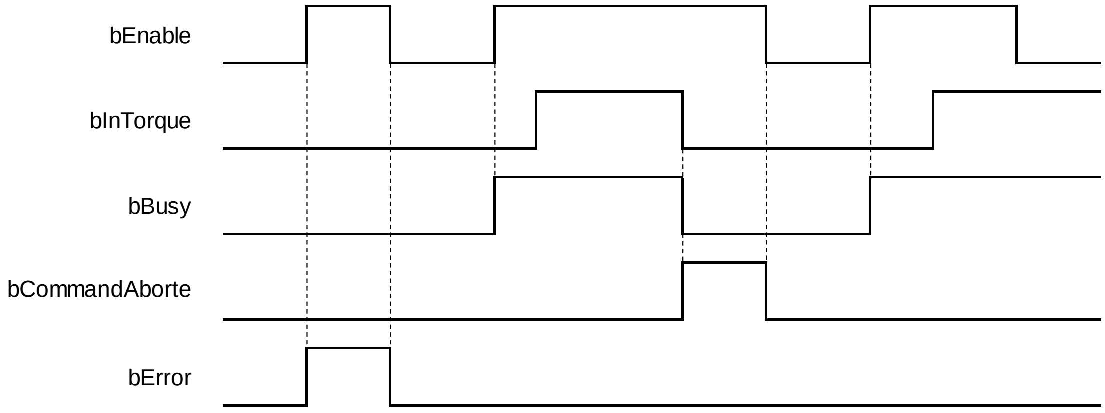
***Note:**

DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bInTorque	● When the *bEnable* is True and the axis motion state can be read.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* shifts to True.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bCommandAborted	● When this instruction is aborted by another instruction.	● When *bEnable* shifts to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* shifts to False. (The value in *ErrorID* is cleared.)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_SM3*	AXIS_REF_SM3	When *bEnable* shifts to True and *bBusy* is False.

***Note:**

AXIS_REF_SM3 (FB): All function blocks contain this variable, which works as the starting program for function blocks.

- **Function**

  - When *bEnable* shifts to True, the values of *lrTorque*, *dwTorqueRamp* and *lrVelocity* of the DMC_TorqueControl instruction are sent to the servo for torque control.

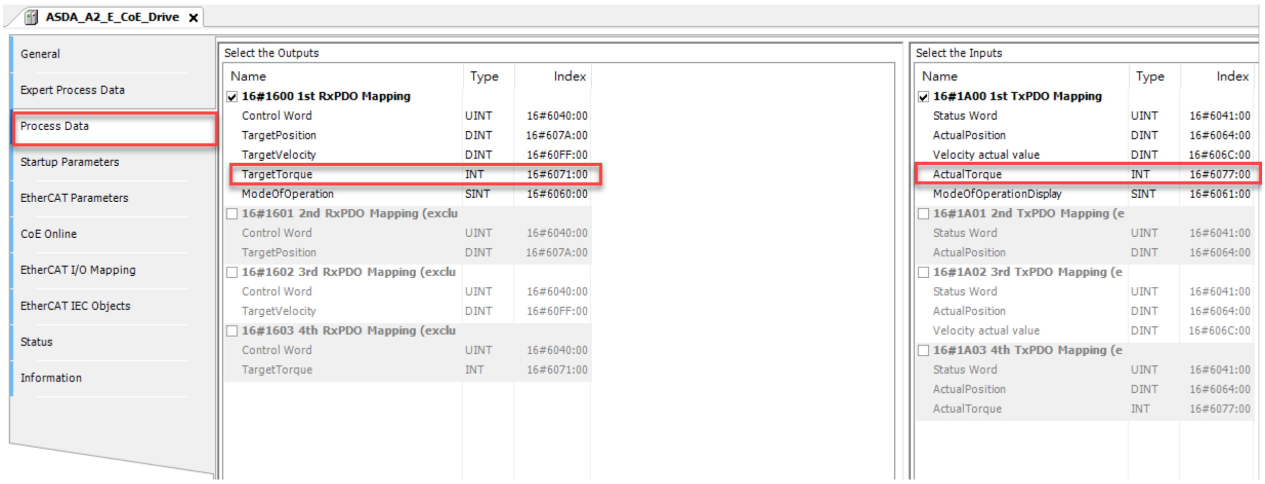
  - When *bEnable* is False, set the target torque *lrTorque* to 0 to make the axis decelerate to a stop. The instruction execution is completed when the axis decelerates to a stop and *bBusy* shifts to False.

  - Ensure that the axis is in Standstill state before instruction execution.

  - The servo will perform an immediate stop if SMC_SetControllerMode interrupts DMC_TorqueControl during instruction execution. Please DO NOT do so.

  - Only one DMC_TorqueControl instruction is allowed to execute at a time. If the second DMC_TorqueControl instruction is also executed at the same moment, an error will occur.

  - When the DMC_TorqueControl instruction is executed, 0x6071 (Target Torque), 0x6077 (Actual Torque), 0x6060 (ModeOfOperation) and 0x6061 (ModeOfOperationDisplay) OD must be included in the slave PDO mapping data. Otherwise, an error will occur.



Select the Outputs

Name	Type	Index
☑ 16#1600 1st RxPDO Mapping		
Control Word	UINT	16#6040:00
TargetPosition	DINT	16#607A:00
TargetVelocity	DINT	16#60FF:00
TargetTorque	INT	16#6071:00
ModeOfOperation	SINT	16#6060:00
□ 16#1601 2nd RxPDO Mapping (exclu		
Control Word	UINT	16#6040:00
TargetPosition	DINT	16#607A:00
□ 16#1602 3rd RxPDO Mapping (exclu		
Control Word	UINT	16#6040:00
TargetVelocity	DINT	16#60FF:00
□ 16#1603 4th RxPDO Mapping (exclu		
Control Word	UINT	16#6040:00
TargetTorque	INT	16#6071:00

Select the Inputs

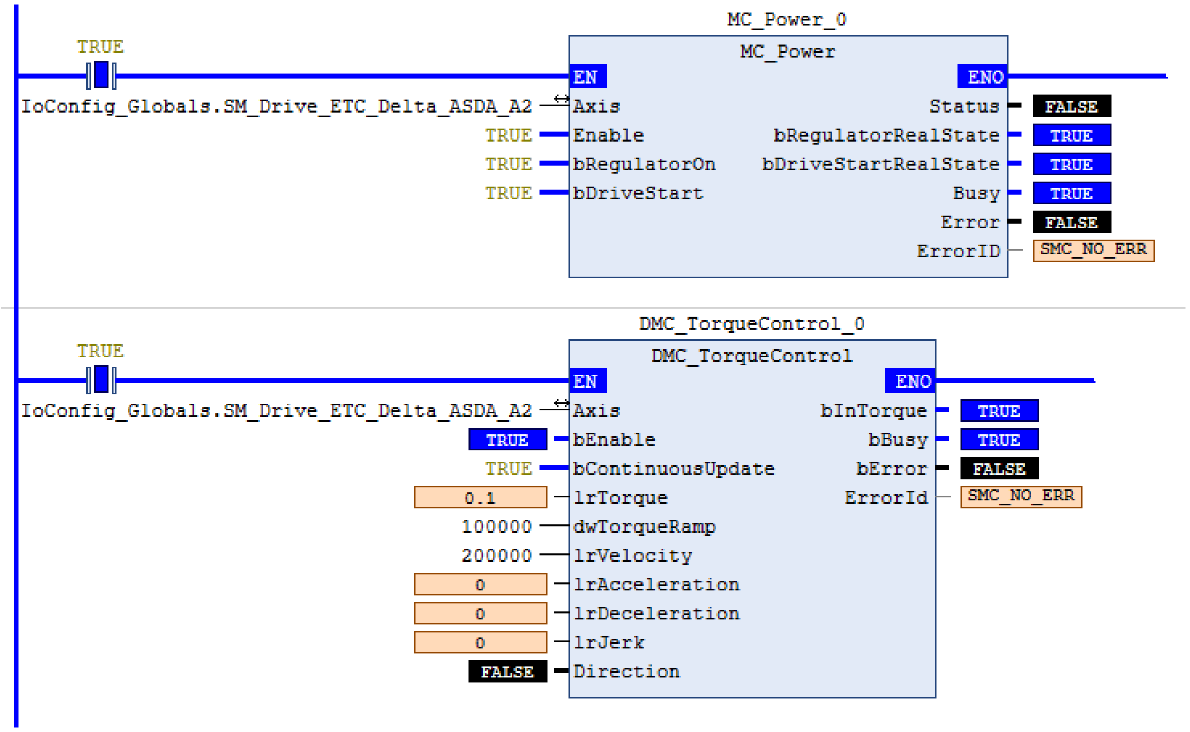
Name	Type	Index
☑ 16#1A00 1st TxPDO Mapping		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
Velocity actual value	DINT	16#606C:00
ActualTorque	INT	16#6077:00
ModeOfOperationDisplay	SINT	16#6061:00
□ 16#1A01 2nd TxPDO Mapping (e		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
□ 16#1A02 3rd TxPDO Mapping (e		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
Velocity actual value	DINT	16#606C:00
□ 16#1A03 4th TxPDO Mapping (e		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
ActualTorque	INT	16#6077:00

- **Troubleshooting**

  - If an error occurs during the execution of the instruction, *bError will* change to True. You can refer to *ErrorID* (Error Code) to address the problem.

- **Programming Example**

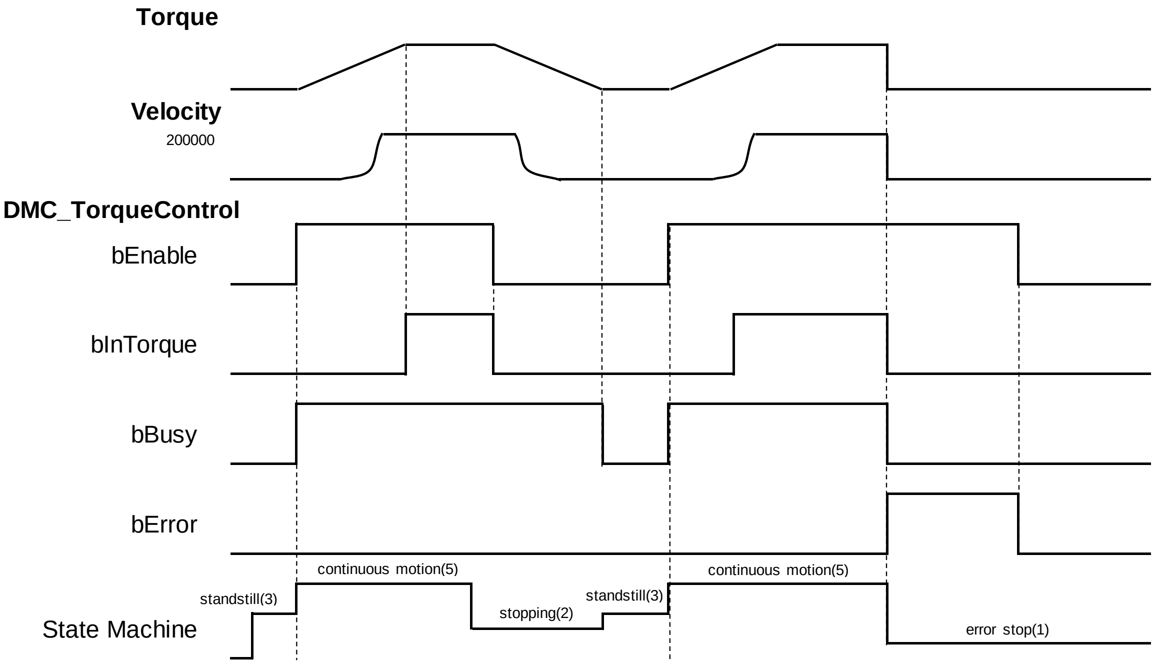
  - This example shows the motion behavior performed by DMC_TorqueControl.



MC_Power_0
MC_Power
TRUE
EN — ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 — Axis — Status — FALSE
TRUE — Enable — bRegulatorRealState — TRUE
TRUE — bRegulatorOn — bDriveStartRealState — TRUE
TRUE — bDriveStart — Busy — TRUE
Error — FALSE
ErrorID — SMC_NO_ERR

DMC_TorqueControl_0
DMC_TorqueControl
TRUE
EN — ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 — Axis — bInTorque — TRUE
TRUE — bEnable — bBusy — TRUE
TRUE — bContinuousUpdate — bError — FALSE
0.1 — lrTorque — ErrorId — SMC_NO_ERR
100000 — dwTorqueRamp
200000 — lrVelocity
0 — lrAcceleration
0 — lrDeceleration
0 — lrJerk
FALSE — Direction

- **Timing Diagram**



- After DMC_TorqueControl is started, the servo starts to run according to the input settings of the instruction for the target torque *lrTorque*, change rate of the torque *dwTorqueRamp* and maximum velocity *lrVelocity*.

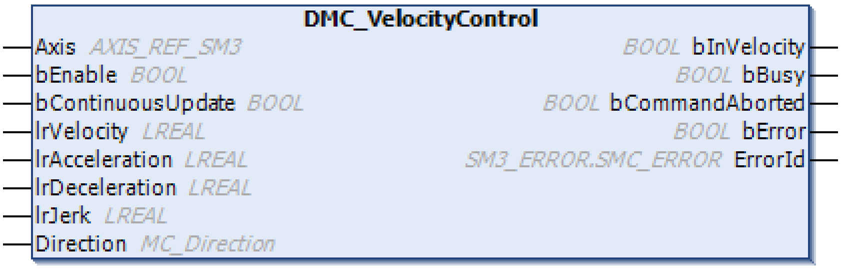
- After *bEnable* of DMC_TorqueControl shifts to False, the axis starts to decelerate till it stops. When the axis decelerates to a stop, *bBusy* shifts to False.

- An error occurs on the axis while DMC_TorqueControl has been executed for a period of time after being started one more time. At the moment, the axis performs an immediate stop for the error and then the instruction will report an error.

- **Supported Products**
  - AX-308E

2.3.1.2 DMC_VelocityControl

DMC_VelocityControl performs a velocity control on a specified axis in the CSV speed mode with the specified behavior and an average velocity.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>DMC_VelocityControl</td><td>DMC_VelocityControl
Axis AXIS_REF_SM3
bEnable BOOL
bContinuousUpdate BOOL
lrVelocity LREAL
lrAcceleration LREAL
lrDeceleration LREAL
lrJerk LREAL
Direction MC_Direction
BOOL bInVelocity
BOOL bBusy
BOOL bCommandAborted
BOOL bError
SM3_ERROR.SMC_ERROR ErrorId</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">DMC_VelocityControl (
Axis :=,
bEnable :=,
bContinuousUpdate :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
lrJerk :=,
Direction :=,
bInVelocity =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>) ;</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *bEnable* changes from FALSE to TRUE.	BOOL	True/False (False)	-
bContinuousUpdate	The target velocity can be updated continuously when *bContinuousUpdate* is True.	BOOL	True/False (False)	When *bEnable* shifts to True and *Busy* is False.
lrVelocity	Target velocity. (Unit: user unit/s)	LREAL	Positive number (0)	When *bEnable* shifts to True and *Busy* is False.
lrAcceleration	Acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bEnable* shifts to True and *Busy* is False.
lrDeceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bEnable* shifts to True and *Busy* is False.

Name	Function	Data type	Setting value (Default value)	Timing for updating
lrJerk	Jerk value. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bEnable* shifts to True and *Busy* is False.
Direction	Specifies the motion direction of the servo motor.	MC_ DIRECTION *1	3：fastest 2：current 1：positive 0：shortest -1：negative (current) *2	When *bEnable* shifts to True and *Busy* is False.

***Note:**

1. MC_DIRECTION: Enumeration (ENUM)
2. The options fastest, current and shortest are only for the rotary axis.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bInVelocity	True when the specified target velocity is reached.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NO_ERROR)

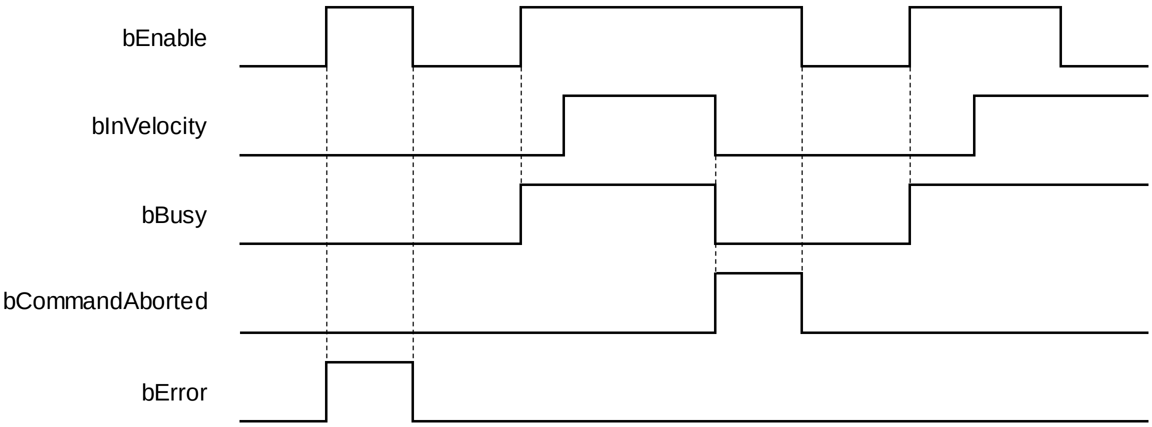
***Note:**

DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bInVelocity	● When the specified target velocity is reached.	● When *bCommandAborted* shifts to True. ● When *bContinuousUpdate* is True. and *lrVelocity* value is changed. ● When *bEnable* shifts to False. ● When *bError* shifts to Ture.
bBusy	● When *bEnable* shifts to True.	● When *bCommandAborted* shifts to True. ● When the axis decelerates to a stop after *bEnable* shifts to False. ● When *bError* shifts to True.
bCommandAborted	● When this instruction is aborted by another instruction. ● When this instruction is aborted via MC_Stop instruction.	● When *bEnable* shifts to False.
bError	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded)	● When *bEnable* shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_SM3*	AXIS_REF_SM3	When *bEnable* shifts to True and *bBusy* is False.

***Note:**

AXIS_REF_SM3 (FB): All function blocks contain this variable, which works as the starting program for function blocks.

- **Function**

  - The instruction performs speed control with specified target velocity (*lrVelocity*), acceleration rate (*lrAcceleration*), deceleration rate (*lrDeceleration*) and Jerk value (*lrJerk*) when *bEnable* changes to True.

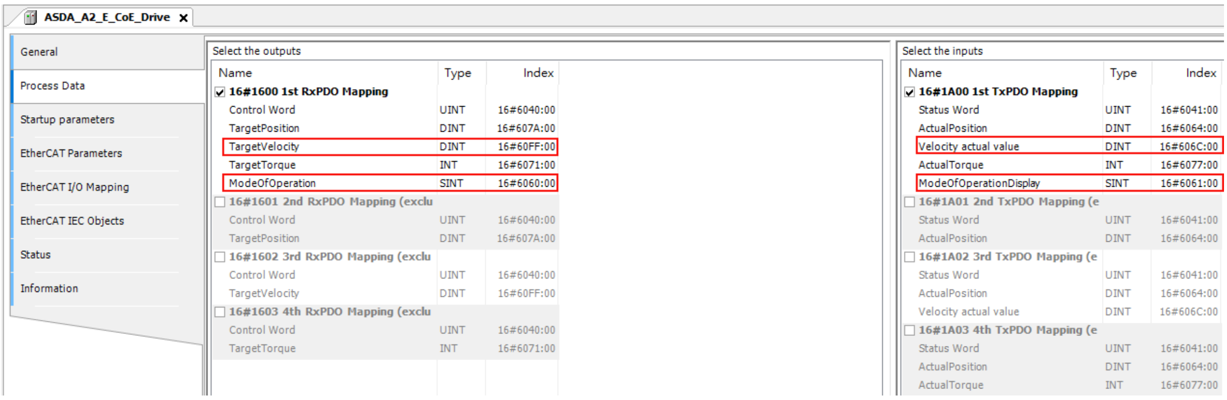
    - You can execute another motion instruction to abort the ongoing motion of MC_VelocityControl. But the servo is in CSV mode and the control mode will not be switched to CSP control mode.

    - When the instruction is aborted by another instruction, the output *bInVelocity* shifts to False and *bCommandAborted* shifts to True.

    - When the input *bContinuousUpdate* of the instruction shifts to True and the target velocity is given a new value, the axis velocity will update to the new speed.

    - When *bEnable* shifts to False, the instruction makes the axis decelerate to a stop and the mode is switched to CSP control mode.

    - When the instruction is executed, 0x60FF (Target Velocity), 0x606C (Velocity actual value), 0x6060 (ModeOfOperation) and 0x6061 (ModeOfOperationDisplay) OD must be included in the slave PDO mapping data. Otherwise, the servo will not be able to work.



Select the outputs

Name	Type	Index
☑ 16#1600 1st RxPDO Mapping		
Control Word	UINT	16#6040:00
TargetPosition	DINT	16#607A:00
TargetVelocity	DINT	16#60FF:00
TargetTorque	INT	16#6071:00
ModeOfOperation	SINT	16#6060:00
□ 16#1601 2nd RxPDO Mapping (exclu		
Control Word	UINT	16#6040:00
TargetPosition	DINT	16#607A:00
□ 16#1602 3rd RxPDO Mapping (exclu		
Control Word	UINT	16#6040:00
TargetVelocity	DINT	16#60FF:00
□ 16#1603 4th RxPDO Mapping (exclu		
Control Word	UINT	16#6040:00
TargetTorque	INT	16#6071:00

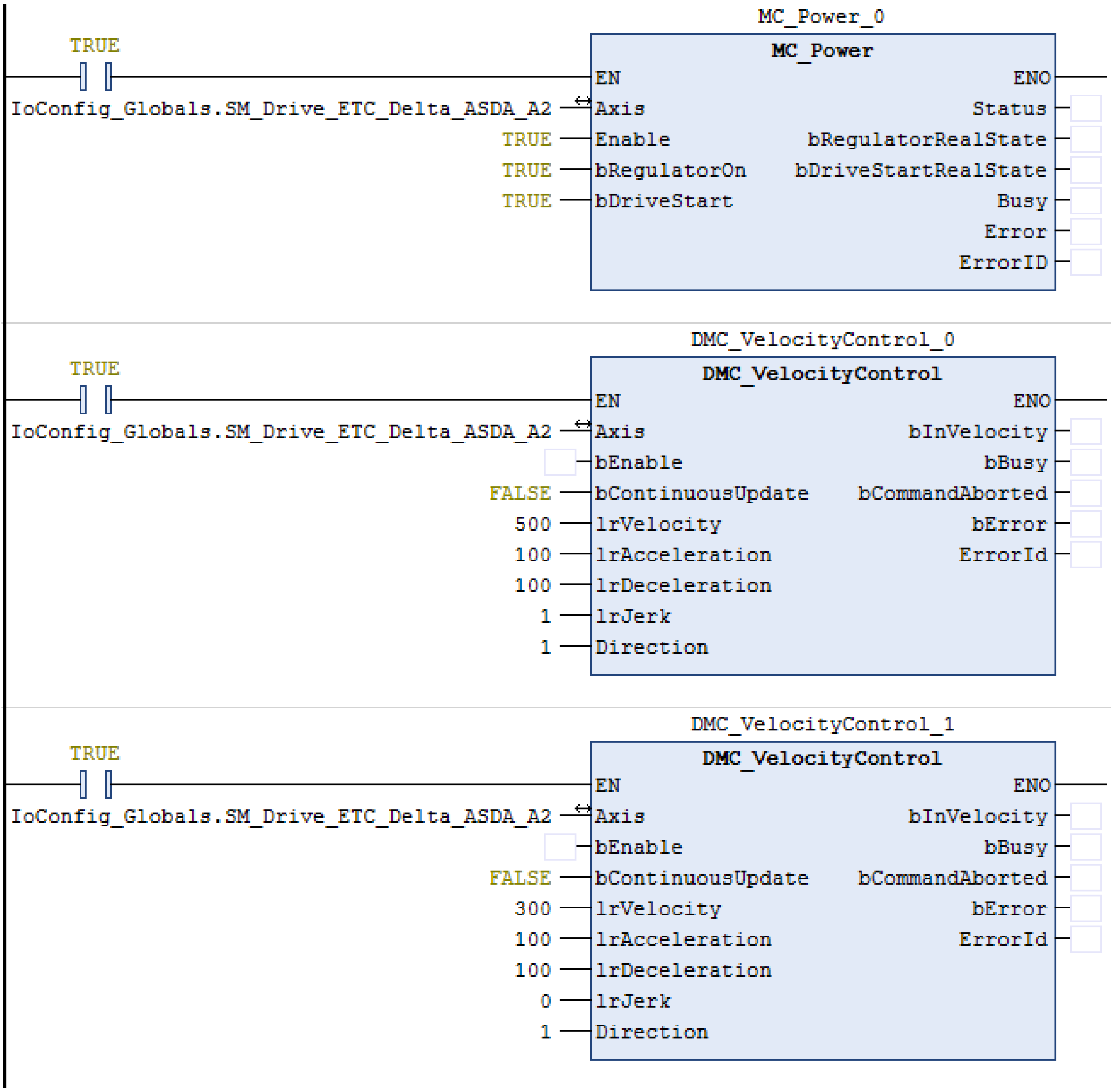
Select the inputs

Name	Type	Index
☑ 16#1A00 1st TxPDO Mapping		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
Velocity actual value	DINT	16#606C:00
ActualTorque	INT	16#6077:00
ModeOfOperationDisplay	SINT	16#6061:00
□ 16#1A01 2nd TxPDO Mapping (e		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
□ 16#1A02 3rd TxPDO Mapping (e		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
Velocity actual value	DINT	16#606C:00
□ 16#1A03 4th TxPDO Mapping (e		
Status Word	UINT	16#6041:00
ActualPosition	DINT	16#6064:00
ActualTorque	INT	16#6077:00

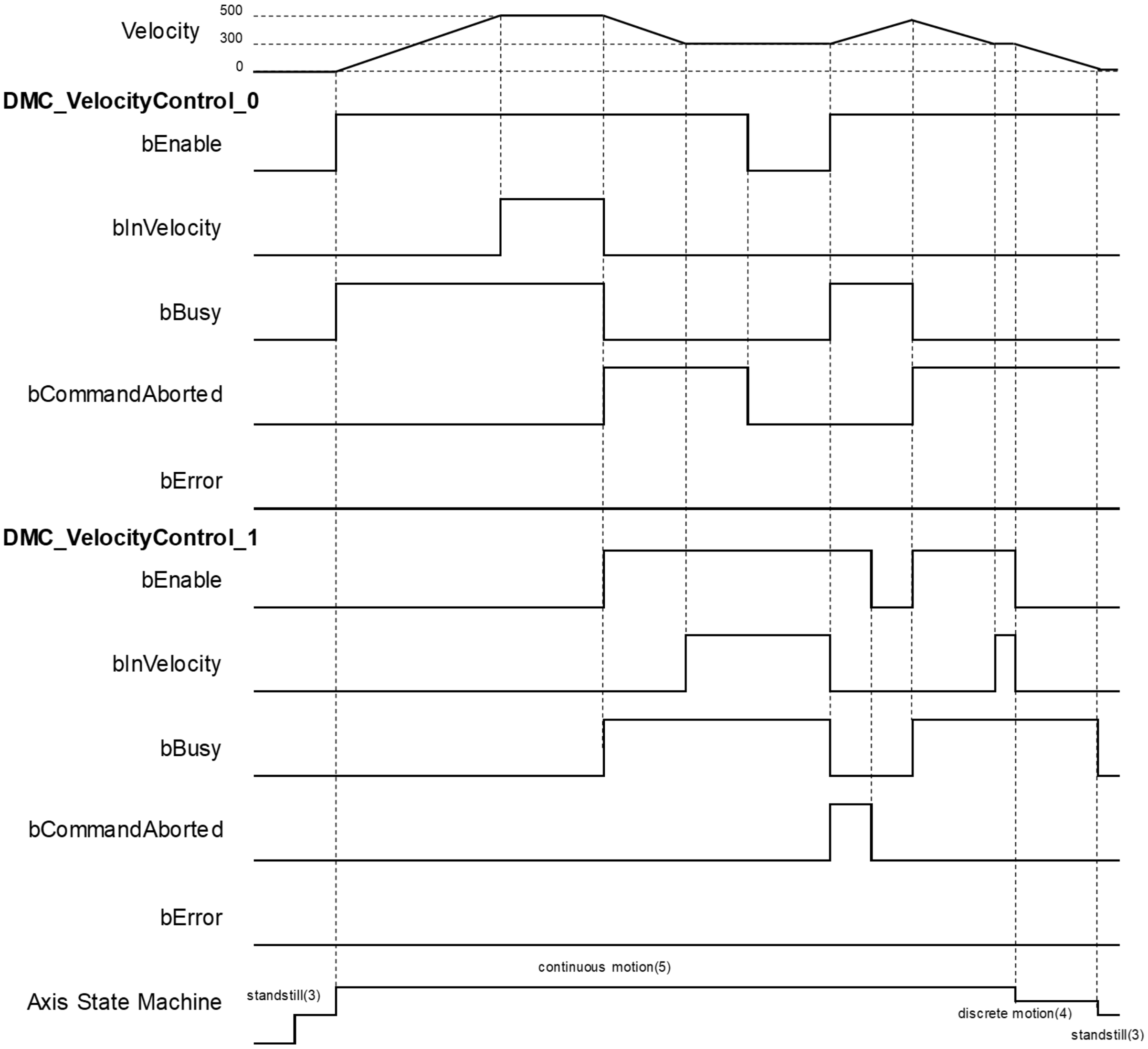
- **Troubleshooting**

  - If an error occurs during the execution of the instruction, *bError w*ill change to True. You can refer to *ErrorID* (Error Code) to address the problem.

- **Programming Example**
  - The example shows the motion behavior performed by DMC_VelocityControl.



- **Timing Diagram**



- When *bEnable* of DMC_VelocityControl_0 changes to True, the instruction controls the axis to reach the specified target velocity 500. When it reaches 500, *bInVelocity* of DMC_VelocityControl_0 changes to True.
- When *bEnable* of DMC_VelocityControl_1 changes to True, DMC_VelocityControl_0 is interrupted and *bInVelocity* of the instruction changes to False and *bCommandAborted* changes to True.
- The DMC_VelocityControl_1 instruction decelerates the axis to the velocity 300. When 300 is reached, *bInVelocity* of DMC_VelocityControl_1 will change to True and remain in this status as long as the velocity is not changed.
- When *bEnable* of DMC_VelocityControl_0 changes to False, *bCommanAborted* changes to False.
- When DMC_VelocityControl_0 is started again through changing *bEnable* of DMC_VelocityControl_0 to True, DMC_VelocityControl_0 will be aborted and the axis will accelerate to 500.
- If *bEnable* of DMC_VelocityControl_1 changes from False to True again when the target velocity of DMC_VelocityControl_0 has not been reached yet, DMC_VelocityControl_0 will be aborted. In this case, the axis will decelerate again without reaching the target velocity 500 of DMC_VelocityControl_0.
- *bInVelocity* of DMC_VelocityControl_1 changes to True when the target velocity of DMC_VelocityControl_1 is reached.

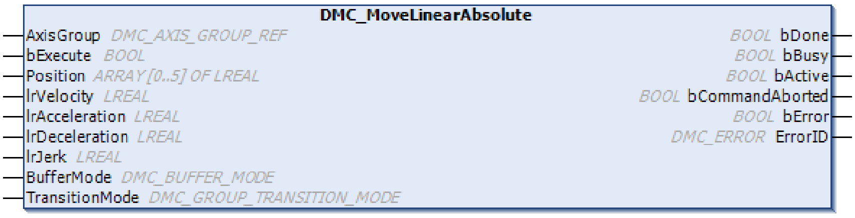
- ◆ When *bEnable* of DMC_VelocityControl_1 changes to True in the next cycle, the axis starts to decelerate to a stop and then *bBusy* of DMC_VelocityControl_1 changes to False.

- **Supported Products**

  - ■ AX-308E

2.3.1.3 DMC_MoveLinearAbsolute

DMC_MoveLinearAbsolute controls a specified axis group to perform the absolute linear interpolation for a specified absolute position.

FB/FC	Instruction	Graphic expression
FB	DMC_MoveLinearAbsolute	

ST expression

```
DMC_MoveLinearAbsolute_instance (
AxisGroup:= ,
bExecute:= ,
Position:= ,
lrVelocity:= ,
lrAcceleration:= ,
lrDeceleration:= ,
lrJerk:= ,
BufferMode:= ,
TransitionMode:= ,
bDone=> ,
bBusy=> ,
bActive=> ,
bCommandAborted=> ,
bError=> ,
ErrorID=> ) ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
Position	Specify the absolute target position for each axis in the specified axis group. (Unit: user unit)	LREAL[6]	[_, _, _, _, _, _] Positive number or negative number ([0, 0, 0, 0, 0, 0])	When *bExecute* shifts to True.
lrVelocity	Specify the target velocity for the specified axis group. (Unit: user unit/s)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrAcceleration	Specify the acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
lrDeceleration	Specify the deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrJerk	Specify the jerk. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bExecute* shifts to True.
BufferMode	Specify a buffer mode for the instruction [*1]	DMC_ BUFFER_ MODE	0: Aborting 1: Buffered 2: BlendingLow 3: BlendingPrevious 4: BlendingNext 5: BlendingHigh (0)	When *bExecute* shifts to True.
TransitionMode	Specify a transition mode for the instruction[*2].	DMC_ GROUP_ TRANSITION_ MODE	0: None 10: Overlap (0)	When *bExecute* shifts to True.

***Note:**

1. Refer to AX-3 Series Operation Manual for details on *BufferMode*.
2. Refer to AX-3 Series Operation Manual for details on *TransitionMode*.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bDone	True when the absolute positioning is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bActive	True when the instruction is controlling axes.	BOOL	True/False (False)
bCommand Aborted	True when the instruction execution is aborted.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR[*]	DMC_ERROR (DMC_NoError)

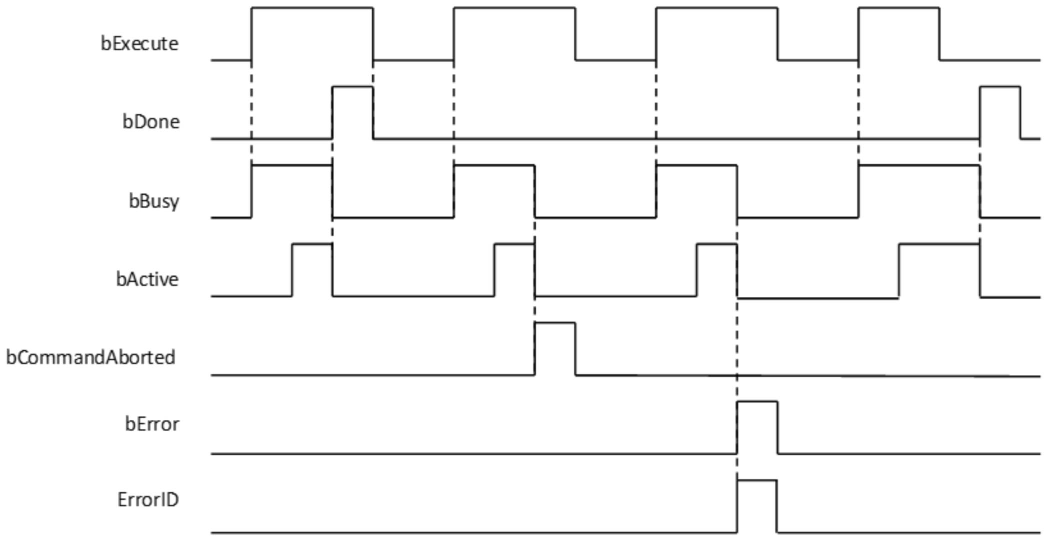
***Note**: DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the absolute positioning is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one cycle when

Name	Timing for shifting to True	Timing for shifting to False
		bExecute is False but *bDone* changes to True.
bBusy	● When *bExecute* changes to True.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bActive	● When axes start being controlled by the instruction.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● *bActive* will change to False after remaining True for at least one cycle when *bExecute* is False but *bActive* changes to True.
bCommand Aborted	● When the instruction is aborted by another instruction *BufferMode* of which is set to Aborting. ● When the instruction is aborted by MC_Stop. ● When the instruction is aborted by DMC_GroupStop.	● When *bExecute* shifts to False. ● *bCommandAborted* will change to False after remaining True for one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError/ErrorID	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared)

■ **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group.	DMC_AXIS_GROUP_REF*	DMC_AXIS_ GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**

  - The instruction supports the absolute linear interpolation of maximum six axes, where the six axes can simultaneously start, stop as well as reach the specified absolute target position.
  - At least one axis is needed for the linear interpolation. An error will occur if there is a travel distance for the axis which is not set.

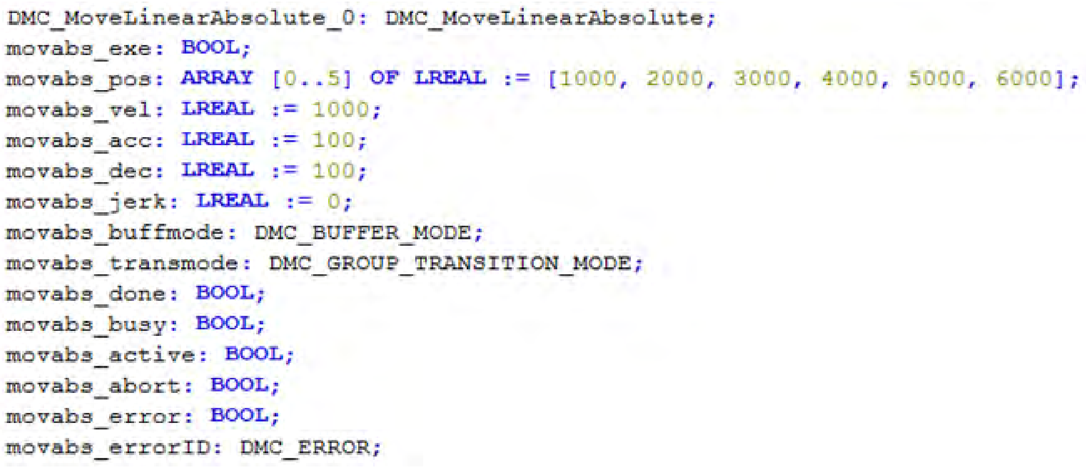
- **Troubleshooting**

  - When an error occurs in the instruction execution or the axis group enters GroupErrorstop state, *bError* changes to True and axes stops running. To confirm current error state, see the error code in *ErrorID*.
  - For error codes and corresponding trouble shootings, refer to **Appendices** for error code descriptions.

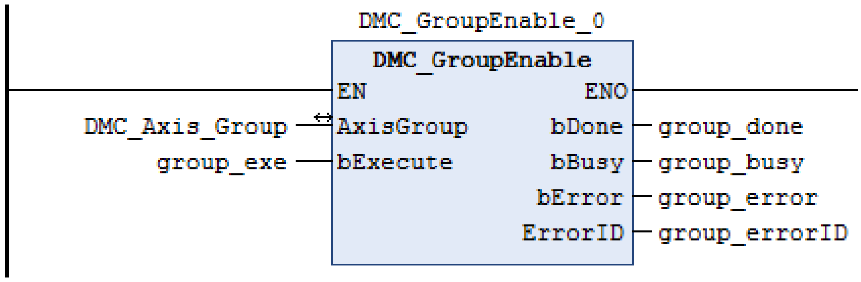
- **Programming Example**

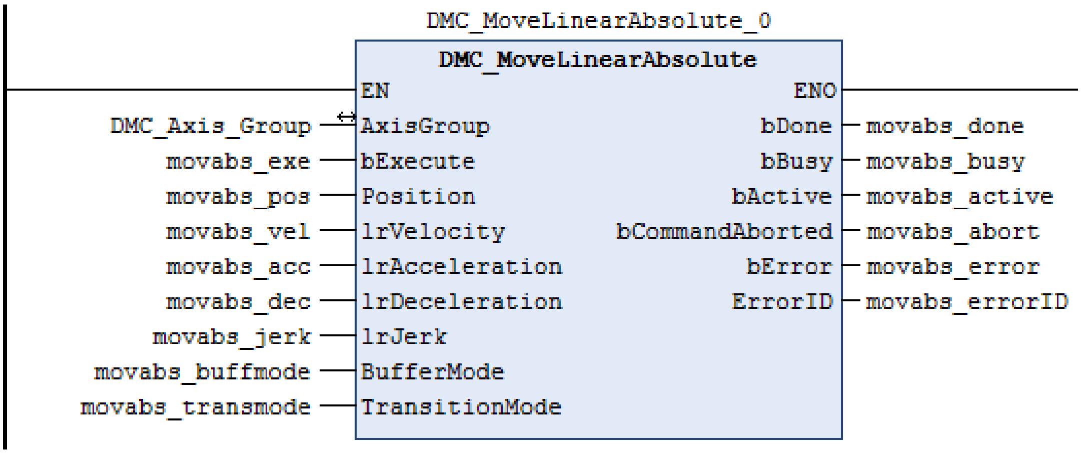
  - In this example, the path for the six-axis absolute linear interpolation is planned and the six axes simultaneously reach the target position through traveling an absolute distance from current positon.

Axis group	Target position
Axis1	1000
Axis2	2000
Axis3	3000
Axis4	4000
Axis5	5000
Axis6	6000



```
DMC_MoveLinearAbsolute_0: DMC_MoveLinearAbsolute;
movabs_exe: BOOL;
movabs_pos: ARRAY [0..5] OF LREAL := [1000, 2000, 3000, 4000, 5000, 6000];
movabs_vel: LREAL := 1000;
movabs_acc: LREAL := 100;
movabs_dec: LREAL := 100;
movabs_jerk: LREAL := 0;
movabs_buffmode: DMC_BUFFER_MODE;
movabs_transmode: DMC_GROUP_TRANSITION_MODE;
movabs_done: BOOL;
movabs_busy: BOOL;
movabs_active: BOOL;
movabs_abort: BOOL;
movabs_error: BOOL;
movabs_errorID: DMC_ERROR;
```



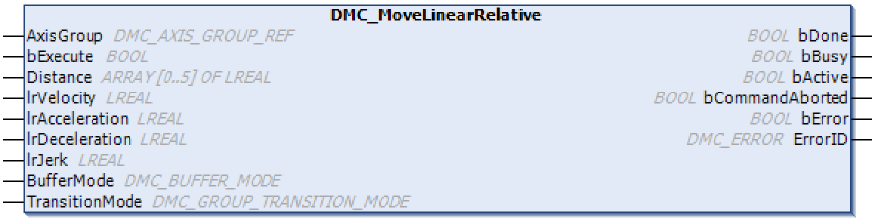


- When moveabs_exe (*bExecute*) changes to True, DMC_MoveLinearAbsolute starts to perform the absolute linear interpolation for six axes.
- When moveabs_done (*bDone*) changes to True, moveabs_busy (*bBusy*) changes to False, which means the specified absolute positioning (1000, 2000, 3000, 4000, 5000, 6000) is completed.
- moveabs_exe (*bExecute*) is switched to False after the absolute linear interpolation is completed. Then moveabs_done (*bDone*) will change to False automatically.
- If moveabs_exe (*bExecute*) is set to True again for the absolute linear interpolation, then no axes will move for positioning.

- **Supported Products**
  - AX-308E

2.3.1.4 DMC_MoveLinearRelative

DMC_MoveLinearRelative controls a specified axis group to perform the relative linear interpolation.

FB/FC	Instruction	Graphic expression
FB	DMC_MoveLinearRelative	 DMC_MoveLinearRelative AxisGroup DMC_AXIS_GROUP_REF bExecute BOOL Distance ARRAY [0..5] OF LREAL lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL lrJerk LREAL BufferMode DMC_BUFFER_MODE TransitionMode DMC_GROUP_TRANSITION_MODE BOOL bDone BOOL bBusy BOOL bActive BOOL bCommandAborted BOOL bError DMC_ERROR ErrorID

ST expression

```
DMC_MoveLinearRelative_instance (
AxisGroup: =,
bExecute: =,
Distance: =,
lrVelocity: =,
lrAcceleration: =,
lrDeceleration: =,
lrJerk: =,
BufferMode: =,
TransitionMode: =,
bDone=>,
bBusy=>,
bActive=>,
bCommandAborted=>,
bError=>,
ErrorID=>)   ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
Distance	Specify the travel distance for each axis in the specified axis group. (Unit: user unit)	LREAL[6]	[_, _, _, _, _, _] Positive number, negative number or 0 ([0, 0, 0, 0, 0, 0])	When *bExecute* shifts to True.
lrVelocity	Specify the target velocity for the axis group. (Unit: user unit/s)	LREAL	Positive number (0)	When *bExecute* shifts to True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
lrAcceleration	Specify the acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrDeceleration	Specify the deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrJerk	Specify the jerk. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bExecute* shifts to True.
BufferMode	Specify a buffer mode for the instruction.*1	DMC_ BUFFER_ MODE	0: Aborting 1: Buffered 2: BlendingLow 3: BlendingPrevious 4: BlendingNext 5: BlendingHigh (0)	When *bExecute* shifts to True.
TransitionMode	Specify a transition mode for the instruction *2	DMC_ GROUP_ TRANSITION_ MODE	0: None 10: Overlap (0)	When *bExecute* shifts to True.

***Note:**

1. Refer to AX-3 Series Operation Manual for details on BufferMode.
2. Refer to AX-3 Series Operation Manual for details on TransitionMode.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bDone	True when the relative positioning is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bActive	True when the instruction is controlling axes.	BOOL	True/False (False)
bCommand Aborted	True when the instruction execution is aborted.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

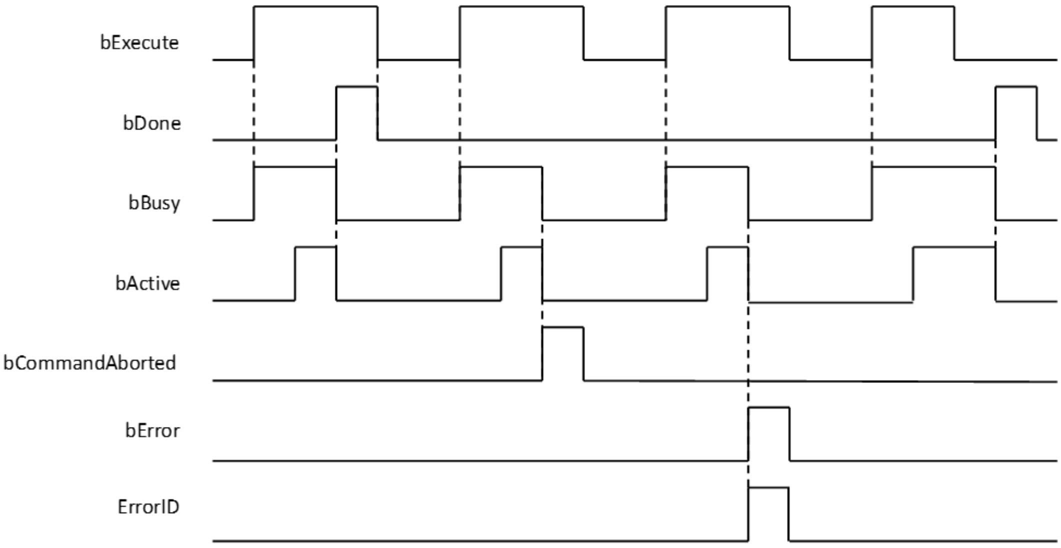
***Note:** DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the relative positioning is completed.	● When *bExecute* shifts to False. ● When *bExecute* is False but *bDone*

Name	Timing for shifting to True	Timing for shifting to False
		shifts to True, *bDone* will remain True for one cycle and then change to False.
bBusy	● When *bExecute* changes to TRUE.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bActive	● When axes start being controlled by the instruction.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● *bActive* will change to False after remaining True for at least one cycle when *bExecute* is False but *bActive* changes to True.
bCommand Aborted	● When the instruction is aborted by another instruction whose *BufferMode* is set to Aborting. ● When the instruction is aborted by MC_Stop. ● When the instruction is aborted by DMC_GroupStop.	● When bExecute shifts to False. ● *bCommandAborted* will change to False after remaining True for at least one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError/ErrorID	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded in *ErrorID*)	● When bExecute shifts from True to False. (Error code is cleared)

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Output range	Name
AxisGroup	Specify the axis group.	DMC_AXIS_GROUP_REF*	DMC_AXIS_GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

*Note:

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**

  - The instruction supports the relative linear interpolation of maximum six axes, where the six axes can simultaneously start, stop as well as reach the specified target relative position.
  - At least one axis is needed for the linear interpolation. An error will occur if there is a travel distance for the axis which is not set.

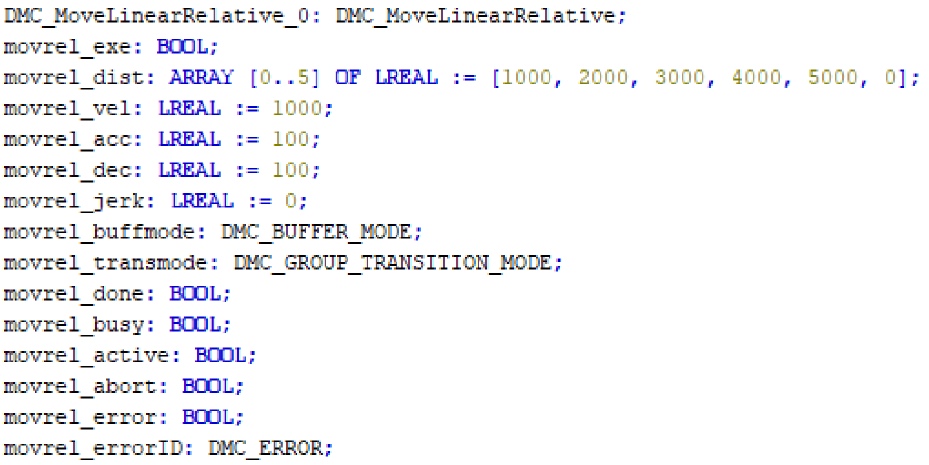
- **Troubleshooting**

  - When an error occurs in the instruction execution or the axis group enters GroupErrorstop state, *bError* changes to True and axes stops running. To confirm current error state, see the error code in *ErrorID*.
  - For error codes and corresponding trouble shootings, refer to **Appendices** for error code descriptions.

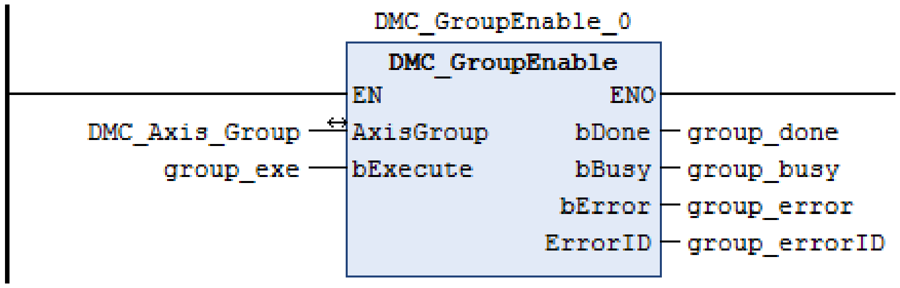
- **Programming Example**

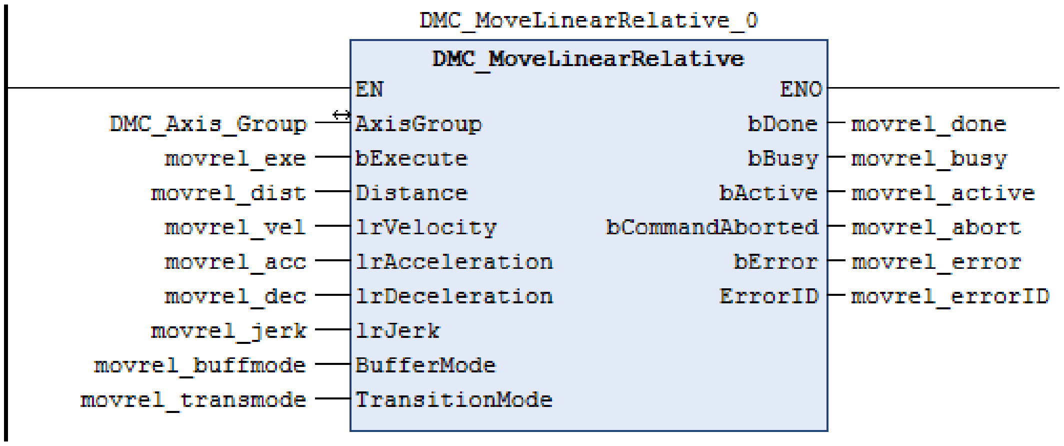
  - In this example, the path for the six-axis relative linear interpolation is planned and six axes simultaneously reach the target relative position through traveling a relative distance from current positon.

The number of each axis in the axis group	Current position	Relative distance	Target position
Axis 1	1000	1000	2000
Axis 2	1000	2000	3000
Axis 3	1000	3000	4000
Axis 4	1000	4000	5000
Axis 5	1000	5000	6000
Axis 6	1000	0	1000



```
DMC_MoveLinearRelative_0: DMC_MoveLinearRelative;
movrel_exe: BOOL;
movrel_dist: ARRAY [0..5] OF LREAL := [1000, 2000, 3000, 4000, 5000, 0];
movrel_vel: LREAL := 1000;
movrel_acc: LREAL := 100;
movrel_dec: LREAL := 100;
movrel_jerk: LREAL := 0;
movrel_buffmode: DMC_BUFFER_MODE;
movrel_transmode: DMC_GROUP_TRANSITION_MODE;
movrel_done: BOOL;
movrel_busy: BOOL;
movrel_active: BOOL;
movrel_abort: BOOL;
movrel_error: BOOL;
movrel_errorID: DMC_ERROR;
```





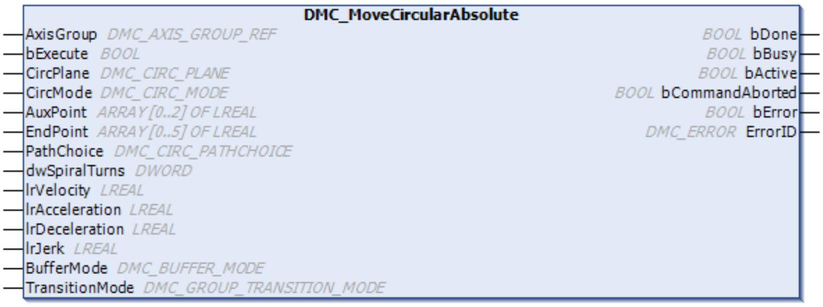
- When movrel_exe (*bExecute*) changes to True, DMC_GroupRelLinear starts to perform the relative linear interpolation for six axes.
- When movrel_done (*bDone*) changes to True, movrel_busy (*bBusy*) and movrel_abort (*bAborted*) change to False, which means the specified relative positioning (1000, 2000, 3000, 4000, 5000, 0) is completed.
- movrel_exe (*bExecute*) is switched to False after the relative linear interpolation is completed. Then movrel_done (*bDone*) will change to False automatically.
- If movrel_exe (*bExecute*) is set to True again, axes will perform the relative linear interpolation one more time to reach the target position (3000, 5000, 7000, 9000, 11000, 1000).
- When the target positioning is completed, movrel_done (*bDone*) changes to True again.

- **Supported Products**

  - AX-308E

2.3.1.5 DMC_MoveCircularAbsolute

DMC_MoveCircularAbsolute controls the axis group to perform circular or helical interpolation for a specified absolute target position.

FB/FC	Instruction	Graphic expression
FB	DMC_MoveCircularAbsolute	 DMC_MoveCircularAbsolute AxisGroup DMC_AXIS_GROUP_REF bExecute BOOL CircPlane DMC_CIRC_PLANE CircMode DMC_CIRC_MODE AuxPoint ARRAY [0..2] OF LREAL EndPoint ARRAY [0..5] OF LREAL PathChoice DMC_CIRC_PATHCHOICE dwSpiralTurns DWORD lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL lrJerk LREAL BufferMode DMC_BUFFER_MODE TransitionMode DMC_GROUP_TRANSITION_MODE BOOL bDone BOOL bBusy BOOL bActive BOOL bCommandAborted BOOL bError DMC_ERROR ErrorID

ST expression
DMC_MoveCircularAbsolute_instance (AxisGroup: = , bExecute: = , CircPlane: = , CircMode: = , AuxPoint: = , EndPoint: = , PathChoice: = , dwSpiralTurns: = , lrVelocity: = , lrAcceleration: = , lrDeceleration: = , lrJerk: = , BufferMode: = , TransitionMode: = , bDone=> , bBusy=> , bActive=> , bCommandAborted=> , bError=> , ErrorID=> ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
CircPlane	Specify the circular or helical plane. *1	DMC_CIRC_ PLANE	0: XY_plane 1: YZ_plane 2: ZX_plane (0)	When *bExecute* shifts to True.
CircMode	Specify the method for circular or helical interpolation. *1	DMC_CIRC_ MODE	0: Border 1: Center 2: Radius (0)	When *bExecute* shifts to True.
AuxPoint	Specify the auxiliary point data. *1	LREAL[3]	[_, _, _] Positive number, negative number or 0 ([0, 0, 0])	When *bExecute* shifts to True.
EndPoint	Specify the target position for each axis in the axis group. (Unit: user unit)	LREAL[6]	[_, _, _, _, _, _] Positive number, negative number or 0 ([0, 0, 0, 0, 0, 0])	When *bExecute* shifts to True.
PathChoice	Specify the circular or helical interpolation direction.	DMC_CIRC_ PATHCHOICE	0: Clockwise 1: CounterClockwise (0)	When *bExecute* shifts to True.
dwSpiralTurns	Specify the number of spiral turns.	DWORD	0~65535 (0)	When *bExecute* shifts to True.
lrVelocity	Specify the target velocity for the axis group. (Unit: user unit/s)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrAcceleration	Specify the acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrDeceleration	Specify the deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrJerk	Specify the jerk. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bExecute* shifts to True.
BufferMode	Specify a buffer mode for the instruction.*2	DMC_BUFFER_ MODE	0: Aborting 1: Buffered 2: BlendingLow 3: BlendingPrevious 4: BlendingNext 5: BlendingHigh (0)	When *bExecute* shifts to True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
TransitionMode	Specify a transition mode for the instruction. [*3]	DMC_GROUP_TRANSITION_MODE	0: None 10: Overlap (0)	When *bExecute* shifts to True.

***Note:**

1. Setting parameters *CircPlane*, *CircMode* and *AuxPoint*.

CircMode setting	Definition *CircMode – AuxPoint*	*CircPlane*		
		XY_Plane	YZ_Plane	ZX_Plane
		Actual input for *AuxPoint* [_, _, _]		
0	Three points – Absolute coordinate values for the border point (X_A, Y_A, Z_A)	Start point, end point and border point [X_A, Y_A, Z_A]		
1	A center point – Absolute coordinate values for the center point (C_X, C_Y)	[C_X, C_Y, N/A]	[N/A, C_X, C_Y]	[C_Y, N/A, C_X]
2	Radius – Radius (R)	[R, N/A, N/A]		

2. Refer to AX-3 Series Operation Manual for details on BufferMode.
3. Refer to AX-3 Series Operation Manual for details on TransitionMode.

● **Outputs**

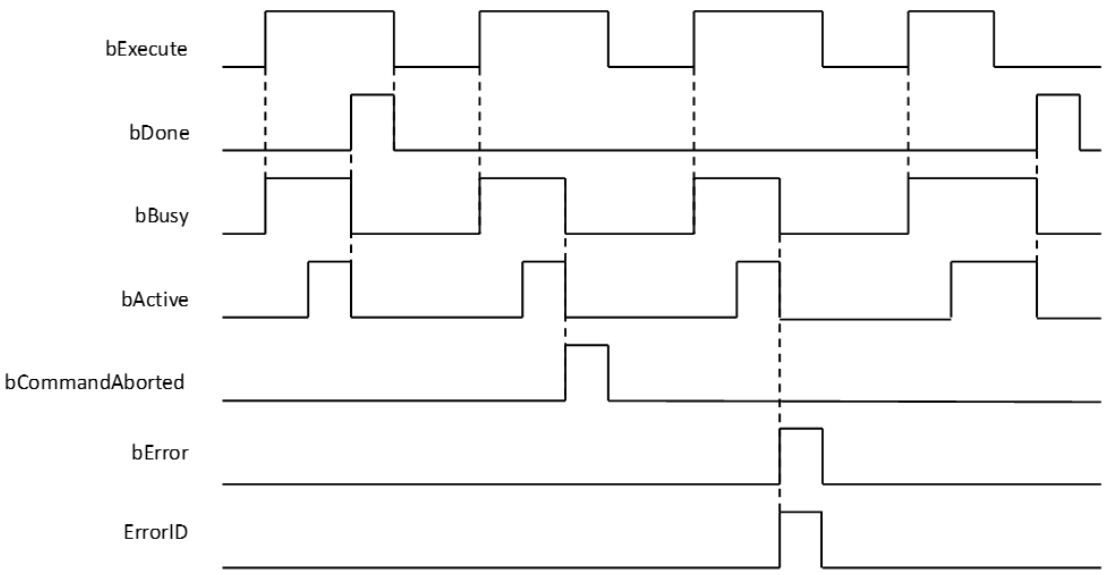
Name	Function	Data type	Output range (Default value)
bDone	True when the absolute positioning is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bActive	True when the instruction is controlling axes.	BOOL	True/False (False)
bCommand Aborted	True when the instruction execution is aborted.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR[*]	DMC_ERROR (DMC_NoError)

***Note:** DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the absolute positioning is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one cycle when *bExecute* is False but *bDone* changes to True.
bBusy	● When *bExecute* changes to TRUE.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bActive	● When axes start being controlled by the instruction.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● *bActive* will change to False after remaining True for at least one cycle when *bExecute* is False but *bActive* changes to True.
bCommand Aborted	● When the instruction is aborted by another instruction *BufferMode* of which is set to Aborting. ● When the instruction is aborted by MC_Stop. ● When the instruction is aborted by DMC_GroupStop.	● When *bExecute* shifts to False. ● *bCommandAborted* will change to False after remaining True for at least one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError/ErrorID	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded)	● When *bExecute* shifts from True to False. (Error code is cleared)

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group.	DMC_AXIS_GROUP_REF*	DMC_AXIS_GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**

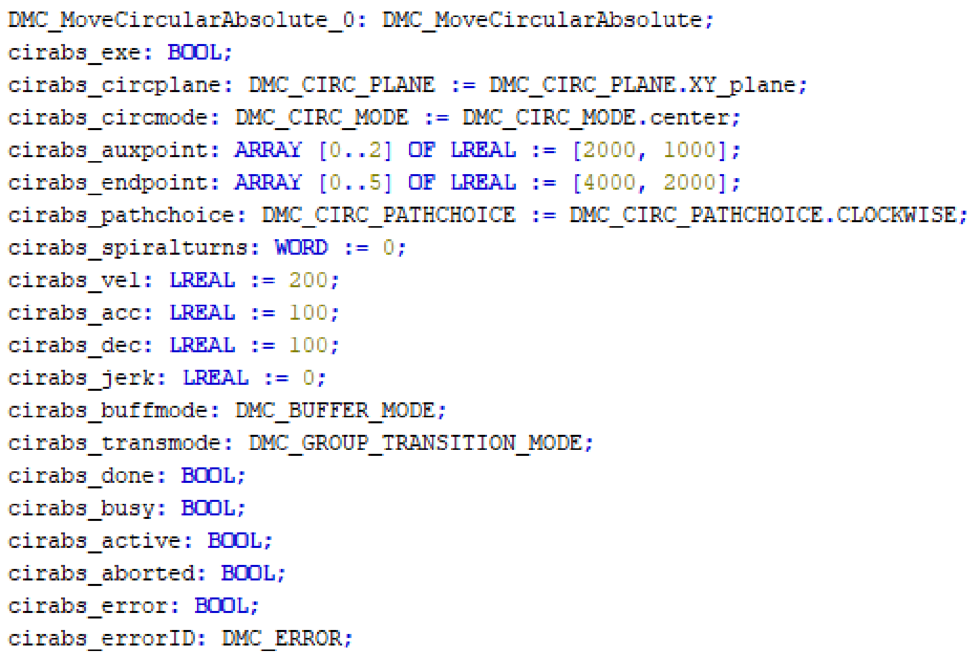
  - The instruction supports the absolute helical interpolation of maximum three axes, where the three axes can simultaneously start, stop as well as reach the specified absolute target position.
  - The instruction can be used to specify the circle drawing on the plane parallel to XY / YZ / ZX and set the height of the helix on Z / X / Y axis.
  - If the start point and end point for circular interpolation are set to the same point, use the center point mode (*CircMode* = Center) for the interpolation.
  - At least two axes are needed for circular interpolation. An error will occur if there is a travel distance for an axis which is not set.
  - When the start point and end point for circular interpolation are set as the same point, the instruction will perform the rotation for one complete circle.

- **Troubleshooting**

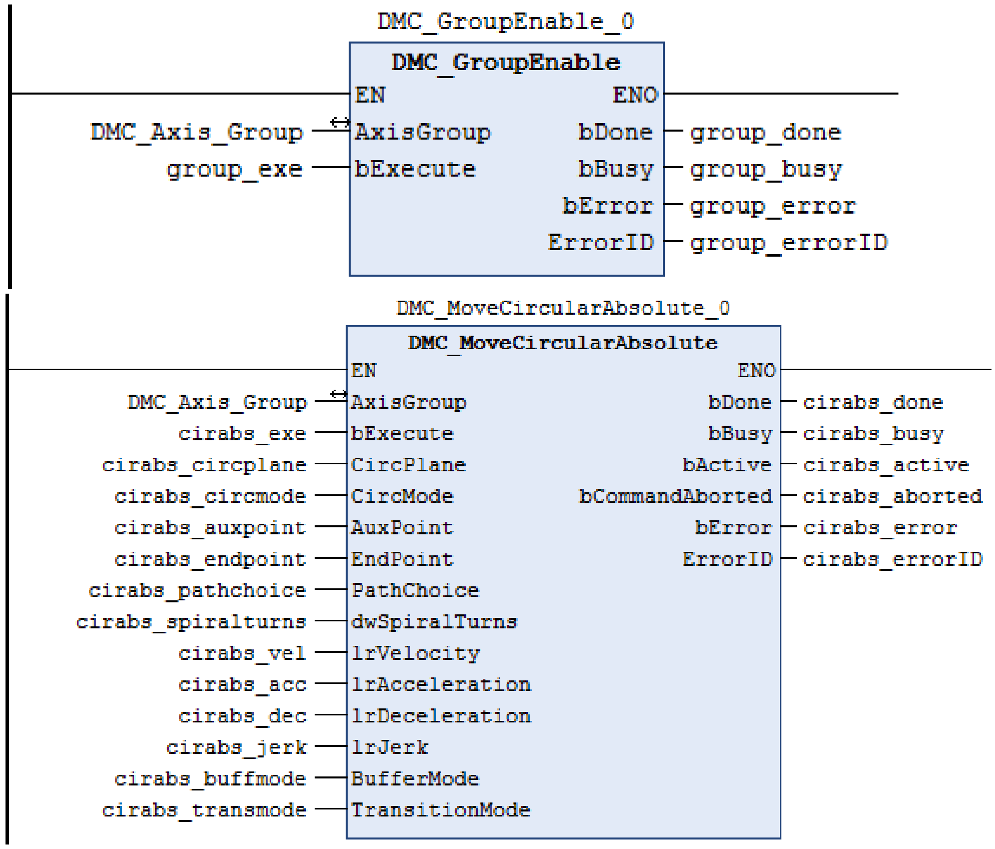
  - When an error occurs in the instruction execution or the axis group enters GroupErrorstop state, *bError* changes to True and axes stops running. To confirm current error state, see the error code in *ErrorID*.
  - For error codes and corresponding trouble shootings, refer to **Appendices** for error code descriptions.

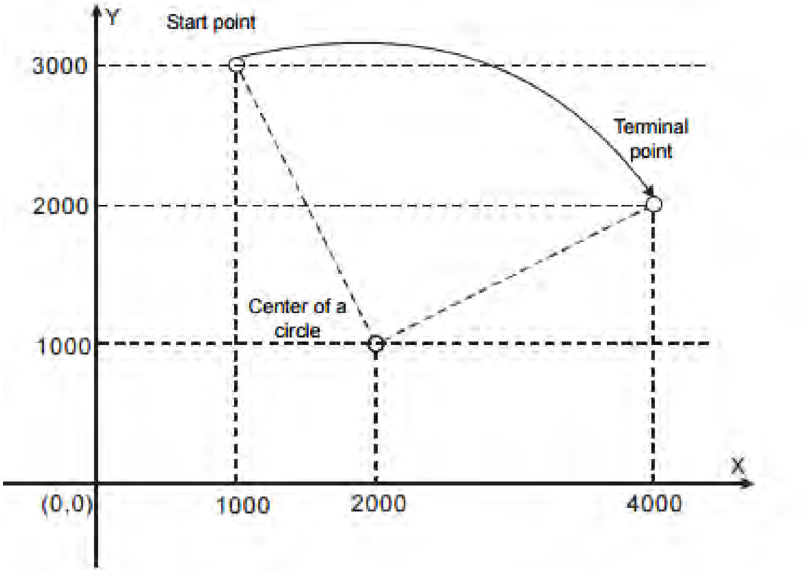
- **Programming Example**

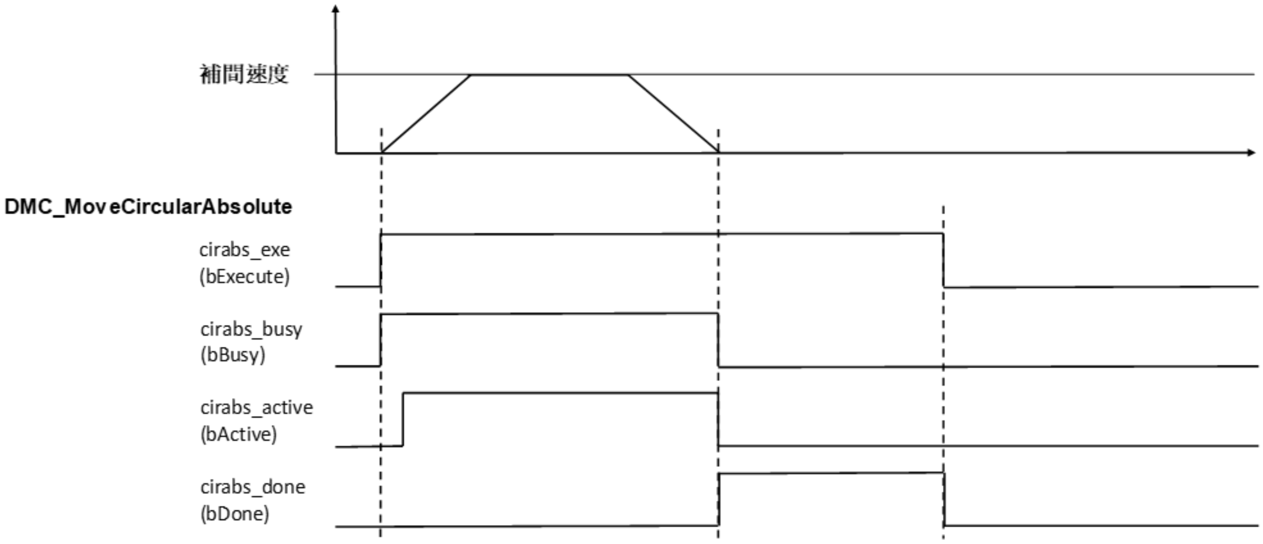
  - In this example, the instruction performs the circular interpolation from current positon (1000, 3000) until the absolute target position (4000, 2000) in the clockwise direction.



```
DMC_MoveCircularAbsolute_0: DMC_MoveCircularAbsolute;
cirabs_exe: BOOL;
cirabs_circplane: DMC_CIRC_PLANE := DMC_CIRC_PLANE.XY_plane;
cirabs_circmode: DMC_CIRC_MODE := DMC_CIRC_MODE.center;
cirabs_auxpoint: ARRAY [0..2] OF LREAL := [2000, 1000];
cirabs_endpoint: ARRAY [0..5] OF LREAL := [4000, 2000];
cirabs_pathchoice: DMC_CIRC_PATHCHOICE := DMC_CIRC_PATHCHOICE.CLOCKWISE;
cirabs_spiralturns: WORD := 0;
cirabs_vel: LREAL := 200;
cirabs_acc: LREAL := 100;
cirabs_dec: LREAL := 100;
cirabs_jerk: LREAL := 0;
cirabs_buffmode: DMC_BUFFER_MODE;
cirabs_transmode: DMC_GROUP_TRANSITION_MODE;
cirabs_done: BOOL;
cirabs_busy: BOOL;
cirabs_active: BOOL;
cirabs_aborted: BOOL;
cirabs_error: BOOL;
cirabs_errorID: DMC_ERROR;
```







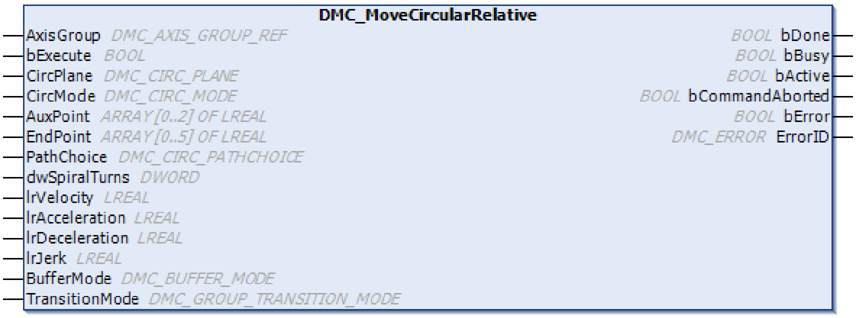
- When cirabs_exe (bExecute) changes to True, DMC_MoveCircularAbsolute performs the absolute positioning toward the terminal point (4000, 2000) from the start point (1000, 3000) in the clockwise direction.
- When cirabs_done (*bDone*) is True and cirabs_busy (*bBusy*) changes to False, which means the absolute target positioning (4000, 2000) is completed. When cirabs_exe (*bExecute*) is switched to False, cirabs_done (*bDone*) will change to False automatically.
- If cirabs_exe (*bExecute*) is set to True again, there will be no positioning motion any longer since the absolute target position has been reached.

Supported Products

- AX-308E

2.3.1.6 DMC_MoveCircularRelative

DMC_MoveCircularRelative controls the axis group to perform circular or helical interpolation for a specified relative target position.

FB/FC	Instruction	Graphic expression
FB	DMC_MoveCircular Relative	 DMC_MoveCircularRelative AxisGroup DMC_AXIS_GROUP_REF bExecute BOOL CircPlane DMC_CIRC_PLANE CircMode DMC_CIRC_MODE AuxPoint ARRAY [0..2] OF LREAL EndPoint ARRAY [0..5] OF LREAL PathChoice DMC_CIRC_PATHCHOICE dwSpiralTurns DWORD lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL lrJerk LREAL BufferMode DMC_BUFFER_MODE TransitionMode DMC_GROUP_TRANSITION_MODE BOOL bDone BOOL bBusy BOOL bActive BOOL bCommandAborted BOOL bError DMC_ERROR ErrorID

ST expression
DMC_MoveCircularRelative_instance (AxisGroup: = , bExecute: = , CircPlane: = , CircMode: = , AuxPoint: = , EndPoint: = , PathChoice: = , dwSpiralTurns: = , lrVelocity: = , lrAcceleration: = , lrDeceleration: = , lrJerk: = , BufferMode: = , TransitionMode: = , bDone=> , bBusy=> , bActive=> , bCommandAborted=> , bError=> , ErrorID=>) ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
CircPlane	Specify the circular or helical plane. *1	DMC_CIRC_PLANE	0: XY_plane 1: YZ_plane 2: ZX_plane (0)	When *bExecute* shifts to True.
CircMode	Specify the method for circular or helical interpolation. *1	DMC_CIRC_MODE	0: Border 1: Center 2: Radius (0)	When *bExecute* shifts to True.
AuxPoint	Specify the auxiliary point data. *1	LREAL[3]	[_, _, _] Positive number, negative number or 0 ([0, 0, 0])	When *bExecute* shifts to True.
EndPoint	Specify the target position for each axis in the axis group. (Unit: user unit)	LREAL[6]	[_, _, _, _, _, _] Positive number, negative number or 0 ([0, 0, 0, 0, 0, 0])	When *bExecute* shifts to True.
PathChoice	Specify the circular or helical interpolation direction.	DMC_CIRC_PATHCHOICE	0: Clockwise 1: CounterClockwise (0)	When *bExecute* shifts to True.
dwSpiralTurns	Specify the number of spiral turns.	DWORD	0~65535 (0)	When *bExecute* shifts to True.
lrVelocity	Specify the target velocity for the axis group. (Unit: user unit/s)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrAcceleration	Specify the acceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrDeceleration	Specify the deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrJerk	Specify the jerk. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bExecute* shifts to True.
BufferMode	Specify a buffer mode for the instruction.*2	DMC_BUFFER_MODE	0: Aborting 1: Buffered 2: BlendingLow 3: BlendingPrevious 4: BlendingNext 5: BlendingHigh (0)	When *bExecute* shifts to True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
TransitionMode	Specify a transition mode for the instruction.[*3]	DMC_GROUP_TRANSITION_MODE	0: None 10: Overlap (0)	When *bExecute* shifts to True.

***Note:**

1. Setting parameters *CircPlane*, *CircMode* and *AuxPoint*.

CircMode setting	Definition CircMode – AuxPoint	*CircPlane*		
		XY_Plane	YZ_Plane	ZX_Plane
		Actual input for *AuxPoint* [_, _, _]		
0	Three points – Relative coordinate values for the border point (X_A, Y_A, Z_A)	Start point, end point and border point [X_A, Y_A, Z_A]		
1	A center point – Relative coordinate values for the center point (C_X, C_Y)	[C_X, C_Y, N/A]	[N/A, C_X, C_Y]	[C_Y, N/A, C_X]
2	Radius – Radius (R)	[R, N/A, N/A]		

2. Refer to AX-3 Series Operation Manual for details on BufferMode.
3. Refer to AX-3 Series Operation Manual for details on TransitionMode.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bDone	True when the relative positioning is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bActive	True when the instruction is controlling axes.	BOOL	True/False (False)
bCommand Aborted	True when the instruction execution is aborted.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR[*]	DMC_ERROR (DMC_NoError)

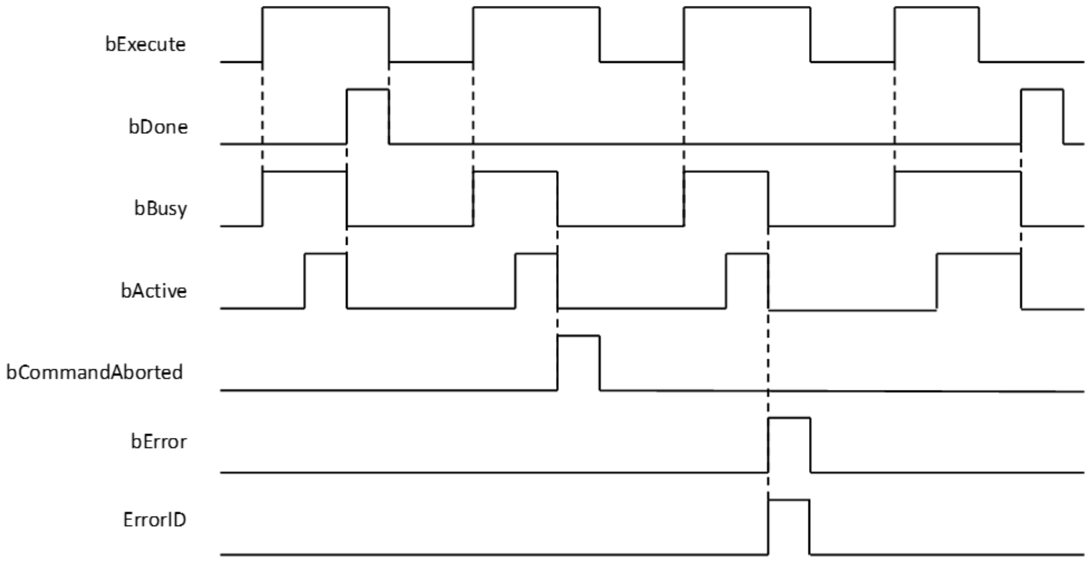
***Note:** DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the relative positioning is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one cycle when *bExecute* is False but *bDone* changes to True.

Name	Timing for shifting to True	Timing for shifting to False
bBusy	● When *bExecute* changes to TRUE.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bActive	● When axes start being controlled by the instruction.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● *bActive* will change to False after remaining True for at least one cycle when *bExecute* is False but *bActive* changes to True.
bCommand Aborted	● When the instruction is aborted by another instruction *BufferMode* of which is set to *Aborting*. ● When the instruction is aborted by MC_Stop. ● When the instruction is aborted by DMC_GroupStop.	● When *bExecute* shifts to False. ● *bCommandAborted* will change to False after remaining True for at least one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError/ErrorID	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded) .	● When *bExecute* shifts from True to False. (Error code is cleared)

■ **Timing Diagram**



● **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group.	DMC_AXIS_GROUP_REF*	DMC_AXIS_GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**

  - The instruction supports the relative helical interpolation of maximum three axes, where the three axes can simultaneously start, stop as well as reach the specified relative target position.

  - The instruction can be used to specify the circle drawing on the plane parallel to XY / YZ / ZX and set the height of the helix on Z / X / Y axis.

  - If the start point and end point for circular interpolation are set to the same point, use the center point mode (*CircMode* = Center) for the interpolation.

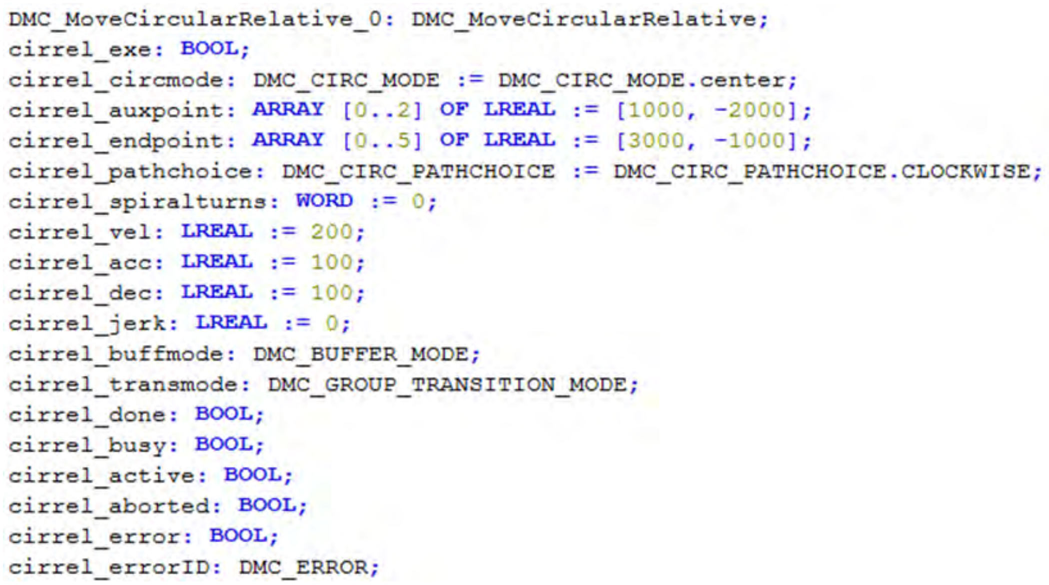
  - At least two axes are needed for circular interpolation. An error will occur if there is a travel distance for an axis which is not set.

  - When the start point and end point for circular interpolation are set to the same point, the instruction will perform the rotation for one complete circle.

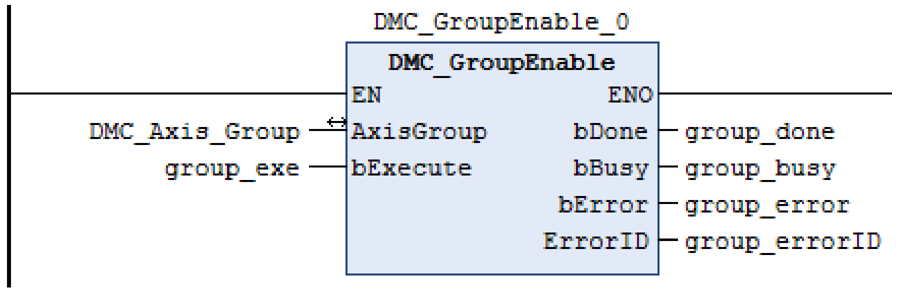
- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis group enters GroupErrorstop state, *bError* changes to True and axes stops running. To confirm current error state, see the error code in *ErrorID*.
  - For error codes and corresponding trouble shootings, refer to **Appendices** for error code descriptions.

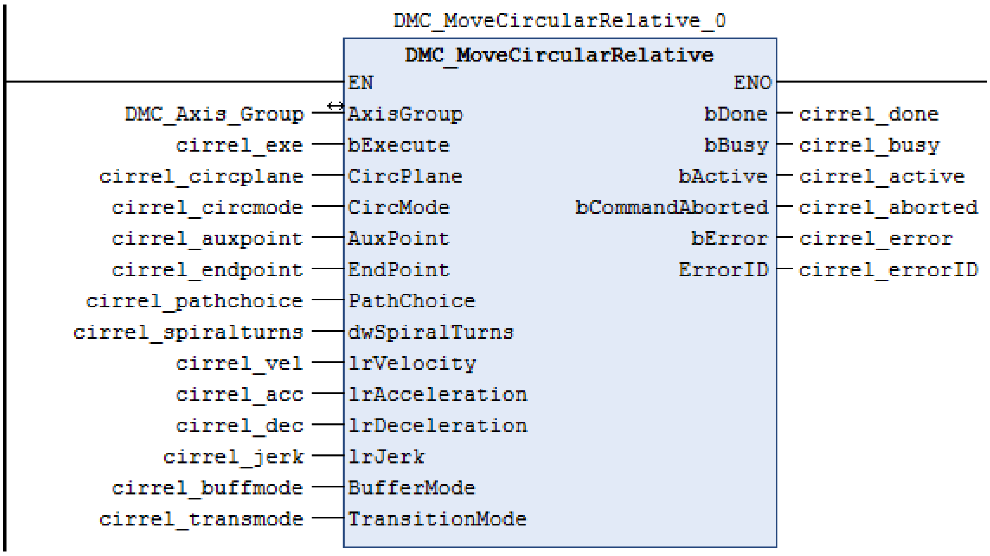
- **Programming Example**

  - In this example, the instruction performs the circular interpolation from current positon (1000, 3000) until the target position (4000, 2000) in the clockwise direction.

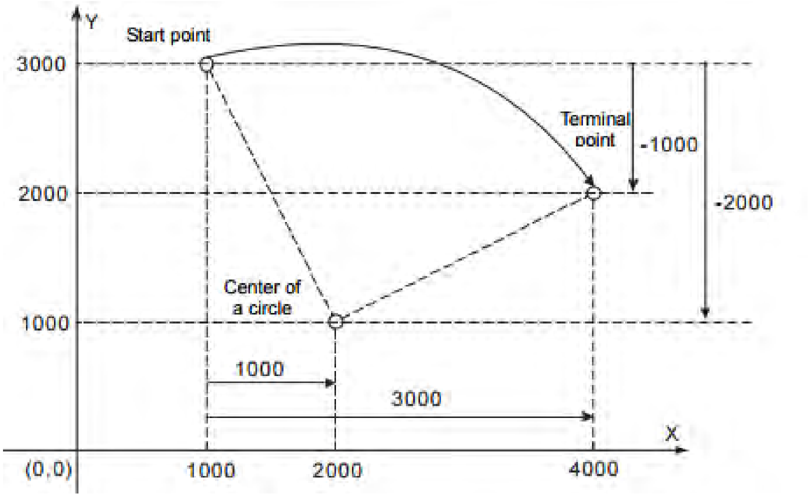


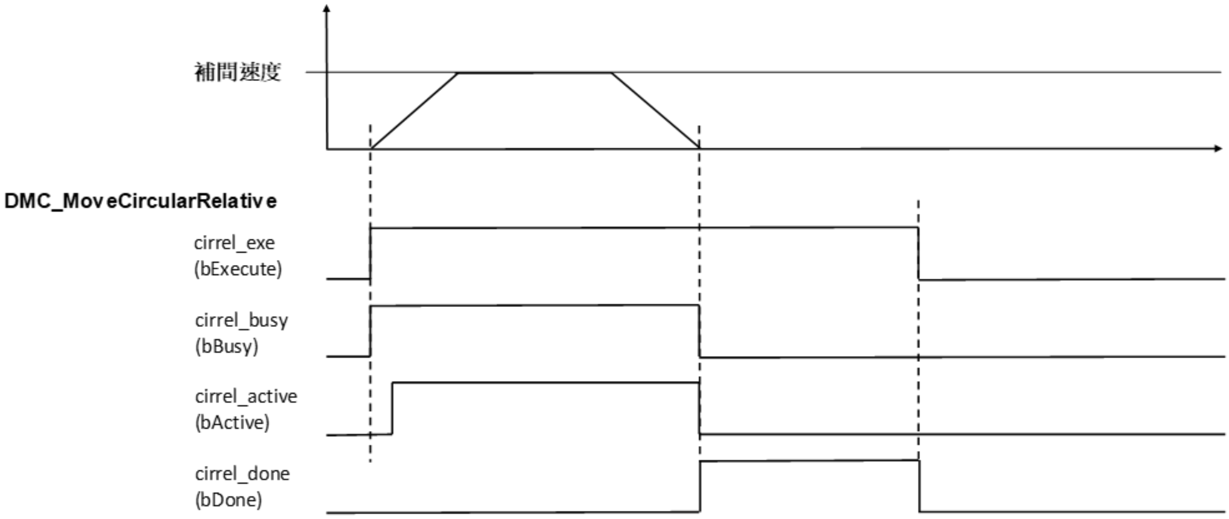
```
DMC_MoveCircularRelative_0: DMC_MoveCircularRelative;
cirrel_exe: BOOL;
cirrel_circmode: DMC_CIRC_MODE := DMC_CIRC_MODE.center;
cirrel_auxpoint: ARRAY [0..2] OF LREAL := [1000, -2000];
cirrel_endpoint: ARRAY [0..5] OF LREAL := [3000, -1000];
cirrel_pathchoice: DMC_CIRC_PATHCHOICE := DMC_CIRC_PATHCHOICE.CLOCKWISE;
cirrel_spiralturns: WORD := 0;
cirrel_vel: LREAL := 200;
cirrel_acc: LREAL := 100;
cirrel_dec: LREAL := 100;
cirrel_jerk: LREAL := 0;
cirrel_buffmode: DMC_BUFFER_MODE;
cirrel_transmode: DMC_GROUP_TRANSITION_MODE;
cirrel_done: BOOL;
cirrel_busy: BOOL;
cirrel_active: BOOL;
cirrel_aborted: BOOL;
cirrel_error: BOOL;
cirrel_errorID: DMC_ERROR;
```





```
                          DMC_MoveCircularRelative_0
                        DMC_MoveCircularRelative
                       EN                                ENO
     DMC_Axis_Group ─↔ AxisGroup                   bDone ─ cirrel_done
         cirrel_exe ── bExecute                    bBusy ─ cirrel_busy
   cirrel_circplane ── CircPlane                 bActive ─ cirrel_active
    cirrel_circmode ── CircMode         bCommandAborted ─ cirrel_aborted
    cirrel_auxpoint ── AuxPoint                   bError ─ cirrel_error
    cirrel_endpoint ── EndPoint                  ErrorID ─ cirrel_errorID
  cirrel_pathchoice ── PathChoice
 cirrel_spiralturns ── dwSpiralTurns
         cirrel_vel ── lrVelocity
         cirrel_acc ── lrAcceleration
         cirrel_dec ── lrDeceleration
        cirrel_jerk ── lrJerk
     cirrel_buffmode ── BufferMode
    cirrel_transmode ── TransitionMode
```





- When cirrel_exe (bExecute) changes to True, DMC_MoveCircularRelative performs the relative positioning toward the terminal point (4000, 2000) from the start point (1000, 3000) in the clockwise direction.
- When cirrel_done (*bDone*) is True and cirrel_busy (*bBusy*) changes to False, which means the relative target

positioning (4000, 2000) is completed. When cirrel_exe (*bExecute*) is switched to False, cirrel_done (*bDone*) will change to False automatically.

- If cirrel_exe (*bExecute*) is set to True again, the instruction will perform the circular interpolation regarding current positon (4000, 2000) as the reference point.

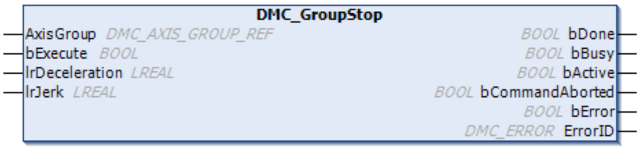
- **Supported Products**

  - AX-308E

2.3.1.7 DMC_GroupStop

DMC_GroupStop decelerates the group axes to a stop.

FB/FC	Instruction	Graphic expression	ST expression

FB	DMC_GroupStop		DMC_GroupStop_instance (
			AxisGroup : =,
			bExecute : =,
			lrDeceleration : =,
			lrJerk : =,
			bDone =>,
			bBusy =>,
			bActive =>,
			bCommandAborted =>,
			bError =>,
			ErrorID =>) ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
lrDeceleration	Specify the deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrJerk	Specify the jerk. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bExecute* shifts to True.

- **Outputs**

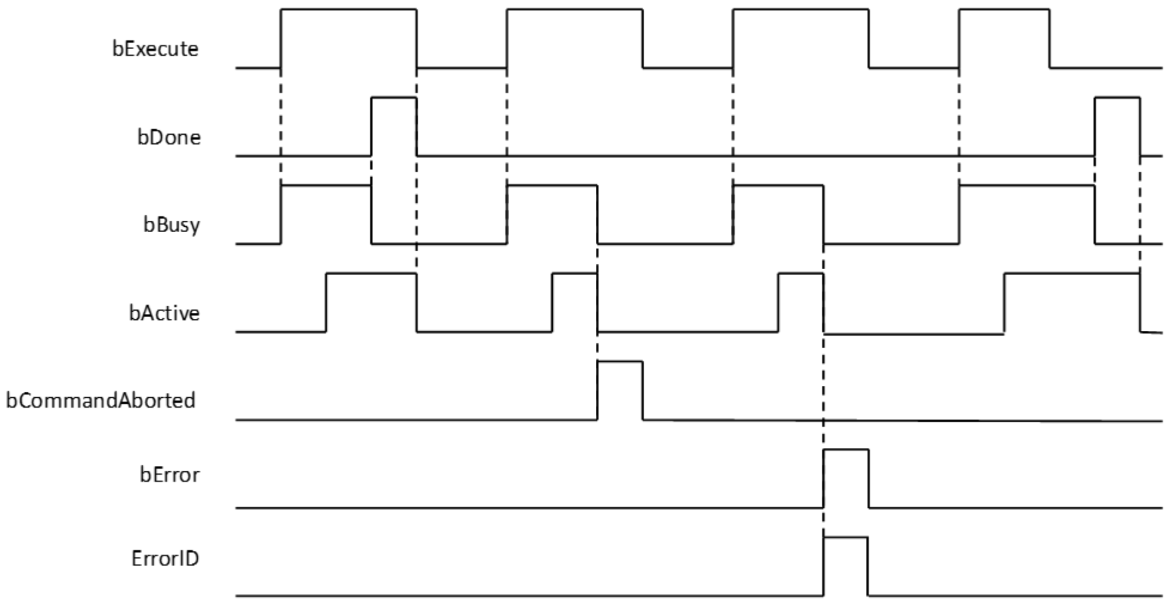
Name	Function	Data type	Output range (Default value)
bDone	True when all axes stop with the velocity 0.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bActive	True when the instruction is controlling axes.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

*Note: DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the axis group decelerates to a stop.	● When *bExecute* shifts to False. ● *bExecute* will change to False after remaining True for one cycle when *bExecute* is False but *bDone* changes to True.
bBusy	● When *bExecute* shifts to True.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bActive	● When axes start being controlled by the instruction.	● When *bExecute* shifts to False. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● *bActive* will change to False after remaining True for at least one cycle if *bExecute* changes to False but *bActive* changes to True.
bCommandAborted	● When the instruction is aborted by another instruction.	● When *bExecute* shifts to False. ● *bCommandAborted* will change to False after remaining True for one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



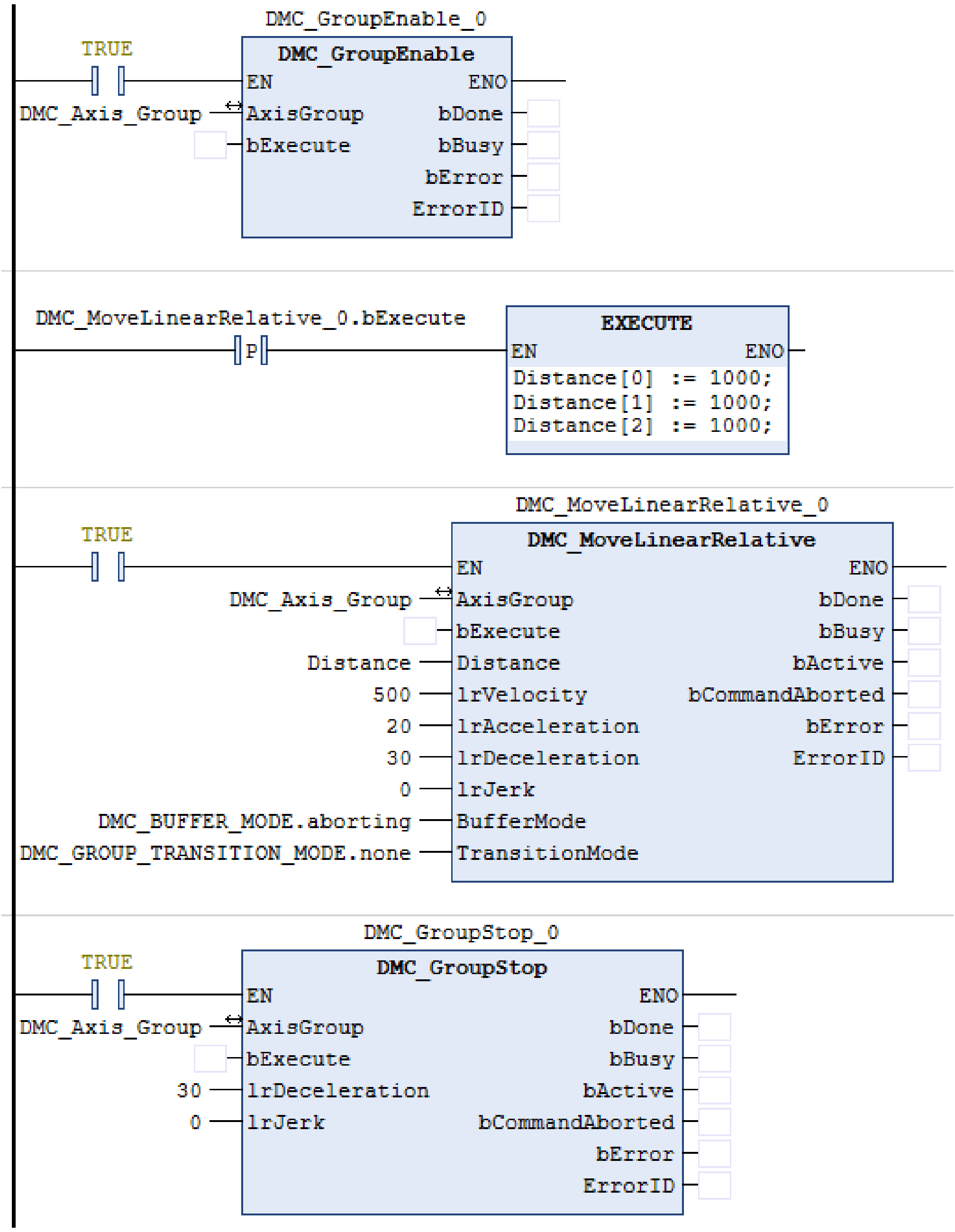
- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group.	DMC_AXIS_GROUP_REF*	DMC_AXIS_GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

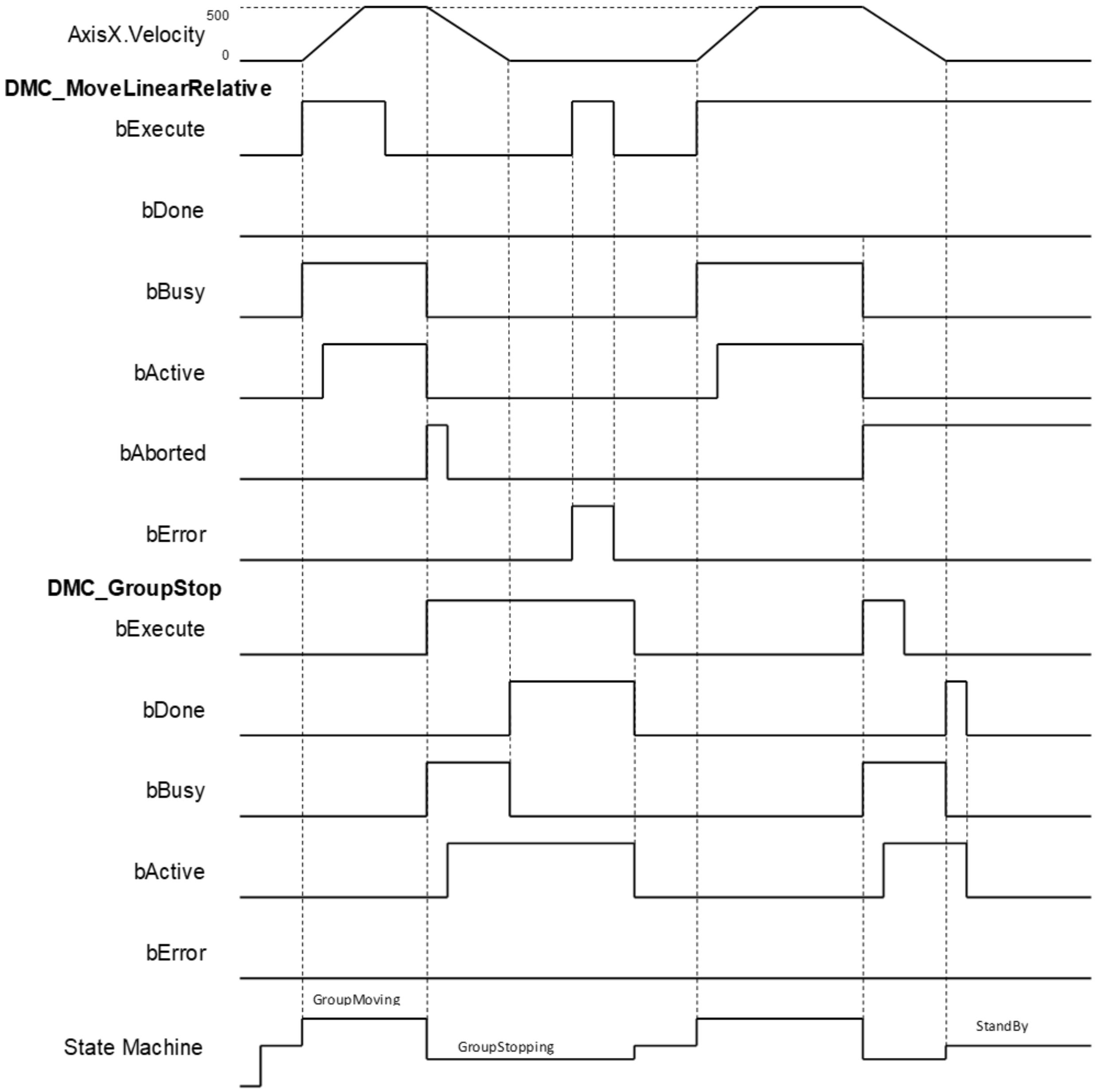
DMC_AXIS_GROUP_REF (FB) : All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**
  - The instruction decelerates the group axes in motion to a stop.
  - The axis group state is switched to GroupStopping via the instruction.
  - The axis group state GroupStopping will continue until *bExecute* changes to False. *bDone* changes to True when the velocity 0 is reached.
- **Troubleshooting**
  - *bError* changes to True when an error occurs. To confirm current error state, see the error code in *ErrorID*.
- **Programming Example**
  - This example shows the motion behavior which is performed by DMC_GroupStop during the execution of DMC_MoveLinearRelative.
  - When the execution of DMC_GroupStop is completed, the axis group enters GroupStandby state.



```
Distance[0] := 1000;
Distance[1] := 1000;
Distance[2] := 1000;
```

- **Timing Diagram**



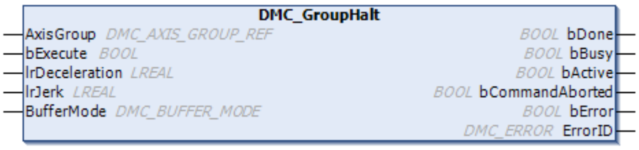
- When *bExecute* of DMC_GroupStop changes to True, *bCommandAboted* of MoveLinearRelative changes to True and axes start to decelerate to a stop. Meanwhile the axis group stays in GroupStopping state.
- When the velocities of axes reach 0, *bDone* of DMC_GroupStop changes to True and the axis group holds GroupStopping state.
- When *bExecute* of DMC_GroupStop changes to False, the state of axes changes from GroupStopping into StandBy.

- **Supported Products**
  - AX-308E

2.3.1.8 DMC_GroupHalt

DMC_GroupHalt decelerates the axis group in motion to a pause.

FB/FC	Instruction	Graphic expression	ST expression

FB	DMC_GroupHalt	 DMC_GroupHalt
		AxisGroup DMC_AXIS_GROUP_REF
		bExecute BOOL
		lrDeceleration LREAL
		lrJerk LREAL
		BufferMode DMC_BUFFER_MODE
		BOOL bDone
		BOOL bBusy
		BOOL bActive
		BOOL bCommandAborted
		BOOL bError
		DMC_ERROR ErrorID
			DMC_GroupHalt_instance (
			AxisGroup : =,
			bExecute : =,
			lrDeceleration : =,
			lrJerk : =,
			BufferMode : =,
			bDone =>,
			bBusy =>,
			bActive =>,
			bCommandAborted =>,
			bError =>,
			ErrorID =>) ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
lrDeceleration	Specify the deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.
lrJerk	Specify the jerk. (Unit: user unit/s^3)	LREAL	Positive number (0)	When *bExecute* shifts to True.
BufferMode	Specify a buffer mode for the instruction.*	DMC_BUFFER_MODE	0: Aborting 1: Buffered (0)	When *bExecute* shifts to True.

***Note**: Refer to AX-3 Series Operation Manual for details on BufferMode.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bDone	True when all axes stop with the velocity 0.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bActive	True when the instruction is controlling axes.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)

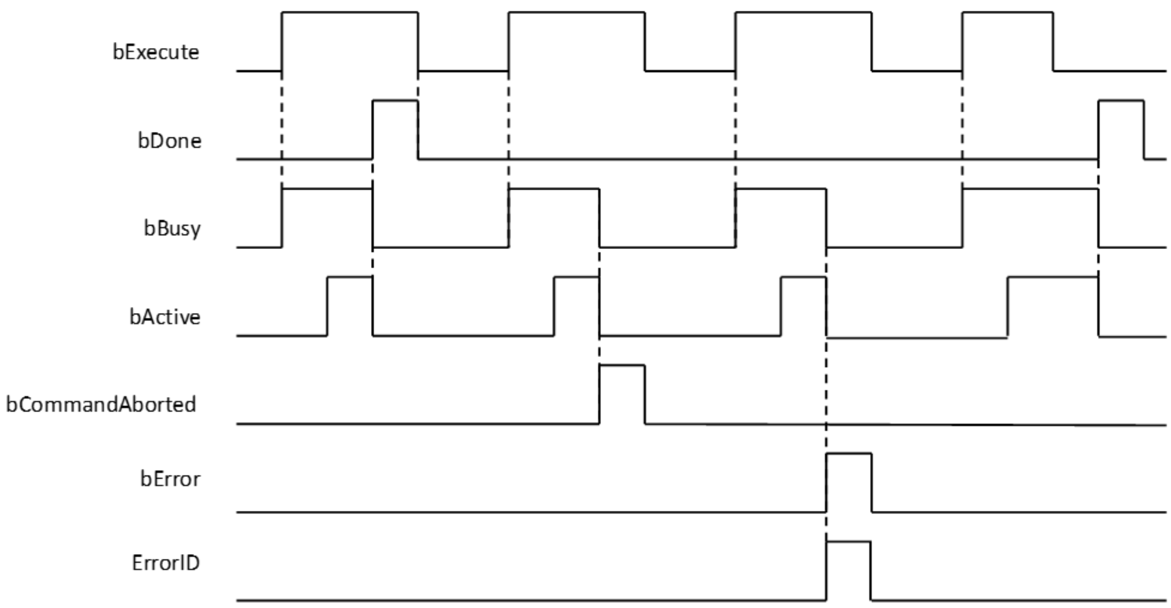
Name	Function	Data type	Output range (Default value)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

*Note: DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Function	Data type
bDone	● When the axis group decelerates to a stop.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one cycle when *bExecute* changes to False but *bDone* changes to True.
bBusy	● When *bExecute* shifts to True.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bActive	● When axes start being controlled by the instruction.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● *bActive* will change to False after remaining True for at least one cycle when *bExecute* changes to False but *bActive* changes to True.
bCommandAborted	● When the instruction is aborted by another instruction.	● When *bExecute* shifts to False. ● *bCommandAborted* will change to False after remaining True for one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared.)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group.	DMC_AXIS_GROUP_REF	DMC_AXIS_GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

AxisGroup_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**

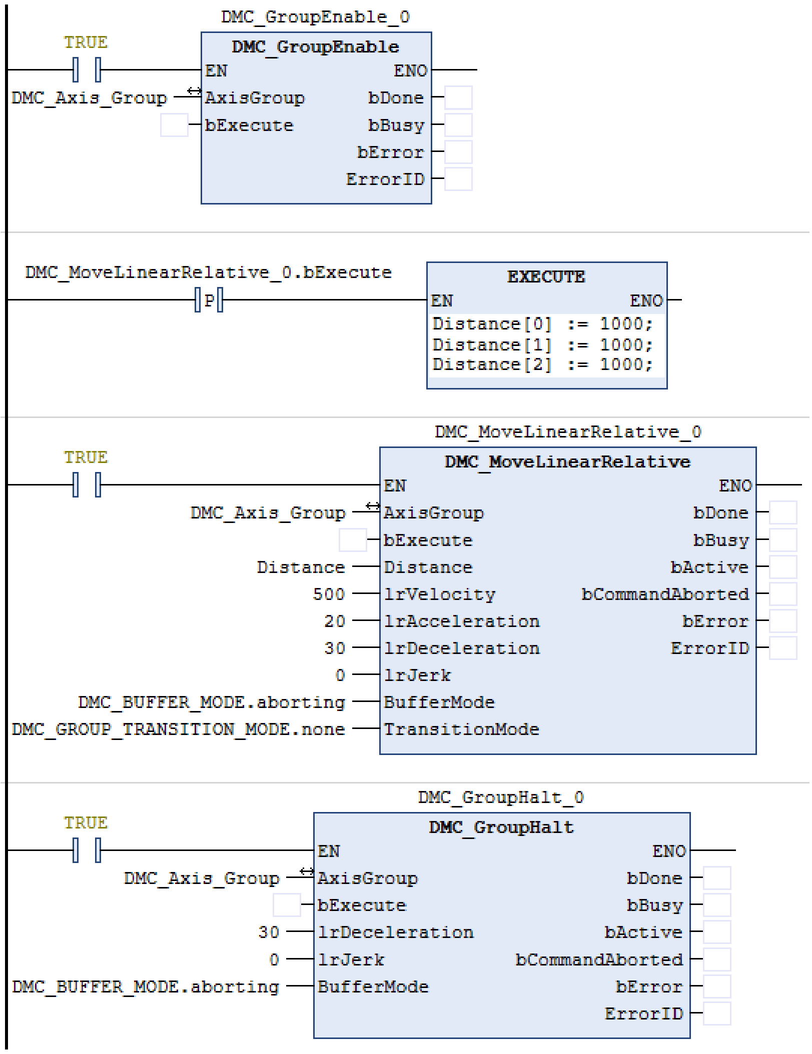
  - The instruction decelerates the group axes in motion to a pause.
  - The axis group enters the state of GroupMoving via the instruction.
  - When the velocity 0 is reached, *bDone* changes to True and the axis group changes to StandBy state.
  - *BufferMode* of DMC_GroupHalt only supports 0: Aborting and 1: Buffered. An error will occur with DMC_ERROR.DMC_GM_INVALID_BUFFER_MODE if other BufferMode is used.

- **Troubleshooting**

  - When an error occurs in the instruction execution or the axis group enters Errorstop state, *bError* changes to True and axes stops running. To confirm current error state, see the error code in *ErrorID*.

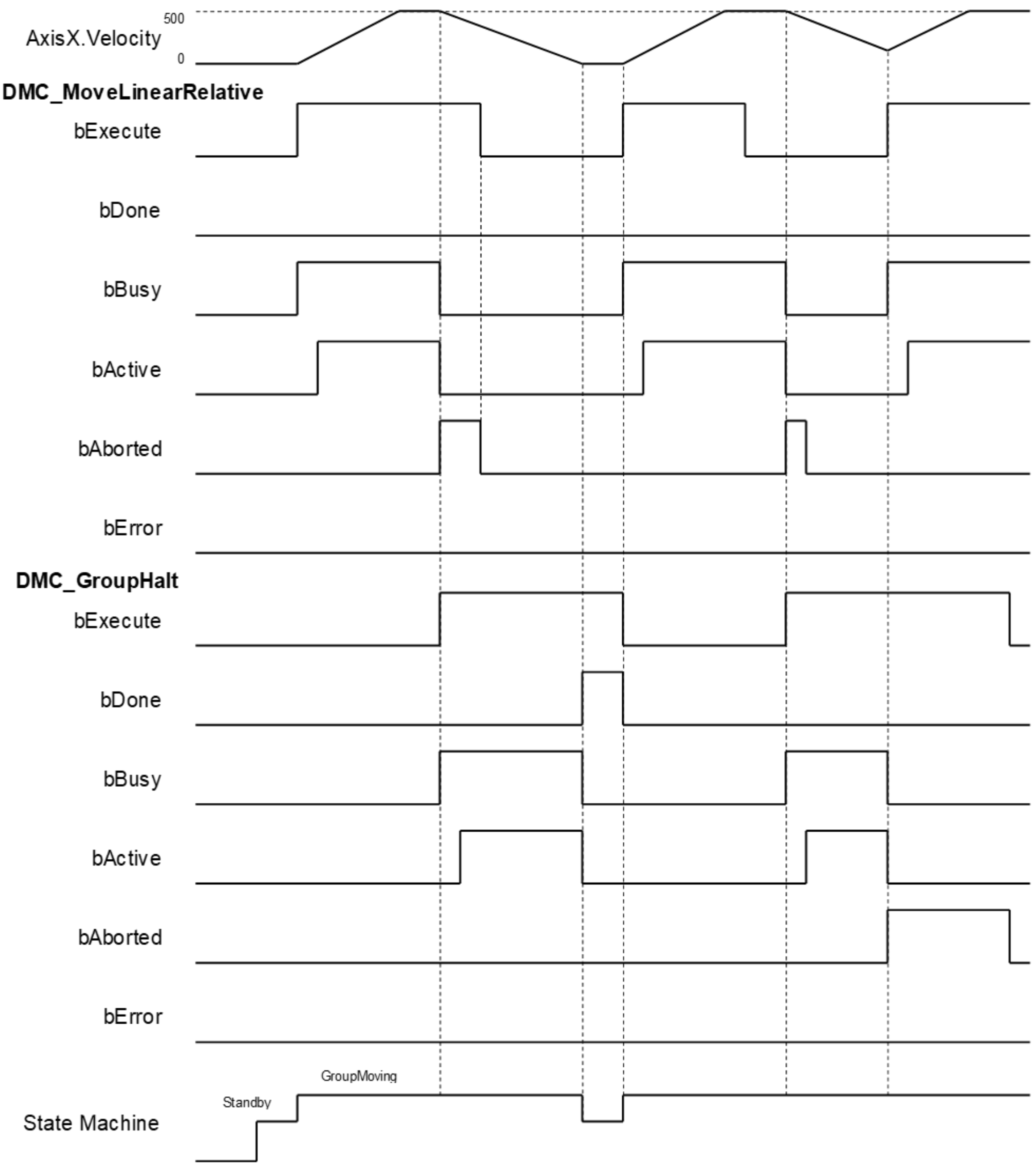
- **Programming Example**

  - This example shows the motion behavior which is performed by DMC_GroupHalt after DMC_MoveLinearRelative is executed.
  - The axes will enter Standby state after deceleration is completed if no other motion instruction is executed during the period when DMC_MoveLinearRelative is paused via DMC_GroupHalt.
  - When DMC_MoveLinearRelative is executed again during deceleration，DMC_GroupHalt will be aborted immediately and the axis group will accelerate again without staying in Standby state any more. The re-execution action described above is allowed for DMC_GroupHalt.



```
Distance[0] := 1000;
Distance[1] := 1000;
Distance[2] := 1000;
```

- **Timing Diagram**



- When *bExecute* of DMC_GroupHalt changes to True, *bCommandAboted* of DMC_MoveLinearRelative changes to True and the axes start to decelerate to a stop. And the axis group stays in GroupMoving state.

- When the velocity 0 is reached, *bDone* of DMC_GroupHalt changes to True and the axis group changes to Standby state.

- When the velocity has not been reduced to 0 yet and *bExecute of* DMC_GroupHalt changes to True during the instruction execution, DMC_GroupHalt will be aborted by changing *bExecute* of DMC_MoveLinearRelative to True again and then its *bCommandAboted* will change to True.

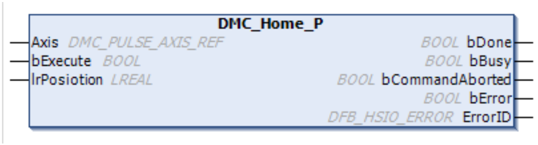
- **Supported Products**

- AX-308E

2.3.1.9 DMC_Home_P

DMC_Home_P, an application function block of pulse output, drives the pulse axis to perform the homing in the set mode.

FB/FC	Instruction	Graphic expression	ST expression

FB	DMC_Home_P	 DMC_Home_P
		Axis DMC_PULSE_AXIS_REF
		bExecute BOOL
		IrPosiotion LREAL
		BOOL bDone
		BOOL bBusy
		BOOL bCommandAborted
		BOOL bError
		DFB_HSIO_ERROR ErrorID
			DMC_Home_P_instance (
			Axis : =,
			bExecute : =,
			lrPosition : =,
			bDone =>,
			bBusy =>,
			bCommandAborted =>,
			bError =>,
			ErrorID =>)　;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
lrPosition	Specify the position after the homing is completed.	LREAL	Positive number, negative number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

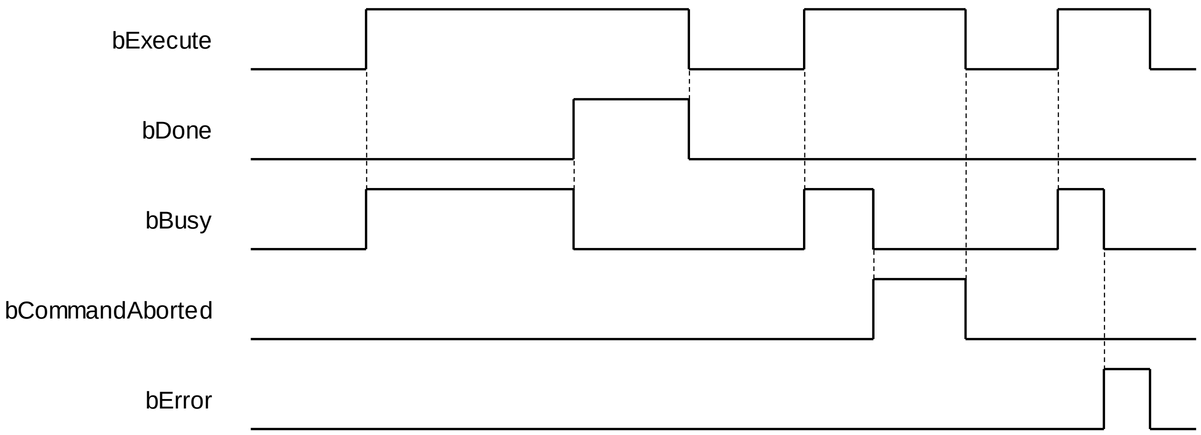
Name	Function	Data type	Output range (Default value)
bDone	True when the homing is completed.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted by another instruction.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DFB_HSIO_ERROR*	DFB_HSIO_ERROR (DFB_HSIO_NO_ERR)

***Note:** DFB_HSIO_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Function	Data type
bDone	● When the homing is completed.	● When *bExecute* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bExecute* changes to TRUE.	● When *bExecute* shifts to False. ● When *bError* shifts to True.
bCommandAborted	● When the instruction is aborted by MC_Stop.	● When *bExecute* shifts to False. ● *bCommandAborted* will change to False after remaining True for one cycle when *bExecute* is False but *bCommandAborted* changes to True.
bError	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared.)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the source of pulse output axis	DMC_PULSE_AXIS_REF (FB) *	DMC_PULSE_AXIS_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

DMC_PULSE_AXIS_REF (FB): The function block serves as the drive interface for the pulse output axis, which contains the axis parameter call and the drive program.

- **Function**

  - The pulse output axis specified by the instruction must be selected in **Hardware IO Configuration** so that the axis can ouput pulses and perform the homing action according to the pulse axis settings i.e. homing mode, acceleration rate and velocity.

  - The instruction can be used only when the pulse output axis is in Standstill state. An error will occur if the instruction is executed in other axis state.

  - DMC_Home_P supports homing modes defined in CiA 402 protocol. For details on homing modes, refer to appendices.

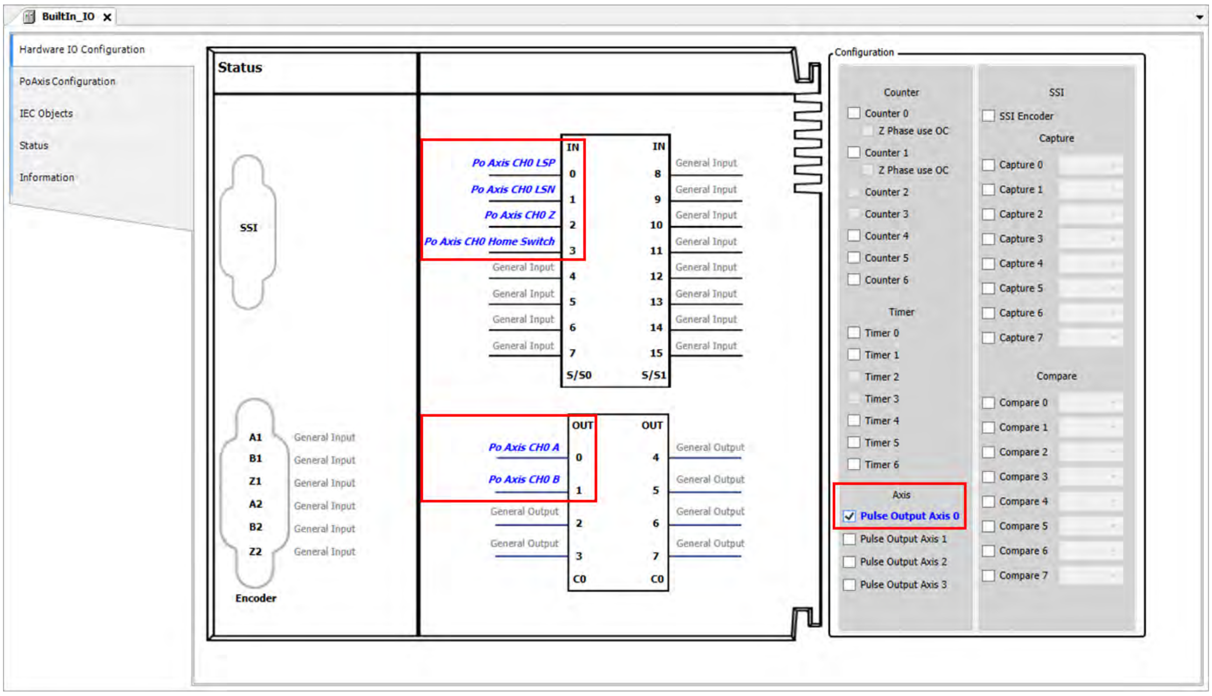
- **Troubleshooting**

  - When an error occurs in the instruction execution, *bError* of the instruction changes to True. To confirm current error state, see the error code in *ErrorID*.

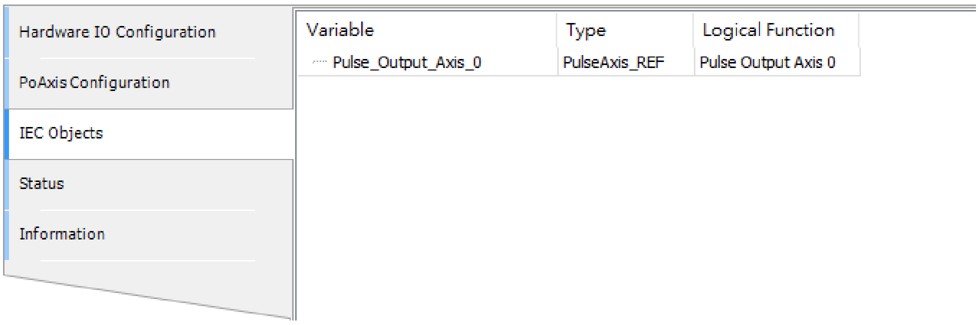
- **Programming Example**

  - In this example, the pulse output axis performs the homing motion via DMC_Home_P after the axis is configured in the IO configuration interface.

  - Select the first pulse output axis (Pulse Output Axis 0) in Hardware IO Configuration of BuiltIn_IO as below. Then you can see corresponding output points (e.g. OUT0, OUT1) and signal trigger points for pulse output (e.g. IN0, IN1, IN2 and IN3) from the software. The homing motion cannot be performed until the signal trigger points for the homing mode have been configured to corresponding input signal sources.

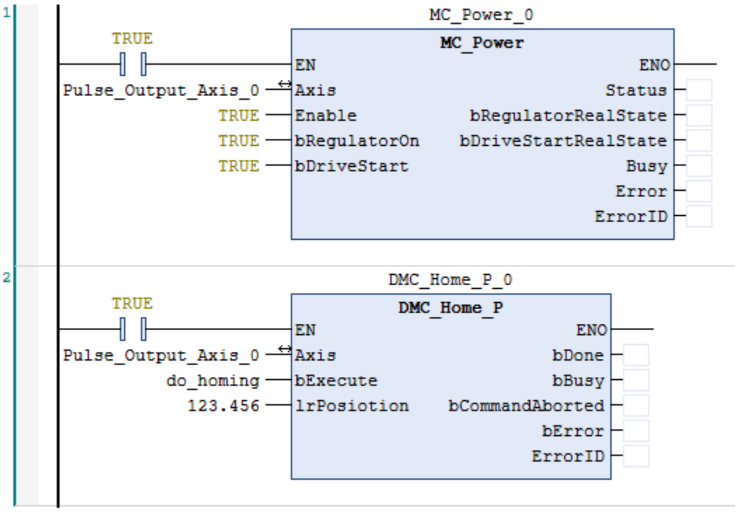


- After the configuration of the pulse output axis, the variable Pulse_Output_Axis_0 configured in IEC Objects can be taken out as a data type to any function block, as shown below.

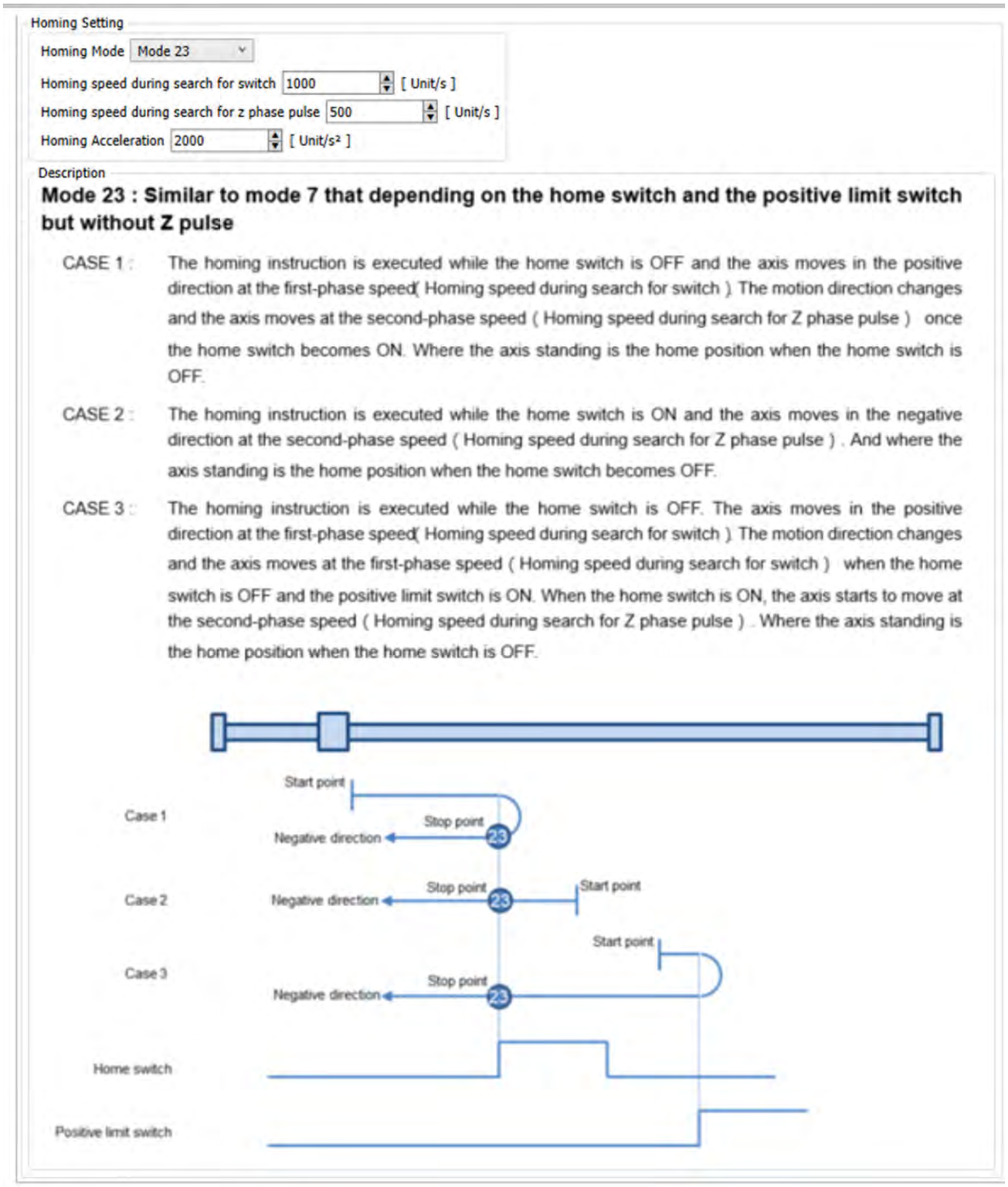


Variable	Type	Logical Function
Pulse_Output_Axis_0	PulseAxis_REF	Pulse Output Axis 0

- Pulse_Output_Axis_0 is connected to the input *Axis* of MC_ Power and DMC_ Home_P as shown in the figure below. When the axis is in Standstill state, the instruction is started to perform the homing motion according to the set homing mode. At the moment, the state machine will switch the state from Standstill to Homing.



- After DMC_Home_P is started, the pulse axis Pulse_Output_Axis_0 will perform the homing motion according to the set Home Mode in PoAxis Configuration below. After the function block is executed, the homing motion will be conducted according to different external signals and cases.
  - Homing Mode: Mode 23;
  - Homing speed during search for switch: 1000 (Unit: user unit /s);
  - Homing speed during search for z phase pulse: 500 (Unit: user unit /s);
  - Homing Acceleration: 2000 (Unit: user unit /s ^{2}).



Homing Setting

Homing Mode: Mode 23

Homing speed during search for switch: 1000 [Unit/s]

Homing speed during search for z phase pulse: 500 [Unit/s]

Homing Acceleration: 2000 [Unit/s²]

Description

Mode 23 : Similar to mode 7 that depending on the home switch and the positive limit switch but without Z pulse

CASE 1 : The homing instruction is executed while the home switch is OFF and the axis moves in the positive direction at the first-phase speed(Homing speed during search for switch). The motion direction changes and the axis moves at the second-phase speed (Homing speed during search for Z phase pulse) once the home switch becomes ON. Where the axis standing is the home position when the home switch is OFF.

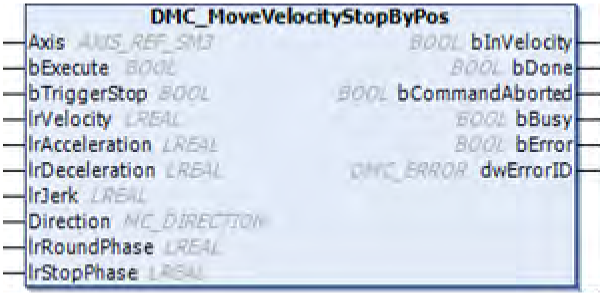
CASE 2 : The homing instruction is executed while the home switch is ON and the axis moves in the negative direction at the second-phase speed (Homing speed during search for Z phase pulse) . And where the axis standing is the home position when the home switch becomes OFF.

CASE 3 : The homing instruction is executed while the home switch is OFF. The axis moves in the positive direction at the first-phase speed(Homing speed during search for switch). The motion direction changes and the axis moves at the first-phase speed (Homing speed during search for switch) when the home switch is OFF and the positive limit switch is ON. When the home switch is ON, the axis starts to move at the second-phase speed (Homing speed during search for Z phase pulse) . Where the axis standing is the home position when the home switch is OFF.

- **Supported Products**
  - AX-308E

2.3.1.10 DMC_MoveVelocityStopByPos

DMC_MoveVelocityStopByPos controls an axis to stop at a specified position after a period of motion.

FB/FC	Instruction	Graphic expression
FB	DMC_MoveVelocityStopByPos	 DMC_MoveVelocityStopByPos Axis AXIS_REF_SM3 bExecute BOOL bTriggerStop BOOL lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL lrJerk LREAL Direction MC_DIRECTION lrRoundPhase LREAL lrStopPhase LREAL BOOL bInVelocity BOOL bDone BOOL bCommandAborted BOOL bBusy BOOL bError DMC_ERROR dwErrorID

ST expression

DMC_MoveVelocityStopByPos_instance(
Axis :=,
bExecute :=,
bTriggerStop :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
lrJerk :=,
Direction :=,
lrRoundPhase:=,
lrStopPhase:=,
bInVelocity =>,
bDone =>,
bCommandAborted =>,
bBusy =>,
bError =>,
dwErrorID =>)

- **Inputs**

Name	**Function**	**Data type**	**Setting value** **(Default value)**	**Timing for updating**
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
bTriggerStop	The stop command is executed when *bExecute* is True.	BOOL	True/False (False)	When *bExecute* is True and the output *bBusy* is True.
lrVelocity	Specify the target velocity. (Unit: User unit/s)	LREAL	Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
lrAcceleration	Specify the acceleration rate when the motion starts. (Unit: User unit/s^2)	LREAL	Positive number (0)	When *bExecute* is True and the output *bBusy* is False.

Name	Function	Data type	Setting value (Default value)	Timing for updating
lrDeceleration	Specify the deceleration rate when the motion ends. (Unit: User unit/s^2)	LREAL	Positive number (0)	When *bExecute* is True and the output *bBusy* is False.
lrJerk	Specify the jerk. (Unit: User unit/s^3)	LREAL	Positive number (0)	When *bExecute* is True and the output *bBusy* is False.
Direction	Specify the motion direction.	MC_DIRECTION*	-1: negative 1: positive (positive)	When *bExecute* is True and the output *bBusy* is False.
lrRoundPhase	Set the modulo.	LREAL	Positive number (0)	When *bExecute* is True and the output *bBusy* is False.
lrStopPhase	Specify a position or a phase in the modulo.	LREAL	Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.

*Note: MC_DIRECTION: Enumeration (Enum)

- **Outputs**

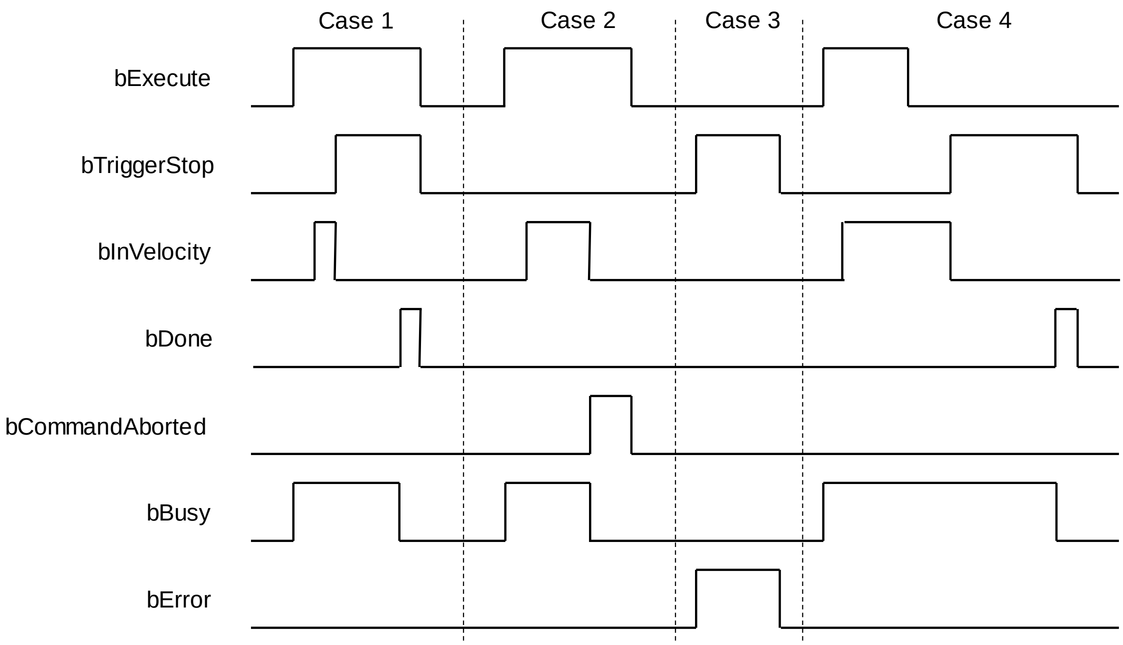
Name	Function	Data type	Output range (Default value)
bDone	True when *bInVelocity* is True and the specified axis position is completed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted by another instruction.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
dwErrorID	Contains error codes.	DMC_ERROR*	DMC_ERROR (DMC_NO_ERROR)

*Note: DMC_ERROR: Enumeration (Enum)

■ **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bDone	● When *bInVelocity* is True and the specified axis position is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one period when *bExecute* is False but *bDone* changes to True.
bCommandAborted	● When the instruction is aborted by another instruction.	● When *bExecute* shifts to False.
bBusy	● When *bExecute* shifts to True.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts from True to False. (Error code is cleared.)
dwErrorID		

- **Timing Diagram**

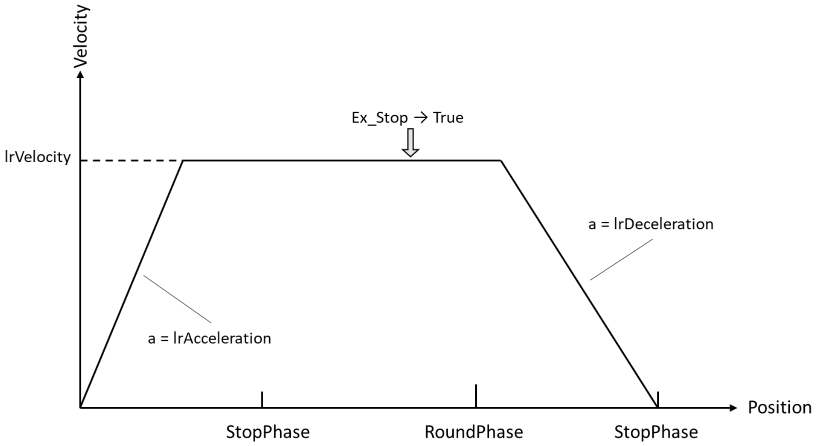


- **In-Outs**

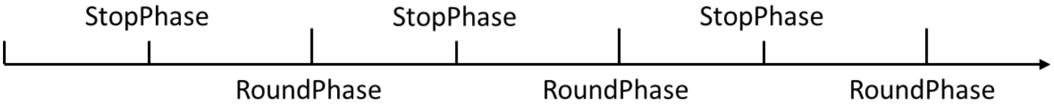
Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *bExecute* shifts to True.

*Note: AXIS_REF_SM3 (FB): The interface is built in every function block and works as the starting program of the function block.

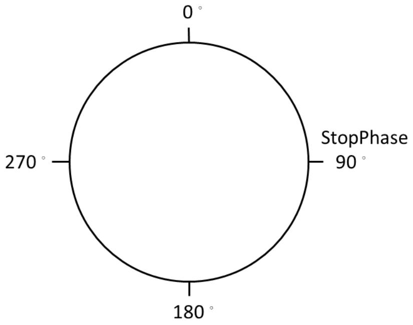
- **Function**
  - After *bExecute* of DMC_MoveVelocityStopByPos changes to True, the axis will move with the velocity specified by *lrVelocity* and acceleration specified by *lrAcceleration* until *bTriggerStop* changes to True. Then the axis positioning will start according to *lrDeceleration* and the target position converted from the values of *RoundPhase* and*StopPhase*.



  - RoundPhase & StopPhase
    - *RoundPhase* specifies a modulo. *StopPhase* is a position in the modulo. The value of *StopPhase* should be less than that of *RoundPhase.*
    - When the axis specified by the function block is a linear axis, *RoundPhase* is the length of the specified modulo. And *StopPhase* is a point in the specified modulo. When *bTriggerStop* changes to True, the axis will stop at the position specified by *StopPhase*, and the final stop position equals an integral multiple of *RoundPhase* value + *StopPhase* value.



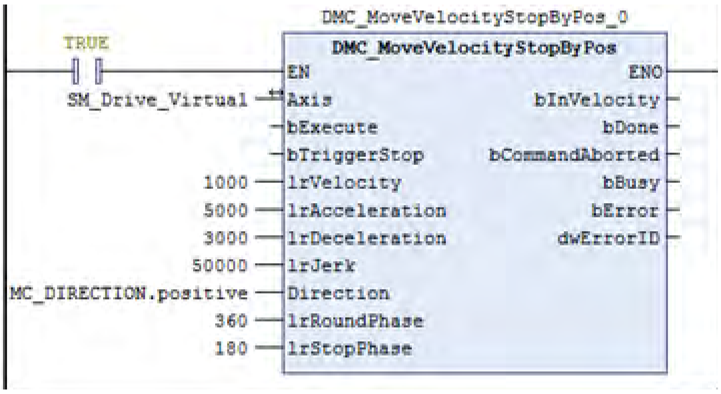
    - When the axis specified by the function block is a rotary axis, *RoundPhase* specifies the entire phase of the modulo and *StopPhase* is a phase in the specified modulo. When *bTriggerStop* changes to True, the axis will stop at the phase specified by StopPhase, and the final stop position is (*StopPhase* value/ *RoundPhase* value) × **Modulo value** of the rotary axis.



- Special Case
  - When the stop command is performed, the axis may not be able to complete the stop action with the deceleration rate specified by *lrdeceleration* if the position of the specified axis is too close to the target stop position. Therefore, the axis positioning will end in the next modulo. In that case, it is suggested to adjust the value of *lrDeceleration* or the position where the stop command is triggered so as to satisfy the path planned in the deceleration motion.

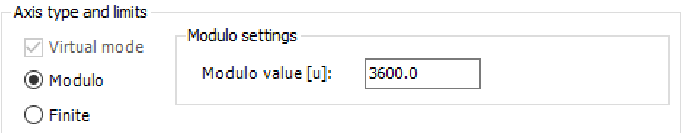
- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, *bError* changes to True and the axis stops running. To confirm current error state, see the error code in *ErrorID*.

- **Programming Examples**
  - Programming Example 1:
    - This example illustrates how to use DMC_MoveVelocityStopByPos for phase positioning after the rotary axis motion starts.



```
                                   DMC_MoveVelocityStopByPos_0
     TRUE               DMC_MoveVelocityStopByPos
                       EN                                   ENO
  SM_Drive_Virtual  ---Axis                          bInVelocity
                       bExecute                            bDone
                       bTriggerStop            bCommandAborted
              1000 --- lrVelocity                          bBusy
              5000 --- lrAcceleration                     bError
              3000 --- lrDeceleration                   dwErrorID
             50000 --- lrJerk
MC_DIRECTION.positive --- Direction
               360 --- lrRoundPhase
               180 --- lrStopPhase
```

  - Rotary axis setting



Axis type and limits

☑ Virtual mode

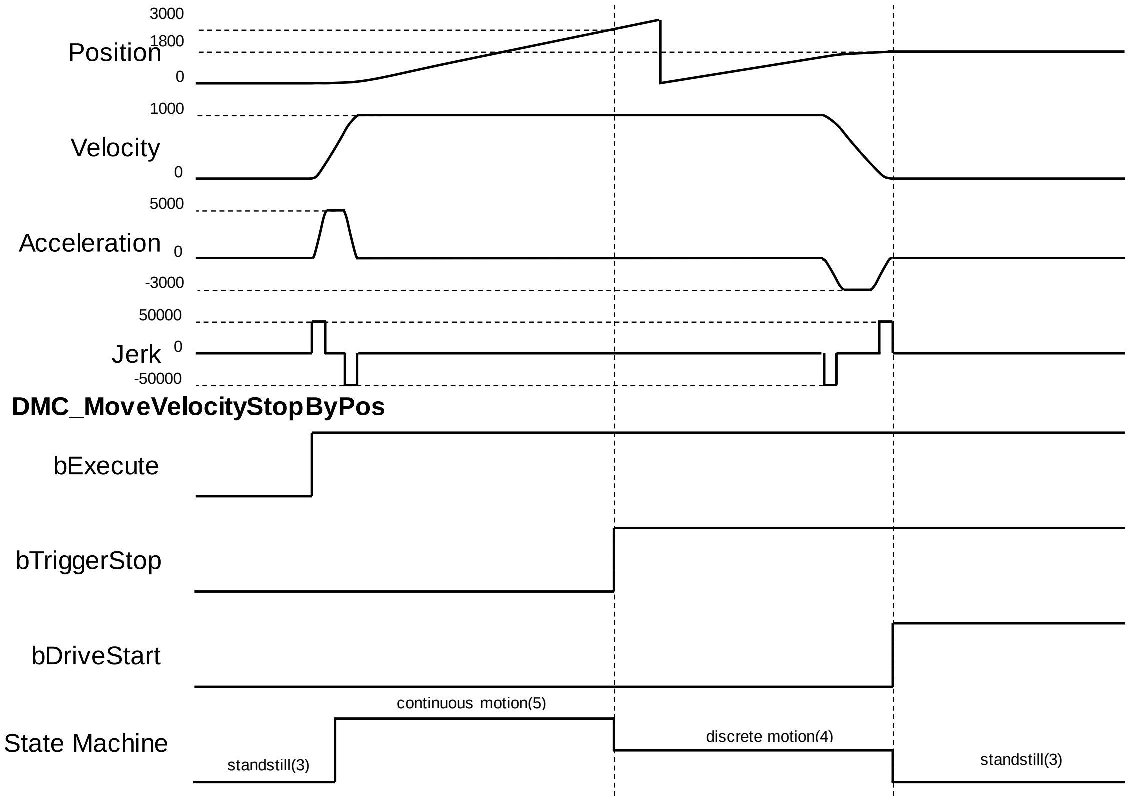
● Modulo

○ Finite

Modulo settings

Modulo value [u]: 3600.0

- Timing Diagram

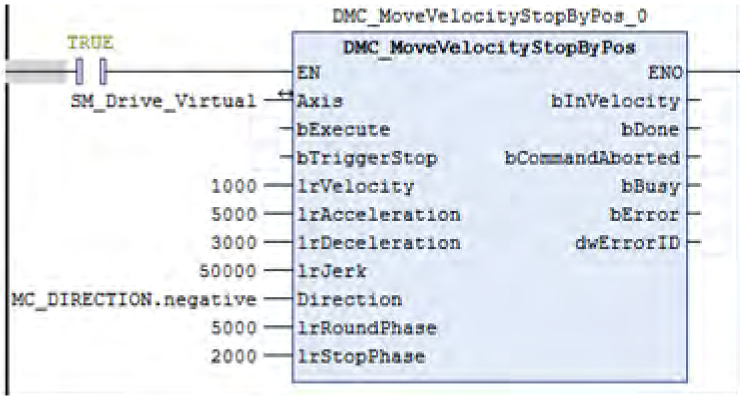


1. After *bExecute* changes to True, the axis starts to move at a constant speed in the set direction until *bTriggerStop* changes to True to start the positioning motion.

2. The *RoundPhase* and *StopPhase* of DMC_MoveVelocityStopByPos are set to 360 and 180, and the Modulo value of the rotary axis is 3600. Therefore, the rotary axis finally stops at 1800.

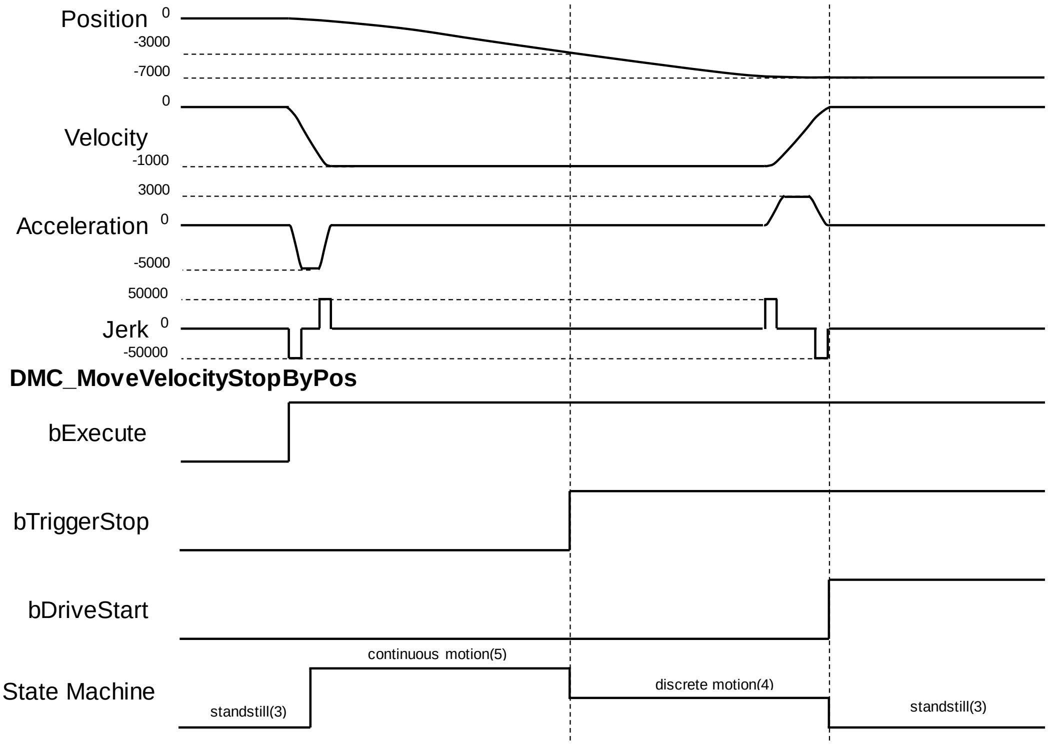
3. Since the axis position exceeds the position specified by *StopPhase* as *bTriggerStop* shifts to True, the axis will stop at the next *StopPhase* position.

- Programming Example 2:
  - This example illustrates how to use DMC_MoveVelocityStopByPos for position locating after the linear axis motion starts.



```
DMC_MoveVelocityStopByPos_0
DMC_MoveVelocityStopByPos
TRUE                   EN                              ENO
SM_Drive_Virtual       Axis                    bInVelocity
                       bExecute                      bDone
                       bTriggerStop        bCommandAborted
1000                   lrVelocity                    bBusy
5000                   lrAcceleration               bError
3000                   lrDeceleration            dwErrorID
50000                  lrJerk
MC_DIRECTION.negative  Direction
5000                   lrRoundPhase
2000                   lrStopPhase
```

- Timing Diagram



1. After *bExecute* changes to True, the axis starts to move at a constant speed in the set direction until *bTriggerStop* changes to True to start the positioning motion.

2. The *RoundPhase* and *StopPhase* of DMC_MoveVelocityStopByPos are set to 5000 and 2000 respectively. Therefore, the linear axis finally stops at the position of an integral multiple of 5000 plus 2000.

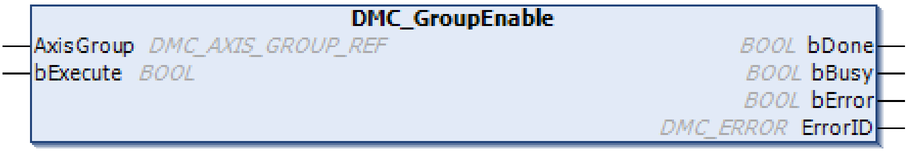
3. Since the axis position exceeds 2000 as *bTriggerStop* shifts to True, the axis will stop at the next 7000.

- **Supported Products**
  - AX-308E

2.3.2 Administrative Instructions

2.3.2.1 DMC_GroupEnable

DMC_GroupEnable switches the axis group state from GroupDisable to GroupStandby.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>DMC_GroupEnable</td><td>DMC_GroupEnable
AxisGroup DMC_AXIS_GROUP_REF
bExecute BOOL
BOOL bDone
BOOL bBusy
BOOL bError
DMC_ERROR ErrorID</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">DMC_GroupEnable_instance (
AxisGroup: = ,
bExecute: = ,
bDone=> ,
bBusy=> ,
bError=> ,
ErrorID=>) ;</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-

- **Outputs**

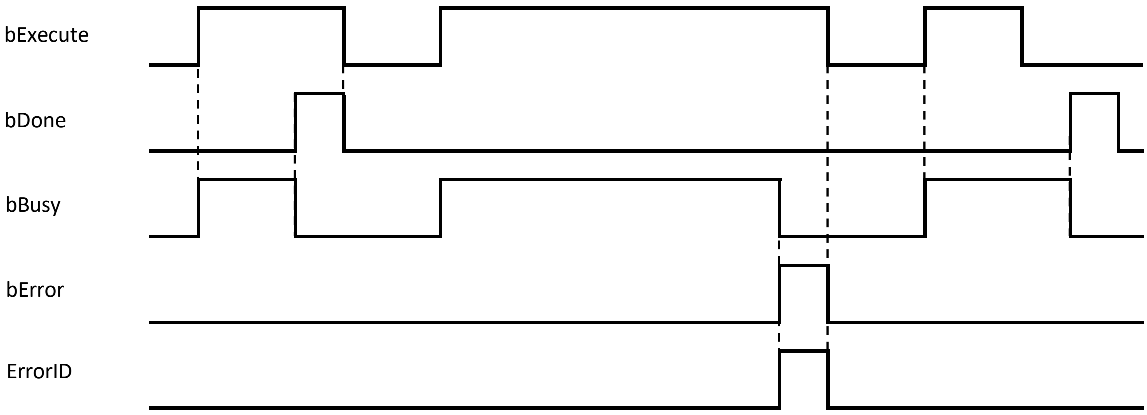
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

***Note:** DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the instruction is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one cycle when *bExecute* is False but *bDone* changes to True.
bBusy	● When *bExecute* changes to TRUE.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError (ErrorID)	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared)

- **Timing Diagram**



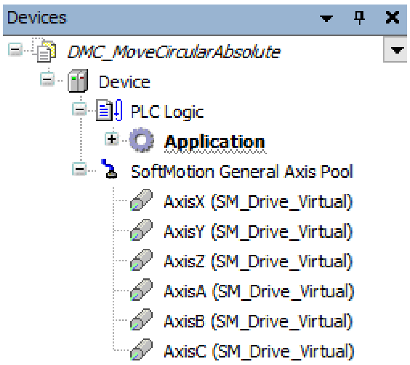
- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group	DMC_AXIS_ GROUP_REF*	DMC_AXIS_ GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

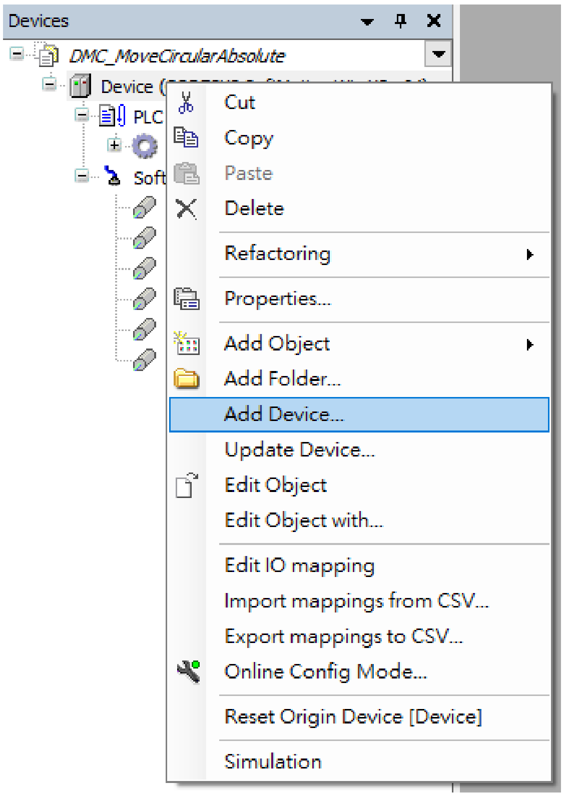
DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**
  - First, add axes to SoftMotion General Axis Pool in the project. In this example, six virtual axes have been established, i.e. AxisX, AxisY, AxisZ, AxisA, AxisB and AxisC.

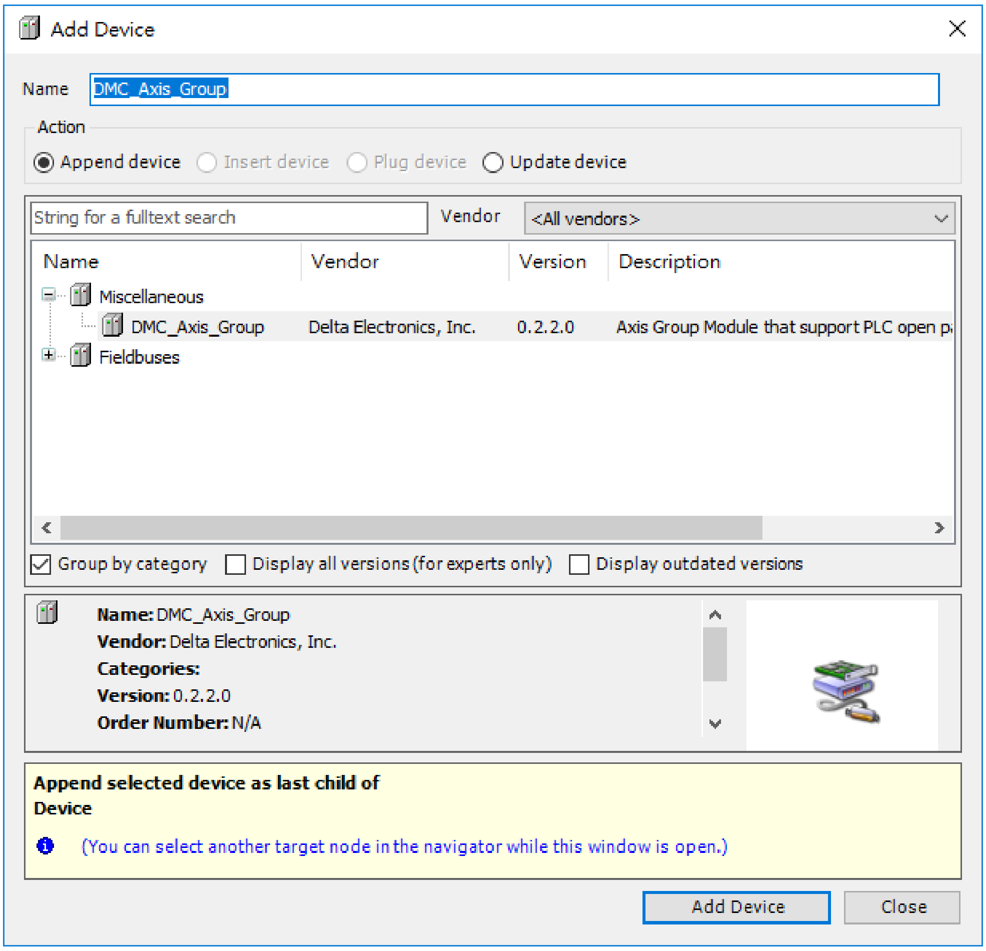


Devices
DMC_MoveCircularAbsolute
Device
PLC Logic
Application
SoftMotion General Axis Pool
AxisX (SM_Drive_Virtual)
AxisY (SM_Drive_Virtual)
AxisZ (SM_Drive_Virtual)
AxisA (SM_Drive_Virtual)
AxisB (SM_Drive_Virtual)
AxisC (SM_Drive_Virtual)

  - Right-click on **Device** in the project and then choose “Add Device”.



- After right-clicking on **Device** and selecting **Add device,** find **DMC_ Axis_Group** and then click **Add Device** button.



Add Device

Name: DMC_Axis_Group

Action

● Append device ○ Insert device ○ Plug device ○ Update device

String for a fulltext search Vendor <All vendors>

Name	Vendor	Version	Description
Miscellaneous			
DMC_Axis_Group	Delta Electronics, Inc.	0.2.2.0	Axis Group Module that support PLC open p
Fieldbuses			

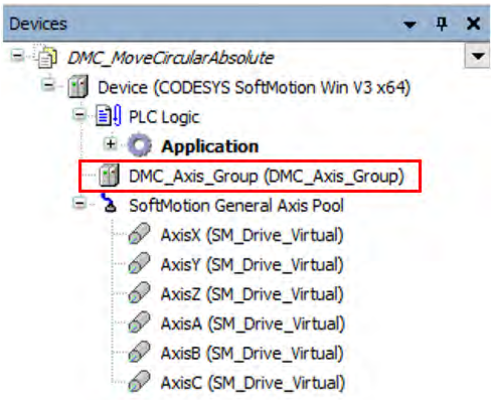
☑ Group by category □ Display all versions (for experts only) □ Display outdated versions

Name: DMC_Axis_Group
Vendor: Delta Electronics, Inc.
Categories:
Version: 0.2.2.0
Order Number: N/A

Append selected device as last child of
Device

(You can select another target node in the navigator while this window is open.)

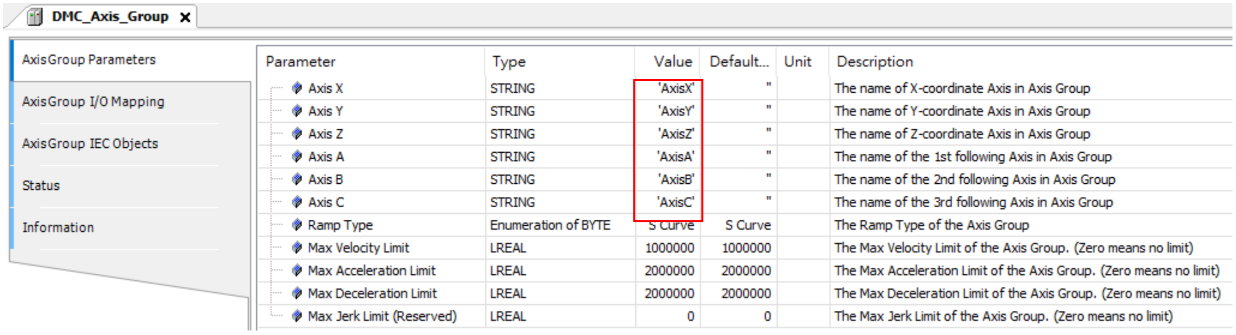
- Once **DMC_Axis_Group (DMC_Axis_Group)** appears in **Device**, it indicates that adding the axis group is successful.



Devices

```
DMC_MoveCircularAbsolute
  Device (CODESYS SoftMotion Win V3 x64)
    PLC Logic
      Application
    DMC_Axis_Group (DMC_Axis_Group)
    SoftMotion General Axis Pool
      AxisX (SM_Drive_Virtual)
      AxisY (SM_Drive_Virtual)
      AxisZ (SM_Drive_Virtual)
      AxisA (SM_Drive_Virtual)
      AxisB (SM_Drive_Virtual)
      AxisC (SM_Drive_Virtual)
```

- ■ Click **DMC_Axis_Group** setting page and then select **AxisGroup Parameters** item. In the **Parameter** column, AxisX~AxisC represent axes 1 ~ 6 in the axis group. Fill in the value field of the Axis X~Axis C with the names of the previously created virtual axes ”AxisX”~”AxisC”, as shown in the red box below. The axis group in this example uses 6 axes, AxisX, AxisY, AxisZ, AxisA, AxisB and AxisC.



Parameter	Type	Value	Default...	Unit	Description
Axis X	STRING	'AxisX'	''		The name of X-coordinate Axis in Axis Group
Axis Y	STRING	'AxisY'	''		The name of Y-coordinate Axis in Axis Group
Axis Z	STRING	'AxisZ'	''		The name of Z-coordinate Axis in Axis Group
Axis A	STRING	'AxisA'	''		The name of the 1st following Axis in Axis Group
Axis B	STRING	'AxisB'	''		The name of the 2nd following Axis in Axis Group
Axis C	STRING	'AxisC'	''		The name of the 3rd following Axis in Axis Group
Ramp Type	Enumeration of BYTE	S Curve	S Curve		The Ramp Type of the Axis Group
Max Velocity Limit	LREAL	1000000	1000000		The Max Velocity Limit of the Axis Group. (Zero means no limit)
Max Acceleration Limit	LREAL	2000000	2000000		The Max Acceleration Limit of the Axis Group. (Zero means no limit)
Max Deceleration Limit	LREAL	2000000	2000000		The Max Deceleration Limit of the Axis Group. (Zero means no limit)
Max Jerk Limit (Reserved)	LREAL	0	0		The Max Jerk Limit of the Axis Group. (Zero means no limit)

- ■ AxisX ~ AxisC in the AxisGroup Parameters represent the axes 1 ~ 6 in the axis group respectively, which denotes a 6D space, i.e. coordinate axes X, Y, Z, A, B and C. No value is required for the coordinate axis which is not set .

- ■ If the Value field for axes in the AxisGroup Parameters is not filled in with the names of axes, no error will occur when DMC_GroupEnable is started. However, the axis group will report an error if it starts to move the axis the name of which is not entered in the Value field.

- ■ If there are same axis names or invalid axis names in the Value field for axes in the AxisGroup Parameters, an error will occur when DMC_GroupEnable is executed.

- ■ At least one axis is specified in the **Value** field for axes in the AxisGroup Parameters. Otherwise, an error will occur when DMC_GroupEnable is executed.

- ■ Whether the specified single axis is in Standstill state or not will not be judged when DMC_GroupEnable is executed. After the axis group is created, the state of the axis group will be switched from GroupDisable to GroupStandby and the axes in the axis group will maintain the current state.

- ■ If axes in the group are in ErrorSTOP state, the axis group state will change from GroupDisable->GroupStandby->GroupErrorStop.

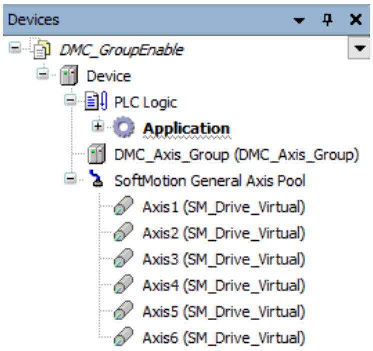
- ■ For more details on axis states, please refer to **Axis State Transitions** in **AX-3 Series Operation Manual**.

- ● **Troubleshooting**

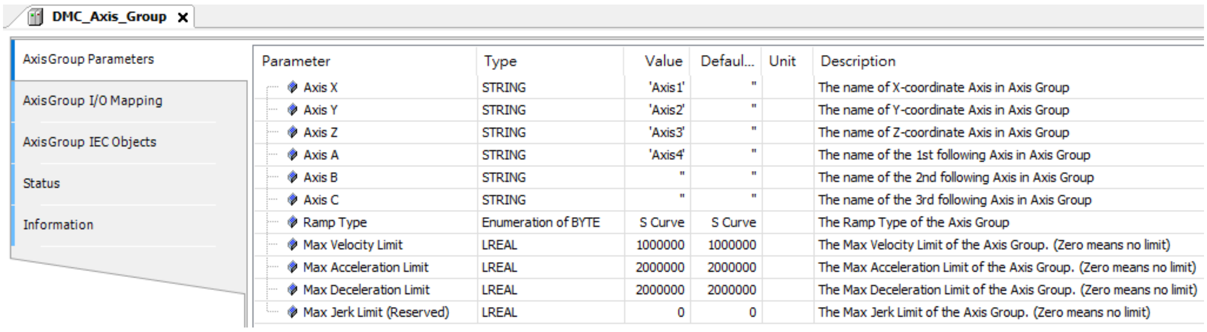
  - ■ When an error occurs during the instruction execution or the axis group enters GroupErrorstop state, *bError* will change to True and the axes will stop running. Refer to *ErrorID* (Error Code) to address the problem.
  - ■ For error codes and corresponding troubleshooting, please refer to Appendices of this manual.

- ● **Programming Example**

  - ■ Based on the limitation of the number of axes for simultaneous motion and the actual demand of axes, DMC_GroupEnable switches the axis group state from GroupDisable to GroupStandby for the upcoming axis group motion.



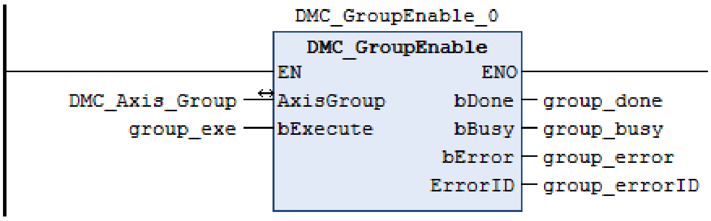
Devices
- DMC_GroupEnable
  - Device
    - PLC Logic
      - Application
    - DMC_Axis_Group (DMC_Axis_Group)
    - SoftMotion General Axis Pool
      - Axis1 (SM_Drive_Virtual)
      - Axis2 (SM_Drive_Virtual)
      - Axis3 (SM_Drive_Virtual)
      - Axis4 (SM_Drive_Virtual)
      - Axis5 (SM_Drive_Virtual)
      - Axis6 (SM_Drive_Virtual)



DMC_Axis_Group

AxisGroup Parameters
AxisGroup I/O Mapping
AxisGroup IEC Objects
Status
Information

Parameter	Type	Value	Defaul...	Unit	Description
Axis X	STRING	'Axis1'	"		The name of X-coordinate Axis in Axis Group
Axis Y	STRING	'Axis2'	"		The name of Y-coordinate Axis in Axis Group
Axis Z	STRING	'Axis3'	"		The name of Z-coordinate Axis in Axis Group
Axis A	STRING	'Axis4'	"		The name of the 1st following Axis in Axis Group
Axis B	STRING	"	"		The name of the 2nd following Axis in Axis Group
Axis C	STRING	"	"		The name of the 3rd following Axis in Axis Group
Ramp Type	Enumeration of BYTE	S Curve	S Curve		The Ramp Type of the Axis Group
Max Velocity Limit	LREAL	1000000	1000000		The Max Velocity Limit of the Axis Group. (Zero means no limit)
Max Acceleration Limit	LREAL	2000000	2000000		The Max Acceleration Limit of the Axis Group. (Zero means no limit)
Max Deceleration Limit	LREAL	2000000	2000000		The Max Deceleration Limit of the Axis Group. (Zero means no limit)
Max Jerk Limit (Reserved)	LREAL	0	0		The Max Jerk Limit of the Axis Group. (Zero means no limit)



1. Before the absolute interpolation motion of Axis1~ Axis4 is performed, create Axis1~Axis4 first, add them to the axis group DMC_Axis_Group and then input Axis1~Axis4 in the Value field for Parameter AxisX~AxisA in the setting page.

2. Use DMC_GroupEnable to create the axis group first before Axis 1 ~ Axis 4 perform the absolute interpolation of simultaneous motion of four axes.

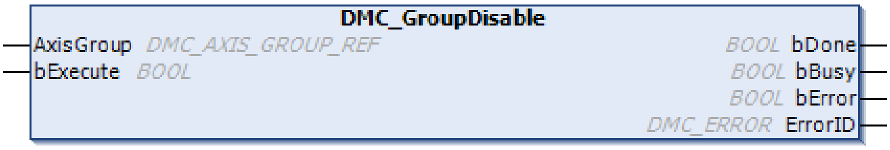
3. DMC_GroupEnable is triggered by changing group_exe (*bExecute*) to True. When group_done (*bDone*) changes to True, the axis group DMC_Axis_Group switches its state from GroupDisable to GroupStandby. The specified axes in the axis group maintain current state.

4. When DMC_GroupEnable is executed after the axis group is created, no error occurs and the axes enter Standstill state. Then the axis group DMC_Axis_Group can be used for the interpolation of simultaneous motion.

- **Supported Products**
  - AX-308E

2.3.2.2 DMC_GroupDisable

DMC_GroupDisable sets the state of an axis group to GroupDisable.

FB/FC	Instruction	Graphic expression
FB	DMC_GroupDisable	 DMC_GroupDisable AxisGroup DMC_AXIS_GROUP_REF bExecute BOOL BOOL bDone BOOL bBusy BOOL bError DMC_ERROR ErrorID

ST expression

```
DMC_GroupDisable_instance (
AxisGroup: = ,
bExecute: = ,
bDone=> ,
bBusy=> ,
bError=> ,
ErrorID=> )    ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-

- **Outputs**

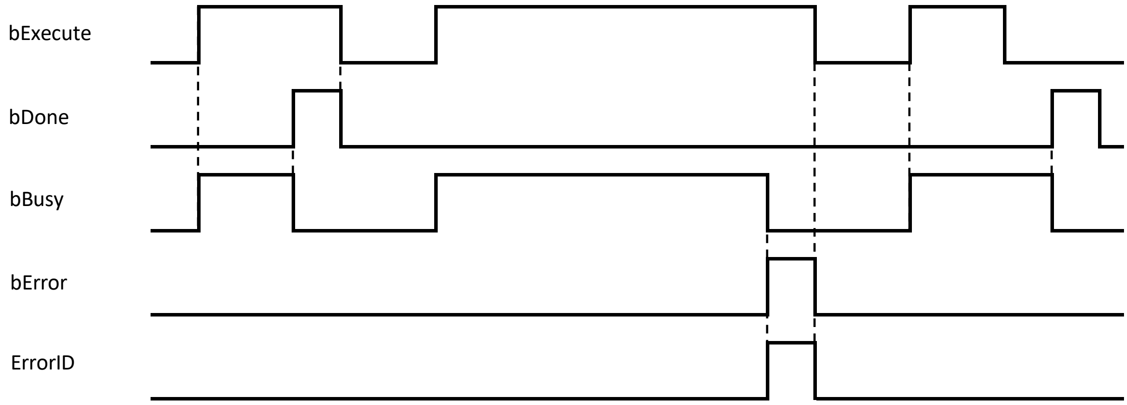
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

*Note: DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the instruction is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one cycle when *bExecute* is False but *bDone* changes to True.
bBusy	● When *bExecute* changes to TRUE.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError (ErrorID)	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared)

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group	DMC_AXIS_GROUP_REF*	DMC_AXIS_GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**

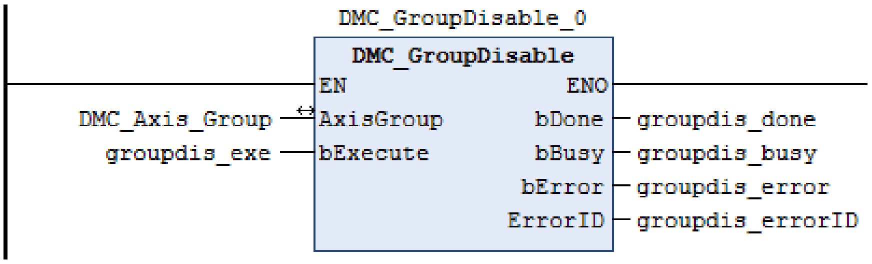
  - When this instruction is executed for an axis group, the axis group state will switch from GroupStandby to GroupDisable but the state of axes in the axis group will remain unchanged.
  - If the axis group is not in GroupStandby state, an error will occur when DMC_GroupDisable is executed.
  - For more details on axis states, please refer to **Axis State Transitions**.

- **Troubleshooting**

  - When an error occurs during the instruction execution or the axis group enters GroupErrorstop state, *bError* will change to True and the axes will stop running. Refer to *ErrorID* (Error Code) to address the problem.
  - For error codes and corresponding troubleshooting, please refer to **Appendices** of this manual.

- **Programming Example**

  - Switch the axis group state from GroupStandby to GroupDisable.

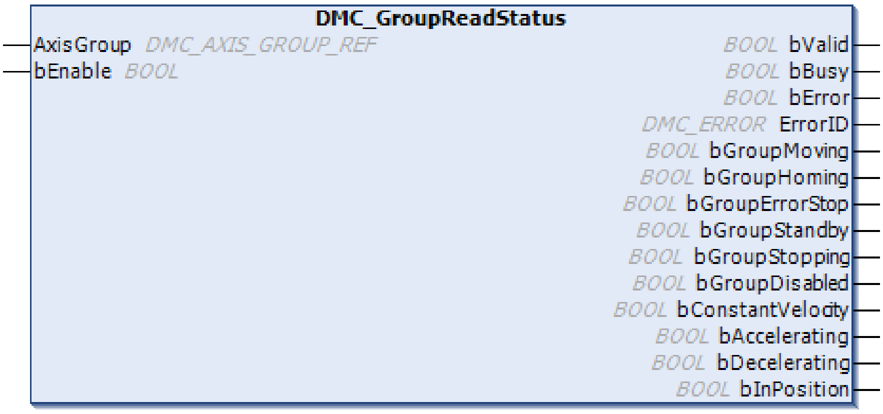


- This instruction enables the group axis DMC_Axis_Group specified by *AxisGroup* to enter the GroupDisable state.
- DMC_GroupDisable is executed when groupdis_exe (*bExecute*) changes to true. When groupdis_done (*bDone*) changes to true, it indicates that DMC_Axis_Group axis group has successfully entered GroupDisable state.

- **Supported Products**
  - AX-308E

2.3.2.3 DMC_GroupReadStatus

DMC_GroupReadStatus reads the state of an axis group.

FB/FC	Instruction	Graphic expression
FB	DMC_GroupReadStatus	

ST expression

```
DMC_GroupReadStatus_instance (
AxisGroup: = ,
bEnable: = ,
bValid=> ,
bBusy=> ,
bError=> ,
ErrorID=> ,
bGroupMoving=> ,
bGroupHoming=> ,
bGroupErrorStop=> ,
bGroupStandby=> ,
bGroupStopping=> ,
bGroupDisabled=> ,
bConstantVelocity=> ,
bAccelerating=> ,
bDecelerating=> ,
bInPosition=> )   ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *bEnable* changes from False to True.	BOOL	True/False (False)	-

- **Outputs**

Name	Function	Data type	Output range (Default value)
bValid	True when the output values are valid.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)
bGroupMoving	True when the axis group state is *bGroupMoving*.	BOOL	True/False (False)
bGroupHoming	True when the axis group state is *bGroupHoming.*	BOOL	True/False (False)
bGroupErrorStop	True when the axis group state is *bGroupErrorStop*.	BOOL	True/False (False)
bGroupStandby	True when the axis group state is *bGroupStandby*.	BOOL	True/False (False)
bGroupStopping	True when the axis group state is *bGroupStopping*.	BOOL	True/False (False)
bGroupDisabled	True when the axis group state is *bGroupDisabled*.	BOOL	True/False (False)
bConstantVelocity	True when the axis group runs at a constant velocity.	BOOL	True/False (False)
bAccelerating	True when the axis group accelerates.	BOOL	True/False (False)
bDecelerating	True when the axis group decelerates.	BOOL	True/False (False)
bInPosition	True when the axis group reaches the target position.	BOOL	True/False (False)

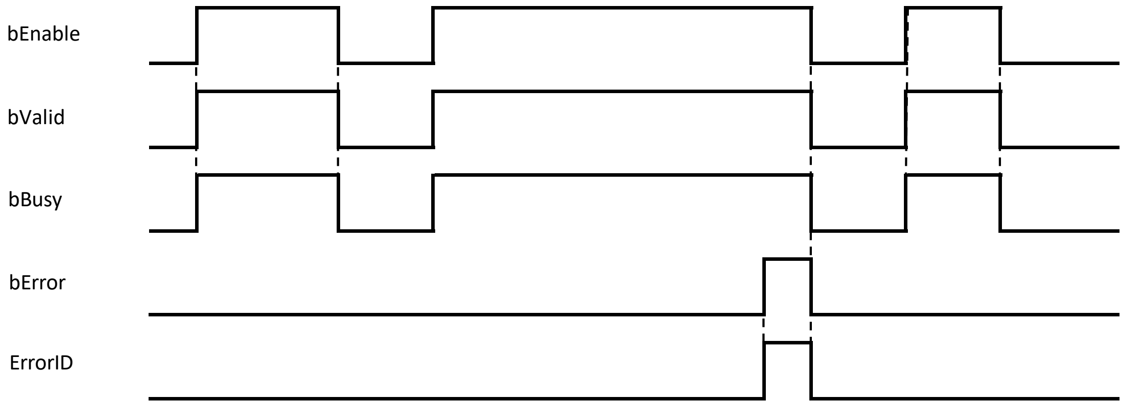
***Note:** DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* is True and other axis group state outputs are valid.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* changes to TRUE.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bError (ErrorID)	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bEnable* shifts from True to False. (Error code is cleared).
bGroupMoving	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.

Name	Timing for shifting to True	Timing for shifting to False
bGroupHoming	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bGroupErrorStop	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bGroupStandby	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bGroupStopping	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bGroupDisabled	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bConstantVelocity	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bAccelerating	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bDecelerating	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.
bInPosition	● When *bEnable* is True and the output keeps updating its value.	● When *bEnable* is True and the output keeps updating its value. ● When *bEnable* shifts to False. ● When *bError* shifts to True.

- **Timing Diagram**



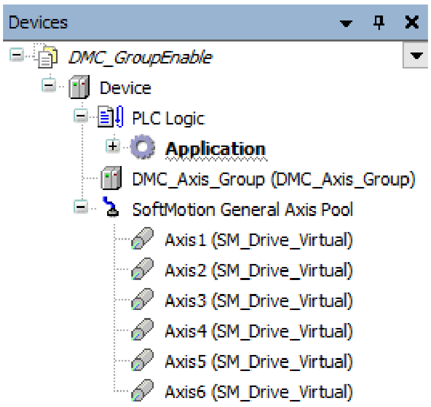
- **In-Outs**

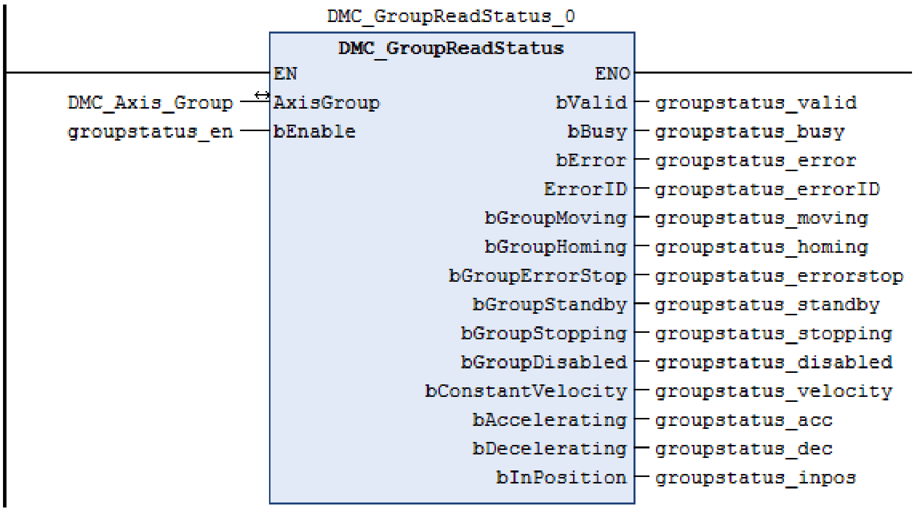
Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group	DMC_AXIS_GROUP_REF*	DMC_AXIS_GROUP_REF	When *bEnable* shifts to True and *bBusy* is False.

***Note:**

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**
  - DMC_GroupReadStatus can be used to read the state of an axis group.
- **Troubleshooting**
  - When an error occurs during the instruction execution or the axis group enters GroupErrorstop state, *bError* will change to True and the axes will stop running. Refer to *ErrorID* (Error Code) to address the problem.
  - For error codes and corresponding troubleshooting, please refer to **Appendices** of this manual.
- **Programming Example**
  - The example shows how DMC_GroupReadStatus is used to read the current state of the specified axis group



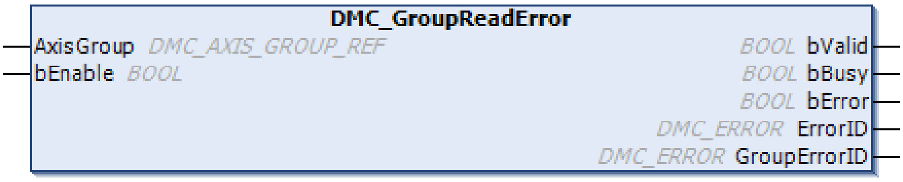


- Add DMC_Axis_Group in **Device** of the software.
- When groupstatus_valid (*bValid*) changes to True after groupstatus_en (*bEnable*) changes to True, DMC_GroupReadStatus reads the state of the axis group DMC_Axis_Group via its outputs.

- **Supported Products**
  - AX-308E

2.3.2.4 DMC_GroupReadError

DMC_GroupReadError reads axis group errors.

FB/FC	Instruction	Graphic expression
FB	DMC_GroupReadError	 DMC_GroupReadError AxisGroup DMC_AXIS_GROUP_REF bEnable BOOL BOOL bValid BOOL bBusy BOOL bError DMC_ERROR ErrorID DMC_ERROR GroupErrorID

ST expression

```
DMC_GroupReadError_instance (
AxisGroup: = ,
bEnable: = ,
bValid=> ,
bBusy=> ,
bError=> ,
ErrorID=> ,
GroupErrorID=> )    ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *bEnable* changes from False to True.	BOOL	True/False (False)	-

- **Outputs**

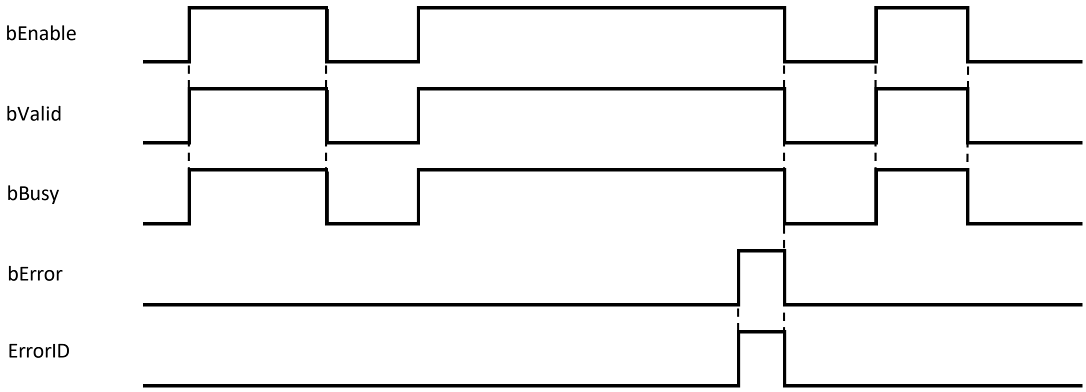
Name	Function	Data type	Output range (Default value)
bValid	True when the output value is valid.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to **Appendices** for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)
GroupErrorID	When the axis group is in ErrorStop state, the output shows an error code for the current axis group. Refer to **Appendices** for error code descriptions.	DMC_ERROR	DMC_ERROR (DMC_NoError)

***Note:** DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* is True and the output value is valid.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* is True	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bError (ErrorID)	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded in *ErrorID* and axis group error code is recorded in *GroupErrorID*).	● When *bEnable* shifts from True to False. (Both the error code in *ErrorID* and axis group error code in *GroupErrorID* are cleared)
GroupErrorID	● When *bEnable* is True and the output keeps updating.	● When *bEnable* is True and the output keeps updating.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group	DMC_AXIS_ GROUP_REF*	DMC_AXIS_ GROUP_REF	When *bEnable* shifts to True and *bBusy* is False.

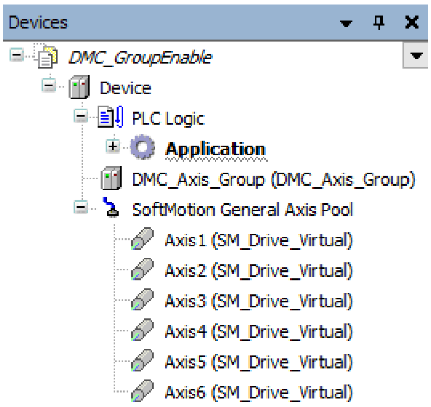
***Note:**

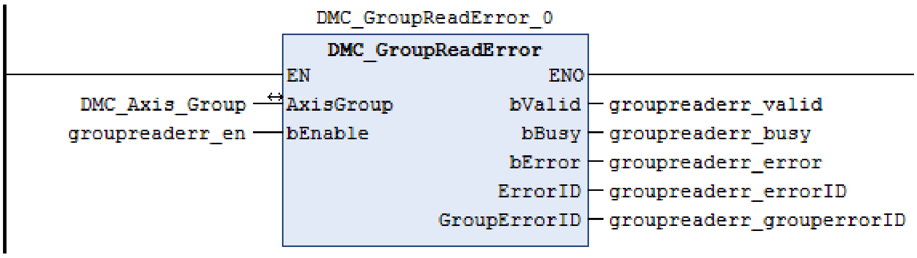
DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**
  - DMC_GroupReadError can be used to read axis group errors.
  - The value of *GroupErrorID* is 0 if no axis group error occurs.

- **Troubleshooting**
  - When an error occurs during the instruction execution or the axis group enters GroupErrorstop state, *bError* will change to True and the axes will stop running. Refer to *ErrorID* (Error Code) to address the problem.
  - For error codes and corresponding troubleshooting, please refer to **Appendices** in this manual.

- **Programming Example**
  - The example shows how DMC_GroupReadError is used to read an axis group error after the axis group is created.



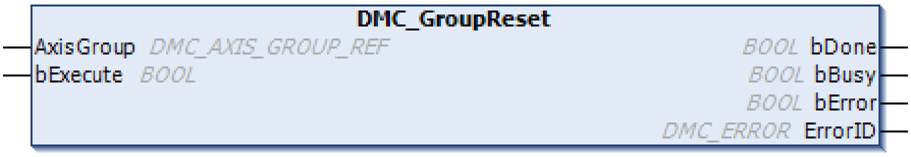


- Add DMC_Axis_Group in **Device**.
- When groupreaderr_valid (*bValid*) changes to True after groupreaderr_en (*bEnable*) changes to True, DMC_GroupReadError reads the state of the axis group DMC_Axis_Group via its output.

- **Supported Products**
  - AX-308E

2.3.2.5 DMC_GroupReset

DMC_GroupReset resets an axis group which is in GroupErrorstop state.

FB/FC	Instruction	Graphic expression
FB	DMC_GroupReset	 DMC_GroupReset AxisGroup DMC_AXIS_GROUP_REF bExecute BOOL BOOL bDone BOOL bBusy BOOL bError DMC_ERROR ErrorID

ST expression

DMC_GroupReset_instance (
AxisGroup: = ,
bExecute: = ,
bDone=> ,
bBusy=> ,
bError=> ,
ErrorID=>) ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-

- **Outputs**

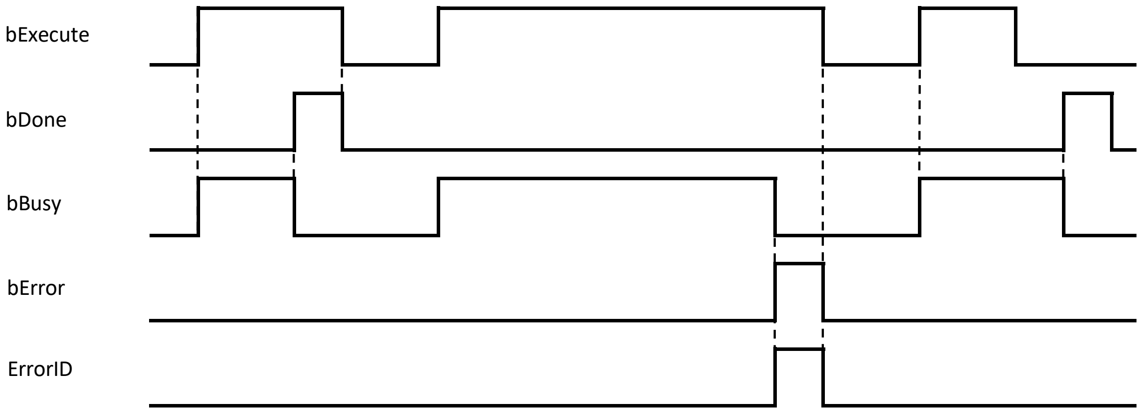
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

*Note: DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When axis group errors clearing is completed.	●When *bExecute* shifts to False. ●*bDone* will change to False after remaining True for one cycle when *bExecute* is False but *bDone* changes to True.
bBusy	● When bExecute changes to TRUE.	●When *bDone* shifts to True. ●When *bError* shifts to True.
bError (ErrorID)	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	●When *bExecute* shifts from True to False. (Error code is cleared)

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
AxisGroup	Specify the axis group	DMC_AXIS_ GROUP_REF*	DMC_AXIS_ GROUP_REF	When *bExecute* shifts to True and *bBusy* is False.

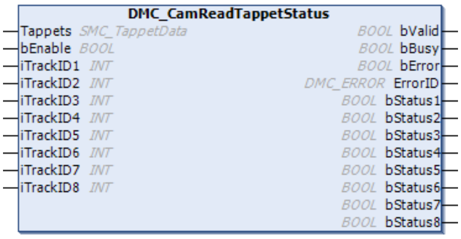
***Note:**

DMC_AXIS_GROUP_REF (FB): All function blocks for an axis group contain this variable, which works as the starting program for function blocks.

- **Function**
  - When an axis group is in GroupErrorstop state, DMC_GroupReset can be used to clear axis group errors and switch the axis group state to GroupStandby.
  - When the axis group enters the GroupStandby state, it indicates that the axis group motion can be performed.
- **Troubleshooting**
  - When an error occurs during the instruction execution or the axis group enters GroupErrorstop state, *bError* will change to True and the axes will stop running. Refer to *ErrorID* (Error Code) to address the problem.
  - For error codes and corresponding troubleshooting, please refer to **Appendices** of this manual.
- **Supported Products**
  - AX-308E

2.3.2.6 DMC_CamReadTappetStatus

DMC_CamReadTappetStatus reads the status of multiple tappets.

FB/FC	Instruction	Graphic expression
FB	DMC_CamReadTappetStatus	 DMC_CamReadTappetStatus Tappets SMC_TappetData bEnable BOOL iTrackID1 INT iTrackID2 INT iTrackID3 INT iTrackID4 INT iTrackID5 INT iTrackID6 INT iTrackID7 INT iTrackID8 INT BOOL bValid BOOL bBusy BOOL bError DMC_ERROR ErrorID BOOL bStatus1 BOOL bStatus2 BOOL bStatus3 BOOL bStatus4 BOOL bStatus5 BOOL bStatus6 BOOL bStatus7 BOOL bStatus8
ST expression		

DMC_CamReadTappetStatus_instance (
Tappets : =,
bEnable : =,
iTrackID1 : =,
iTrackID2 : =,
iTrackID3 : =,
iTrackID4 : =,
iTrackID5 : =,
iTrackID6 : =,
iTrackID7 : =,
iTrackID8 : =,
bValid =>,
bBusy =>,
bError =>,
ErrorID =>,
bStatus1 =>,
bStatus2 =>,
bStatus3 =>,
bStatus4 =>,
bStatus5 =>,
bStatus6 =>,
bStatus7 =>,
bStatus8 =>) ;		

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *bEnable* changes from False to True.	BOOL	True/False (True)	-
iTrackID1	Specify the tappet number.	INT	0~512 (0) *	When *bEnable* is True.
iTrackID2	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
iTrackID3	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.
iTrackID4	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.
iTrackID5	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.
iTrackID6	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.
iTrackID7	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.
iTrackID8	Specify the tappet number.	INT	0~512 (0)	When *bEnable* is True.

***Note:**

If the Track ID is set to 0, the corresponding output will not be used to read the tappet status.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bValid	True when the outputs are valid.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)
bStatus1	The status of the tappet specified by *iTrackID1.*	BOOL	True/False (False)
bStatus2	The status of the tappet specified by *iTrackID2.*	BOOL	True/False (False)
bStatus3	The status of the tappet specified by *iTrackID3.*	BOOL	True/False (False)
bStatus4	The status of the tappet specified by *iTrackID4.*	BOOL	True/False (False)
bStatus5	The status of the tappet specified by *iTrackID5.*	BOOL	True/False (False)
bStatus6	The status of the tappet specified by *iTrackID6.*	BOOL	True/False (False)
bStatus7	The status of the tappet specified by *iTrackID7.*	BOOL	True/False (False)

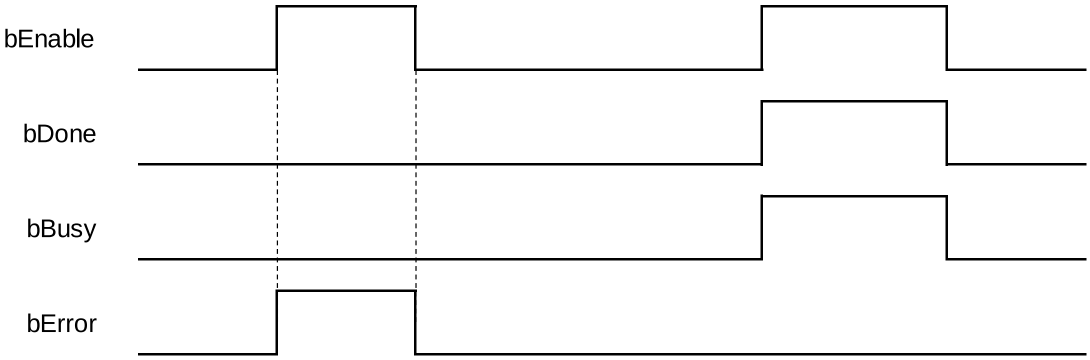
Name	Function	Data type	Output range (Default value)
bStatus8	The status of the tappet specified by *iTrackID8.*	BOOL	True/False (False)

***Note:** DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* shifts to True.	● When *bError* shifts to True. ● When *bEnable* shifts to False.
bBusy	● When *bEnable* shifts to True.	● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* shifts to False.
ErrorID		
bStatus1	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus2	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus3	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus4	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus5	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus6	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus7	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.
bStatus8	● When the status of the specified tappet is True.	● When the status of the specified tappet is False.

■ **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Tappets	Tappet signal source	SMC_TappetData	SMC_TappetData*	When *bEnable* shifts to True.

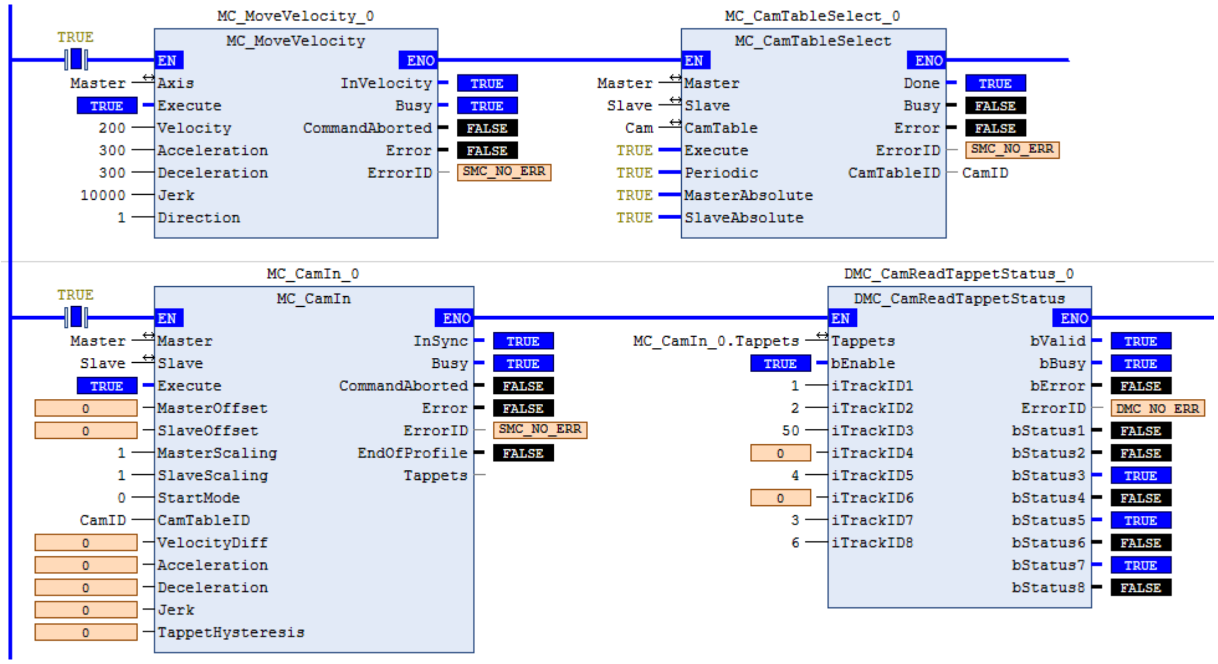
***Note:**

SMC_TappetData (STRUCT): the internal interface between MC_CamIn and SMC_GetTappetValue for tappet data trasmission.

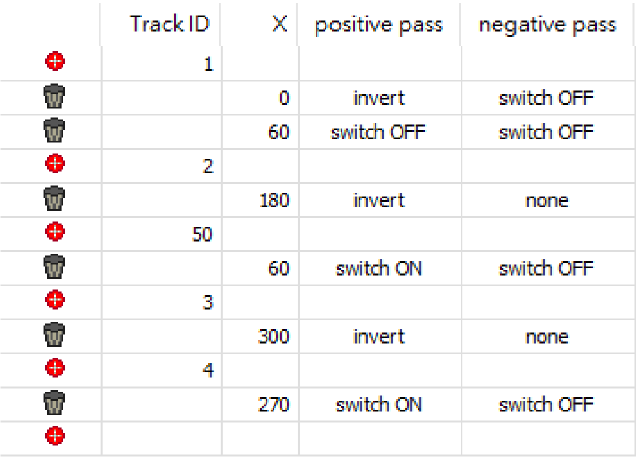
Name	Function	Data type	Setting range (Default value)
ctt	Specify the direction in which a tappet point is passed so that there will be an action then.	SMC_CAMTAPPETTYPE	0: TAPPET_pos (pass in the positive direction) 1: TAPPET_all (pass in both positive and negative directions) 2: TAPPET_neg (pass in the negative direction) (TAPPET_pos)
cta	Specify the action when the tappet point is passed.	SMC_CAMTAPPETACTION	0: TAPPETACTION_on (switch to ON) 1: TAPPETACTION_off (switch to OFF) 2: TAPPETACTION_inv (Invert) 3: TAPPETACTION_time (be ON for a period of time and then switch to OFF.) (TAPPETACTION_on)

Name	Function	Data type	Setting range (Default value)
dwDelay	Specify the delay time before the tappet changes to ON under TAPPETACTION_time mode.	DWORD	Positive number or 0 (0)
dwDuration	For how long the tappet is ON under TAPPETACTION_time mode.	DWORD	Positive number or 0 (0)
iGroupID	Specify the track ID of the tappet.	INT	Positive number, negative number or 0 (0)
x	Tappet position	LREAL	Positive number, negative number or 0 (0)
dwActive	The internal variable	DWORD	Positive number or 0 (0)

- **Function**
  - The instruction allows users to watch the states of eight tappets. The tappet number range for iTrackID1~8 is 0~512. If the setting is outside the range, the instruction will report an error, which indicates that the output status is disabled.
  - After *bEnable* changes to False, the instruction will not update the states of tappets any more and then outputs will maintain current tappet states.
- **Troubleshooting**
  - When an error occurs during the instruction execution, *bError* will change to True and the axes will stop running. Refer to *ErrorID* (Error Code) to address the problem.
- **Programming Example**
  - The example explains the action principle for CamReadTappetStatus.
  - Input 1 for *iTrackID1*, 2 for *iTrackID2*, 50 for *iTrackID3*, 4 for *iTrackID4*, 3 for *iTrackID7*. No tappet numbers for *iTrackID4*, *iTrackID6* and *iTrackID8* are given and thus there will be no actions on these tappets.

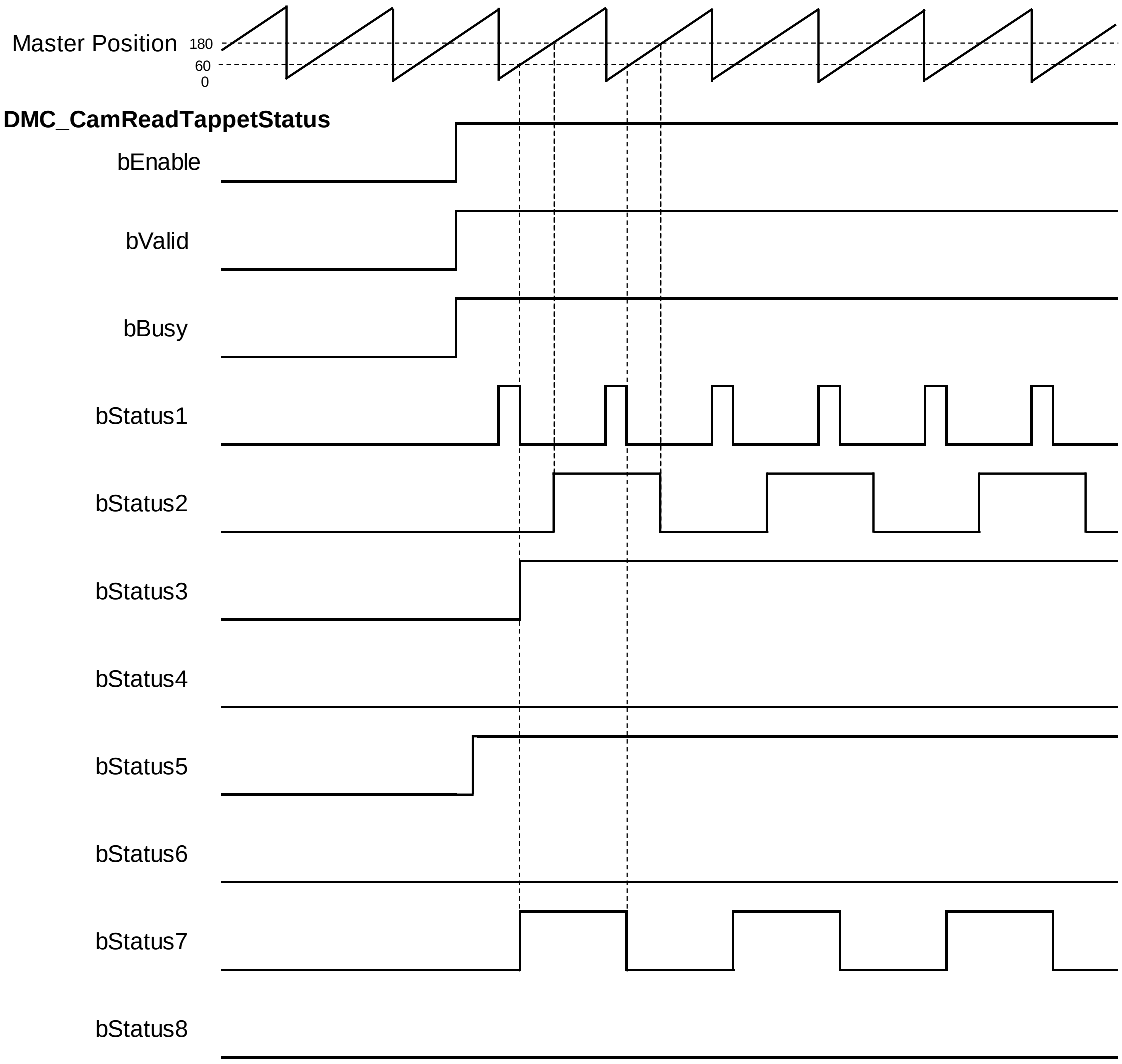


- Tappets Setting



	Track ID	X	positive pass	negative pass
⊕	1			
🗑		0	invert	switch OFF
🗑		60	switch OFF	switch OFF
⊕	2			
🗑		180	invert	none
⊕	50			
🗑		60	switch ON	switch OFF
⊕	3			
🗑		300	invert	none
⊕	4			
🗑		270	switch ON	switch OFF
⊕				

- Timing Diagram

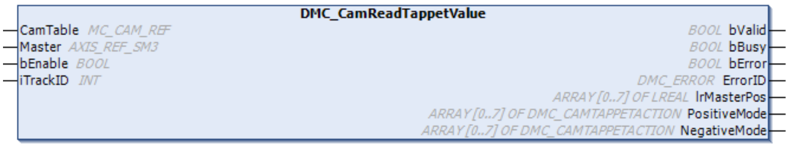


- When *bEnable* changes to True, DMC_CamReadTappetStatus starts to update the statuses of tappets.
- Take the second output point (*bStatus2*) for example. The corresponding tappet ID is 2 and the action is to invert its status when the position 180 is reached.
- When *bEnable* changes to False, the outputs maintain current statuses of tappets.

- **Supported Products**
  - AX-308E

2.3.2.7 DMC_CamReadTappetValue

DMC_CamReadTappetValue reads the data of one single tappet.

FB/FC	Instruction	Graphic expression
FB	DMC_CamReadTappetValue	 DMC_CamReadTappetValue CamTable MC_CAM_REF Master AXIS_REF_SM3 bEnable BOOL iTrackID INT BOOL bValid BOOL bBusy BOOL bError DMC_ERROR ErrorID ARRAY [0..7] OF LREAL lrMasterPos ARRAY [0..7] OF DMC_CAMTAPPETACTION PositiveMode ARRAY [0..7] OF DMC_CAMTAPPETACTION NegativeMode

ST expression

```
DMC_CamReadTappetValue_instance (
bEnable : =,
CamTable : =,
Master: =,
iTrackID: =,
bValid =>,
bBusy =>,
bError =>,
ErrorID =>,
lrMasterPos =>,
PositiveMode =>,
NegativeMode =>)    ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *bEnable* changes from False to True.	BOOL	True/False (True)	-
iTrackID	Specify the ID of the Track to be read.	INT	1~512 (0)	When *bEnable* is True.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bValid	True when the outputs are valid.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to	DMC_ERROR[*1]	DMC_ERROR (DMC_NoError)

Name	Function	Data type	Output range (Default value)
	Appendices for error code descriptions.		
lrMasterPos	The tappet position which is corresponded to master axis position (Unit: user unit)	LREAL[0..7][*2]	Positive number, negative number or 0 (0)[*3]
PositiveMode	Specify the mode for the tappet point when it is passed in the positive direction.	DMC_CAMTAPPETACTION[0..7][*2]	0: TAPPETACTION_none 1: TAPPETACTION_on 2: TAPPETACTION_off 3: TAPPETACTION_inv 4: TAPPETACTION_time (TAPPETACTION_none)
NegativeMode	Specify the mode for the tappet point when it is passed in the negative direction.	DMC_CAMTAPPETACTION[0..7][*2]	0: TAPPETACTION_none 1: TAPPETACTION_on 2: TAPPETACTION_off 3: TAPPETACTION_inv 4: TAPPETACTION_time (TAPPETACTION_none)

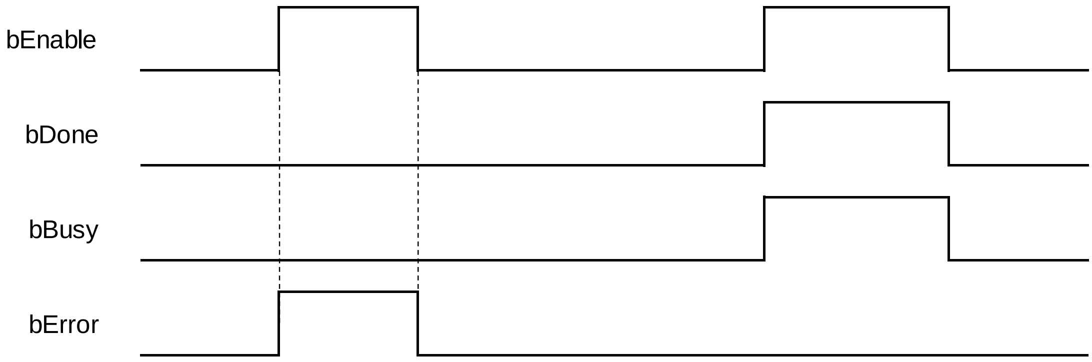
***Note:**

1. DMC_ERROR: Enumeration (ENUM)
2. One Track can have multiple tappet points set inside it. 8 tappet points at most can be read from the same Track via this instruction by default.
3. There is no tappet data to be output when *lrMasterPos* is set to 0 and *PositiveMode* and *NegativeMode* are both set to TAPPETACTION_none.

■ Output Update Timing

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* shifts to True.	● When *bError* shifts to True. ● When *bEnable* shifts to False.
bBusy	● When *bEnable* shifts to True.	● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* shifts to False.
ErrorID		

■ Timing Diagram



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	The specified cam table	MC_CAM_REF[*1]	MC_CAM_REF	When *bEnable* shifts to True.
Master	The reference master axis	AXIS_REF_SM3[*2]	AXIS_REF_SM3	

***Note:**

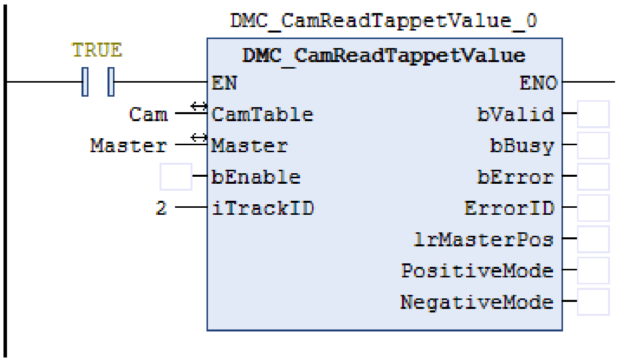
1. MC_CAM_REF (FB): The basic CAM
2. AXIS_REF_SM3 (FB): Generally, all motion function blocks have the InOut AXIS_REF_SM3.

- **Function**
  - A tappet table can be set in the Cam table. Multiple tappet points can be set for one Track. 8 tappet points at most can be read from the same tappet track via the function block.
  - The tappet data contains the master axis position corresponding to the tappet point, the positive passing mode and the negative passing mode. The modes include TAPPETACTION_none, TAPPETACTION_on, TAPPETACTION_off, TAPPETACTION_inv and TAPPETACTION_time
  - See the meanings of the modes in the following table.

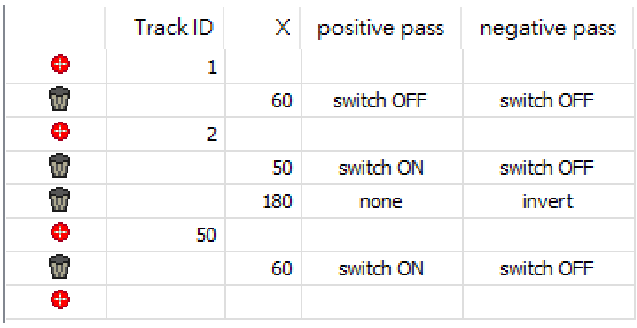
Mode	Function	Action
TAPPETACTION_none	No action	The tappet does not take an action when the master axis passes the point.
TAPPETACTION_on	ON	The tappet is enabled when the master axis passes the point.
TAPPETACTION_off	OFF	The tappet is disabled when the master axis passes the point.
TAPPETACTION_inv	Invert	The tappet status is inverted when the master axis passes the point.
TAPPETACTION_time	ON	When the master axis passes the point, the tappet is ON for a set period of time and then turns OFF.

- **Troubleshooting**
  - When an error occurs during the instruction execution, *bError* will change to True. Refer to *ErrorID* (Error Code) to address the problem.

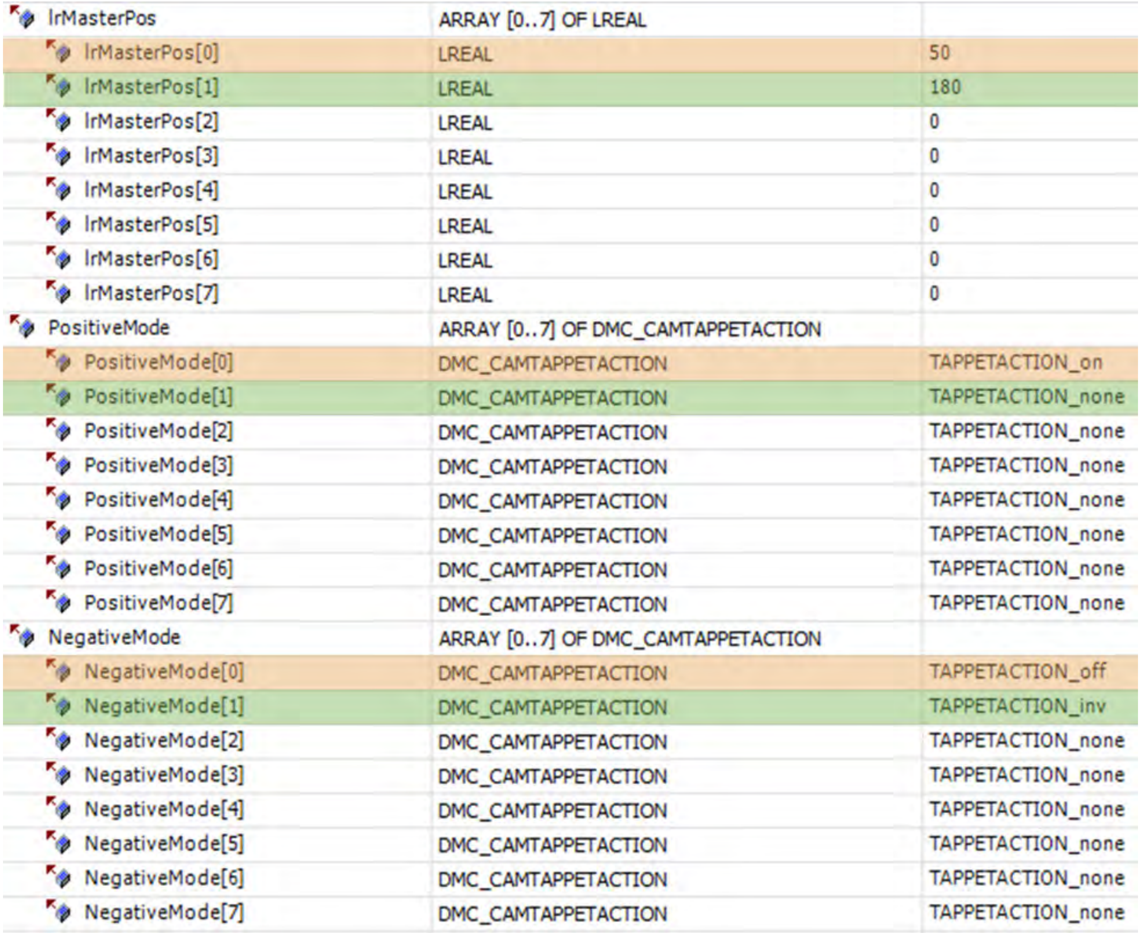
- **Programming Example**
  - The example explains the action principle for DMC_CamReadTappetValue and how to read the tappet data of Track ID 2.
  - The example explains the action principle for DMC_CamReadTappetValue by reading the tappet data of Track ID 2.



- Setting tappet points



	Track ID	X	positive pass	negative pass
⊕	1			
🗑		60	switch OFF	switch OFF
⊕	2			
🗑		50	switch ON	switch OFF
🗑		180	none	invert
⊕	50			
🗑		60	switch ON	switch OFF
⊕				



lrMasterPos	ARRAY [0..7] OF LREAL	
lrMasterPos[0]	LREAL	50
lrMasterPos[1]	LREAL	180
lrMasterPos[2]	LREAL	0
lrMasterPos[3]	LREAL	0
lrMasterPos[4]	LREAL	0
lrMasterPos[5]	LREAL	0
lrMasterPos[6]	LREAL	0
lrMasterPos[7]	LREAL	0
PositiveMode	ARRAY [0..7] OF DMC_CAMTAPPETACTION	
PositiveMode[0]	DMC_CAMTAPPETACTION	TAPPETACTION_on
PositiveMode[1]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PositiveMode[2]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PositiveMode[3]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PositiveMode[4]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PositiveMode[5]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PositiveMode[6]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PositiveMode[7]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NegativeMode	ARRAY [0..7] OF DMC_CAMTAPPETACTION	
NegativeMode[0]	DMC_CAMTAPPETACTION	TAPPETACTION_off
NegativeMode[1]	DMC_CAMTAPPETACTION	TAPPETACTION_inv
NegativeMode[2]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NegativeMode[3]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NegativeMode[4]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NegativeMode[5]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NegativeMode[6]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NegativeMode[7]	DMC_CAMTAPPETACTION	TAPPETACTION_none

- The tappet of Track ID 2 has two switch points: 50 and 180. Array 1 stores the data of switch position 50 and array 2 stores the data of switch position 180. The position -1 indicates no tappet switch data.

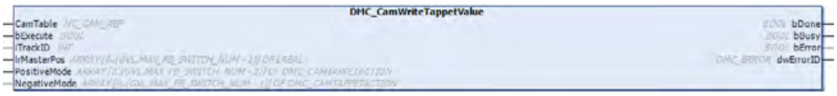
Track ID	MasterPosition	Positive Pass	Negative Pass
2	50	TAPPETACTION_on	TAPPETACTION_off
2	180	TAPPETACTION_none	TAPPETACTION_inv

- **Supported Products**

- AX-308E

2.3.2.8 DMC_CamWriteTappetValue

DMC_CamWriteTappetValue modifies the tappet data for the specified existing track.

FB/FC	Instruction	Graphic expression
FB	DMC_ CamWriteTappetValue	 DMC_CamWriteTappetValue CamTable MC_CAM_REF bExecute BOOL iTrackID INT lrMasterPos ARRAY[0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF LREAL PositiveMode ARRAY[0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPETACTION NegativeMode ARRAY[0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPETACTION BOOL bDone BOOL bBusy BOOL bError DMC_ERROR dwErrorID
ST expression		
DMC_CamWriteTappetValue_instance (CamTable : =, bExecute : =, iTrackID: =, lrMasterPosition : =, PositiveMode : =, NegativeMode : =, bDone =>, bBusy =>, bError =>, ErrorID =>) ;		

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (True)	-
iTrackID	Specify the Track ID where the tappet data is to be modified.	INT	1~512 (0)	When *bExecute* shifts from False to True.
lrMasterPos	The master axis position of the tappet point (Unit: user unit)	LREAL[0..7]*1	Positive number, negative number or 0 (-1)	When *bExecute* shifts from False to True.
PositiveMode	Specify the mode for the tappet point when it is passed in the positive direction.	DMC_CAMTAPPETACTION[0..7]*	0: TAPPETACTION_none 1: TAPPETACTION_on 2: TAPPETACTION_off 3: TAPPETACTION_inv 4: TAPPETACTION_time (TAPPETACTION_none)	When *bExecute* shifts from False to True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
NegativeMode	Specify the mode for the tappet point when it is passed in the negative direction.	DMC_CAMTAPPETACTION[0..7]*	0: TAPPETACTION_none 1: TAPPETACTION_on 2: TAPPETACTION_off 3: TAPPETACTION_inv 4: TAPPETACTION_time (TAPPETACTION_none)	When *bExecute* shifts from False to True.

***Note:**

One tappet track can be set with multiple tappet points. Maximum 8 tappet points can be written for one tappet track via the function block.

- **Outputs**

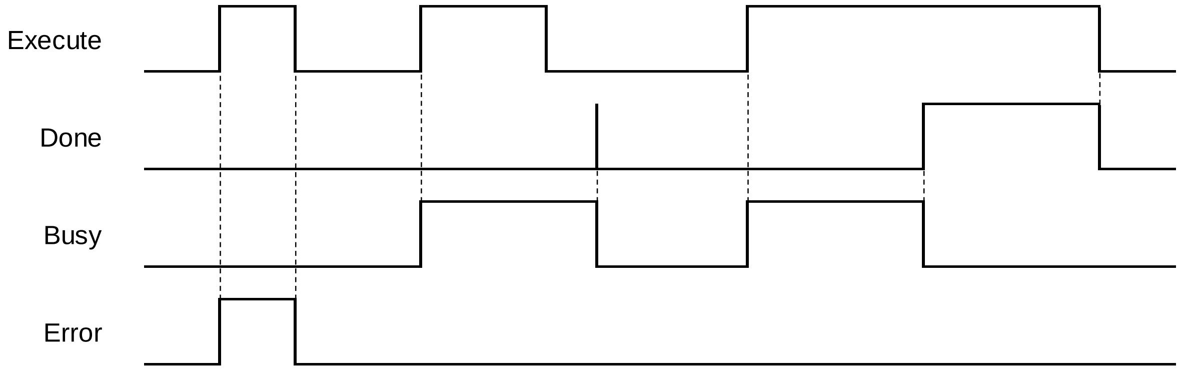
Name	Function	Data type	Output range (Default value)
bDone	When the output is valid.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

*Note: DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the instruction is completed.	● When *bError* shifts to True. ● When *bExecute* shifts to False.
bBusy	● When *bExecute* shifts to True.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts from True to False.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	The specified cam table	MC_CAM_REF*	MC_CAM_REF	When *bExecute* is True.

***Note:** MC_CAM_REF (FB): The basic CAM

- **Function**

  - The tappet table can be set in the cam table. Tappets can be used to track the current position of the master axis and output a signal to trigger an event under particular conditions.
  - One tappet table has multiple tappet tracks.
  - This function block can delete all original tappet points in the specified tappet track and replace them with maximum 8 tappet points which are descriped in the inputs of the function block.
  - The Tappet Track
    - One tappet track contains one track ID, one tappet switch (Boolean signal) and multiple tappet points.
  - The Tappet
    - One tappet point includes the Track ID of the tappet track where the tappet point is, the master axis position corresponding to the tappet point, the positive passing mode and the negative passing mode.

      See the modes and their meanings in the following table.

Mode	Function	Action
TAPPETACTION_none	No action	The tappet switch does not take an action when the master axis passes the point.
TAPPETACTION_on	ON	The tappet switch is enabled when the master axis passes the point.
TAPPETACTION_off	OFF	The tappet switch is disabled when the master axis passes the point.
TAPPETACTION_inv	Invert	The tappet switch status is inverted when the master axis passes the point.
TAPPETACTION_time	ON	When the master axis passes the point, the tappet switch is ON for a set period of time and then turns OFF.

***Note:**

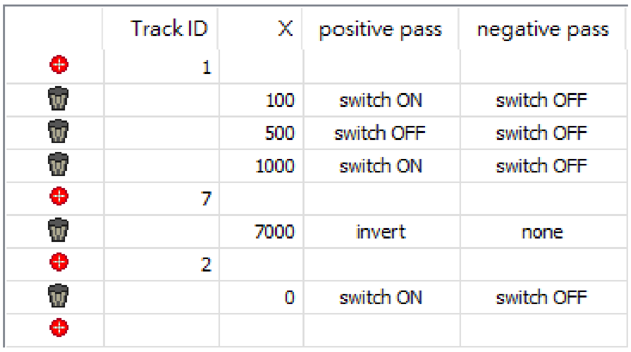
When the mode is set to TAPPETACTION_time in this function block, the tappet switch will turn off after being ON for a fixed 100 ms.

- **Troubleshooting**

  - When an error occurs during the instruction execution or the axis is in Errorstop state, *bError* will change to True. Refer to *ErrorID* (Error Code) to address the problem.

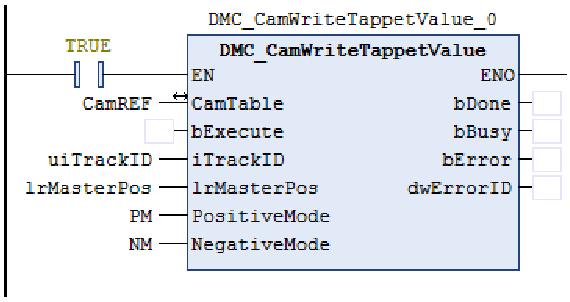
- **Programming Example**

  - The example explains the action principle for DMC_CamWriteTappetValue by writing the tappet data of Track ID 1.
  - Initial setting for tappet points

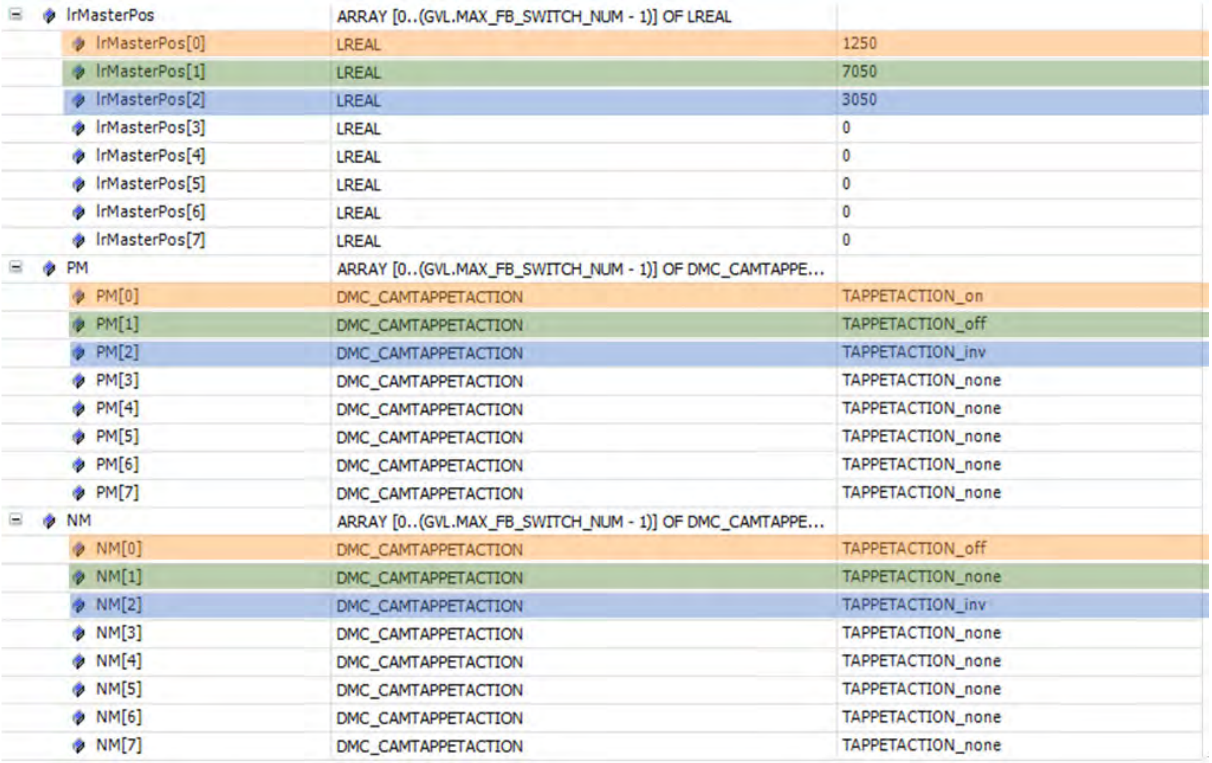


	Track ID	X	positive pass	negative pass
⊕	1			
🗑		100	switch ON	switch OFF
🗑		500	switch OFF	switch OFF
🗑		1000	switch ON	switch OFF
⊕	7			
🗑		7000	invert	none
⊕	2			
🗑		0	switch ON	switch OFF
⊕				

  - Function block setting



  - Input 1 for uiTrackID. Refer to the figure below for the setup of IrMasterPos, PositiveMode and NegativeMode.



⊟ lrMasterPos	ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF LREAL	
lrMasterPos[0]	LREAL	1250
lrMasterPos[1]	LREAL	7050
lrMasterPos[2]	LREAL	3050
lrMasterPos[3]	LREAL	0
lrMasterPos[4]	LREAL	0
lrMasterPos[5]	LREAL	0
lrMasterPos[6]	LREAL	0
lrMasterPos[7]	LREAL	0
⊟ PM	ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPE...	
PM[0]	DMC_CAMTAPPETACTION	TAPPETACTION_on
PM[1]	DMC_CAMTAPPETACTION	TAPPETACTION_off
PM[2]	DMC_CAMTAPPETACTION	TAPPETACTION_inv
PM[3]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[4]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[5]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[6]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[7]	DMC_CAMTAPPETACTION	TAPPETACTION_none
⊟ NM	ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPE...	
NM[0]	DMC_CAMTAPPETACTION	TAPPETACTION_off
NM[1]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[2]	DMC_CAMTAPPETACTION	TAPPETACTION_inv
NM[3]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[4]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[5]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[6]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[7]	DMC_CAMTAPPETACTION	TAPPETACTION_none

- The tappet table before the function block is executed

Track ID	Master axis position	Direction	Passing mode
1	100	Negative	TAPPETACTION_off
1	100	Positive	TAPPETACTION_on
1	500	Negative	TAPPETACTION_off
1	500	Positive	TAPPETACTION_off
1	1000	Negative	TAPPETACTION_off
1	1000	Positive	TAPPETACTION_on
7	7000	Positive	TAPPETACTION_inv
2	0	Negative	TAPPETACTION_off
2	0	Positive	TAPPETACTION_on

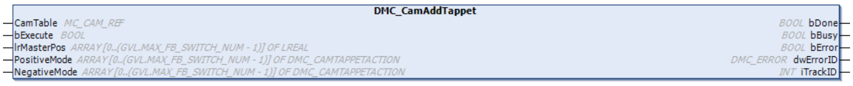
- The tappet table after the function block is executed

Track ID	Master axis position	Direction	Passing mode
1	1250	Negative	TAPPETACTION_off
1	1250	Positive	TAPPETACTION_on
1	7050	Positive	TAPPETACTION_off
1	3050	Negative	TAPPETACTION_inv
1	3050	Positive	TAPPETACTION_inv
7	7000	Positive	TAPPETACTION_inv
2	0	Negative	TAPPETACTION_off
2	0	Positive	TAPPETACTION_on

- **Supported Products**
  - AX308E

2.3.2.9 DMC_CamAddTappet

DMC_CamAddTappet adds a new tappet track at the end of the tappet table.

FB/FC	Instruction	Graphic expression
FB	DMC_CamAddTappet	 DMC_CamAddTappet CamTable MC_CAM_REF bExecute BOOL lrMasterPos ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF LREAL PositiveMode ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPETACTION NegativeMode ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPETACTION BOOL bDone BOOL bBusy BOOL bError DMC_ERROR dwErrorID INT iTrackID
ST expression		
DMC_CamAddTappet_instance (CamTable : =, bExecute : =, lrMasterPosition : =, PositiveMode : =, NegativeMode : =, bDone =>, bBusy =>, bError =>, ErrorID =>, iTrackID =>) ;		

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (True)	-
lrMasterPos	The master axis position of the tappet point (Unit: user unit)	LREAL[0..7]*	Positive number, negative number or 0 (-1)	When *bExecute* shifts from False to True.
PositiveMode	Specify the mode for the tappet point when it is passed in the positive direction.	DMC_CAM TAPPETACTI ON [0..7]*	0: TAPPETACTION_none 1: TAPPETACTION_on 2: TAPPETACTION_off 3: TAPPETACTION_inv 4: TAPPETACTION_time (TAPPETACTION_none)	When *bExecute* shifts from False to True.
NegativeMode	Specify the mode for the tappet point when	DMC_CAM TAPPETACTI	0: TAPPETACTION_none 1: TAPPETACTION_on 2: TAPPETACTION_off	When *bExecute* shifts from False to True.

Name	Function	Data type	Setting value (Default value)	Timing for updating
	it is passed in the negative direction.	ON [0..7]*	3: TAPPETACTION_inv 4: TAPPETACTION_time (TAPPETACTION_none)	

***Note:**
One tappet track can be set with multiple tappet points. Maximum 8 tappet points can be written for one tappet track via the function block.

- **Outputs**

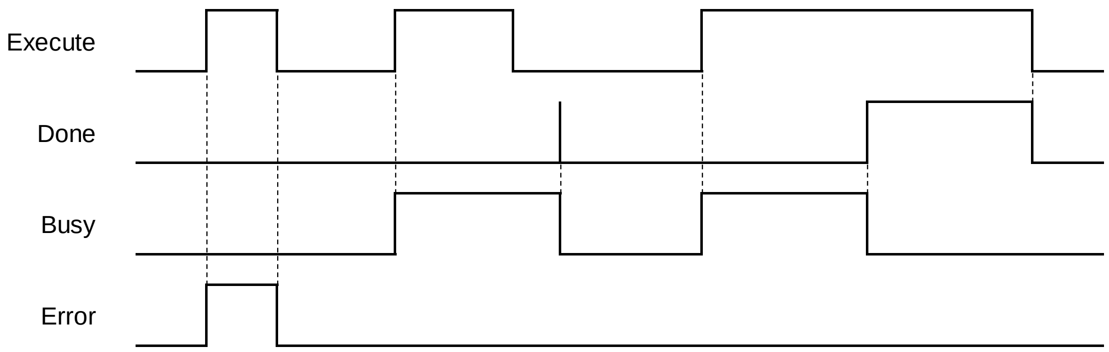
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)
iTrackID	The Track ID of the new tappet track	INT	1~512

*Note: DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the instruction is completed.	● When *bError* is True. ● When *bExecute* shifts to False.
bBusy	● When *bExecute* is True.	● When *bDone* is True. ● When *bError* is True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts from True to False

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	The specified cam table	MC_CAM_REF	MC_CAM_REF	When *bExecute* changes to True.

*Note:

MC_CAM_REF (FB): The basic CAM.

- **Function**

  - The tappet table can be set in the cam table. Tappets can be used to track the current position of the master axis and output a signal to trigger an event under particular conditions.
  - One tappet table has multiple tappet tracks.
  - This function block adds a tappet track and outputs its track ID to its output *uiTappetNum*. The track ID is the smallest one which has not been used yet.
  - The tappet track
    - One tappet track contains one track ID, one tappet switch (Boolean signal) and multiple tappet points.
  - The tappet
    - One tappet point includes the Track ID of the tappet track where the tappet point is, the master axis position corresponding to the tappet point, the positive passing mode and the negative passing mode. See the modes and their meanings in the following table.

Mode	Function	Action
TAPPETACTION_none	No action	The tappet switch does not take an action when the master axis passes the point.
TAPPETACTION_on	ON	The tappet switch is enabled when the master axis passes the point.
TAPPETACTION_off	OFF	The tappet switch is disabled when the master axis passes the point.
TAPPETACTION_inv	Invert	The tappet switch status is inverted when the master axis passes the point.
TAPPETACTION_time	ON	When the master axis passes the point, the tappet switch is ON for a set period of time and then turns OFF.

***Note:**

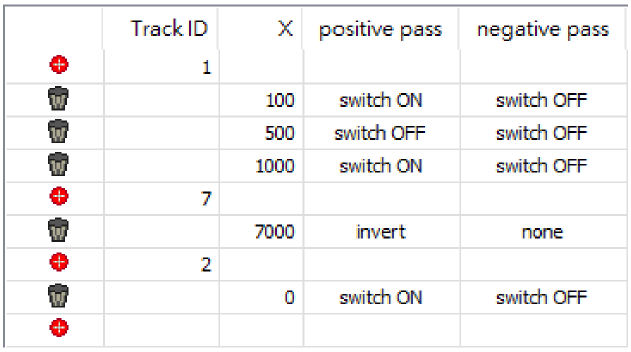
When the mode is set to TAPPETACTION_time in this function block, the tappet switch will turn off after being ON for a fixed 100 ms.

- **Troubleshooting**

  - When an error occurs during the instruction execution or the axis is in Errorstop state, *bError* will change to True and the axis stops running. Refer to *ErrorID* (Error Code) to address the problem.

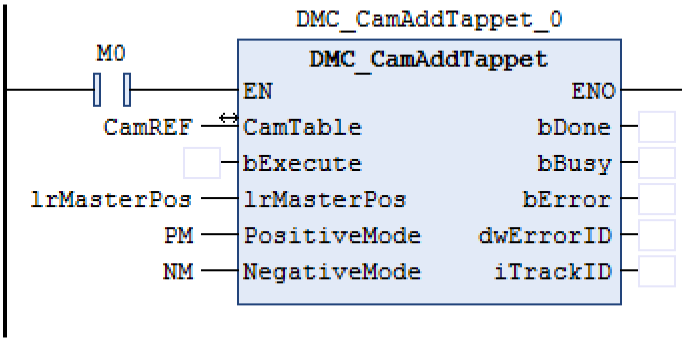
- **Programming Example**

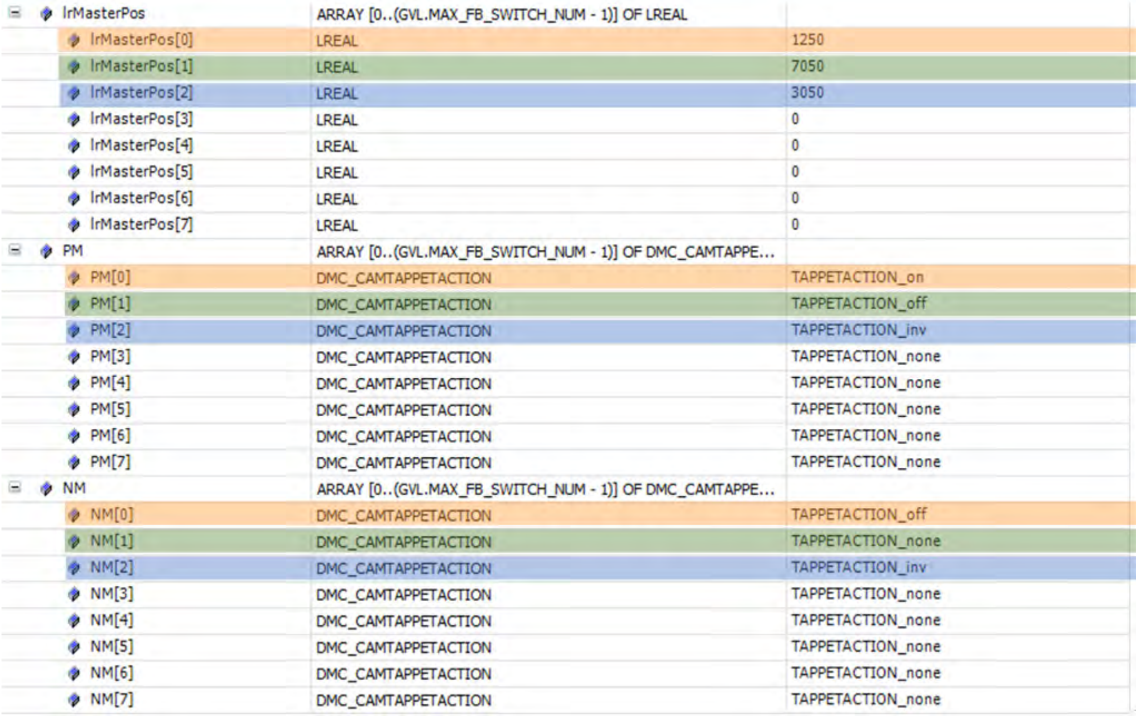
  - The example explains the action principle for DMC_CamAddTappetValue by adding a new track of tappet points.
  - Initial setting for tappet points



	Track ID	X	positive pass	negative pass
⊕	1			
🗑		100	switch ON	switch OFF
🗑		500	switch OFF	switch OFF
🗑		1000	switch ON	switch OFF
⊕	7			
🗑		7000	invert	none
⊕	2			
🗑		0	switch ON	switch OFF
⊕				

- Function block setting





⊟ lrMasterPos	ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF LREAL	
lrMasterPos[0]	LREAL	1250
lrMasterPos[1]	LREAL	7050
lrMasterPos[2]	LREAL	3050
lrMasterPos[3]	LREAL	0
lrMasterPos[4]	LREAL	0
lrMasterPos[5]	LREAL	0
lrMasterPos[6]	LREAL	0
lrMasterPos[7]	LREAL	0
⊟ PM	ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPE...	
PM[0]	DMC_CAMTAPPETACTION	TAPPETACTION_on
PM[1]	DMC_CAMTAPPETACTION	TAPPETACTION_off
PM[2]	DMC_CAMTAPPETACTION	TAPPETACTION_inv
PM[3]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[4]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[5]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[6]	DMC_CAMTAPPETACTION	TAPPETACTION_none
PM[7]	DMC_CAMTAPPETACTION	TAPPETACTION_none
⊟ NM	ARRAY [0..(GVL.MAX_FB_SWITCH_NUM - 1)] OF DMC_CAMTAPPE...	
NM[0]	DMC_CAMTAPPETACTION	TAPPETACTION_off
NM[1]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[2]	DMC_CAMTAPPETACTION	TAPPETACTION_inv
NM[3]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[4]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[5]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[6]	DMC_CAMTAPPETACTION	TAPPETACTION_none
NM[7]	DMC_CAMTAPPETACTION	TAPPETACTION_none

- See the tappet table before the function block is executed

Track ID	Master axis position	Direction	Mode
1	100	Negative	TAPPETACTION_off
1	100	Positive	TAPPETACTION_on
1	500	Negative	TAPPETACTION_off
1	500	Positive	TAPPETACTION_off
1	1000	Negative	TAPPETACTION_off
1	1000	Positive	TAPPETACTION_on
7	7000	Positive	TAPPETACTION_inv
2	0	Negative	TAPPETACTION_off
2	0	Positive	TAPPETACTION_on

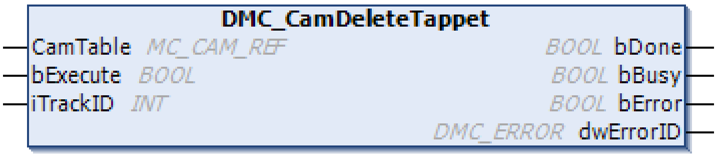
- See the tappet table after the function block is executed

Track ID	Master axis position	Direction	Mode
1	100	Negative	TAPPETACTION_off
1	100	Positive	TAPPETACTION_on
1	500	Negative	TAPPETACTION_off
1	500	Positive	TAPPETACTION_off
1	1000	Negative	TAPPETACTION_off
1	1000	Positive	TAPPETACTION_on
7	7000	Positive	TAPPETACTION_inv
2	0	Negative	TAPPETACTION_off
2	0	Positive	TAPPETACTION_on
3	1250	Negative	TAPPETACTION_off
3	1250	Positive	TAPPETACTION_on
3	7050	Positive	TAPPETACTION_off
3	3050	Negative	TAPPETACTION_inv
3	3050	Positive	TAPPETACTION_inv

- **Supported Products**
  - AX308E

2.3.2.10 DMC_CamDeleteTappet

DMC_CamDeleteTappet deletes the specified tappet track.

FB/FC	Instruction	Graphic expression
FB	DMC_CamDeleteTappet	 DMC_CamDeleteTappet CamTable MC_CAM_REF bExecute BOOL iTrackID INT BOOL bDone BOOL bBusy BOOL bError DMC_ERROR dwErrorID

ST expression

DMC_CamDeleteTappet_instance (
CamTable : =,
bExecute : =,
iTrackID : =,
bDone =>,
bBusy =>,
bError =>,
dwErrorID =>)   ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (True)	-
iTrackID	Specify the ID of the track to be deleted.	INT	1~512 (0)	When *bExecute* is True.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

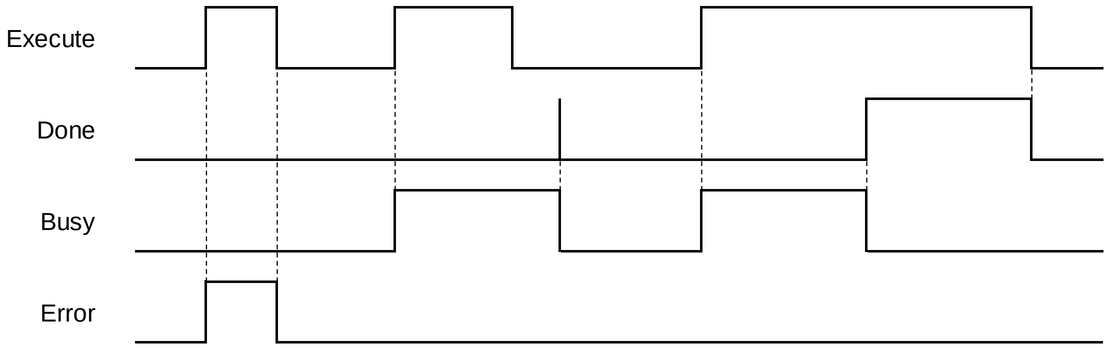
***Note:**

DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the instruction is completed.	● When *bError* is True. ● When *bExecute* shifts to False.
bBusy	● When *bExecute* is True.	● When *bDone* is True. ● When *bError* is True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts from True to False.
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	The specified cam table	MC_CAM_REF	MC_CAM_REF*	When *bExecute* changes to True.

***Note:**

MC_CAM_REF (FB): The basic CAM.

- **Function**

  - The tappet table can be set in the cam table. Tappets can be used to track the current position of the master axis and output a signal to trigger an event under particular conditions.
  - One tappet table has multiple tappet tracks.
  - The function block deletes a specified tappet track from the tappet table.
  - The tappet track
    - One tappet track contains one track ID, one tappet switch (Boolean signal) and multiple tappet points.
  - The tappet
    - One tappet point includes the Track ID of the tappet track where the tappet point is, the master axis position corresponding to the tappet point, the positive passing mode and the negative passing mode.

See the modes and their meanings in the following table.

Mode	Function	Action
TAPPETACTION_none	No action	The tappet switch does not take an action when the master axis passes the point.
TAPPETACTION_on	ON	The tappet switch is enabled when the master axis passes the point.
TAPPETACTION_off	OFF	The tappet switch is disabled when the master axis passes the point.
TAPPETACTION_inv	Invert	The tappet switch status is inverted when the master axis passes the point.
TAPPETACTION_time	ON	When the master axis passes the point, the tappet switch is ON for a set period of time and then turns OFF.

***Note:**

When the mode is set to TAPPETACTION_time in this function block, the tappet switch will turn off after being ON for a fixed 100 ms.

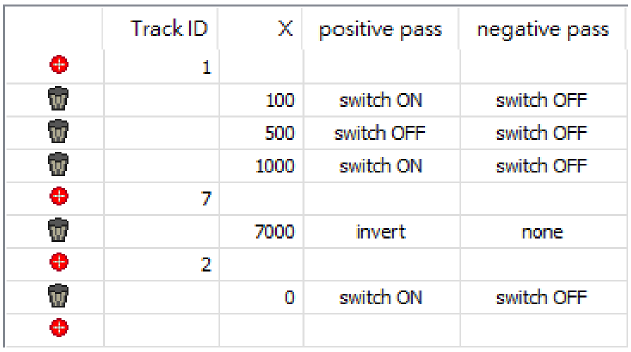
- **Troubleshooting**

  ■ When an error occurs during the instruction execution or the axis enters Errorstop state, *bError* will change to True and the axis stops running. Refer to *ErrorID* (Error Code) to address the problem.

- **Programming Example**

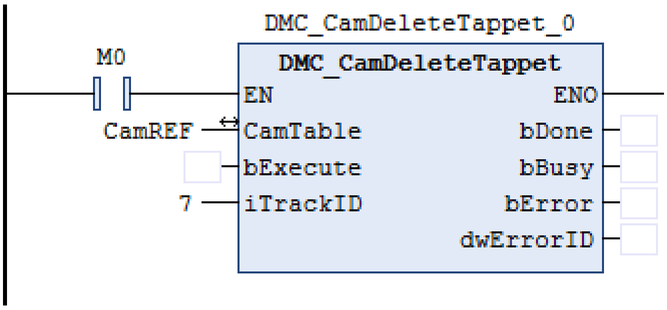
The example explains the action principle for DMC_CamDeleteTappet by deleting the specified track data from the tappet table.

■ Initial tappets setting



	Track ID	X	positive pass	negative pass
⊕	1			
🗑		100	switch ON	switch OFF
🗑		500	switch OFF	switch OFF
🗑		1000	switch ON	switch OFF
⊕	7			
🗑		7000	invert	none
⊕	2			
🗑		0	switch ON	switch OFF
⊕				

■ Function block setting



- See the tappet table before the function block is executed

Track ID	Master axis position	Direction	Mode
1	100	Negative	TAPPETACTION_off
1	100	Positive	TAPPETACTION_on
1	500	Positive and negative	TAPPETACTION_off
1	1000	Negative	TAPPETACTION_off
1	1000	Positive	TAPPETACTION_on
7	7000	Positive	TAPPETACTION_inv
2	0	Negative	TAPPETACTION_off
2	0	Positive	TAPPETACTION_on

- See the tappet table after the function block is executed

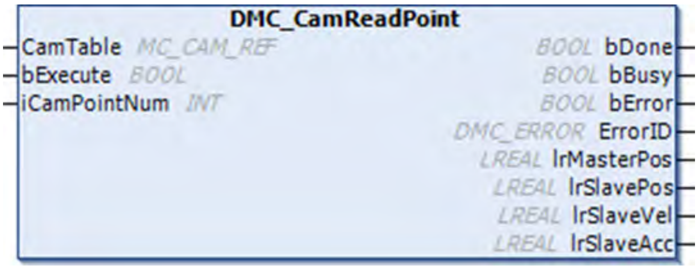
Track ID	Master axis position	Direction	Mode
1	100	Negative	TAPPETACTION_off
1	100	Positive	TAPPETACTION_on
1	500	Positive and negative	TAPPETACTION_off
1	1000	Negative	TAPPETACTION_off
1	1000	Positive	TAPPETACTION_on
2	0	Negative	TAPPETACTION_off
2	0	Positive	TAPPETACTION_on

- **Supported Products**

  - AX308E

2.3.2.11 DMC_CamReadPoint

DMC_CamReadPoint reads the data of one single cam point.

FB/FC	Instruction	Graphic expression
FB	DMC_CamReadPoint	

ST expression

```
DMC_CamReadPoint_instance (
CamTable : =parameter,
bExecute: =parameter,
iCamPointNum: =parameter,
bDone =>parameter,
bBusy =>parameter,
bError =>parameter,
ErrorID =>parameter,
lrMasterPos =>parameter,
lrSlavePos =>parameter,
lrSlaveVel =>parameter,
lrSlaveAcc =>parameter)    ;
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (True)	-
iCamPointNum	Specify the number of the cam point to be read.	INT	0~256 (0)	When *bExecute* is True.

- **Outputs**

Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NO_ERR)

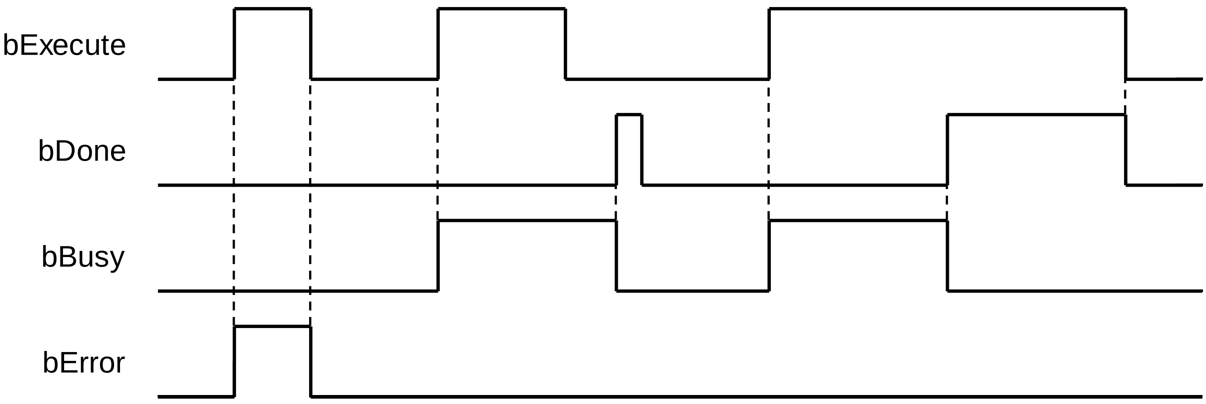
Name	Function	Data type	Output range (Default value)
lrMasterPos	The position of the cam master axis	LREAL	Positive number, negative number or 0 (0)
lrSlavePos	The position of the cam slave axis	LREAL	Positive number, negative number or 0 (0)
lrSlaveVel	The velocity of the cam slave axis	LREAL	Positive number, negative number or 0 (0)
lrSlaveAcc	The acceleration of the cam slave axis	LREAL	Positive number, negative number or 0 (0)

***Note:** DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When *bExecute* changes to True.	● When *bError* shifts to True. ● When *bExecute* shifts to False.
bBusy	● When *bExecute* changes to True.	● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False.

■ **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	The specified cam table	MC_CAM_REF	MC_CAM_REF*	When *bExecute* changes to True.

***Note:** MC_CAM_REF (FB): The basic CAM.

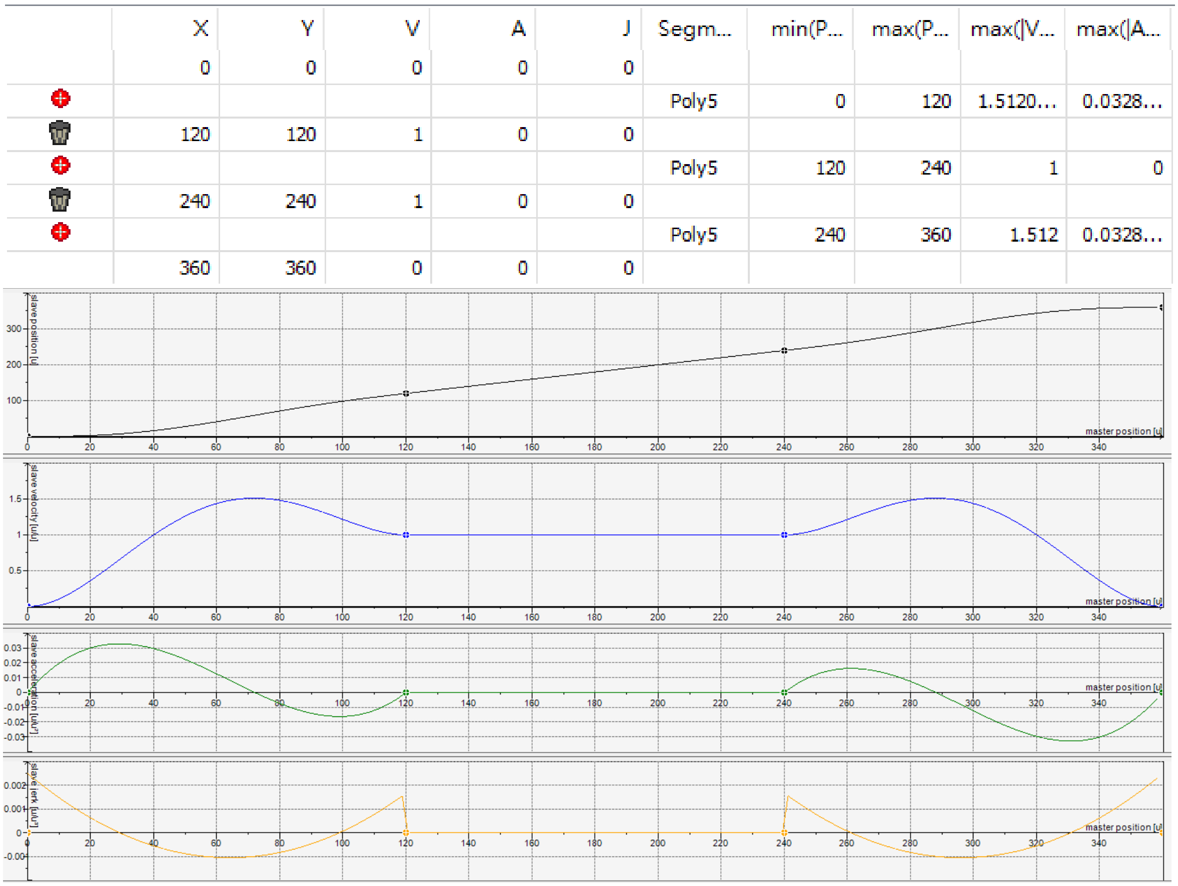
- **Function**

■ *CamTable* determines which cam table is to read. *iCamPointNum* determines the number of the cam point to read. *lrMasterPos* shows the master position that the cam point corresponds to. *lrSlavePos* shows the slave position that the cam point corresponds to, i.e. coordinates of the cam point. *lrSlaveVel* is the slave velocity that the cam point corresponds to. *lrSlaveAcc* is the slave acceleration rate that the cam point corresponds to.

■ If no data of the specified cam point exists, the output will show Infinity.

- **Programming Example**

1. Build a cam table: “Cam”.

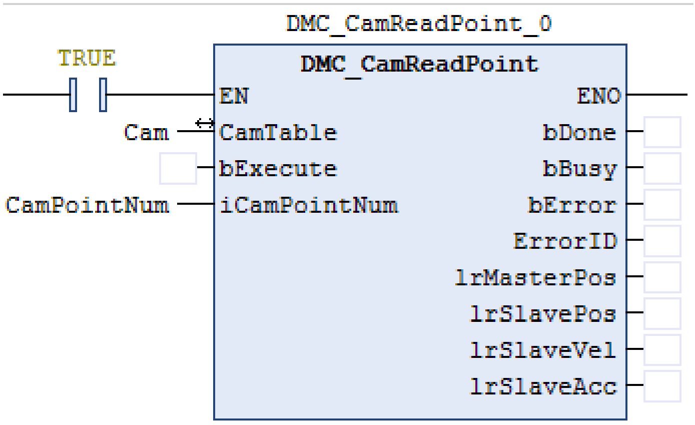


	X	Y	V	A	J	Segm...	min(P...	max(P...	max(|V...	max(|A...
	0	0	0	0	0					
⊕						Poly5	0	120	1.5120...	0.0328...
🗑	120	120	1	0	0					
⊕						Poly5	120	240	1	0
🗑	240	240	1	0	0					
⊕						Poly5	240	360	1.512	0.0328...
	360	360	0	0	0					

2. Input Cam for the specified *CamTable* and 2 for *CamPointNum*, the number of the cam point to be read.

3. Set DMC_CamReadPoint_0.bExecute to True.

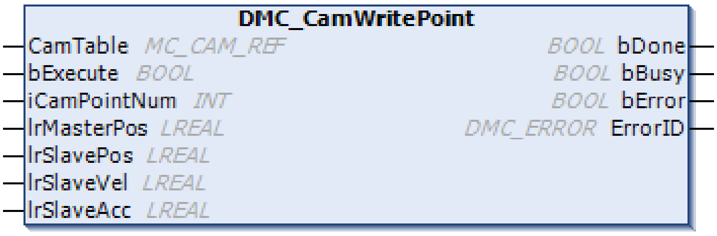
4. DMC_CamReadPoint_0.lrMasterPos：240, DMC_CamReadPoint_0.lrSlavePos：240, DMC_CamReadPoint_0.lrSlaveVel：1 and DMC_CamReadPoint_0.lrSlaveAcc：0 can be read while the instruction is waiting until DMC_CamReadPoint_0.bDone changes from False to True.



- **Supported Products**
  - AX308E

2.3.2.12 DMC_CamWritePoint

DMC_CamWritePoint writes the data of one single cam point.

FB/FC	Instruction	Graphic expression
FB	DMC_CamWritePoint	 DMC_CamWritePoint CamTable MC_CAM_REF bExecute BOOL iCamPointNum INT lrMasterPos LREAL lrSlavePos LREAL lrSlaveVel LREAL lrSlaveAcc LREAL BOOL bDone BOOL bBusy BOOL bError DMC_ERROR ErrorID
ST expression		
DMC_CamWritePoint_instance (CamTable : =, bExecute: =, bCamChangedPoint: =, lrMasterPos : =, lrSlavePos : =, lrSlaveVel : =, lrSlaveAcc : =, bDone =>, bBusy =>, bError =>, ErrorID =>) ;		

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (True)	-
iCamPointNum	Specify the number of the cam point to be written.	INT	0~256 (0)	When *bExecute* is True.
lrMasterPos	Specify the position of the cam master axis	LREAL	Positive number, negative number or 0 (0)	When *bExecute* is True.
lrSlavePos	Specify the position of the cam slave axis	LREAL	Positive number, negative number or 0 (0)	When *bExecute* is True.
lrSlaveVel	Specify the velocity of the cam slave axis	LREAL	Positive number, negative number or 0 (0)	When *bExecute* is True.
lrSlaveAcc	Specify the acceleration of the cam slave axis	LREAL	Positive number, negative number or 0 (0)	When *bExecute* is True.

- **Outputs**

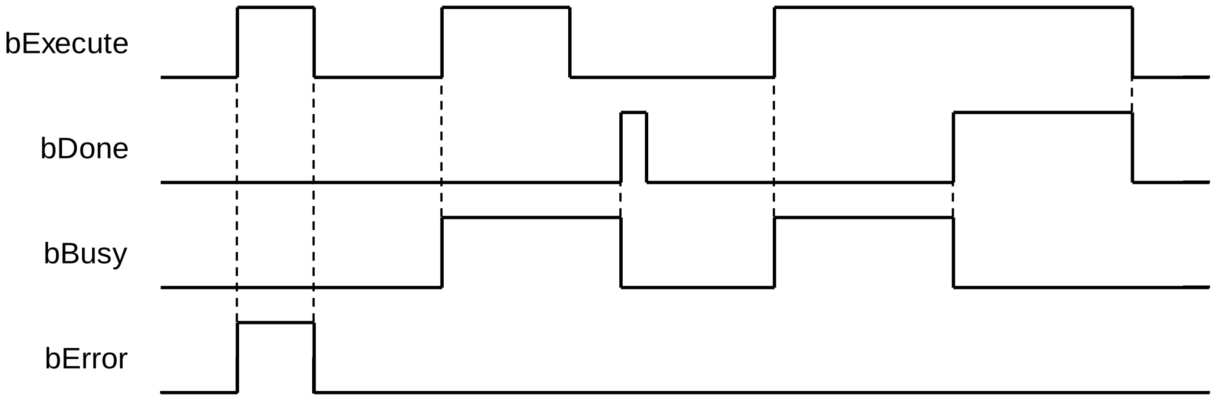
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NO_ERR)

***Note:** DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When *bExecute* changes to True.	● When *bError* shifts to True. ● When *bExecute* shifts to False.
bBusy	● When *bExecute* changes to True.	● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False.

■ **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	The specified cam table	MC_CAM_REF	MC_CAM_REF*	When *bExecute* changes to True.

***Note:** MC_CAM_REF (FB): The basic CAM.

- **Function**

  - The *CamTable* determines which cam table to write data to and *iCamPointNum* determines the cam point number to be written. *lrMasterPos* and *lrSlavePos* write respectively the master axis position and slave axis position of the cam data point (i.e. cam point coordinate positions). *lrSlaveVel* writes the slave axis velocity corresponding to the cam data point and *lrSlaveAcc* writes the slave axis acceleration corresponding to the cam data point.

  - The data point information accessed in the cam table corresponds to different data according to different cam data types. When DMC_CamWritePoint writes data, the cam operation will not be affected even if parameters are input to the inputs of the instruction if no specific data (e.g. *lrSlaveVel* and *lrSlaveAcc*) is accessed in the cam table.

  - When DMC_CamWritePoint modifies the cam table data in the synchronized cam motion, the slave axis in synchronization will change its path immediately, which may cause a jolt of the mechanism.

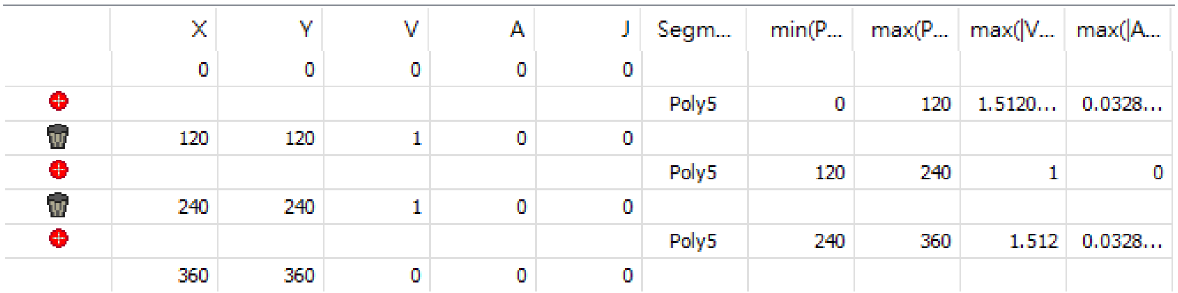
  - When the starting or ending cam data points are modified and the master axis position written by *lrMasterPos* exceeds the range of the original cam table, the running cam will have no change. And the cam table with the modified boundary range cannot work until the MC_Camtableselect is restarted.

- **Troubleshooting**

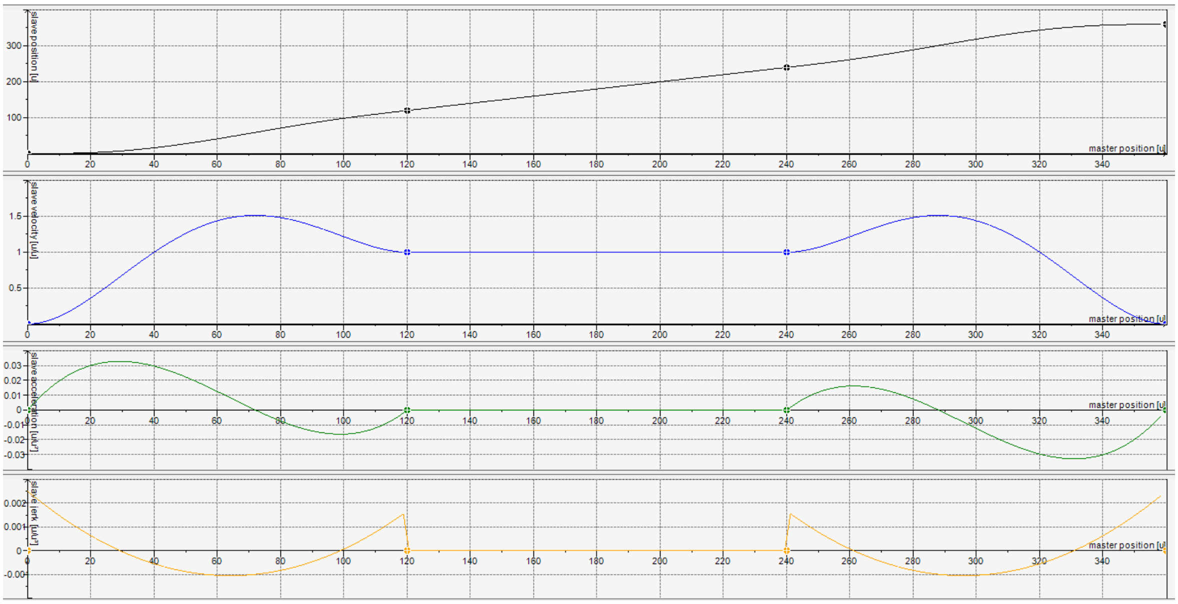
  - When an error occurs during the instruction execution or the axis enters Errorstop state, *bError* will change to True and the axis stops running. Refer to *ErrorID* (Error Code) to address the problem.

- **Programming Example**

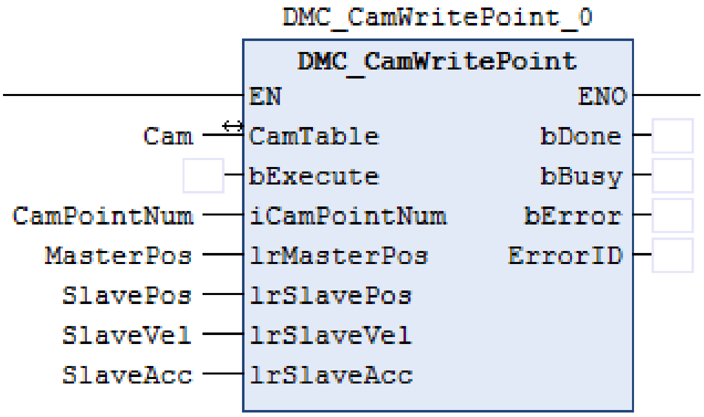
1. Build a cam table: "Cam".



	X	Y	V	A	J	Segm...	min(P...	max(P...	max(|V...	max(|A...
	0	0	0	0	0					
⊕						Poly5	0	120	1.5120...	0.0328...
🗑	120	120	1	0	0					
⊕						Poly5	120	240	1	0
🗑	240	240	1	0	0					
⊕						Poly5	240	360	1.512	0.0328...
	360	360	0	0	0					



2. Input Cam for the specified *CamTable* and 2 for *CamPointNum*, the number of the cam point to be written.
3. Input 300 for *MasterPos*, 250 for *SlavePos*, 2 for *SlaveVel*, 4 for *SlaveAcc* in the selected cam point data.
4. Set DMC_CamWritePoint_0.bExecute to True.
5. The data writing is completed when DMC_CamWritePoint_0.bDone changes from False to True.



6. After the writing is finished, the actual values for the cam table "Cam" are shown as below.

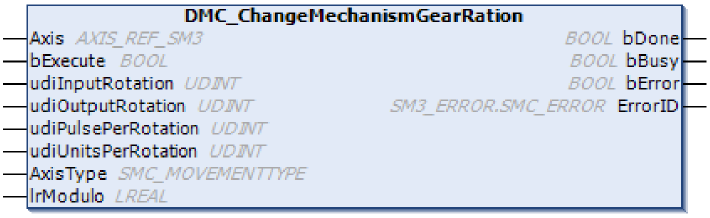
	X	Y	V	A
0	0	0	0	0
1	120	120	1	0
2	300	250	2	4
3	360	360	0	0

- **Supported Products**

  - AX308E

2.3.2.13 DMC_ChangeMechanismGearRation

DMC_ChangeMechanismGearRation modifies the ratio between user units and pulses, axis type and user units per rotation of the rotary axis.

FB/FC	Instruction	Graphic expression
FB	DMC_ChangeMechanismGearRation	 DMC_ChangeMechanismGearRation Axis AXIS_REF_SM3 bExecute BOOL udiInputRotation UDINT udiOutputRotation UDINT udiPulsePerRotation UDINT udiUnitsPerRotation UDINT AxisType SMC_MOVEMENTTYPE lrModulo LREAL BOOL bDone BOOL bBusy BOOL bError SM3_ERROR.SMC_ERROR ErrorID

ST expression
DMC_ReadMotionState_instance (Axis : =, bExecute: =, udiInputRotation: =, udiOutputRotation: =, udiPulsePerRotation: =, udiUnitsPerRotation: =, AxisType: =, lrModulo: =, bDone=>, bBusy=>, bError=>, ErrorID=>,)   ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
udiInputRotation	Specify the input of the gearbox.	UDINT	Positive number (0)	When *bExecute* is True and *bBusy* is False.
udiOutputRotation	Specify the output of the gearbox.	UDINT	Positive number (0)	When *bExecute* is True and *bBusy* is False.
udiPulsePerRotation	Specify how many pulses per rotation of the input end of the gearbox. (Unit: pulses/ rotation)	UDINT	Positive number (0)	When *bExecute* is True and *bBusy* is False.
udiUnitsPerRotation	Specify how many units the terminal actuator travels per rotation of the output end of the gearbox	UDINT	Positive number (0)	When *bExecute* is True and *bBusy* is False.

Name	Function	Data type	Setting value (Default value)	Timing for updating
AxisType	Specify the axis type.	SMC_MOVEMENTTYPE*	0: rotary 1: linear (rotary)	When *bExecute* is True and *bBusy* is False.
lrModulo	Specify how many units per rotation of the rotary axis.	LREAL	Positive number (0)	When *bExecute* is True and *bBusy* is False.

*Note: SMC_MOVEMENTTYPE: Enumeration (ENUM)

- **Outputs**

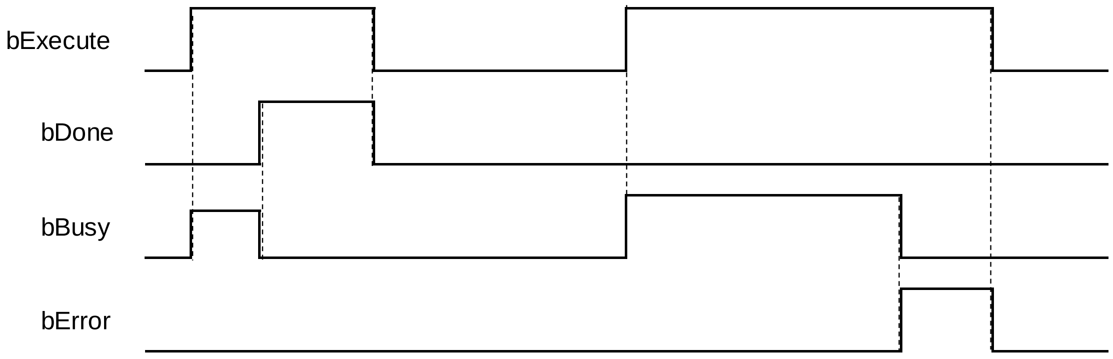
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)

***Note:** DMC_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the instruction is completed.	● When *bExecute* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bExecute* shifts to True.	● When *bExecute* shifts to False. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bExecute* shifts from True to False. (Error code is cleared)
ErrorID		

■ **Timing Diagram**

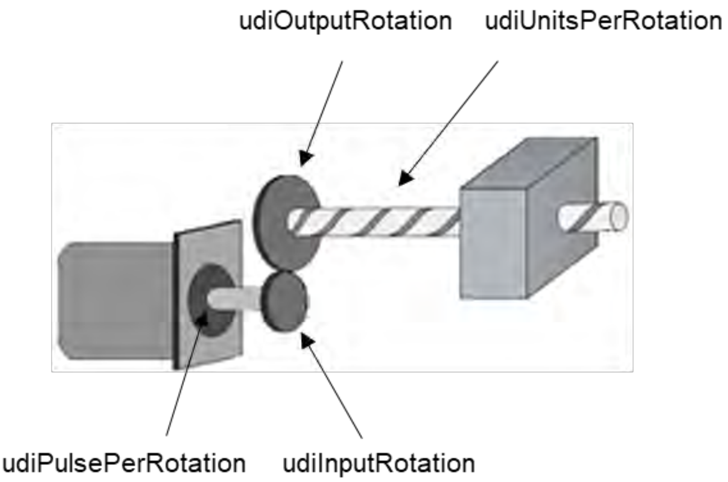


- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_SM3*	AXIS_REF_SM3	When *bExecute* is True and *bBusy* is False.

*Note: AXIS_REF_SM3 (FB): All function blocks contain this variable, which works as the starting program for function blocks.

- **Function**



- DMC_ChangeMechanismGearRation modifies the ratio between user units and pulses, axis type and user units per rotation of the rotary axis.
- The corresponding relationship between the function block inputs and the mechanism is shown in the figure above. *udiPulsePerRotation* is the number of pulses required per rotation of the input end of the gearbox, *udiInputRotation* is the input of the gearbox, *udiOutputRotation* is the output of the gearbox and *udiUnitsPerRotation* is the number of terminal actuator travel units per rotation of the output end of the gearbox.

- **Troubleshooting**

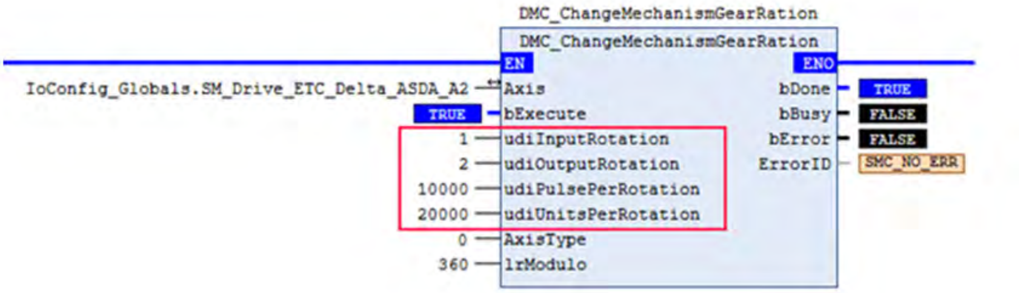
- The instruction can be executed only when the state machine is power_off. Refer to *ErrorID* (Error Code) to address the problem if an error occurs during the instruction execution.

- **Programming Example**

  - The example shows the behavior of DMC_ChangeMechanismGearRation.
  - Relevant parameters setting:

    udiOutputRotation: udiInputRotation= 2: 1
    udiPulsePerRotation: 10000 pulses
    udiUnitsPerRotation: 20000 us
    AxisType: 0 (rotary axis)
    lrModulo: 360

  - **Function block setting**



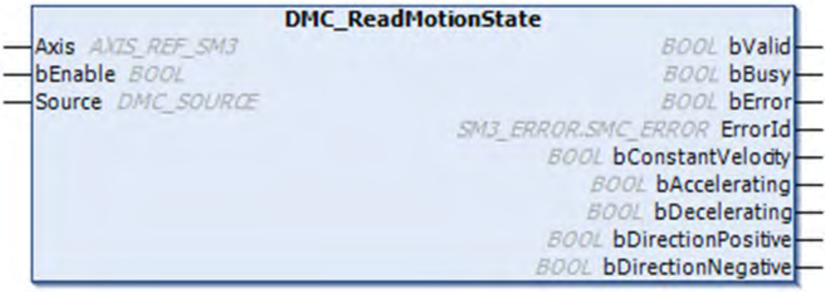
```
                                           DMC_ChangeMechanismGearRation
                                         DMC_ChangeMechanismGearRation
                                         EN                          ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 —Axis              bDone — TRUE
                                    TRUE —bExecute             bBusy — FALSE
                                       1 —udiInputRotation    bError — FALSE
                                       2 —udiOutputRotation  ErrorID — SMC_NO_ERR
                                   10000 —udiPulsePerRotation
                                   20000 —udiUnitsPerRotation
                                       0 —AxisType
                                     360 —lrModulo
```

- **Supported Products**

  - AX-308E

2.3.2.14 DMC_ReadMotionState

DMC_ReadMotionState reads the behavior state of the axis in motion.

FB/FC	Instruction	Graphic expression
FB	DMC_ReadMotionState	 DMC_ReadMotionState Axis AXIS_REF_SM3 bEnable BOOL Source DMC_SOURCE BOOL bValid BOOL bBusy BOOL bError SM3_ERROR.SMC_ERROR ErrorId BOOL bConstantVelocity BOOL bAccelerating BOOL bDecelerating BOOL bDirectionPositive BOOL bDirectionNegative

ST expression

DMC_ReadMotionState_instance (
Axis : =,
bEnable : =,
Source : =,
bValid =>,
bBusy =>,
bError =>,
bConstantVelocity=>,
bAcclerating =>,
bDecelerating =>,
bDirectionPositive =>,
bDirectionNegative =>,)   ;

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is enabled when *Enable* changes from FALSE to TRUE.	BOOL	True/False (False)	-
Source	Select the data source. Commanded: instruction-commanded value. Actual: actual value of the axis.	DMC_ SOURCE*	0: dmcCommandedValue (0)	When *bEnable* shifts to True and *bBusy* is False.

***Note:** MC_SOURCE: Enumeration (ENUM)

- **Outputs**

Name	Function	Data type	Output range (Default value)
bValid	True when the axis stops and the velocity reaches 0.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)

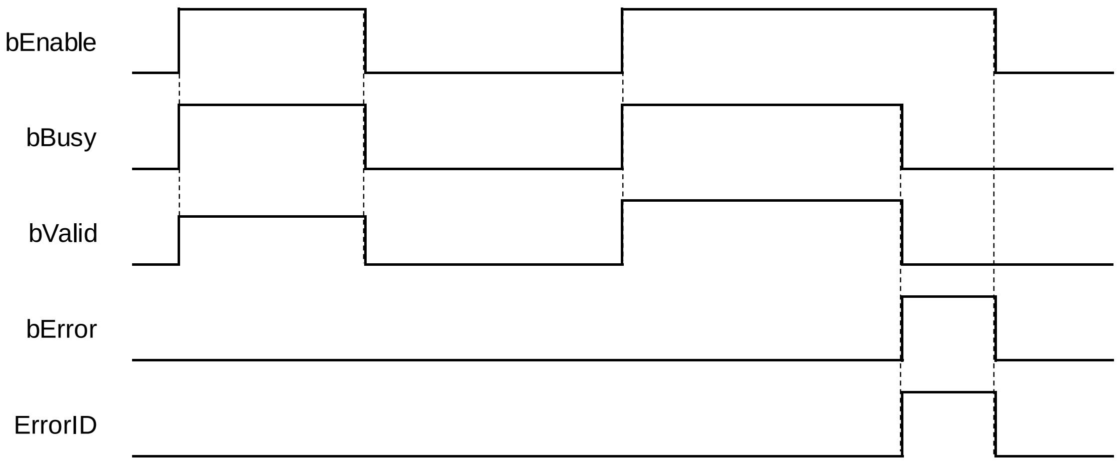
Name	Function	Data type	Output range (Default value)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Record the error code when an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NoError)
bConstantVelocity	Shows that the current velocity is constant.	BOOL	True/False (False)
bAccelerating	Shows that the absolute value of the current velocity is increasing.	BOOL	True/False (False)
bDecelerating	Shows that the absolute value of the current velocity is decreasing.	BOOL	True/False (False)
bDirectionPositive	Shows that the current position is increasing.	BOOL	True/False (False)
bDirectionNegative	Shows that the current position is decreasing.	BOOL	True/False (False)

***Note:** DMC_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* is True and the axis motion state can be read.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* is True.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction. (Error code is recorded).	● When *bEnable* shifts to False. (The error code in *ErrorID* is cleared.)
ErrorID		
bConstantVelocity	● When the current velocity is constant.	● When *bEnable* is True but the velocity is not constant.
bAccelerating	● When the absolute value of the current velocity is increasing.	● When *bEnable* is True but the velocity does not increase.
bDecelerating	● When the absolute value of the current velocity is decreasing.	● When *bEnable* is True but the velocity does not decrease.
bDirectionPositive	● When the current position is increasing.	● When *bEnable* is True and the direction of motion is not positive. ● When *bEnable* is True and the axis does not move any more.
bDirectionNegative	● When the current position is decreasing.	● When *bEnable* is True and the direction of motion is not negative. ● When *bEnable* is True and the axis does not move any more.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_SM3*	AXIS_REF_SM3	When *bEnable* is True and *bBusy* is False.

***Note:** AXIS_REF_SM3 (FB): All function blocks contain this variable, which works as the starting program for function blocks.

- **Function**

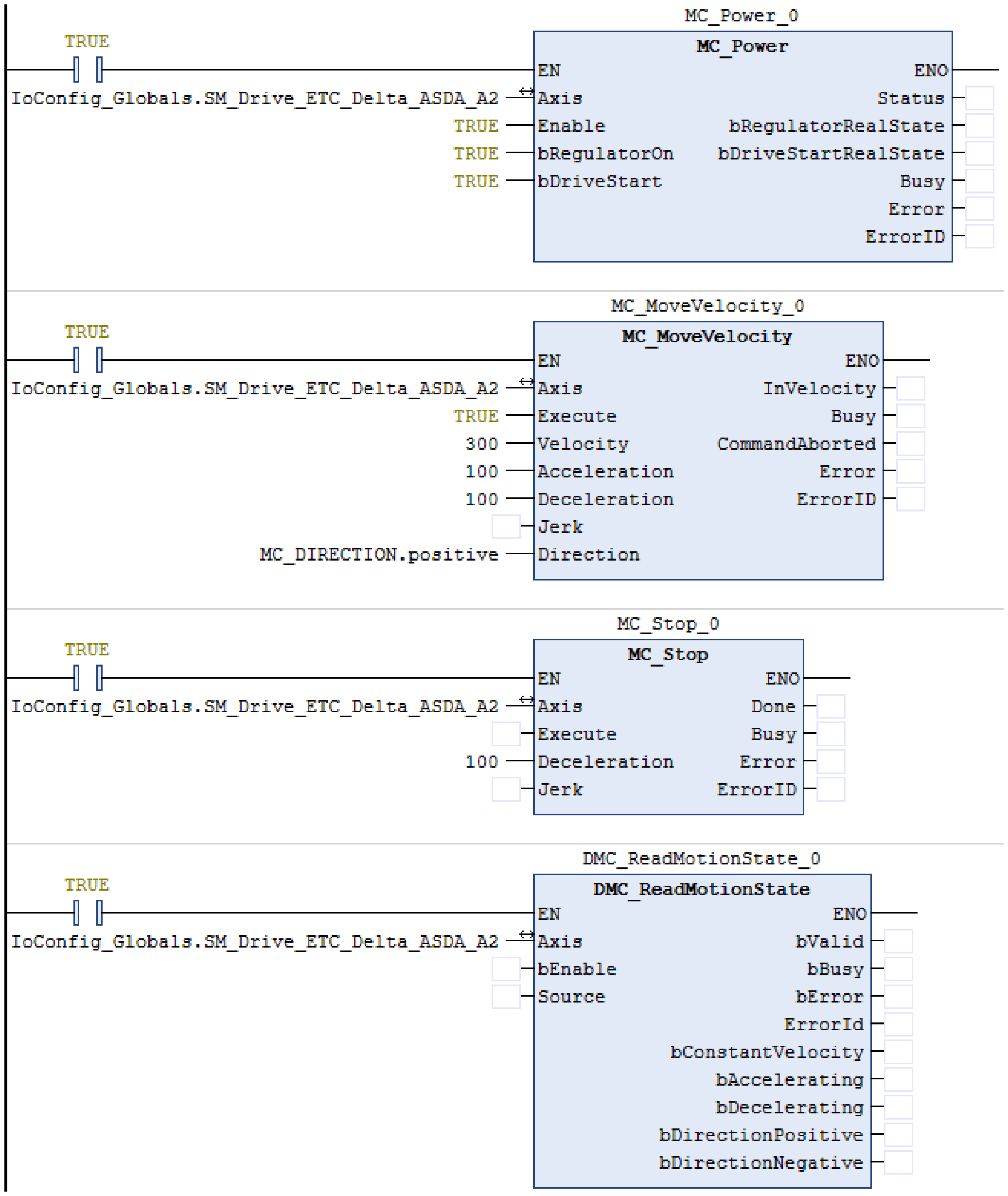
  - DMC_ReadMotionState reads the behavior state of the axis in motion (i.e. acceleration/deceleration, constant velocity, positive/negative direction of motion).
  - When the velocity is 0, the output *bConstantVelocity* changes to True.

- **Troubleshooting**

  - When an error occurs during the instruction execution, *bError* will change to True. Refer to *ErrorID* (Error Code) to address the problem.

- **Programming Example**

  - The example shows the motion behavior that MC_ReadMotionState reads MC_MoveVelocity and MC_Stop.



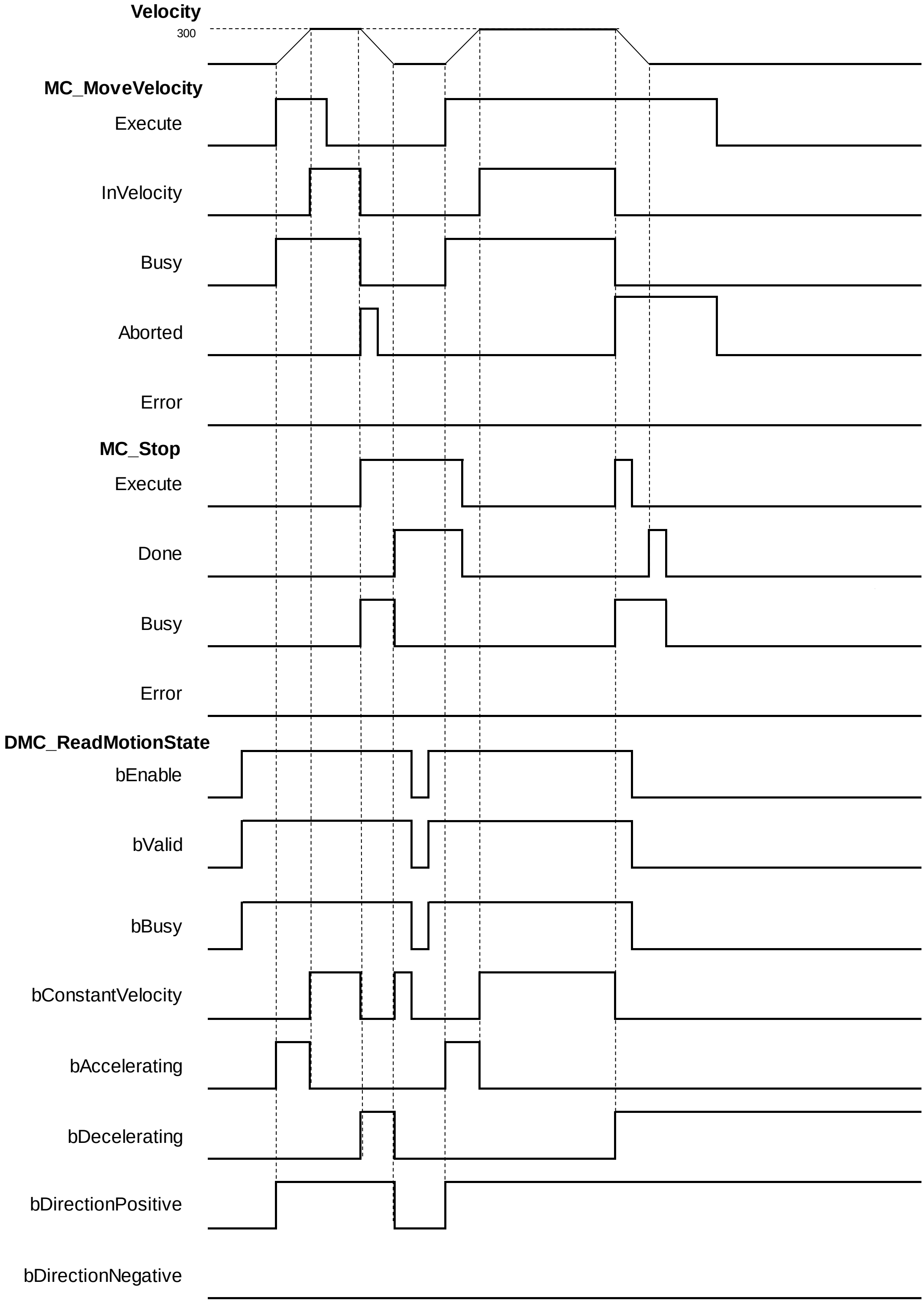
MC_Power_0
MC_Power
TRUE — EN ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 — Axis Status
TRUE — Enable bRegulatorRealState
TRUE — bRegulatorOn bDriveStartRealState
TRUE — bDriveStart Busy
Error
ErrorID

MC_MoveVelocity_0
MC_MoveVelocity
TRUE — EN ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 — Axis InVelocity
TRUE — Execute Busy
300 — Velocity CommandAborted
100 — Acceleration Error
100 — Deceleration ErrorID
Jerk
MC_DIRECTION.positive — Direction

MC_Stop_0
MC_Stop
TRUE — EN ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 — Axis Done
Execute Busy
100 — Deceleration Error
Jerk ErrorID

DMC_ReadMotionState_0
DMC_ReadMotionState
TRUE — EN ENO
IoConfig_Globals.SM_Drive_ETC_Delta_ASDA_A2 — Axis bValid
bEnable bBusy
Source bError
ErrorId
bConstantVelocity
bAccelerating
bDecelerating
bDirectionPositive
bDirectionNegative

- **Timing Diagram**



1. When *bvalid* and *bBusy* change to True after DMC_ReadMotionState is started, it indicates that the motion state can be read.

2. The axis starts to accelerate until the target velocity after MC_MoveVelocity is started. When

bAccelerating and *bDirectionPositive* change to True, it indicates that the axis is accelerating in the positive direction.

3. When the axis reaches the specified target velocity, the instruction maintains a constant velocity, *bAccelerating* changes to False and *bConstantVelocity* changes to True.

4. When MC_Stop is started, MC_MoveVelocity is interrupted and the axis starts to decelerate to a stop. Then *bConstantVelocity* changes to False and *bDecelerating* changes to True.

5. When the velocity of the axis reaches 0, *bDecelerating* and *bDirectionPositive* change to False and *bConstantVelocity* changes to True.

6. When DMC_ReadMotionState is disabled during the deceleration of the axis in the next motion cycle, both *bDecelerating* and *bDirectionPositive* will remain True and will not update any longer no matter how motion instructions work.

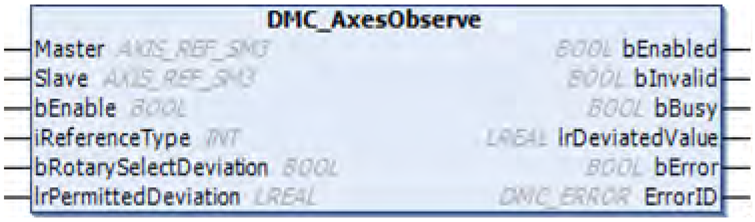
- **Supported Products**

  - AX-308E

2.3.2.15

2.3.2.15 DMC_AxesObserve

DMC_AxesObserve monitors the deviation between the master axis position and slave axis position and it will output a reminder when the deviation exceeds the allowed setting value.

FB/FC	Instruction	Graphic expression
FB	DMC_AxesObserve	 DMC_AxesObserve Master AXIS_REF_SM3 Slave AXIS_REF_SM3 bEnable BOOL iReferenceType INT bRotarySelectDeviation BOOL lrPermittedDeviation LREAL BOOL bEnabled BOOL bInvalid BOOL bBusy LREAL lrDeviatedValue BOOL bError DMC_ERROR ErrorID

ST expression

DMC_AxesObserve_instance (
Master :=,
Slave :=,
bEnable :=,
iReferenceType :=,
bRotarySelectDeviation:=,
lrPermittedDeviation :=,
bEnabled =>,
bInvalid=>,
bBusy =>,
lrDeviatedValue=>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is executed when *bEable* changes from False to True.	BOOL	True/False (False)	-
iReferenceType	Specify the position type.	INT	0: Command position 1: Actual position (0)	When *bEnable* is True.
bRotarySelect Deviation	Specify the shorter or longer distance between axes. The parameter is valid only when the master axis and slave axis are both rotary axes.	BOOL	True/False (False)	When *bEnable* is True.
lrPermitted Deviation	Specify the permitted deviation between the two axes.	LREAL	Positive number or 0(0)	When *bEnable* is True.

- **Outputs**

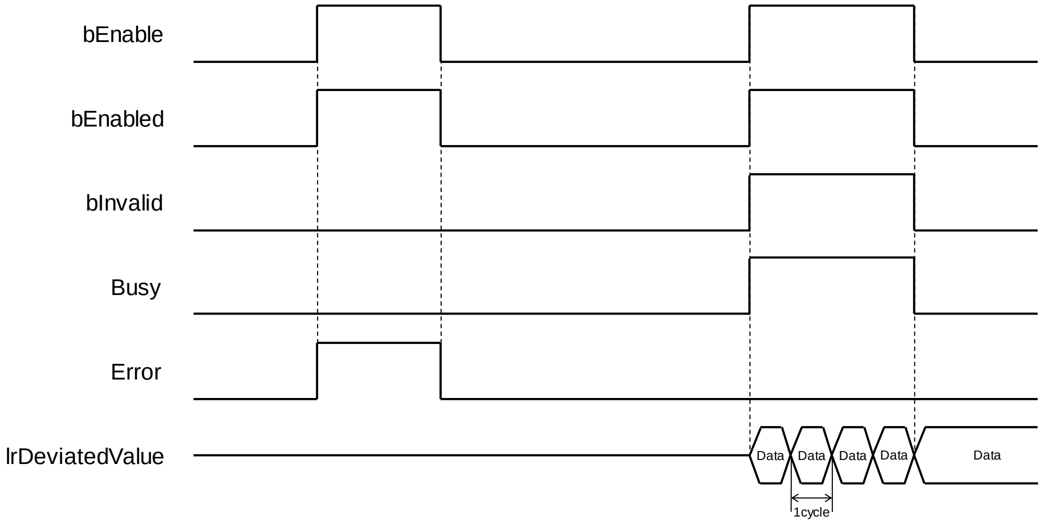
Name	Function	Data type	Output range (Default value)
bEnabled	True when the instruction outputs are valid.	BOOL	True/False (False)
bInvalid	True when the difference between the two axes exceeds the allowed value.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
lrDeviatedValue	The error value between the two axes	LREAL	Positive number, negative number or 0 (0)
bError	True when an error occurs during instruction execution.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DMC_ERROR*	DMC_ERROR (DMC_NO_ERR)

***Note:** DMC_ERROR: Enumeration (Enum)

■ **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bEnabled	● When *bEnable* shifts to True.	● When *bError* shifts to True. ● When *bEnable* shifts to False.
bInvalid	● When *bEnable* shifts to True. ● When the difference between the two axes exceeds the allowed value.	● When *bError* shifts to True. ● When *bEnable* shifts to False.
bBusy	● When *bEnable* shifts to True.	● When *bError* shifts to True. ● When *bEnable* shifts to False.
lrDeviatedValue	● When *bEnable* shifts to True.	● When *bEnable* shifts to False, the data update stops.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* shifts to False.
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Master	Specify the master axis.	AXIS_REF_SM3	AXIS_REF_SM3*	When *bEnable* shifts to True.
Slave	Specify the slave axis.	AXIS_REF_SM3	AXIS_REF_SM3*	

***Note:**
AXIS_REF_SM3 (FB): The interface is built in every function block and works as the starting program of the function block.

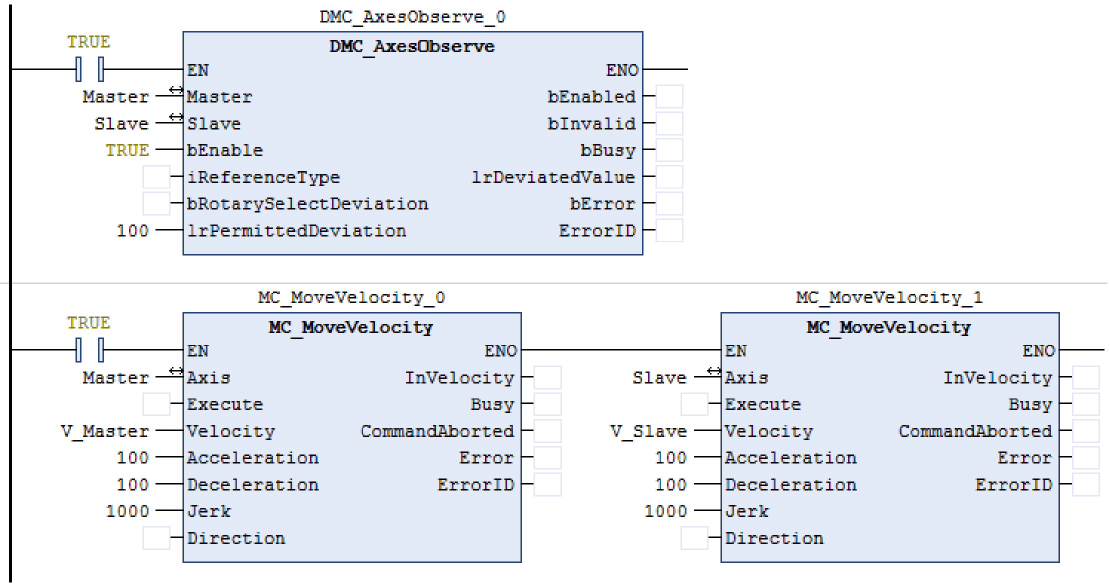
- **Function**
  - DMC_AxesObserve checks if the deviation between the master axis position and slave axis position exceeds the allowed setting value.

    When │lrDeviatedValue│is > lrPermittedDeviation, *bInvalid* changes to True.
  - It is suggested that the same mode should be set for the master and slave axes. If one is set as a linear axis and the other is set as a rotary axis, the calculation of the error between axes will be done in linear axis mode.
  - When both the master axis and slave axis are rotary axes but their distances per rotation are different, the calculation of *lrDeviatedValue* (error between axes) is done in linear axis mode.
  - *bRotarySelectDeviation* is valid only when the distances per rotation for the master axis and slave axis are the same. False means to read the shorter direction and True means to read the longer direction.
  - Calculation of *lrDeviatedValue*

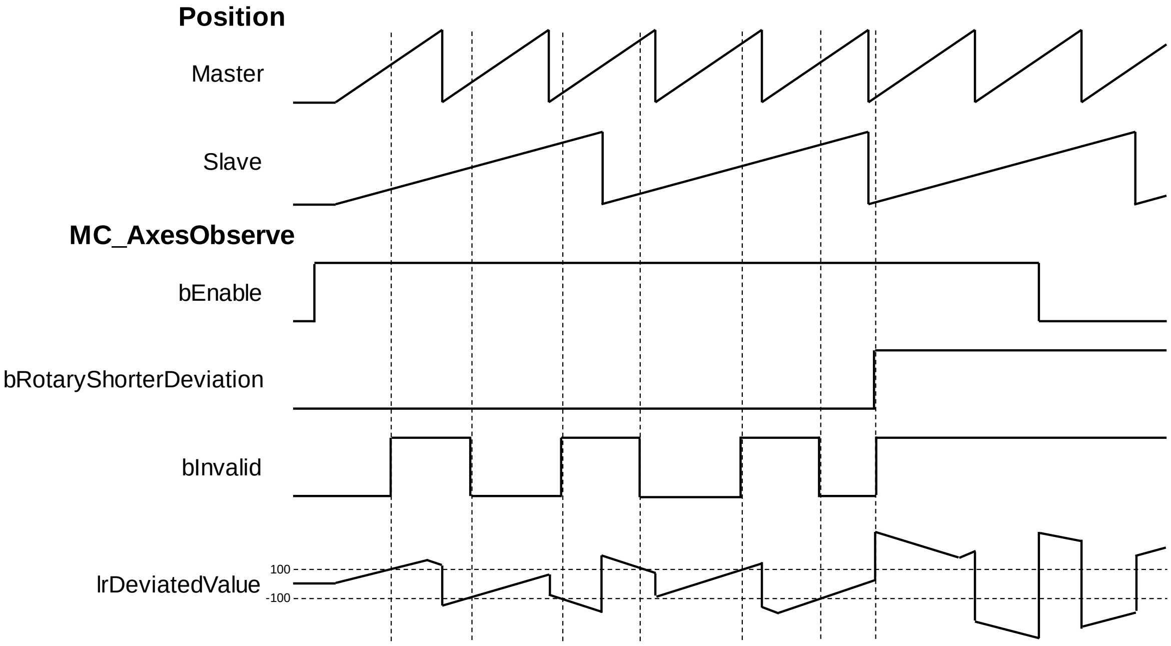
Master axis mode	Slave axis mode	Calculation method
Linear axis	Linear axis	*lrDeviatedValue* (Error between axes) = Master axis position – Slave axis position
Rotary axis	Linear axis	
Linear axis	Rotary axis	
Rotary axis	Rotary axis	*lrDeviatedValue* (Error between axes) = Master axis position – Slave axis position *lrDeviatedValue* (Error between axes) = Distance per rotation– (Master axis position – Slave axis position) *lrDeviatedValue* outputs the value of the longer or shorter distance according to the setting of *bRotarySelectDeviation.* When the current positon of the master axis > the current positon of the slave axis, the sign of *lrDeviatedValue* (Error between axes) is positive (+). When the current positon of the master axis < he current positon of the slave axis, the sign of *lrDeviatedValue* (Error between axes) is negative (-).

- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, bError changes to True and the axis stops running. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - The example illustrates the behavior performed by DMC_AxesObserve when both the master and slave axes are rotary axes with the cycle of 360.



- Timing Diagram

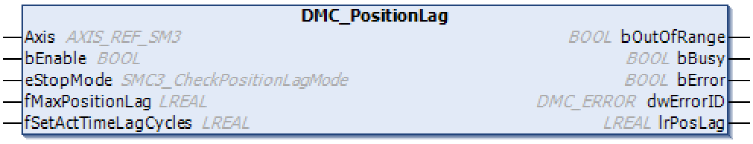


1. When the master and slave axes operate together, the master axis is faster than the slave axis in velocity. Therefore, the starting master axis position is greater than the slave axis position, and the value of *lrDeviatedValue* is positive. When the *lrDeviatedValue* value is greater than the setting value of *lrPermittedDeviation* 100, *bInvalid* changes to True.
2. When the master axis rotates one circle, the master axis returns to 0 behind the slave axis in position, then the slave axis position is negative.
3. When *bRotarySelectDeviation* changes to True, it means that *lrDeviatedValue* selects the longer distance between axes and the value must exceed 180 based on the calculation of *lrDeviatedValue* value mentioned in Function section. Since the *lrPermittedDeviation* is set to 100, *bInvalid* must be True.

- **Supported Products**
  - AX-308E

2.3.2.16 DMC_PositionLag

DMC_PositionLag sets the allowed range of lag error and observe whether the allowed position lag is exceeded.

FB/FC	Instruction	Graphic expression
FB	DMC_PositionLag	 DMC_PositionLag Axis AXIS_REF_SM3 bEnable BOOL eStopMode SMC3_CheckPositionLagMode fMaxPositionLag LREAL fSetActTimeLagCycles LREAL BOOL bOutOfRange BOOL bBusy BOOL bError DMC_ERROR dwErrorID LREAL lrPosLag
ST expression		

DMC_PositionLag_instance (
Axis:=,
bEnable :=,
eStopMode :=,
fMaxPositionLag :=,
fSetActTimeLagCycles :=,
bOutOfRange=>,
bBusy =>,
bError =>,
dwErrorID =>,
 lrPosLag =>);		

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is executed when *bEable* changes from False to True.	BOOL	True/False (False)	-
eStopMode	Specify the stop mode for axis motion when the lag error occurs.	SMC3_CheckPositionLag Mode	0: SMC3_PCL_OFF 1: SMC3_PCL_DISABLE 2: SMC3_PCL_HALT 3: SMC3_PCL_ENABLE （SMC3_PCL_OFF）	When *bEnable* and *bBusy* are True.
fMaxPositionLag	Specify the maximum lag error value.	LREAL	Positive number or 0 (0)	When *bEnable* and *bBusy* are True.
fSetActTimeLagCycles	Specify the lag cycle between command value and actual value.	LREAL	Positive number or 0 (3)	When *bEnable* and *bBusy* are True.

- **Outputs**

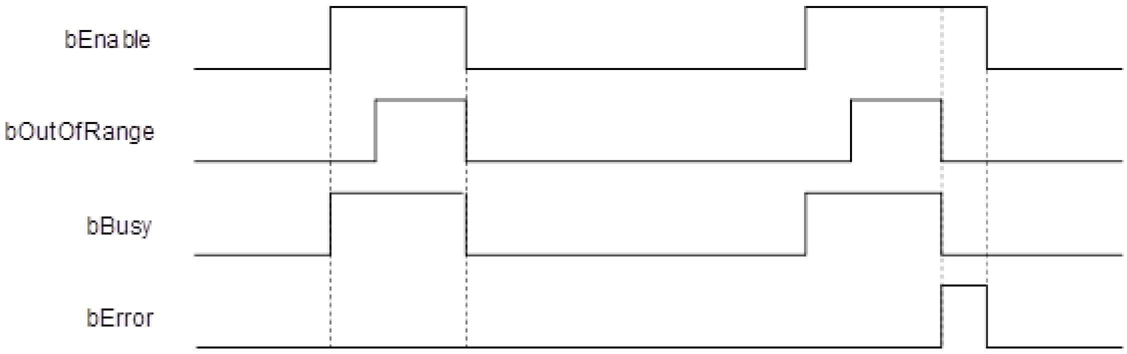
Name	Function	Data type	Output range (Default value)
bOutOfRange	True when LagTime exceeds the setting value.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
dwErrorID	Contains error codes.	DMC_ERROR*	DMC_ERROR (DMC_NO_ERROR)
lrPosLag	Contains current LagError value	LREAL	Positive number or 0 (0)

***Note:** DMC_ERROR: Enumeration (Enum)

- **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bOutOfRange	● True when LagTime exceeds the setting value.	● When *bEnable* shifts to False.
bBusy	● When the instruction is being executed.	● When *bError* shifts to Ture.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* shifts to False. (Error code is cleared.)
dwErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *bExecute* shifts to True.

***Note:** AXIS_REF_SM3 (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - Explanation of *eStopMode*

ENUM	ENUM Name
0	SMC3_PCL_OFF
1	SMC3_PCL_DISABLE
2	SMC3_PCL_HALT
3	SMC3_PCL_ENABLE

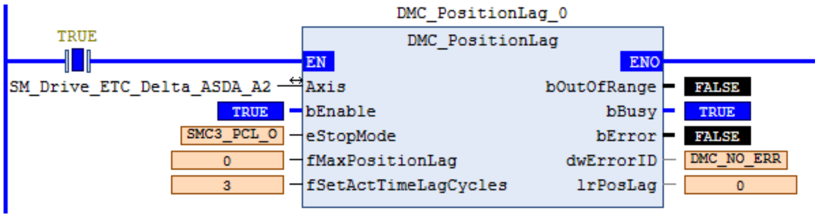
  - ◆ SMC3_PCL_OFF: When LagError is out of the allowed range, the axis is still running.
  - ◆ SMC3_PCL_DISABLE: When LagError is out of the allowed range, the axis parameter *bRegulatorOn* changes to False.
  - ◆ SMC3_PCL_HALT: When LagError is out of the allowed range, the axis parameter *bDriveStart* changes to False.
  - ◆ SMC3_PCL_ENABLE: When LagError is out of the allowed range and the axis stops, there are no change for *bRegulatorOn* and *bDriveStar*t states.

  - Calculation and Judgement of Lag Error

  - ◆ ｜(fActPosition + fSetActTimeLagCycles * [Task cycle time] * fActVelocity – fSetPosition)｜> fMaxPositionLag
  - ◆ *fSetActTimeLagCycles* sets the lag cycle between the command value and actual value. The bigger the difference between the set lag cycle and actual lag cycle, the more likely the lag error will occur due to the error between the command position and actual position.
  - When LagError is out of the range, *bOutOfRange* will change to True and the axis will perform corresponding action based on the setting of *eStopMode*.

- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, bError changes to True and the axis stops running. To confirm current error state, see the error code in ErrorID.

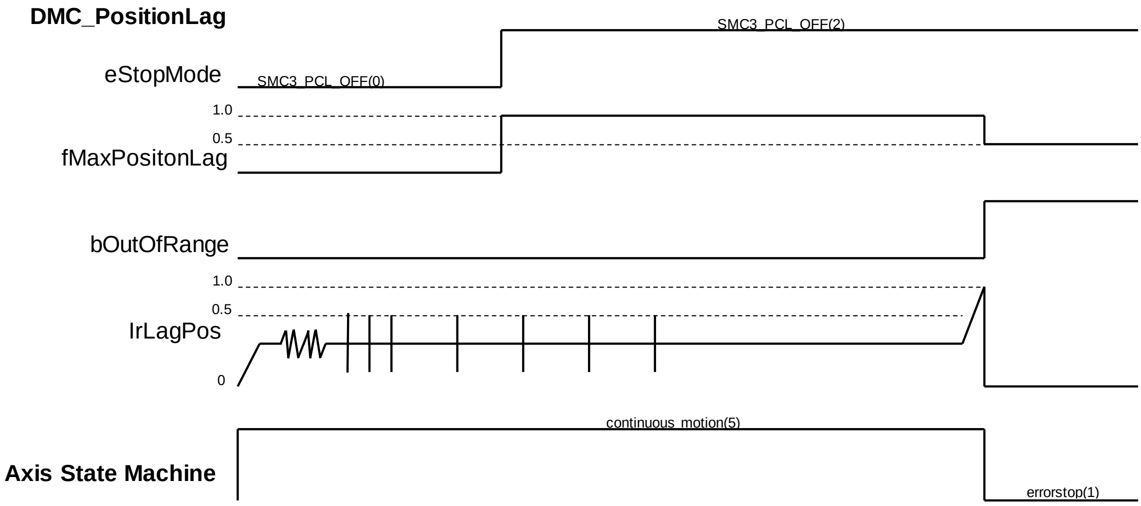
- **Programming Example**
  - The example illustrates how to observe the position lag state and modify the position lag parameters via DMC_PositionLag.



DMC_PositionLag_0
DMC_PositionLag

Input value	Input	Output	Output value
TRUE	EN	ENO	
SM_Drive_ETC_Delta_ASDA_A2	Axis	bOutOfRange	FALSE
TRUE	bEnable	bBusy	TRUE
SMC3_PCL_O	eStopMode	bError	FALSE
0	fMaxPositionLag	dwErrorID	DMC_NO_ERR
3	fSetActTimeLagCycles	lrPosLag	0

  - Timing Diagram

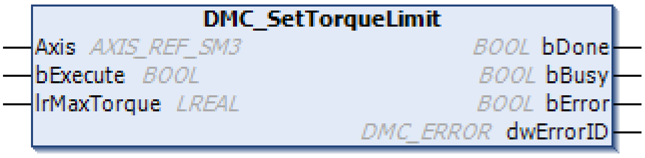


1. Firstly, set *eStopMode* to SMC3_PCL_OFF. The axis starts to run. Then no matter whether the lag error occurs, the axis will not stop running with *bOutOfRange* of the instruction always being False.
2. Then set *eStopMode* to SMC3_PCL_HALT and set *fMaxPositionLag* to a value which is greater than LagError value. Then the position lag value will never be out of the range for the constant-velocity motion.
3. Eventually, adjust *fMaxPositionLag* to a value which is less than LagError value. Then it can be found that the axis stops running and enters ErrorStop state. And *bOutOfRange* of the instruction shifts to True.

- **Supported Products**
  - AX-308E

2.3.2.17 DMC_SetTorqueLimit

DMC_SetTorqueLimit sets the maximum torque of an axis.

FB/FC	Instruction	Graphic expression
FB	DMC_SetTorqueLimit	 DMC_SetTorqueLimit Axis AXIS_REF_SM3 bExecute BOOL lrMaxTorque LREAL BOOL bDone BOOL bBusy BOOL bError DMC_ERROR dwErrorID

ST expression

DMC_SetTorqueLimit_instance (
Axis:=,
bExecute :=,
lrMaxTorque :=,
bDone =>,
bBusy =>,
bCommandAborted =>,
bError =>,
dwErrorID =>);

- **Inputs**

Name	**Function**	**Data type**	**Setting value** **(Default value)**	**Timing for updating**
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
lrMaxTorque	Specify the maximum rated torque. (Unit: Nm)	LREAL	Positive number or 0(0)	When *bExecute* is True and the output *bBusy* is False.

- **Outputs**

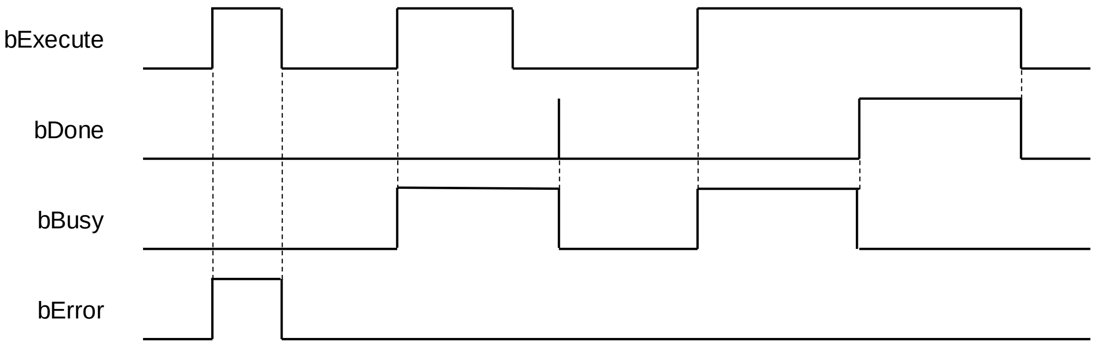
Name	**Function**	**Data type**	**Output range** **(Default value)**
bDone	True when the setting is done.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
dwErrorID	Contains error codes.	DMC_ERROR*[1]	DMC_ERROR(DMC_NO_ERROR)

***Note:** DMC_ERROR: Enumeration (Enum)

- **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bDone	● When the setting is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one period when *bExecute* is False but *bDone* changes to True.
bBusy	● When *bExecute* shifts to True.	● When *bDone* shifts to Ture. ● When *bError* shifts to Ture.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts from True to False. (Error code is cleared.)
dwErrorID		

- **Timing Diagram**



- **In-Outs**

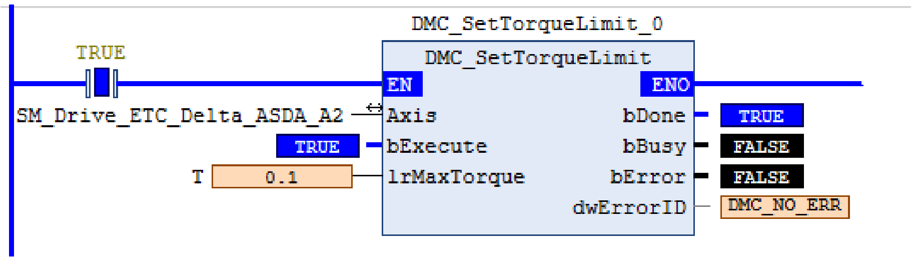
Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *bExecute* shifts to True.

***Note:** AXIS_REF_SM3 (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - After the maximum torque is set, the torque of the motor will be limited to the setting value to protect the motor from damage caused by the excessive torque when large resistance is encountered.
  - DMC_SetTorqueLimit can be used to set the maximum torque of an axis with the unit: Nm.
  - DMC_SetTorqueLimit can be used only in CSP or CSV mode.

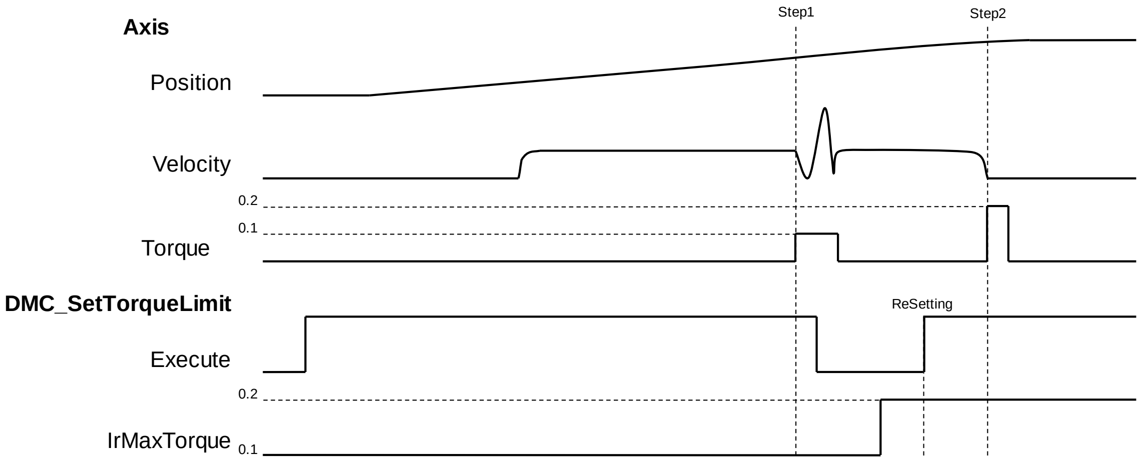
- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, *bError* changes to True and the axis stops running. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - The example illustrates how to use DMC_SetTorqueLimit.



DMC_SetTorqueLimit_0
DMC_SetTorqueLimit
TRUE
EN ENO
SM_Drive_ETC_Delta_ASDA_A2 Axis bDone TRUE
TRUE bExecute bBusy FALSE
T 0.1 lrMaxTorque bError FALSE
dwErrorID DMC_NO_ERR

  - Timing Diagram

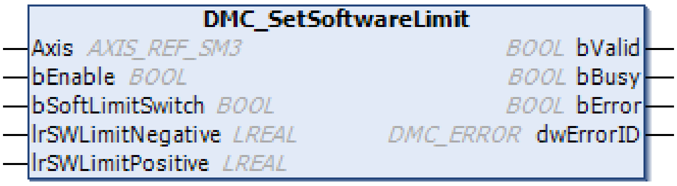


1. Set the maximum rated torque of the axis to 0.1Nm before the operation. Then the operation is performed at a constant velocity.
2. Use the external force to make the axis stop (Step 1) during operation. It can be found that the actual torque of the axis reaches 0.1Nm. Then remove the external force.
3. Set the maximum rated torque to 0.2 Nm and use the external force to make the axis stop (Step2). It is found that the actual torque of the axis reaches 0.2 Nm at the moment. Finally, the Following Error is generated and Servo ON is disabled.

- **Supported Products**
  - AX-308E

2.3.2.18 DMC_SetSoftwareLimit

DMC_SetSoftwareLimit is used to enable, disable and set the upper and lower software limits

FB/FC	Instruction	Graphic expression
FB	DMC_SetSoftwareLimit	 DMC_SetSoftwareLimit Axis AXIS_REF_SM3 bEnable BOOL bSoftLimitSwitch BOOL lrSWLimitNegative LREAL lrSWLimitPositive LREAL BOOL bValid BOOL bBusy BOOL bError DMC_ERROR dwErrorID

ST expression

DMC_SetSoftwareLimit_instance (
Axis:=,
bEnable :=,
bSoftLimitSwitch :=,
lrSWLimitNegative :=,
lrSWLimitPositive :=,
bValid =>,
bBusy =>,
bError =>,
dwErrorID =>);

- **Inputs**

Name	**Function**	**Data type**	**Setting value** **(Default value)**	**Timing for updating**
bEnable	The instruction is executed when *bEable* changes from False to True.	BOOL	True/False (False)	-
bSoftLimitSwitch	Enables or disables software limits.	BOOL	True/False (False)	When *bEnable* and *bBusy* are True.
lrSWLimitNegative	Negative software limit (User unit)	LREAL	Positive number, negative number or 0	When *bEnable* and *bBusy* are True.
lrSWLimitPositive	Positive software limit (User unit)	LREAL	Positive number, negative number or 0	When *bEnable* and *bBusy* are True.

- **Outputs**

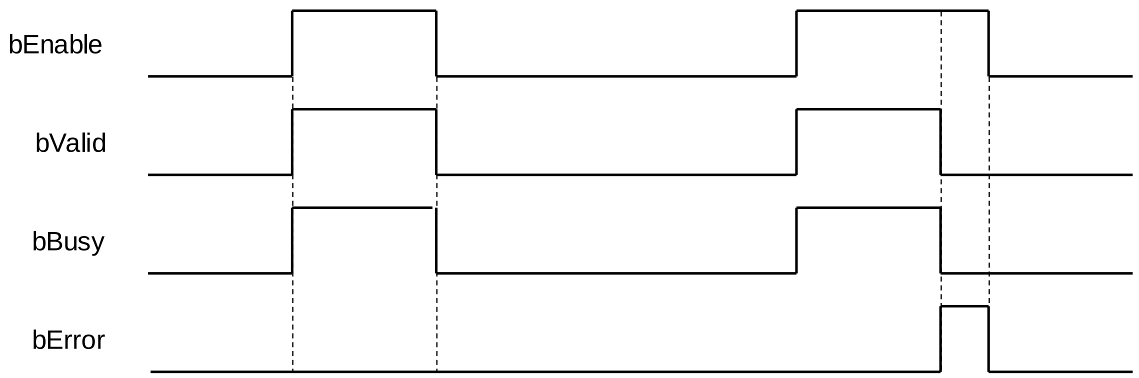
Name	Function	Data type	Output range (Default value)
bValid	True when the control over software limit parameters is valid.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
dwErrorID	Contains error codes.	DMC_ERROR*	DMC_ERROR(DMC_NO_ERROR)

***Note:** DMC_ERROR: Enumeration (Enum)

- ■ **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bValid	● When the control over software limit parameters is valid.	● When *bEnable* shifts to False. ● When *bError* shifts to Ture.
bBusy	● When *bEnable* shifts to True.	● When *bEnable* shifts to False. ● When *bError* shifts to Ture.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* is False. (Error code is cleared.)
dwErrorID		

- **Timing Diagram**



- **In-Outs**

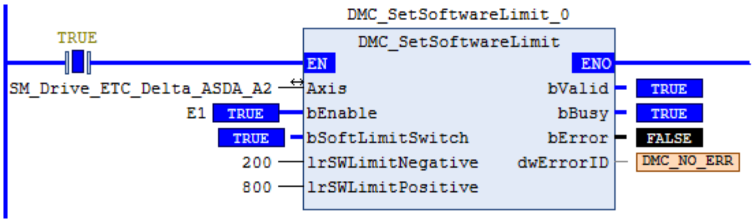
Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis.	AXIS_REF_SM3*	AXIS_REF_SM3	When *bEnable* shifts to True.

***Note:** AXIS_REF_SM3 (FB): The interface is built in every function block and works as the starting program of the function block.

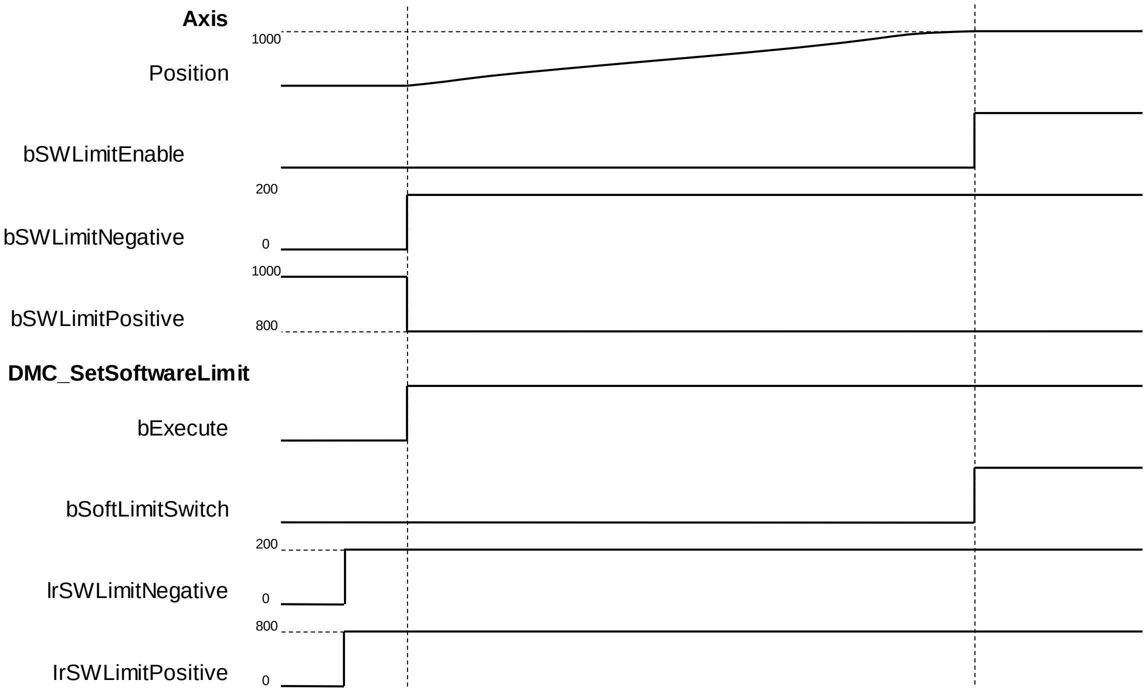
- **Function**
  - After DMC_SetSoftwareLimit is executed, the writing and monitoring of *bSoftLimitSwitch*, fSWLimitNegative and *fSWLimitPositive* axis parameters will be conducted according to the settings on *bSoftLimitSwitch*, *lrSWLimitNegative* and *lrSWLimitPositive* of the instruction.

- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, *bError* changes to True and the axis stops running. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - The example illustrates how to use DMC_SetSoftwareLimit to set the software limits.



  - Timing Diagram

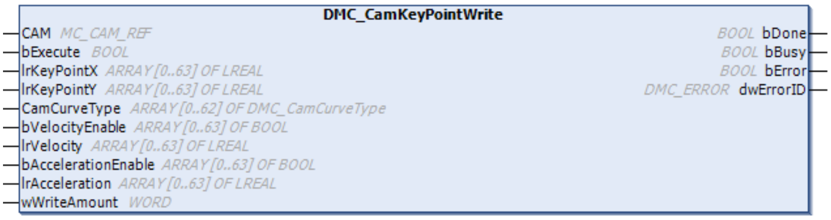


1. After DMC_SetSoftwareLimit is started, the axis parameters writing is conducted based on the set input parameters of the instruction.
2. When *bSoftLimitSwitch* is True, the axis stopping starts as the axis position is outside the software limits.

- **Supported Products**
  - AX-308E

2.3.2.19 DMC_CamKeyPointWrite

DMC_CamKeyPointWrite writes key cam points by selecting a curve type and generating corresponding cam curve based on related parameters. After the new cam curve is generated, the selected cam table will be changed accordingly.

FB/FC	Instruction	Graphic expression
FB	DMC_CamKeyPointWrite	

DMC_CamKeyPointWrite
CAM MC_CAM_REF
bExecute BOOL
lrKeyPointX ARRAY [0..63] OF LREAL
lrKeyPointY ARRAY [0..63] OF LREAL
CamCurveType ARRAY [0..62] OF DMC_CamCurveType
bVelocityEnable ARRAY [0..63] OF BOOL
lrVelocity ARRAY [0..63] OF LREAL
bAccelerationEnable ARRAY [0..63] OF BOOL
lrAcceleration ARRAY [0..63] OF LREAL
wWriteAmount WORD
BOOL bDone
BOOL bBusy
BOOL bError
DMC_ERROR dwErrorID

ST expression

```
DMC_CamKeyPointWrite_instance (
CAM :=,
bExecute :=,
lrKeyPointX :=,
lrKeyPointY :=,
CamCurveType :=,
bVelocityEnable :=,
lrVelocity :=,
bAccelerationEnable :=,
lrAcceleration :=,
wWriteAmount :=,
bDone =>,
bBusy =>,
bError =>,
 dwErrorID =>);
```

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bExecute	The instruction is executed when *bExecute* changes from False to True.	BOOL	True/False (False)	-
lrKeyPointX	The master axis positions of key points which are set (Unit: user unit)	LREAL [0..63]	Negative number, Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
lrKeyPointY	The slave axis positions of key points which are set. (Unit: user unit)	LREAL [0..63]	Negative number, Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
CamCurve Type	Select types of cam curves between key cam points.	DMC_ Cam Curve Type [0..62]*	0: Line 1: Quadratic_Parabola 2: Poly5 3: Basic_Sine 4: Inclined_Sine	When *bExecute* is True and the output *bBusy* is False.

Name	Function	Data type	Setting value (Default value)	Timing for updating
			5: Mod_Acc_Sine 6: Mod_Acc_Trapezoidal 7: Cubic_Spline_Nature 8: Cubic_Spline_Clamp 9: Cubic_Spline (Line)	
bVelocity Enable	Enable or disable velocity settings of key points.	BOOL [0..63]	Negative number, Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
lrVelocity	Velocities of key cam points	LREAL [0..63]	Negative number, Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
bAccelerationEnable	Enable or disable acceleration settings of key points	BOOL [0..63]	Negative number, Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
lrAcceleration	Acceleration rates of key cam points	LREAL [0..63]	Negative number, Positive number or 0 (0)	When *bExecute* is True and the output *bBusy* is False.
wWriteAmount	The amount of key cam points which are set	WORD	2~64 (2)	When *bExecute* is True and the output *bBusy* is False.

***Note**: DMC_CamCurveType: Enumeration (ENUM)

- **Outputs**

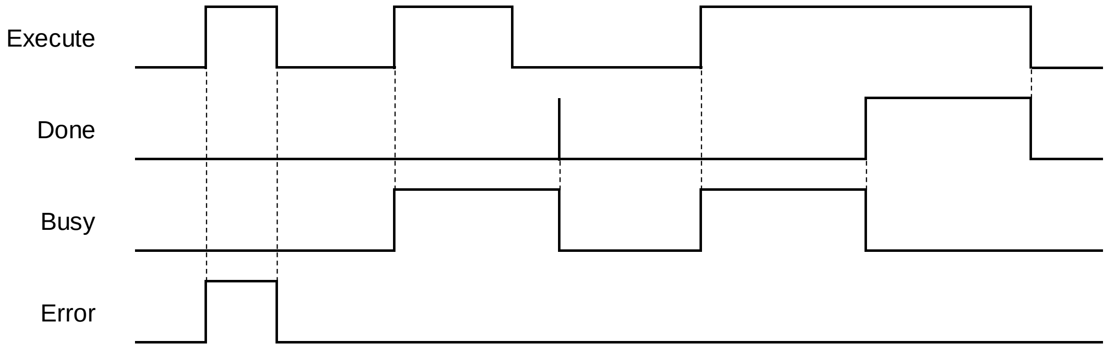
Name	Function	Data type	Output range (Default value)
bDone	True when the instruction is complete.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
dwErrorID	Contains error codes.	DMC_ERROR*	DMC_ERROR(DMC_NO_ERROR)

***Note**: DMC_ERROR: Enumeration (Enum)

- **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bDone	● When the instruction is completed.	● When *bExecute* shifts to False. ● *bDone* will change to False after remaining True for one period when *bExecute* is False but *bDone* changes to True.
bBusy	● When *bExecute* shifts to True.	● When *bDone* shifts to True. ● When *bError* shifts to Ture.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts from True to False. (Error code is cleared.)
dwErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
CamTable	Specify a cam table	MC_CAM_REF*	MC_CAM_REF	When *bExecute* shifts to True.

***Note:** MC_CAM_REF (FB): The basic CAM.

- **Function**

- This function block may take a long time to perform the calculation of curves and cam points. Therefore, it is suggested that this function block should be used in a non-EtherCAT Task in order to avoid the issue of Lost Sync in EtherCAT Task when DMC_CamKeyPointWrite is executed.

- CamCurveType

CamCurveType	Description
Line (0)	Used in the situation where the velocity-constant motion is maintained. There will be a large force on the start point and end point of a straight line (the accelerations for the start point and end point of the line segment approaches infinity), which is very obvious in the high speed operation. So the curve type is suitable for use in low-speed operation.
Quadratic_Parabola (1)	Used in the situation where the acceleration must maintain a constant-velocity motion. This type of curve (with non-zero acceleration rates at the start point and end point of the line segment) is more likely to cause shock as well as vibration. So the curve type is suitable for use in lower speed operation.

CamCurveType	Description
Poly5 (2)	Users can set the velocity and acceleration boundaries of the start point and the end point, or automatically continue the velocity and acceleration boundary value of the previous or next segment (via disabling *bVelocityEnable* / *bAccelerationEnable*)
Basic_Sine (2)	Used in the situation where the follower needs to do a simple harmonic motion. This curve is a cosine curve in the acceleration diagram. The positive maximum acceleration rate and negative maximum acceleration rate are at the start position and the end position respectively and it is zero at the middle point. So the Jerk is infinite at the start position and end position, which is prone to shock and vibration. So the curve type is suitable for applications in the medium and low speed operation.
Inclined_Sine (3)	Used in the situation where the follower needs to perform a cycloid motion. This curve is a sine curve in the acceleration diagram, and the acceleration at the start position and the end position is zero, so the jump produced will not cause the acceleration to reach infinity. And thus the curve type can be applied for high-speed operation due to smooth operation.
Mod_Acc_Sine (4)	The acceleration graph of the curve is a sine curve changed from a typical ladder graph, so that the acceleration is smoother. The curve type is applied for high-speed operation.
Mod_Acc_Trapezoidal (5)	The acceleration graph of the curve is a sine curve changed from oblique straight lines for the acceleration and deceleration segments of a typical ladder diagram. So the acceleration has better smoothness. The curve type is applied for high-speed operation.
Cubic_Spline_Nature (6)	The acceleration at the start and end points of the cubic curve is zero. That is, there is no force on both ends of the follower.
Cubic_Spline_Clamp (7)	The velocities for the start and end points of the cubic curve are user-set values. The acceleration rates for both ends are the positive maximum and negative maximum, so shock and vibration are likely to occur.
Cubic_Spline (8)	The cubic curve is used when four or more key points are used as interpolation points in order to link two boundary curves as well as avoid the Runge phenomenon of multi-order curves.

- Velocity Enable / Acceleration Enable

  - Users can enable or disable the velocities and accelerations of key points through the *bVelocityEnable* and *bAccelerationEnable* parameters of the instruction.

  - False means that the user-set velocity or acceleration value is not enabled for curve planning. The boundary condition values of key points will automatically obtain the velocity or acceleration calculated for the previous or next curve segment so as to achieve continuous velocity or acceleration for the intersection of curves. True means that a curve will be produced based on the velocities and accelerations of key points, which are the condition values of *lrVelocity* and *lrAcceleration* set by user.

  - For some of the following curves, the velocity and acceleration of their key points can be specified via *bVelocityEnable* and *bAccelerationEnable.* See details in the following table.

No.	Curve type	VelocityEnable	AccelerationEnable	Velocity	Acceleration
0	Straight line	Not possible *1	Not possible *1	Automatically calculated	0
1	Parabola	Not possible	Not possible	0	Automatically calculated
2	Poly5	Possible	Possible	User can define	User can define
3	Acceleration cosine curve	Not possible	Not possible	0	Automatically calculated
4	Acceleration sine curve	Not possible	Not possible	0	0
5	Modified acceleration sine curve	Possible	Not possible	User can define	0
6	Modified acceleration trapezoidal curve	Not possible	Not possible	0	0
7	Cubic spline curve (nature boundary)*2	Not possible	Not possible	Automatically calculated	0
8	Cubic spline curve (clamp boundary)*2	Possible	Not possible	User can define	Automatically calculated
9	Cubic spline curve*3	Not possible	Not possible	Automatically calculated	Automatically calculated

***Note:**

1. Not possible: the setting value is invalid; Possible: the setting value is valid.
2. The boundary conditions of the cubic spline curve are classified into nature boundary and clamp boundary. The nature boundary means that the acceleration of the spline curve is specified as 0 and the velocity for both ends of the curve cannot be specified. The clamp boundary means the velocity for both ends of the curve can be specified but the acceleration cannot be specified.
3. The Cubic_Spline curve is a curve for connecting two boundaries and the boundary curves at the two ends of the cubic spline curve must be the same as follows.

```
CamCurve_Type[0] := Cubic_Spline_Nature;

CamCurve_Type[1] := Cubic_Spline;

CamCurve_Type[2] := Cubic_Spline_Nature;
```

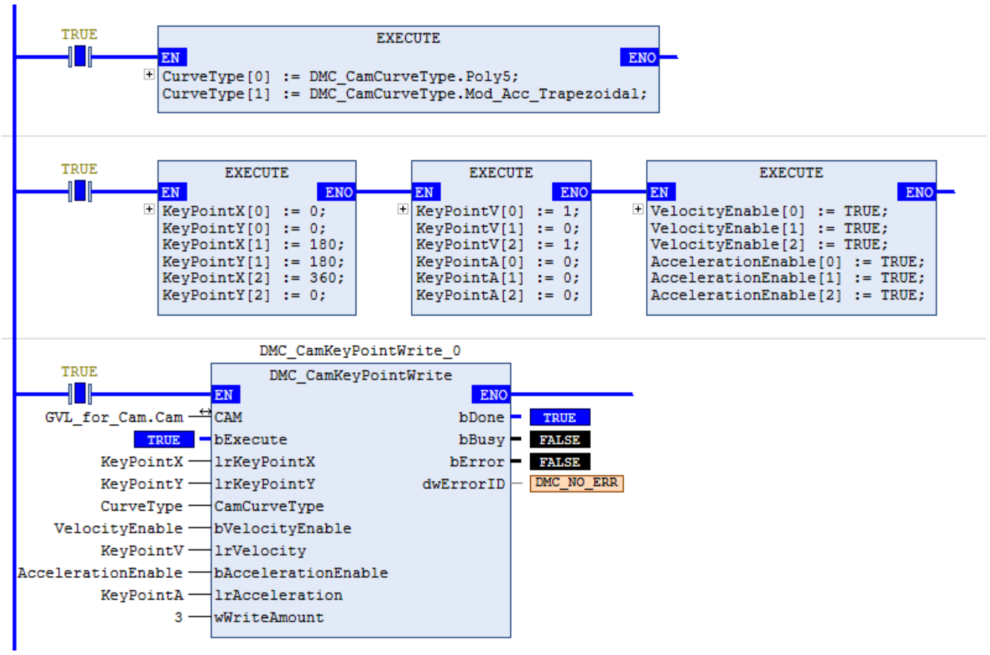
- ■ Key point number specified by *WriteAmount*
  - ◆ The amount of key points specified by *WriteAmount* is up to 64 points, but the key point amount cannot exceed the total resolution of the cam table.
  - ◆ Each key point (except the last point) needs to select a curve type, the resolution between the straight lines is fixed as 1, and the resolution of the remaining curves is averaged by the remaining analytical points; but when there are only straight lines in the entire cam table, then the points of the entire cam table will be divided equally by all straight lines.

- Curve Types

Curve Type	Description
Line (0)	Used in the situation where the velocity-constant motion is maintained. There will be a large force on the start point and end point of a straight line (the accelerations for the start point and end point of the line segment approaches infinity), which is very obvious in the high speed operation. So the curve type is suitable for use in low-speed operation.
Parabola	Used in the situation where the acceleration must maintain a constant-velocity motion. This type of curve (with non-zero acceleration rates at the start point and end point of the line segment) is more likely to cause shock as well as vibration. So the curve type is suitable for use in lower speed operation.
Poly5	Users can set the velocity and acceleration boundaries of the start point and the end point, or automatically continue the velocity and acceleration boundary value of the previous or next segment (via disabling *bVelocityEnable* / *bAccelerationEnable*)
Acceleration cosine curve	Used in the situation where the follower needs to do a simple harmonic motion. This curve is a cosine curve in the acceleration diagram. The positive maximum acceleration rate and negative maximum acceleration rate are at the start position and the end position respectively and it is zero at the middle point. So the Jerk is infinite at the start position and end position, which is prone to shock and vibration. So the curve type is suitable for applications in the medium and low speed operation.
Acceleration sine curve	Used in the situation where the follower needs to perform a cycloid motion. This curve is a sine curve in the acceleration diagram, and the acceleration at the start position and the end position is zero, so the jump produced will not cause the acceleration to reach infinity. And thus the curve type can be applied for high-speed operation due to smooth operation.
Modified acceleration sine curve	The acceleration graph of the curve is a sine curve changed from a typical ladder graph, so that the acceleration is smoother. The curve type is applied for high-speed operation.
Modified acceleration trapezoidal curve	The acceleration graph of the curve is a sine curve changed from oblique straight lines for the acceleration and deceleration segments of a typical ladder diagram. So the acceleration has better smoothness. The curve type is applied for high-speed operation.
Cubic spline curve (nature boundary)	The acceleration at the start and end points of the cubic curve is zero. That is, there is no force on both ends of the follower.
Cubic spline curve (clamp boundary)	The velocities for the start and end points of the cubic curve are user-set values. The acceleration rates for both ends are the positive maximum and negative maximum, so shock and vibration are likely to occur.
Cubic spline curve	The cubic curve is used when four or more key points are used as interpolation points in order to link two boundary curves as well as avoid the Runge phenomenon of multi-order curves.

- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, *bError* changes to True and the axis stops running. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - Programming Example1:
    - The example illustrates the synchronized motion based on the cam table generated from DMC_ DMC_CamKeyPointWrite.



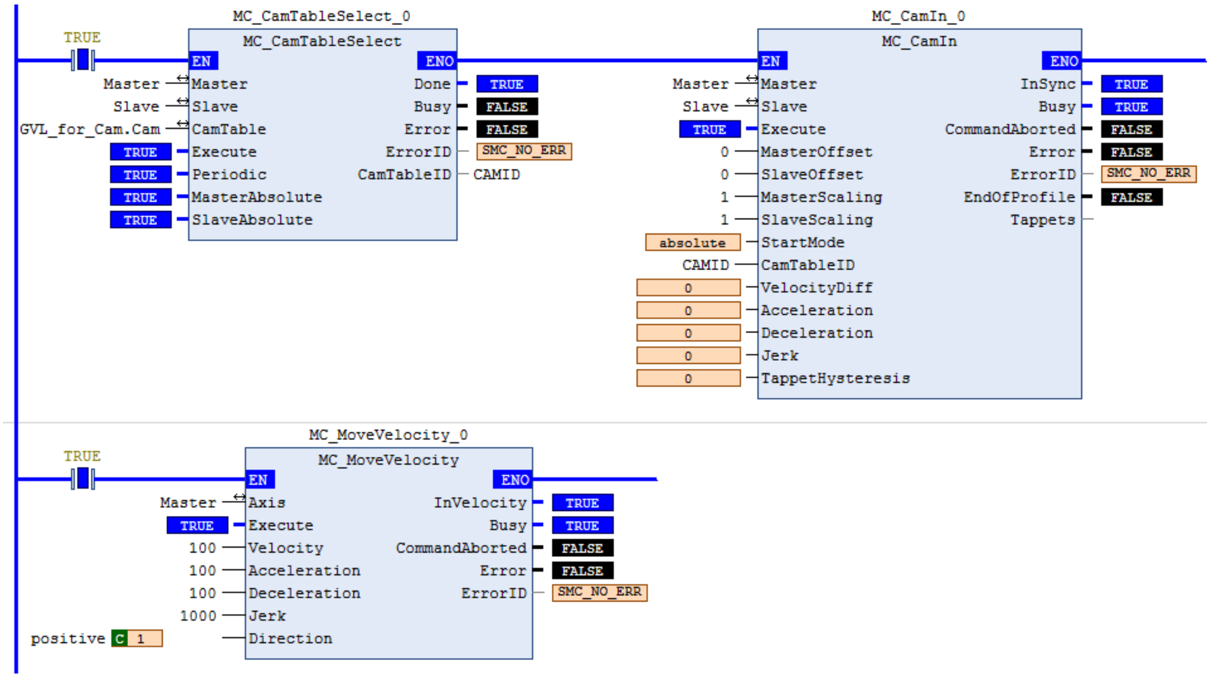
```
CurveType[0] := DMC_CamCurveType.Poly5;
CurveType[1] := DMC_CamCurveType.Mod_Acc_Trapezoidal;
```

```
KeyPointX[0] := 0;
KeyPointY[0] := 0;
KeyPointX[1] := 180;
KeyPointY[1] := 180;
KeyPointX[2] := 360;
KeyPointY[2] := 0;
```

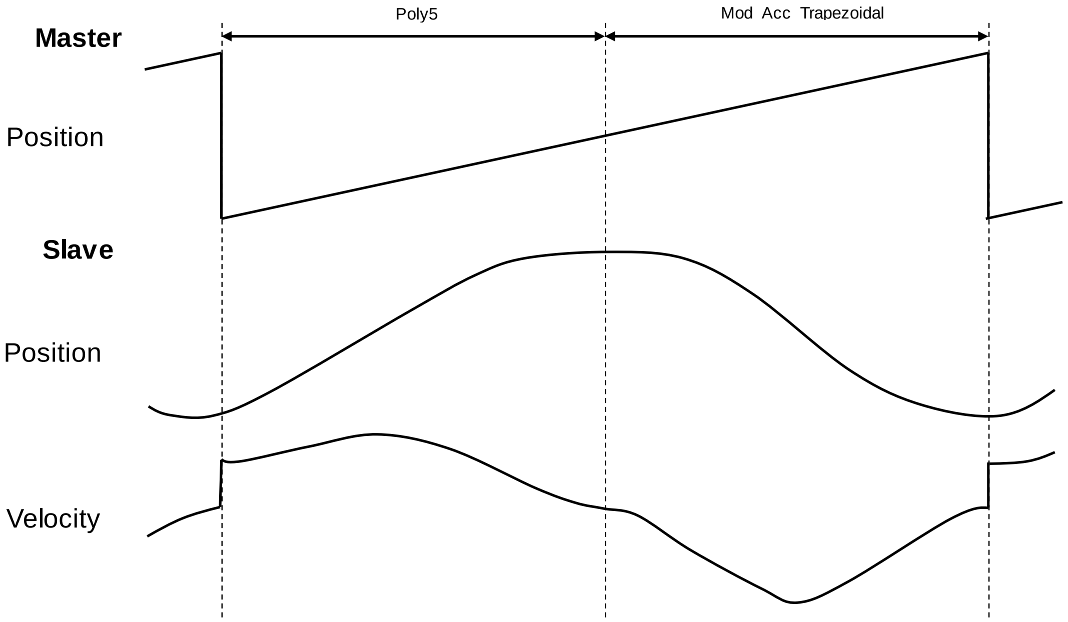
```
KeyPointV[0] := 1;
KeyPointV[1] := 0;
KeyPointV[2] := 1;
KeyPointA[0] := 0;
KeyPointA[1] := 0;
KeyPointA[2] := 0;
```

```
VelocityEnable[0] := TRUE;
VelocityEnable[1] := TRUE;
VelocityEnable[2] := TRUE;
AccelerationEnable[0] := TRUE;
AccelerationEnable[1] := TRUE;
AccelerationEnable[2] := TRUE;
```

◆ The cam table generated from DMC_CamKeyPointWrite can be used by MC_CamTableSelect directly.

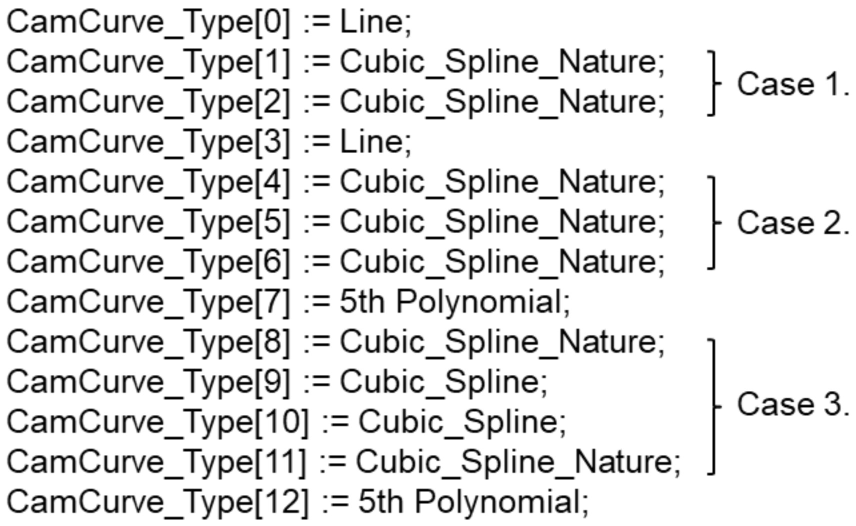


- Timing Diagram



Three key points are used to make up a curve. The first segment of the curve is a 5th Polynomial curve, and the second segment is a Mod_Acc Trapezoidal curve.

- Programming Example2:
  - The example illustrates the special applications of cubic interpolation curves:



```
CamCurve_Type[0] := Line;
CamCurve_Type[1] := Cubic_Spline_Nature;  ┐
                                           ├ Case 1.
CamCurve_Type[2] := Cubic_Spline_Nature;  ┘
CamCurve_Type[3] := Line;
CamCurve_Type[4] := Cubic_Spline_Nature;  ┐
CamCurve_Type[5] := Cubic_Spline_Nature;  ├ Case 2.
CamCurve_Type[6] := Cubic_Spline_Nature;  ┘
CamCurve_Type[7] := 5th Polynomial;
CamCurve_Type[8] := Cubic_Spline_Nature;  ┐
CamCurve_Type[9] := Cubic_Spline;         │
CamCurve_Type[10] := Cubic_Spline;        ├ Case 3.
CamCurve_Type[11] := Cubic_Spline_Nature; ┘
CamCurve_Type[12] := 5th Polynomial;
```

Case 1. If you want to plan a cubic curve with three key points, just select the curve types with the same boundaries.

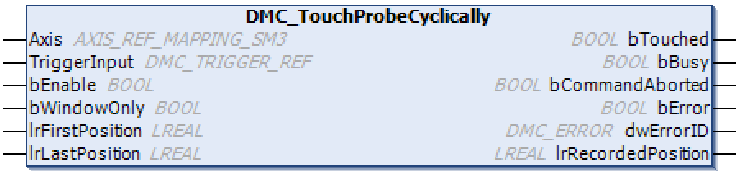
Case 2. If there are three or more boundary condition curves, each two curves will be counted as a segment in the curve planning calculation. If there are no continuous boundary curves, the single curve will be calculated as a boundary condition curve.

Case 3. If there are four or more key points to be on the same curve, you can use spline curves as the continuous line segments of the continuous boundary condition curves at both ends. Then the key points between the two ends will be used as inner interpolation points of the cubic curve.

- **Supported Products**
  - AX-308E

2.3.2.20 DMC_TouchProbeCyclically

DMC_TouchProbeCyclically can continuously record the captured position of an axis.

FB/FC	Instruction	Graphic expression
FB	DMC_TouchProbeCyclically	
ST expression		

DMC_TouchProbeCyclically_instance (
Axis :=,
TriggerInput :=,
bEnable :=,
bWindowOnly :=,
lrFirstPosition :=,
lrLastPosition :=,
bTouched =>,
bBusy =>,
bCommandAborted =>,
bError =>,
dwErrorID =>,
lrRecordedPosition =>);		

- **Inputs**

Name	Function	Data type	Setting value (Default value)	Timing for updating
bEnable	The instruction is executed when *bEnable* changes from False to True.	BOOL	True/False (False)	-
bWindowOnly	Enable the Window range setting.	BOOL	True/False (False)	When *bEnable* and *bBusy* are False.
lrFirstPosition	Defines the start position of Window. (Unit: user unit)	LREAL	Negative number, Positive number or 0 (0)	When *bEnable* and *bBusy* are False.
lrLastPosition	Defines the end position of Window. (Unit: user unit)	LREAL	Negative number, Positive number or 0 (0)	When *bEnable* and *bBusy* are False.

- **Outputs**

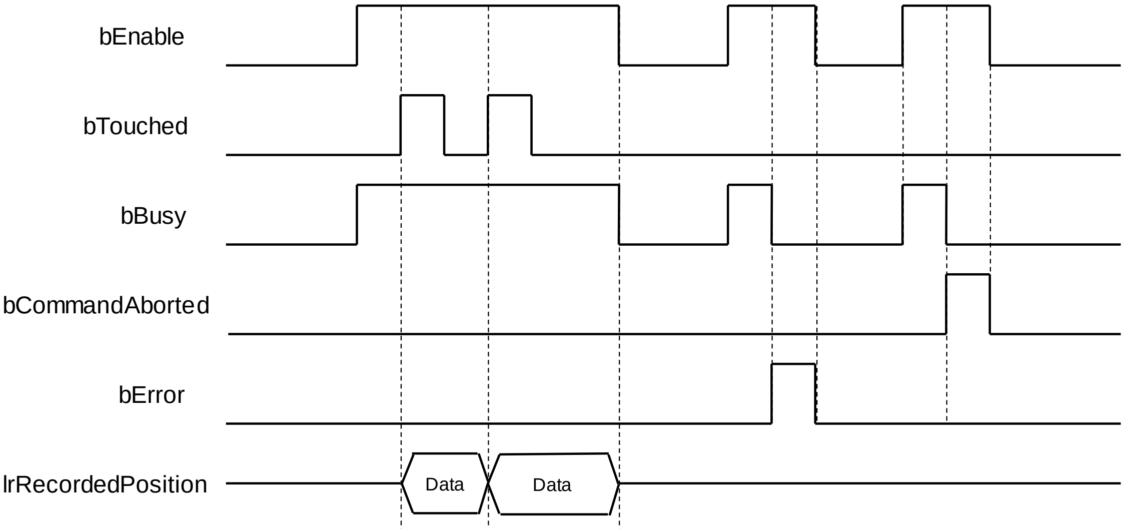
Name	Function	Data type	Output range (Default value)
bTouched	True when the trigger signal is True and axis position recording is completed.	BOOL	True/False (False)
bBusy	True when the instruction is being executed.	BOOL	True/False (False)
bCommand Aborted	True when the instruction is aborted by another instruction.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
dwErrorID	Contains error codes.	DMC_ERROR*1	DMC_ERROR(DMC_NO_ERROR)
lrRecorded Position	Contains the position when a trigger occurs.	LREAL	Positive number, negative number or 0 (0)

***Note:** DMC_ERROR: Enumeration (Enum)

- **Output Update Timing**

Name	Timing for changing to TRUE	Timing for changing to FALSE
bTouched	● When the trigger signal is True and axis position recording is completed.	● When *bEnable* shifts to False. ● When *bCommandAborted* shifts to True. ● When *bError* shifts to True. ● After a period when *bEnable* shifts to True.
bBusy	● True when the instruction execution starts.	● When *bEnable* shifts to False. ● When *bCommandAborted* shifts to True. ● When *bError* shifts to True.
bCommand Aborted	● When the instruction is aborted by another instruction.	● When *bEnable* shifts to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* changes from True to False.
dwErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis.	AXIS_REF_SM3*1	AXIS_REF_SM3 (Must be specified)	-
TriggerInput	Trigger signal	DMC_TRIGGER_REF *2	TRIGGER_REF (-1)	When *bEnable* shifts to True and *bBusy* is False.

***Note:**

1. AXIS_REF_SM3 (FB): The interface is built in every function block and works as the starting program of the function block.
2. DMC_TRIGGER_REF: Structure (STRUCT)

Name	Function	Data type	Setting value (Default)
iTriggerNumber	The trigger channel	INT	0: Touch Probe 1 1: Touch Probe 2 (-1)
eFastLatching	The trigger signal	DMC_LATCH_MODE	0: DRIVE_MODE 1: CONTRL_MODE (DRIVE_MODE)
bInput	The trigger signal source when the controller is triggered	BOOL	The trigger signal source
bActive	Activate or deactivate the trigger signal	BOOL	True: Activate the trigger signal (False)
iCtrlTriggerSource	The recorded position source	INT	0: Set Position 1: Act Position (0)
iCtrlTriggerNumber	The mode of triggering the controller	INT	0: Rising edge data capture 1: Falling edge data capture 2: Rising/falling edge data capture (-1)

- **Function**

  - When the trigger signal (*eFastLatching*) is DRIVE_MODE, then the position is provided by the servo and *iCtrlTriggerSource* is meaningless. *iCtrlTriggerSource* is available only for CONTRL_MODE.

  - When DMC_TouchProbeCyclically is used, the Touch Probe Function (60B8h) cannot be configured to PDO. If users configure it to PDO, the function block will report an error when being executed.

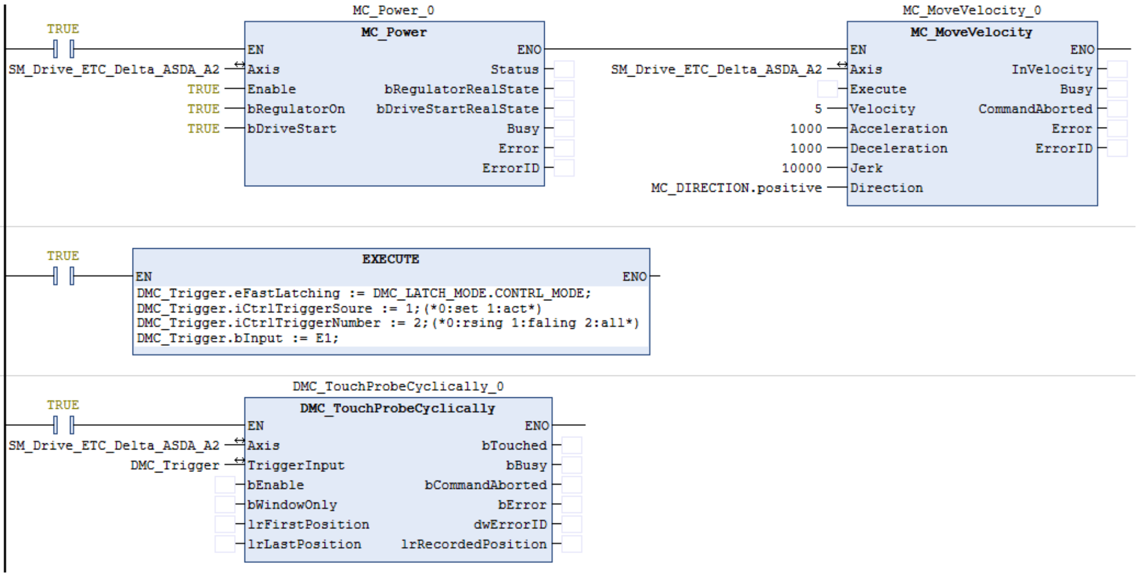
  - DMC_TouchProbeCyclically cannot be used with MC_TouchProbe together. If MC_TouchProbe is already executed, an error will occur when DMC_TouchProbeCyclically is executed. DMC_TouchProbeCyclically will also report an error if MC_TouchProbe is executed during DMC_TouchProbeCyclically execution.

  - If the trigger signal is DRIVE_MODE, the position stored in the servo is read directly and then the *iCtrlTriggerSource* of *TriggerInput* is an invalid parameter.

  - *bInput* of *TriggerInput* is the trigger signal source under CONTRL_MODE, and it is an invalid parameter under DRIVE_MODE mode.

  - When DRIVE_MODE is used, TouchProbe1 and TouchProbe2 can be started respectively in two independent DMC_TouchProbeCyclically instructions.

- **Troubleshooting**
  - When an error occurs in the instruction execution or the axis enters Errorstop state, *bError* changes to True and the axis stops running. To confirm current error state, see the error code in ErrorID.
- **Programming Example**
  - Programming Example1:
    - This example illustrates how to use DMC_TouchProbeCyclically in CONTRL_MODE mode.



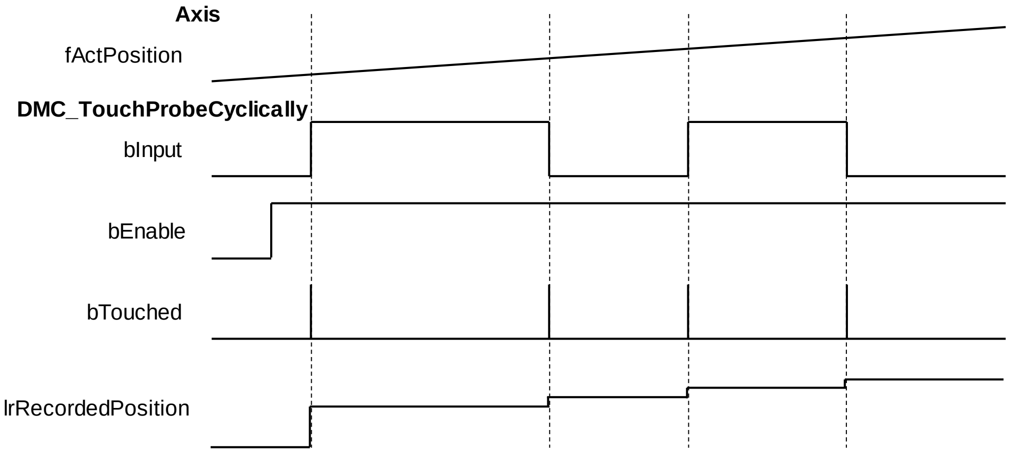
```
MC_Power_0
MC_Power
TRUE ─┤├─ EN                                  ENO
SM_Drive_ETC_Delta_ASDA_A2 ─ Axis             Status
TRUE ─ Enable                                  bRegulatorRealState
TRUE ─ bRegulatorOn                            bDriveStartRealState
TRUE ─ bDriveStart                             Busy
                                               Error
                                               ErrorID

MC_MoveVelocity_0
MC_MoveVelocity
EN                                             ENO
SM_Drive_ETC_Delta_ASDA_A2 ─ Axis              InVelocity
Execute                                        Busy
5 ─ Velocity                                   CommandAborted
1000 ─ Acceleration                            Error
1000 ─ Deceleration                            ErrorID
10000 ─ Jerk
MC_DIRECTION.positive ─ Direction

TRUE ─┤├─ EXECUTE
EN                                             ENO
DMC_Trigger.eFastLatching := DMC_LATCH_MODE.CONTRL_MODE;
DMC_Trigger.iCtrlTriggerSoure := 1;(*0:set 1:act*)
DMC_Trigger.iCtrlTriggerNumber := 2;(*0:rsing 1:faling 2:all*)
DMC_Trigger.bInput := E1;

DMC_TouchProbeCyclically_0
DMC_TouchProbeCyclically
TRUE ─┤├─ EN                                  ENO
SM_Drive_ETC_Delta_ASDA_A2 ─ Axis             bTouched
DMC_Trigger ─ TriggerInput                     bBusy
bEnable                                        bCommandAborted
bWindowOnly                                    bError
lrFirstPosition                                dwErrorID
lrLastPosition                                 lrRecordedPosition
```

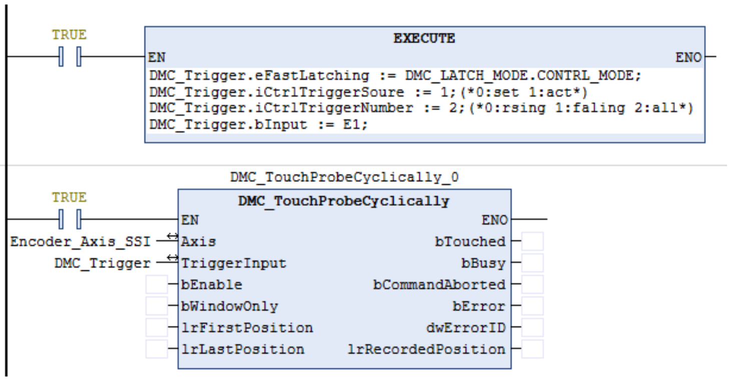
    - Timing Diagram



1. DMC_TouchProbeCyclically specifies CONTRL_MODE and uses the rising edge or falling edge as the trigger signal with the actual position of the axis (*fActPosition*) as the reference position.
2. When the signal source *bInput* of *TriggerInput* is triggered in CONTRL_MODE mode and the state of *bInput* changes, the function block will record the actual position of the current axis, and *bTouched* will remain True for one period.

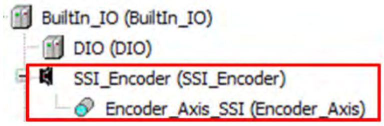
- Programming Example 2:

  - The example illustrates how to use DMC_TouchProbeCyclically by using SSI Encoder as the signal source in CONTRL_MODE.

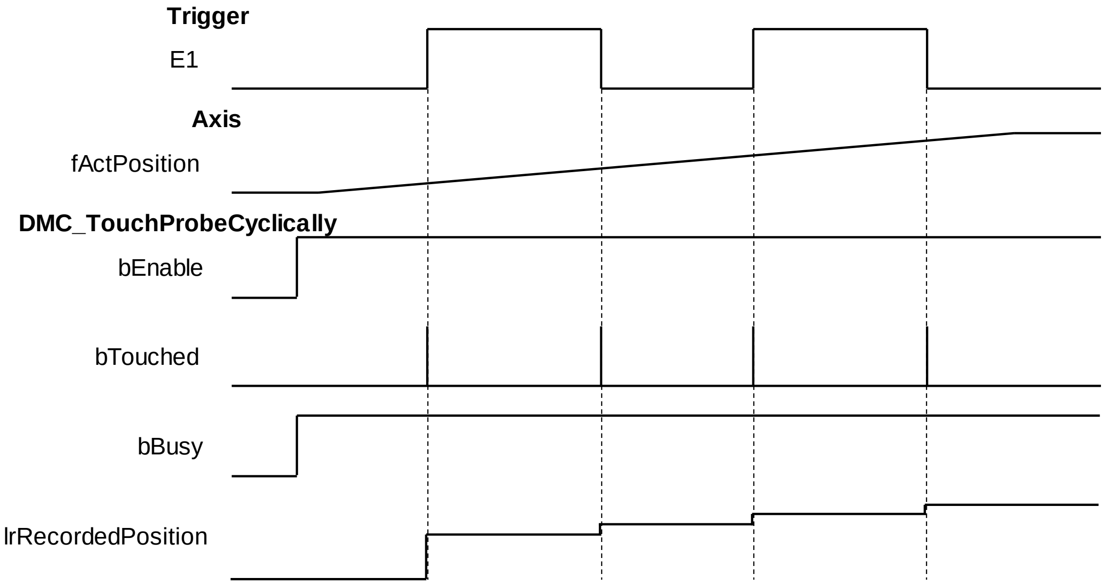


```
DMC_Trigger.eFastLatching := DMC_LATCH_MODE.CONTRL_MODE;
DMC_Trigger.iCtrlTriggerSoure := 1;(*0:set 1:act*)
DMC_Trigger.iCtrlTriggerNumber := 2;(*0:rsing 1:faling 2:all*)
DMC_Trigger.bInput := E1;
```

  - Device tree setting



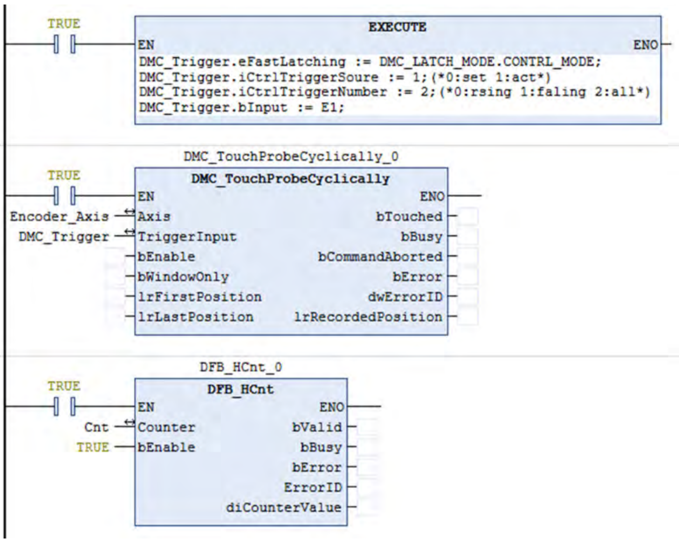
  - Timing Diagram



1. Select SSI Encoder as the signal source for DMC_TouchProbeCyclically.
2. Add an SSI Encoder to the device tree and then connect the SSI Encoder to the AX-308 module. For wiring, please refer to section 2.2.4 CPU Module Input and Output Terminals in **AX-3 Series Operation Manual**.
3. When the *bInput* of *TriggerInput* is triggered, DMC_TouchProbeCyclically will record the position of the current SSI Encoder.

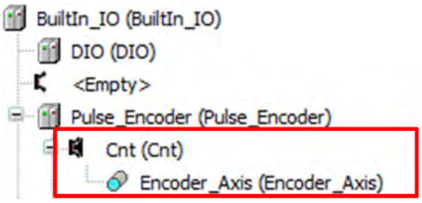
- Programming Example 3:

◆ The example illustrates how to use DMC_TouchProbeCyclically with Pulse Encoder as the signal source in CONTRL_MODE mode.

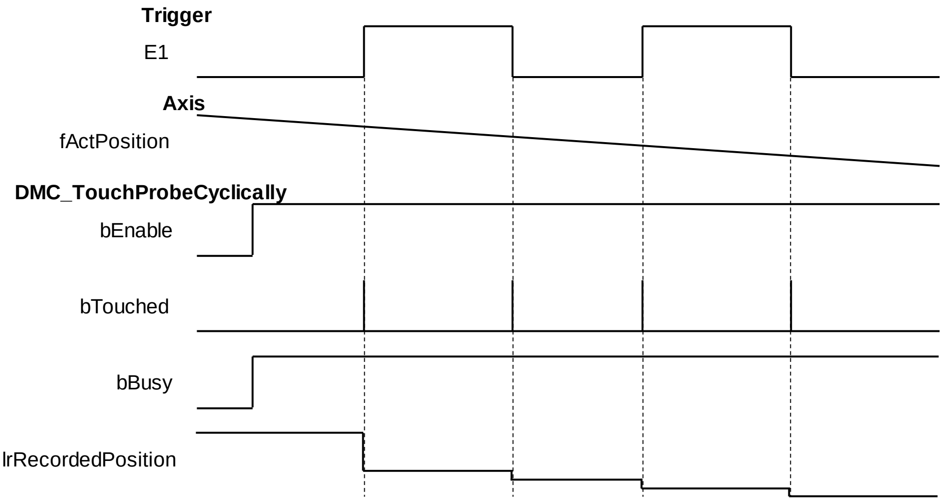


```
EXECUTE
DMC_Trigger.eFastLatching := DMC_LATCH_MODE.CONTRL_MODE;
DMC_Trigger.iCtrlTriggerSoure := 1;(*0:set 1:act*)
DMC_Trigger.iCtrlTriggerNumber := 2;(*0:rsing 1:faling 2:all*)
DMC_Trigger.bInput := E1;
```

◆ Device tree setting



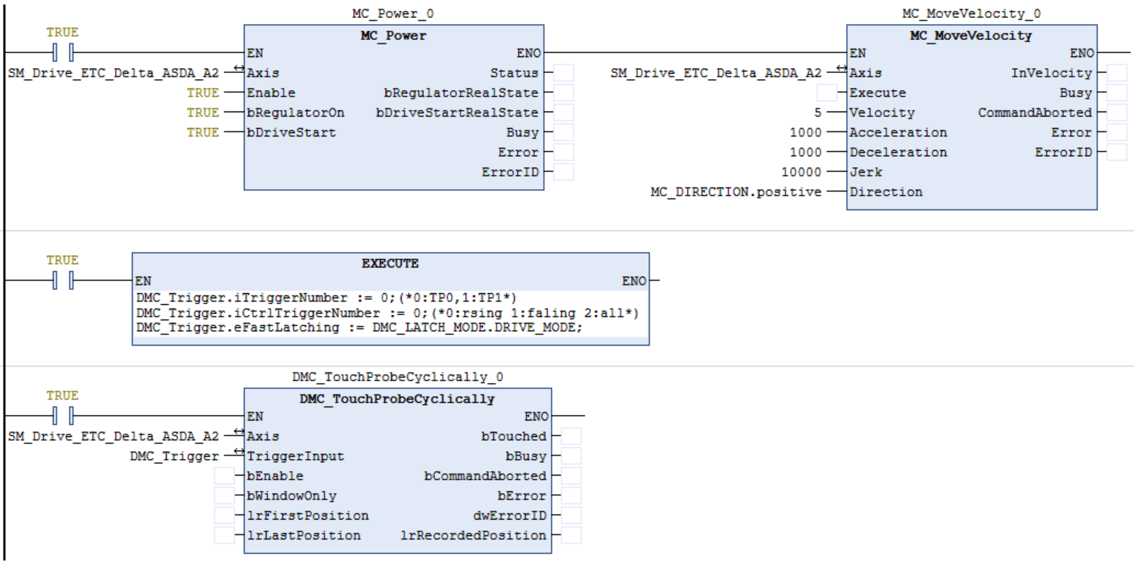
◆ Timing Diagram



1. Select Pulse Encoder as the signal source for DMC_TouchProbeCyclically.
2. Add a Count to the device tree (here is Count 1), and then add DFB_HCnt to the program to read the value of the Pulse Encoder. Finally connect the Pulse Encoder. For wiring, please refer to Section 2.2.4 CPU Module Input and Output Terminals in **AX-3 Series Operation Manual**.

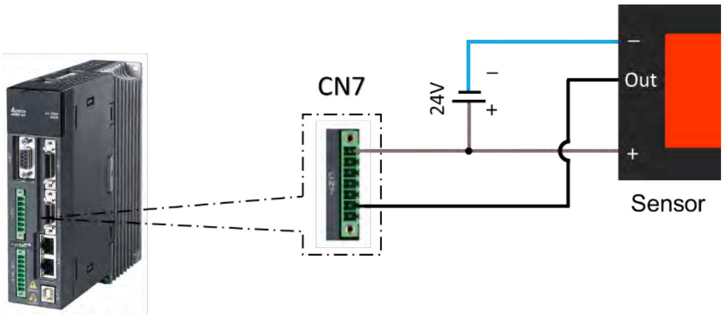
3. When the *bInput* of *TriggerInput* is triggered, DMC_TouchProbeCyclically will record the position of the current Pulse Encoder.

- Programming Example 4:
  - The example illustrates how to use DMC_TouchProbeCyclically in DRIVE_MODE.



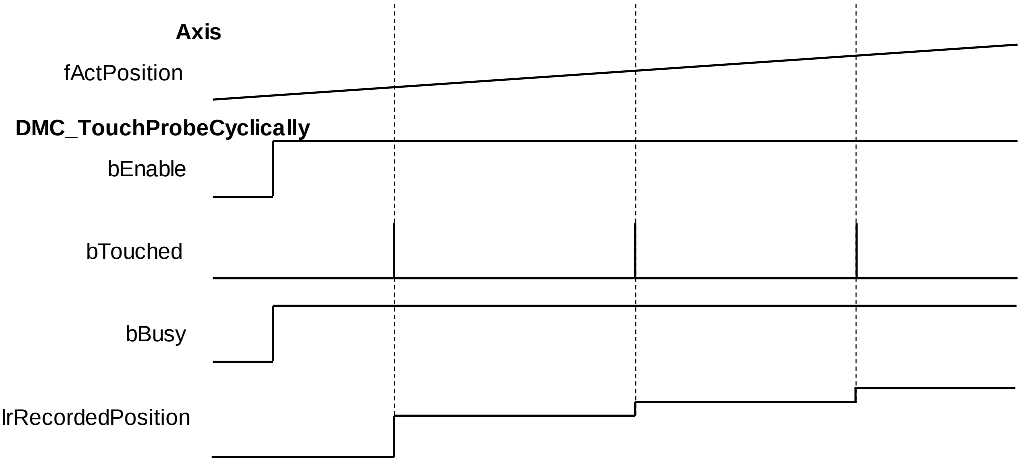
```
DMC_Trigger.iTriggerNumber := 0;(*0:TP0,1:TP1*)
DMC_Trigger.iCtrlTriggerNumber := 0;(*0:rsing 1:faling 2:all*)
DMC_Trigger.eFastLatching := DMC_LATCH_MODE.DRIVE_MODE;
```

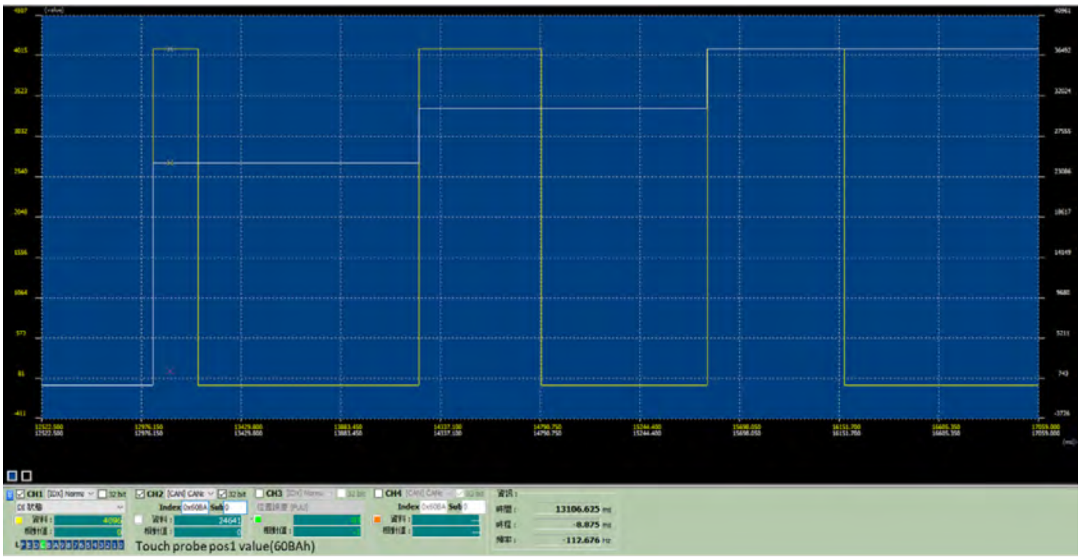
- Wiring figure



1. The trigger signal comes from DI13 of extension DIs of the servo drive's CN7. Please refer to the wiring figure above for configuration.
2. In this example, the TouchProbe 1 trigger is taken as an example and so the photoelectric switch is connected to DI13. If the TouchProbe 2 trigger is selected, the photoelectric switch should be connected to DI14.

- Timing Diagram





1. DMC_TouchProbeCyclically specifies DRIVE_MODE with TouchProbe 1 which is triggered by the rising edge signal.
2. When the switch trigger occurs, the drive will record the current position, send it back to the controller and record it in the function block lrRecordedPosition, and *bTouched* will remain True for one period.
3. In DRIVE_MODE, the drive will record the current position in real time and thus the recorded position will be earlier than the actual feedback position of the controller.

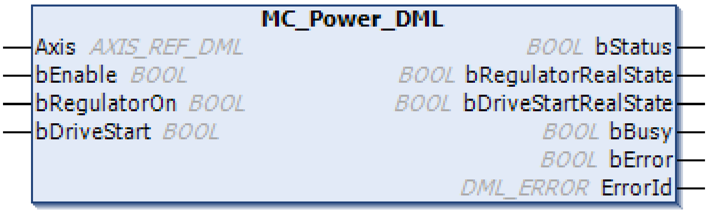
- **Supported Products**
  - AX-308

2.3.3 Positioning Axis Instructions

The function blocks in this section come from the function library「DL_MotionControlLight」. The main motion curve planning and calculation of function blocks are handled by the drive. So please select the positioning axis when setting the axis. For related settings on a positioning axis, please refer to section 7.4 in AX-3 Series Operation Manual.

2.3.3.1 MC_Power_DML

MC_Power_DML is used to enable, disable and immediately stop the specified axis.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>_DML</td><td>
</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">MC_Power_DML_instance (
Axis :=,
bEnable :=,
bRegulatorOn:=,
bDriveStart :=,
bStatus =>,
bRegulatorRealState =>,
bDriveStartRealState =>,
bBusy =>,
bError =>,
ErrorID =>);</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bEnable	The instruction is executed when *bEnable* shifts from False to True.	BOOL	True/False (False)	-
bRegulatorOn	Power ON	BOOL	True/False (False)	Only valid when *bEnable* is True.
bDriveStart	Disable the immediate stop mechanism.	BOOL	True/False (False)	Only valid when *bEnable* is True.

- **Outputs**

Name	Function	Data type	Output range (Default)
bStatus	True when the specified axis can move.	BOOL	True/False (False)
bRegulatorRealState	True when the power is ON	BOOL	True/False (False)
bDriveStartRealState	True when the immediate stop mechanism can be used.	BOOL	True/False (False)

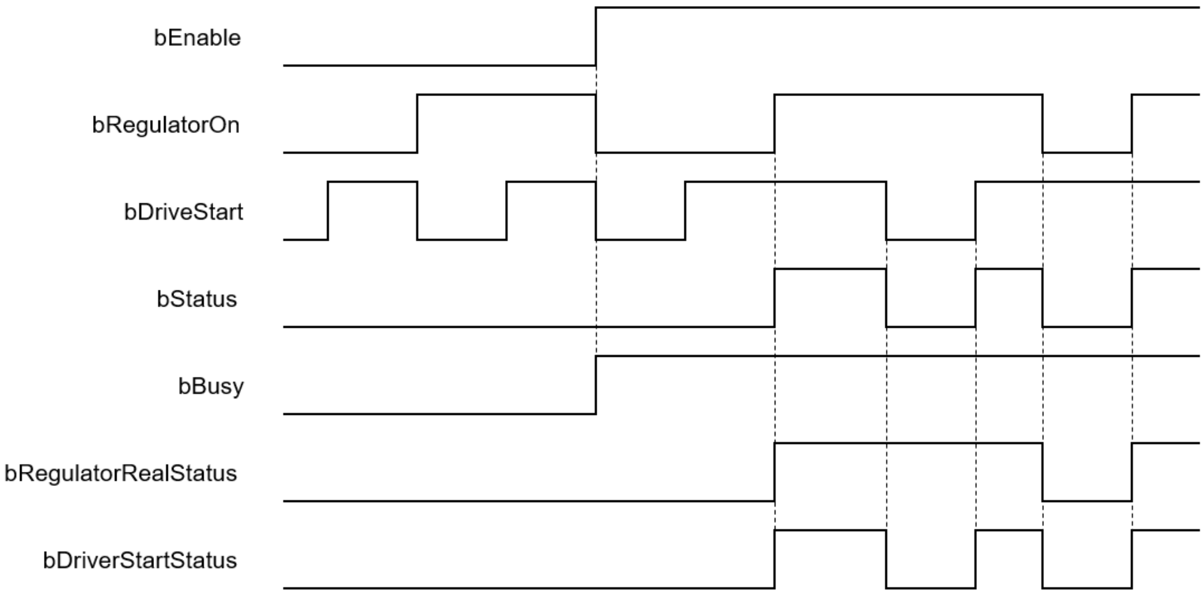
Name	Function	Data type	Output range (Default)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bError	True when an error occurs in execution of the instruction.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to **Appendices** for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bStatus	● When *bEnable* is True and *bRegulatorRealState* and *bDriveStartRealState* shift to True.	● When *bEnable* is True and *bRegulatorRealState* or *bDriveStartRealState* shifts to False. ● When *bError* shifts to True.
bRegulatorRealState	● When *bEnable* and *bRegulatorRealState* are True.	● When *bEnable* is True and *bRegulatorRealState* shifts to False. ● When *bError* shifts to True.
bDriveStartRealState	● When *bEnable* and *bRegulatorRealState*, *bDriveStartRealState* are True.	● *When bEnable is True,* and *bRegulatorRealState* or *bDriveStartRealState* shifts to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* shifts to True.	● When *bEnable* shifts to False. ● When *bError* shifts to True.
bError ErrorID	● When an error occurs in the execution conditions or input values for the instruction.	● When error code is cleared.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bEnable* shifts to True.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

  - *bRegulatorOn* and *bDriveStart* are effective only when *bEnable* is True.
  - When *bEnable*, *bRegulatorOn* and *bDriveStart* are all True, *bStatus* changes to True and *nAxisState* (state machine) changes to Standstill
  - When *bEnable* and *bRegulatorOn* are True and then *bDriveStart* is set to False, *nAxisState* (state machine) changes to Stopping.
  - When *bEnable* and *bDriveStart* are True and then *bRegulatorOn* is set to False, *nAxisState* (state machine) changes to Disabled.

- **Troubleshooting**

  - When an error occurs in the instruction execution or the axis enters Errorstop state, *bError* changes to True and the axis stops running. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**

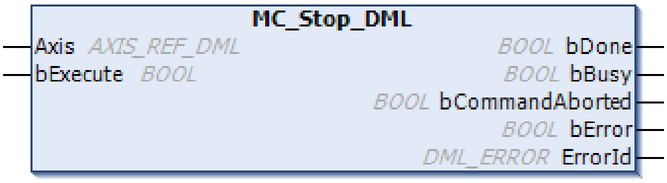
  - For the example, please refer to the programming example for MC_Power function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**

  - AX-308E and AX-8

2.3.3.2 MC_Stop_DML

MC_Stop_DML decelerates the specified axis to a stop.

FB/FC	Instruction	Graphic expression
FB	MC_Stop_DML	 MC_Stop_DML Axis AXIS_REF_DML bExecute BOOL BOOL bDone BOOL bBusy BOOL bCommandAborted BOOL bError DML_ERROR ErrorId

ST expression

MC_Stop_DML_instance (
Axis :=,
bExecute :=,
bDone =>,
bBusy =>,
bCommandAborted=>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-

- **Outputs**

Name	Function	Data type	Output range (Default)
bDone	True when the velocity reaches 0.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Contains error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

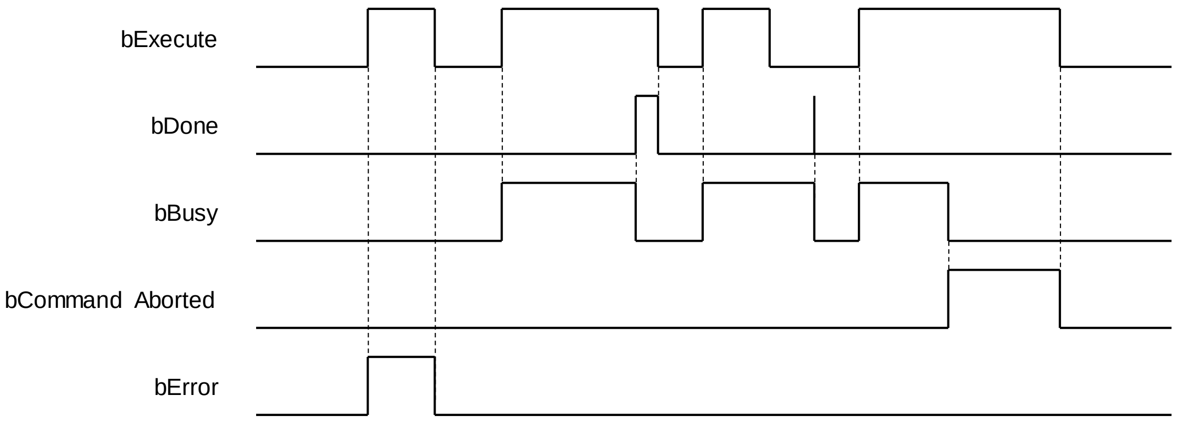
***Note:**

DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the axis decelerates to a stop or the velocity is 0.	● When *bExecute* shifts from True to False. ● If *bExecute* is False and *bDone* shifts to True, *bDone* will be True for one period and then immediately shift to False.
bCommandAborted	● When the axis state switches to Disabled during instruction execution.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and then immediately shift to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

*Note:

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

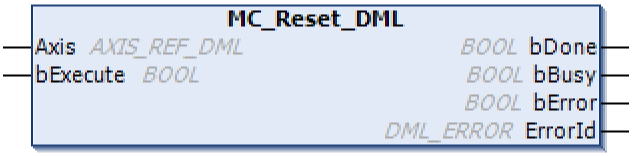
- **Function**
  - MC_Stop_DML can be used to stop the axis in motion and State Machine enters Stopping.
  - When MC_Power is set to False during deceleration, the motor will be in Free Run.
  - When the axis velocity is already decreased to 0 and *Done* of MC_Stop changes to True, *Execute* of MC_Stop changes to False and State Machine changes from Stopping to Standstill.
  - The deceleration rate can follow the setting value of Quick stop deceleration (16#6085) in the CiA402 object dictionary.
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_Stop function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.3 MC_Reset_DML

MC_Reset_DML clears axis-related errors.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>MC_Reset_DML</td><td>

MC_Reset_DML
Axis AXIS_REF_DML
bExecute BOOL
BOOL bDone
BOOL bBusy
BOOL bError
DML_ERROR ErrorId

</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">MC_Reset_DML_instance (
Axis :=,
bExecute :=,
bDone =>,
bBusy =>,
bError =>,
ErrorID =>);</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-

- **Outputs**

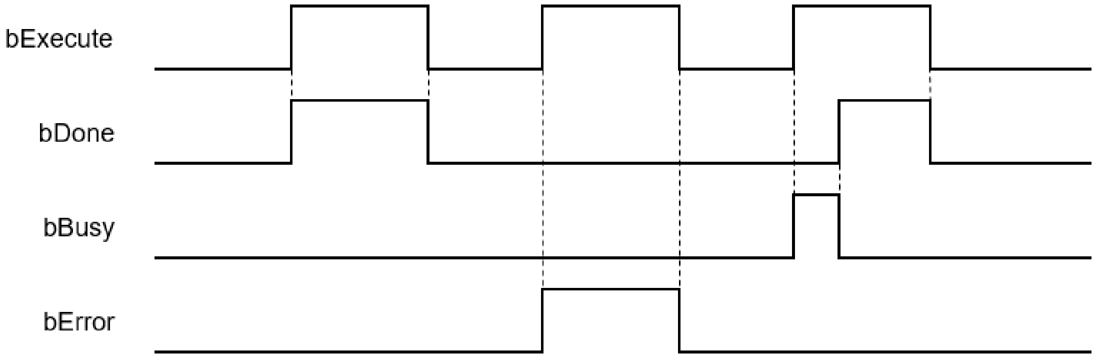
Name	Function	Data type	Output range (Default)
bDone	True when axis error clearing is completed and the axis enters Standstill or Disabled.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- ■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When axis error clearing is completed.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bDone* shifts to True, *bDone* will be True for one period and then immediately shift to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bError* shifts to True. ● When *bDone* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

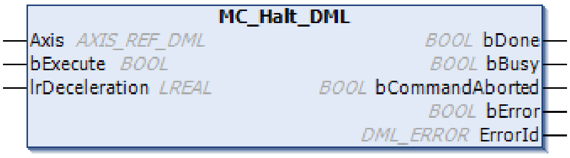
***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - ■ MC_Reset_DML can change the axis from abnormal error state to normal operational state.
    When MC_Power_DML.Enable is True, the axis state changes from Errorstop to Standstill,
    When MC_Power_DML.Enable is False, the axis state changes from Errorstop to Disabled.
  - ■ When the servo controller reports an error, MC_Reset_DML can be used to clear the error. After the error is cleared, the axis state will return to Standstill or Disabled.
  - ■ If errors (e.g. a communication error) can not be cleared by MC_Reset_DML, the instruction will report DML_R_ERROR_NOT_RESETTABLE (122) error.
- **Troubleshooting**
  - ■ When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.
- **Programming Example**
  - ■ For the example, please refer to the programming example for MC_Reset function block
  - ■ For *Axis* of the function block, please enter a positioning axis.
- **Supported Products**
  - ■ AX-308E and AX-8

2.3.3.4 MC_Halt_DML

MC_Halt_DML halts an axis in a controllable way.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>MC_Halt_DML</td><td>
</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">MC_Halt_DML_instance (
Axis :=,
bExecute :=,
lrDeceleration :=,
bDone =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-
lrDeceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True.

- **Outputs**

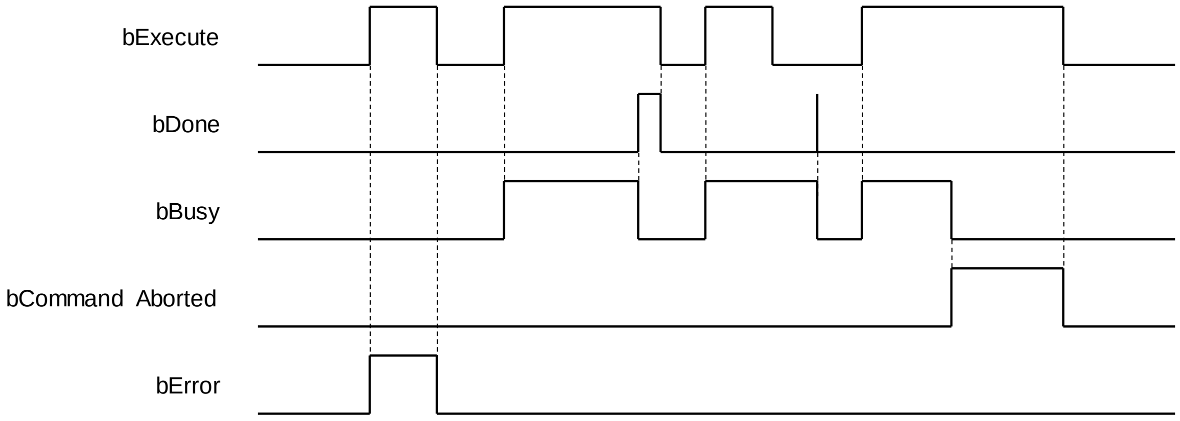
Name	Function	Data type	Output range (Default)
bDone	True when the axis stops and the velocity is 0.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the axis decelerates to a stop.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bDone* shifts to True, *bDone* will be True for one period and then immediately shift to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bCommandAborted	● When this instruction is aborted by another function block.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and immediately shift to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

*Note:

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - MC_Halt_DML is different from MC_Stop_DML in stopping the axis motion. MC_Halt_DML can be interrupted by other motion function blocks.
  - When MC_Halt_DML is executed, the axis will enter discrete_motion state. When the velocity of the axis reaches zero, the axis will enter Standstill state.

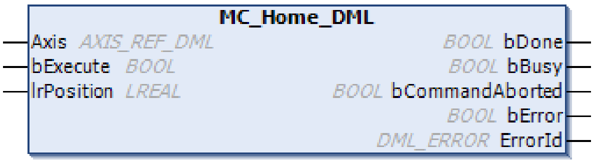
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_Halt function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.5 MC_Home_DML

MC_Home_DML controls the axis to perform the homing operation.

FB/FC	Instruction	Graphic expression
FB	MC_Home_DML	 MC_Home_DML Axis AXIS_REF_DML bExecute BOOL lrPosition LREAL BOOL bDone BOOL bBusy BOOL bCommandAborted BOOL bError DML_ERROR ErrorId

ST expression

MC_Home_DML_instance (
Axis :=,
bExecute:=,
lrPosition:=,
bDone =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-
lrPosition	Specify the absolute home position. (Unit: user unit)	LREAL	Positive number, negative number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

Name	Function	Data type	Output range (Default)
bDone	True when homing is completed and the axis is in Standstill state.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to	DML_ERROR*	DML_ERROR (DML _NO_ERROR)

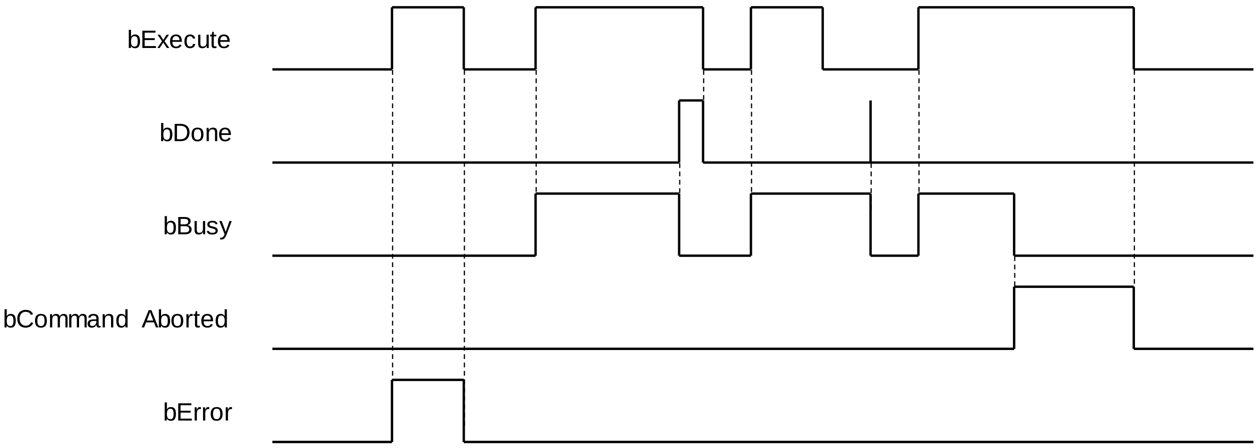
Name	Function	Data type	Output range (Default)
	Appendices for error code descriptions.		

***Note:** DML_ERROR: Enumeration (ENUM)

■ Output Update Timing

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the homing is completed.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bDone* shifts to True, *bDone* will be True for one period and then immediately shift to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bCommandAborted	● When this instruction is aborted by another instruction. ● When the instruction is aborted by MC_Stop_DML.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and immediately shift to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When error code is cleared.
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - This function block is executed only when the axis is in Standstill state and the state is Homing during the instruction execution. The function block cannot be executed when the axis is in any state else.
  - *Position* is the absolute position when the homing is completed.
  - The home mode can be selected from the axis parameter page.

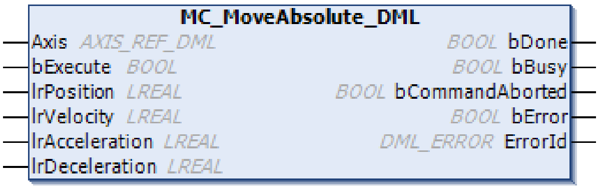
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_Home function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.6 MC_MoveAbsolute_DML

MC_MoveAbsolute_DML controls the specified axis to move to the specified absolute target position based on the specified motion behavior.

FB/FC	Instruction	Graphic expression
FB	MC_MoveAbsolute_DML	 MC_MoveAbsolute_DML Axis AXIS_REF_DML bExecute BOOL lrPosition LREAL lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL BOOL bDone BOOL bBusy BOOL bCommandAborted BOOL bError DML_ERROR ErrorId
ST expression		

MC_MoveAbsolute_DML_instance(
Axis :=,
bExecute :=,
lrPosition :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
bDone =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);		

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-
lrPosition	Absolute target position (Unit: user unit)	LREAL	Negative number, positive number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
lrVelocity	Target velocity (Unit: user unit/s)	LREAL	Positive number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
lrAcceleration	Acceleration rate (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True and *bBusy* is False.
lrDeceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

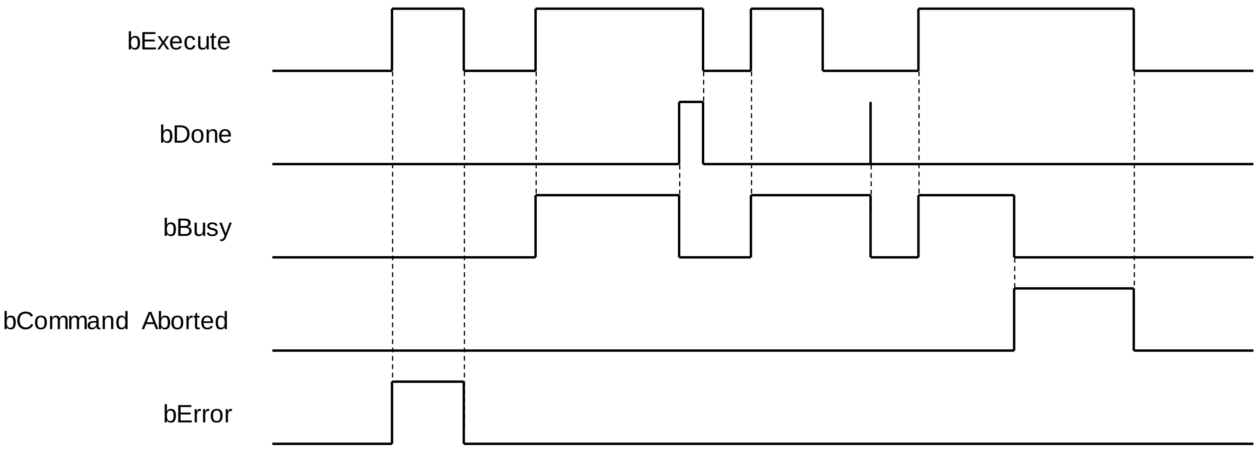
Name	Function	Data type	Output range (Default)
bDone	True when the absolute target position is reached.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● True when the absolute target position is reached.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bDone* shifts to True, *bDone* will be True for one period and then immediately shift to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bCommandAborted	● When this instruction is aborted by another instruction. ● When the instruction is aborted by MC_Stop_DML.	● When bExecute shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and then immediately shift to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When bExecute shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - MC_MoveAbsolute_DML performs absolute positioning according to specified target velocity (*lrVelocity*), acceleration rate (*lrAcceleration*) and deceleration rate (*lrDeceleration*) when *bExecute* shifts to True.

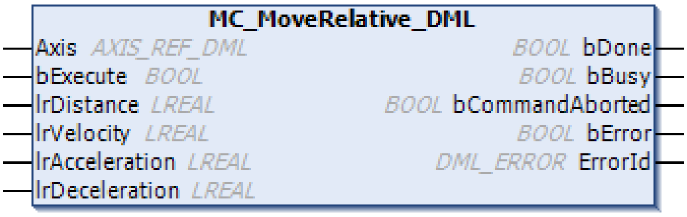
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_MoveAbsolute function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.7 MC_MoveRelative_DML

MC_MoveRelative_DML controls the specified axis to move to the specified relative target position according to the specified motion behavior.

FB/FC	Instruction	Graphic expression
FB	MC_MoveRelative_DML	

ST expression

MC_MoveRelative_DML_instance(
Axis :=,
bExecute :=,
lrDistance :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
bDone =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *Execute* shifts to True.	BOOL	True/False (False)	-
lrDistance	Relative distance to be moved (Unit: user unit)	LREAL	Negative number, positive number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
lrVelocity	Target velocity (Unit: user unit/s)	LREAL	Positive number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
lrAcceleration	Acceleration rate (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True and *bBusy* is False.
lrDeceleration	Deceleration rate (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

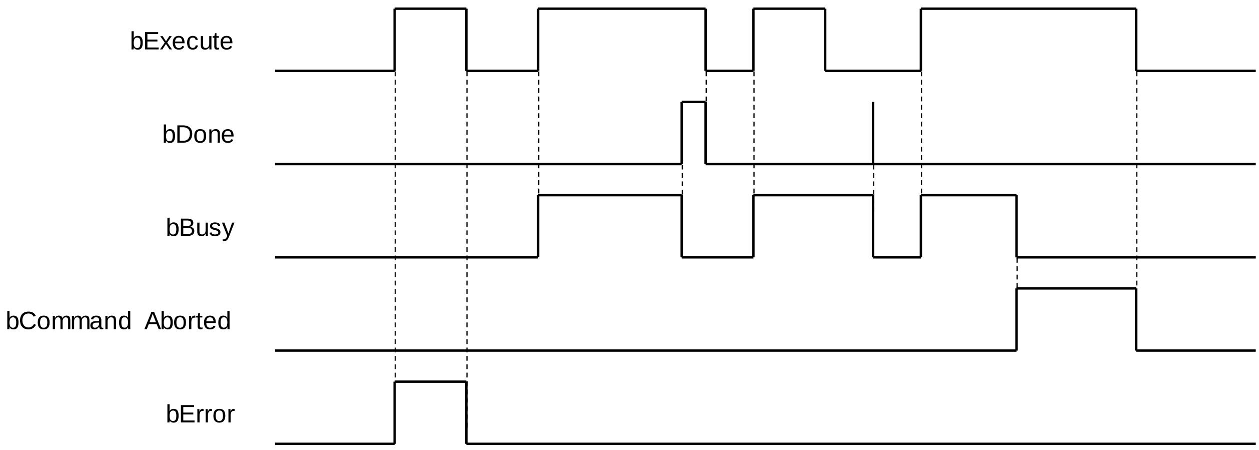
Name	Function	Data type	Output range (Default)
bDone	True when the relative distance is completed.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the relative positioning is completed.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bDone* shifts to True, *bDone* will be True for one period and immediately shift to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bDone* shifts to True. ● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bCommandAborted	● When this instruction is aborted by another instruction. ● When the instruction is aborted by MC_Stop_DML.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and then immediately shift to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**
AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - MC_MoveRelative_DML performs relative positioning according to specified target velocity (*lrVelocity*), acceleration rate (*lrAcceleration*) and deceleration rate (*lrDeceleration*) when *bExecute* shifts to True.

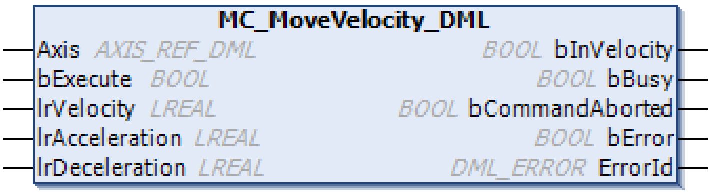
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_MoveRelative function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.8 MC_MoveVelocity_DML

MC_MoveVelocity_DML performs velocity control on an axis in the position mode with a specified behavior and a constant velocity.

FB/FC	Instruction	Graphic expression
FB	MC_MoveVelocity_DML	 MC_MoveVelocity_DML Axis AXIS_REF_DML bExecute BOOL lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL BOOL bInVelocity BOOL bBusy BOOL bCommandAborted BOOL bError DML_ERROR ErrorId
ST expression		

MC_MoveVelocity_DML_instance (
Axis :=,
bExecute :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
bInVelocity =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);		

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-
lrVelocity	Target velocity (Unit: user unit/s)	LREAL	Positive number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
lrAcceleration	Acceleration rate (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True and *bBusy* is False.
lrDeceleration	Deceleration rate. (Unit: user unit/s^2)	LREAL	Positive number (0)	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

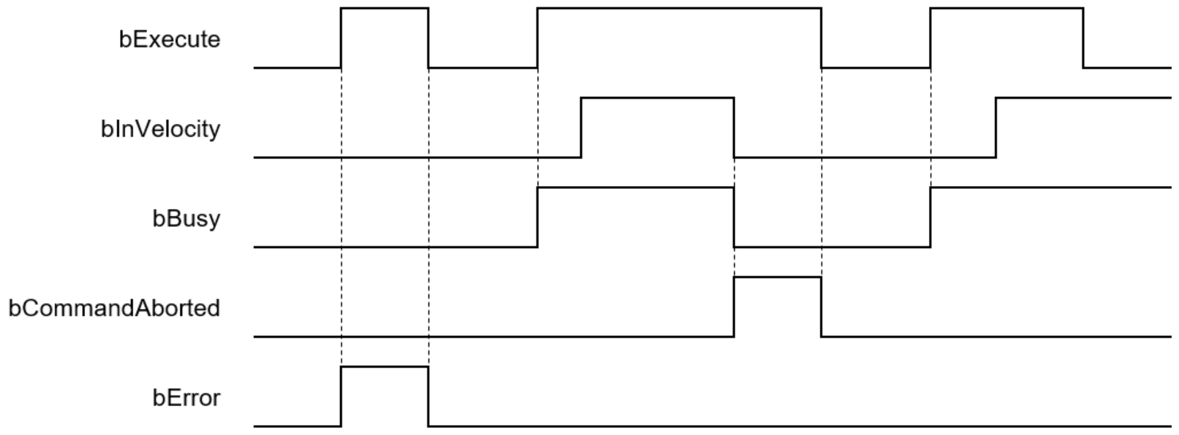
Name	Function	Data type	Output range (Default)
bInVelocity	True when the target velocity is reached.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bInVelocity	● When the target velocity is reached.	● When *bCommandAborted* shifts to True. ● When *bExecute* shifts to True again and *lrVelocity* value is changed.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bCommandAborted	● When this instruction is aborted by another instruction. ● When the instruction is aborted by MC_Stop_DML.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and then immediately shift to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

  - When *bExecute* shifts to True, the instruction will perform constant-velocity motion according to the specified target speed (*lrVelocity*), acceleration (*lrAcceleration*), and deceleration (*lrDeceleration*).

  - The executing MC_MoveVelocity_DML can be aborted by another motion instruction.

  - When the instruction is aborted by another instruction, the output *bInVelocity* changes to False and the output *bCommandAborted* changes to True.

  - When *bExecute* of MC_MoveVelocity_DML switches to True, the axis will start to move at the target velocity. Even if *bExecute* switches to False, the operation of the function block will not be affected.

  - When *bExecuteInputs* of MC_MoveVelocity_DML changes to True again and a new target velocity is assigned, the axis velocity is adjusted to the new velocity.

  - When *bExecute* changes to False after the function block is executed and then the target velocity is reached, the *bInVelocity* of MC_MoveVelocity_DML changes to True. Afterward, *bInVelocity* will be True until it is aborted by another instruction.

- **Troubleshooting**

  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**

  - For the example, please refer to the programming example for MC_MoveVelocity function block

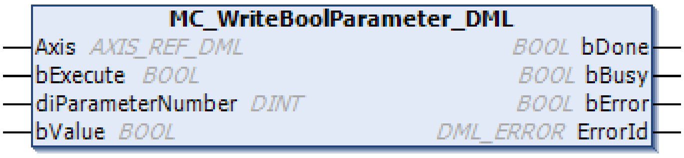
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**

  - AX-308E and AX-8

2.3.3.9 MC_WriteBoolParameter_DML

MC_WriteBoolParameter_DML writes a Boolean value in the specified parameter.

FB/FC	Instruction	Graphic expression
FB	MC_WriteBoolParameter_DML	 MC_WriteBoolParameter_DML Axis AXIS_REF_DML bExecute BOOL diParameterNumber DINT bValue BOOL BOOL bDone BOOL bBusy BOOL bError DML_ERROR ErrorId

ST expression

MC_WriteBoolParameter_instance(
Axis :=,
bExecute :=,
diParameterNumber :=,
bValue :=,
bDone =>,
bBusy =>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts to True.	BOOL	True/False (False)	-
diParameterNumber	Specify the number of the axis parameter.	DINT	Positive number, negative number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
bValue	Set a Boolean value of the parameter to write.	BOOL	True/False (False)	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

Name	Function	Data type	Output range (Default)
bDone	True when the parameter writing is completed.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)

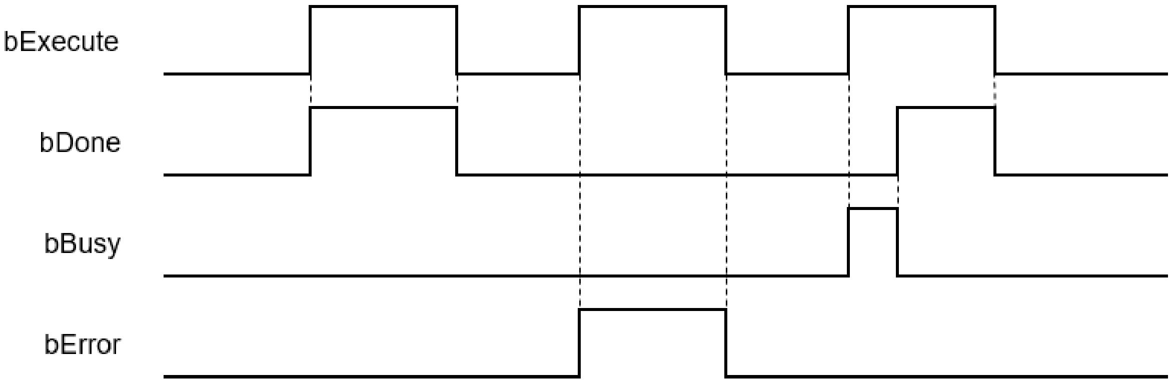
Name	Function	Data type	Output range (Default)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the parameter writing is completed.	● When *bExecute* shifts from True to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed. ● When parameter writing is in progress.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**
AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

  - How to use MC_WriteBoolParameter_DML to write the number of an EtherCAT object dictionary

    - Use the SHL instruction to shift the data length of the object dictionary where a value is to be written to the left by 24 bits

    - Use the SHL instruction to shift the index of the object dictionary where a value is to be written to the left by 8 bits

    - Add up the above parameters and the sub-index.

      See the reference formula as follows.

      diParameterNumber :=-DWORD_TO_DINT (SHL (TO_DWORD (object dictionary data length), 24) + SHL (TO_DWORD (object dictionary index), 8) + object dictionary sub-index);

  - To write a value in an axis parameter, please refer to the axis parameter AXIS_REF_DML (FB) and fill in its number in the *diParameterNumber* input parameter.

- **Troubleshooting**

  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**

  - For the example, please refer to the programming example for MC_WriteParameter function block

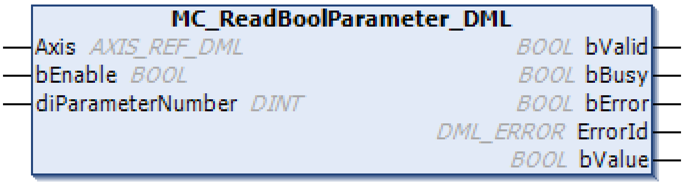
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**

  - AX-308E and AX-8

2.3.3.10 MC_ReadBoolParameter_DML

MC_ReadBoolParameter_DML reads the Boolean value of a specified parameter.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>MC_ReadBoolParameter_DML</td><td>

MC_ReadBoolParameter_DML
Axis AXIS_REF_DML
bEnable BOOL
diParameterNumber DINT
BOOL bValid
BOOL bBusy
BOOL bError
DML_ERROR ErrorId
BOOL bValue

</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">MC_ReadBoolParameter_DML_instance(
Axis :=,
bEnable :=,
diParameterNumber :=,
bValid =>,
bBusy =>,
bError =>,
ErrorID =>,
bValue =>);</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bEnable	The instruction is executed when *bEnable* shifts to True.	BOOL	True/False (False)	-
diParameterNumber	Specify the number of the axis parameter.	DINT	Positive number, negative number or 0 (0)	When *bEnable* shifts to True.

- **Outputs**

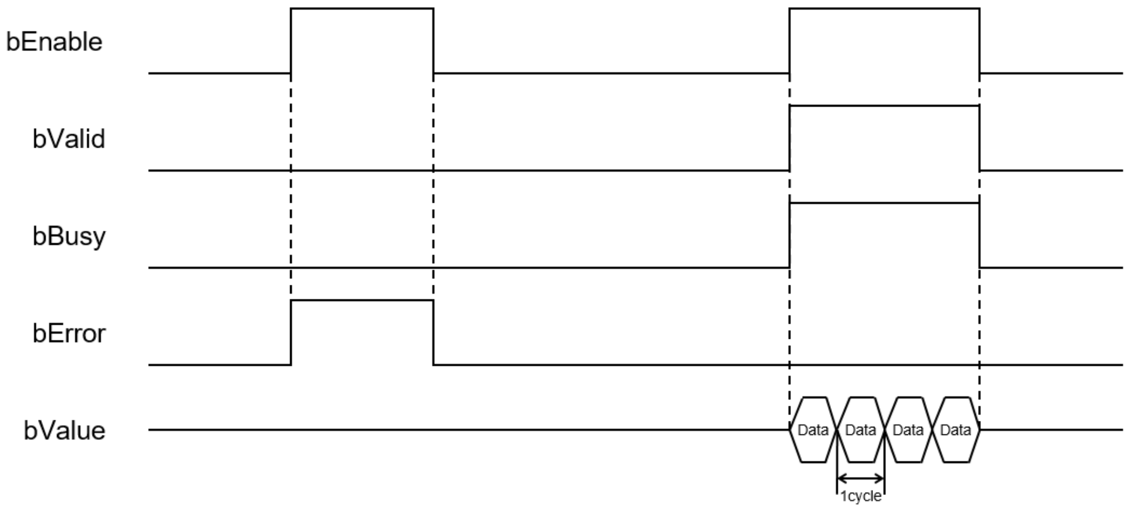
Name	Function	Data type	Output range (Default)
bValid	True when the read parameter value is available.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)
bValue	The read parameter value	BOOL	True/False (False)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* shifts to True. ● When the parameter to be read is available.	● When *bEnable* shifts from True to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* shifts to True and the instruction is executed. ● When the parameter to be read is available.	● When *bEnable* shifts from True to False. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* is False. (Error code is cleared)
ErrorID		
bValue	● Updates continuously when *bValid* is True.	● Update stops when *bValid* is False.

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bEnable* changes to True.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - How to use MC_ReadBoolParameter_DML to read the number of an EtherCAT object dictionary.
    - Use the SHL instruction to shift the data length of the object dictionary to be read to the left by 24 bits
    - Use the SHL instruction to shift the index of the object dictionary to be read to the left by 8 bits
    - Add up the above parameters and the sub-index.

      See the reference formula as follows.
    - diParameterNumber := - DWORD_TO_DINT (SHL (TO_DWORD (object dictionary data length), 24) + SHL (TO_DWORD (object dictionary index), 8) + object dictionary sub-index);
  - To read an axis parameter value, please refer to the axis parameter AXIS_REF_DML (FB) and fill in its number in the *diParameterNumber* input parameter.

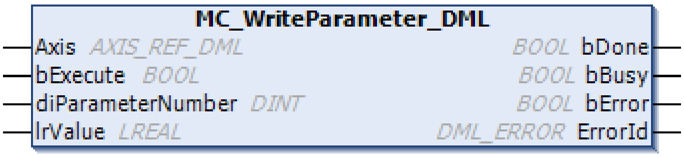
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_ReadParameter function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.11 MC_WriteParameter_DML

MC_WriteParameter_DML writes a value in the specified parameter.

B/FC	Instruction	Graphic expression
FB	MC_WriteParameter_DML	 MC_WriteParameter_DML Axis AXIS_REF_DML bExecute BOOL diParameterNumber DINT lrValue LREAL BOOL bDone BOOL bBusy BOOL bError DML_ERROR ErrorId

ST expression

```
MC_WriteParameter_DML_instance(
Axis :=,
bExecute :=,
diParameterNumber :=,
lrValue :=,
bDone =>,
bBusy =>,
bError =>,
ErrorID => );
```

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts to True.	BOOL	True/False (False)	-
diParameter Number	Specify the number of the axis parameter.	DINT	Positive number, negative number or 0 (0)	When *bExecute* is True and bBusy is False.
lrValue	Set a parameter value to write.	LREAL	Positive number, negative number or 0 (0)	When *bExecute* is True and bBusy is False.

- **Outputs**

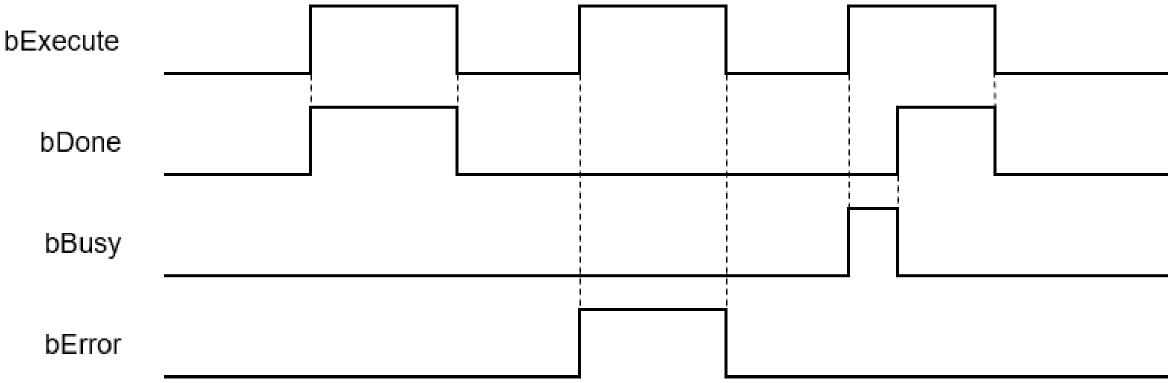
Name	Function	Data type	Output range (Default)
bDone	True when the parameter writing is completed.	BOOL	True/False (False)
bBusy	True when the instruction is exectued.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the parameter writing is completed.	● When *bExecute* shifts from True to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed. ● When parameter writing is in progress.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**
AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

  - How to use MC_WriteParameter_DML to write the number of an EtherCAT object dictionary.

    - Use the SHL instruction to shift the data length of the object dictionary where a value is to be written to the left by 24 bits

    - Use the SHL instruction to shift the index of the object dictionary where a value is to be written to the left by 8 bits

    - Add up the above parameters and the sub-index.

    See the reference formula as follows.

    diParameterNumber :=-DWORD_TO_DINT (SHL (TO_DWORD (object dictionary data length), 24) + SHL (TO_DWORD (object dictionary index), 8) + object dictionary sub-index);

  - To write a value in an axis parameter, please refer to the axis parameter AXIS_REF_DML (FB) and fill in its number in the *diParameterNumber* input parameter.

- **Troubleshooting**

  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**

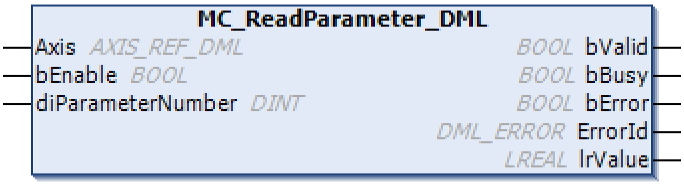
  - For the example, please refer to the programming example for MC_WriteParameter function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**

  - AX-308E and AX-8

2.3.3.12 MC_ReadParameter_DML

MC_ ReadParameter_DML reads the value of a specified parameter.

FB/FC	Instruction	Graphic expression
FB	MC_ReadParameter_DML	 MC_ReadParameter_DML Axis AXIS_REF_DML bEnable BOOL diParameterNumber DINT BOOL bValid BOOL bBusy BOOL bError DML_ERROR ErrorId LREAL lrValue
ST expression		

MC_ReadParameter_DML_instance (
Axis :=,
bEnable :=,
diParameterNumber :=,
bValid =>,
bBusy =>,
bError =>,
ErrorID =>,
lrValue =>);		

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bEnable	The instruction is executed when *bEnable* shifts to True.	BOOL	True/False (False)	-
diParameterNumber	Specify the number of the axis parameter.	DINT	Positive number, negative number or 0 (0)	*When bEnable shifts to True.*

- **Outputs**

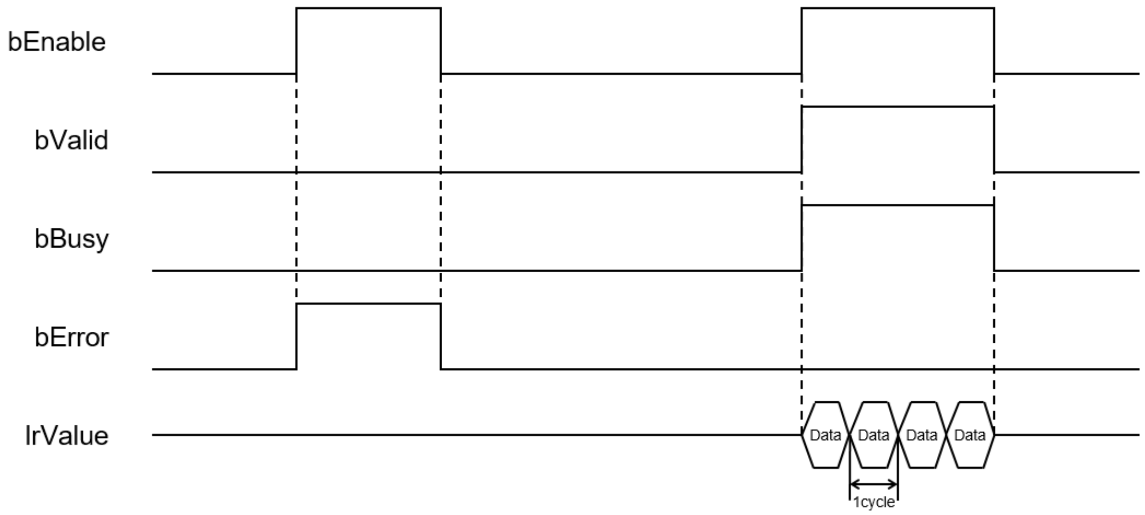
Name	Function	Data type	Output range (Default)
bValid	True when the read parameter value is available.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)
lrValue	The read parameter value	LREAL	Positive number, negative number or 0 (0)

***Note:** DML_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* shifts to True. ● When the read parameter value is available.	● When *bEnable* shifts from True to False. ● When *bError* shifts to True.
bBusy	● When *Enable* shifts to True and the instruction is executed. ● When the read parameter value is available.	● When *bEnable* shifts from True to False. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* is False. (Error code is cleared)
ErrorID		
lrValue	● Updates continuously when *bValid* is True.	● Update stops when *bValid* is False.

- **Timing Diagram**



***Note:**
Data = Parameter values
1 cycle = One task cycle

- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bEnable* shifts to True.

***Note:**
AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

■ How to use MC_ReadParameter_DML to read the number of an EtherCAT object dictionary

◆ Use the SHL instruction to shift the data length of the object dictionary to be read to the left by 24 bits

◆ Use the SHL instruction to shift the index of the object dictionary to be read to the left by 8 bits

- ◆ Add up the above parameters and the sub-index.

  See the reference formula as follows.

- ◆ diParameterNumber := - DWORD_TO_DINT (SHL (TO_DWORD (object dictionary data length), 24) + SHL (TO_DWORD (object dictionary index), 8) + object dictionary sub-index);

- ■ To read an axis parameter, please refer to the axis parameter AXIS_REF_DML (FB) and fill in its number in the *diParameterNumber* input parameter.

- **Troubleshooting**

  - ■ When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**

  - ■ For the example, please refer to the programming example for MC_ReadParameter function block

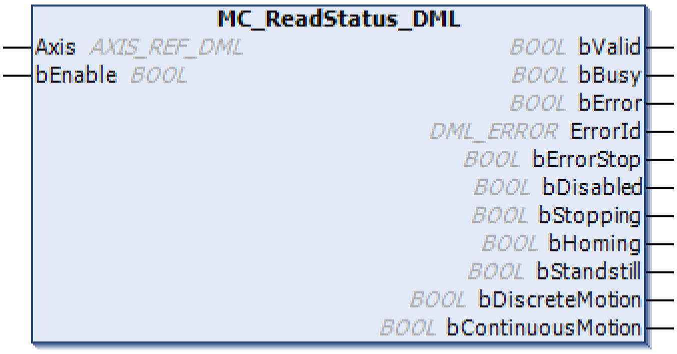
  - ■ For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**

  - ■ AX-308E and AX-8

2.3.3.13 MC_ReadStatus_DML

MC_ReadStatus_DML reads the state of a specified axis.

FB/FC	Instruction	Graphic expression
FB	MC_ReadStatus_DML	 MC_ReadStatus_DML Axis AXIS_REF_DML bEnable BOOL BOOL bValid BOOL bBusy BOOL bError DML_ERROR ErrorId BOOL bErrorStop BOOL bDisabled BOOL bStopping BOOL bHoming BOOL bStandstill BOOL bDiscreteMotion BOOL bContinuousMotion

ST expression

MC_ReadStatus_DML_instance (
Axis :=,
bEnable :=,
bValid =>,
bBusy =>,
bError =>,
ErrorID =>,
bErrorStop=>,
bDisabled=>,
bStopping=>,
bHoming=>,
bStandStill=>,
bDiscreteMotion=>,
bContinuousMotion=>);

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bEnable	The instruction is executed when *bEnable* shifts to True.	BOOL	True/False (False)	-

- **Outputs**

Name	Function	Data type	Output range (Default)
bValid	True when the axis state at the output is available.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)

Name	Function	Data type	Output range (Default)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*[1]	DML_ERROR (DML_NO_ERROR)
bErrorStop	To know details on the axis state machine, refer to SML_AXIS_STATE.*[2]	BOOL	True/False (False)
bDisabled		BOOL	True/False (False)
bStopping		BOOL	True/False (False)
bHoming		BOOL	True/False (False)
bStandStill		BOOL	True/False (False)
bDiscreteMotion		BOOL	True/False (False)
bContinuousMotion		BOOL	True/False (False)

***Note:**

1. DML_ERROR: Enumeration (ENUM)
2. SML_AXIS_STATE: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bValid	● When *bEnable* shifts to True. ● When the axis state at the output is available.	● When *bEnable* shifts from True to False. ● When *bError* shifts to True.
bBusy	● When *bEnable* shifts to True and the instruction is executed.	● When *bEnable* shifts from True to False. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bEnable* is False. (Error code is cleared)
ErrorID		
bDisabled	● When the axis is in Disabled state.	● When the axis is not in Disabled state.
bErrorstop	● When the axis is in Errorstop state.	● When the axis is not in Errorstop state.
bStopping	● When the axis is in Stopping state.	● When the axis is not in Stopping state.
bStandStill	● When the axis is in StandStill state.	● When the axis is not in StandStill state.
bDiscreteMotion	● When the axis is in Discrete Motion state.	● When the axis is not in Discrete Motion state.
bContinuousMotion	● When the axis is in Continuous Motion state.	● When the axis is not in Continuous Motion state.
bHoming	● When the axis is in Homing state.	● When the axis is not in Homing state.

- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bEnable* shifts to True.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

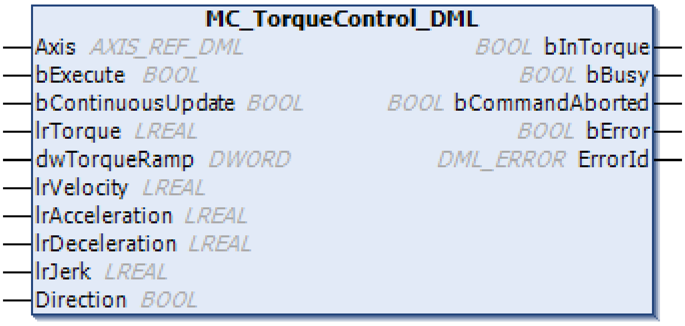
- **Troubleshooting**
  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - For the example, please refer to the programming example for MC_ReadStatus function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E and AX-8

2.3.3.14 MC_TorqueControl_DML

MC_TorqueControl_DML controls the torque by using the torque control mode of the applied servo drive.

FB/FC	Instruction	Graphic expression
FB	DMC_TorqueControl	 MC_TorqueControl_DML Axis AXIS_REF_DML bExecute BOOL bContinuousUpdate BOOL lrTorque LREAL dwTorqueRamp DWORD lrVelocity LREAL lrAcceleration LREAL lrDeceleration LREAL lrJerk LREAL Direction BOOL BOOL bInTorque BOOL bBusy BOOL bCommandAborted BOOL bError DML_ERROR ErrorId

ST expression

```
MC_TorqueControl_DML_instance(
Axis :=,
bExecute :=,
bContinuousUpdate :=,
lrTorque :=,
dwTorqueRamp :=,
lrVelocity :=,
lrAcceleration :=,
lrDeceleration :=,
lrJerk :=,
Direction :=,
bInTorque =>,
bBusy =>,
bCommandAborted =>,
bError =>,
ErrorID =>);
```

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts from False to True.	BOOL	True/False (False)	-
bContinuousUpdate	Continuously updates the target torque when *Continuousupdate* is True.	BOOL	True/False (False)	When *bExecute* shifts to True and *Busy* is False.
lrTorque	Specify the target torque. (Unit: N.m)	LREAL	Positive number, negative number or 0 (0)	When *bExecute* shifts to True and *Busy* is False.
dwTorqueRamp	Specify the change rate of the torque from current torque to target torque. (Unit: ms) *	DWORD	Positive number (0)	When *bExecute* shifts to True and *Busy* is False.

Name	Function	Data type	Setting value (Default)	Timing for updating
lrVelocity	Specify the maximum velocity.	LREAL	Positive number (0)	When *bExecute* shifts to True and *Busy* is False.
lrAcceleration	Reserved	LREAL	-	-
lrDeceleration	Reserved	LREAL	-	-
lrJerk	Reserved	LREAL	-	-
Direction	Reserved	BOOL	-	-

***Note:**

Here is ASDA-A2 as an example with the unit of microsecond. For other servo models, please refer to 0x6087 in the object dictionary.

- **Outputs**

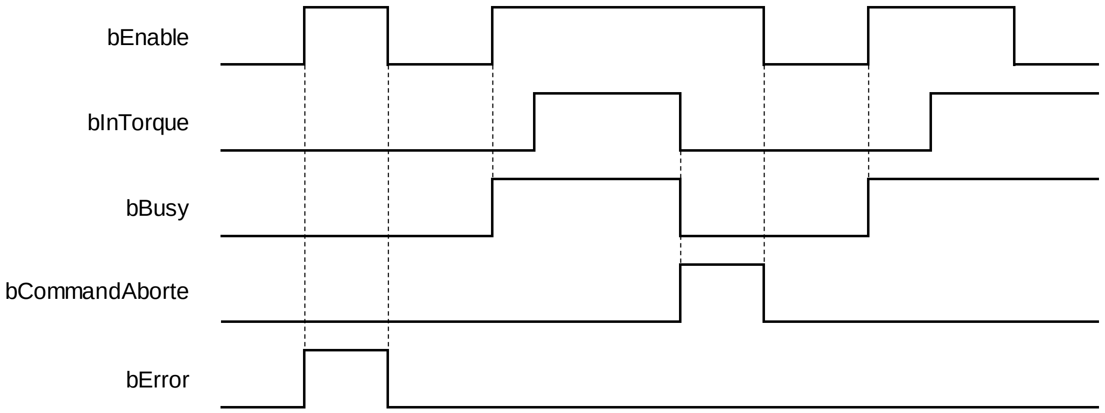
Name	Function	Data type	Output range (Default)
bInTorque	True when the target torque is reached.	BOOL	True/False (False)
bBusy	True when the instruction is executed.	BOOL	True/False (False)
bCommandAborted	True when the instruction is aborted.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NoError)

***Note:** DML_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bInTorque	● When *bExecute* shifts to True and the axis state is available.	● When *bError* shifts to True. ● When *bCommandAborted* shifts to True. ● When *bExecute* shifts to True again and *lrTorque* value changes.
bBusy	● When *bExecute* shifts to True and the instruction is executed.	● When *bError* shifts to True. ● When *bCommandAborted* shifts to True.
bCommandAborted	● When the instruction is aborted.	● When *bExecute* shifts to False. ● If *bExecute* is False and *bCommandAborted* shifts to True, *bCommandAborted* will be True for one period and then immediately shift to False.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**
AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

  - When *bExecute* of the instruction shifts to True, the target torque (*lrTorque*), torque ramp (*dwTorqueRamp*) and maximum velocity (*lrVelocity*) will be sent to the servo for servo torque control.
  - When *bExecute* is True, the instruction is executed for a velocity-constant motion according to the specified target velocity (*lrVelocity*), acceleration rate (*lrAcceleration*) and deceleration rate (*lrDeceleration*).
  - The executing MC_TorqueControl_DML can be aborted by executing another motion instruction.
  - When MC_TorqueControl_DML is aborted by another motion instruction, the output *bInTorque* will shift to False and the output *bCommandAborted* will shift to True.
  - When *bExecute* of MC_TorqueControl_DML shifts to True, the axis starts to move according to the target velocity. Even if *bExecute* shifts to False, the instruction execution will not be affected.
  - When *bExecute* of MC_TorqueControl_DML shifts to True again and a new *lrTorque* value is set, the torque of the axis is adjusted to the new torque value.
  - When its *bExecute* shifts to False after the instruction is executed and the target torque is reached, *bInTorque* of MC_TorqueControl_DML shifts to True. Afterward, *bInTorque* will remain True until it is aborted by another instruction.

- **Troubleshooting**

  - When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

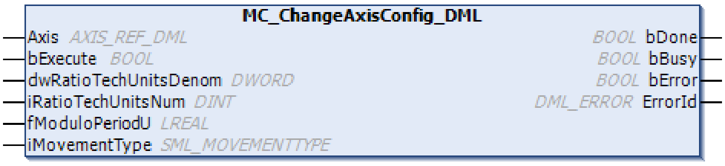
- **Programming Example**

  - For the example, please refer to the programming example for MC_TorqueControl function block
  - For *Axis* of the function block, please enter a positioning axis.

- **Supported Products**
  - AX-308E

2.3.3.15 MC_ChangeAxisConfig_DML

MC_ChangeAxisConfig_DML modifies basic axis settings including the ratio between user units and pulse number, axis type and user units per rotary axis rotation.

FB/FC	Instruction	Graphic expression
FB	MC_ChangeAxisConfig_DML	 MC_ChangeAxisConfig_DML Axis AXIS_REF_DML bExecute BOOL dwRatioTechUnitsDenom DWORD iRatioTechUnitsNum DINT fModuloPeriodU LREAL iMovementType SML_MOVEMENTTYPE BOOL bDone BOOL bBusy BOOL bError DML_ERROR ErrorId

ST expression

MC_ChangeAxisConfig_DML_instance(
Axis :=,
bExecute :=,
dwRatioTechUnitsDenom :=,
iRatioTechUnitsNum :=,
fModuloPeriodU :=,
fMovementType :=,
bDone =>,
bBusy =>,
bError =>,
ErrorID =>);

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts to True.	BOOL	True/False (False)	-
dwRatioTechUnitsDenom	Electronic gear ratio denominator (Pulse number)	DWORD	Positive number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
iRatioTechUnitsNum	Electronic gear ratio numerator (User units)	DINT	Positive number, negative number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
fModuloPeriodU	Maximum position of the rotary axis	LREAL	Positive number, negative number or 0 (0)	When *bExecute* shifts to True and *bBusy* is False.
fMovementType	Linear axis/rotary axis	SML_MOVEMENTTYPE	0: SML_MT_MODULO 1: SML_MT_FINITE	When *bExecute* shifts to True and *bBusy* is False.

- **Outputs**

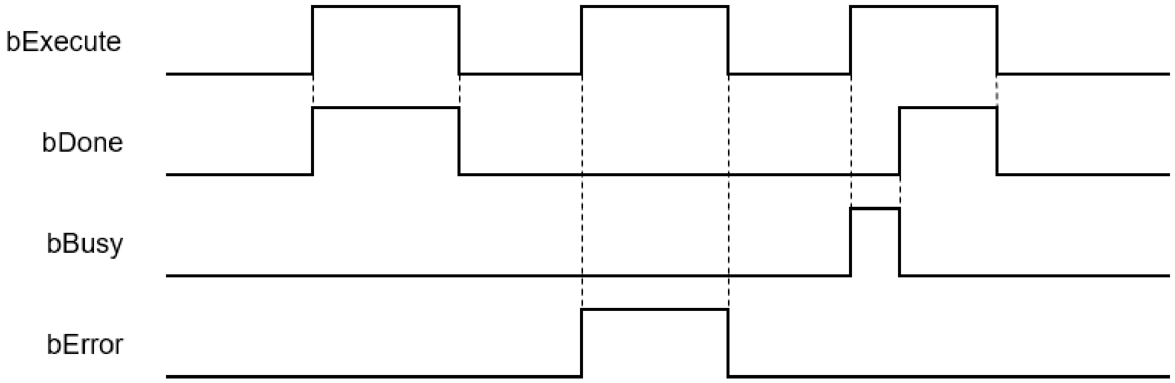
Name	Function	Data type	Output range (Default)
bDone	True when the parameter writing is completed.	BOOL	True/False (False)
bBusy	True when the instruction is exectued.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

■ **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When the parameter writing is completed.	● When *bExecute* shifts from True to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed. ● When the parameter writing is in progress.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

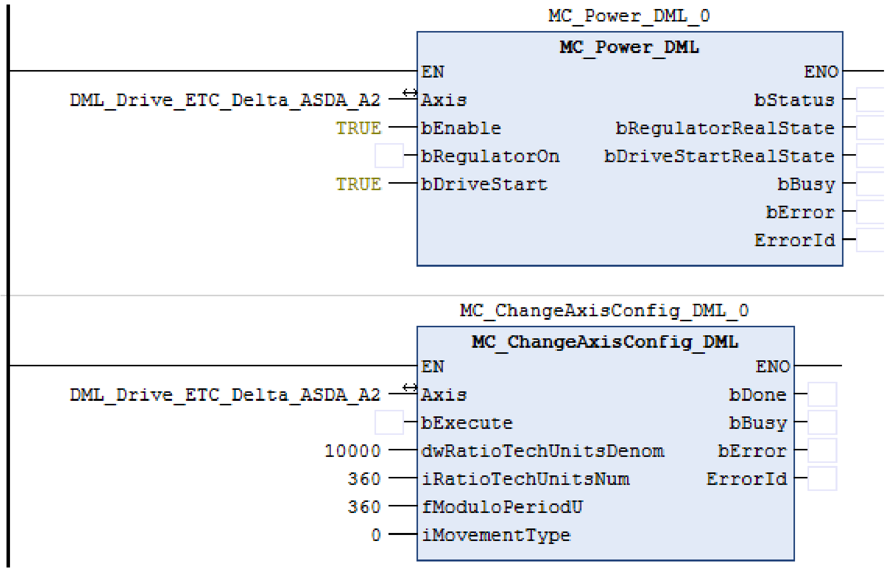
***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**
  - ■ MC_ChangeAxisConfig_DML can be used to modify basic axis settings including the ratio between user units and pulse number (electronic gear ratio), axis type and user units per rotary axis rotation.
  - ■ The axis state must be Disabled if this function block is used.
  - ■ After modification, the new axis settings cannot be retained when power off and so they will disappear after repowering or resetting. And the axis will still run based on the settings on the axis parameter page next time.

- **Troubleshooting**
  - ■ When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Programming Example**
  - ■ This example explains how MC_ChangeAxisConfig_DML is used to modifiy axis parameters.

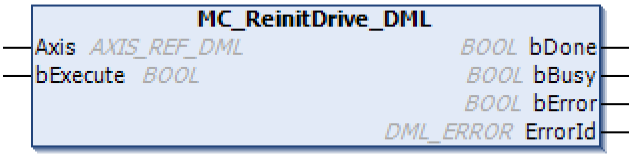


  - ■ Set the gear ratio denominator and gear ratio numerator (10000: 360), maximum position of the rotary axis (360) and axis type (0). Change *bRegulatorOn* of MC_Power to False before MC_ChangeAxisConfig is executed.

- **Supported Products**
  - ■ AX-308E and AX-8

2.3.3.16 MC_ReinitDrive_DML

MC_ReinitDrive_DML reinitializes the specified axis.

<table>
<tr><th>FB/FC</th><th>Instruction</th><th>Graphic expression</th></tr>
<tr><td>FB</td><td>MC_ReinitDrive_DML</td><td>

MC_ReinitDrive_DML
Axis AXIS_REF_DML
bExecute BOOL
BOOL bDone
BOOL bBusy
BOOL bError
DML_ERROR ErrorId

</td></tr>
<tr><th colspan="3">ST expression</th></tr>
<tr><td colspan="3">MC_ReinitDrive_DML_instance(
Axis :=,
bExecute :=,
bDone =>,
bBusy =>,
bError =>,
ErrorID =>);</td></tr>
</table>

- **Inputs**

Name	Function	Data type	Setting value (Default)	Timing for updating
bExecute	The instruction is executed when *bExecute* shifts to True.	BOOL	True/False (False)	-

- **Outputs**

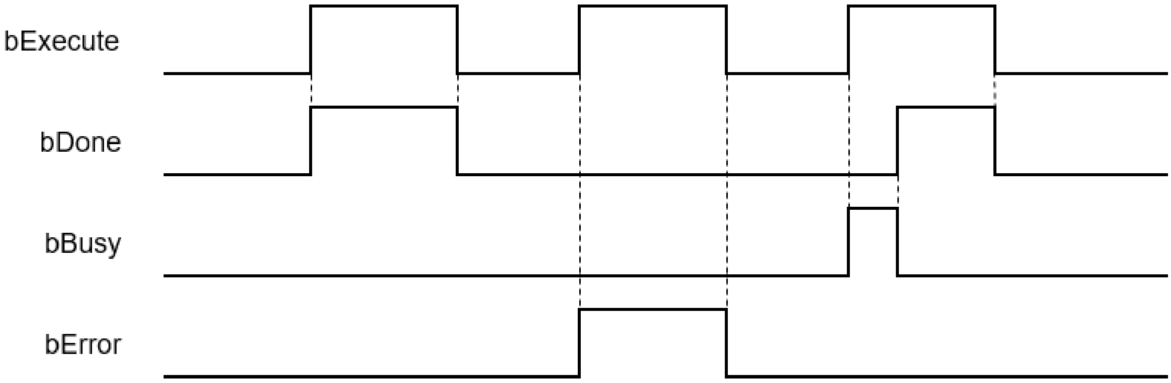
Name	Function	Data type	Output range (Default)
bDone	True when initialization is completed.	BOOL	True/False (False)
bBusy	True when the instruction is exectued.	BOOL	True/False (False)
bError	True when an error occurs.	BOOL	True/False (False)
ErrorID	Indicates the error code if an error occurs. Refer to Appendices for error code descriptions.	DML_ERROR*	DML_ERROR (DML_NO_ERROR)

***Note:** DML_ERROR: Enumeration (ENUM)

- **Output Update Timing**

Name	Timing for shifting to True	Timing for shifting to False
bDone	● When initialization is completed.	● When *bExecute* shifts from True to False.
bBusy	● When *bExecute* shifts to True and the instruction is executed. ● When initializating is in progress.	● When *bDone* shifts to True. ● When *bError* shifts to True.
bError	● When an error occurs in the execution conditions or input values for the instruction.	● When *bExecute* shifts to False. (Error code is cleared)
ErrorID		

- **Timing Diagram**



- **In-Outs**

Name	Function	Data type	Setting value	Timing for updating
Axis	Specify the axis	AXIS_REF_DML*	AXIS_REF_DML	When *bExecute* shifts to True and *bBusy* is False.

***Note:**

AXIS_REF_DML (FB): The interface is built in every function block and works as the starting program of the function block.

- **Function**

- MC_ReinitDrive_DML is used to initialize the drive which has an error or is no longer in synchronization.
- Resetting the network cannot clear errors of the positioning axis itself, e.g. left and right limits, emergency stop and etc.
- The function block resets EtherCAT network of the positioning axis and keeps the servo in the previous servo state. Please refer to the following table for details.

Before the reset action	Resetting	After the reset action
Servo Off	Servo Off	Servo Off
Servo On	Servo Off	Servo On

- **Troubleshooting**

- When an error occurs in the instruction execution, *bError* changes to True. To confirm current error state, see the error code in *ErrorID*.

- **Supported Products**

- AX-308E and AX-8

MEMO

Appendix A

Table of Contents

A.1 Instruction Tables and Indexes

A.1.1 By Function

- **Single-axis Motion Control Instruction (Synchronous axes)**

Categories	Name	Description
Positioning control	MC_Home	MC_Home controls the axis to perform the homing operation.
	MC_Stop	MC_Stop decelerates an axis to a stop.
	MC_Halt	MC_Halt commands a controlled axis motion stop.
	MC_MoveAbsolute	MC_MoveAbsolute controls the axis to move to the specified absolute target position at a specified behavior.
	MC_MoveRelative	MC_MoveRelative controls the axis to move a specified relative distance with a specified behavior.
	MC_MoveAdditive	MC_MoveAdditive controls the axis to move an additional distance at a given speed and acceleration.
	MC_MoveSuperimposed	MC_MoveSuperimposed controls the axis to move a relative superimposed distance at a specified behavior while the axis is moving.
	MC_PositionProfile	MC_PositionProfile commands a time-position locked motion profile.
	MC_Jog	MC_Jog enables an axis to be moved forward or backward.
	DMC_Home_P	DMC_Home_P, an application function block of pulse output, drives the pulse axis to perform the homing in the set mode.
	DMC_MoveVelocityStopByPos	DMC_MoveVelocityStopByPos controls an axis to stop at a specified position after a period of motion.
Velocity control	MC_MoveVelocity	MC_MoveVelocity performs velocity control on an axis in the position mode with a specified behavior and an average velocity.
	MC_VelocityProfile	MC_VelocityProfile commands a time-position locked motion profile; however, its position points are defined by the velocity variables in MC_TV_REF.
	MC_AccelerationProfile	MC_AccelerationProfile commands a time-position locked motion profile; however, its position points are defined by acceleration variables in MC_TV_REF.
	DMC_VelocityControl	DMC_VelocityControl performs a velocity control on a specified axis in the CSV speed mode with the specified behavior and an average velocity.
Torque control	DMC_TorqueControl	DMC_TorqueControl controls the torque according to the torque control mode of the servo drive.
Sync control	MC_CamIn	MC_CamIn performs cam operation.
	MC_CamOut	MC_CamOut deactivates the coupling between master and slave axis.
	MC_GearIn	The function block MC_GearIn activates a linear master-slave coupling.
	MC_GearOut	MC_GearOut disconnects the gear relation (velocity) between master and slave axis.

Categories	Name	Description
	MC_GearInPos	MC_GearInPos commands position synchronization which is achieved over a defined region of travel for both master and slave axis.
	MC_Phasing	MC_Phasing specifies the phase shift value between the master and slave axis. 。
Administrative	MC_Power	MC_Power enables or disables the specific axis.
	MC_SetPosition	MC_SetPositionn changes the current position by shifting the coordinates of an axis.
	MC_ReadParameter	MC_ReadParameter reads a value of a specific axis parameter.
	MC_WriteParameter	MC_WriteParameter writes a value to a specific parameter.
	MC_ReadBoolParameter	MC_ReadBoolParameter reads the value of a specific Boolean parameter.
	MC_WriteBoolParameter	MC_WriteBoolParameter writes a Boolean value to a specific parameter.
	MC_ReadActualPosition	MC_ReadActualPosition reads the current axis position.
	MC_ReadActualVelocity	MC_ReadActualVelocity reads the actual axis velocity value.
	MC_ReadActualTorque	MC_ReadActualTorque reads the actual torque value of axis.
	MC_Reset	MC_Reset clears axis-related errors so that the error memory is available for new error messages.
	MC_ReadStatus	MC_ReadStatus reads ther status of the specified axis.
	MC_ReadAxisError	MC_ReadAxisError reads the error information of axis.
	MC_CamTableSelect	MC_CamTableSelect selects the cam table for use with MC_CamIn.
	MC_TouchProbe	MC_TouchProbe records an axis position at the time when a trigger event occurs.
	MC_AbortTrigger	MC_AbortTrigger aborts the instruction MC_TouchProbe which are intended to capture trigger events.
	MC_DigitalCamSwitch	MC_DigitalCamSwitch uses the axis position to control a switch of a digital output.
Administrative	DMC_ChangeMechanism GearRation	DMC_ChangeMechanismGearRation modifies the ratio between user units and pulses, axis type and user units per rotation of the rotary axis.
	DMC_ReadMotionState	DMC_ReadMotionState reads the behavior state of the axis in motion.
	DMC_CamReadTappetStatus	DMC_CamReadTappetStatus reads the status of multiple tappets.
	DMC_CamReadTappetValue	DMC_CamReadTappetValue reads the data of one single tappet.
	DMC_CamWriteTappetValue	DMC_CamWriteTappetValue modifies the tappet data for the specified existing track.
	DMC_CamAddTappet	DMC_CamAddTappet adds a new tappet track at the end of the tappet table.

Categories	Name	Description
	DMC_CamDeleteTappet	DMC_CamDeleteTappet deletes the specified tappet track.
	DMC_CamReadPoint	DMC_CamReadPoint reads the data of one single cam point.
	DMC_CamWritePoint	DMC_CamWritePoint writes the data of one single cam point.
	DMC_AxesObserve	DMC_AxesObserve monitors the deviation between the master axis position and slave axis position and it will output a reminder when the deviation exceeds the allowed setting value.
	DMC_PositionLag	DMC_PositionLag sets the allowed range of lag error and observe whether the allowed position lag is exceeded
	DMC_SetTorqueLimit	DMC_SetTorqueLimit sets the maximum torque of an axis.
	DMC_SetSoftwareLimit	DMC_SetSoftwareLimit is used to enable, disable and set the upper and lower software limits
	DMC_CamKeyPointWrite	DMC_CamKeyPointWrite writes key cam points by selecting a curve type and generating corresponding cam curve based on related parameters. After the new cam curve is generated, the selected cam table will be changed accordingly.
	DMC_TouchProbeCyclically	DMC_TouchProbeCyclically can continuously record the captured position of an axis.

- **Single-axis Motion Control Instruction (Positioning axes)**

Categories	Name	Description
Positioning control	MC_Home_DML	MC_Home_DML controls the axis to perform the homing operation.
	MC_Stop_DML	MC_Stop_DML decelerates an axis to a stop.
	MC_Halt_DML	MC_Halt_DML commands a controlled axis motion stop.
Positioning control	MC_MoveAbsolute_DML	MC_MoveAbsolute_DML controls the axis to move to the specified absolute target position at a specified behavior.
	MC_MoveRelative_DML	MC_MoveRelative_DML controls the axis to move a specified relative distance with a specified behavior.
Velocity Control	MC_MoveVelocity_DML	MC_MoveVelocity_DML performs velocity control on an axis in the position mode with a specified behavior and an average velocity.
Torque Control	MC_TorqueControl_DML	MC_TorqueControl_DML controls the torque according to the torque control mode of the servo drive.
Administrative	MC_Power_DML	MC_Power_DML enables or disables the specific axis.
	MC_ReadParameter_DML	MC_ReadParameter_DML reads a value of a specific axis parameter.
	MC_WriteParameter_DML	MC_WriteParameter_DML writes a value to a specific parameter.
	MC_ReadBoolParameter_DML	MC_ReadBoolParameter_DML reads the value of a specific Boolean parameter.
	MC_WriteBoolParameter_DML	MC_WriteBoolParameter_DML writes a Boolean value to a specific parameter.
	MC_Reset_DML	MC_Reset_DML clears axis-related errors.

Categories	Name	Description
	MC_ReadStatus_DML	MC_ReadStatus_DML reads ther status of the specified axis.
	MC_ChangeAxisConfig_DML	MC_ChangeAxisConfig_DML modifies basic axis settings including the ratio between user units and pulse number, axis type and user units per rotary axis rotation.
	MC_ReinitDrive_DML	MC_ReinitDrive_DML re-initializes the axis.

- **Multi-axis Motion Control Instruction**

Categories	Name	Description
Group Motion	DMC_MoveLinearAbsolute	DMC_MoveLinearAbsolute controls a specified axis group to perform the absolute linear interpolation for a specified absolute position.
	DMC_MoveLinearRelative	DMC_MoveLinearRelative controls a specified axis group to perform the relative linear interpolation.
	DMC_MoveCircularAbsolute	DMC_MoveCircularAbsolute controls the axis group to perform circular or helical interpolation for a specified absolute target position.
	DMC_MoveCircularRelative	DMC_ MoveCircularRelative controls the axis group to perform circular or helical interpolation for a specified relative target position.
	DMC_GroupStop	DMC_GroupStop decelerates the group axes to a stop.
	DMC_GroupHalt	DMC_GroupHalt decelerates the axis group in motion to a pause.
Administrative	DMC_GroupEnable	DMC_GroupEnable switches the axis group state from GroupDisable to GroupStandby.
	DMC_GroupDisable	DMC_GroupDisable sets the state of an axis group to GroupDisable.
	DMC_GroupReadStatus	DMC_GroupReadStatus reads the state of an axis group.
	DMC_GroupReadError	DMC_GroupReadError reads axis group errors.
	DMC_GroupReset	DMC_GroupReset resets an axis group which is in GroupErrorstop state.

A.1.2 By Model

- The supported model types are listed as follows:

Function Block	Model Type	
	AX-3	AX-8
MC_Home	●	
MC_Stop	●	
MC_Halt	●	
MC_MoveAbsolute	●	
MC_MoveRelative	●	
MC_MoveAdditive	●	
MC_MoveSuperimposed	●	
MC_CamIn	●	
MC_CamOut	●	
MC_MoveVelocity	●	
MC_PositionProfile	●	
MC_VelocityProfile	●	
MC_AccelerationProfile	●	
MC_Jog	●	
MC_GearIn	●	
MC_GearOut	●	
MC_GearInPos	●	
MC_Phasing	●	
MC_Power	●	
MC_SetPosition	●	
MC_ReadParameter	●	
MC_WriteParameter	●	
MC_ReadBoolParameter	●	
MC_WriteBoolParameter	●	
MC_ReadActualPosition	●	
MC_ReadActualVelocity	●	
MC_ReadActualTorque	●	
MC_Reset	●	
MC_ReadStatus	●	
MC_ReadAxisError	●	
MC_CamTableSelect	●	
MC_TouchProbe	●	

Function Block	Model Type	
	AX-3	AX-8
MC_AbortTrigger	●	
MC_DigitalCamSwitch	●	
MC_Home_DML	●	
MC_Stop_DML	●	
MC_Halt_DML	●	
MC_MoveAbsolute_DML	●	
MC_MoveRelative_DML	●	
MC_MoveVelocity_DML	●	
MC_TorqueControl_DML	●	
MC_Power_DML	●	
MC_ReadParameter_DML	●	
MC_WriteParameter_DML	●	
MC_ReadBoolParameter_DML	●	
MC_WriteBoolParameter_DML	●	
MC_Reset_DML	●	
MC_ReadStatus_DML	●	
MC_ChangeAxisConfig_DML	●	
MC_ReinitDrive_DML	●	
DMC_MoveVelocityStopByPos	●	
DMC_TorqueControl	●	
DMC_VelocityControl	●	
DMC_MoveLinearAbsolute	●	
DMC_MoveLinearRelative	●	
DMC_MoveCircularAbsolute	●	
DMC_MoveCircularRelative	●	
DMC_GroupStop	●	
DMC_GroupHalt	●	
DMC_Home_P	●	
DMC_GroupEnable	●	
DMC_GroupDisable	●	
DMC_GroupReadStatus	●	
DMC_GroupReadError	●	
DMC_GroupReset	●	
DMC_CamReadTappetStatus	●	

Function Block	Model Type	
	AX-3	AX-8
DMC_CamReadTappetValue	●	
DMC_CamWriteTappetValue	●	
DMC_CamAddTappet	●	
DMC_CamDeleteTappet	●	
DMC_CamReadPoint	●	
DMC_CamWritePoint	●	
DMC_ChangeMechanismGearRation	●	
DMC_ReadMotionState	●	
DMC_AxesObserve	●	
DMC_PositionLag	●	
DMC_SetTorqueLimit	●	
DMC_SetSoftwareLimit	●	
DMC_CamKeyPointWrite	●	
DMC_TouchProbeCyclically	●	

A.1.3 By Letter

- A
  - MC_AbortTrigger
  - MC_AccelerationProfile
  - DMC_AxesObserve
- C
  - MC_CamIn
  - MC_CamOut
  - MC_CamTableSelect
  - MC_ChangeAxisConfig_DML
  - DMC_CamAddTappet
  - DMC_CamDeleteTappet
  - DMC_CamReadPoint
  - DMC_CamReadTappetStatus
  - DMC_CamReadTappetValue
  - DMC_CamWritePoint
  - DMC_CamWriteTappetValue
  - DMC_ChangeMechanismGearRation
  - DMC_CamKeyPointWrite
- D
  - MC_DigitalCamSwitch
- G
  - MC_GearIn
  - MC_GearInPos
  - MC_GearOut
  - DMC_GroupDisable
  - DMC_GroupEnable
  - DMC_GroupHalt
  - DMC_GroupReadError
  - DMC_GroupReadStatus
  - DMC_GroupReset
  - DMC_GroupStop
- H
  - MC_Halt
  - MC_Halt_DML
  - MC_Home
  - MC_Home_DML
  - DMC_Home_P
- J
  - MC_Jog
- M
  - MC_MoveAbsolute
  - MC_MoveAbsolute_DML
  - MC_MoveAdditive
  - MC_MoveRelative
  - MC_MoveRelative_DML
  - MC_MoveSuperimposed
  - MC_MoveVelocity
  - MC_MoveVelocity_DML
  - DMC_MoveCircularAbsolute
  - DMC_MoveCircularRelative
  - DMC_MoveLinearAbsolute
  - DMC_MoveLinearRelative
  - DMC_MoveVelocityStopByPos
- P
  - MC_Phasing
  - MC_PositionProfile
  - MC_Power
  - MC_Power_DML
  - DMC_PositionLag
- R
  - MC_ReadActualPosition
  - MC_ReadActualTorque
  - MC_ReadActualVelocity
  - MC_ReadAxisError
  - MC_ReadBoolParameter
  - MC_ReadBoolParameter_DML
  - MC_ReadParameter
  - MC_ReadParameter_DML
  - MC_ReadStatus
  - MC_ReadStatus_DML
  - MC_ReinitDrive_DML
  - MC_Reset

MC_Reset_DML

DMC_ReadMotionState

- S

MC_SetPosition

MC_Stop

MC_Stop_DML

DMC_SetTorqueLimit

DMC_SetSoftwareLimit

- T

MC_TouchProbe

MC_TorqueControl_DML

DMC_TorqueControl

DMC_TouchProbeCyclically

MC_WriteBoolParameter

- V

MC_VelocityProfile

DMC_VelocityControl

- W

MC_WriteBoolParameter

MC_WriteBoolParameter_DML

MC_WriteParameter

MC_WriteParameter_DML

A.2 Data Type: Enumeration and Structure

- The data types listed below are Enumeration type:

Data Type	Value	Description	Applicable Function Block Instruction and its Interface
MC_SOURCE	0:mcCommandedValue 1:mcActualValue	0: The commanded value of instruction 1: The actual value of motion axis	Function block: DMC_ReadMotionState Interface: *Source*
MC_StartMode	0:absolute 1:relative 2:ramp_in 3:ramp_in_pos 4:ramp_in_neg	0: Absolute mode 1: Relative mode 2:Ramp in mode 3: Positive Ramp in 4: Negative Ramp in mode	Function block: MC_CamIn Interface: *StartMode*
SMC_CAM TAPPETTYPE	0:TAPPET_pos 1:TAPPET_all 2:TAPPET_neg	0: When pass in a positive direction 1: When pass in both direction 2: When pass in a negative direction	Function block: MC_CamIn. Tappets.pTaps Interface: *ctt*
SMC_CAM TAPPET ACTION	0:TAPPETACTION_on 1:TAPPETACTION_off 2:TAPPETACTION_inv 3:TAPPETACTION_time	0: Switches ON 1: Switches OFF 2: Inverts 3: Switches on after a delay for a certain time period.	Function block: MC_CamIn. Tappets.pTaps Interface: *cta*
MC_TAPPET MODE	0:tp_mode_auto 1:tp_mode_demandposition 2:tp_mode_actualposition	0: Auto mode 1: Use set values 2: Use actual values	Function block: MC_DigitalCamSwitch Interface: *TappetMode*
DMC_BUFFER_ MODE	0:aborting 1:buffered 2:blending_low 3:blending_previous 4:blending_next 5:blending_high	0: Any ongoing motion is aborted. 1: Start FB after current move has finished. 2: The velocity is blended with lowest velocity of both FBs 3: The velocity is blended with the velocity of the first FB. 4: The velocity is blended with velocity of the second FB. 5: The velocity is blended with highest velocity of both FBs	Function block: DMC_MoveLinearAbsolute DMC_MoveLinearRelative DMC_MoveCircularAbsolute DMC_MoveCircularRelative DMC_GroupHalt Interface: BufferMode
DMC_GROUP_TRANSITION_MODE	0:None 1:Overlap	0: No effect 1: Overlap without considering time for deceleration.	Function block: DMC_MoveLinearAbsolute DMC_MoveLinearRelative DMC_MoveCircularAbsolute DMC_MoveCircularRelative Interface: TransitionMode
DMC_CIRC_ PLANE	0:XY_plane 1:YZ_plane 2:ZX_plane	0: The circle is parallel to the XY plane. 1: The circle is parallel to the YZ plane. 2: The circle is parallel to the ZX plane.	Function block: DMC_MoveCircularAbsolute DMC_MoveCircularRelative Interface: CircPlane

Data Type	Value	Description	Applicable Function Block Instruction and its Interface
DMC_CIRC_ MODE	0:radius 1:center 2:border	0: Defines radius of a circle. 1: Defines a center point of a circle. 2: Defines a point on the circle which is crossed on the path from the starting to the end point.	Function block: DMC_MoveCircularAbsolute DMC_MoveCircularRelative Interface: CircMode
DMC_CIRC_ PATHCHOICE	0:CLOCKWISE 1:COUNTER_CLOCKWISE	0: Clockwise 1: Counterclockwise	Function block: DMC_MoveCircularAbsolute DMC_MoveCircularRelative Interface: PathChoice
DMC_GROUP_S TATE	0:GroupDisabled 1:GroupStandby 2:GroupMoving 3:GroupHoming 4:GroupStopping 5:GroupErrorstop	0: Group state is Disabled 1: Group state is Standby 2: Group state is Moving 3: Group state is Homing 4: Group state is Stopping 5: Group state is Errorstop	Function block: DMC_AXIS_GROUP_REF Interface: GroupState
DMC_GROUP_ RAMP_TYPE	0:Trapezoid 1:S_Curve	0: The velocity curve is trapezoidal. 1: The velocity curve is S Curve.	Function block: DMC_AXIS_GROUP_REF Interface: RampType

- The data types listed below are Structure type:

Data Type	Function Block	Definition
AXIS_REF_SM3*	Applied to MC_ / DMC_ function block.	This structure contains all the reqired data and parameters for axis motion.
DMC_AXIS_GROUP_REF*	Applied to DMC_Group function block.	This structure contains all the reqired data and parameters for group motion.
AXIS_REF_VIRTUAL_SM3	Applied to MC_ / DMC_ function block.	This structure contains all the reqired data and parameters for virtual axis motion.
TRIGGER_REF	MC_TouchProbe MC_AbortTrigger	Includes the trigger information ● The specified trigger channel ● The trigger condition and the mode (Triggered on the rising or falling edge of the trigger signal.)
MC_CAM_REF	MC_CamTableSelect	This structure contains information of the CAM table and points.

***Note**: Please refer to AX-3 Series operational manual for more details of struct type parameters.

A.3 Error Codes and Troubleshooting

A.3.1 For Synchronus Axes

When an error occurs, you can troubleshoot errors through error codes and the corresponding indicators. Please refer to **AX-3 Operational Manual** for more details of troubleshooting process.

The following table lists the error codes and the contents of the errors:

Error code	Description	Contents	Corrective Action
0x00000	SMC_NO_ERROR	No error messages.	
0x00001	SMC_DI_GENERAL_ COMMUNICATION_ ERROR	Communication error.	Make sure the servo's network cable is properly plugged, then reset EtherCAT master and execute MC_Reset.
0x00002	SMC_DI_AXIS_ERROR	Axis error.	Check the error information and troubleshoot the error according to each servo's user manual, then execute MC_Reset.
0x00003	SMC_DI_FIELDBUS_ LOST_SYNCRONICITY	Loss of synchronicity.	Execute SMC3_ReinitDrive. If the error occurs frequently, please refer to Task max cycle time and increase EtherCAT DC time.
0x0000A	SMC_DI_SWLIMITS_ EXCEEDED	Software limit errors.	Execute MC_Reset and run reversely away from the limit.
0x0000B	SMC_DI_HWLIMITS_ EXCEEDED	Hardware limit errors.	Execute MC_Reset.
0x0000C	SMC_DI_LINEAR_AXIS_ OUTOFRANGE	Incremental position of a linear axis is out of range.	Execute MC_Reset
0x0000D	SMC_DI_HALT_OR_ QUICKSTOP_NOT_ SUPPORTED	Not support Halt or Quickstop state of drivers.	Execute MC_Reset
0x00010	SMC_DI_ POSITIONLAGERROR	Excessive position error.	Execute MC_Reset
0x00011	SMC_DI_HOMING_ ERROR	Homing error occurs.	Execute MC_Reset
0x00014	SMC_REGULATOR_OR_ START_NOT_SET	The motion FB can not be executed under the current axis state.	Enable the servo and execute MC_Reset, then re-execute the motion FB.
0x00015	SMC_WRONG_ CONTROLLER_MODE	The axis is under wrong controller mode.	Execute SMC_SetControllerMode to switch the axis to the proper controller mode.
0x00019	SMC_INVALID_ACTION_ FOR_LOGICAL	Invalid action for logical axes.	Do not perform inproper operation to logical axes, such as powering on logical axes.
0x0001E	SMC_FB_WASNT_ CALLED_DURING_ MOTION	Function blocks can not be called in movement state.	Please execte FBs in bus cycle task.
0x0001F	SMC_AXIS_IS_NO_AXIS_R EF	AXIS_REF variable type errors.	The pointer must points to the register.
0x00020	SMC_AXIS_REF_ CHANGED_DURING_ OPERATION	AXIS_REF variables have been changed while the modules being activated.	Execute MC_Reset and do not change the axis input of the function block.
0x00021	SMC_FB_ACTIVE_AXIS_ DISABLED	Execte servo off while axis is in motion.	Power on the servo and execute MC_Reset.
0x00022	SMC_AXIS_NOT_READY_F OR_MOTION	The motion instruction can not be executed under the current axis state.	As a result that the axis is not able to be controlled, please check if the state is power ON or an error exists, then enables the axis or execute MC_Reset depending on the situation.
0x00023	SMC_AXIS_ERROR_ DURING_MOTION	Error occurs during motion.	Please refer to the servo user manual to check on the error information, then execute

Error code	Description	Contents	Corrective Action
			MC_Reset.
0x00028	SMC_VD_MAX_ VELOCITY_EXCEEDED	Exceeds the maximum velocity limit fMaxVelocity.	Troubleshoots with MC_Reset.
0x00029	SMC_VD_MAX_ ACCELERATION_ EXCEEDED	Exceeds the maximum acceleration limit fMaxAcceleration.	Troubleshoots with MC_Reset.
0x0002A	SMC_VD_MAX_ DECELERATION_ EXCEEDED	Exceeds the maximum deceleration limit fMaxDeceleration.	Troubleshoots with MC_Reset.
0x00032	SMC_3SH_INVALID_ VELACC_VALUES	Invalid setting values of velocity or acceleration.	Please insert the value of velocity or acceleration again and then re-execute the function block.
0x00033	SMC_3SH_MODE_ NEEDS_HWLIMIT	The current mode needs the hardware limit to be activated.	bIgnoreHWLimit should not be True in the current mode. Please select the proper mode.
0x00046	SMC_SCM_NOT_ SUPPORTED	The mode is not supported.	Device does not support this mode, please re-select the proper mode then activate the function block.
0x00047	SMC_SCM_AXIS_IN_ WRONG_STATE	The controller mode can notbe changed in the current state.	Use MC_Reset to troubleshoot the error.
0x00048	SMC_SCM_ INTERRUPTED	SMC_SetControllerMode is interrupted by MC_Stop or errorstop.	Please reactivate the function block.
0x0004B	SMC_ST_WRONG_ CONTROLLER_MODE	The axis is under the wrong controller mode.	Use MC_Reset to troubleshoot the error.
0x00050	SMC_RAG_ERROR_ DURING_STARTUP	Error occurs when the axis group is activated.	Make sure the confogration is normal and re-execute SMC3_ReinitDrive.
0x00051	SMC_RAG_ERROR_AXIS_NOT_INITIALIZED	The axis is not in the required state.	SMC3_ReinitDrive can not be executed when EtherCAT Master is in Initial state.
0x00055	SMC_PP_WRONG_AXIS_TYPE	Virtual axes or logical axes are not supported by the function block	SMC3_PersistPosition can not be used on the virtual axis.
0x00056	SMC_PP_NUMBER_OF_ ABSOLUTE_BITS_ INVALID	Invalid absolute bits, which must be within 8~32 bits.	The value input to usiNumberOfAbsoluteBits of SMC3_PersistPositionSingleturn is incorrect, please re-enter the value.
0x0005A	SMC_CGR_ZERO_ VALUES	Invalid value.	Change the values of dwRatioTechUnitsDenomand to non-zero values and then re-execute the function block.。
0x0005B	SMC_CGR_DRIVE_ POWERED	The gear ratio parameters of the drive can not be modified when it is under controlled.	Make the axis enter Disable state, then re-execute the function block.
0x0005C	SMC_CGR_INVALID_ POSPERIOD	Invalid position period (less than or equal to 0, or exceeds half the width of the band)	When iMovementType = 0, fPositionPeriod is set to a value greater than zero and smaller than half the value of dwBusBandWidth.
0x0005D	SMC_CGR_POSPERIOD_NOT_INTEGRAL	The increment of the period is not integral and the case of modulo values is completed by the drive.	After modifying the parameters of fPositionPeriod, re-execute the function block.

Error code	Description	Contents	Corrective Action
0x0006E	SMC_P_FTASKCYCLE_ EMPTY	There's no cycle informationin the axis.(fTaskCycle = 0)	Change the value of TaskCycle into a non-zero value.
0x00078	SMC_R_NO_ERROR_TO_RESET	There's no errors after using MC_Reset.	Before execute the function block, check if there're any errors in the specified axis.
0x0007A	SMC_R_ERROR_NOT_ RESETTABLE	The error is not resettable.	Before reactivate MC_Reset, please make sure all errors in the drive have been removed.
0x00083	SMC_RP_REQUESTING_ERROR	Please check the error code output by the FB ReadDriveParameter if an error occurs when attempts to communicate to the drive.	The OD you're trying to access does not exist, please confirm the correct OD input.
0x00084	SMC_RP_DRIVE_ PARAMETER_NOT_ MAPPED	The parameter is not mapped to a specific drive.	The parameter you're trying to access does not exist.
0x0008D	SMC_WP_SENDING_ ERROR	Error code to the FB WriteDriveParameter.	The OD you're writing does not exist, please confirm the correct OD input.
0x0008E	SMC_WP_DRIVE_ PARAMETER_NOT_ MAPPED	Enter a parameter number of a non-existent axis.	The written parameter does not exist.
0x000AA	SMC_H_AXIS_WASNT_ STANDSTILL	The axis is not in standstill state.	Re-execute the FB after the axis enters standstill state.
0x000AB	SMC_H_AXIS_DIDNT_ START_HOMING	Errors occur while homing.	Before execute SMC3_ReinitDrive, please make sure the drive you're using is complied with standards and there's no existing error.
0x000AC	SMC_H_AXIS_DIDNT_ ANSWER	The drive does not answer after the homing is completed.	Before execute SMC3_ReinitDrive, please make sure the drive you're using is complied with standards and there's no existing error.
0x000AE	SMC_H_AXIS_IN_ ERRORSTOP	The homing mode can not be executed as a result because the drive is in errorstop state.	Re-execute the FB after the axis leaving ErrorStop state.
0x000B5	SMC_MS_INVALID_ ACCDEC_VALUES	Invalid setting value of velocity or acceleration.	Re-execute the FB after correcting the input value of "Deceleration".
0x000B7	SMC_MS_AXIS_IN_ ERRORSTOP	Drive in operating state Errorstop.	Re-execute the FB after the axis leaving ErrorStop state.
0x000B8	SMC_BLOCKING_MC_ STOP_WASNT_CALLED	Please set Execute to falling edge when the axis is locked and MC_Stop can not ba called.	Check the input setting of MC_Stop and re-execute the FB.
0x000B9	SMC_MS_AXIS_ ALREADY_STOPPING	A stop cannot be aborted while executing MC_Stop.	Re-execute the FB after the axis leaving Stopping state.
0x000C9	SMC_MA_INVALID_ VELACC_VALUES	Invalid values of velocity or acceleration.	Re-execute the FB after correcting the input values of "Velocity", "Acceleration", "Deceleration" and "Jerk".
0x000E2	SMC_MR_INVALID_ VELACC_VALUES	Invalid values of velocity or acceleration.	Re-execute the FB after correcting the input values of "Velocity", "Acceleration", "Deceleration" and "Jerk".
0x000FB	SMC_MAD_INVALID_ VELACC_VALUES	Invalid values of velocity or acceleration.	Re-execute the FB after correcting the input values of "VelocityDiff", "Acceleration", "Deceleration" and "Jerk".
0x00114	SMC_MSI_INVALID_ VELACC_VALUES	Invalid values of velocity or acceleration.	Re-execute the FB after correcting the input values of "VelocityDiff", "Acceleration" and "Deceleration".

Error code	Description	Contents	Corrective Action
0x00116	SMC_MSI_INVALID_ EXECUTION_ORDER	An error will occur if activates the second MC_MoveSuperimposed while the frst one is still being executed.	Re-execute the FB after the execution of first MC_MoveSuperimposed is completed.
0x0012D	SMC_MV_INVALID_ ACCDEC_VALUES	Invalid values of velocity or acceleration.	Re-execute the FB after correcting the input values of “Velocity”, “Acceleration”, “Deceleration” and “Jerk”.
0x0012E	SMC_MV_DIRECTION_ NOT_APPLICABLE	Direction = shortest/fastest is not applicable.	After correcting the input value of “Dierction” to be not in shortest / fastest state, re-execute the function block.
0x00145	SMC_PP_ARRAYSIZE	Incorrect array size.	Please re-enter the correct ArraySize, then re-execute the function blocks.
0x00146	SMC_PP_STEP0MS	Delta_time is not allowed to be set to 0.	Please re-enter the correct Delta_Time, then re-execute the function blocks.
0x0015E	SMC_VP_ARRAYSIZE	Incorrect array size	Please re-enter the correct ArraySize, then re-execute the function blocks.
0x0015F	SMC_VP_STEP0MS	Delta_time is not allowed to be set to 0.	Please re-enter the correct Delta_Time, then re-execute the function blocks.
0x00177	SMC_AP_ARRAYSIZE	Incorrect array size.	Please re-enter the correct ArraySize, then re-execute the function blocks.
0x00178	SMC_AP_STEP0MS	Delta_time is not allowed to be set to 0.	Please re-enter the correct Delta_Time, then re-execute the function blocks.
0x00190	SMC_TP_ TRIGGEROCCUPIED	The trigger has been activated.	Correct TriggerInput.bActive back to False, then re-execute the function block.
0x00191	SMC_TP_COULDNT_SET_ WINDOW	The driver interface does not support Mask function.	The specified devices do not support Window functions, please turn off Window functions to re-execute the function block.
0x0019A	SMC_AT_ TRIGGERNOTOCCUPIED	Triggering has been reset.	Check if MC_TouchProbe is executed and the axis position has not been captured then re-activate the function block.
0x001AA	SMC_MCR_INVALID_ VELACC_VALUES	Invalid values of velocity or acceleration.	Correct the input value of “Velocity”, “EndVelocity”, “Deceleration” and “Jerk”, then re-execute the function block.
0x001C3	SMC_MCA_INVALID_ VELACC_VALUES	Invalid values of velocity or acceleration.	Correct the input value of “Velocity”, “EndVelocity”, “Deceleration” and “Jerk”, then re-execute the function block.
0x001C5	SMC_MCA_DIRECTION_ NOT_APPLICABLE	Cannot set the shortest distance.	After correcting the input value of “EndVelocityDirection” to be not in shortest / fastest state, re-execute the function block.
0x001DB	SMC_SDL_INVALID_ AXIS_STATE	SMC_ChangeDynamic Limits can only be called in standstill or power_off state.	Check if the axis is in power_off or standstill state and then re-activate the function block.
0x001DC	SMC_SDL_INVALID_ VELACC_VALUES	Invalid values of velocity, acceleration, deceleration and jerk.	After fixing the input value if "fMaxVelocity" 、"fMaxAcceleration", "fMaxDeceleration" and "fMaxJerk", re-execute the function block.
0x00258	SMC_CR_NO_TAPPETS_ IN_CAM	There’re no tappets set in the CAM.	Please set tappets in the cam table and then re-execute the function block.

Error code	Description	Contents	Corrective Action
0x00259	SMC_CR_TOO_MANY_ TAPPETS	The number of tappet groupID exceeds MAX_NUM_TAPPETS	As a result of too many tappets in the cam table, you have to modify the number before re-executing the function block.
0x00271	SMC_CI_NO_CAM_ SELECTED	No cam is selected.	Please enter the correct value given by MC_CamTableSelect after it's successfully executed to the input "CamTableID" and then re-execute the function block.
0x00272	SMC_CI_MASTER_OUT_ OF_SCALE	The master exceeds the valid scale.	1. Execute MC_Reset to make the axis back to standstill state and check the inputs of MC_CamTableSelect. 2. Make sure that the cam master position, which is calculated by "Periodic" /"MasterAbsolute" of MC_CamTableSelect and "MasterOffset" of MC_CamIn, is in the range of master scale on the cam table before you re-execute the function block.
0x00273	SMC_CI_RAMPIN_ NEEDS_VELACC_ VALUES	The value of Velocity and acceleration must be set in the function of ramp_in.	1. Execute MC_Reset to make the axis back to standstill state and check the StartMode input. 2. When "StartMode" is set to "ramp_in"/ "ramp_in_pos"/"ramp_in_neg", the input values of "VelocityDiff"/"Acceleration"/ "Deceleration" need to be non-zero. Then you can re-execute the function block.
0x00274	SMC_CI_SCALING_ INCORRECT	Scaling variables fEditor/TableMasterMin/Max are not correct	1. Execute MC_Reset to make the axis back to standstill state and check the inputs. 2. Make sure that the max values must be bigger than the min values in fEditor / fTable while using a cam table not in "XYVA" format. Then you can re-execute the function block.
0x00275	SMC_CI_TOO_MANY_ TAPPETS_PER_CYCLE	Activate too many tappets in the same period.	Modify the tappets on the cam table and make sure that there're no too many tappets gathering on the same position. After download the cam table again, re-execute the function block.
0x00280	SMC_CB_NOT_ IMPLEMENTED	The selected cam format is not executed.	Modify the cam table format to the one supported by the function block, then re-execute the function block.(Currently only support "XYVA" format)
0x002A3	SMC_GI_RATIO_DENOM	RatioDenominator = 0	Change the value of "RatioDenominator" to be non-zero and re-execute the function block.
0x002A4	SMC_GI_INVALID_ACC	Invalid value of acceleration.	1. Execute MC_Reset to make the axis back to standstill state and check the inputs. 2. Make sure the value of "Acceleration" is greater than zero, then re-execute the function block.
0x002A5	SMC_GI_INVALID_DEC	Invalid value of deceleration.	1. Execute MC_Reset to make the axis back to standstill state and check the inputs. 2. Make sure the value of "Deceleration" is greater than zero, then re-execute the function block.
0x002A6	SMC_GI_MASTER_ REGULATOR_CHANGED	The master state (Enable/Disable) is changed without the permission.	Execute MC_Reset to make the axis back to standstill state and re-execute the function block.

Error code	Description	Contents	Corrective Action
0x002A7	SMC_GI_INVALID_JERK	Invalid jerk value.	1. Execute MC_Reset to make the axis back to standstill state and check the inputs. 2. Make sure the value of "Jerk" is greater than zero, then re-execute the function block.
0x002D5	SMC_PH_INVALID_ VELACCDEC	Invalid values of velocity, acceleration and deceleration.	Make sure the values of "Velocity", "Acceleration" and "Deceleration" are non-zero before re-execute the function block.
0x002EE	SMC_NO_CAM_REF_ TYPE	The chosen cam type is not MC_CAM_REF	Correct the input variable to the correct variable in "MC_CAM_REF" type.
0x002EF	SMC_CAM_TABLE_ DOES_NOT_COVER_ MASTER_SCALE	The curve data does not include the master scale, xStart and xEnd, on the CamTable.	Correct the values of "xStart" and "xEnd" to make these two values are included in the master scale.
0x002F0	SMC_CAM_TABLE_ EMPTY_MASTER_ RANGE	There’s no master range on the cam table.	Fix the "xStart" and "xEnd" on the cam table to make the "xEnd" value greater than the value of “xStart”.
0x002F2	SMC_CAM_TABLE_ INVALID_SLAVE_ MINMAX	Invalid min/ max values of the slave axis on the cam table.	Make sure that the values of fTableSlaveMin and fTableSlaveMax are not equal before you re-execute the function block.
0x00307	SMC_GIP_MASTER_ DIRECTION_CHANGE	The master axis changes itsdirection while being scychronizing with the slave axis.	Execute MC_Reset to make the axis back to standstill state and re-execute the FB. At the same time, do not reverse the master direction when "StartSync" is True.
0x00308	SMC_GIP_SLAVE_ REVERSAL_CANNOT_BE_AVOIDED	AvoidReversal is set but cannot avoid the slave axis to be reversed.	Adjust the input values of "MasterSyncPosition", "SlaveSyncPosition" and "MasterStartDistance" as well as the velocity of master and slave axis after being coupled. Then re-execute the function block.。
0x00309	SMC_GIP_AVOID_ REVERSAL_FOR_ FINITE_AXIS	AvoidReversal cannot be configured while using linear axes.	Change the "Axis type" of slave to "Modulo"(Need to re-download) or set the input "AvoidReversal" to False, then re-execute the function block.
0x186A0	DMC_TPC_INVALID_PDO_ MAPPING	PDO mapping error.	Please do not configure Touch probe function(60B8h) in PDO.
0x186A1	DMC_TPC_TRIGGER OCCUPIED	Trigger has been created.	Please do not execute the function block with MC_TouchProbe, which has been executed.
0x186A2	DMC_TPC_ETC_CO_ FIRST_ERROR	SDO read-write error	Invalid SDO command, please check the related configuration.
0x186A3	DMC_TPC_ETC_CO_ OTHER_ERROR	Communication error	Cannot find the corresponding master station, please check the master status first.
0x186A4	DMC_TPC_ETC_CO_ DATA_OVERFLOW	Communication error	The size of SDO is too large to be sent. Please re-execute the FB after modification.
0x186A5	DMC_TPC_ETC_CO_ TIMEOUT	Communication error	SDO time outs. Please check if there’s a corresponding OD to the servo.
0x186A8	DMC_TPC_ECAT_ MASTER_DISABLE	Communication error	Master initialization failed. Please check the status of the master station.
0x187CC	DMC_CRTS_TAPPETID_ VALUE_OUTOFRANGE	The value of track ID of the tappet is set out of range.	Re-execute the FB after correcting Track ID.
0x187D2	DMC_CRTV_TAPPETID_ VALUE_OUTOFRANGE	The value of track ID of the tappet is set out of range.	Re-execute the FB after correcting Track ID.
0x187D3	DMC_CRTV_NO_ TAPPETID	The track ID to read does not exeist.	Re-execute the FB after checking the tappet inputs.
0x187D4	DMC_CRTV_NO_ TAPPETS_IN_CAM	There’s no tappets set in the cam table.	Re-execute the FB after adding new tappets.

Error code	Description	Contents	Corrective Action
0x187DA	DMC_CWTV_INVALID_TAPPETID	Invalid Track ID.	Re-execute the FB after correcting Track ID.
0x187DB	DMC_CWTV_INVALID_MASTER_POS	Invalid master position.	Please correct the input of master position, then re-execute the FB.
0x187DC	DMC_CWTV_CAM_TABLE_NUM_EXCEED_LIMIT	The number of cam table exceeds the limit.	The limit has been reached. Cannot write in more tappets.
0x187DD	DMC_CWTV_TAPPETID_NOT_FOUND	The track ID to modify does not exist.	Re-execute the FB after correcting Track ID.
0x187DE	DMC_CWTV_TAPPET_NUM_EXCEED_LIMIT	The number of tappets exceeds the limit.	Re-execute the FB after checking the tappet number.
0x187E4	DMC_CAT_INVALID_MASTER_POS	The user-defined master position is out of range.	Re-execute the FB after correcting the master position.
0x187E5	DMC_CAT_CAM_TABLE_NUM_EXCEED_LIMIT	The number of cam table exceeds the limit.	The limit has been reached. Cannot write in more tappets.
0x187E6	DMC_CAT_TAPPET_NUM_EXCEED_LIMIT	The number of tappets exceeds the limit.	Re-execute the FB after checking the tappet number.
0x187E7	DMC_CAT_NO_TAPPET_TO_BE_ADDED	No tappet action set in the input variable.	There’re no newly-added tappets in the input data. Please confirm that either PositiveMode or NegativeMode is not set to TAPPETACTION_none before re-execute the function block.
0x187ED	DMC_CDT_NO_TAPPETS_IN_CAM	There’s no tappet in the tappet table.	Re-execte the FB after specifying a tappet table which has tappets in it.
0x187EE	DMC_CDT_CAM_TABLE_NUM_EXCEED_LIMIT	The number of cam table exceeds the limit.	The limit has been reached. Cannot write in more tappets.
0x187F4	DMC_CRP_INVALID_POINTNUM	Invalid point number.	Check if the point number of specified data is more than the point number of cam data. Re-execute the FB after modification.
0x187FA	DMC_CWP_INVALID_POINTNUM	Invalid point number.	Check if the point number of specified data is more than the point number of cam data. Re-execute the FB after modification.
0x187FB	DMC_CWP_INVALID_MASTERPOS	Invalid master poition.	Please check if the master position of data point to be modified exceeds the master position of thr front and back point. Re-execute the FB after modification.
0x1882E	DMC_GM_NO_ERROR_TO_RESET	There’s no error to be reset.	Re-execute DMC_GroupReset when an error occurs in the axis group.
0x1882F	DMC_GM_DRIVE_DOESNT_ANSWER	One or more axes in the group does not execute the reset action.	After the communication status of the axis is back to normal, re-execute the FB. (DFB_ResetECATMaster/DFB_ResetECATSlave)
0x18830	DMC_GM_ERROR_NOT_RESETTABLE	Error is not resettable.	Remove the error in axis group (Modify parameter settings/ check on a normal axis path) before download the program once again.
0x18831	DMC_GM_DRIVE_DOESNT_ANSWER_IN_TIME	Communication timeout	After the communication status of the axis is back to normal (DFB_ResetECATMaster/DFB_ResetECATSlave), re-execute the FB.

Error code	Description	Contents	Corrective Action
0x18832	DMC_GM_CANNOT_ RESET_ COMMUNICATION_ ERROR	Communication error cannotbe reset.	After the communication status of the axis is back to normal (DFB_ResetECATMaster/DFB_ResetECATSlave), re-execute the FB.
0x18833	DMC_GM_AXIS_GROUP_RESET_FAILED	Fail to reset the axis group.	Remove the error in axis group (Modify parameter settings/ check on a normal axis path) before download the program once again.
0x18839	DMC_GM_LINEAR_AXIS_MAPPING_ERROR	Command a non-zero displacement to an axis, which does not exist.	Execute MC_GroupReset to make the axis group back to GroupStandby state. Then check the parameter setting and the input position of axis group so as to make sure the existing displacement has been mappeded to an appointed axis.
0x1883F	DMC_GM_CIRCULAR_ AXIS_MAPPING_ERROR	Command a non-zero displacement to an axis, which does not exist, in a circular movement.	Execute MC_GroupReset to make the axis group back to GroupStandby state. Then check the parameter setting and the input position of axis group so as to make sure the existing displacement has been mappeded to an appointed axis.
0x18840	DMC_GM_HELIX_AXIS_ MAPPING_ERROR	Command a non-zero displacement to an axis, which does not exist, in a helical movement.	Execute MC_GroupReset to make the axis group back to GroupStandby state. Then check the parameter setting and the input position of axis group so as to make sure the existing displacement has been mappeded to an appointed axis.
0x18841	DMC_GM_CIRCLE_ DISTANCE_LARGER_ THAN_DIAMETER	Under DMC_CIRC_MODE. radius mode, the distance between the start and end point is larger than the diameter.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. While using DMC_CIRC_MODE.radius, the input value of radius must be larger than half of the distance between the start and end point. 3. Re-execute the function block.
0x18842	DMC_GM_CIRCLE_ START_AND_ENDPOINT_EQUAL	Under DMC_CIRC_MODE. radius / DMC_CIRC_ MODE.border mode, the start point and the end point are at the same position.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. While using DMC_CIRC_MODE.radius / DMC_CIRC_MODE.border the input value of radius must be larger than half of the distance between the start and end point. 3. Re-execute the function block.
0x18843	DMC_GM_CIRCLE_ COLLINEAR_POINTS	Under DMC_CIRC_MODE. border mode, three points are defined to lie on a same line.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. While using DMC_CIRC_MODE.border, start point, end point and assist point should not be set on the same line. 3. Re-execute the function block.

Error code	Description	Contents	Corrective Action
0x18844	DMC_GM_CIRCLE_ CENTER_NOT_ON_ BISECTOR	Under DMC_CIRC_MODE. center mode, the center of a circle is not on the bisector line.	1. Execute MC_GroupReset to make the group state back to GroupStandby. Make sure that the center must locates on the bisector line between the start and end point. 2. Re-execute the function block.
0x18845	DMC_GM_CIRCLE_ RADIUS_ZERO	Under DMC_CIRC_MODE. radius mode, the radius is zero.	1. Make sure the radius is not 0 while using DMC_CIRC_MODE.radius mode. 2. Re-execute the function block.
0x1884B	DMC_GM_CONTINUE_ WRONG_POSITION	The current position is not the start position recorded in continue data.	1. Move the axis group to the position recorded in Continue Data. (DMC_AXIS_GROUP_REF.ContinuePos) 2. Re-execute the function block.
0x1884C	DMC_GM_CONTINUE_ DATA_NOT_WRITTEN	ContinueData is not written.	After confirming there’s Continue Data in the axis group (DMC_AXIS_GROUP_REF.bContinueDataWriten), then execute DMC_GroupContinue.
0x18852	DMC_GM_NO_AXIS_IN_ AXIS_GROUP	There’re no axes in the axis group.	At least one axis must be specified in the parameter setting of axis group before re-execute the function block.
0x18853	DMC_GM_SINGLE_AXIS_E RROR	Axis error occurs in the axis group.	1. After troubleshoot the error, execute MC_GroupReset to make the group state back to GroupStandby, while each axis leaves errorstop state. 2. Re-execute the function block.
0x18854	DMC_GM_AXIS_NOT_ READY_FOR_MOTION	One or more axes in the group are not ready for motion.	1. Execute MC_GroupReset to make the group state back to GroupStandby, while each axis leaves errorstop state. 2. Make sure that each axis has been successfully powered on and entered standstill state. 3. Re-execute the function block.
0x18855	DMC_GM_AXIS_LIMIT_ VIOLATED	One or more limits for an axis are violated.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure that the position, velocity, acceleration and jerk of each axis do not exceed the limits. 3. Re-execute the function block.
0x18856	DMC_GM_AXIS_GROUP_W RONG_STATE	Axis group is in wrong status.	Make sure the axis group is under the proper state and ready to be executed before execute the function block.
0x18857	DMC_GM_AXIS_GROUP_A XIS_IN_DIFFERENT_ TASK	Some axes in the group and the axis group itself are not in the same task.	Correct the settings of the axis and the group so as to make both bus cycle tasks are appointed to the same task...

Error code	Description	Contents	Corrective Action
0x18858	DMC_GM_INVALID_VEL_ACC_DEC_JERK	Invalid values of velocity, acceleration, deceleration and jerk.	1. Adjust the values to be reasonable and non-zero. 2. Re-execute the function block.
0x18859	DMC_GM_INVALID_BUFFER_MODE	Invalid buffer mode.	1. Change to a supported buffer mode. 2. Re-execute the function block.
0x1885A	DMC_GM_CMD_ABORTED_DUE_TO_ERROR	Command is aborted due to an error.	1. Troubleshoot the error. 2. Execute MC_GroupReset to make the group state back to GroupStandby. 3. Re-execute the function block.
0x1885B	DMC_GM_TRANSITIONING_FROM_SINGLE_AXIS_MOVEMENT_NOT_SUPPORTED	Transitioning from the single-axis movement is not spported.	1. 執 Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure each axis is back to standstill. 3. Re-execute the function block.
0x1885C	DMC_GM_AXIS_GROUP_VELOCITY_EXCEED_LIMIT	The velocity of axis group exceeds the limit set in the parameter setting.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure the group velocity does not exceed the limit set in the parameter setting. 3. Re-execute the function block.
0x1885D	DMC_GM_AXIS_GROUP_ACCELERATION_EXCEED_LIMIT	The acceleration of axis group exceeds the limit set in the parameter setting.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure the group acceleration does not exceed the limit set in the parameter setting. 3. Re-execute the function block.
0x1885E	DMC_GM_AXIS_GROUP_DECELERATION_EXCEED_LIMIT	The deceleration of axis group exceeds the limit set in the parameter setting.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure the group deceleration does not exceed the limit set in the parameter setting. 3. Re-execute the function block.
0x1885F	DMC_GM_AXIS_GROUP_JERK_EXCEED_LIMIT	The jerk of axis group exceeds the limit set in the parameter setting.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure the group jerk does not exceed the limit set in the parameter setting. 3. Re-execute the function block.

Error code	Description	Contents	Corrective Action
0x18860	DMC_GM_AXIS_GROUP_PLANNING_ERROR	Axis group planning error.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure the parameters set for the motion instruction are reasonable for planning paths. 3. Re-execute the function block.
0x18861	DMC_GM_AXIS_GROUP_MOVE_ERROR	Axis group move error.	1. Execute MC_GroupReset to make the group state back to GroupStandby. 2. Make sure the parameters set for the motion instruction are reasonable for planning paths. 3. Re-execute the function block.
0x18862	DMC_GM_CMD_BUF_ FULL	Command buffer is full.	1. Make sure there’s still some space in the command buffer. 2. Re-execute the function block.
0x18881	DMC_GM_AXIS_GROUP_INIT_FAILED	Axis group initialization failed.	1. Please use the axis group in the device tree as the input to the instruction. 2. Re-execute the function block.
0x18882	DMC_GM_INVALID_AXIS_IN_AXIS_GROUP	Invalid axes in axis group.	1. Make sure all the axes specified in the parameter setting exist in the device tree. 2. Download the program again. 3. Re-execute the function block.
0x18883	DMC_GM_DUPLICATE_ AXIS_IN_AXIS_GROUP	Duplicated axes in axis group.	1. Make sure there’s no duplicated axis specified in the parameter setting. 2. Download the program again. 3. Re-execute the function block.
0x18884	DMC_GM_AXIS_ ALREADY_IN_OTHER_ ENABLED_AXIS_GROUP	Some axes have been already existed in another enabled axis group.	1. Make sure the specified axis does not exist in other enabled axis group or disable the axis group which has the axis in it. 2. Re-execute the function block.
0x18885	DMC_GM_AXIS_GROUP_INVALID_TASK_ CONFIGURATION	Task is not configured correctly.	1. Make sure that the setting values of bus cycle task meet the requirment. (Type:Cyclic，Interval:> 1ms) 2. Download the program again. 3. Re-execute the function block.
0x18886	DMC_GM_AXIS_GROUP_COUNT_REACH_LIMIT	The axis group count has reached the limit.	1. To activate more groups, please make sure the number of activated axis group is less than the max.value. 2. Re-execute the function block.

Error code	Description	Contents	Corrective Action
0x188B5	DMC_CKPW_WRITE AMOUNT_OUTOFRANGE	WriteAmount input error	Check and correct the input value of WriteAmount before execute the function block.
0x188B6	DMC_CKPW_INVALID_ MASTERPOS	Invalid master position.	Re-execute the FB after correcting the input of master position.
0x188B7	DMC_CKPW_INVALID_ ACC	Invalid acceleration.	Re-execute the FB after correcting the acceleration input value of master position.
0x188B8	DMC_CKPW_INVALID_ ACC_SETTING	Invalid acceleration setting.	Re-execute the FB after determining the velocity, acceleration and curve type.
0x188B9	DMC_CKPW_INVALID_ CURVE_TYPE_SETTING	Invalid curve type setting.	The input curve type is not supported. Re-execute the FB after correcting the curve type.
0x188BA	DMC_CKPW_SPLINE_ HAS_NO_BOUNDARY	Spine has no boundary.	Make sure there's boundary condition (Nature or Clamp) set for the the previous and the latter part of the selected curve "Spline", which the condition should be the same at the start and end of the boundary. Then re-execute the FB.
0x188BB	DMC_CKPW_CAM_IS_ WRITING_BY_OTHER_ FUNCTION	Failure to write CAM.	Check if the cam table you're currently using is being written by other FBs, then wait for the writing completedbefore you re-execute the FB.
0x188C5	DMC_HP_INVALID_ HOME_SPEED	Invalid home speed value.	Please set "Search for switch" and "Search for Z phase pulse" with non-zero values for the home speed setting on Pulse Axis configuration page.
0x188C6	DMC_HP_INVALID_ HOME_ACC_DEC	Invalid home acceleration or deceleration value.	Please set the homing acceleration and deceletation with non-zero values on Pulse Axis configuration page.
0x188C7	DMC_HP_INVALID_ HOME_POSITION	Invalid setting value of homeposition.	Set "lrPosiotion" to be in the rotary range of pulse axis. [0 ~ PulseAxis.Modulo Value]。
0x188C8	DMC_HP_AXIS_NOT_ PULSEAXIS	The input variable type is not set to be PulseAxis_REF.	After select "Pulse Axis" in IO Configuration, enter the IEC Object variable to the input "Axis" of FB DMC_Home_P.
0x188C9	DMC_HP_HOMING_ METHOD_RESERVED	Homing method is not supported by current version.	Check if the homing method is supported by the version you're currently using. Please refer to the specification document for mode modification.
0x188CA	DMC_HP_HOMING_ MOVEMENT_HW_LIMIT	Positive or negative limit signal is activated and axis cannot perform homing in this circumstances.	Check if the hardware limit signal you're using is supported by the current homing mode. Please refer to the specification document for changing the mode and hardware limit signal configuration.
0x188CB	DMC_HP_HOMING_AXIS_S TATE_NOT_STAND STILL	Axis state is not Standstill.	Confirm that DMC_Home_P is executed when the axis state is Standstill.
0x1896C	DMC_STL_WP_PARAM_ INVALID	Invalid parameter.	The input parameter is too large. Re-execute the FB after correcting the input parameter.

Error code	Description	Contents	Corrective Action
0x1896D	DMC_STL_WP_SENDING_ERROR	No corresponding OD or the OD is not allowed to be written.	No such error should occur while matching ASDA-A2-E to use. Please check if the servo you're currently using meets Cia402, or the function block cannot be executed.
0x1896E	DMC_STL_WP_DRIVE_ PARAMETER_NOT_ MAPPED	The input parameter number does not exist.	No such error should occur while matching ASDA-A2-E to use. Please check if the servo you're currently using meets Cia402, or the function block cannot be executed.
0x1896F	DMC_STL_WP_PARAM_CONVERSION_ERROR	Parameter conversion error.	No such error should occur while matching ASDA-A2-E to use. Please check if the servo you're currently using meets Cia402, or the function block cannot be executed.
0x1897A	DMC_SSWL_LIMIT_ SETTING_OPPOSITE	Negative limit input error.	Negative software limit is greater than positive software limit. Please correct the input limit before yu re-execute the FB.
0x1897B	DMC_SSWL_NEGPOS_ LIMT_EQUAL	Negative limit input error.	Negative software limit is equal to positive software limit. Please correct the input limit before yu re-execute the FB.
0x1898A	DMC_PL_INVALID_ POSITIONLAG	Invalid MaxPositionLag input.	The input value of fMaxPositionLag is negative, please correct the value before re-execute the FB.
0x1898B	DMC_PL_INVALID_ LAGCYCIES	Invalid SetActTimeLagCycles input.	The input value of SetActTimeLagCycles is negative, please correct the value before re-execute the FB.
0x18996	DMC_MVSBP_INVALID_DIRECTION	Invalid direction.	Only positive and negative direction are allowed, please correct the direction of movement before re-execute the FB.
0x18997	DMC_MVSBP_INVALID_PHASE	Invalid phase input.	RoundPhase/ StopPhase input error. Please correct the input parameters before re-execute the FB.
0x18998	DMC_MVSBP_AXIS_NOT_READY_FOR_MOTION	Slave axis is not ready for motion.	The slave is not under control. Please check if the target axis is powered on or in error, then enable the axis or execute MC_Reset depending on the situation.
0x18999	DMC_MVSBP_AXIS_ ERROR_DURING_ MOTION	Errors occur during motion.	Please check the error information. Refer to the corresponding servo's user manual to troubleshoot the error and execute MC_Reset.
0x1899A	DMC_MVSBP_ REGULATOR_OR_ START_NOT_SET	The motion control instruction cannot be executed under the current axis state.	After activating the servo, execute MC_Reset before re-execute the FB.

A.3.2 For Positioning Axis

When an error occurs, you can troubleshoot errors through error codes and the corresponding indicators. Please refer to **AX-3 Operational Manual** for more details of troubleshooting process.

The following table lists the error codes and the contents of the errors:

Error code	Description	Contents	Corrective Action
0x000000	SML_NO_ERROR	No error messages.	
0x000001	SML_DI_GENERAL_ COMMUNICATION_ ERROR	Communication error.	Make sure the slave's network cable is properly plugged, then execute DFB_ResetECATMaster to reset EtherCAT master and execute MC_ReinitDrive_DML again.
0x000002	SML_DI_AXIS_ERROR	Axis error.	Check the error informationand troubleshoot the error before you re-execute MC_Reset_DML.
0x000015	SML_WRONG_ OPMODE	Wrong controller mode.	The function block cannot be executed under the current mode. Please execute SMC_SetControllerMode first then switch to a proper axis mode.
0x000022	SML_AXIS_NOT_ READY_FOR_MOTION	Slave axis is not ready for motion.	The slave is not under control. Please check if the target axis is powered on or in error, then enable the axis or execute MC_Reset_DML depending on the situation.
0x000023	SML_MA_MR_ MODULO_ACT_POS_ NOT_MAPPED	The necessary parameter is missing in PDO.	Configure Actual Position (16#6064) in PDO.
0x000024	SML_MV_INVALID_ VELACCDEC_VALUES	Invalid setting values of velocity, acceleration and deceleration.	Use MC_Reset_DML to troubleshoot the error.
0x000050	SMC_RAG_ERROR_ DURING_STARTUP	Error occurs during startup.	Please check if the wiring is normal and re-execute MC_ReinitDrive_DML.
0x00005A	SML_CGR_ZERO_ VALUES	The input values of dwRatioTechUnitsDenom and iRatioTechUnitsNum cannot be zero.	Modify the values of dwRatioTechUnitsDenom and iRatioTechUnitsNum with non-zero values then re-execute the FB.
0x00005B	SML_CGR_AXIS_ POWERED	The gear ratio parameters cannot be modified when axis is under wrong state.	Re-execute the FB after the axis enters Disable state.
0x00005D	SML_CGR_MODUL OPERIOD_NOT_ INTEGRAL	Modulo period is not integral.	Re-execute the FB after correcting fModuloPeriodU parameters.
0x00005E	SML_CGR_ MOVEMENTTYPE_ INVALID	Wrong axis type.(Must be either linear axis or rotary axis)	Re-execute the FB after correcting iMovementType parameters.
0x00005F	SML_CGR_MODUL OPERIOD_NON_ POSITIVE	Modulo period is not positive.	Re-execute the FB after correcting fPositionPeriod parameters.
0x000060	SML_CGR_MODUL OPERIOD_TOO_ SMALL	Modulo period is too small.	Re-execute the FB after correcting fPositionPeriod parameters.
0x000061	SML_CGR_MODUL OPERIOD_TOO_ LARGE	Modulo period is too large.	Re-execute the FB after correcting fPositionPeriod parameters.
0x000078	SML_R_NO_ERROR_ TO_RESET	No error occurs after executing MC_Reset_DML.	Re-execute the FB after check if there're any errors in the specified axis.
0x00007A	SML_R_ERROR_NOT_RE SETTABLE	The error is not resettable.	Please check if the error in the slave has been removed, then re-execute MC_Reset_DML.

Error code	Description	Contents	Corrective Action
0x000083	SML_RP_REQUESTING_ERROR	No corresponding OD or the OD is not allowed to be read.	The target OD does not exist or not being allowed to be accessed. Please make sure the input OD is correct and can be retrieved.
0x000084	SML_RP_RCV_PARAM_CONVERSION_ERROR	A wrong SoftMotionLight parameter is created during the conversion between axis parameters and servo OD.	The accessed parameter does not exist.
0x00008D	SML_WP_SENDING_ERROR	No corresponding OD or the OD is not allowed to be written.	The target OD does not exist or not being allowed to be written. Please make sure the input OD is correct and writable.
0x00008E	SML_WP_TMT_PARAM_CONVERSION_ERROR	A wrong SoftMotionLight parameter is created during the conversion between axis parameters and servo OD.	The accessed parameter does not exist.
0x0000AA	SML_H_AXIS_WASNT_STANDSTILL	The axis is not in Standstill state.	Re-execute the FB after the axis enters Standstill state.
0x0000B7	SML_MS_AXIS_IN_ERRORSTOP	The drive is in Errorstop state. Cannot executeMC_Stop_DML	Re-execute the FB after the axis leaves ErrorStop state.
0x018801	DML_TC_INVALID_VALUES	Invalid input values.	Please correct the input values before re-execute the function block.
0x018802	DML_TC_FB_CONFLICT	FBs conflict. Torque mode has been executed.	Only one FB DMC_TorqueControl is allowed to be executed at a time.
0x018803	DML_TC_SDO_RW_FAIL	SDO read-write failure.	SDO read write failed. Re-execute the FB after the servo communication is back to normal.
0x018805	DML_TC_SCM_AXIS_IN_WRONG_STATE	The axis is in wrong state.	Use MC_Reset_DML to troubleshoot the error.
0x018807	DML_TC_AXIS_NOT_READY_FOR_MOTION	The axis is not ready for motion.	Power on the servo and re-execute the FB
0x018809	DML_TC_TIMEOUT_CHANGING_OPMODE	OP Mode changing failure.	Check if the control mode is supported by the servo.
0x0189AA	DML_MR_SDO_RW_FAIL	SDO read-write failure.	SDO read write failed. Re-execute the FB after the servo communication is back to normal.

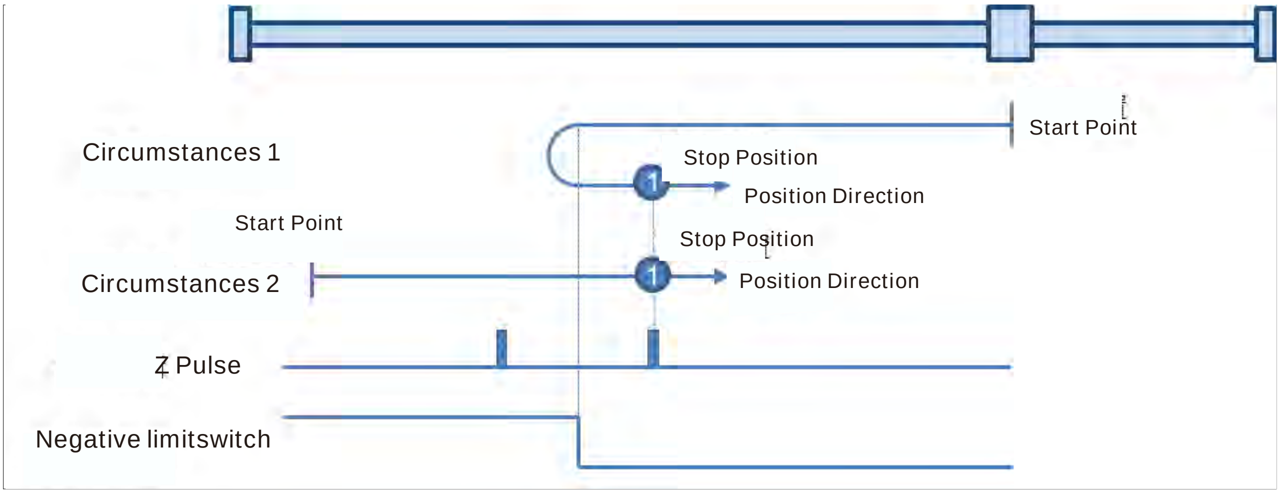
A.4 Explanation of DMC_Home_P

DFB_Home_P provides many homing modes from which user can choose the appropriate one in accordance with the field condition and technical requirement.

- **Mode 1: Homing which depends on the negative limit switch and Z pulse.**

Circumstance 1： MC_Home instruction is executed when the negative limit switch is OFF and the axis moves in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed when the axis encounters that the negative limit switch is ON. Where the first Z pulse is met is the home position when the negative limit switch is OFF.

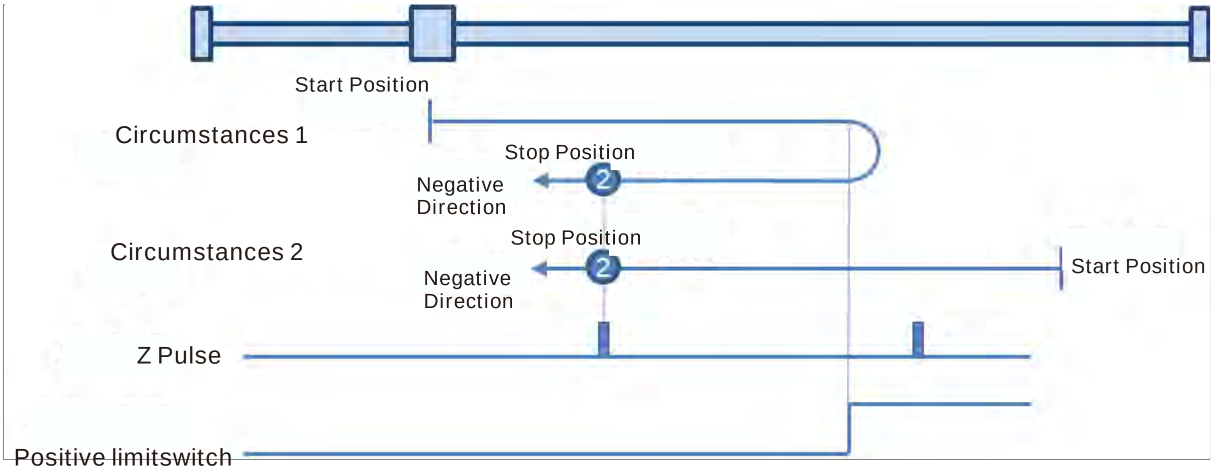
Circumstance 2： MC_Home instruction is executed when the negative limit switch is ON and the axis moves in the positive direction at the second-phase speed. Where the first Z pulse is met is the home position when the negative limit switch is OFF.



- **Mode 2: Homing which depends on the positive limit switch and Z pulse**

Circumstance 1 : MC_Home instruction is executed when the positive limit switch is OFF and the axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed when the axis encounters that the positive limit switch is ON. Where the first Z pulse is met is the home position while the positive limit switch is OFF.

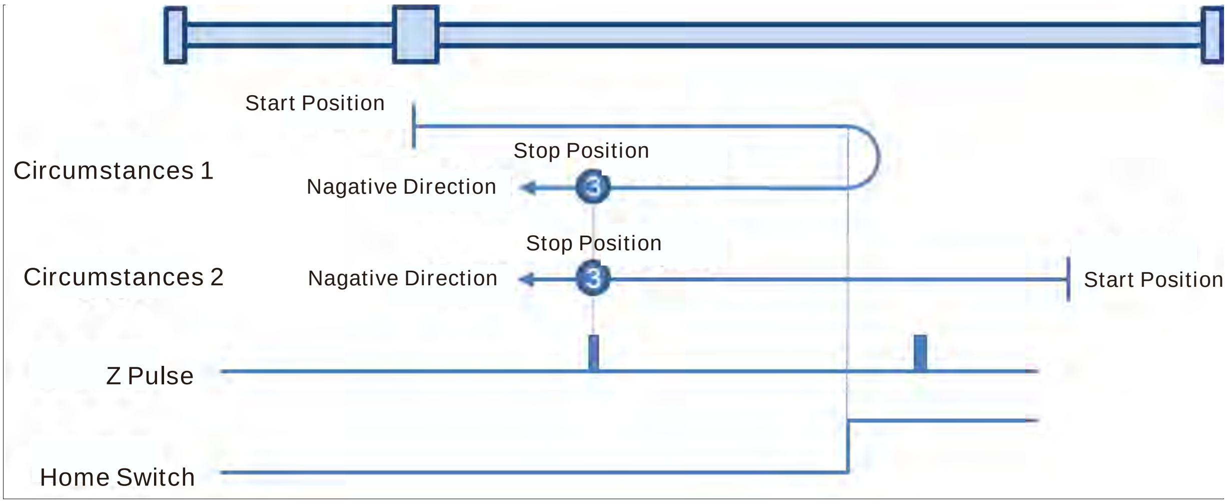
Circumstance 2 : MC_Home instruction is executed when the positive limit switch is ON and the axis moves in the negative direction at the second-phase speed. Where the first Z pulse is met is the home position while the positive limit switch is OFF.



- **Mode 3: Homing which depends on the home switch and Z pulse**

Circumstance 1 : When the home switch is OFF, MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed. When the axis encounters that the home switch is ON, the motion direction changes and the axis moves at the second-phase speed. Where the first Z pulse is met is the home position when the home switch is OFF.

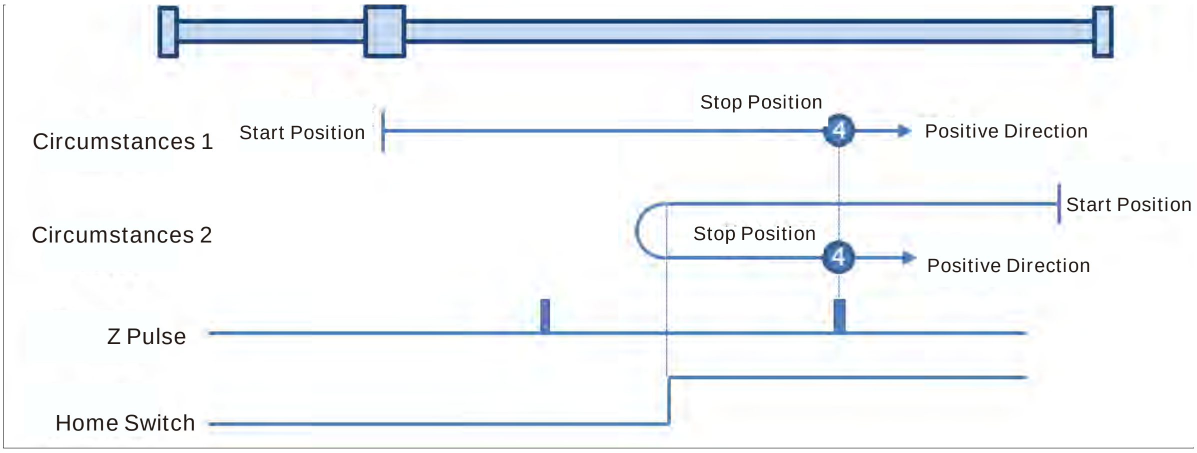
Circumstance 2 : When the home switch is ON, MC_Home instruction is executed and the axis directly moves in the negative direction at the second-phase speed. Where the first Z pulse is met is the home position while the home switch is OFF.



- **Mode 4: Homing which depends on the home switch and Z pulse**

Circumstance 1 : When the home switch is OFF, MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed. The axis moves at the second-phase speed when the axis encounters that the home switch is ON. Where the first Z pulse is met is the home position.

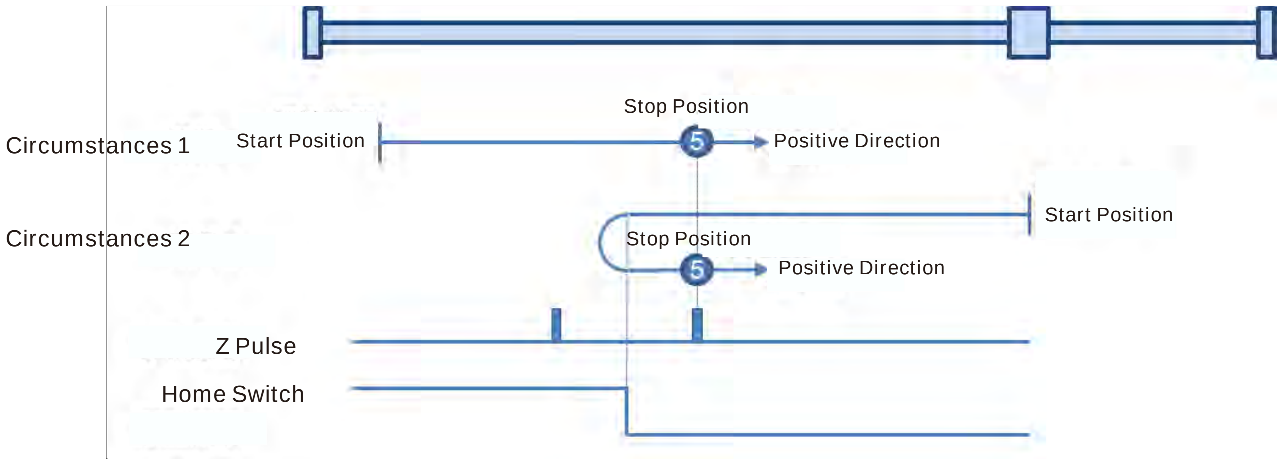
Circumstance 2 : When the home switch is ON, MC_Home instruction is executed and the axis moves in the negative direction at the second-phase speed. When the axis encounters that the home switch is OFF, the motion direction changes and the axis moves at the second-phase speed. Where the first Z pulse is met is the home position.



- **Mode 5 : Homing which depends on the home switch and Z pulse**

Circumstance 1 : When the home switch is ON, MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed. Where the first Z pulse is met is the home position while the home switch is OFF.

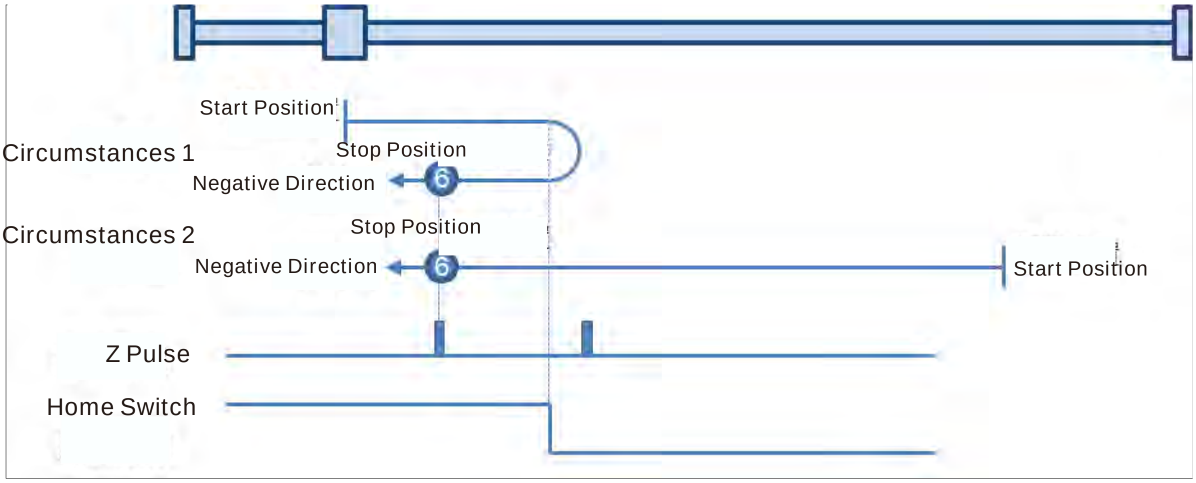
Circumstance 2 : When the home switch is OFF, MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed. When the home switch is ON, the motion direction changes and the axis moves at the second-phase speed. Where the first Z pulse is met is the home position when the home switch is OFF.



- **Mode 6: Homing which depends on the home switch and Z pulse**

Circumstance 1： When the home switch is ON, MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed. When the home switch is OFF, the motion direction changes and the axis moves at the second-phase speed. Where the first Z pulse is met is the home position.

Circumstance 2： When the home switch is OFF, MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed. While the home switch is ON, the axis moves at the second-phase speed and where the first Z pulse is met is the home position.

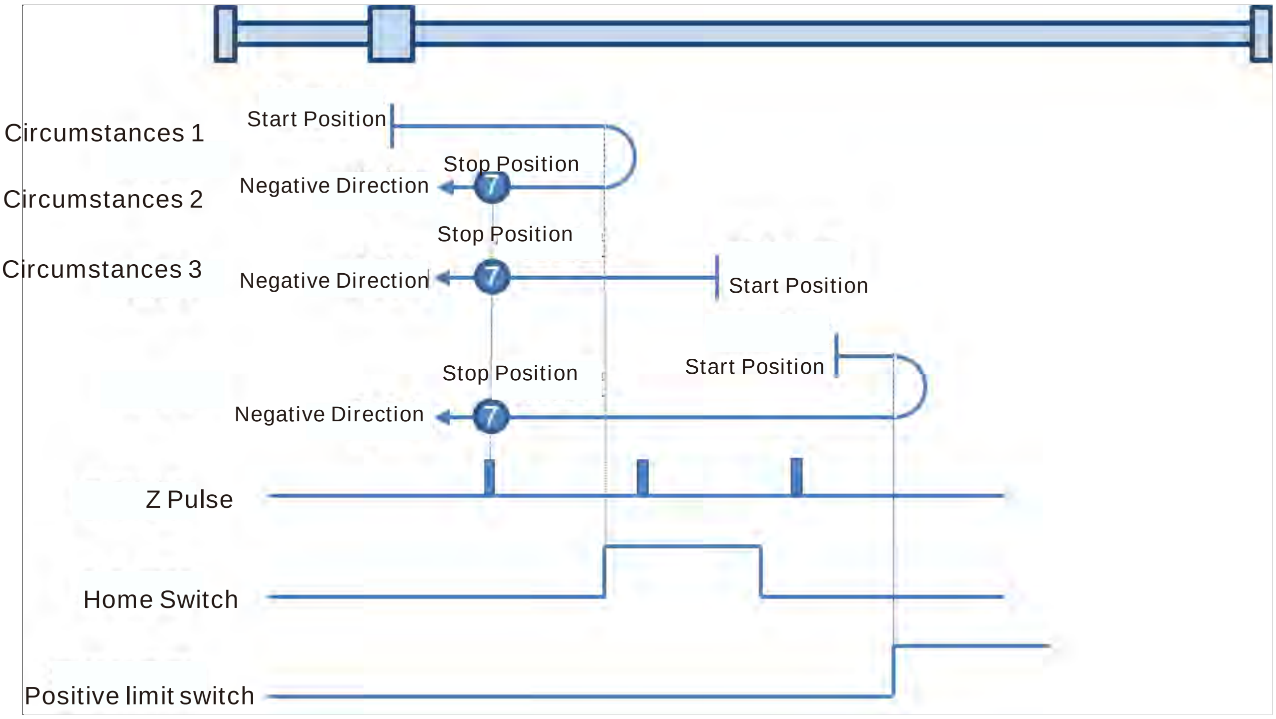


- **Mode 7: Homing which depending on the home switch, positive limit switch and Z pulse**

Circumstance 1： When the home switch is OFF, MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is ON. Where the first Z pulse is met is the home position when the home switch is OFF.

Circumstance 2： When the home switch is ON, MC_Home instruction is executed and the axis moves in the negative direction at the second-phase speed. Where the first Z pulse is met is the home position when the home switch is OFF.

Circumstance 3： When the home switch is OFF, MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. The axis starts to move at the second-phase speed when the home switch is ON. Where the first Z pulse is met is the home position when the home switch is OFF.

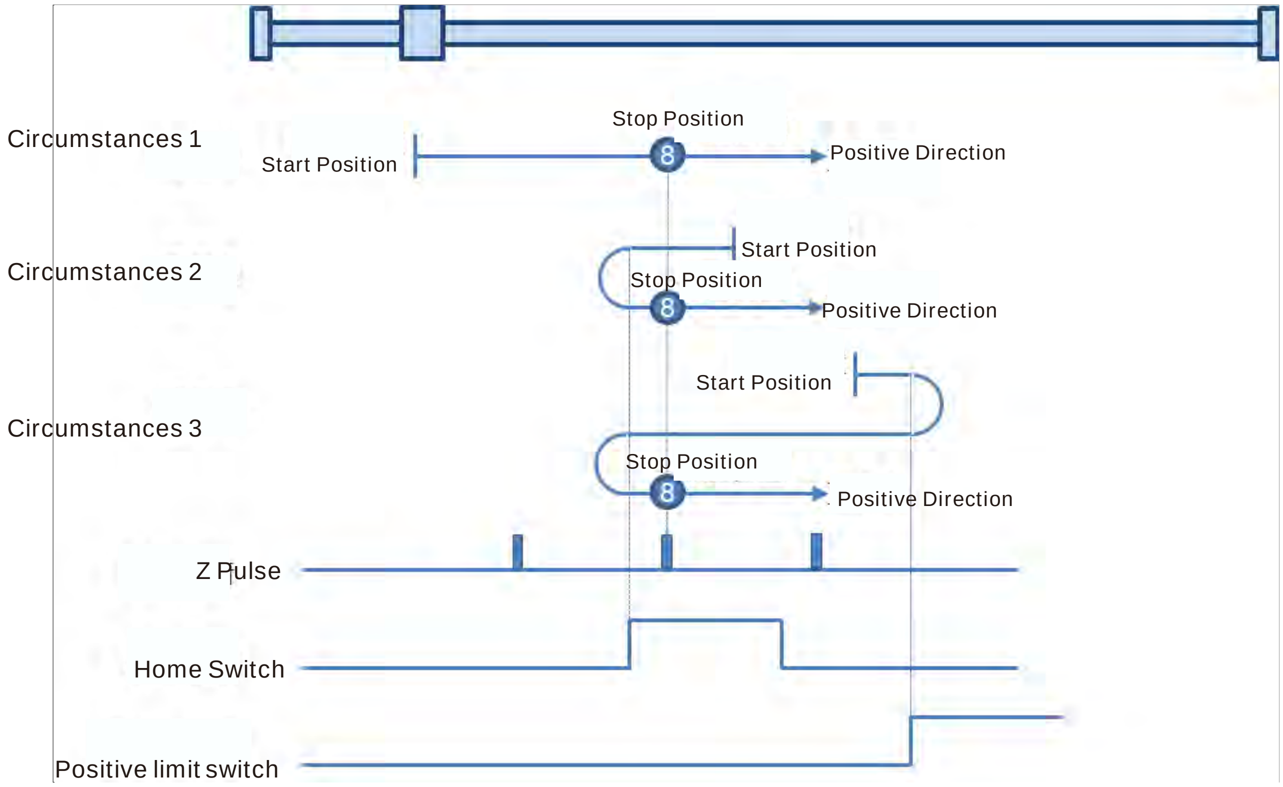


- **Mode 8: Homing depending on the home switch, positive limit switch and Z pulse.**

Circumstance 1 : When the home switch is OFF, MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed. The axis moves at the second-phase speed when the home switch is ON and where the first Z pulse is met is the home position.

Circumstance 2 : MC_Home instruction is executed and the axis moves in the negative direction at the second-phase speed when the home switch is ON. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. And where the first Z pulse is met is the home position.

Circumstance 3 : When the home switch is OFF, MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. The axis still moves at the first-phase speed when the home switch is ON. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF. The axis moves at the second-phase speed and where the first Z pulse is met is the home position when the home switch is ON.

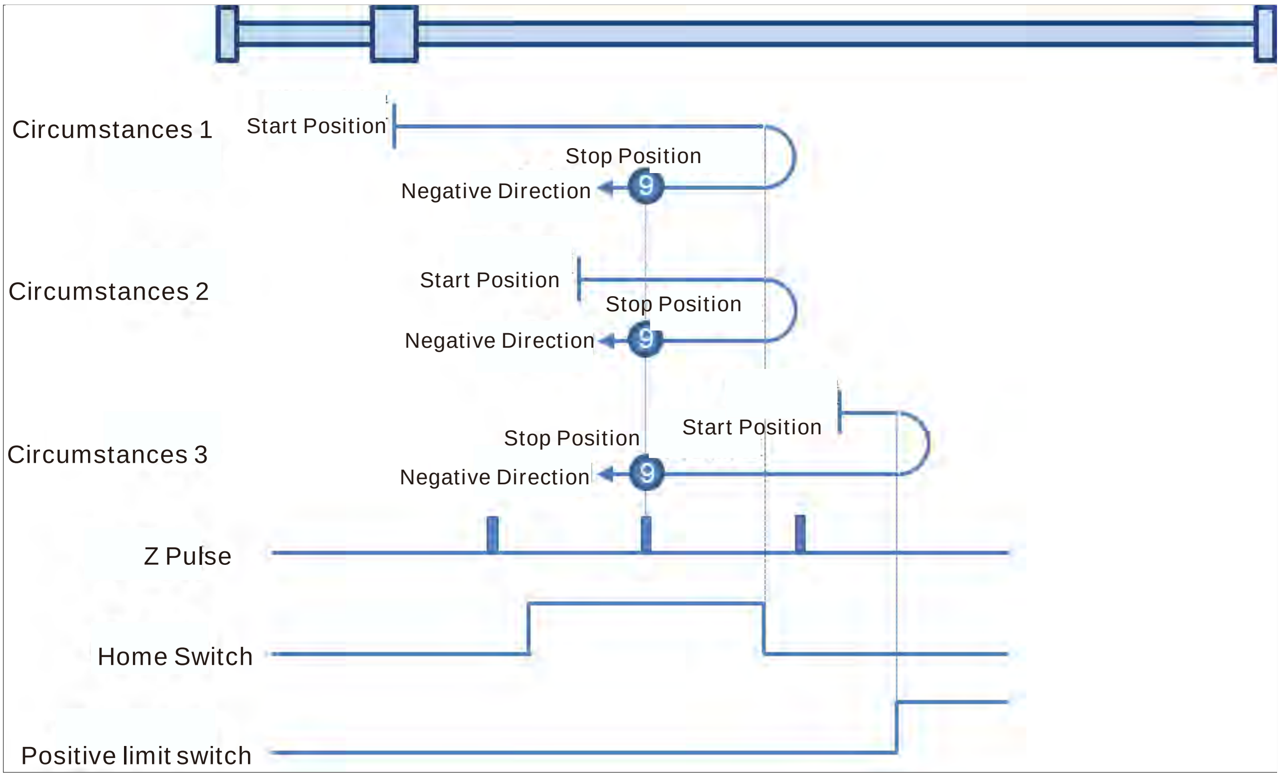


- **Mode 9: Homing depending on the home switch, positive limit switch and Z pulse**

Circumstance 1 : MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed when the home switch is OFF. The axis moves at the second-phase speed when the home switch is ON. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. And where the first Z pulse is met is the home position.

Circumstance 2 : When the home switch is ON MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. And where the first Z pulse is met is the home position.

Circumstance 3 : MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed when the home switch is OFF. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. The axis moves at the second-phase speed and where the first Z pulse is met is the home position when the home switch is ON.

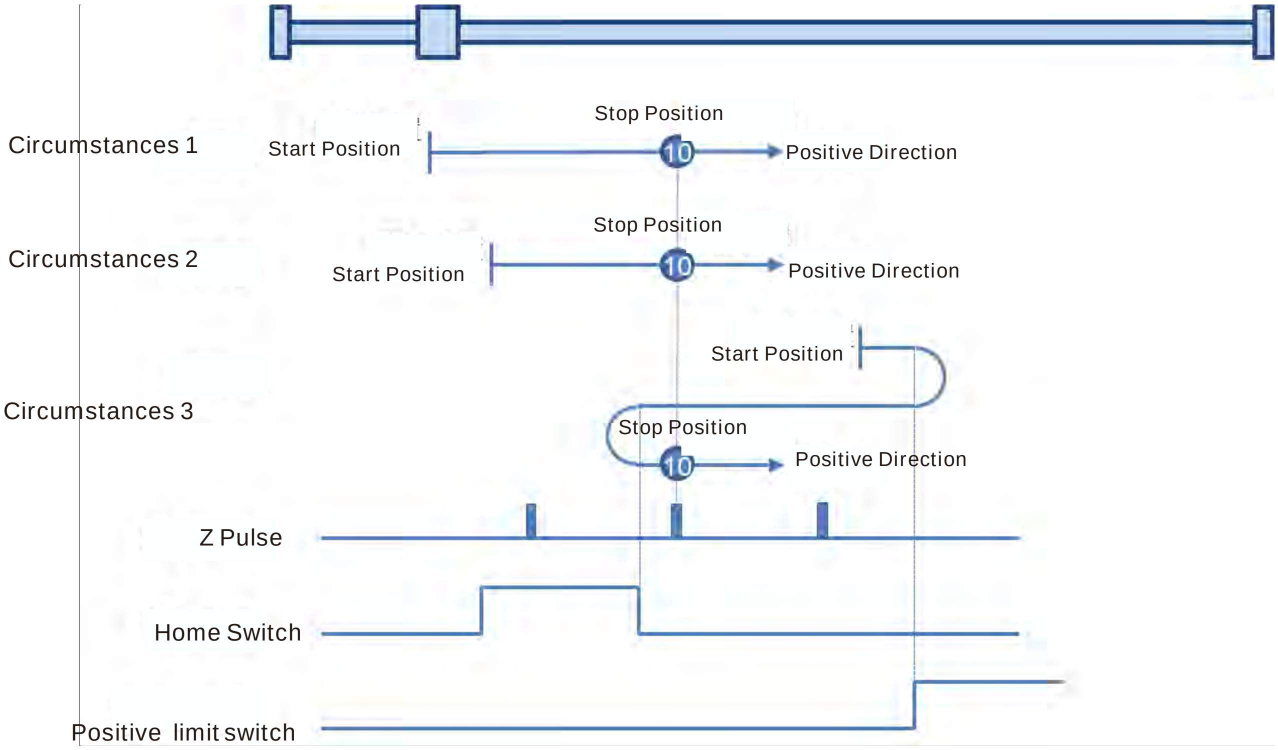


- **Mode 10: Homing depending on the home switch, positive limit switch and Z pulse.**

Circumstance 1： MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed when the home switch is OFF. The axis moves at the second-phase speed when the home switch is ON. And where the first Z pulse is met is the home position while the home switch is OFF.

Circumstance 2： MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed when the home switch is ON. And where the first Z pulse is met is the home position while the home switch is OFF.

Circumstance 3： MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed when the home switch is OFF. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. The motion direction changes again and the axis moves at the second-phase speed when the home switch is ON. Where the first Z pulse is met is the home position while the home switch is OFF.



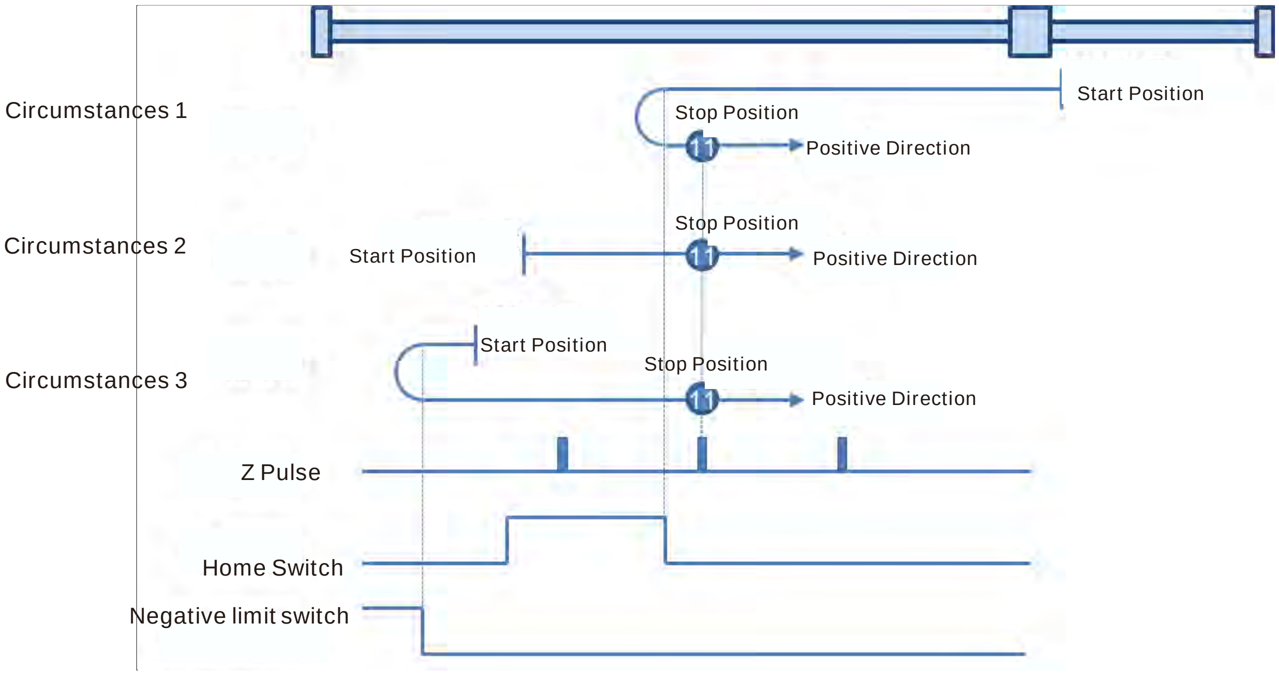
Mode 11~ mode 14 Homing which depends on the home switch, negative limit switch and Z pulse

- **Mode 11:**

Circumstance 1 : MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed when the home switch is OFF. The motion direction changes and the axis moves at the second-phase speed when the home switch is ON. And where the first Z pulse is met is the home position while the home switch is OFF.

Circumstance 2 : MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed while the home switch is ON. And where the first Z pulse is met is the home position while the home switch is OFF.

Circumstance 3 : MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The motion direction changes and the axis moves at the first-phase speed while the home switch is OFF and the negative limit switch is ON. The axis moves at the second-phase speed when the home switch is ON. Where the first Z pulse is met is the home position while the home switch is OFF.

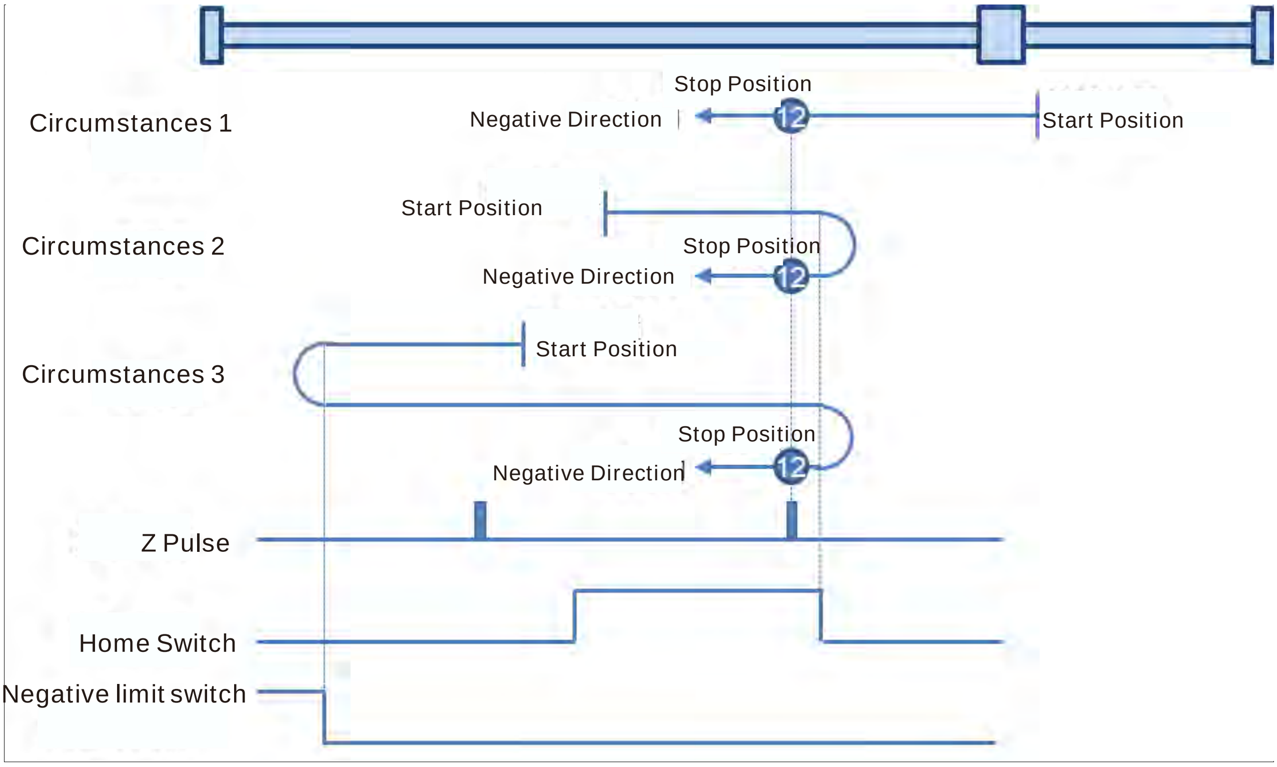


- **Mode 12: Homing depending on the home switch, negative limit switch and Z pulse**

Circumstance 1 : MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed when the home switch is OFF. The axis moves at the second-phase speed when the home switch is ON. And where the first Z pulse is met is the home position.

Circumstance 2 : MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed while the home switch is ON. The motion direction changes and the axis moves at the second-phase speed while the home switch is OFF. And where the first Z pulse is met is the home position.

Circumstance 3 : MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The motion direction changes and the axis moves at the first-phase speed while the home switch is OFF and the negative limit switch is ON. The axis still moves at the first-phase speed when the home switch is ON. The motion direction changes and the axis moves at the first-phase speed while the home switch is OFF. The axis moves at the second-phase speed while the home switch is ON. And where the first Z pulse is met is the home position.

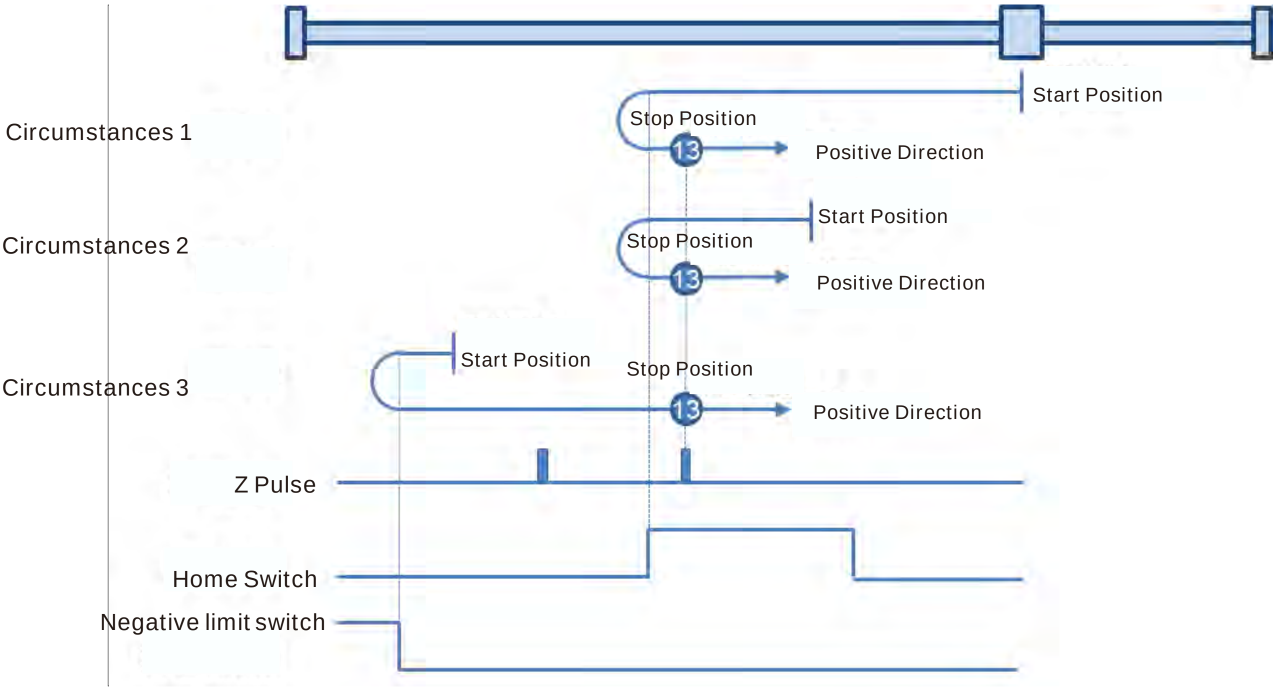


- **Mode 13: Homing depending on the home switch, negative limit switch and Z pulse**

Circumstance 1 : MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The axis moves at the second-phase speed while the home switch is ON. The motion direction changes and the axis moves at the second-phase speed while the home switch is OFF. And where the first Z pulse is met is the home position.

Circumstance 2 : MC_Home instruction is executed and the axis moves in the negative direction at the second-phase speed while the home switch is ON. The motion direction changes and the axis moves at the second-phase speed while the home switch is OFF. And where the first Z pulse is met is the home position.

Circumstance 3 : MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The motion direction changes and the axis moves at the first-phase speed while the home switch is OFF and the negative limit switch is ON. The axis moves at the second-phase speed and where the first Z pulse is met is the home position when the home switch is ON and the negative limit switch is OFF.

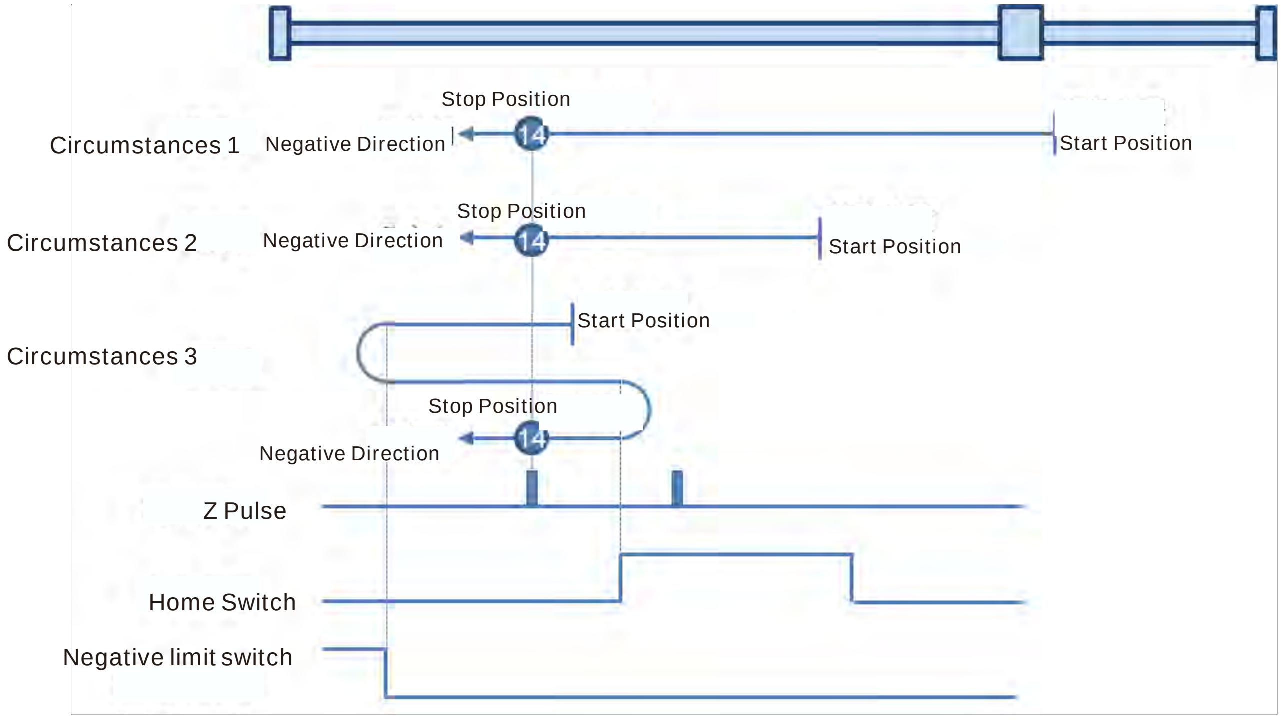


- **Mode 14: Homing depending on the home switch, negative limit switch and Z pulse**

Circumstance 1： MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The axis moves at the second-phase speed once the home switch is ON. And where the first Z pulse is met is the home position while the home switch is OFF.

Circumstance 2： MC_Home instruction is executed and the axis moves in the negative direction at the second-phase speed while the home switch is ON. Where the first Z pulse is met is the home position while the home switch is OFF.

Circumstance 3： MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The motion direction changes and the axis moves at the first-phase speed while the home switch is OFF and the negative limit switch is ON. The motion direction changes again and the axis moves at the second-phase speed when the home switch is ON. Where the first Z pulse is met is the home position while the home switch is OFF.



Mode 15 and mode 16 are reserved for future development.

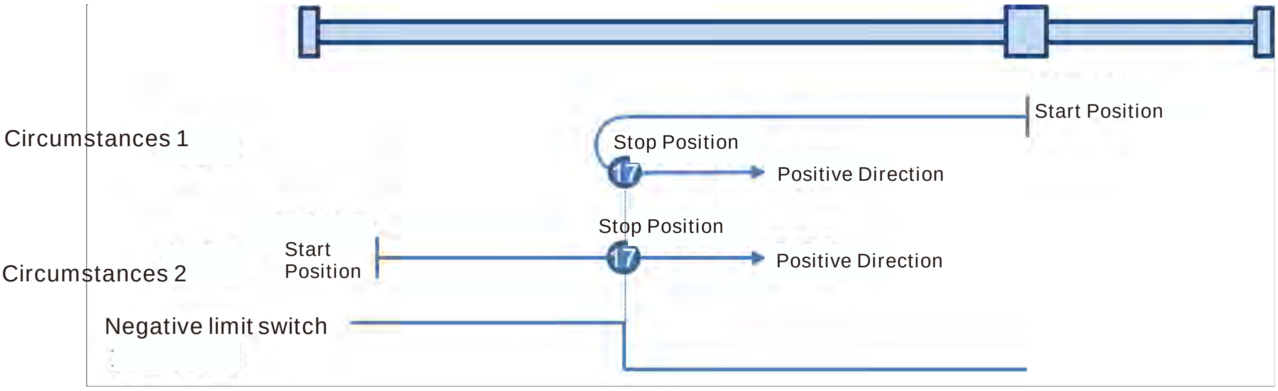
Mode 17~mode 30 Homing which has nothing to do with Z pulse

In mode 17~mode 30 which are respectively similar to mode1~mode 14 mentioned previously, the axis has nothing to do with Z pulse but the relevant home switch and limit switch status while returning to the home position.

- **Mode 17: Homing which depends on the negative limit switch, similar to mode 1, but has nothing to do with Z pulse.**

Circumstance 1： MC_Home instruction is executed when the negative limit switch is OFF and the axis moves in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed when the axis encounters that the negative limit switch is ON. Where the servo is when the negative limit switch is OFF is the home position.

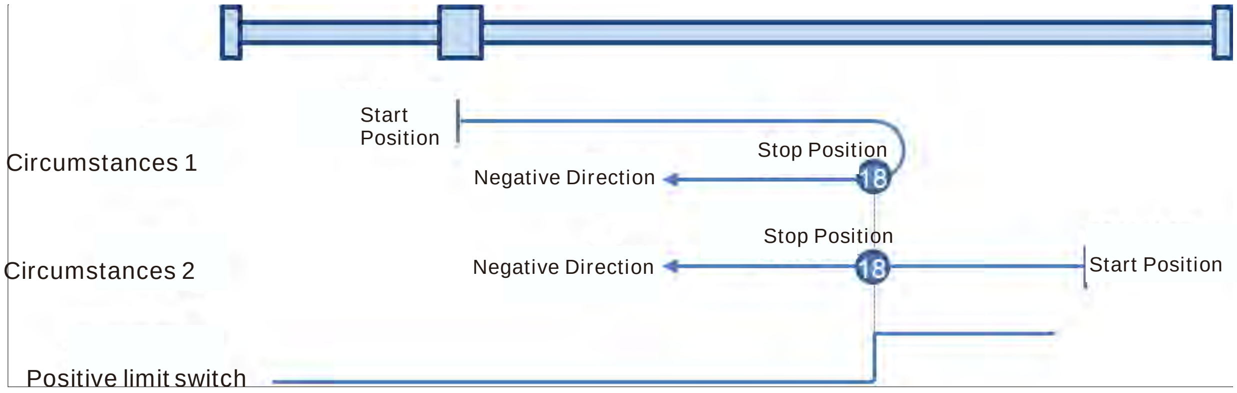
Circumstance 2： MC_Home instruction is executed when the negative limit switch is ON and the axis moves in the positive direction at the second-phase speed. Where the servo is is the home position when the negative limit switch is OFF.



- **Mode 18: Homing which depends on the positive limit switch, similar to mode 2, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed when the positive limit switch is OFF and the axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed when the axis encounters that the positive limit switch is ON. Where the servo is is the home position while the positive limit switch is OFF.

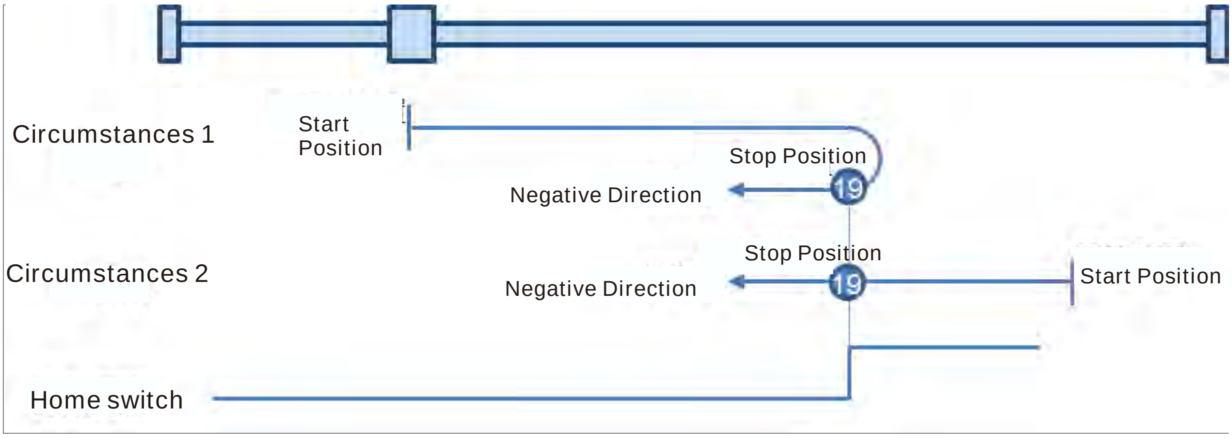
Circumstance 2 : MC_Home instruction is executed when the positive limit switch is ON and the axis moves in the negative direction at the second-phase speed. Where the servo is is the home position while the positive limit switch is OFF.



- **Mode 19: Homing which depends on the home switch, similar to mode 3, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed and the axis moves in the positive direction at the first-phase speed while the home switch is OFF. The motion direction changes and the axis moves at the second-phase speed once the home switch becomes ON. And where the axis stands is the home position at the moment the home switch becomes OFF.

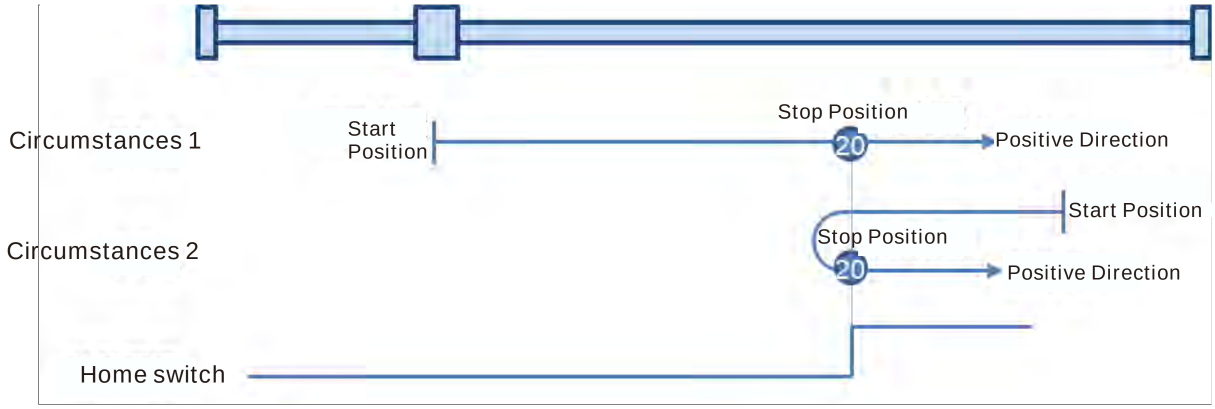
Circumstance 2 : MC_Home instruction is executed and the axis directly moves in the negative direction at the second-phase speed while the home switch is ON. And where the axis stands is the home position at the moment when the home switch becomes OFF.



- **Mode 20: Homing which depends on the home switch, similar to mode 4, but has nothing to do with Z pulse.**

Circumstance 1： MC_Home instruction is executed when the home switch is OFF and the axis moves in the positive direction at the first-phase speed. Where the servo is is the home position when the home switch is ON.

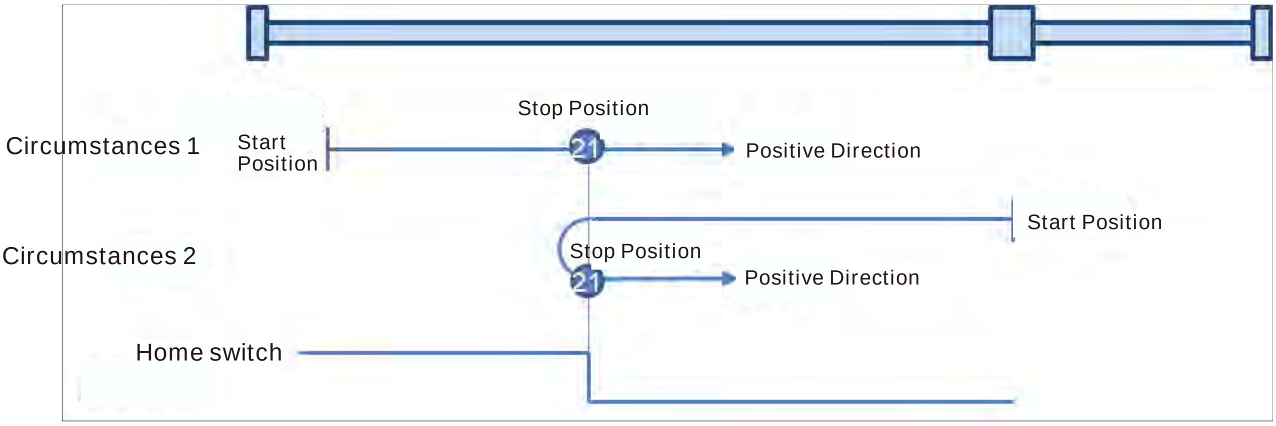
Circumstance 2： MC_Home instruction is executed when the home switch is ON and the axis moves in the negative direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch becomes OFF. Where the servo is is the home position when the home switch is ON.



- **Mode 21: Homing which depends on the home switch, similar to mode 5, but has nothing to do with Z pulse.**

Circumstance 1： MC_Home instruction is executed and the axis moves in the positive direction at the second-phase speed while the home switch is ON. And where the axis stands is the home position at the moment the home switch becomes OFF.

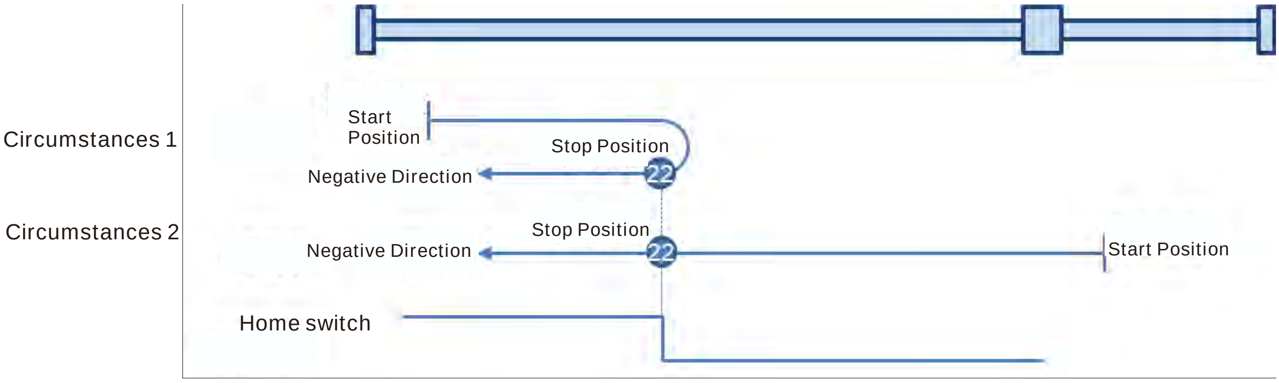
Circumstance 2： MC_Home instruction is executed and the axis moves in the negative direction at the first-phase speed while the home switch is OFF. The motion direction changes and the axis moves at the second-phase speed once the home switch becomes ON. And where the axis stands is the home position at the moment the home switch becomes OFF.



- **Mode 22: Homing which depends on the home switch, similar to mode 6, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed while the home switch is ON and the axis moves in the positive direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed once the home switch becomes OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 2 : MC_Home instruction is executed while the home switch is OFF and the axis moves in the negative direction at the first-phase speed. Where the axis stands is the home position when the home switch becomes ON.

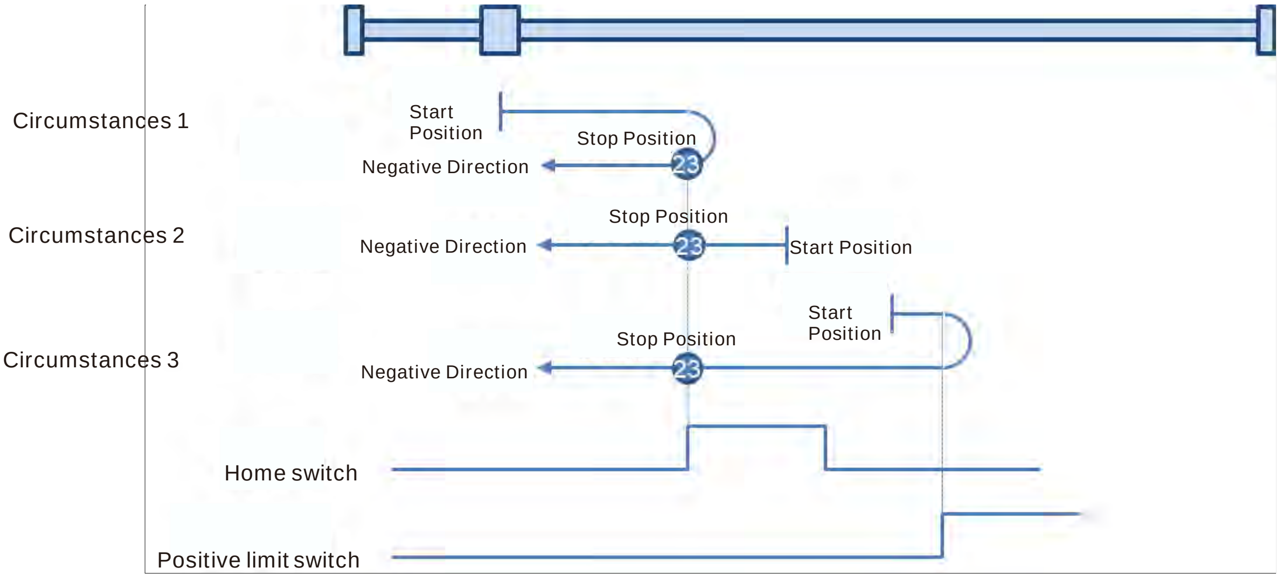


- **Mode 23: Homing which depends on the home switch and positive limit switch, similar to mode 7, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed while the home switch is OFF and the axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed once the home switch becomes ON. Where the axis stands is the home position when the home switch is OFF.

Circumstance 2 : MC_Home instruction is executed while the home switch is ON and the axis moves in the negative direction at the second-phase speed. And where the axis stands is the home position when the home switch becomes OFF.

Circumstance 3 : MC_Home instruction is executed while the home switch is OFF. The axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. When the home switch is ON, the axis starts to move at the second-phase speed. Where the axis stands is the home position when the home switch is OFF.

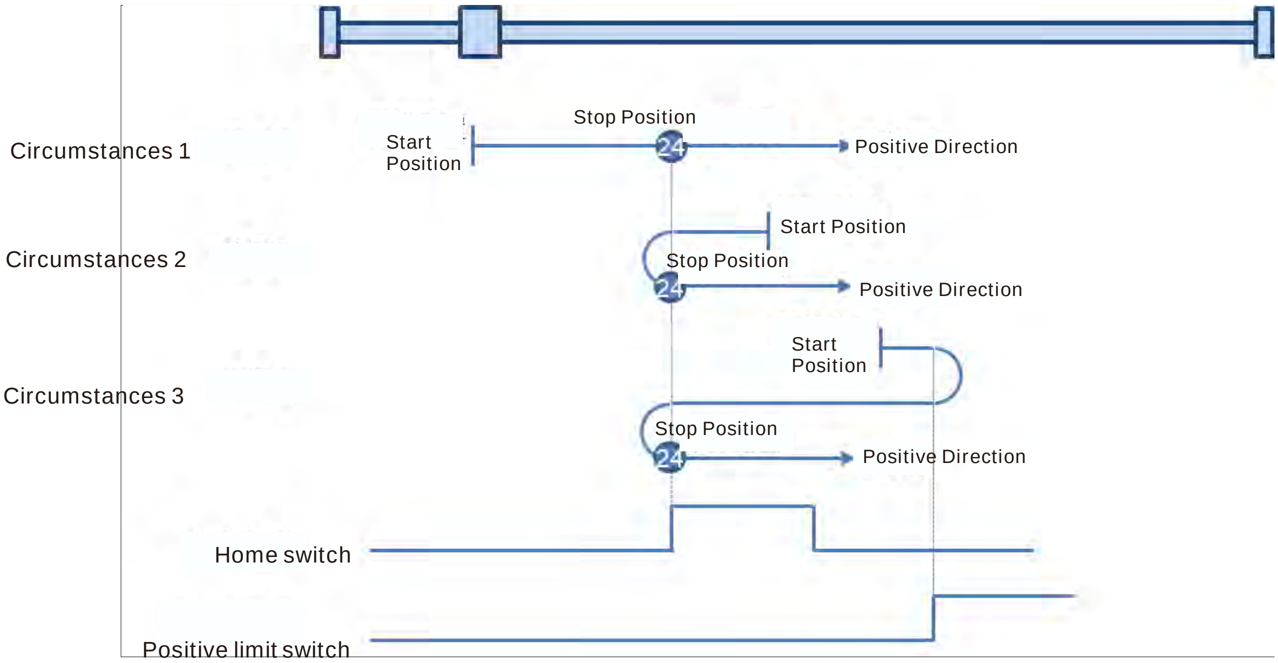


- **Mode 24: Homing which depends on the home switch and positive limit switch, similar to mode 8, but has nothing to do with Z pulse.**

Circumstance 1： MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the positive direction at the first-phase speed. Where the axis stands is the home position when the home switch is ON.

Circumstance 2： MC_Home instruction is executed while the home switch is ON and the axis moves in the negative direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 3： MC_Home instruction is executed while the home switch is OFF. The axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. When the home switch is ON, the axis still moves at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

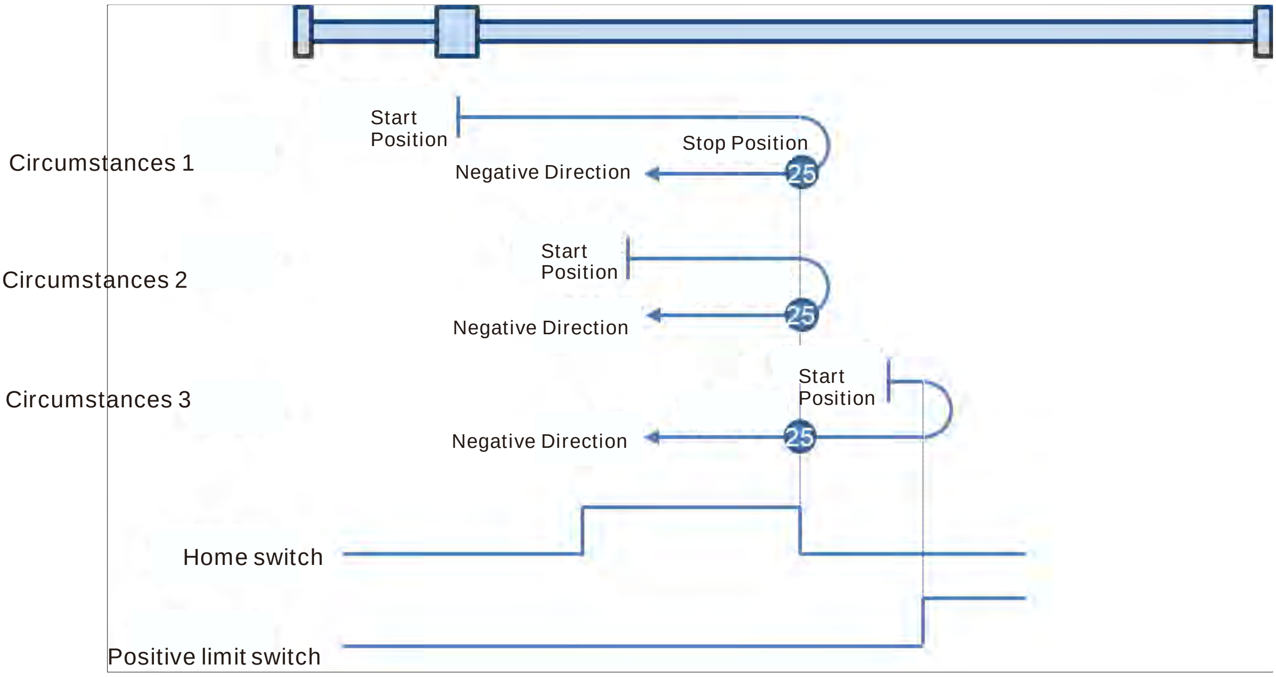


- **Mode 25: Homing which depends on the home switch and positive limit switch, similar to mode 9, but has nothing to do with Z pulse.**

Circumstance 1： MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the positive direction at the first-phase speed. The axis moves at the second-phase speed when the home switch is ON. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 2： MC_Home instruction is executed while the home switch is ON and the axis moves in the positive direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 3： MC_Home instruction is executed while the home switch is OFF. The axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. Where the axis stands is the home position when the home switch is ON.

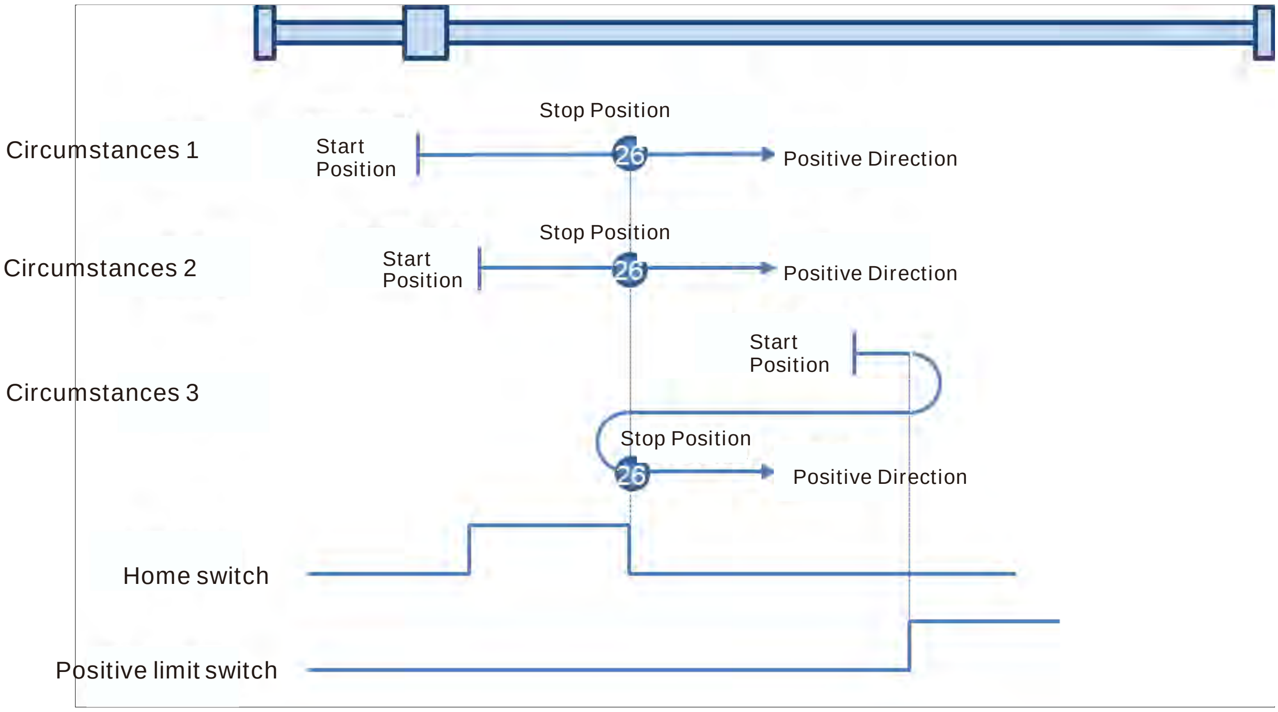


- **Mode 26: Homing which depends on the home switch and positive limit switch, similar to mode 10, but has nothing to do with Z pulse.**

Circumstance 1： MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the positive direction at the first-phase speed. The axis moves at the second-phase speed when the home switch is ON. Where the axis stands is the home position when the home switch is OFF.

Circumstance 2： MC_Home instruction is executed while the home switch is ON and the axis moves in the positive direction at the second-phase speed. Where the axis stands is the home position when the home switch is OFF.

Circumstance 3： MC_Home instruction is executed while the home switch is OFF. The axis moves in the positive direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the positive limit switch is ON. The motion direction changes again and the axis moves at the second-phase speed when the home switch is ON. Where the axis stands is the home position when the home switch is OFF.

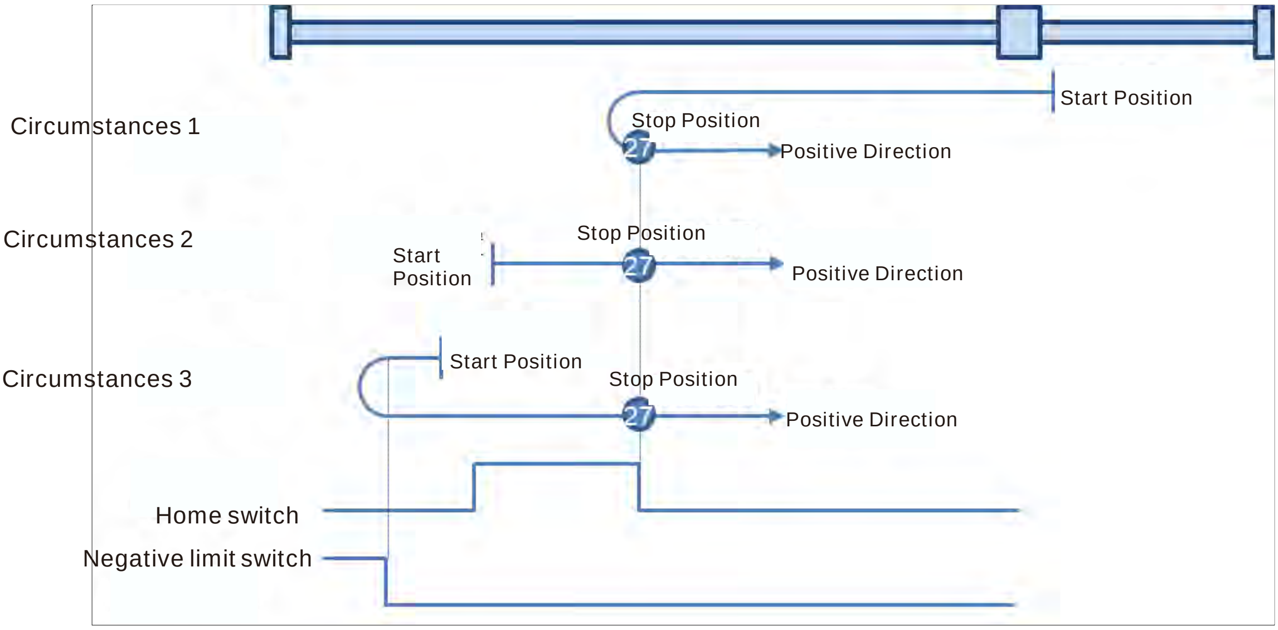


- **Mode 27: Homing which depends on the home switch and negative limit switch, similar to mode 11, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is ON. Where the axis stands is the home position when the home switch is OFF.

Circumstance 2 : MC_Home instruction is executed while the home switch is ON and the axis moves in the positive direction at the second-phase speed. Where the axis stands is the home position when the home switch is OFF.

Circumstance 3 : MC_Home instruction is executed while the home switch is OFF. The axis moves in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the negative limit switch is ON. When the home switch is ON, the axis starts to move at the second-phase speed. Where the axis stands is the home position when the home switch is OFF.

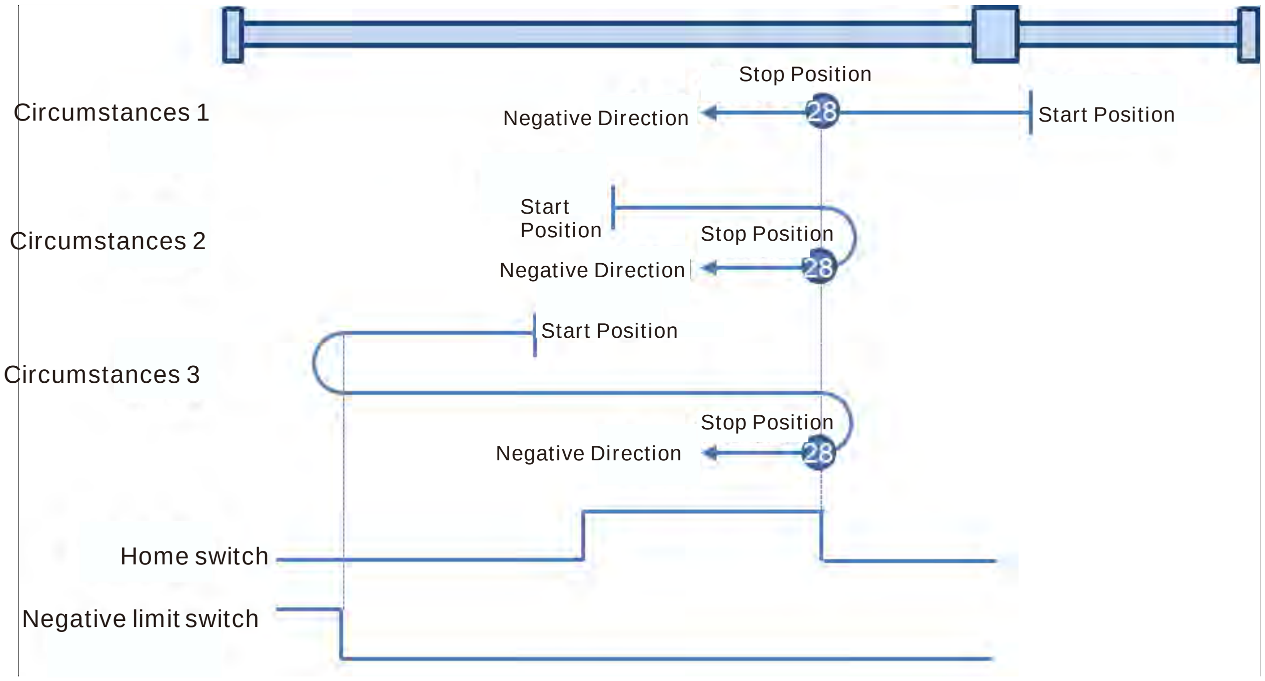


- **Mode 28: Homing which depends on the home switch and negative limit switch, similar to mode 12, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the negative direction at the first-phase speed. Where the axis stands is the home position when the home switch is ON.

Circumstance 2 : MC_Home instruction is executed while the home switch is ON and the axis moves in the positive direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 3 : MC_Home instruction is executed while the home switch is OFF. The axis moves in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the negative limit switch is ON. When the home switch is ON, the axis still moves at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

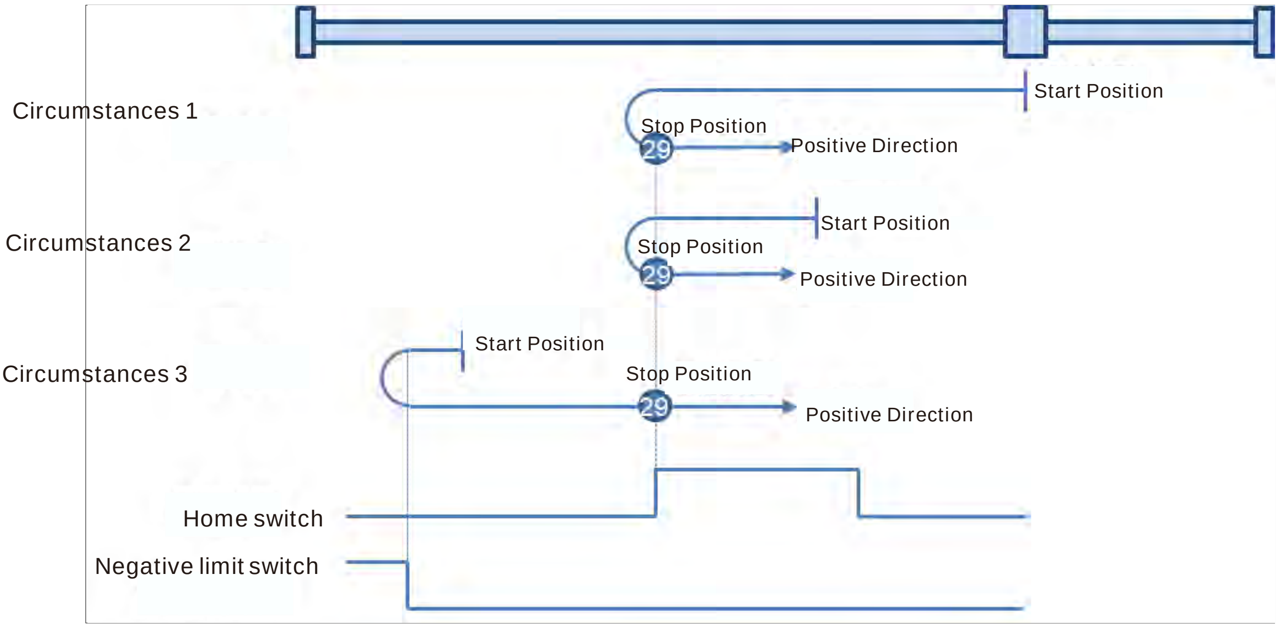


- **Mode 29: Homing which depends on the home switch and negative limit switch, similar to mode 13, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the negative direction at the first-phase speed. When the home switch is ON, the axis starts to move at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 2 : MC_Home instruction is executed while the home switch is ON and the axis moves in the negative direction at the second-phase speed. The motion direction changes and the axis moves at the second-phase speed when the home switch is OFF. Where the axis stands is the home position when the home switch is ON.

Circumstance 3 : MC_Home instruction is executed while the home switch is OFF. The axis moves in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the negative limit switch is ON. Where the axis stands is the home position when the home switch is ON.

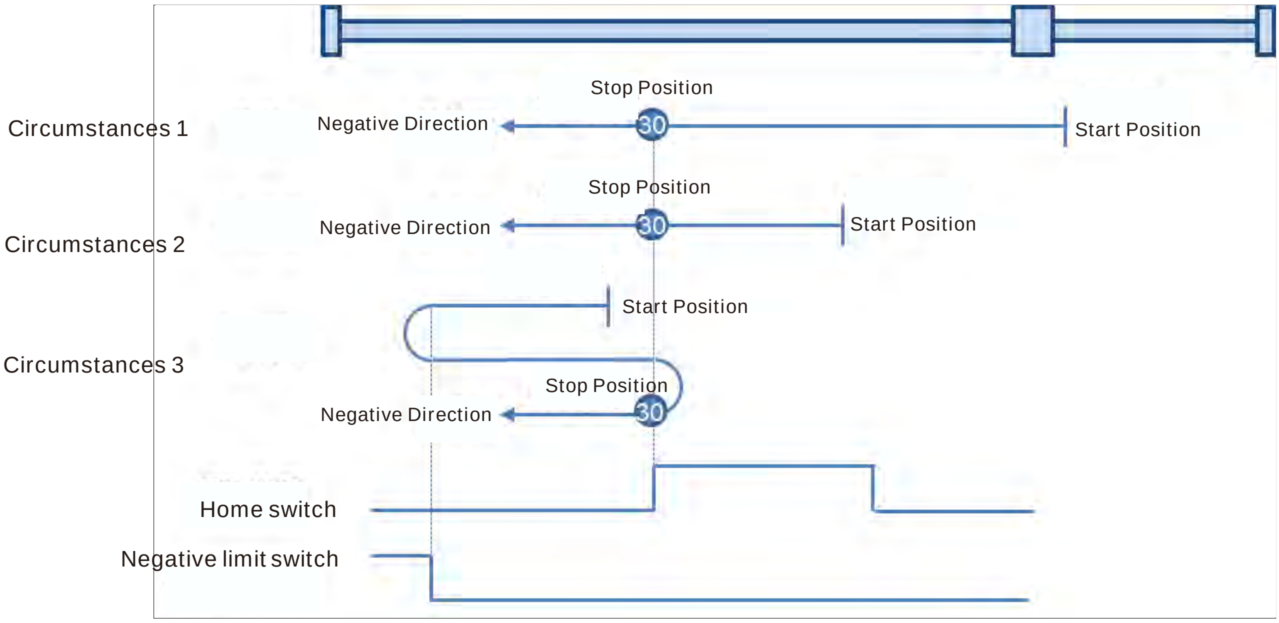


- **Mode 30: Homing which depends on the home switch and negative limit switch, similar to mode 14, but has nothing to do with Z pulse.**

Circumstance 1 : MC_Home instruction is executed while the home switch is OFF and the axis starts to move in the negative direction at the first-phase speed. When the home switch is ON, the axis starts to move at the second-phase speed. Where the axis stands is the home position when the home switch is OFF.

Circumstance 2 : MC_Home instruction is executed while the home switch is ON and the axis moves in the negative direction at the second-phase speed. Where the axis stands is the home position when the home switch is OFF.

Circumstance 3 : MC_Home instruction is executed while the home switch is OFF. The axis moves in the negative direction at the first-phase speed. The motion direction changes and the axis moves at the first-phase speed when the home switch is OFF and the negative limit switch is ON. When the home switch is ON, the motion direction changes again and the axis moves at the second-phase speed. Where the axis stands is the home position when the home switch is OFF



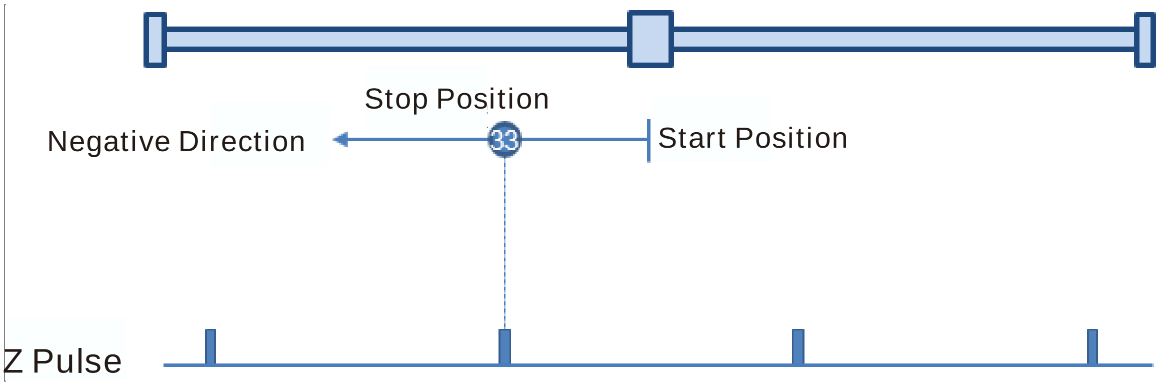
Mode 31 and mode 32: Reserved

Mode 31 and mode 32 Reserved for future development.

Mode 33 ~ mode 34 Homing which only depends on Z pulse

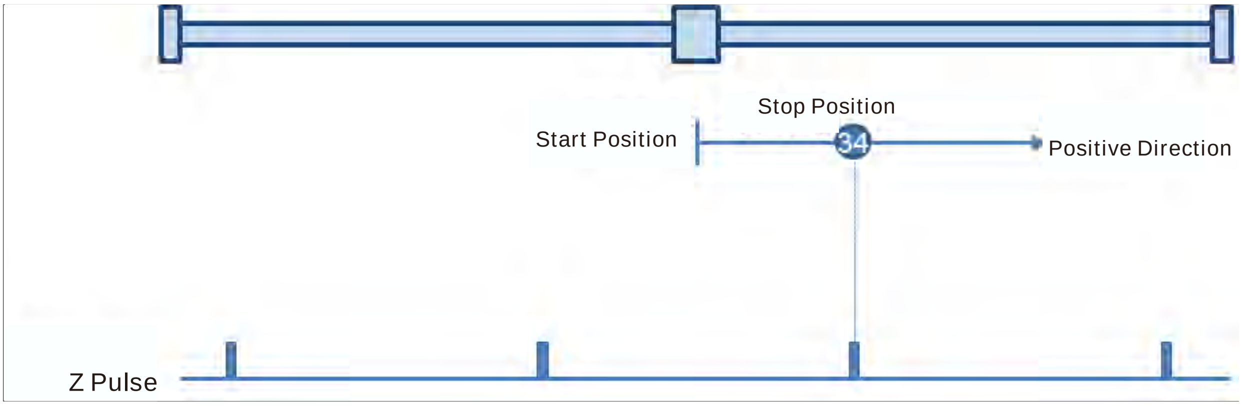
- **Mode 33: Homing depending on Z pulse (Negative direction)**

  MC_Home instruction is executed and the axis moves at the second-phase speed in the negative direction. And the place where the axis stands is the home position once the first Z pulse is met.



- **Mode 34: Homing depending on Z pulse (Positive direction)**

MC_Home instruction is executed and the axis moves at the second-phase speed in the positive direction. And the place where the axis stands is the home position once the first Z pulse is met.



- **Mode 35: Homing which depends on the current position**

MC_Home instruction is executed, the axis does not move and its current position is regarded as the home position.

MEMO