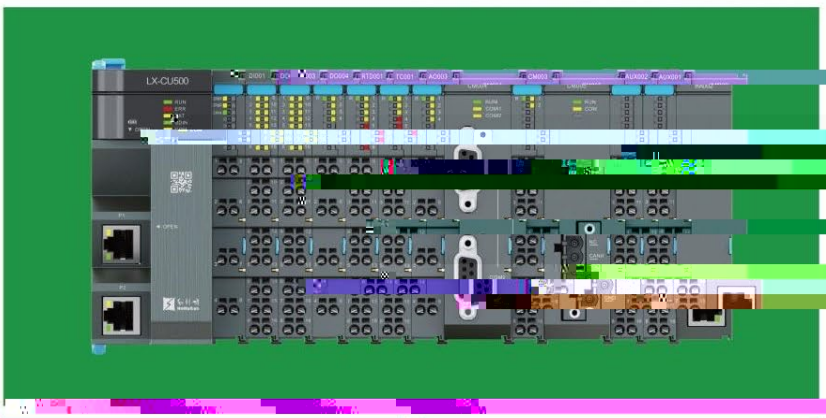
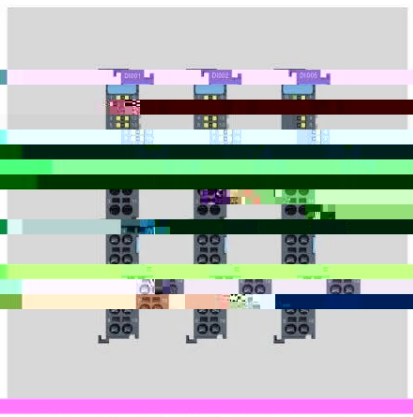




PLCopen Command Manual

LX Series Programmable Logic Controllers

7FSTJPO

5.18.2 Functions 138
5.18.3 Examples 138
5.18.4 Precautions 141
5.18.5 Reference 141
5.19 MC_ReadActualTorque (Read Axis Actual Torque) 141
5.19.1 Parameter 141
5.19.2 Functions 142
5.19.3 Examples 143
5.19.4 Precautions 146
5.20 MC_ReadStatus (Read Axis Status) 146
5.20.1 Parameter 146
5.20.2 Functions 147
5.20.3 Examples 148
5.20.4 Precautions 150
5.20.5 Reference 150
5.21 MC_ReadAxisInfo (Read Axis Info) 150
5.21.1 Parameter 150
5.21.2 Functions 151
5.21.3 Examples 153
5.22 MC_ReadAxisError (Read Axis Error) 156
5.22.1 Parameter 156
5.22.2 Functions 157
5.22.3 Examples 158
5.22.4 Precautions 160
5.22.5 Reference 160
5.23 SMC_ReadAxisLimitStatus (Read Axis Limit Status) 160
5.23.1 Parameter ..... 2 Tc 0 Tw 22.916 0 Td ( )Tj -0.001 Tc 0.001 Tw 0.229 072.96 13538 23330 Td ET
Ru.6 Tm (68 Tm (( )Tj 7T 1.1 (S)T1 1 Tf

7.10.4 Precautions 340
Chapter 8 APPENDICES 341
8.1 Error Code Description 341
8.2 Description of Buffer Mode 346
8.2.1 Buffer mode 346

Chapter 1 Foreword

(https://cn.hollysys.com/other/download.html) to download the file.

Chapter 4 Overview commands

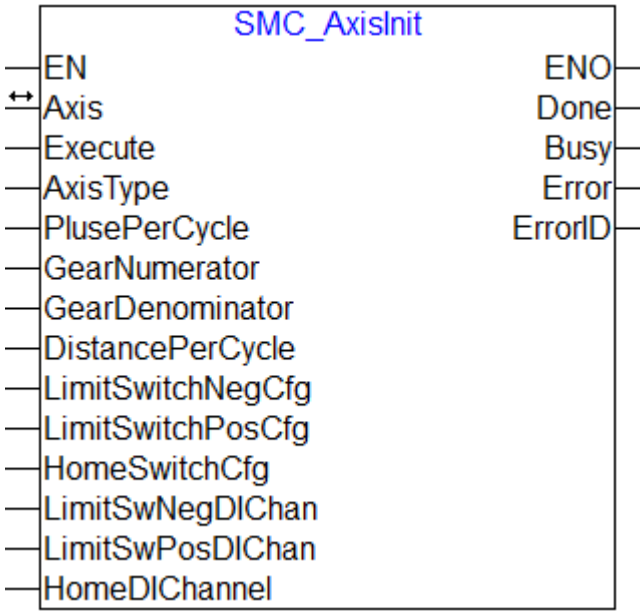
4.1 List of commands

Command Library	Commands	Name	Function Description
Single-axis motion command	SMC_AxisInit	Axis initialization configuration	Used to initialize the axis configuration. With this function block, you can set the basic configuration required for axis operation, such as axis type, user unit conversion, limit switch channel, etc.
	MC_Power	Axis enable	Control servo enabling status command
	MC_Home	Homing	Single axis origin return command
	MC_Stop	Axis stop	The control axis stops at the specified deceleration, any motion control command being executed on this axis is suspended, and all motion buffer commands on this axis are canceled
	MC_MoveJog	Jog	Single-axis inching command in positive/negative direction
	MC_MoveAbsolute	Absolute positioning	Single-axis absolute positioning command: control axis moves to specified absolute position
	MC_MoveRelative	Relative positioning	Single-axis relative positioning command: The control axis moves to the specified position incrementally.
	MC_MoveVelocity	Speed control	Single-axis speed control command, simulating speed control in position control mode
	SMC_SyncMoveVelocity	Cyclic Synchronous Speed Control	Speed control in cyclic synchronous speed mode
	MC_TorqueControl	Torque control	Single-axis torque control command, which controls the axis with a ramp in the torque mode
	SMC_SetTorqueLimit	Axial torque limit setting	Set the limit value of servo positive/negative torque
	MC_SetPosition	Set the current position	Redefines the current position of the specified axis absolutely or relative
	MC_SetOverride	Motion override setting (override)	This command sets the speed, acceleration/deceleration and jerk scale factor, which can increase or decrease the axis running speed, acceleration/deceleration and jerk according to the set ratio.
	SMC_SetDynamicLimits	Setting of motion parameter limit value	This command sets the limit values of axis motion parameters.
	MC_ReadDigitalInput		

4.2 Description of commands

Before using the command, please understand the relevant information involved in the content of the command for better understanding and use.

Information Item	Description
Command/Function Block	Indicates the command name, for example: ADD
Name	It indicates the Chinese name of an command, such as: addition command.
FB/FUN	Indicates whether the command is a function block (FB) or a function (FUN). Function block type command: It can be called by program or FB. Functional command: It can be called by any one of program, FB and FUN.
Graphic Examples	Examples of command Application in LD Ladder Programming Language
Example of ST	Examples of command Application in ST Programming Language
Parameter	Describe the parameter information such as input, output and intermediate variables of commands.
Functions	Description of command functions
Examples	The application example of the command is illustrated by programming examples in LD and ST.
Precautions	Describe the precautions when using commands, including applications in special scenarios and occurrence of abnormal situations.
Reference	Other content information to be referenced when using this command



SMC_AxisInit

5.2.1 Parameter

Parameter Type	Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
	AxisType	Shaft type	No	-	-1:Unused 0:Virtual 50:EtherCAT_Position 51:EtherCAT_Speed 52:EtherCAT_Torque 53:EtherCAT_Encoder	EMAXISTYPE
	PlusePerCycle	Number of pulses per revolution of motor/encoder	No	-	Positive value	LREAL
	GearNumerator	Reduction ratio numerator (Revolutions of motor input shaft)	YES	1	Positive value	UDINT
	GearDenominator	Reduction ratio denominator (Number of retarder output shaft revolutions)	YES	1	Positive value	UDINT
	DistancePerCycle	When the reduction ratio is 1:1, it indicates the movement amount of the table for one rotation of the motor/encoder When the reduction ratio is not 1:1, it indicates the movement amount of the table for one revolution of the reducer output shaft	No	-	Positive value	LREAL
	LimitSwitchNegCfg	Negative Limit Switch Property Configuration	YES	-1	-1: Not used 0: Use drive body DI	INT

					(normally open) 1: Use the drive body DI (normally closed) 10: Use universal DI (normally open) 20: use universal DI (normally closed)	
	LimitSwitchPosCfg	Positive Limit Switch Property Configuration	YES	-1	-1: Not used 0: Use drive body DI (normally open) 1: Use the drive body DI (normally closed) 10: Use universal DI (normally open) 20: use universal DI (normally closed)	INT

HomeSwitchCfg

Axis type setting

The axis type of the specified axis can be set by setting the parameter AxisType to the axis type supported by Controller model. The axis types supported by LX series Controller are as follows:

Unused (-1): The axis is not used.

Virtual axis (0): virtual axis

EtherCAT_Position(50): EtherCAT bus connection axis, position control

EtherCAT_Speed(51): EtherCAT bus connecting axis, speed control

EtherCAT_Torque(52): EtherCAT bus connecting shaft, torque control

EtherCAT_Encoder(53): EtherCAT bus connection axis, encoder count

Attention shall be paid to different shaft types:

Axes of type EtherCAT_Encoder do not accept motion control commands and only track encoder feedback values.

When there are commands running in the axis command buffer of the axis, the set axis type does not take effect.

After the axis type is successfully set, the system will automatically modify the axis parameter RepMode according to different axis types. When the user sets the axis type as virtual axis, the system will set RepMode=2; when the user sets the axis type as other axis types, the system will set RepMode=0.

Unit Conversion Factor Setting

Set the number of pulses (1) for the motor/encoder to rotate one circle through input parameter PlusePerCycle, unit: pulse/rev;

Set the reduction ratio numerator through input parameter GearNumerator, i.e. the number of teeth in (4) in the figure below;

Set the reduction ratio denominator through input parameter GearDenominator, i.e. the number of teeth in (5) in the following figure;

When the transmission is not used, the input parameter DistancePerCycle represents the movement amount (2) of the table for one revolution of the motor and encoder, in user unit/rev; when the transmission is used, the input parameter DistancePerCycle can be used to set the movement amount (3) of the table for one revolution of the reducer output shaft, in user unit/rev.

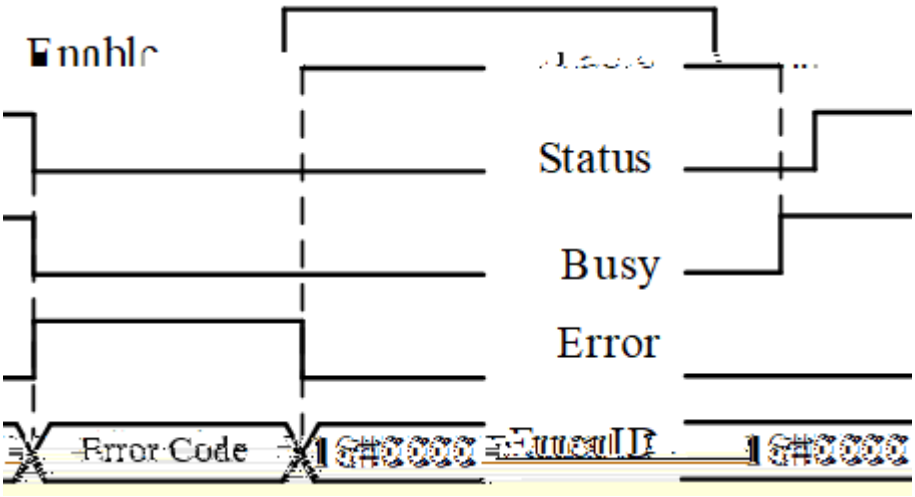
The user unit can be defined according to the user's needs, such as pulse number, millimeter and degree.

When the output is linear displacement, the schematic diagram is as follows:

M: MOTOR/ENCODER, W:TABLE

When the output is rotational displacement, the schematic diagram is as follows:

(2) The Timing diagram in case of abnormality is as follows:



5.3.3 Examples

Create an MC_Power example program to set the axis enable.

Definition of Primary Variables

Title	Data Type	Initial Value	Description

```
   Axis := Axis0,
   Status=> PowerStatus,
   Busy=>Busy2,
   Error=>Error2,
   ErrorID=>ErrorID2);
END_IF
```

(3) Reset processing can be carried out in case of error.

```
MC_Reset1(
Execute := StartReset,
Axis := Axis0,
Done=>Done4,
Busy=>Busy5,
Error=>Error5,
ErrorID=>ErrorID5);
```

5.3.4 Precautions

- Because the function block works regardless of whether its Enable is TRUE or FALSE, a trigger module should be added before calling Mc_Power.
- In some working environments such as vertical axis, disabling the enabling may cause sharp changes in the axis position. Therefore, be careful when calling this function block.

5.3.5 Refererence

- If an error occurs, refer to Function Block Error Codes.

5.4 MC_Home (Home)

Single axis homing commands.

5.4.1 Parameter

Parameter Type	Title	Description	Empty

	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Active	In control, TRUE when the command is executed	YES	FALSE	TRUE/FALSE	BOOL
	CommandAborted	Termination of Execution	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR

5.4.2 Functions

This command is used for the homing function of a single axis. The origin of the specified axis is determined by detecting the origin switch signal or Z-phase signal of the servo motor. The homing mode is designated by the parameter HomeMode, which follows 15 regression modes in CIA402 protocol: 1/2, 3/5, 7/11, 17/18, 19/21, 23/27, 33/34 and 35.

Before using this command, call the SMC_AxisInit command to set necessary positive and negative limit switch and origin switch parameters. The origin switch signal can be set as servo driver body DI or Controller general DI. For specific configuration methods, please refer to the section of SMC_AxisInit.

This command can be used normally only after the specified axis is switched to the enabled state by using the MC_Power command and the supported axis type is set. This command supports the origin return function of the virtual axis, but does not support the origin return mode related to Z signal when it is set as a virtual axis.

The rising edge of the command is valid. When the rising edge of the command is executed, the input parameters such as locked axis name, origin return mode, speed and acceleration/deceleration are invalid when Execute=TRUE or Execute=FALSE is held. When the MC_Home command is run, the axis status is in Homing. The command input parameter HomeMode is used to set the homing mode, Velocity is used to set the speed of high-speed zero searching, and ApproachVelocity is used to set the speed of low-speed zero searching. The set value of ApproachVelocity should be less than that of Velocity. Acceleration and Deceleration are used to set acceleration and deceleration in the process of homing starting. Jerk is used to configure whether to enable the S-shaped acceleration and deceleration function, and Position is used to define the user's origin position after the homing is completed. After this function block is successfully executed, Busy = TRUE; after the Home command is sent and activated, Active = TRUE; after the normal execution of origin returning is completed, Done = TRUE. When this command is valid, the axis is in Homing state. The MC_Stop command that can make the axis in Stopping state can interrupt the MC_Home command. CommandAboarted=TRUE after the command is interrupted. After finding the origin by returning to the original point, you can check the completion status of homing through the IsHomed pin of the MC_ReadAxisInfo command.

When the same MC_Home command function block is enabled repeatedly, the new Home command will be bufferd, and the command function block will lose monitoring of the previous command and monitor the execution of the newly delivered command instead.

Origin: motor Z signal

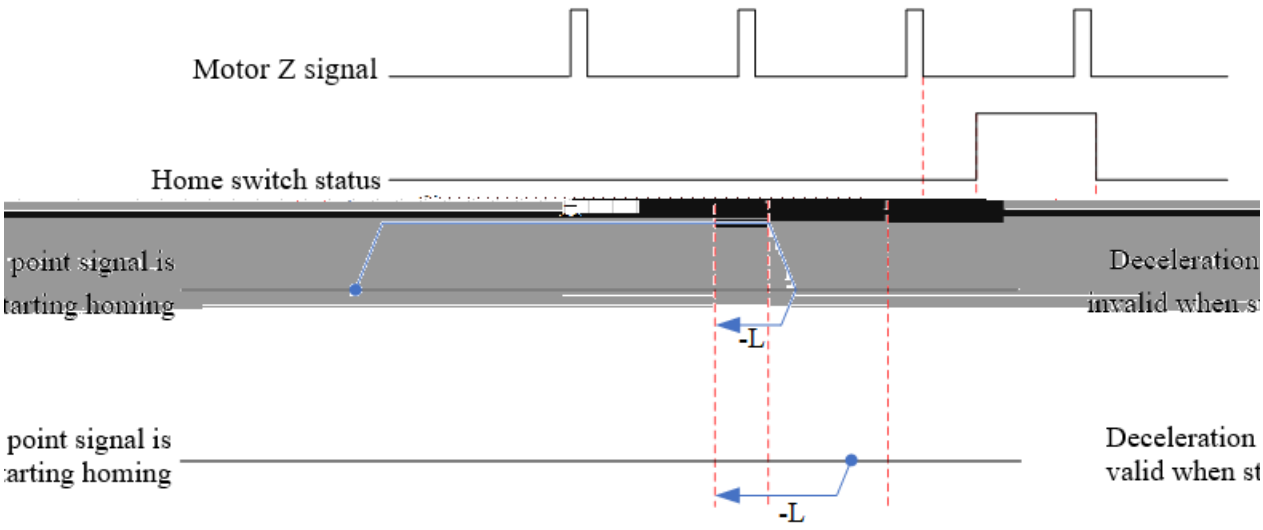
Deceleration point: negative limit switch

Motion track 1: If the negative limit switch status is invalid when starting homing, it moves towards the negative direction at high speed. After encountering the rising edge of the negative limit switch, it decelerates and stops, and then changes to low speed to move towards the positive direction. After encountering the falling edge of the negative limit switch when moving forward at low speed, continue to find the nearest Z-signal position as the origin at low speed.

negative direction at low speed. After encountering the falling edge of the positive limit switch in the negative direction, continue to find the nearest Z-signal position as origin at low speed in the negative direction.

In this mode, if the negative limit switch is in a valid state, the process of returning to the original point will be suspended, that is, the command will be interrupted.

The homing process for mode 2 is shown in the figure below:

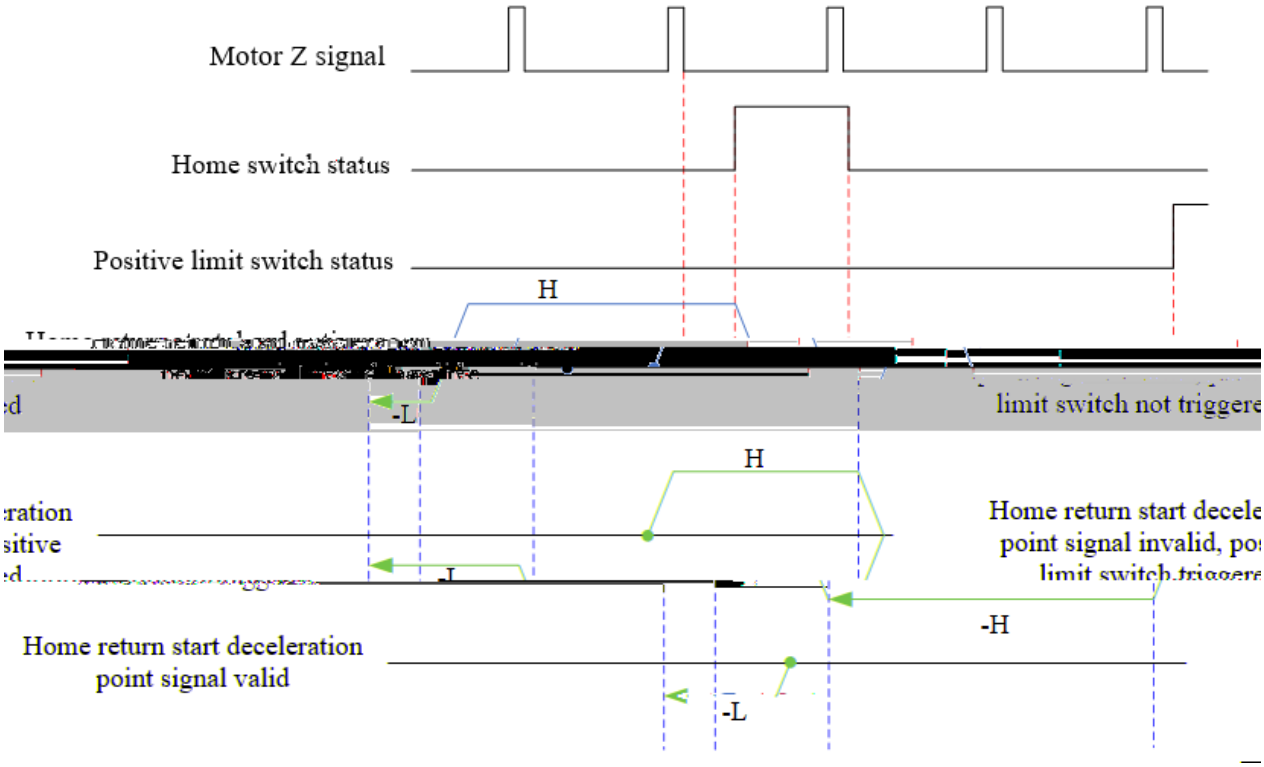


(4) Mode 5, seek for the falling edge position and Z signal of the origin switch when moving forward.

Origin: motor Z signal

Deceleration point: origin switch

Motion track 1: If the origin switch is invalid when homing starting, it will move towards negative direction at high speed. When homing in the negative direction, decelerate and stop after encountering the rising edge of the origin switch, and then move to the positive direction at a low speed. After encountering the falling edge of the origin switch during low-speed forward motion, continue to find the noon, (ch)]TJ op22-8.1 (i)-8.9 (n7)-12.3 (l)3.1 (er) 12.3 (eed-8 (i)3.1 (2)-13.2 (o)-.1 (on)-12.3



(6) Mode 11, seek for the falling edge position and Z signal of the origin switch when moving forward, and automatically reverse in case of negative limit.

Origin: motor Z signal

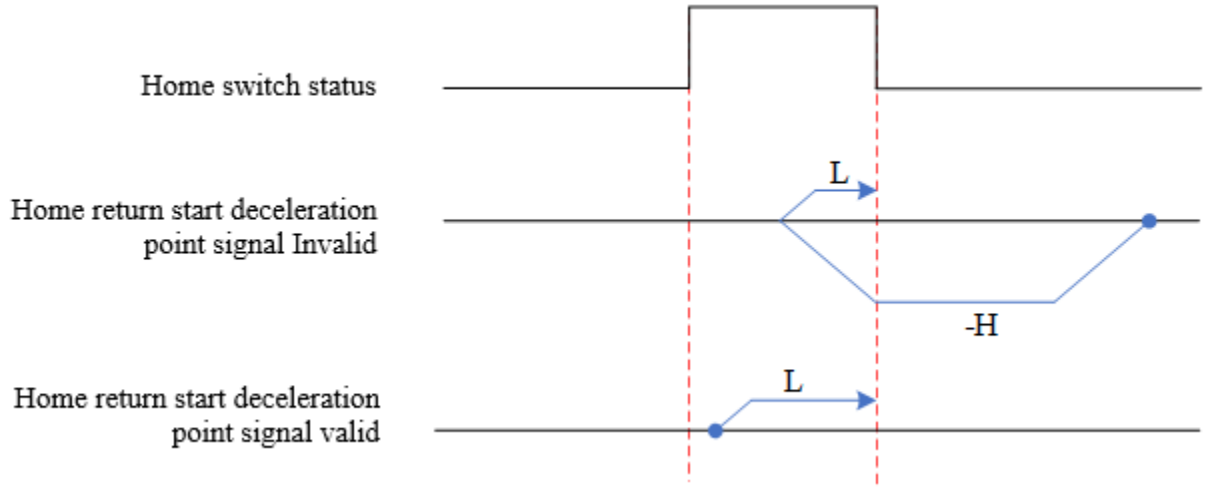
(8) Mode 18: Seek for positive limit switch.

Origin: positive limit switch

Deceleration point: positive limi t switch

Motion track 1: If the positive limit switch state is invalid when starting back to zero, it moves towards twh1.7 1 (o)-12.3 (w)9.2 i3.1d1h1.7 1.1 ( )-1e8 (t)10. (o)-12.3 (w)9.2 -1.akwe potctarkitinchmit swti-
oswtve t1.2-8 (k)-20.e (t)-1 (h12.3 (w)9.2 -1.a1 (h.)]T1 (t1-24. (d)-12

The homing process of Pattern 21 is shown in the following figure:



(11) Mode 23: Seek for the falling edge position of the origin switch when it moves towards the negative direction, and automatically reverse in case of positive limit.

Origin: origi n switch

Deceleration point: origin switch

Motion track 1: If the origin switch state is invalid when homing starting, it moves at high speed towards the positive direction. When moving in the positive direction, it decelerates and stops after encountering the rising edge of the origin switch, and then changes to low speed to move towards the negative direction. During low-speed movement towards the negative direction, find the position where the falling edge of the home switch is located as the origin.

Motion track 2: If the origin switch state is invalid when homing starting, it moves towards the positive direction at high speed. When moving in the positive direction, it decelerates and stops when encountering the valid state of the positive limit switch, and then moves towards the negative direction at high speed. When homing in the negative direction, it starts to decelerate to low speed after encountering the rising edge of the origin switch, and then continues to move towards the negative direction at low speed. During low-speed movement towards the negative direction, find the position where the falling edge of the home switch is located as the origin.

Motion track 3: If the origin switch state is valid when homing starting, it moves towards negative direction at low speed. During low-speed movement towards the negative direction, find the position where the falling edge of the home switch is located as the origin.

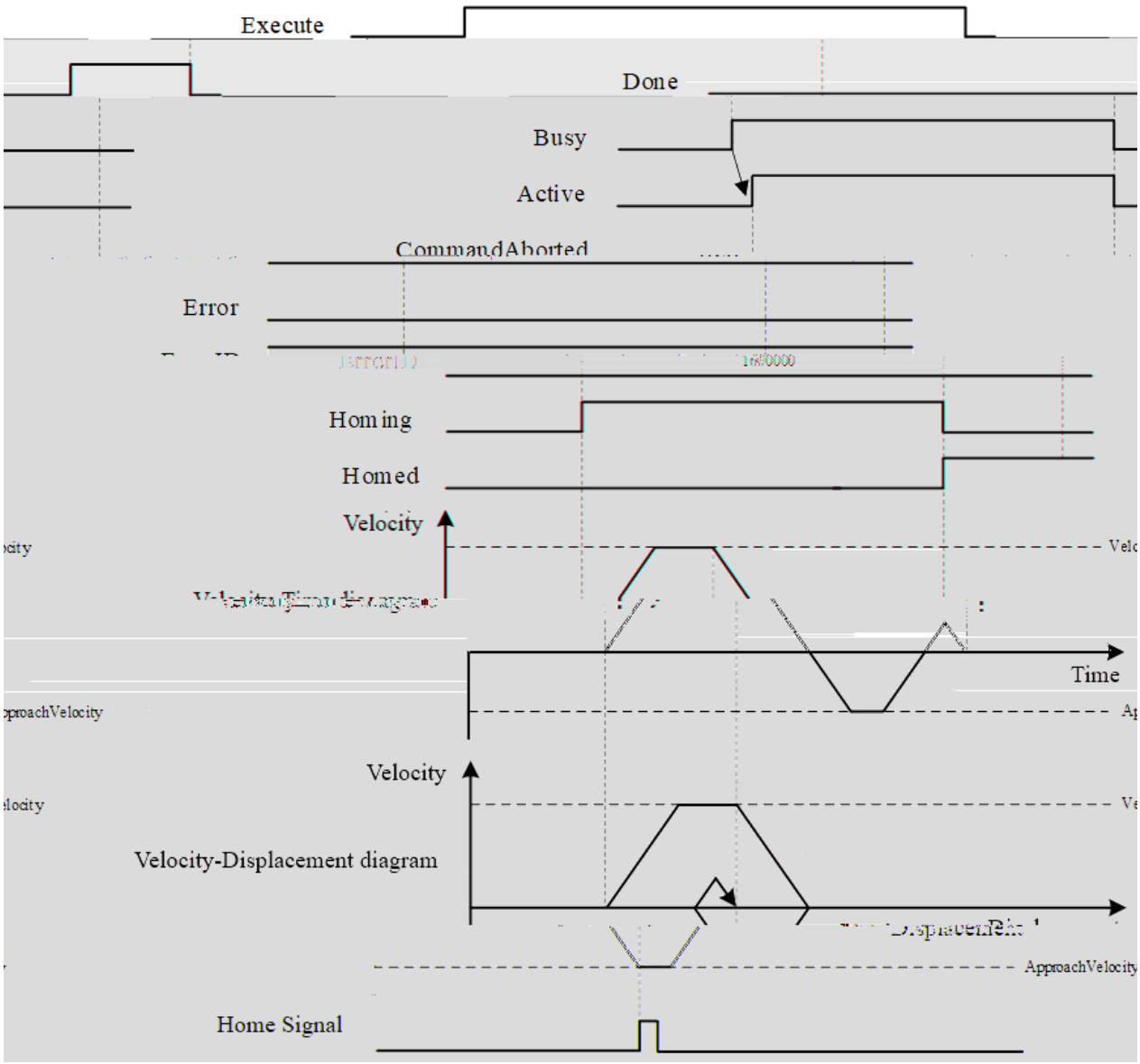
In this mode, when the positive movement first encounters the valid state of the positive limit

(12) Mode 27: Seek for the falling edge position of the origin switch when moving in positive direction, and automatically reverse in case of negative limit.

Origin: origin switch

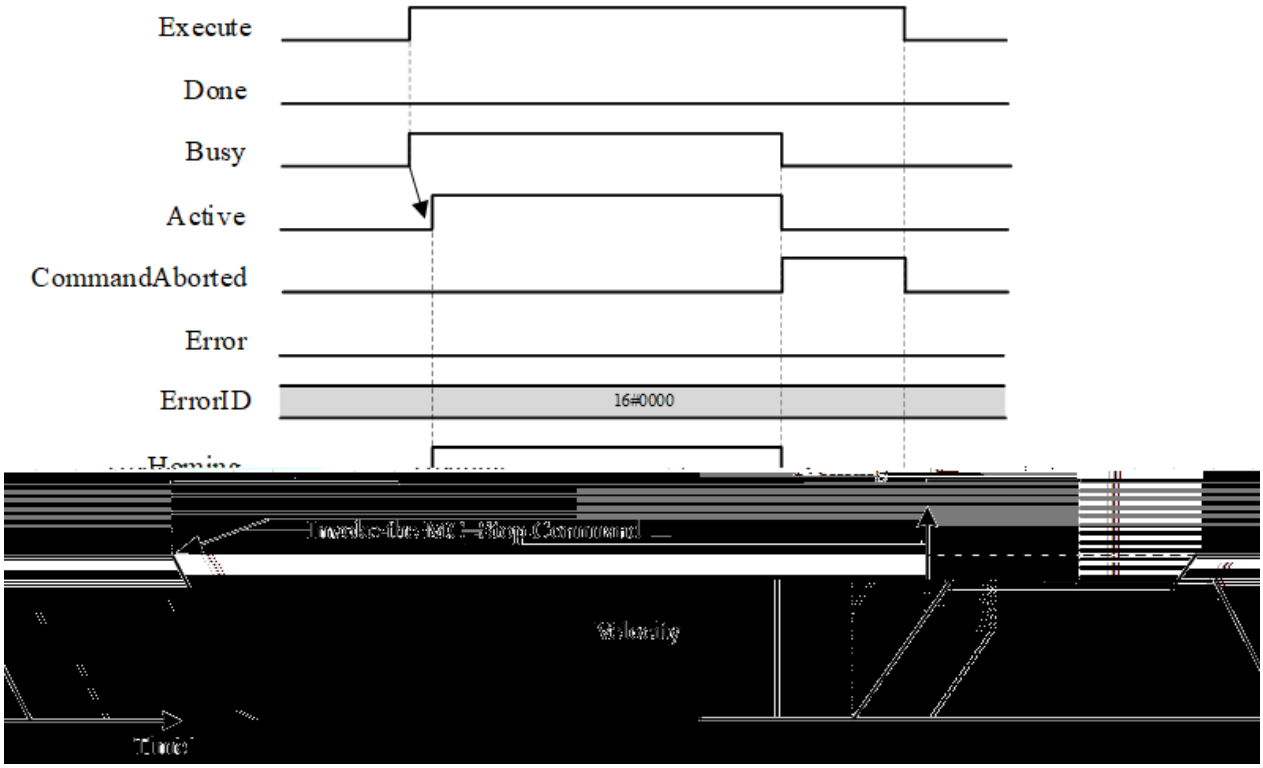
Deceleration point: origin switch

Motion track 1: If the origin switch state is invalid when homing starting, it moves at high speed towards the negative direction. When homing in the negative direction, it decelerates and stops

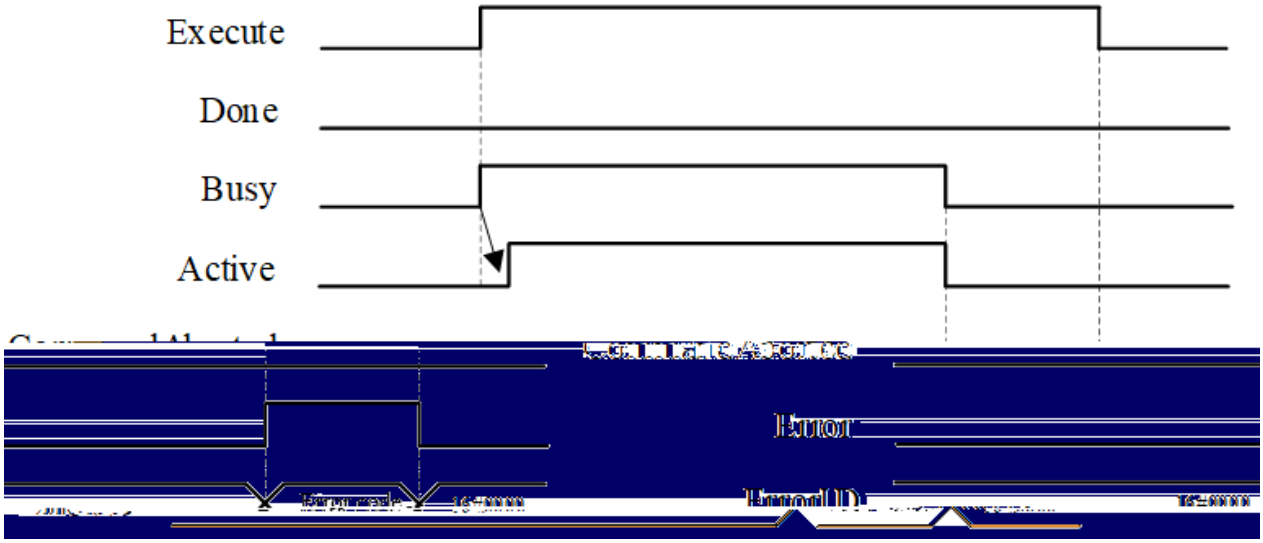


(2) The Timing diagram after the origin return action is interrupted by calling MC_Stop command in the process of homing.

When MC_Stop is called to interrupt, the deceleration set by the input parameter of MC_Stop will be used.



(3) Timing diagram of failure in the homing process.

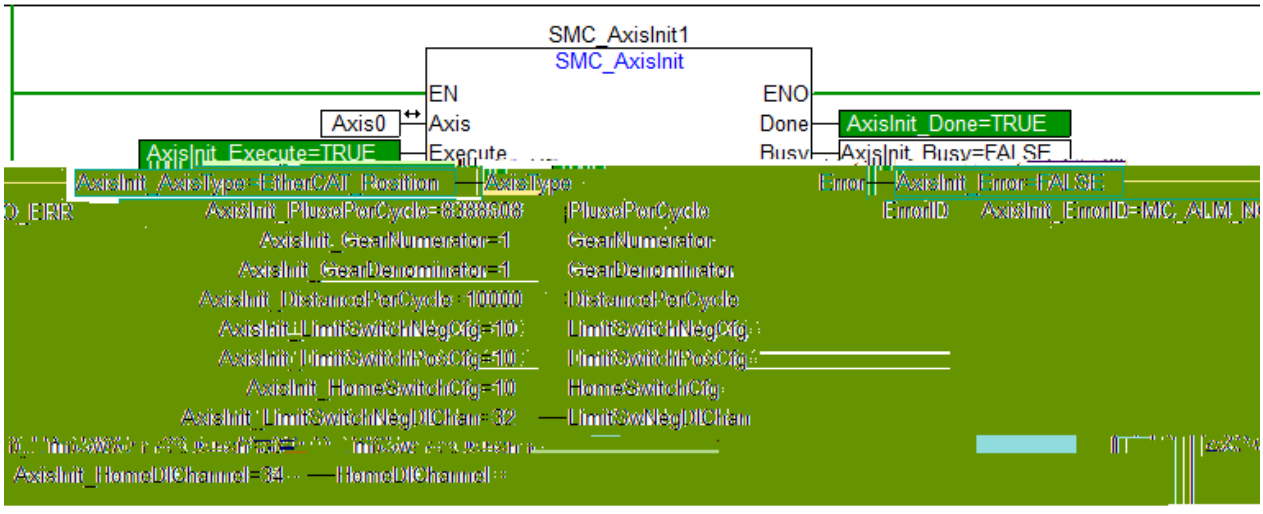


(4) Timing diagram of error, interruption by MC_Stop and normal execution completion during the operation of functional block when Execute is maintained for a cycle.

2	Home_En	BOOL	FALSE	Home enable mark
3	Home_HomeMode	SMC_HOME_MODE	MC_HM1_NEG_LIMIT_Z	Home mode
4	Home_Vel	LREAL	10000	Home speed for finding deceleration point at high speed
5	Home_Vel2	LREAL	5000	Home low-speed origin finding speed
6	Home_Acc	LREAL	10000	Home acceleration
7	Home_Dec	LREAL	10000	Homing deceleration
8	Home_Jerk	LREAL	0	Home jerk
9	Home_Pos	LREAL	0	Zero position after homing
10	Home_Done	BOOL	FALSE	Home completion flag
11	Home_Busy	BOOL	FALSE	Home busy status flag
12	Home_Active	BOOL	FALSE	Home command operation mark
13	Home_Abort	BOOL	FALSE	Home abort flag
14	Home_Error	BOOL	FALSE	Home error flag
15	Home_ErrorID	SMC_ERROR	MC_ALM_NO_ERR	Home error code
16	MC_MoveAbsolute	MC_MOVEABSOLUTE	-	Absolute position positioning command
17	MoveAbs_En	BOOL	FALSE	Absolute position positioning enable mark
18	MoveAbs_Pos	LREAL	20000	Absolute position locating target position
19	MoveAbs_Vel	LREAL	10000	Absolute position positioning speed
20	MoveAbs_Acc	LREAL	100000000	Absolute position positioning acceleration
21	MoveAbs_Dec	LREAL	100000000	Absolute position positioning deceleration
22	MoveAbs_Dir	MC_DIRECTION	mcPositiveDirection	Absolute position positioning direction
23	MoveAbs_Jerk	LREAL	0	Absolute position positioning jerk
24	MoveAbs_Buff	MC_BUFFER_MODE	Aborting	Absolute position positioning buffer mode
25	MoveAbs_Done	BOOL	FALSE	Absolute position positioning completion flag
26	MoveAbs_Busy	BOOL	FALSE	Absolute position positioning busy status flag
27	MoveAbs_Active	BOOL	FALSE	Absolute position positioning command active flag
28	MoveAbs_Abort	BOOL	FALSE	Absolute position positioning abort flag
29	MoveAbs_Error	BOOL	FALSE	Absolute position positioning error flag
30	MoveAbs_ErrorID	SMC_ERROR	MC_ALM_NO_ERR	Absolute position positioning error codes

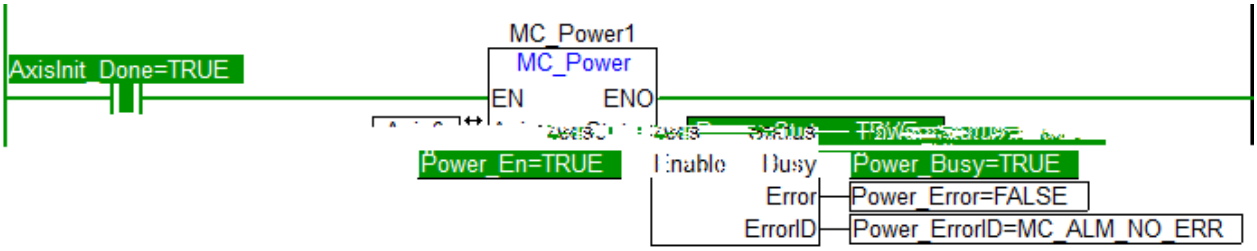
LD program implementation

(1) First, call the SMC_AxisInit command to initialize the axis parameters, set the axis type as EtherCAT_Poaition, and set the user unit, origin switch signal and positive/negative limit switch signal. The commands are as follows:

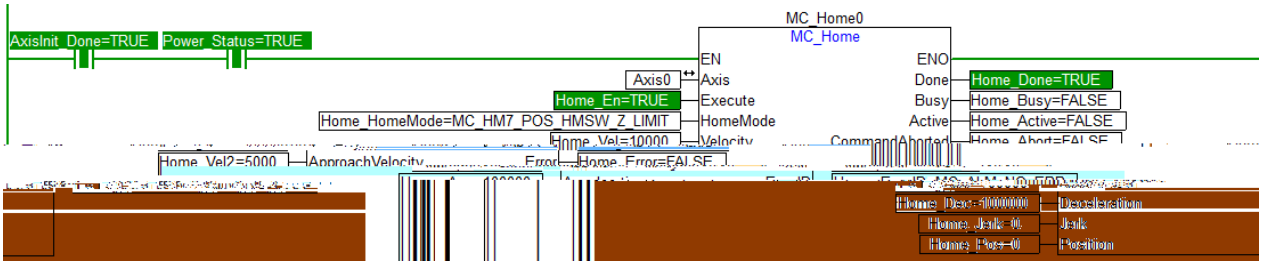


(2) Before calling the MC_Home command, call the MC_Power command to enable the

corresponding axis.



(3) Call the MC_Home command for homing operation, and set the homing mode to 7.



(4)

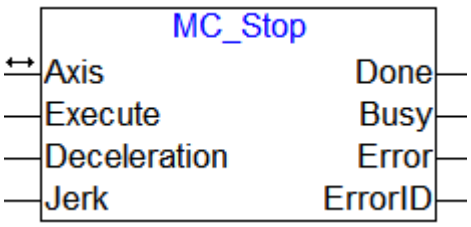
```
    Execute := MoveAbs_En,
    Position := MoveAbs_Pos,
    Velocity := MoveAbs_Vel,
    Acceleration := MoveAbs_Acc,
    Deceleration := MoveAbs_Dec,
    Jerk := MoveAbs_Jerk,
    Direction := MoveAbs_Dir,
    BufferMode := MoveAbs_Buff,
    Axis := Axis0,
    Done=>MoveAbs_Done,
    Busy=>MoveAbs_Busy,
    Active=>MoveAbs_Active,
    CommandAborted=>MoveAbs_Abort,
    Error=>MoveAbs_Error,
    ErrorID=>MoveAbs_ErrorID);
END_IF
```

5.4.4 Precautions

When this command is set to the homing mode related to Z signal, if the probe capture channel 1 has been occupied, the command reports an error. The MC_TouchProbe and HMC_GantryInit commands will occupy the probe capture channel. During use, the probe capture channel 1 is used for Z signal capture in a staggered manner.

5.5 MC_Stop (Axis Stop)

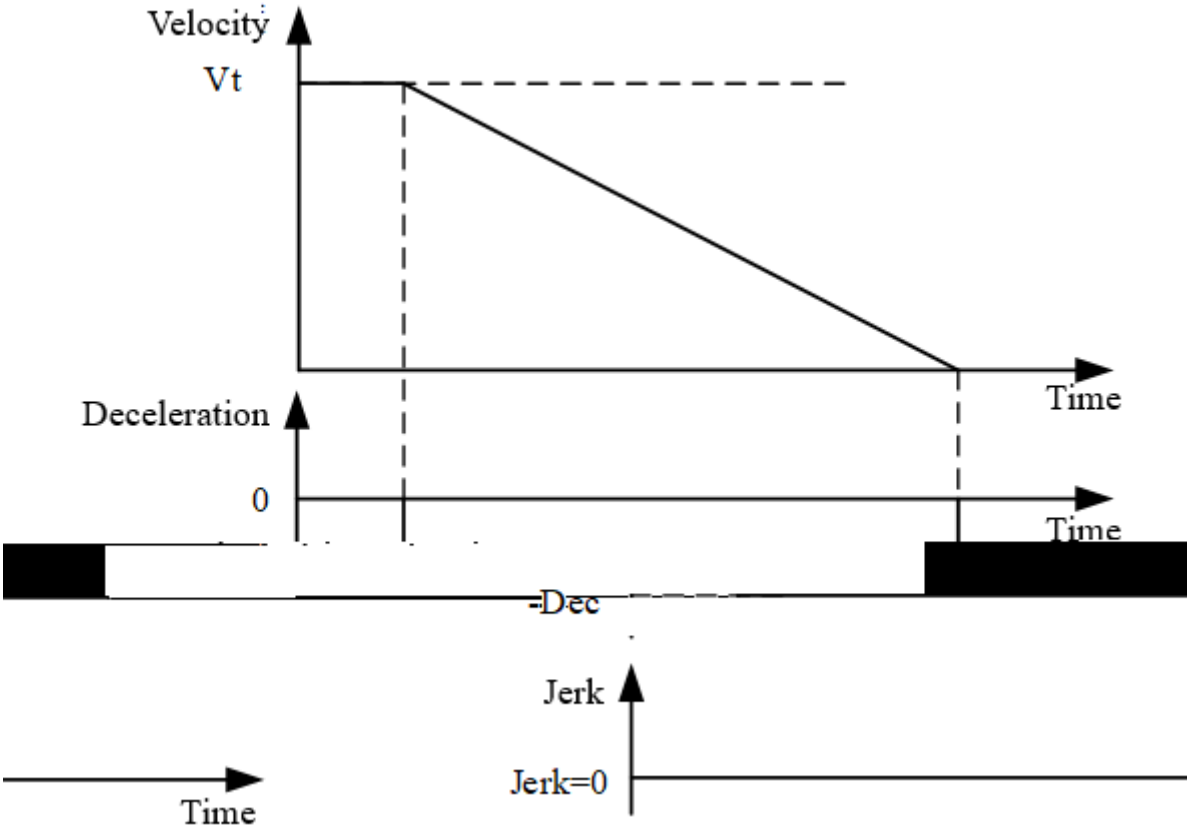
The control axis stops at the specified deceleration, any motion control command being executed on this axis is suspended, and all motion buffer commands on this axis are canceled.



5.5.1 Parameter

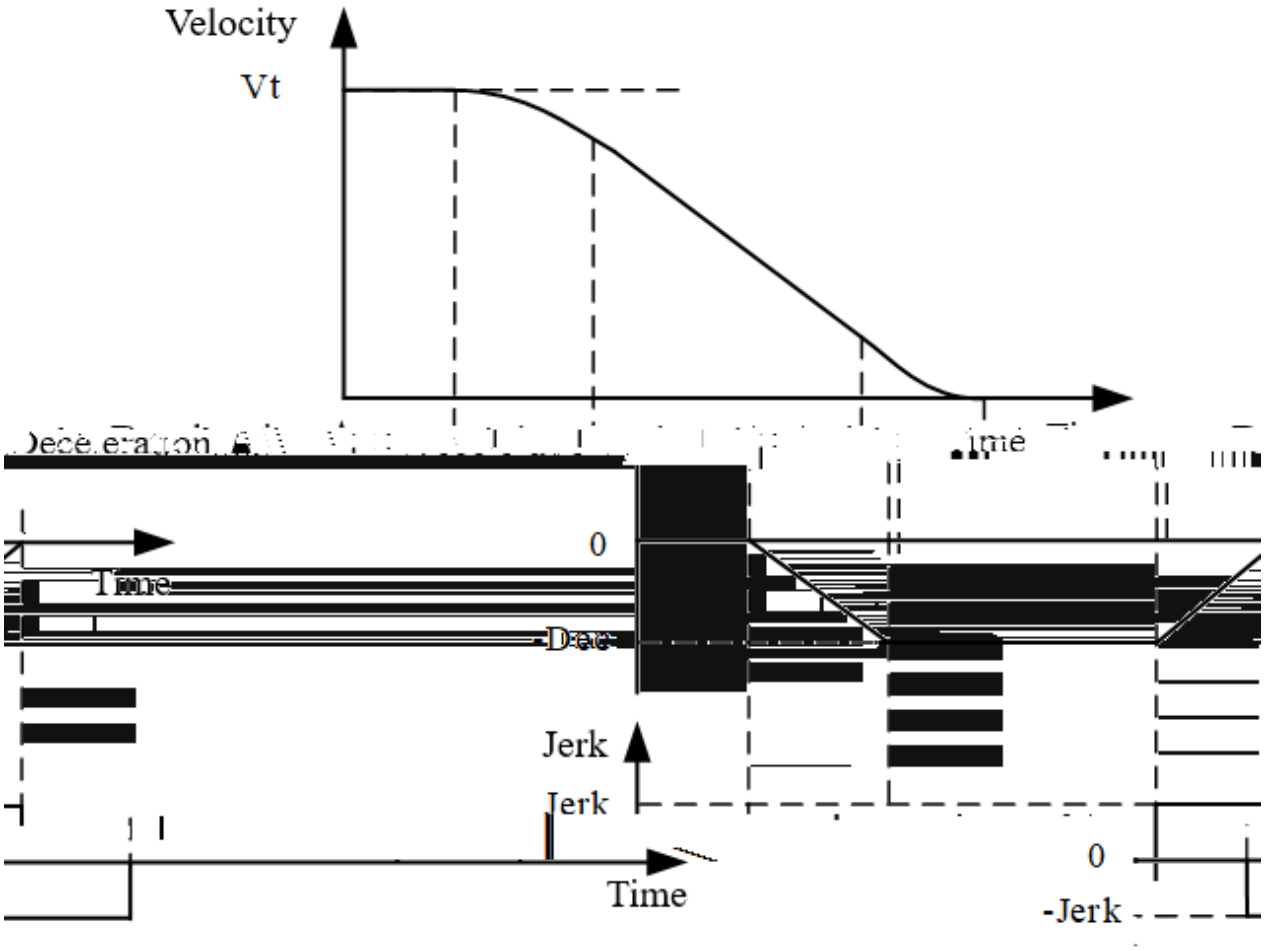
Parameter Type	Name	Description	Empty or	Default	Scope	Data Type
IN_OUT	Axis	Axis name	mo	-	-	AXIS_REF
INPUT	Execute	Rising edge trigger	mo	-	TRUE/FALSE	BOOL

	Deceleration	Deceleration	No	-	Positive value	LREAL
	Jerk	Jerk 0: Trapezoidal acceleration and deceleration Positive: S-shaped acceleration and deceleration	YES	0		



Vt: Velocity at which deceleration starts, Dec: set value of deceleration, Jerk: set value of jerk

- When Jerk>0, the command value of speed is generated by deceleration. The deceleration curve is an S-shaped acceleration/deceleration curve as follows:



Vt: Velocity at the start of deceleration, Dec: Deceleration setting value, Jerk: Jerk setting value

Axis types supported by this command:

Virtual axis (0): virtual axis

EtherCAT_Position(50): EtherCAT bus connection axis, position control

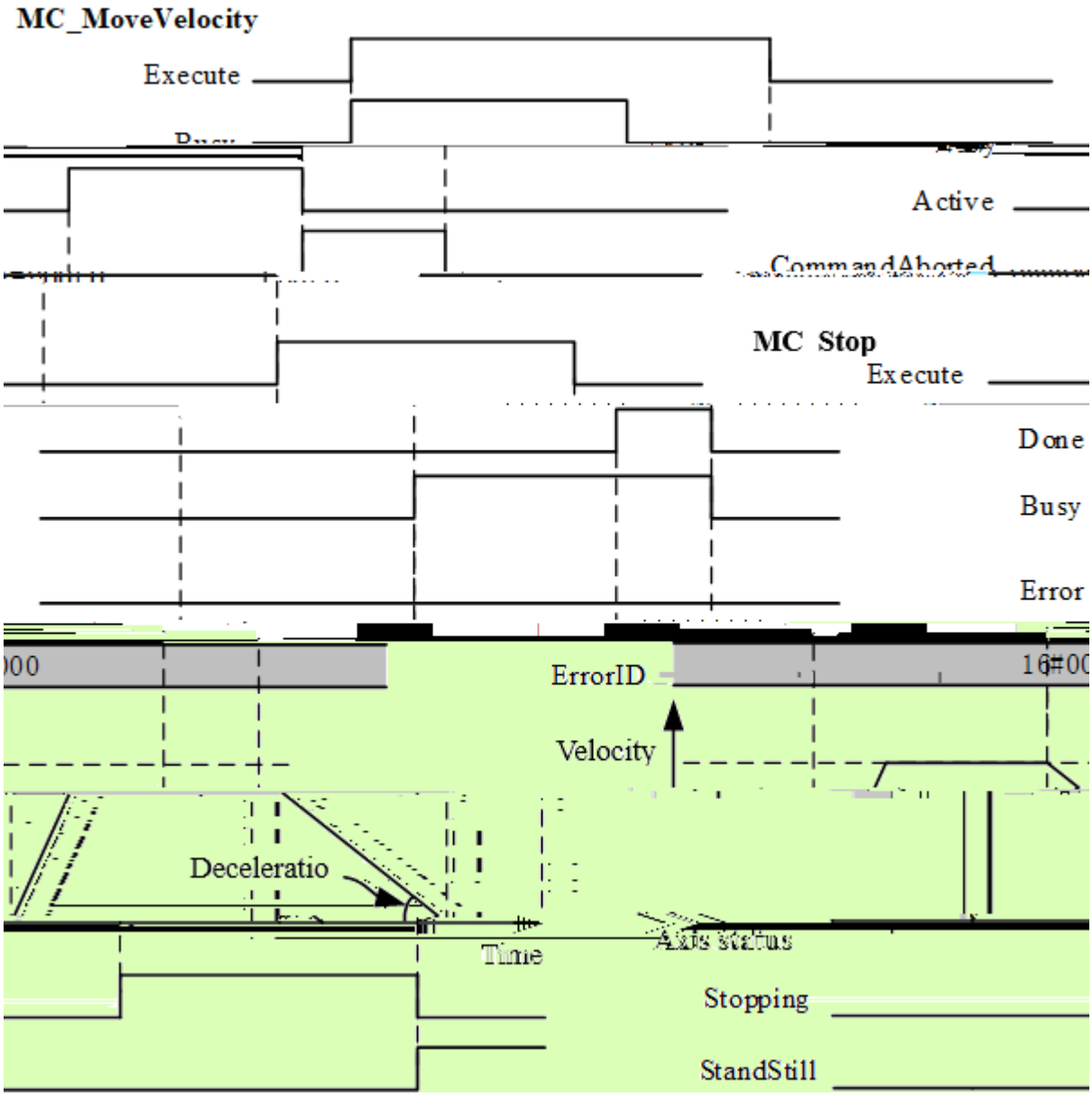
EtherCAT_Speed(51): EtherCAT bus connection axis, speed control

EtherCAT_Torque(52): EtherCAT bus connecting shaft, torque control

Timing diagram

(1) Normal timing diagram

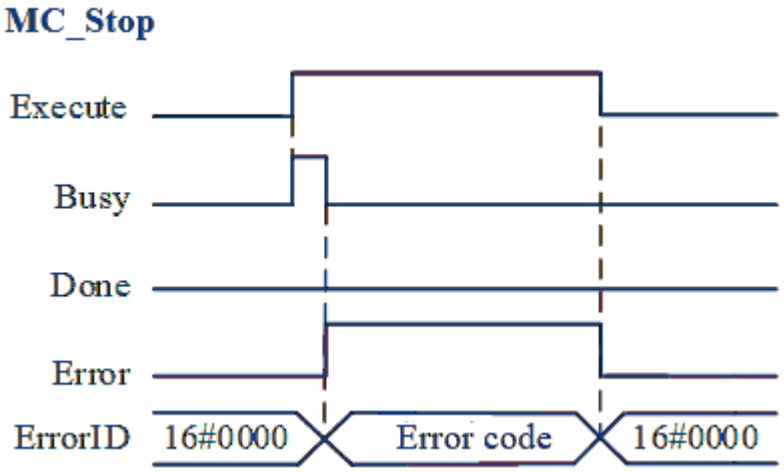
On the rising edge of Execute input, if there is no abnormality, the Busy pin is set to TRUE. When



(2) Pin timing when error occurs

When this command is executed, if the command input parameter is illegal, the axis is in Disabled status or ErrorStop status and other abnormalities occur, the call of this command will fail. At this time, the command output pin Error=True, and the error code ErrorID reports an abnormal error value. Please refer to the appendix Description of error codes to understand the cause of the error.

When the command input pin Execute is low, all output pins are restored to their Default Values.

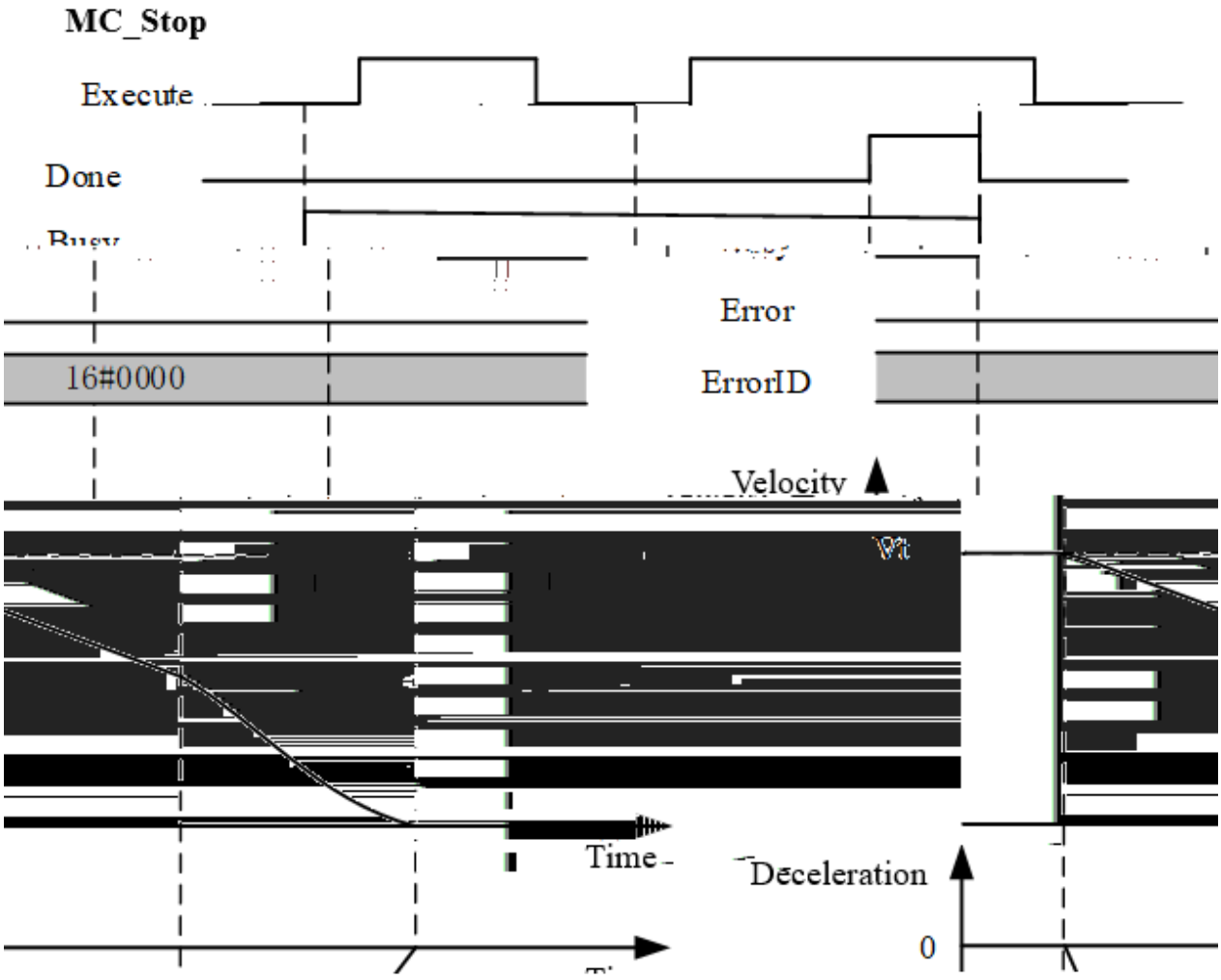


When the command error is solved, it is necessary to trigger a rising edge on the Execute input to initiate the MC_Stop function block.

(3) Repeat delivery command

When multiple MC_Stop functional blocks are triggered repeatedly, if there is a change in Deceleration and Jerk, the deceleration-stop control will be performed with the deceleration and jerk of the last delivery command, and the command will change the deceleration and jerk.

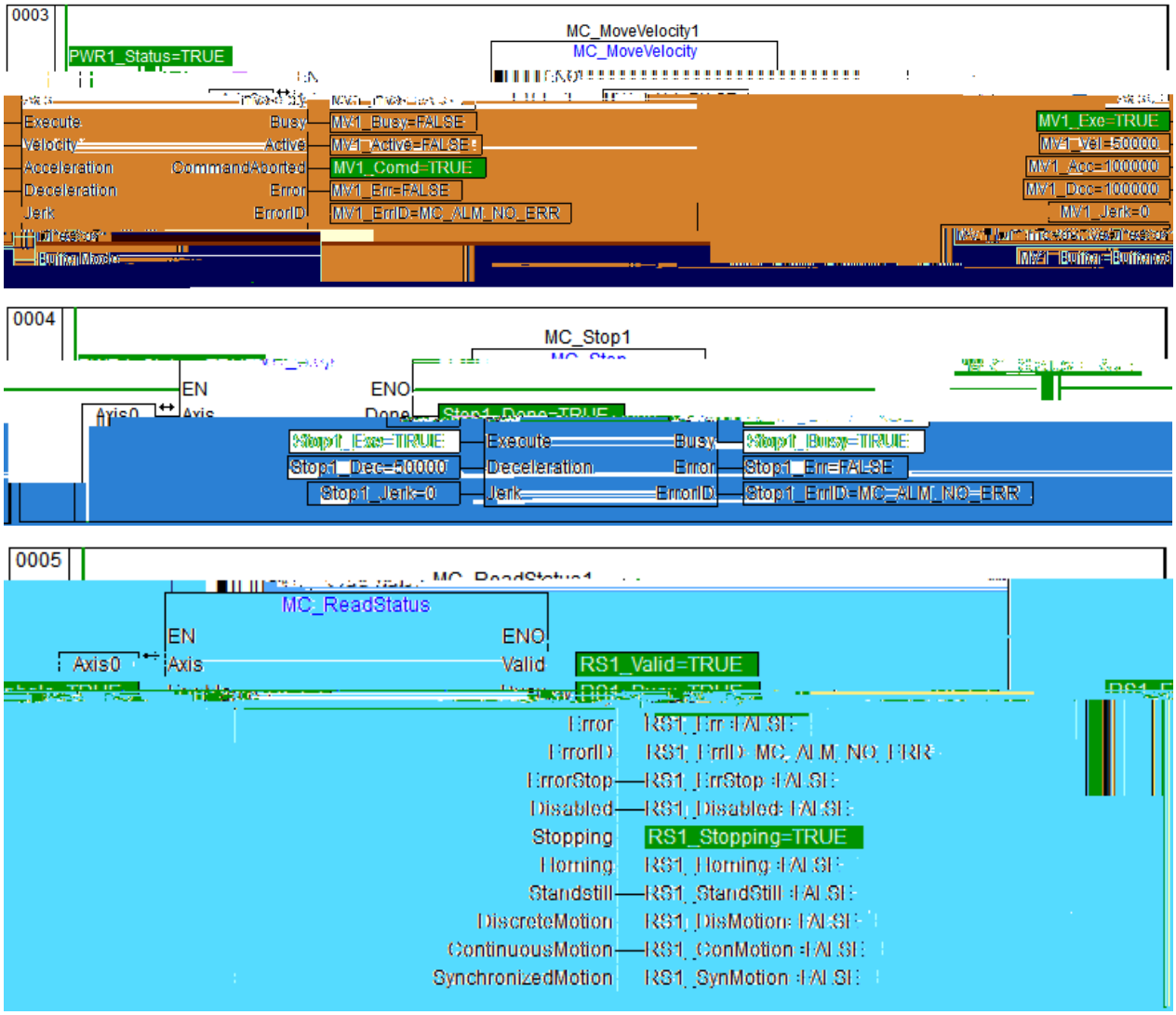
The following figure shows the timing diagram for a repeat delivery command.



Vt: Velocity at the start of deceleration, Dec1: Deceleration during the first command execution, Dec2: Deceleration during the second comma nd execution, Jerk1: Jerk during the first command execution, Dec2: Jerk during the second command execution

5.5.3 Examples

In this example, Axis0 is set as the command axis and the Variable Type is AXIS_REF. The MC_MoveVelocity motion command is stopped with the MC_Stop command, which reads the current axis



ST language

Use ST language to set up the program, input parameters Deceleration 10000, Jerk 0, trapezoidal acceleration and deceleration.

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
Axis0	AXIS_REF	-	Axis variable for axis 0
Power1_Status	BOOL	FALSE	Variable for which the axis enable is successful. This variable becomes TRUE when the axis is enabled successfully.
MV1_Active	BOOL	FALSE	Variable when the axis speed control command acts. This variable becomes TRUE when the axis is controlling motion.
Stop1_Exe	BOOL	FALSE	Trigger variable for the rising edge of an axis stop command.
Stop1_Done	BOOL	FALSE	Variable at which the axis was successfully stopped. Done = TRUE, axis stop is finished.
Switch1	BOOL	FALSE	Axis ready state variable. After axis initialization, Switch to TRUE and the axis is ready.
InitFlag	BOOL	FALSE	If this variable is TRUE, the initialization parameter setting is completed.

(1) Axis initialization

Initial configuration of Axis0 with axis type set to 50:EtherCAT_Position. For the usage of SMC_AxisInit command, please refer to its command Description section.

```
Execute := MV1_ Exe,
Velocity := MV1_ Vel,
Acceleration := MV1_ Acc,
Deceleration := MV1_ Dcc,
```

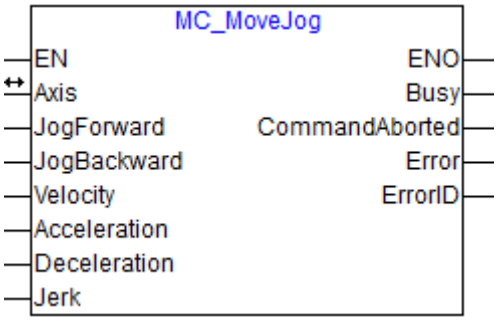
```
ContinuousMotion=> RS1_ConMotion,
SynchronizedMotion=> RS1_SynMotion);
```

The schematic diagram of MC_MoveVelocity motion command deceleration stop is as follows:

5.5.4 Precautions

The MC_Stop command is successfully executed. If the input parameter Execute keeps high level, the axis remains in Stopping status and motion command delivery is prohibited.

When the input parameter Execute is at low level, the stop function will be released and the axis will snshe s1.1 (f)-1o18.4 (y)1.1 (n5)-12. (c)-8 (us)-8-8 (us).1 (n)-12.2 (S)-11.4 (w)9.2 (i)- ( )0.9 (n)-12. (c)-8 (us)t(t)1.2



5.6.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	JogForward	Positive enable	No	-	TRUE/FALSE	BOOL
	JogBackward	Negative enable	No	-	TRUE/FALSE	BOOL
	Velocity	Target speed	No	-	Non-negative	LREAL
	Acceleration	Acceleration	No	-	Positive value	LREAL
	Deceleration	Deceleration	No	-	Positive value	LREAL
	Jerk	0: Trapezoidal acceleration and deceleration Positive: S-shaped acceleration and deceleration acceleration	YES	0	0/positive	LREAL
OUTPUT	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	CommandAborted	Termination of Execution	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR

5.6.2 Functions

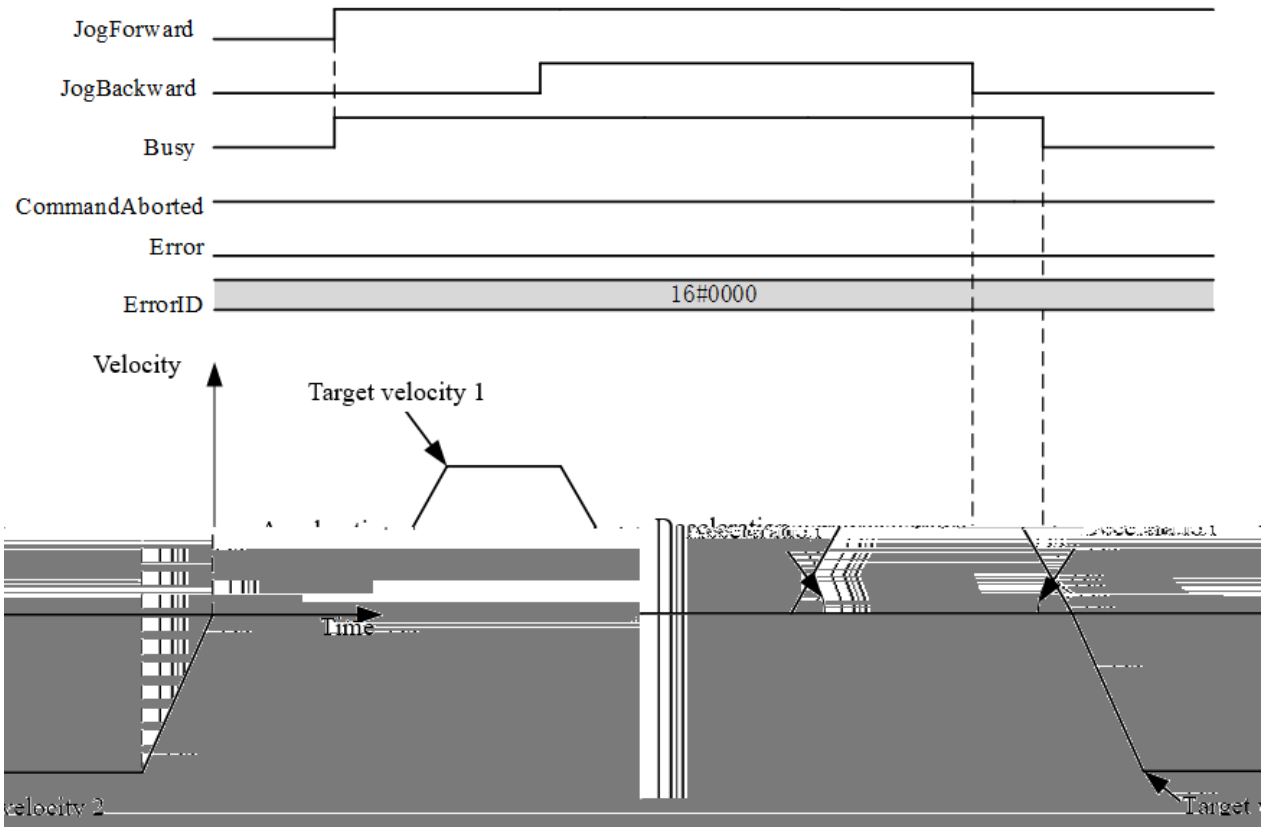
This command can carry out positive and negative inching motion according to the set speed, acceleration and deceleration, jerk and other parameters. The detailed Description of this command is as follows:

(1) High level triggering of this command is valid, and the axis moves according to the set parameters. During execution, the modified latched parameters are invalid until high level takes effect after being triggered again. When JogForward (positive enable) is set as TRUE, the axis moves in positive direction; when JogBackward (negative enable) is set as TRUE, the axis moves in negative direction.

(2) When JogForward and JogBackward are both TRUE, the axis moves in positive direction.

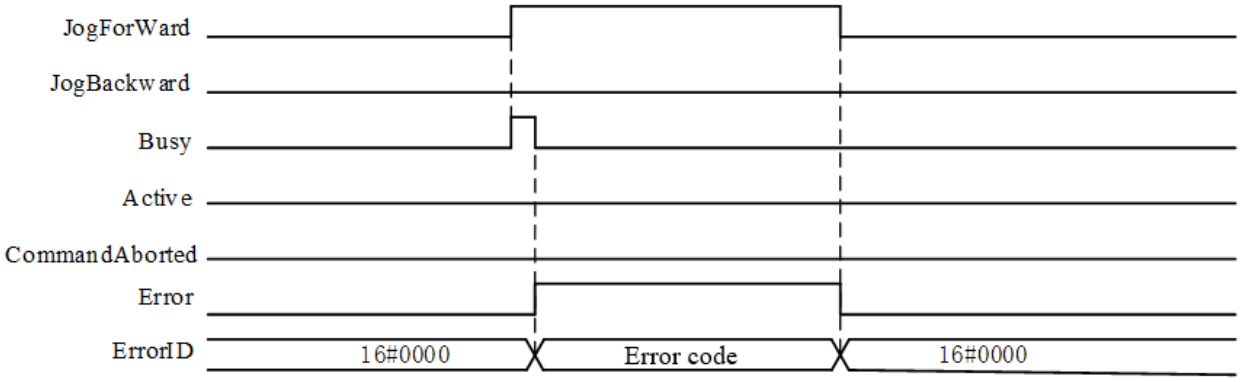
(3) When JogForward is changed from TRUE to FALSE, the axis decelerates and stops according to the set deceleration. During deceleration, since the axis does not stop, Busy (busy flag) will not change to FALSE, which is also the case in negative direction.

(3) Timing diagram of restart in different directions



(4) T

The value of ErrorID can be viewed to understand the specific cause of the error.



5.6.3 Examples

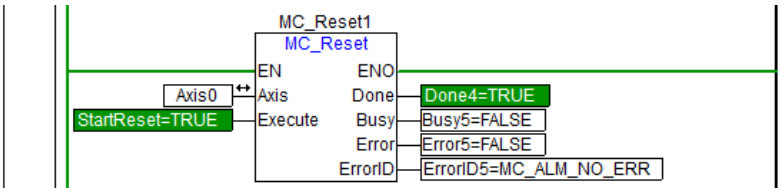
Set up MC_MoveJog sample program to run simulated velocity motion command.

Definition of Primary Variable

Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0

(2) After the axis is enabled, you can enter specified motion parameters in the input variables of the function block to send velocity motion commands and repeat commands. It can be seen that when repeated sending occurs, the next function block will interrupt the previous one.

(3) When there is an error in the axis, the MC_Reset function block can be called for reset processing.



ST language program

(1) Initialize and enable the axis

```
(The * axis is initialized*)
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
```

ac1596 0 0.9

```
ErrorID=>ErrorID1);
(*Enable the axis*)
```

```
Busy=>Busy5,
Error=>Error5,
ErrorID=>ErrorID5);
```

5.6.4 Precautions

It is not allowed to input 0 value for acceleration and deceleration in the input variables.

When the positive enabling and negative enabling are both TRUE, the positive enabling takes precedence, so it is necessary to determine the motion direction before use.

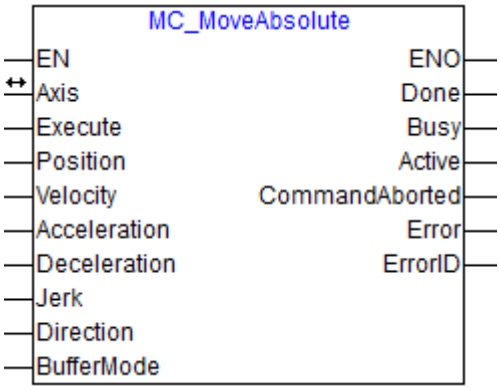
5.6.5 Reference

If an error occurs, refer to the appendix Function Block Error Codes.

For information about initialization and enabling, please refer to the section of Initialization and Enabling.

5.7 MC_MoveAbsolute (Absolute Positioning)

Single-axis absolute positioning command. The control axis moves to the specified absolute position.

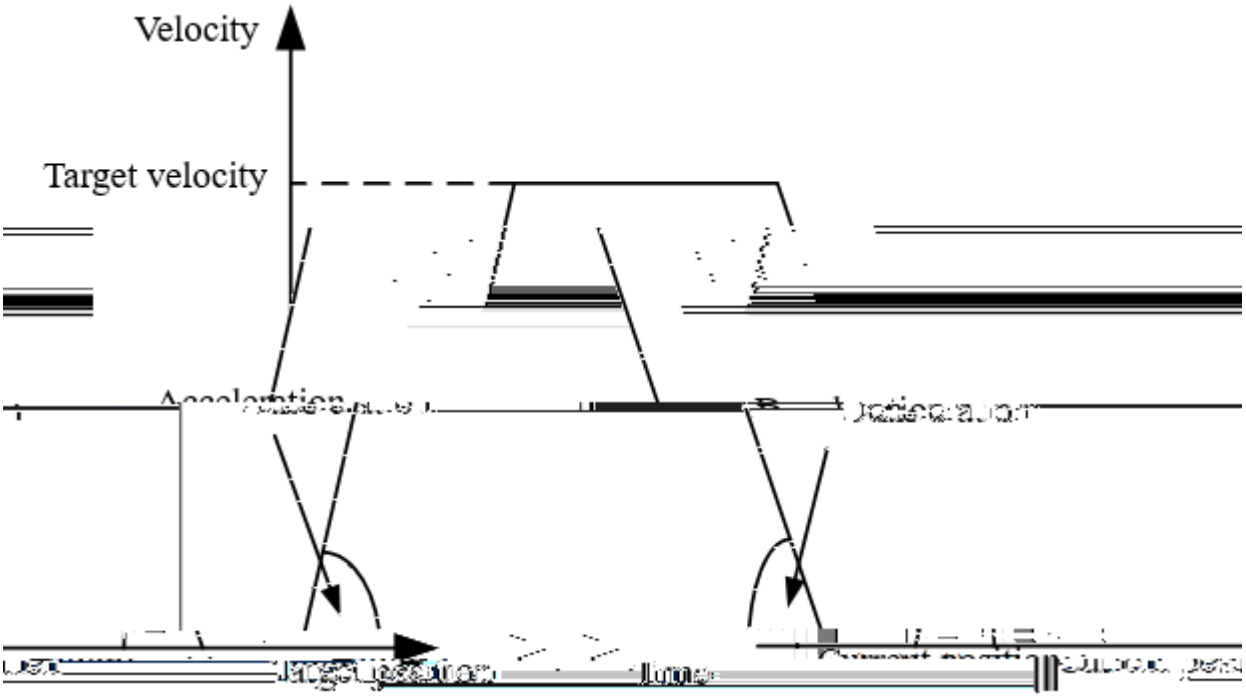


5.7.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT						

	Jerk	0: Trapezoidal acceleration and deceleration Positive: S-shaped acceleration and deceleration acceleration	YES	0	0/positive	LREAL

Direction



Direction (direction selection) introduction

When Direction is mcPositiveDirection, the direction of motion is positive.

When the Direction is mcShortestWay, the shortest distance will be calculated according to the forward and backward movement distances calculated by the algorithm. If they are the same, the direction is positive.

When Direction is mcNegativeDirection, the direction of motion is negative.

When Direction is set to mcCurrentDirection, the command will act along the command direction of the previous action. If the direction of the previous action is positive, the current motion direction is positive; otherwise, it is negative.

When Direction is set to mcNoDirection, the command determines the motion direction based on the values of current position and target position. If the current position is smaller than the target position, the movement direction is positive; if the current position is larger than the target position, the movement direction is positive.

BufferMode

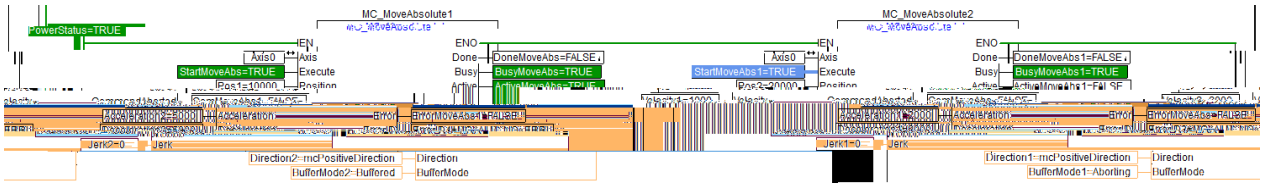
Absolute positioning commands support BufferMode (buffer mode selection), BufferMode supports 0: Aborting, 1: Buffered, 2: BlendingLow, 3: BlendingPrevious, 4: BlendingNext, and 5: BlendingHigh.

When sending this command, if BufferMode is 0: Aborting, it will interrupt the currently executing command. If BufferMode is 1: Buffered, from the cycle when the currently executing command ends normally, the buffered command starts automatically. If BufferMode is 2: BlendingLow, it blends at a low speed between adjacent commands. If BufferMode is 3: BlendingPrevious, it blends at the speed of the previous command. If BufferMode is 4: BlendingNext, it blends at the

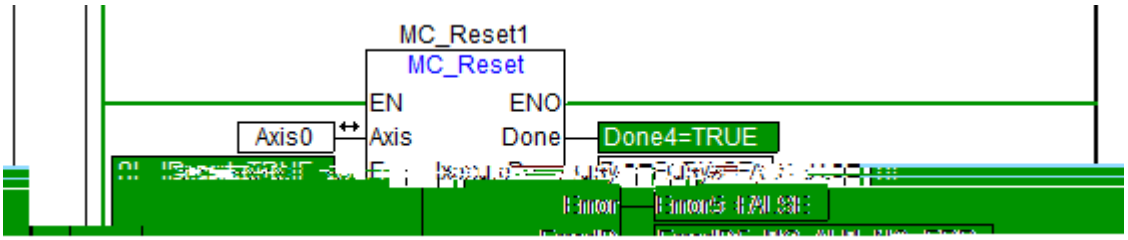
Aborting, and the rising edge triggers the control axis to move. The example procedure is as follows:

(1) First, initialize the axis and set its type and various parameters. After the initialization is completed, enable the axis. When it enters the ON state of servo enable, motion control can be realized for the axis.

(2) After the axis is enabled, specified motion parameters can be entered in the input variables of the functional block to send absolute positioning motion commands and repeat commands.



(3) When there is an error in the axis, the MC_Reset function block can be called for reset processing.



ST language program

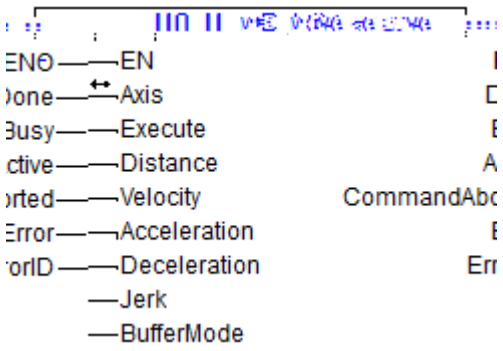
(1) Initialize and enable the axis

```
(The * axis is initialized*)
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
```

```
HomeDIChannel := 2,
Axis := Axis0,
Done=>Done1,
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
(*Enable the axis*)
IF DoneInit THEN
  MC_Power1(
  Enable := PowerEnable,
  Axis := Axis0,
  Status=> PowerStatus,
  Busy=>Busy2,
  Error=>Error2,
  ErrorID=>ErrorID2);
END_IF
```

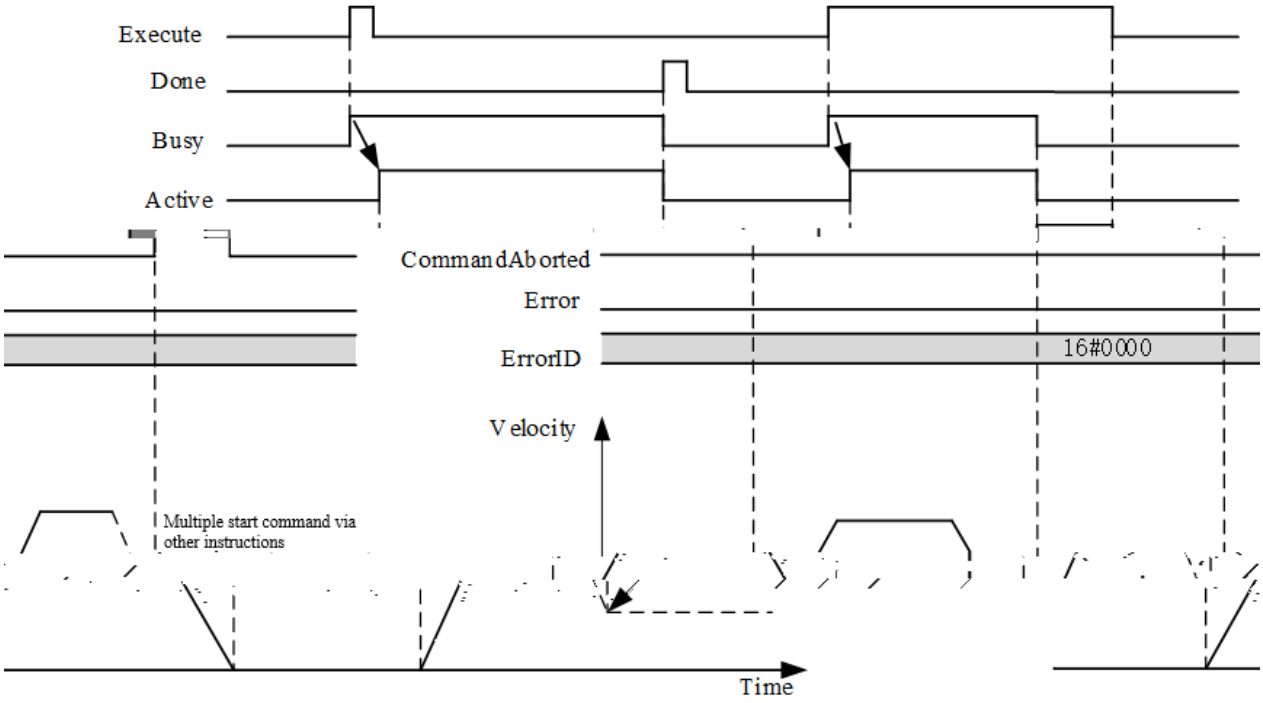
(2) The rising edge triggers the MC_MoveAbsolute command or this command is executed repeatedly to control axis motion.

```
IF PowerStatus THEN
(*Start absolute positioning command*)
MC_MoveAbsolute1(
Execute := StartMoveAbs,
Position := 10000,
Velocity := 1000,
Acceleration := 2000,
Deceleration := 2000,
Jerk := 0,
Direction := 4,
BufferMode := 0,
Axis := Axis0,
Done=> DoneMoveAbs,
Busy=> BusyMoveAbs,
Active=> ActiveMoveAbs,
CommandAborted=> ComMoveAbs,
Error=> ErrorMoveAbs,
ErrorID=>ErrorID3);
(* Repeated start of absolute positioning command *)
MC_MoveAbsolute2(
Execute := StartMoveAbs1,
Position := 50000,
Velocity := 2000,
Acceleration := 5000,
Deceleration := 5000,
Jerk := 0,
Direction := 4,
```



5.8.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default V 9 D 1 -0.



(2) Timing diagram in case of abnormality

When an exception occurs during the execution of this command, Error turns FALSE and the axis stops.

The cause of the exception can be understood by viewing the output value of ErrorID.

5.8.3 Examples

Set up MC_MoveRelative sample program to run relative positioning.

Definition of Primary Variable

Variable Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	Change to TRUE at the start of initialization
DoneInit	BOOL	FALSE	Change to TRUE upon completion of initialization

PowerEnable	BOOL	FALSE	Change to TRUE at the beginning of enabling
PowerStatus	BOOL	FALSE	Change to TRUE when enabled
StartMoveRel	BOOL	FALSE	TRUE when relative positioning command is started
BusyMoveRel	BOOL	FALSE	Relative positioning command delivery completed changes to TRUE
ActiveMoveRel	BOOL	FALSE	Change to TRUE when the command is running

```
ErrorID=>ErrorID3);
```

(3) Repeated start of relative motion command

```
(* Repeated start motion command *)
 MC_MoveRelative2(
 Execute := StartMoveRel1,
 Distance := 20000,
 Velocity := 2000,
 Acceleration := 5000,
 Deceleration := 5000,
 Jerk := 0,
 BufferMode := 0,
 Axis := Axis0,
 Done=> DoneMoveRel1,
 Busy=> BusyMoveRel1,
 Active=> ActiveMoveRel1,
 CommandAborted=> ComMoveRel1,
 Error=> ErrorMoveRel1,
 ErrorID=>ErrorID4);
END_IF
```

(4) Reset in case of error

```
(*Reset processing can be performed in case of error *)
MC_Reset1(
Execute := StartReset,
Axis := Axis0,
Done=>Done4,
Busy=>Busy5,
Error=>Error5,
ErrorID=>ErrorID5);
```

5.8.4 Precautions

When entering the parameter Distance (movement distance), it is not allowed to enter a value of 0, otherwise an error will be reported.

If the positioning is completed before the functional block module is restarted, the position at the time of restarting shall be the relative position of the reference for positioning.

If the positioning is not completed before the function block module is restarted, when the restart of the function block interrupts the restart, the position at the time of command execution will be taken as the reference.

5.8.5 Reference

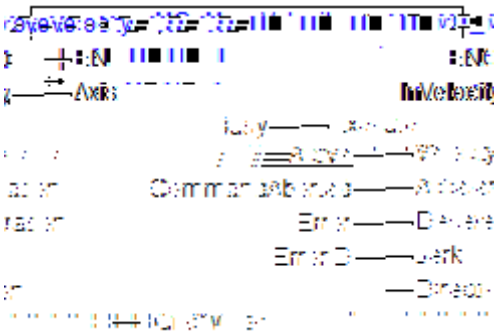
If an error occurs, refer to the appendix Function Block Error Codes.

Refer to the MC_MoveAbsolute function block for its use.

BufferMode (buffer mode selection)Refer to the appendix Description of Buffer Mode.

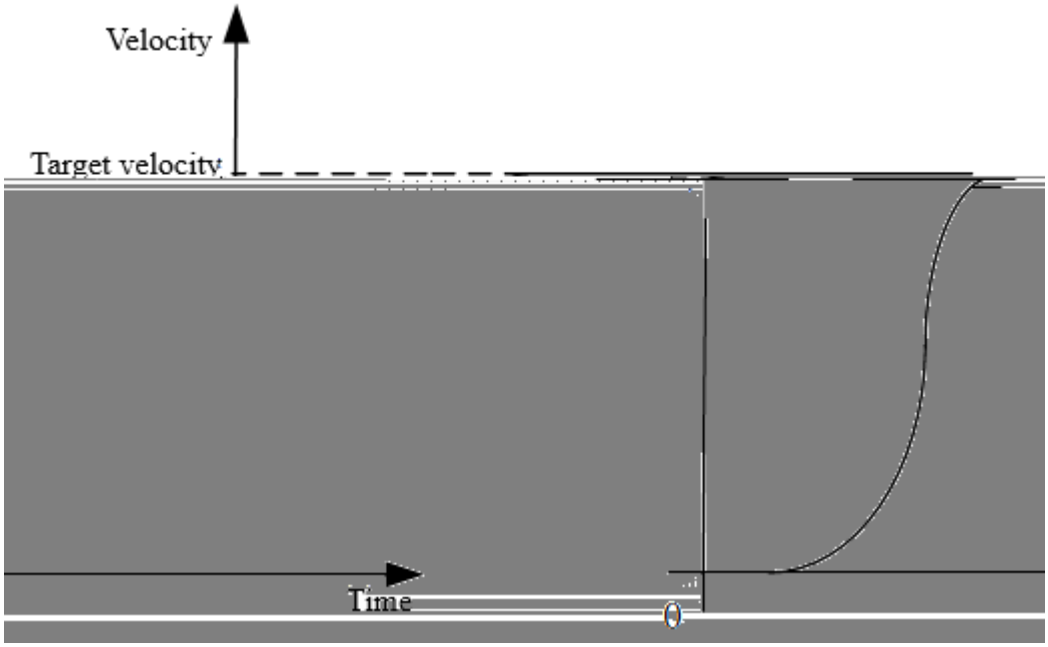
5.9 MC_MoveVelocity (Velocity Control)

Single-axis speed control command, simulating speed control in position control mode.



5.9.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	



Direction (direction selection) introduction

When Direction is mcPositiveDirection, the direction of motion is positive.

When Direction is mcNegativeDirection, the direction of motion is negative.

When Direction is set to mcCurrentDirection, the command will act along the command direction of the previous action. If the direction of the previous action is positive, the current motion direction is positive; otherwise, it is negative.

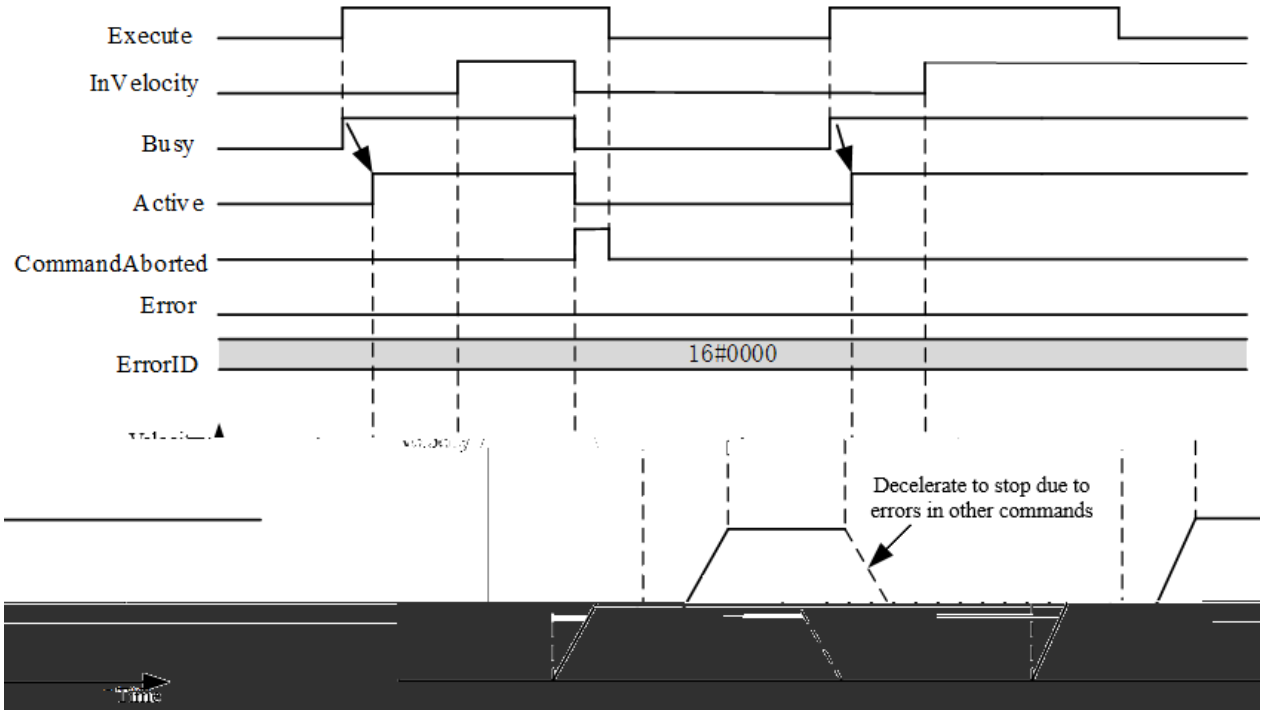
Repeated startup motion command

Change the input variable in the positioning operation, set BufferMode to Aborting (interrupt), and then set Execute to TRUE again to change the action of this command.

When the BufferMode is set to Aborting for repeated startup commands, parameters such as Velocity (target speed), Acceleration, Deceleration and Direction of the command can be changed.

The action of this command during repeated startup is specified by BufferMode (buffer mode selection). Interruption, buffering and fusion can be selected for repeated startup of this command.

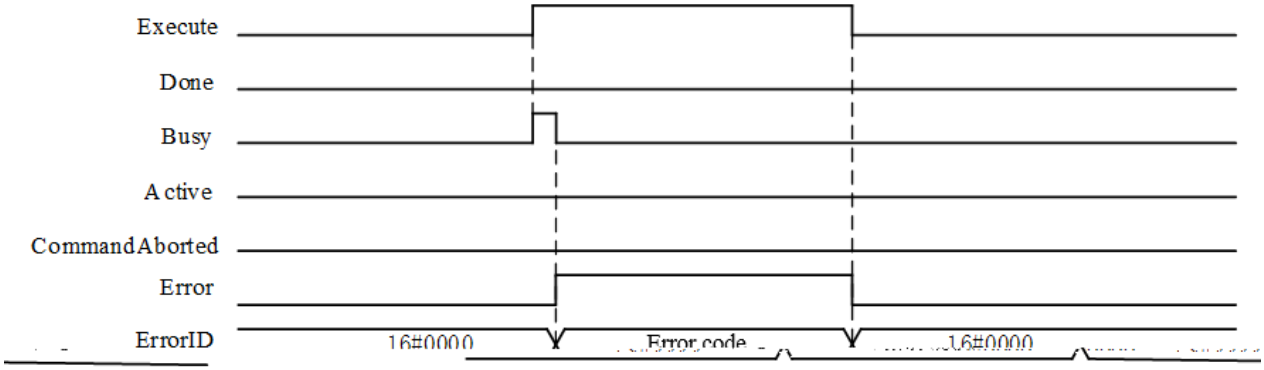
fering and fusifmmand.



(2) Error Timing diagram

When an error occurs during the execution of this command, Error becomes TRUE and the axis stops.

The output value of ErrorID can be viewed to understand the cause of the exception.



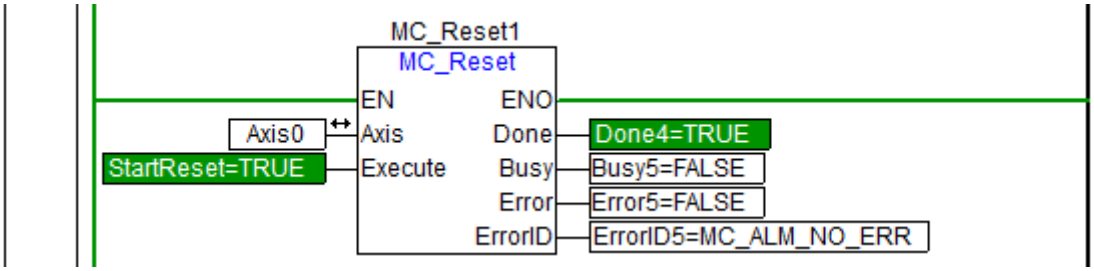
5.9.3 Examples

Set up MC_MoveVelocity example program to run simulated velocity motion command.

Definition of Primary Variable

Variable Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	Change to TRUE at the start of initialization
DoneInit	BOOL	FALSE	Change to TRUE upon completion of initialization
PowerEnable	BOOL	FALSE	Change to TRUE at the beginning of enabling

PowerStatus	BOOL	FALSE	Change to TRUE when enabled
StartMoveVel	BOOL	FALSE	TRUE on start command
BusyMoveVel	BOOL	FALSE	Change to TRUE when delivery command is completed
ActiveMoveVel	BOOL	FALSE	Change to TRUE when the command is running
InVelMoveVel	BOOL	FALSE	



ST language program

(1) Initialize and enable the axis

```
(The * axis is initialized*)
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
HomeDIChannel := 2,
Axis := Axis0,
[redacted]
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
(*Enable the axis*)
IF DoneInit THEN
   MC_Power1(
   Enable := Po01(E)8.3 (rro)5.81  (P)2.472b3.7 ( )-12.3 (l)12.3 (,)]TJ 0 Tc 0 80w 5.867 0 Td ( )Tj EMC  ET /Artifact BMC
```

[redacted]

```
Position := 10000,
Velocity := 1000,
Acceleration := 2000,
Deceleration := 2000,
Jerk := 0,
Direction := 4,
BufferMode := 0,
```

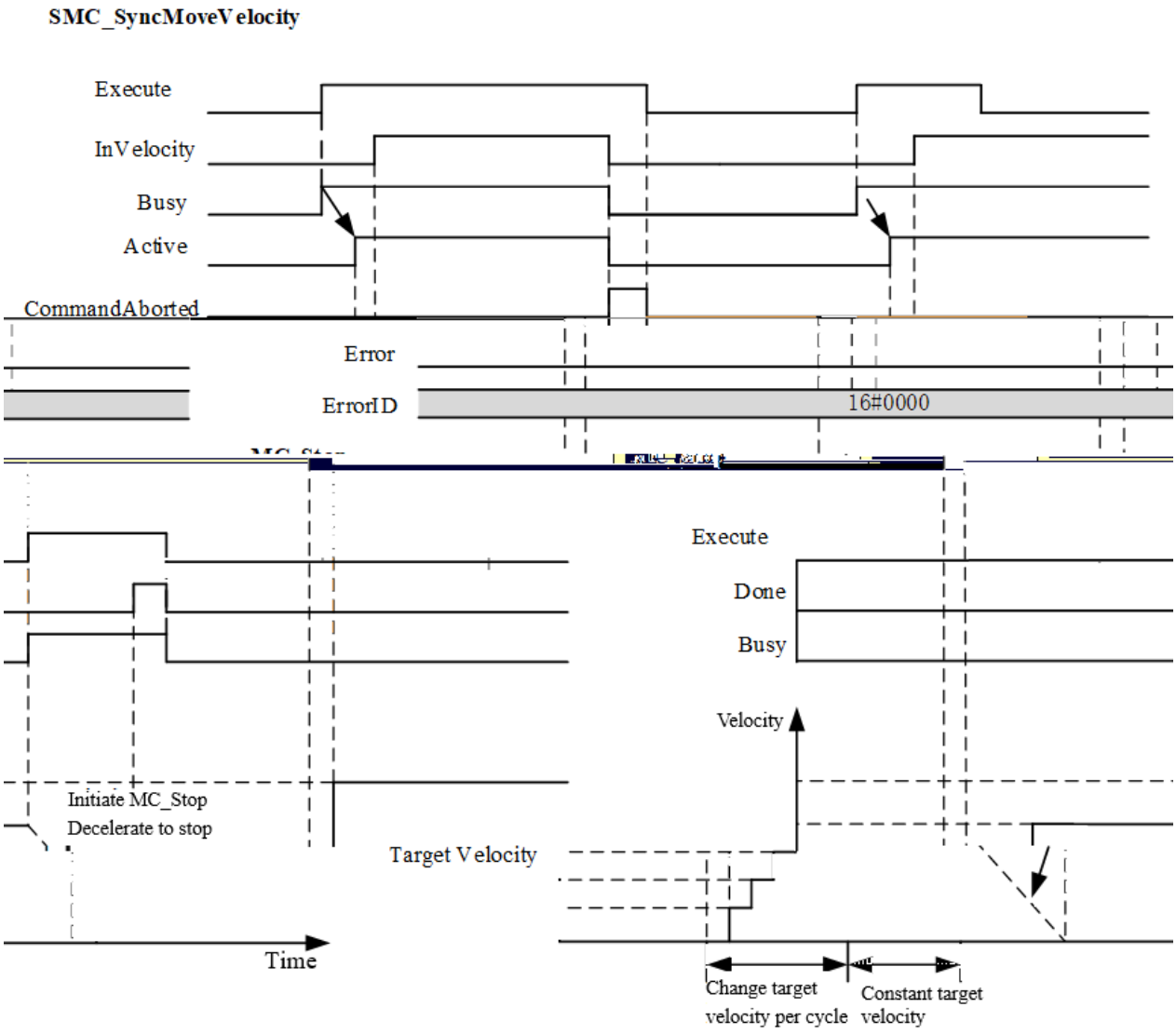
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Active	In control, TRUE when the command is executed	YES	FALSE	TRUE/FALSE	BOOL
	CommandAborted	Termination of Execution	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag				

the target speed of the previous cycle will be used for movement.

This command supports interruption and buffering in BufferMode (buffer mode selection). It can interrupt other motion commands or be interrupted by other motion commands. When this command is restarted, you can choose buffering or interruption. To end this command, use the MC_Stop command to stop.

Function block timing diagram

(1) Timing diagram under normal conditions

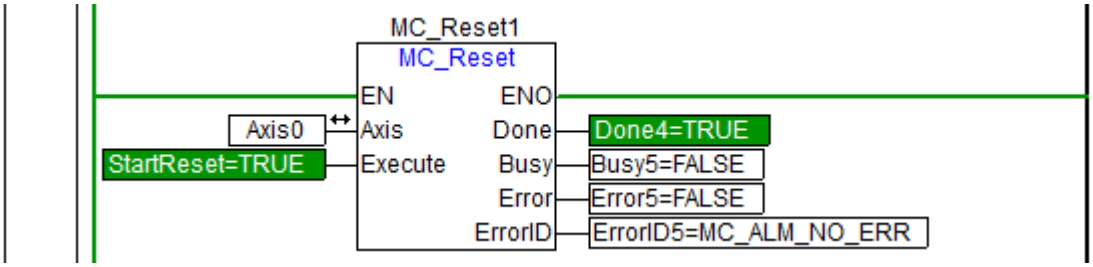


(2) Abnormal timing diagram

When an error occurs during the execution of this command, Error turns to TRUE, other pins turn to FALSE, the axis stops moving, and ErrorID outputs relevant error codes. The timing diagram for this error event is as follows.

the functional block to send synchronous speed command and repeat command.

(3) When there is an error in the axis, the MC_Reset function block can be called for reset processing.



ST langua(t)-1.1 re program

(1) Initialize the axis

```
SMC_AxisInit1(
Execute := tartInit,
AxisType := 50,
PluseerCycle := 10
GearNumerator := 1,
GearDenominator := 1,
DistancePerCyle := 100,
LimitSwtchNegCfg := - 1,
LimitSwitch osCf := -  1,
HomeSwitchCfg := -1,
LimitSwegDICh=
LimitSwosDIChan := 1,
HomeDIChann := 2,
Axis := Axis0,
Done=> DoneInit,
Busy=>Bu1sy
Error=>Error1,
ErrorID=>ErrorID1);
```

(2)

```
IF DoneInit THEN
   MC_Pwr1(
   Enable := owerEnabe,
   Axis := Axis0,
```

5.10.5 Refererence

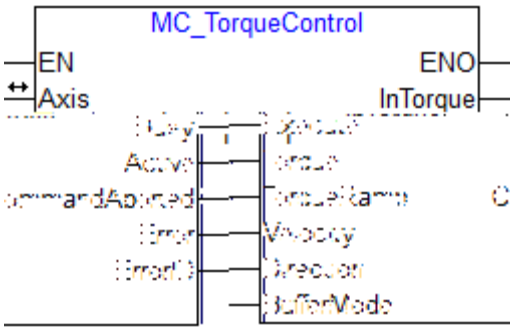
If an error occurs, refer to the appendix Function Block Error Codes.

For information about initialization and enabling, please refer to the section of Initialization and Enabling.

BufferMode (buffer mode selection) refer to the appendix Buffer Mode Description.

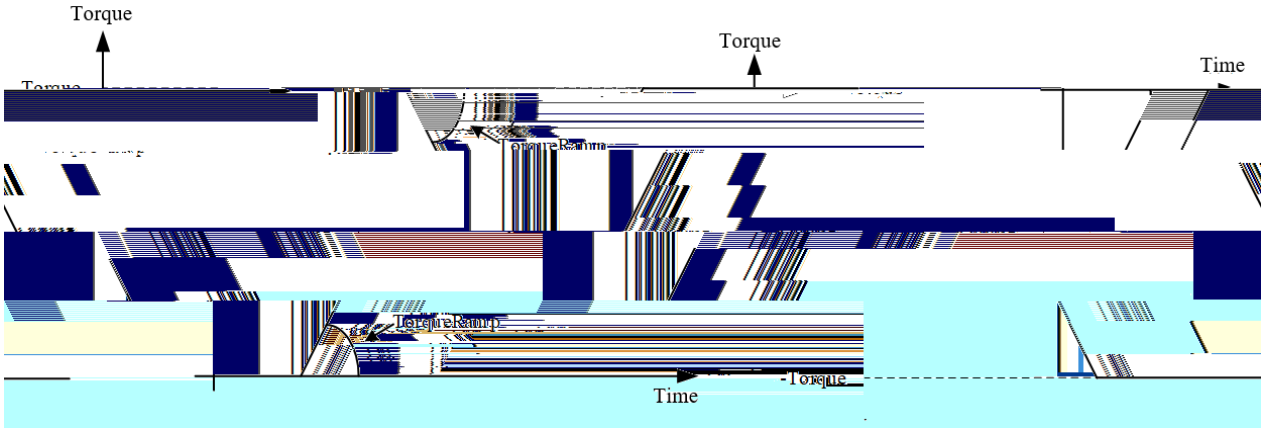
5.11 MC_TorqueControl (Torque Control)

Single-axis torque control command, which controls the axis in torque mode with a ramp.



5.11.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type

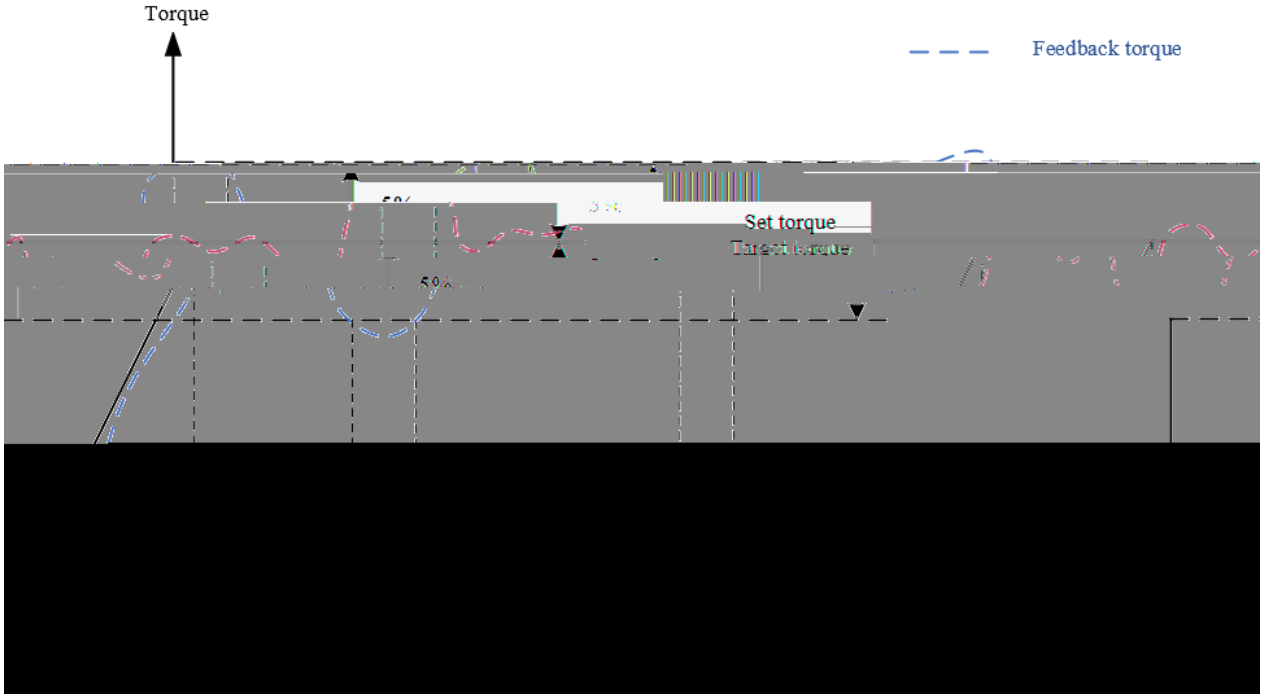


TorqueRamp Diag ram

(3) Velocity: The speed limit parameter is used to restrict the maximum speed during torque control of the drive, measured in pulse unit/s. The object dictionary for driver speed limiting is 0x607F. If 0x607F is mapped in the PDO parameters, the command writes the parameter Velocity setting value into 0x607F through PDO when the Execute rising edge occurs; if 0x607F is not mapped in the PDO parameters, the command writes the parameter Velocity setting value into 0x607F through SDO when the Execute rising edge occurs. Multi-axis scene object dictionary 0x607F needs to be offset by 0x800, please refer to the relevant manual of the drive for details.

When using this command for torque control, the following two conditions must be met to use Velocity as a maximum speed limit:

- The servo can limit the maximum speed of the motor through object dictionary 0x607F;
- Servo Object Dictionary 0x607F has pulse units instead of speed units.



InTorque Diagram

Object Dictionary Description

Torque control requires the drive to map the following object dictionary (offset 0x800 is required for multi-axis scene object dictionary):

Controlword (6040 hex)

Statusword (6041 hex)

Mode of Operation (6060 hex)

Mode of Operation Display (6061 hex)

Target Torque (6071 hex)

Torque Actual Value (6077 hex)

Velocity Actual Value (606C hex)

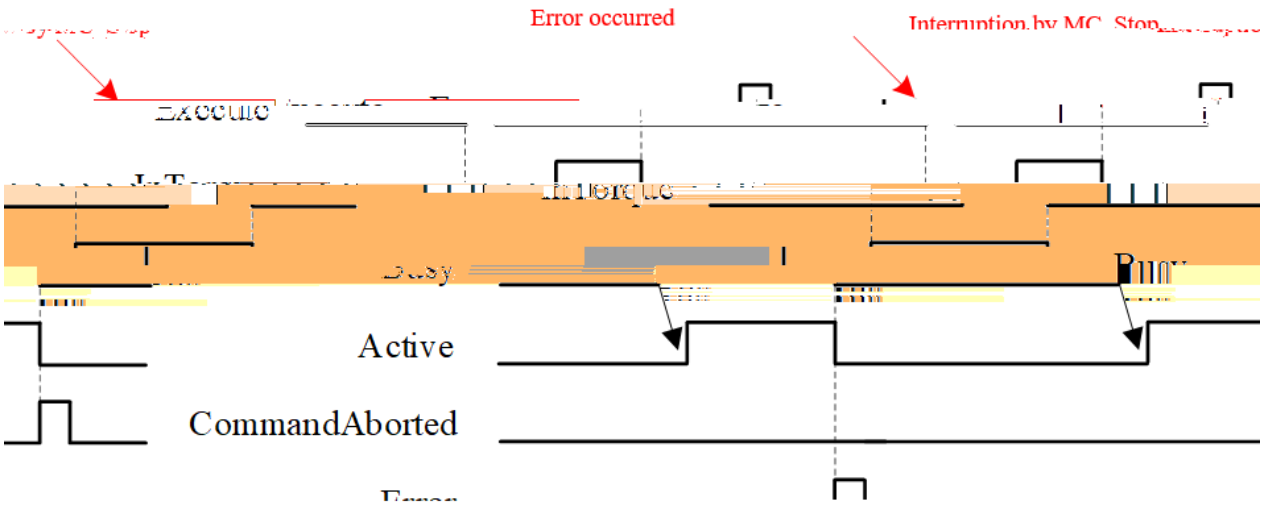
Stop control in torque control mode

During this command, the MC_Stop command can be called to stop the torque movement. When stopping, it decelerates the target torque from the current actual torque to 0 according to the set value of parameter TorqueRamp, and then cancels the torque command. After stopping, switch the drive control mode to the control mode before torque command movement.

When the MC_Stop command is called to interrupt the MC_TorqueControl command, the Axis status switches from ContinuousMotion to Stopping. When the MC_TorqueControl command is interrupted, its output pin CommandAbort is valid.

Function block timing diagram

(1) Timing diagram with normal execution of functional block and no error (actual torque can reach



5.11.3 Examples

This example uses the MC_TorqueControl function block for torque motion control, reads the actual torque value and finally calls the MC_Stop command to stop the torque motion command.

The servo needs to map the required PDO during torque motion control, otherwise an error will be reported.

Before torque motion control, use SMC_AxisInit to initialize axis-related parameters and call MC_Power to enable the axis.

Variable

No.	Variable Name	Variable Type	Initialization Value	Variable Description
1	MC_TorqueControl0	MC_TORQUECONTROL	-	Definition of Torque Control Function Block Instance
2	TorqueControl_En	BOOL	FALSE	Torque control enable signal
3	TorqueControl_Torque	LREAL	0	Target torque value
4	TorqueControl_TorqueRamp	LREAL	0	Torque Ramp
5	TorqueControl_Vel	LRE Stop Function Block AL	0	Torque control maximum speed limit
6	TorqueControl_Dir	MC_DIRECTION	moPositiveDirection	Direction of motion
7	TorqueControl_Buffer	MC_BUFFER_MODE	Aborting	Buffer mode
8	TorqueControl_InTorque	BOOL	FALSE	Torque reaches the flag position
9	TorqueControl_Busy	BOOL	FALSE	Torque control execution flag bit
10	TorqueControl_Active	BOOL	FALSE	Torque control activation flag bit
11	TorqueControl_Abort	BOOL	FALSE	Torque control suspension flag bit
12	TorqueControl_Error	BOOL	FALSE	Torque control error flag bit
13	TorqueControl_ErrorID	SMC_ERROR	MC_ALM_NO_ERR	Error code
14	MC_ReadActualTorque1	MC_READACTUALTORQUE	-	Actual Torque Reading Function Block Instance Definition
15	ReadTorque_En	BOOL	FALSE	Actual torque reading enable signal

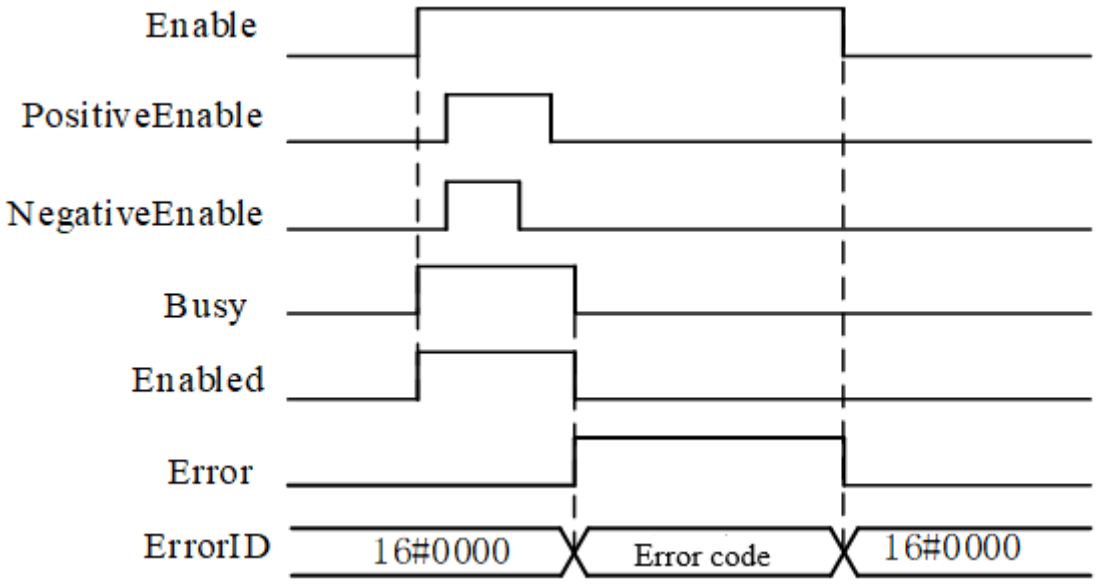
```
Axis := Axis0,
Valid=>ReadTorque_Valid,
Busy=>ReadTorque_Busy,
Error=>ReadTorque_Error,
ErrorID=>ReadTorque_ErrorID,
Torque=>ReadTorque_Torque);
```

(4) When it is necessary to stop the torque motion control, call MC_Stop to interrupt the torque control command. After interruption, the MC_TorqueControl CommandAborted command will be valid.

```
MC_Stop1(
Execute := Stop_En,
Deceleration := Stop_Dec,
Jerk := Stop_Jerk,
Axis := Axis0,
Done=>Stop_Done,
Busy=>Stop_Busy,
Error=>Stop_Error,
ErrorID=>Stop_ErrorID);
```

5.12 SMC_SetTorqueLimit (Set Axis Torque Limit)

Set the limit value of servo positive/negative torque.



5.12.3 Examples

Set up SMC_SetTorqueLimit sample program to set the limit value of servo positive/negative torque.

Primary variables

Variable Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	becomes TRUE at the start of initialization
DoneInit	BOOL	FALSE	Changes to TRUE upon completion of initialization
SetTorqEnable	BOOL	FALSE	Changes to TRUE at the beginning of enabling
PosSetTorqEnable	BOOL	FALSE	Changes to TRUE when forward enable is set

(2) Torque limit setting for the axis

```
SMC_SetTorqueLimit1(
Enable := SetTorqEnable,
PositiveEnable := PosSetTorqEnable,
PositiveValue := 300,
NegativeEnable := NegSetTorqEnable,
NegativeValue := 300,
Axis := Axis0,
Enabled=>EnabledSetTor,
Busy=>BusySetTor,
Error=>ErrorSetTor,
ErrorID=>ErrorID_SetTorq1);
```

5.12.4 Precautions

Virtual and encoder axes are not supported by this function block.

Due to differences in servo motors, the Default Value of some servo internal torque limits is 350 or 300. When the set value exceeds this value, an error will be reported, and the maximum value inside the servo is its Default Value.

5.12.5 Reference

If an error occurs, please refer to the appendix Function Block Error Code Description.

Refer to the chapter of MC_TorqueControl command for use after setting the limit value.

5.13 MC_SetPosition (Set Current Position)

Redefines the current position of a specified axis, either absolutely or relative.

5.13.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF

	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
	Position	LOCATION	No	-	Positive/negative/0	LREAL
INPUT	Relative	Mode selection FALSE: Absolute mode, defining the Position as the current position				

(2) Error pin timing diagram

When this command is executed, if the parameter and axis type are illegal (the axis type is Unused), the call of this command will fail. At this time, the command output pin Error=True, and the error code ErrorID reports an abnormal error value. Refer to the appendix Description of Error Codes to obtain the cause of the error. When the command input pin Execute is at a low level, all output pins are restored to their Default Values.

Þ When the abnormal situation of the command is solved, it is necessary to trigger Execute again on the rising edge and start the MC_SetPosition functional block to complete the current position setting of the axis.

5.13.3 Examples

Create an MC_SetPosition example program to set the current position of the axis.

LD program implementation

Using a ladder diagram, set the Position parameter to 0 and Relative to FALSE, defining the current axis position as 0 in absolute mode.

5.14.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
	Enable	Enable	No	-	TRUE/FALSE	BOOL
	VelFactor	is the speed scaling factor in 1%	YES	100	0 ~>>B4re W1 (9	

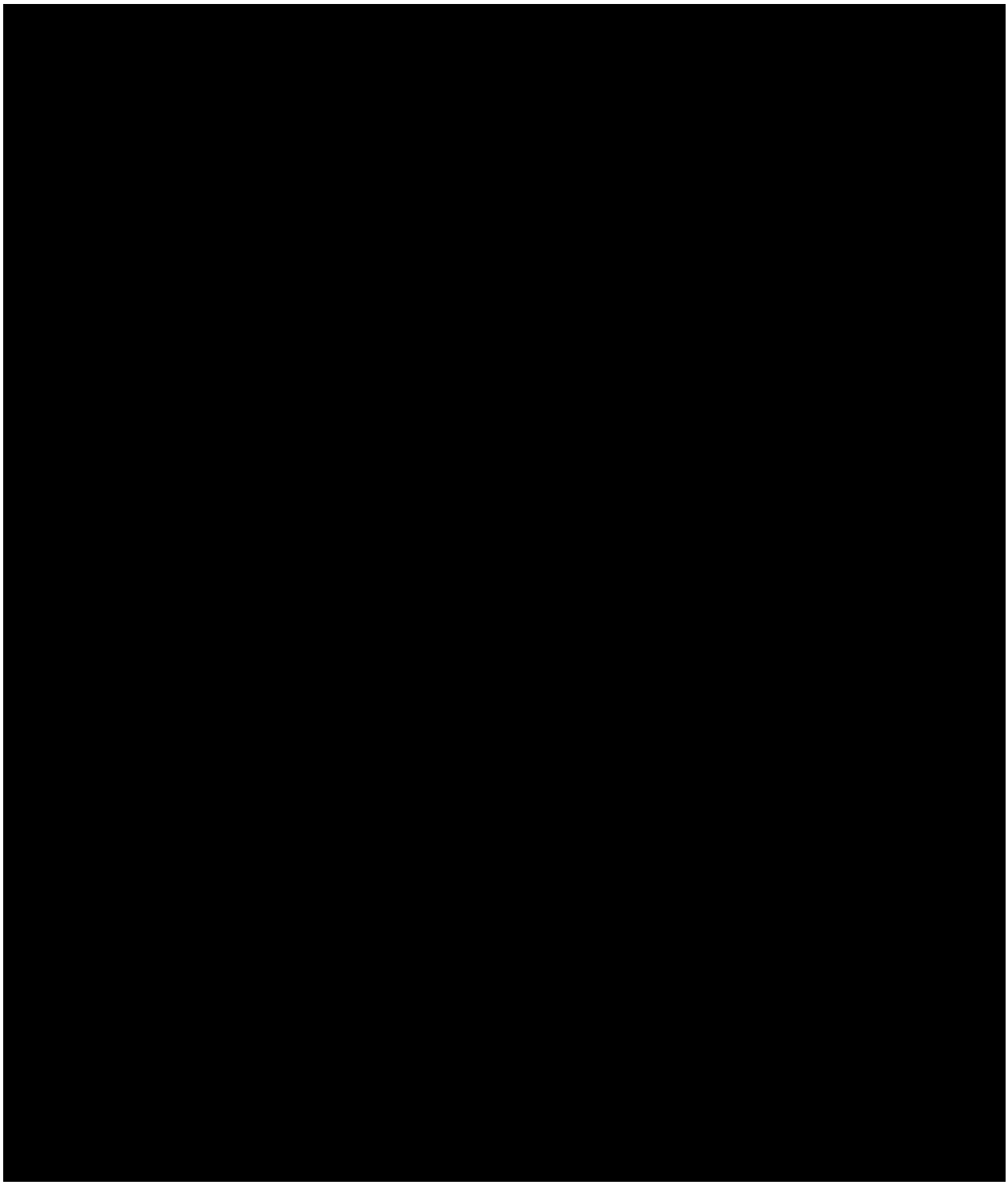
INPUT

When Enable is TRUE, the function block continues to set the override of scale factor. When Enable is FALSE, the axis keeps the value of override scale factor set last time for motion.

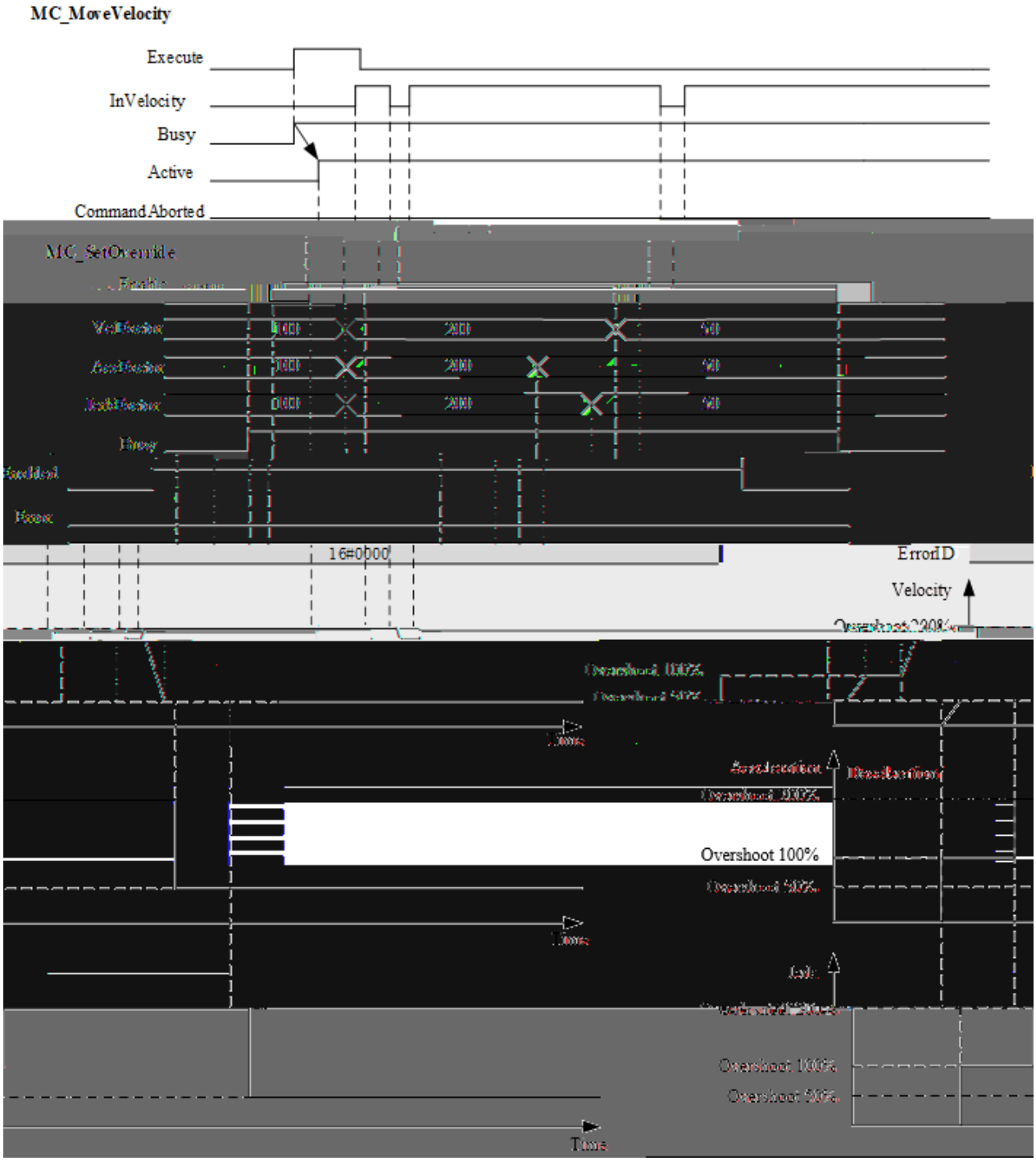
Function block timing diagram

(1) Override the MC_MoveAbsolute

When Enable is set to TRUE, Busy and Enabled immediately turns TRUE; when Enable is set to FALSE, Busy and Enabled immediately turns FALSE. The timing diagram of using override command in MC_MoveAbsolute is as follows.

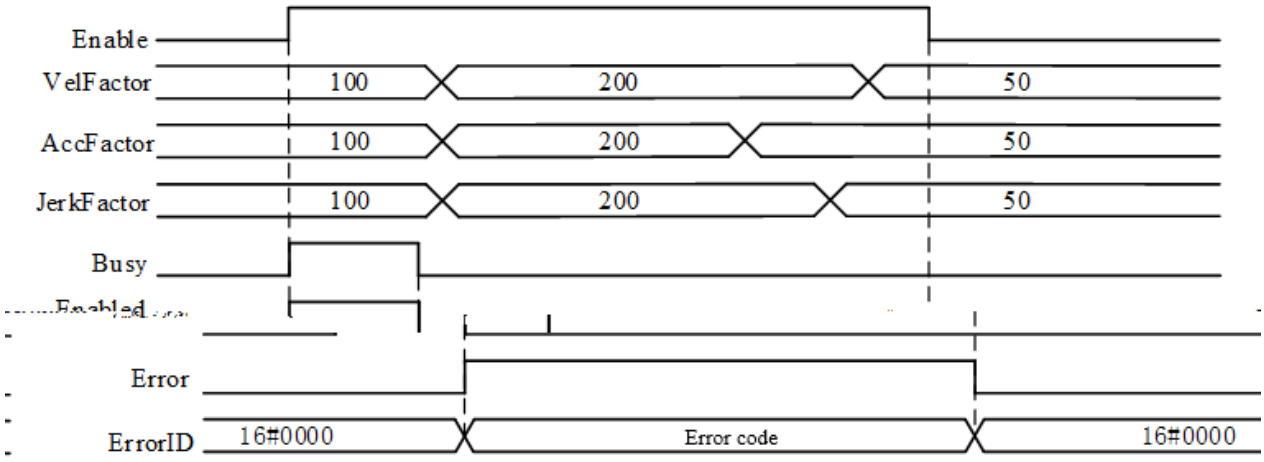


(2) Perform override on the MC_MoveVelocity command. When InVelocity becomes TRUE, use override to modify the target speed, and then InVelocity will become FALSE; when the new target speed is reached, InVelocity will become TRUE. When Enable of MC_SetOverride is set to FALSE, the axis moves with the override scaling factor value set last time. The command timing diagram is as follows.



(3) Error Timing diagram

When an error occurs during the execution of this command, Error becomes TRUE and Enabled and Busy become FALSE. After the error is cleared, Error becomes FALSE and Enabled and Busy becomes TRUE. The timing diagram is shown below.



Restart and multiple startup commands

The input of this command is Enable type. When Enable is TRUE, the function block keeps running. When there is an MC_SetOverride command being executed and other MC_SetOverride commands are started, the later-executed command will be processed first.

5.14.3 Examples

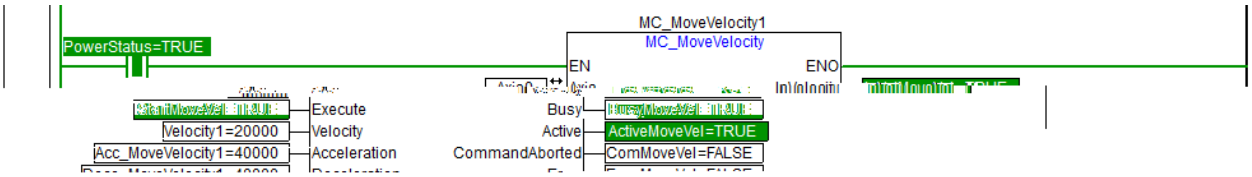
Create MC_SetOverride example program to set the override value of axis motion parameters.

Primary variables

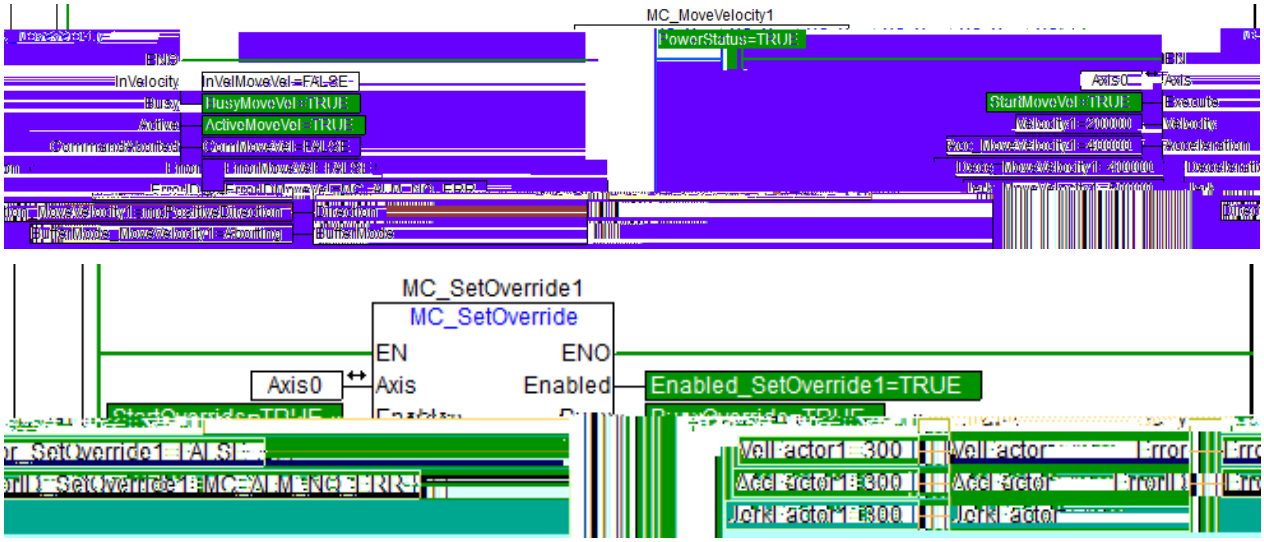
Variable Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	Change to TRUE at the start of initialization

(2)

TRUE, and the actual velocity instructed by M _MoveVelocity reaches the set command velocity 20000.



(3) Keep the Enable pin of M _SetOverride at high level and set the override parameter. It can be seen that after setting the parameter, the output pin variable InVelocity pin of the M _MoveVelocity instance becomes FALSE for a period of time, and then becomes TRUE again when the speed is reached.



ST language program

(1) Initialize the axis number and axis type settings, and enable the axis after initialization.

```
SM _AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePer ycle := 10000,
GearNumerator := 1,
```

```
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
HomeDIChannel := 2,
Axis := Axis0,
Done=> DoneInit,
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
(*Enable the axis*)
IF DoneInit THEN
   MC_Power1(
   Enable := PowerEnable,
   Axis := Axis0,
   Status=> PowerStatus,
   Busy=>Busy2,
   Error=>Error2,
   ErrorID=>ErrorID2);
END_IF
```

(2) Trigger the MC_MoveVelocity command on a rising edge to control the axis motion.

```
MC_MoveVelocity1(
Execute := StartMoveVel,
Velocity := 20000,
Acceleration := 40000,
Deceleration := 40000,
Jerk := 50000,
Direction := 0,
BufferMode := 0,
Axis := Axis0,
InVelocity=> InVelMoveVel,
Busy=> BusyMoveVel,
Active=> ActiveMoveVel,
CommandAborted=> ComMoveVel,
Error=> ErrorMoveVel,
ErrorID=> ErrorIDMoveVel);
```

(3) Keep MC_SetOverride at high level to configure the motion override settings.

```
MC_SetOverride1(
Enable := StartOverride,
VelFactor := 300,
AccFactor := 300,
JerkFactor := 300,
```

	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR

5.15.2 Functions

This command sets the limit values of axis motion parameters.

command function details

(1) Input parameter lock

The rising edge triggering of this command is valid. At the rising edge of the Execute input, the axis ID of the input/output parameter Axis and the values of the input parameters MaxVelocity, MaxAcceleration, MaxDeceleration and MaxJerk are latched. During command execution, the modified latched parameters are invalid until another rising edge triggers this command.

(2) Functional details

After setting the limit value of the axis motion parameter, the motion parameter of the axis motion command takes the minimum value of its own motion parameter and the set limit parameter. As shown in the figure belo eto.8 t the ml etetei tw(ec)-20.1 ,(t)-1.1 ( )0.6(a)-12.3 (nd)-12.2 ( )0.6 (s)-8 (et)o.8 d MaxA

In the above figure, V0 is a command speed of the motion command, Acc0 is a command acceleration of the motion command, Dec0 is a command deceleration of the motion command, and Jerk0 is a command jerk of the motion command.

At time t1, the maximum speed limit MaxVelocity (Max_V1 value in the figure) is successfully set. If the maximum speed limit MaxVelocity is less than the axis motion command speed V0, the actual maximum speed of the axis motion command is the limit value MaxVelocity, and the axis motion command decelerates. Meanwhile, Max_Acc1, Max_Dec1 and Max_Jerk1 are the set limit values of other motion parameters respectively.

At time t2, the maximum speed limit MaxVelocity is successfully set again. MaxVelocity=Max_V2. MaxVelocity is greater than the speed V0 of axis motion command. The axis motion command takes the self-set command speed as the actual running speed, and the axis motion command accelerates. Meanwhile, Max_Acc2, Max_Dec2 and Max_Jerk2 are the set limit values of other motion parameters respectively.

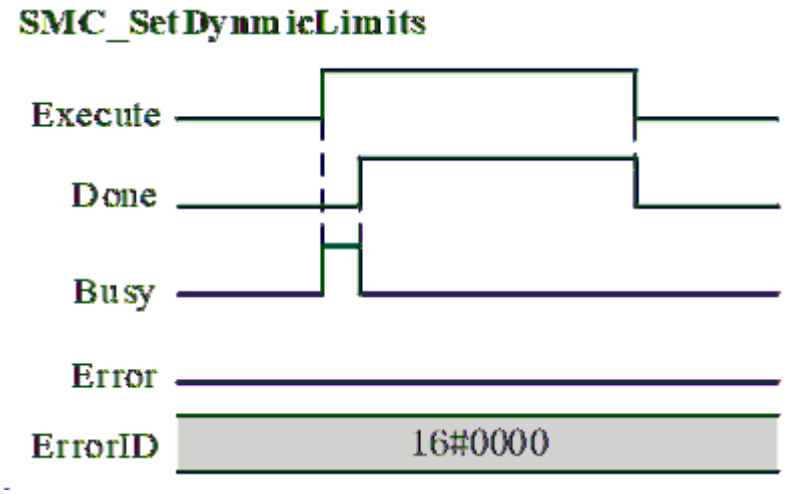
Timing diagram

(1) Normal timing diagram

At the rising edge of Execute input, if no errors occur, the Busy pin will be set to TRUE. After the

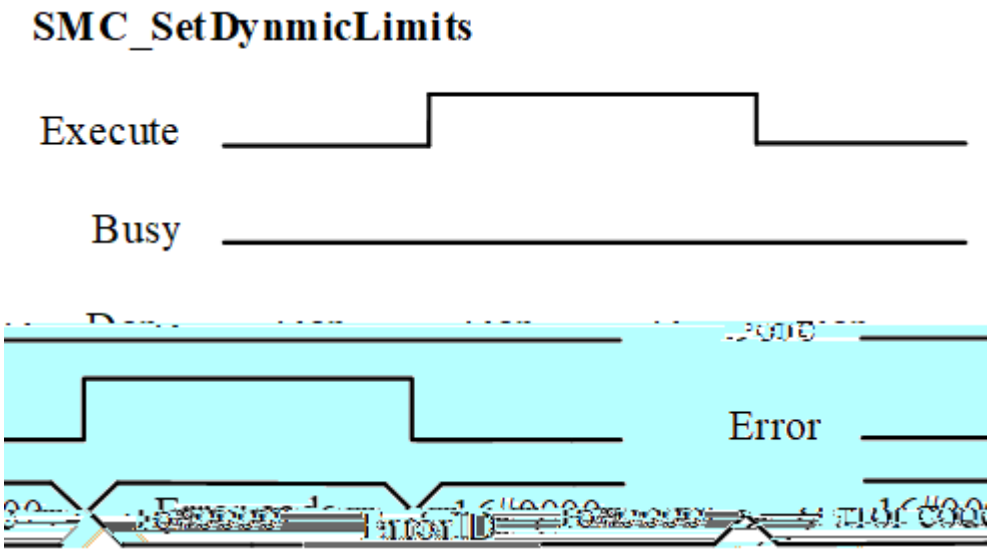
parameter setting is completed, the Done will be set to TRUE and Busy will be set to FALSE. If Execute keeps high level, the Done pin will remain true; if Execute is set low level, the Done pin will be set to FALSE.

The correct timing sequence for command execution is as follows:



(2) Error pin timing diagram

When executing this command, if the parameter is illegal and the axis type is illegal (the axis type is Unused), it will fail to call this command. The command output pin Error=True, and the error code ErrorID reports an abnormal error value. Please refer to the appendix Error Code Description to obtain the cause of the error. When the command input pin Execute goes low, all output pins revert to their Default Values.



þ When the abnormal situation of the command is solved, it is necessary to trigger Execute again on the rising edge and start the SMC_SetDynamicLimits functional block to complete the setting of axis motion parameter limit.

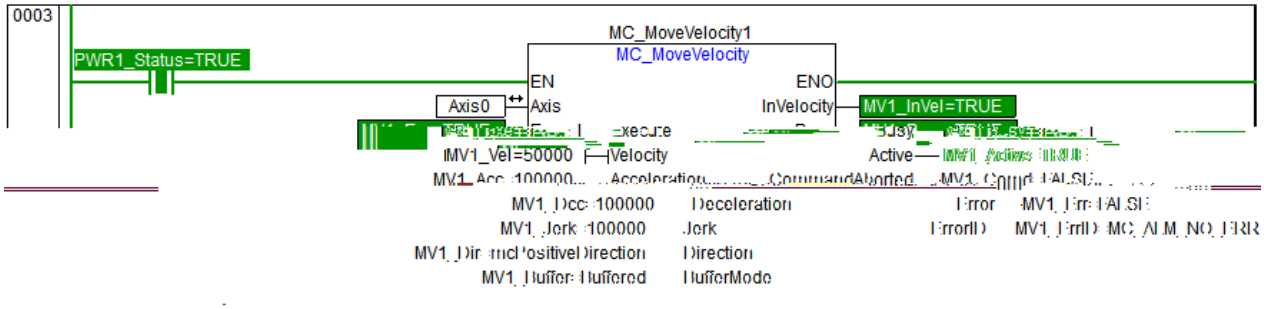
5.15.3 Examples

Set up SMC_SetDynamicLimits example program to set the limit values of axis motion parameters.

After the axis initialization is completed and the axis ready state Switch1 is TRUE, call MC_Power to complete the axis enable.

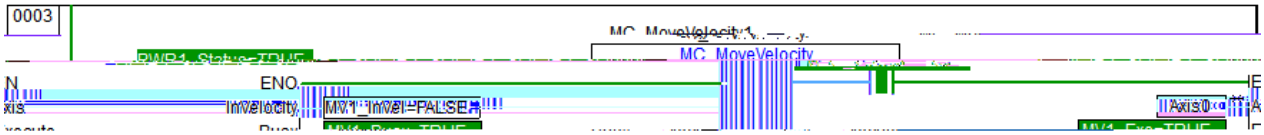
(3) Axis delivery motion command

In section 0003, after the axis is enabled successfully, the rising edge triggers the MC_MoveVelocity motion command to control the axis motion. The state of the pin variable is output through the MC_MoveVelocity instance. The InVelocity pin is TRUE, and the actual velocity of the MC_MoveVelocity command reaches the set command velocity MV1_Vel= 50000.



(4) Set motion parameter limits

In section 0004, trigger the Execute pin of SMC_SetDynamicLimits1 to complete the setting of motion parameter limit. In section 0003, the output pin variable InVelocity pin of the MC_MoveVelocity instance is FALSE, which can intuitively show that the speed of the MC_MoveVelocity command is limited, and the actual velocity of this command does not reach the set command velocity.



ST language

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
Axis0	AXIS_REF	-	Axis variable for axis 0
Power1_Status	BOOL	FALSE	Variable for which the axis enable is successful. This variable becomes TRUE when the axis is enabled successfully.
MV1_Vel	LREAL	-	Velocity variable of the input shaft speed control command. The commanded speed that controls the motion of the axis.
MV1_Active	BOOL	FALSE	Variable when the axis speed control command acts. This variable becomes TRUE when the axis is controlling motion.
MV1_InVel	BOOL	FALSE	Identification variable for the axis speed control command to reach the target speed. This variable becomes TRUE and the speed commanded reaches the target speed.
Switch1	BOOL	FALSE	Axis ready state variable. After axis initialization, Switch to TRUE and the axis is ready.
InitFlag	BOOL	FALSE	If this variable is TRUE, the initialization parameter setting is completed.

(1) Axis initialization

Call the SMC_AxisInit command to complete the initial configuration of Axis0. Set the axis type to 50: EtherCAT_Position. For the SMC_AxisInit commands, please refer to its Introduction Chapter.

```
SMC_AxisInit1(
Execute := AxisInit_Execute,
AxisType := AxisInit_AxisType,
PlusePerCycle := AxisInit_PlusePerCycle,
GearNumerator := AxisInit_GearNumerator,
GearDenominator := AxisInit_GearDenominator,
DistancePerCycle := AxisInit_DistancePerCycle,
LimitSwitchNegCfg := AxisInit_LimitSwitchNegCfg,
LimitSwitchPosCfg := AxisInit_LimitSwitchPosCfg,
HomeSwitchCfg := AxisInit_HomeSwitchCfg,
LimitSwNegDIChan := AxisInit_LimitSwitchNegDIChan,
```

```
LimitSwPosDIChan := AxisInit_LimitSwitchPosDIChan,
HomeDIChannel := AxisInit_HomeDIChannel,
Axis := Axis0,
Done=>AxisInit_Done,
Busy=>AxisInit_Busy,
Error=>AxisInit_Error,
ErrorID=>AxisInit_ErrorID);
```

(2) Initialize motion limit parameters

```
    Axis := Axis0,
    InVelocity=> MV1_ InVel,
    Busy=> MV1_ Busy,
    Active=> MV1_ Active,
    CommandAborted=> MV1_ Comd,
    Error=> MV1_ Err,
    ErrorID=> MV1_ Error);
END_IF
```

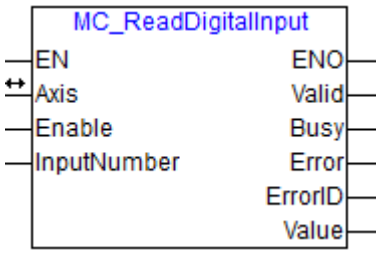
(5) Motion parameter limit setting

The rising edge triggers the SMC_SetDynamicLimits command to complete the setting of motion parameter limits.

```
IF  Switch1 THEN
```

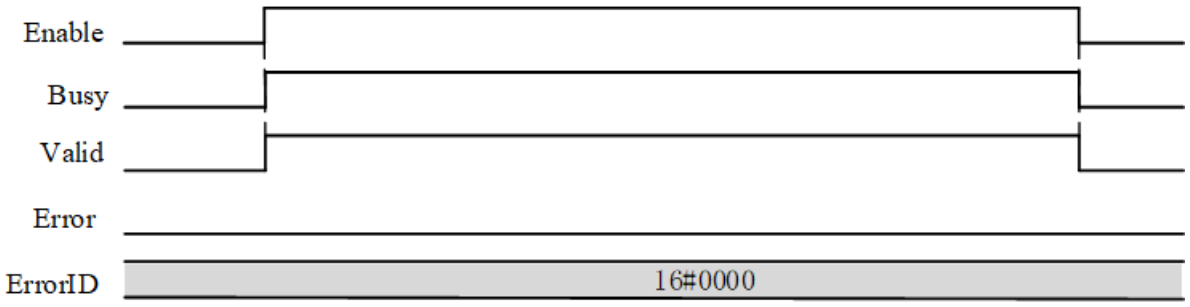
5.16 MC_ReadDigitalInput (Read Axis Digital Input)

Read the Digital Input of the servo equipment.



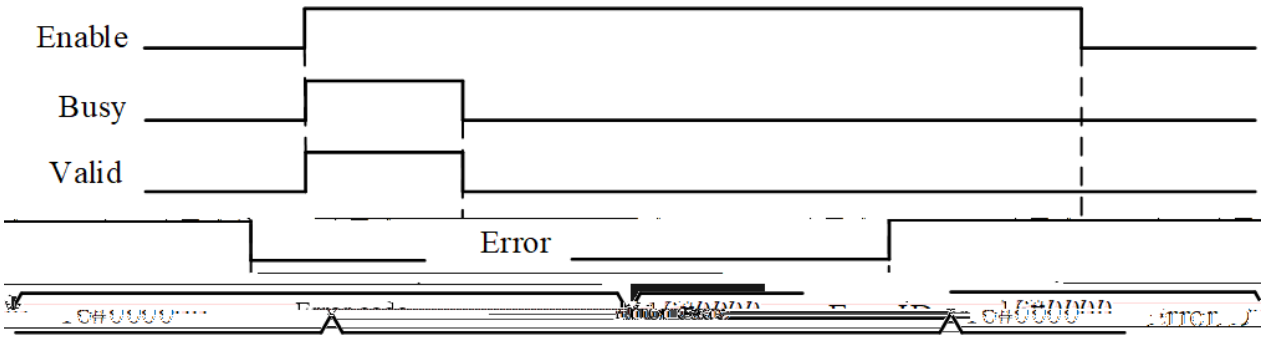
5.16.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	Enable	Enable	No	-	TRUE/FALSE	BOOL
	InputNumber	Input code, indicating the bit number in Digtal inputs object dictionary	No	-	0 ~ 31	INT
OUTPUT	Valid	Output valid	No	-	TRUE/FALSE	BOOL



(2) Error timing diagram

When an error occurs, Error is TRUE and the other pins are FALSE.



5.16.3 Examples

Set up MC_ReadDigitalInput sample program to read the Digital Input value of servo equipment.

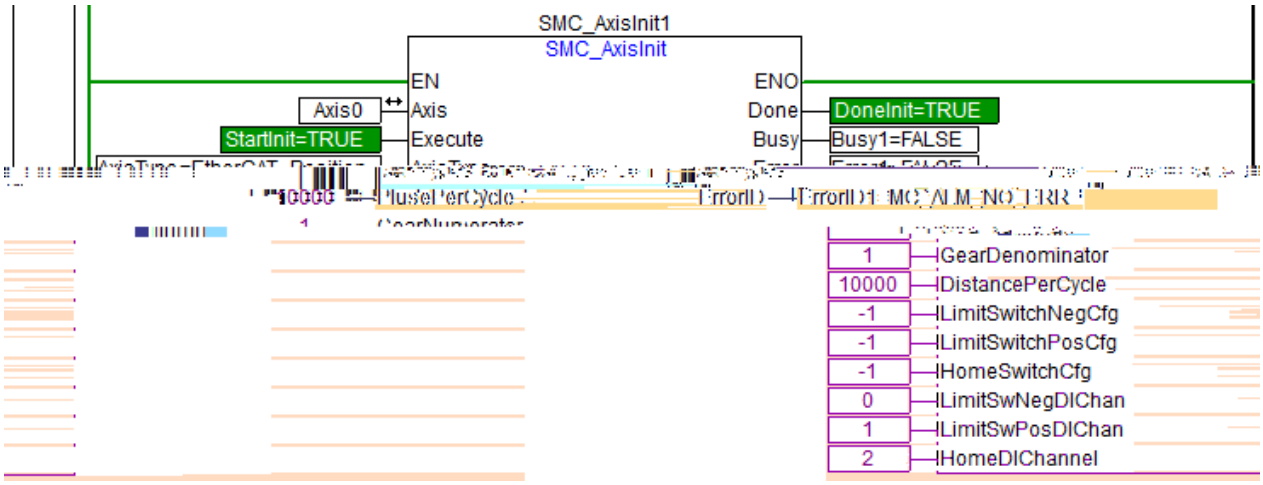
Description of Primary Variables

Variable Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	becomes TRUE at the start of initialization
DoneInit	BOOL	FALSE	Changes to TRUE upon completion of initialization
ReadInputEnable	BOOL	FALSE	Changes to TRUE at the beginning of reading
ValidReadInput	BOOL	FALSE	Changes to TRUE when reading is successful
BusyReadInput	BOOL	FALSE	Changes to TRUE on reading
ValueReadInput	BOOL	FALSE	Changes to TRUE after reading the value
ErrorReadInput	BOOL	FALSE	Changes to TRUE when an error occurs

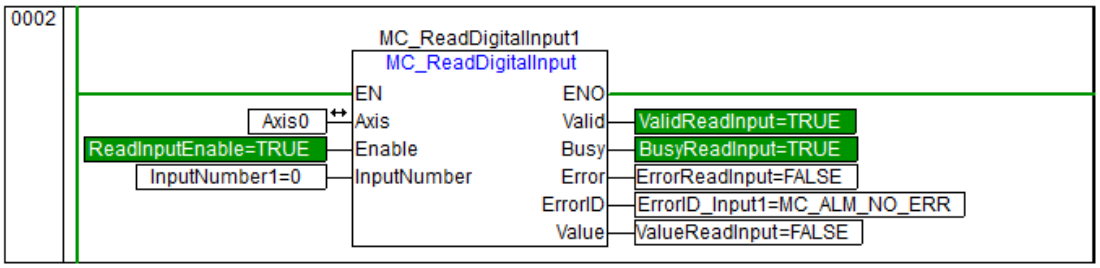
LD program implementation

Use the ladder diagram to build a program and input the corresponding object dictionary number. Set Enable of MC_ReadDigitalInput to TRUE. The sample procedure is as follows:

(1) Before reading the Digital Input of the axis, initialize the axis first.



(2) After initialization, enter the corresponding object dictionary digits to read Digital Input.



ST language program

(1) Initialize the axis number and axis type settings

```
ST language program
Initialize the axis number and axis type settings
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
HomeDIChannel := 2,
Axis := Axis0,
Done=> DoneInit,
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
```

(2) Read Axis Digital Input

```
MC_ReadDigitalInput1(
Enable := ReadInputEnable,
InputNumber := 0,
Axis := Axis0,
Valid=>ValidReadInput,
Busy=>BusyReadInput,
Error=>ErrorReadInput,
ErrorID=>ErrorID_Input1,
Value=>ValueReadInput);
```

5.16.4 Precautions

This command does not support virtual axes, but supports EtherCAT bus axes.

Before using this command, it is necessary to map the PDO (0x60FD) corresponding to Digtial Inputs.

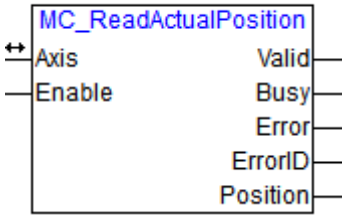
5.16.5 Reference

Refer to Initialization Chapter for details about the initialization.

If an error occurs, please refer to the appendix Introduction of Function Block Error Code Description.

5.17 MC_ReadActualPosition (Read Axis Actual Position)

This command reads the actual position of the axis.



5.17.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	Enable	Enable	No	-	TRUE/FALSE	BOOL

OUTPUT	Valid	Output valid	No	-	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR
	Position	Reading results	No	-	Positive/negative/0	LREAL

5.17.2 Functions

This command reads the actual position of the axis.

Command function details

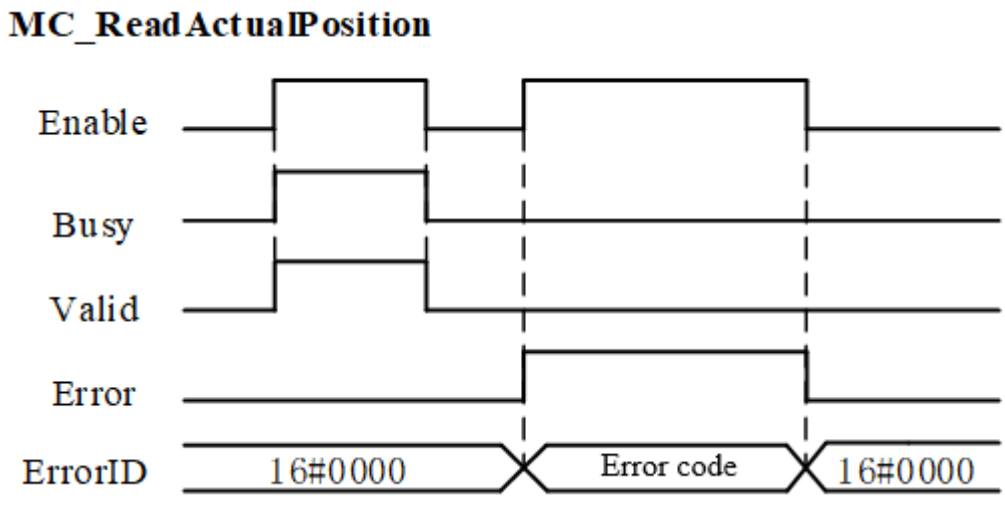
(1) This command is valid at high level. When the input parameters are legal and there is no abnormality in the axis, set Enable as high level TRUE, and set the output parameter Valid to high level, which is valid. The command continuously reads the actual position of the axis.

(2) The axis types supported by this command are:

- Virtual axis (0): Virtual axis
- EtherCAT_Position(50): EtherCAT bus connecting shaft, position control
- EtherCAT_Speed(51): EtherCAT bus connecting shaft, speed control
- EtherCAT_Torque(52): EtherCAT bus connecting shaft, torque control

Timing diagram

The following is the normal and error timing diagram during command execution.



- When the command enable pin is at high level and there is no abnormal error, the Busy and Valid pins are immediately set to TRUE to read out the actual position of the axis.
- When an abnormal error occurs during the execution of a command, the command output pins Busy and Valid are set to FLASE, the output result Position remains the current value, Error is set to True, and

the error code ErrorID reports an abnormal error value. Please refer to the appendix Error Code Description to obtain the cause of the error. The command input Enable goes low, so the output pin reverts to its Default Value.

5.17.3 Examples

```
GearDenominator := AxisInit_GearDenominator,
DistancePerCycle := AxisInit_DistancePerCycle,
LimitSwitchNegCfg := AxisInit_LimitSwitchNegCfg,
LimitSwitchPosCfg := AxisInit_LimitSwitchPosCfg,
HomeSwitchCfg := AxisInit_HomeSwitchCfg,
LimitSwNegDIChan := AxisInit_LimitSwitchNegDIChan,
LimitSwPosDIChan := AxisInit_LimitSwitchPosDIChan,
HomeDIChannel := AxisInit_HomeDIChannel,
Axis := Axis0,
Done=>AxisInit_Done,
Busy=>AxisInit_Busy,
Error=>AxisInit_Error,
ErrorID=>AxisInit_ErrorID);
```

(4) Axis enable

After the axis initialization is completed and the axis ready state Switch1 is TRUE, call MC_Power to complete the axis enable.

```
IF  Switch1 THEN
   MC_Power1(
   Enable := PWR1_Enable,
   Axis := Axis0,
   Status=> PWR1_Status,
   Busy=> PWR1_Busy,
   Error=> PWR1_Err
   ErrorID=> PWR1_ErrID);
END_IF
```

(5) After the axis is enabled successfully, execute the MC_MoveVelocity motion command to control the axis motion.

```
IF  PWR1_Status THEN
   MC_MoveVelocity1(
   Execute := MV1_ Exe,
   Velocity := MV1_ Vel,
   Acceleration := MV1_ Acc,
   Deceleration := MV1_ Dcc,
   Jerk := MV1_ Jerk,
   Direction := MV1_ Dir,
   BufferMode := MV1_ Buffer,
   Axis := Axis0,
   InVelocity=> MV1_ InVel,
   Busy=> MV1_ Busy,
   Active=> MV1_ Active,
   CommandAborted=> MV1_ Comd,
   Error=> MV1_ Err,
   ErrorID=> MV1_ Error);
END_IF
```

5.18.2 Functions

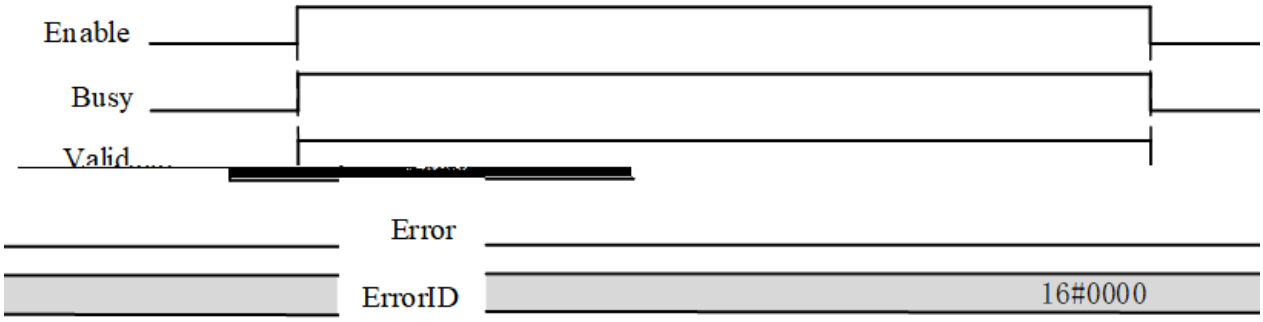
This command reads the actual velocity of the axis. The details of this command are as follows:

(1) This command is valid at high level. After the axis initialization is completed, the actual velocity can be read directly.

(2) When Enable is TRUE, the function block continuously reads the actual velocity, and when Enable is FALSE, the function block pin returns to the Default Value.

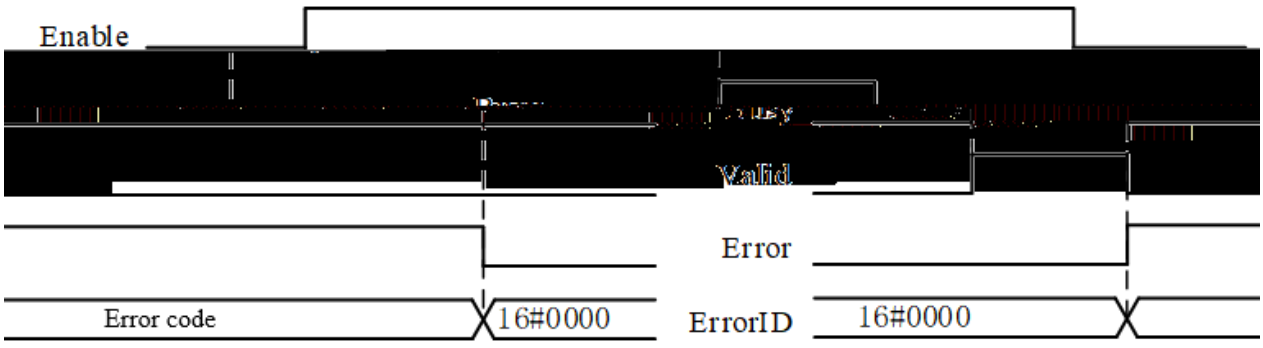
Function block timing diagram

(1) Normal timing diagram



(2) Error timing diagram

When an error occurs, Error is TRUE and the other pins are FALSE.



5.18.3 Examples

Set up MC_ReadActualVelocity sample program to read the actual velocity of the axis.

Description of Primary Variables 6 99.48 62.879 0.72 Q 124. (n)]TJ(i)-0.f 233.76 1

Title	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	Change to TRUE at the start of initialization

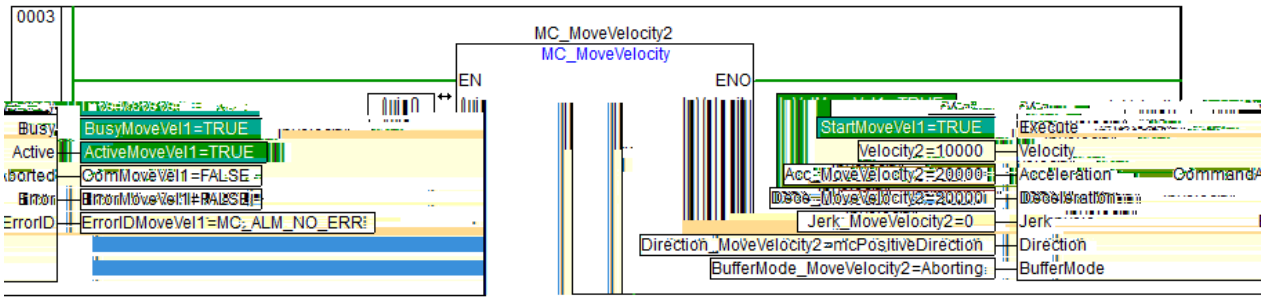
ReadVelEnable	BOOL	FALSE	Change to TRUE at the beginning of reading
ValidReadVel	BOOL	FALSE	Change to TRUE when reading is successful
BusyReadVel	BOOL	FALSE	Change to TRUE on reading
ValueReadVel	LREAL		Actual velocity read
ErrorReadVel	BOOL	FALSE	Change to TRUE when an error occurs

LD program implementation

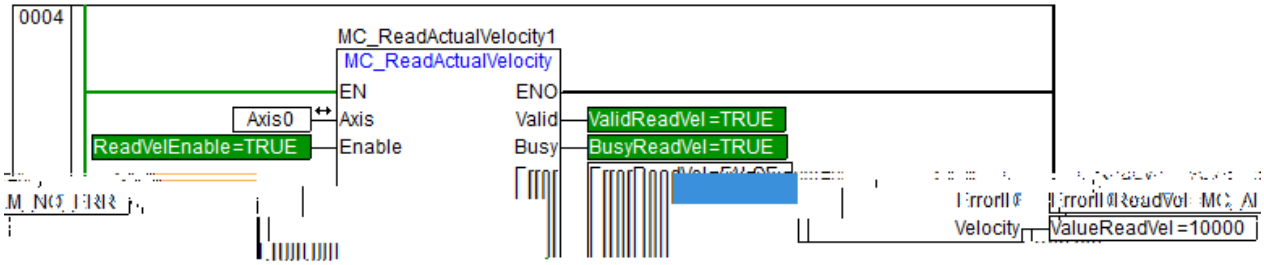
Use the ladder diagram to build a program and input corresponding object dictionary changes. To set MC_ReadActualVelocity Enable to TRUE, the sample procedure is as follows:

(1) First, initialize the axis and set its type and various parameters. After the initialization is completed, enable the axis. When it enters the ON state of servo enabling, motion control can be realized for the axis.

(2) Run the MC_MoveVelocity command, input Velocity as 10000 and control the axis to move.



(3) Call the MC_ReadActualVelocity command to read its velocity, and you can see the actual velocity value.



ST language program

(1) Initialize the axis number and axis type settings

```
SMC_AxisInit1(
```

[redacted]

AxisType := 0(er)I.002 TcChann.1 (ni)3el Td ( )Tj EMC ET /Ar2ifact BMC 0.9379BDC.6 712.56 473.52 14.4 re f* EMC
PlusePerCycle := 10000,

[redacted]

GearDenominator := 1,
DistancePerCycle := 10000,
Li904 631.3199 Tm [(L4645.72 Tm [(D)-2.92 78 (D)-22.4 (er)-6Tj -0.002 Tc 0.001 Tw 0.783 0 Td [(ear)er)-6.4 (er)-9/Art:e

```
MC_ReadActualVelocity1(
Enable := ReadVelEnable,
Axis := Axis0,
Valid=>ValidReadVel,
Busy=>BusyReadVel,
Error=>ErrorReadVel,
ErrorID=>ErrorIDReadVel,
Velocity=>ValueReadVel);
```

5.18.4 Precautions

Over-limit axis number error will be reported.

This command supports virtual axis and Ethercat bus axis.

5.18.5 Reference

For MC_MoveVelocity and other related motion commands, please refer to the chapter of MC_MoveVelocity and Other Motion Commands.

For detailed errors related to ErrorID, refer to the appendix Function Block Error Codes.

5.19 MC_ReadActualTorque (Read Axis Actual Torque)

This command reads the torque value of a servo device conforming to the CIA402 standard.

5.19.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	Enable	Enable	No	-	TRUE/FALSE	BOOL
OUTPUT	Valid	Output valid	No	-	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL

5.19.3 Examples

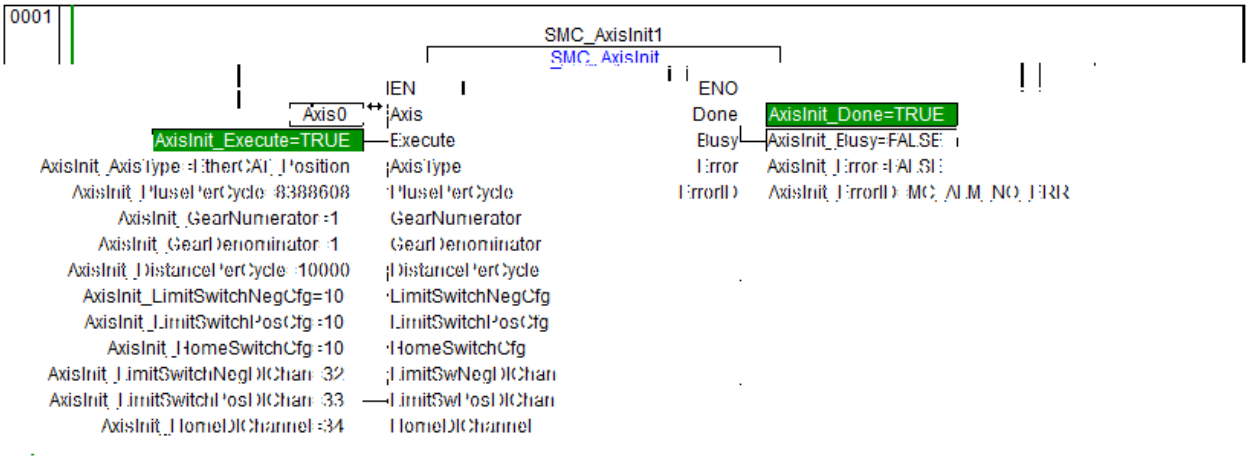
LD program implementation

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
Axis0	AXIS_REF	-	Axis variable for axis 0
Power1_Status	BOOL	FALSE	Variable for which the axis enable is successful. This variable becomes TRUE when the axis is enabled successfully.
ReadTor_Valid	BOOL	FALSE	The axis reads the variable when the actual torque is active. This variable becomes TRUE when the shaft actual torque is successfully read.
ReadTor_Tor	LREAL	-	The axis reads the variable of the actual torque. The output is the actual torque of the read shaft in 1% rated torque.
Switch1	BOOL	FALSE	The state variable of axis readiness. If Switch is TRUE, the axis is ready.

(1) Axis initialization

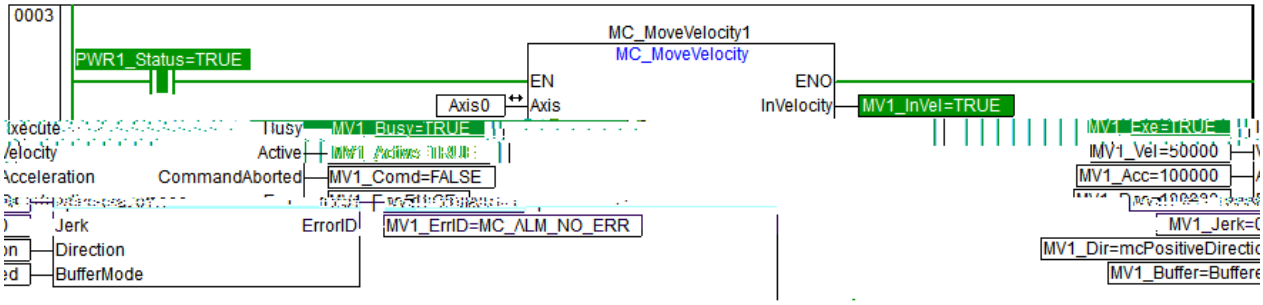
The AxisID of Axis0 is set to 0, the axis type is set to 50:EtherCAT_Position, and the axis initialization SMC_AxisInit command is called to complete the initial configuration of Axis0. For the SMC_AxisInit command, please refer to its Introduction Chapter.



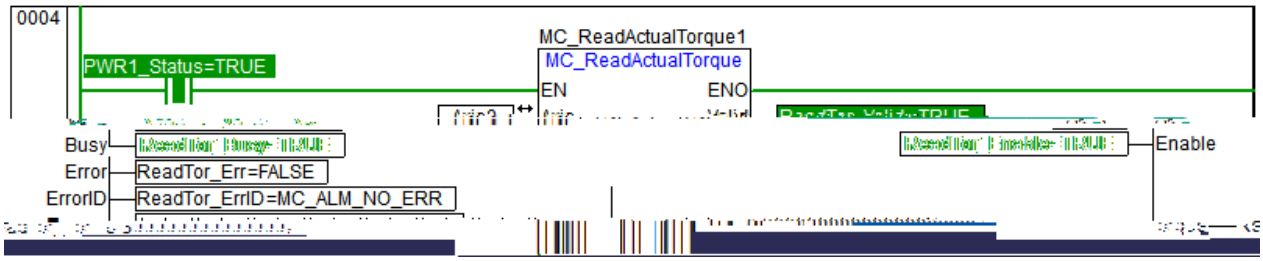
(2) Axis enable

After the axis initialization is completed and the axis ready state Switch1 is TRUE, call MC_Power to complete the axis enable.

(3) In section 0003, after the axis is enabled successfully, the rising edge triggers the MC_MoveVelocity motion command to control the axis motion.



(4) In section 0004, the high level triggers the Enable pin of MC_ReadActualTorque to start reading the actual torque of the axis. ReadTor_Valid is TRUE, the reading torque is valid, and ReadPos_Tor is 1% of rated torque.



ST language program

The primary variables of ST language program are the same as those of LD language program.

(1) Set the axis number of Axis0

Axis0.AxisID:=0;

(2) Axis initialization

Call SMC_AxisInit command to complete the initial configuration of Axis0, and set the axis type as 50:EtherCAT_Position. For the SMC_AxisInit commands, please refer to its introduction.

```
SMC_AxisInit1(
Execute := AxisInit_Execute,
AxisType := AxisInit_AxisType,
PlusePerCycle := AxisInit_PlusePerCycle,
GearNumerator := AxisInit_GearNumerator,
```

(5) The high level triggers the MC_ReadActualTorque command to read the actual torque of the axis.

```
IF  MC_Power1. Status THEN
   MC_ReadActualTorque1(
   Enable := ReadTor_Enable,
   Axis := Axis0,
   Valid=> ReadTor_Valid,
   Busy=> ReadTor_Busy,
   Error=> ReadTor_Err,
   ErrorID=> ReadTor_ErrID,
   Torque=> ReadTor);
END_IF
```

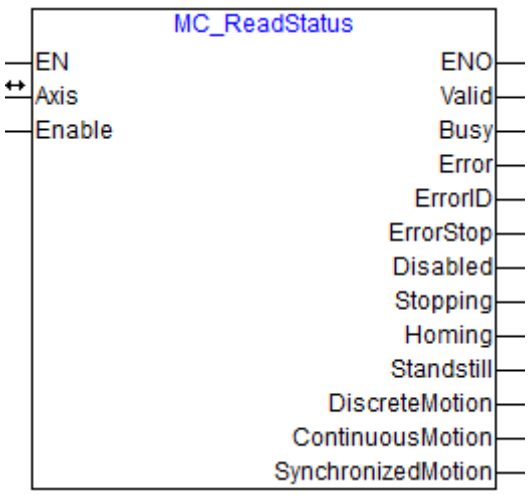
5.19.4 Precautions

The servo equipment calling the command must be a servo equipment conforming to CIA402 standard.

The command reads the PDO (0x60FD) corresponding to the feedback torque (16#6077) of servo equipment, so the slave station of servo equipment needs to be configured with relevant PDO.

5.20 MC_ReadStatus (Read Axis Status)

Read the status of the axis PLCopen status machine.

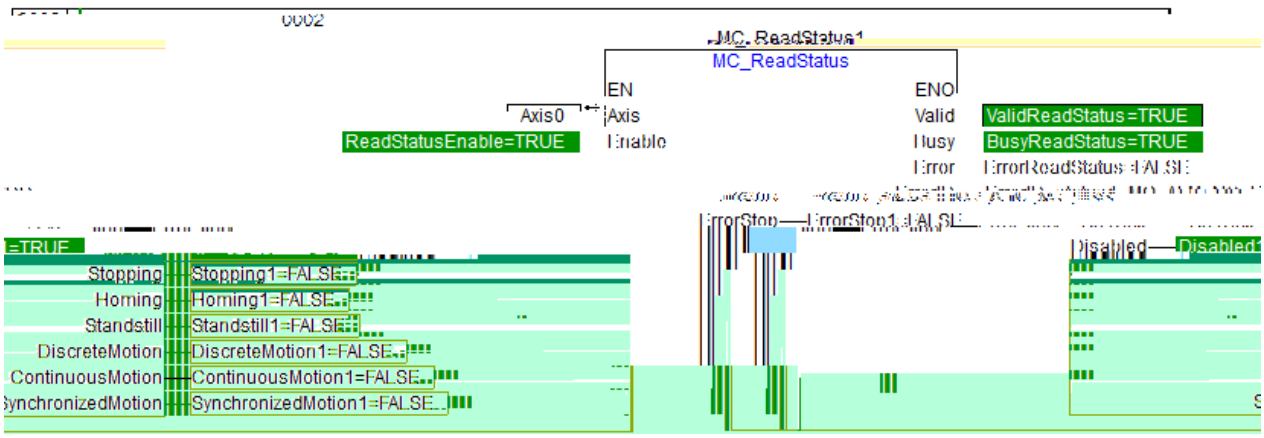


5.20.1 Parameter

Parameter Type	Variable Name	Description	Empty or Not	Default Value	Scope	Data Type

IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	Enable	Enable	No	-	TRUE/FALSE	BOOL
	Valid	Output valid	No	-	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag				

OUTPUT



ST language program

(1) Initialize the axis number and axis type settings

```
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
HomeDIChannel := 2,
Axis := Axis0,
Done=> DoneInit,
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
```

(2) Read the status of axis PLCopen status machine

```
MC_ReadStatus1(
Enable := ReadStatusEnable,
Axis := Axis0,
Valid=>ValidReadStatus,
Busy=>BusyReadStatus,
Error=>ErrorReadStatus,
ErrorID=>ErrorID_status1,
ErrorStop=>ErrorStop1,
Disabled=>Disabled1,
Stopping=>Stopping1,
Homing=>Homing1,
Standstill=>Standstill1,
```

	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR
	HomeAbsSwitch	Origin switch status	YES	FALSE	TRUE/FALSE	BOOL
	LimitSwitchPos	Positive Limit Switch Status	YES	FALSE	TRUE/FALSE	BOOL
	LimitSwitchNeg	Negative Limit Switch Status	YES	FALSE	TRUE/FALSE	BOOL
	Simulation	Axis in simulation state (retained parameter, FALSE)	YES	FALSE	TRUE/FALSE	BOOL
	CommunicationReady	Axis communication status is normal	YES	FALSE	TRUE/FALSE	BOOL
	ReadyForPowerOn	Axis ready enable		FALSE	TRUE/FALSE	

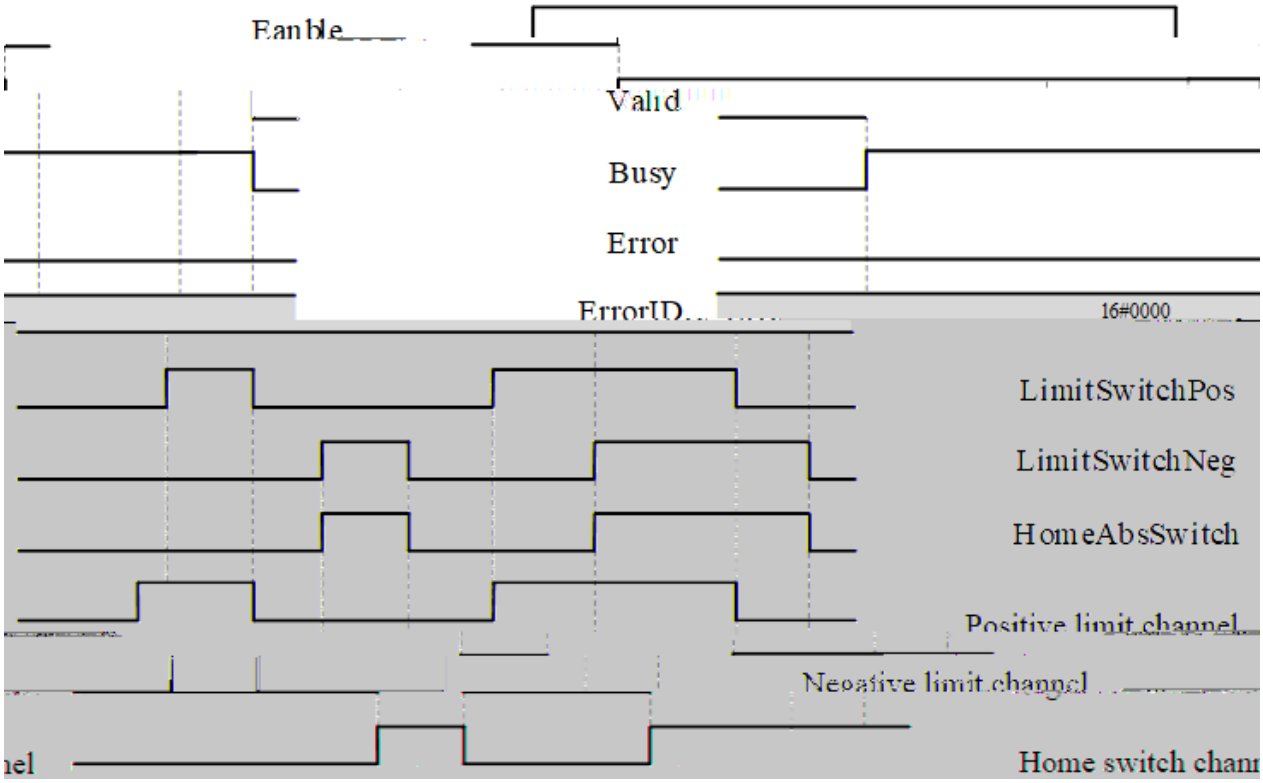
axis is an EtherCAT axis and a warning is reported, AxisWarning outputs TRUE; otherwise, it is FALSE;

The output pin AxisFault indicates whether the current axis is faulty or not. TRUE indicates that the current axis is faulty, and FALSE indicates that the current axis is fault-free.

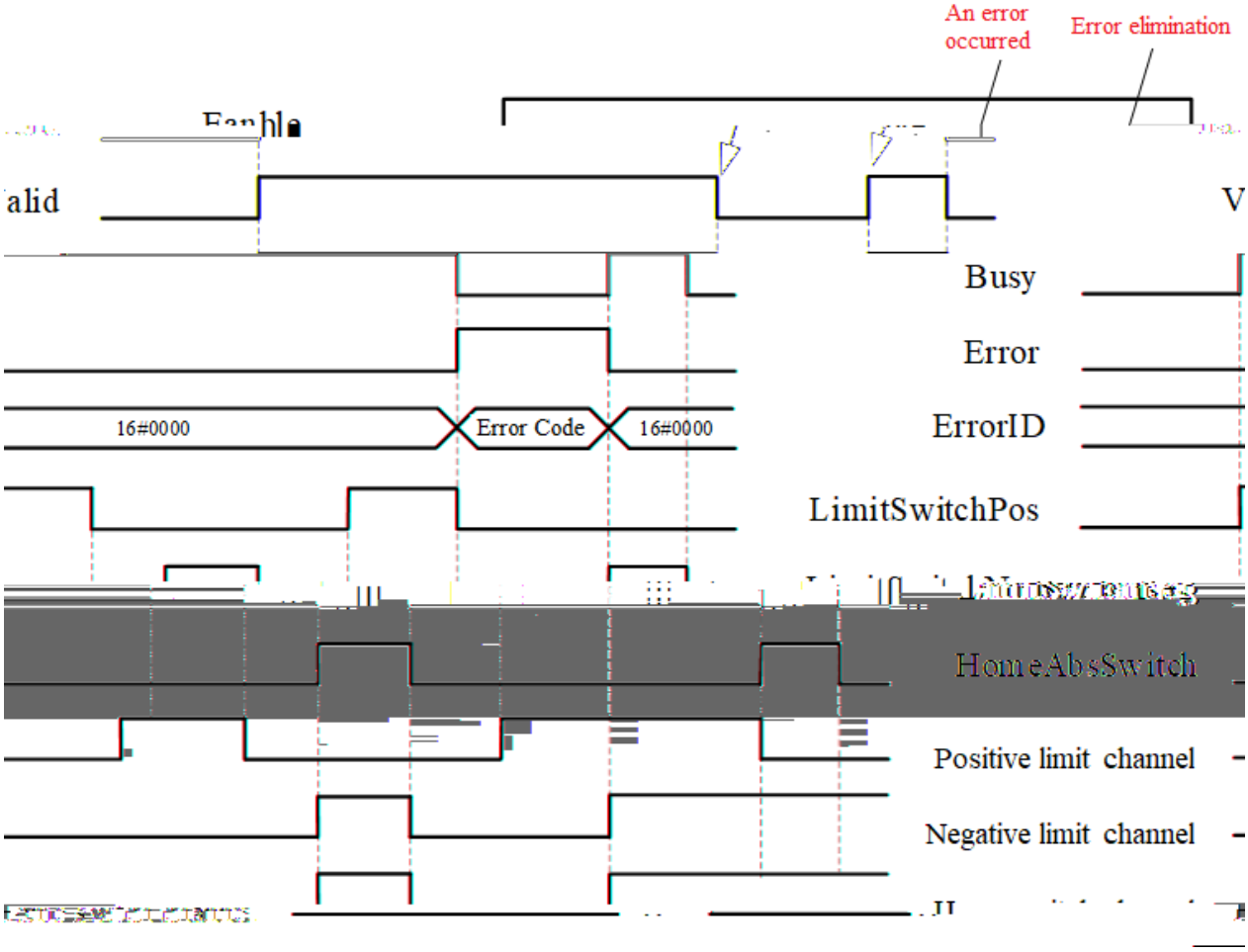
The above output pins are not forced to be mutually exclusive, and the output status is displayed according to the actual axis status information.

Function block timing diagram

(1) Timing diagram of positive and negative hard limit status, origin switch status, and other axis information status Timing diagrams are similar.



(2) Timing diagram of recovery after an error occurs when obtaining the positive and negative hard limit status and the lower limit switch status.



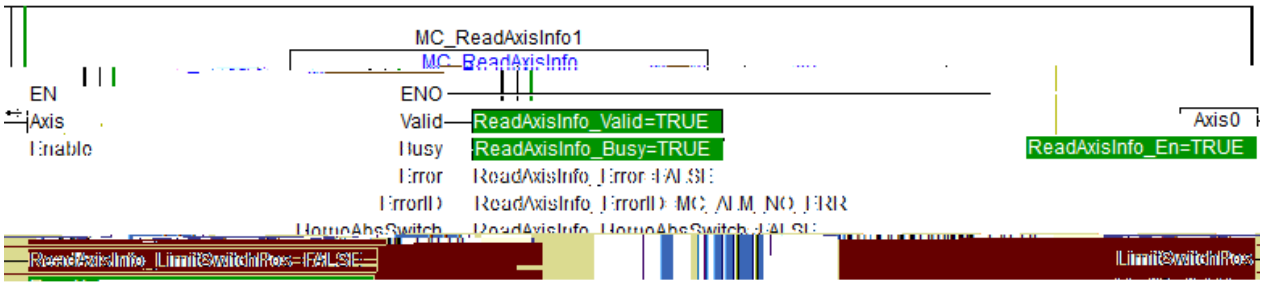
5.21.3 Examples

The example uses the function block MC_ReadAxisInfo to get the axis information status of Axis0. EXAMPLE Call MC_Power to enable Axis0 by determining whether there is an error in Axis0 and whether Axis0 is ready for PowerOn.

Variable

No.	Variable Name	Variable Type	Initial Value	Variable Description
1				

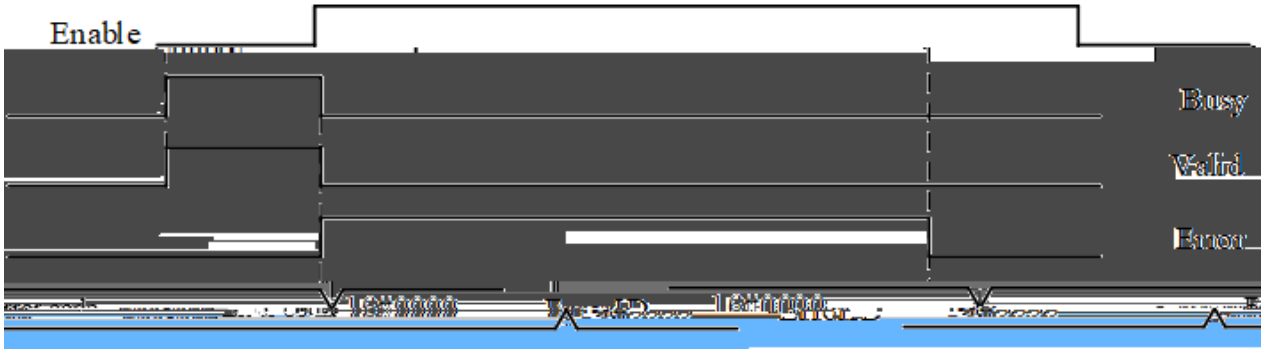
13	ReadAxisInfo_PowerON	BOOL	FALSE	Axis enabling status flag
14	ReadAxisInfo_Homed	BOOL	FALSE	Axis origin return completion flag
15	ReadAxisInfo_Warning	BOOL	FALSE	Warning sign for reading axis information
16	ReadAxisInfo_Fault	BOOL	FALSE	Fault flag for reading axis information
17	MC_Power1	MC_POWER		Axis enable command
18	Power_En	BOOL	FALSE	Axis enabled, level valid
19	Power_Status			



ST program implementation

(1) First, call the MC_AxisInit command to initialize axis-related information.

```
MC_ReadAxisInfo1(
Enable := ReadAxisInfo_En,
Axis := Axis0,
Valid=>ReadAxisInfo_Valid,
Busy=>ReadAxisInfo_Busy,
Error=>ReadAxisInfo_Error,
ErrorID=>ReadAxisInfo_ErrorID,
HomeAbsSwitch=>ReadAxisInfo_HomeAbsSwitch,
LimitSwitchPos=>ReadAxisInfo_LimitSwitchPos,
LimitSwitchNeg=>ReadAxisInfo_LimitSwitchNeg,
Simulation=>ReadAxisInfo_Simulation,
CommunicationReady=>ReadAxisInfo_ComRdy,
ReadyForPowerOn=>ReadAxisInfo_RdyPowerON,
PowerOn=>ReadAxisInfo_PowerON,
IsHomed=>ReadAxisInfo_Homed,
```



5.22.3 Examples

Set up MC_ ReadAxisError sample program to read axis error information.

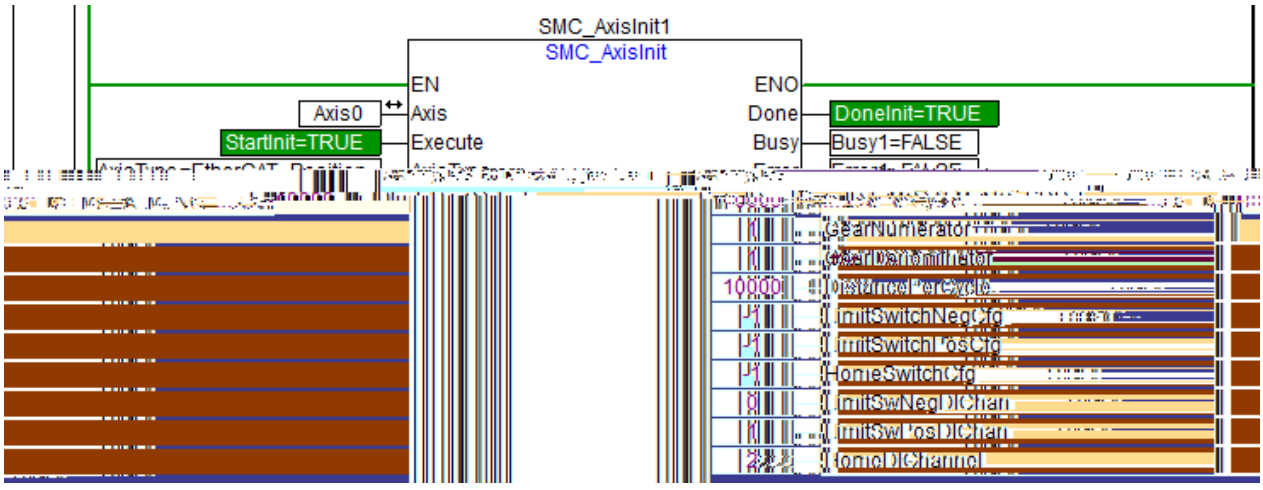
Primary variables

Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	Change to TRUE at the start of initialization
DoneInit	BOOL	FALSE	Change to TRUE upon completion of initialization
ReadAxisErrEnable	BOOL	FALSE	Change to TRUE at the beginning of reading
ValidReadAxisErr	BOOL	FALSE	Change to TRUE when reading is successful
BusyReadAxisErr	BOOL	FALSE	Change to TRUE on reading
ErrorReadAxisErr	BOOL	FALSE	Change to TRUE when an error occurs
AxisErrorId	DWORD	0	Read result

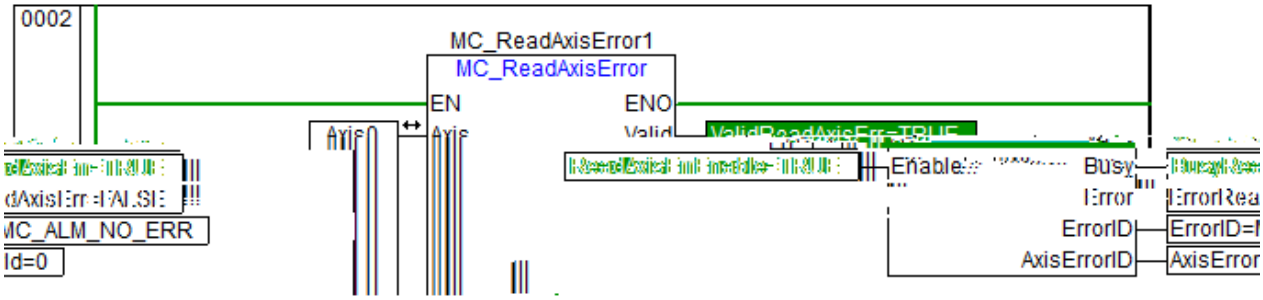
LD program implementation

Use the ladder diagram to form a program. To set MC_ ReadAxisError Enable to TRUE, the sample procedure is as follows:

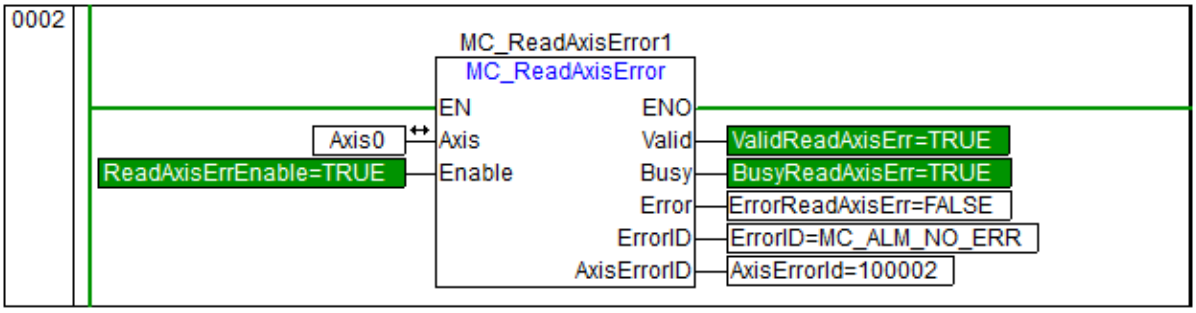
(1) Before reading the axis error information, initialize the axis first.



(2) After initialization, the axis error information can be read directly.



(3) Make Slave Station offline error and read the error.



ST language program

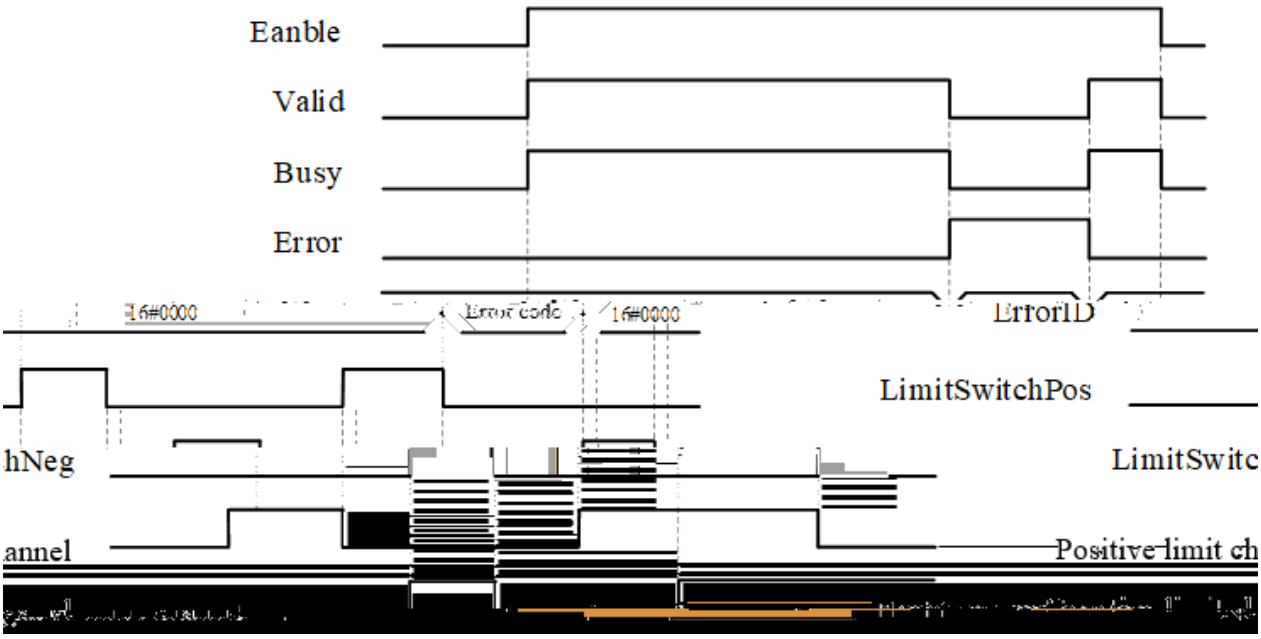
(1) Initialize the axis number and axis type settings

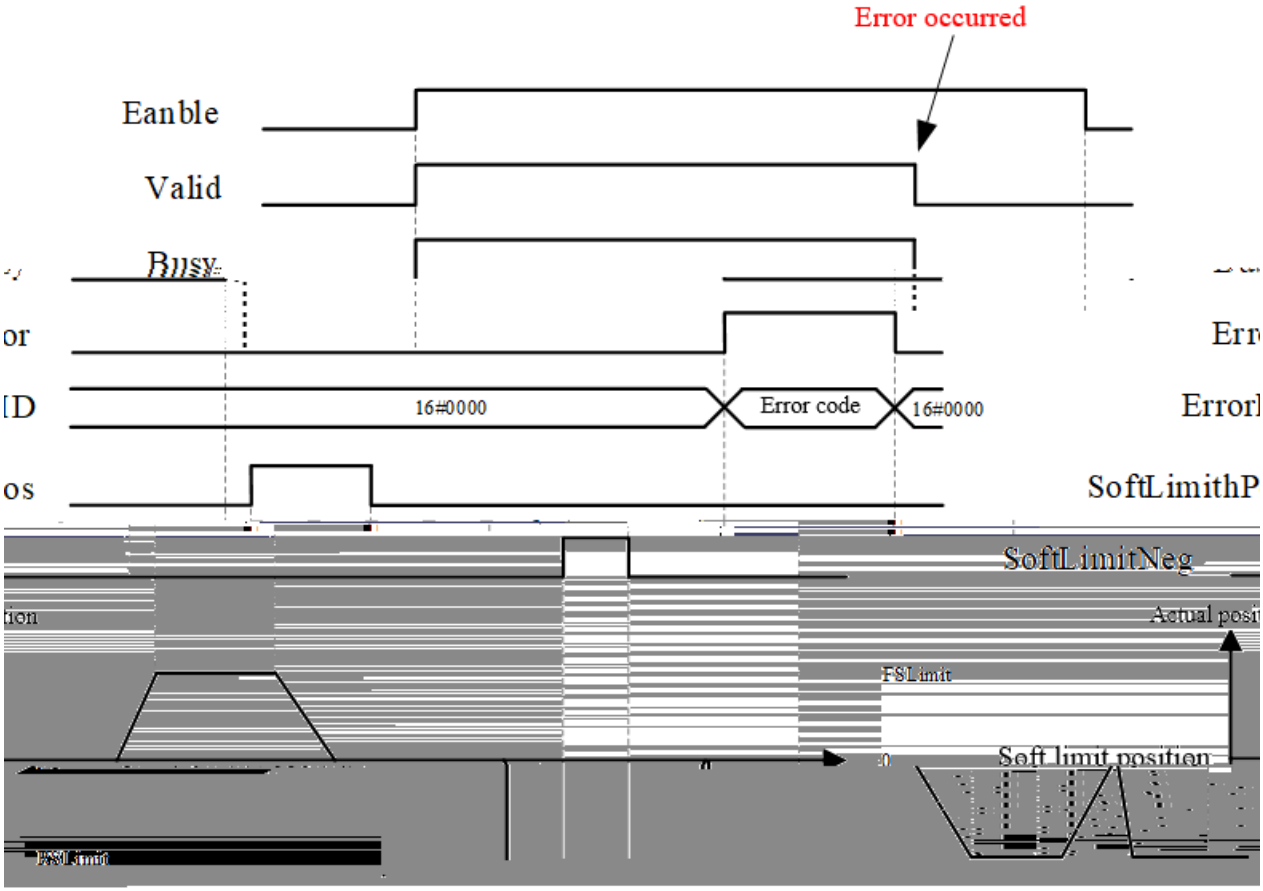
```
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
HomeDIChannel := 2,
Axis := Axis0,
Done=> DoneInit,
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
```

(2) Read axis error

```
MC_ReadAxisError1(
Enable := ReadAxisErrEnable,
Axis := Axis0,
Valid=>ValidReadAxisErr,
```

	ErrorID	Command error code	YES	0	-	SMC_ERROR
	LimitSwitchPos	Positive hard limit active	YES	FALSE	TRUE/FALSE	BOOL
	LimitSwitchNeg	Negative hard limit active	YES	FALSE	TRUE/FALSE	BOOL
	SoftLimitPos					


Eanble
Valid
Busy
Error
16#0000
Error code
16#0000
ErrorID
LimitSwitchPos
hNeg
LimitSwitc
annel
Positive limit ch



5.23.3 Examples

This example uses the SMC_AxisInit functional block to illustrate the initialized positive and negative hard limit attribute configuration and channel configuration results.

The function block SMC_ReadAxisLimitStatus is used to obtain the positive and negative hard limit channel status and positive and negative soft limit status of Axis0.

The function block SMC_ReadAxisLimitStatus output pin LimitSwitchNeg is TRUE when the negative limit switch channel (%IX4.0) logic is active.

Variable

No.	Variable Name	Variable Type	Initial Value	Variable Description
1	SMC_ReadAxisLimitStatus1	SMC_ReadAxisLimitStatus		Read axis limit status command
2	ReadAxisLimitStatus_En	BOOL	FALSE	Enable reading axis limit command
3	ReadAxisLimitStatus_Valid	BOOL	FALSE	Valid flag of reading axis limit status
4	ReadAxisLimitStatus_Busy	BOOL	FALSE	Read axis limit status busy flag
5	ReadAxisLimitStatus_Error	BOOL	FALSE	Error flag for reading axis limit status
6	ReadAxisLimitStatus_ErrorID	SMC_ERROR	MC_ALM_NO_ERROR	Read axis limit status error ID
7	LimitPosStatus	BOOL	FALSE	Positive limit status
8	LimitNegStatus	BOOL	FALSE	Negative limit status
9	SoftLimitPosStatus	BOOL	FALSE	Positive soft limit status

```
Axis := Axis0,
Done=>AxisInit_Done,
Busy=>AxisInit_Busy,
Error=>AxisInit_Error,
ErrorID=>AxisInit_ErrorID);
```

(2) Then call the command SMC_ReadAxisLimitStatus to obtain limit status information.

```
SMC_ReadAxisLimitStatus1(
Enable := ReadAxisLimitStatus_En,
Axis := Axis0,
Valid=>ReadAxisLimitStatus_Valid,
Busy=>ReadAxisLimitStatus_Busy,
Error=>ReadAxisLimitStatus_Error,
ErrorID=>ReadAxisLimitStatus_ErrorID,
LimitSwitchPos=>LimitPosStatus,
LimitSwitchNeg=>LimitNegStatus,
SoftLimitPos=>SoftLimitPosStatus,
SoftLimitNeg=>SoftLimitNegStatus);
```

5.23.4 Precautions

If the positive and negative hard limit status are valid at the same time, the positive soft limit setting value is less than or equal to the negative soft limit setting value, or the soft limit status and the hard limit status are valid at the same time, an error will be reported by the function block.

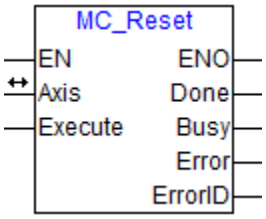
When the travel mode is set to cyclic travel (RepMode = 1 or 2), the limit status is invalid and has no practical significance.

5.23.5 Reference

Command SMC_AxisInit.

5.24 MC_Reset (Reset Axis Error)

Reset axis error.



5.24.1 Parameter

Parameter Type	Title	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Axis	Axis name	No	-	-	AXIS_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
OUTPUT	Done	Completed	YES	FALSE	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR

5.24.2 Functions

Axis errors can be reset by this command, which is described in detail as follows:

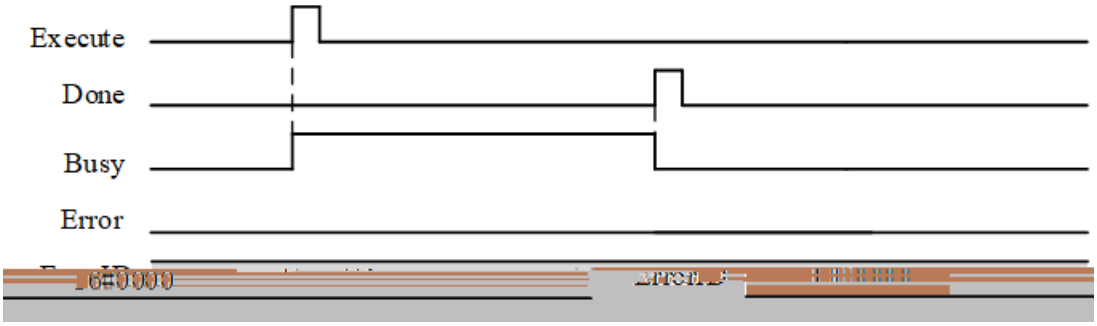
(1) This command can reset the faults of Ethercat bus axis and virtual axis.

(2) This command is valid when triggered by the rising edge of Execute, and Busy (busy flag) is TRUE during reset. After reset, the Done pin is TRUE, other pins are FALSE, Error after reset fails is TRUE, and ErrorID outputs corresponding error code.

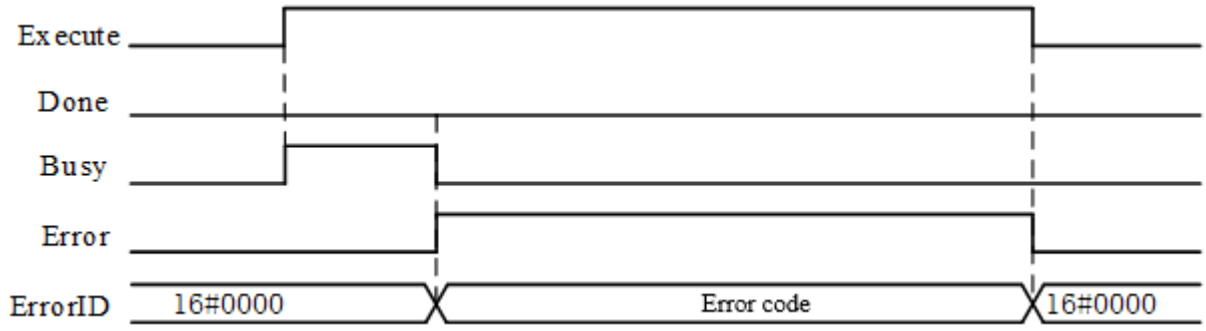
(3) After a successful reset, if the driver is enabled, the axis PLCopen status machine will enter the StandStill state; otherwise, it will enter the Disabled state.

Function block timing diagram

(1) Timing diagram of axis failure but successful reset.



(2) Timing diagram of axis reset failure.



5.24.3 Examples

Create an MC_ Reset example program to reset the axis error.

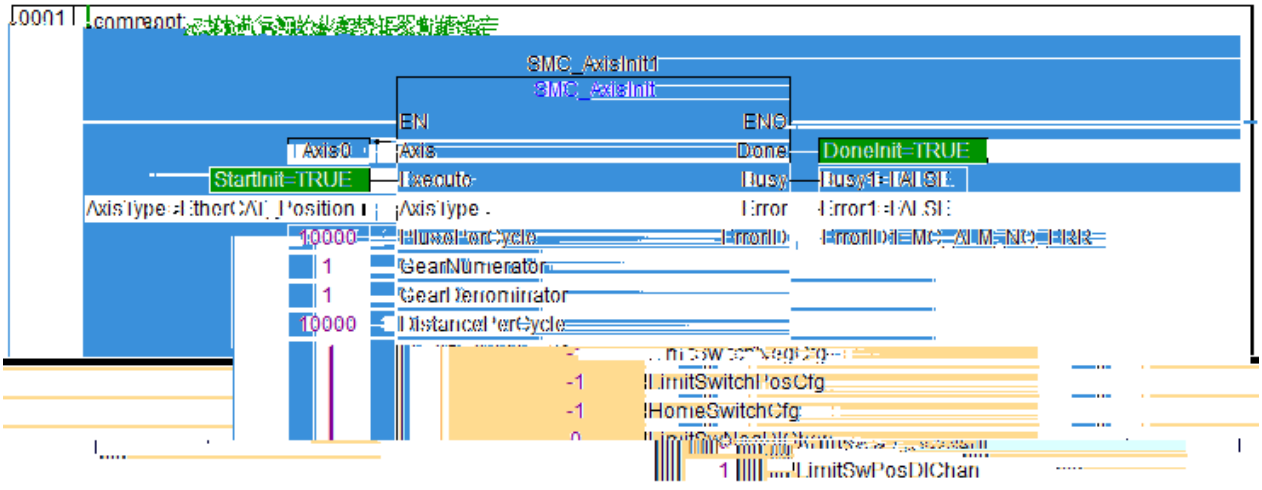
Primary variables

Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0
StartInit	BOOL	FALSE	Change to TRUE at the start of initialization
DoneInit	BOOL	FALSE	Change to TRUE upon completion of initialization
ResetExe	BOOL	FALSE	Change to TRUE at the start of reset
DoneReset	BOOL	FALSE	Change to TRUE when reset is successful
BusyReset	BOOL	FALSE	Change to TRUE at reset
ErrorReset	BOOL	FALSE	Change to TRUE on reset failure

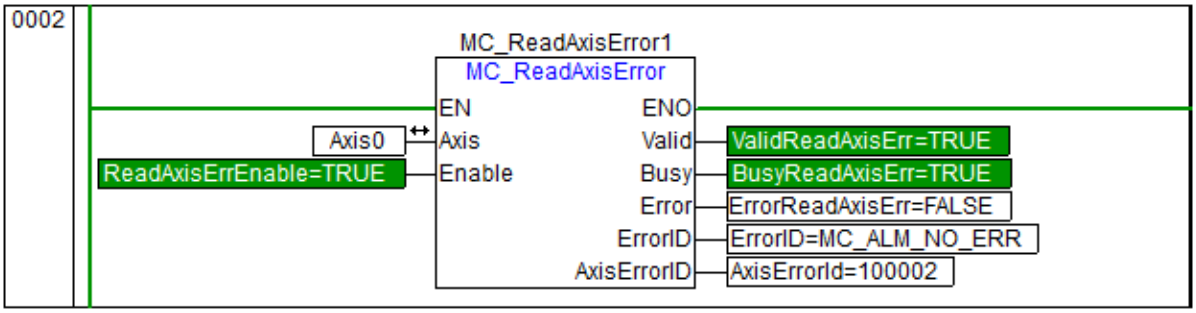
LD program implementation

Use the ladder diagram to form a program. Set MC_ Reset Execute (rising edge trigger) to TRUE, sample procedure is as follows:

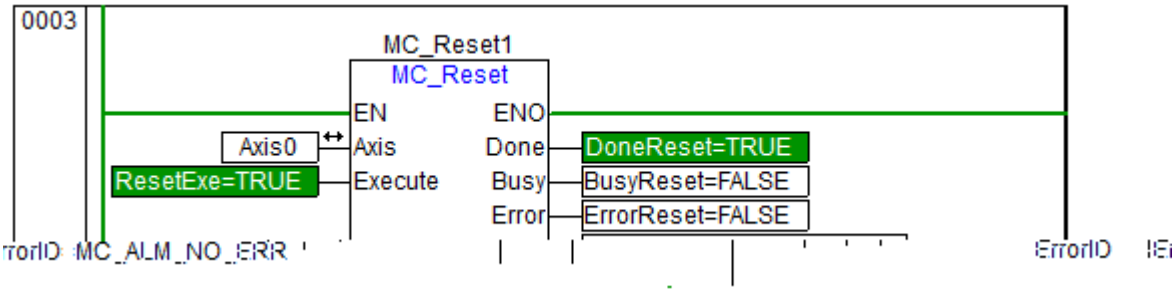
(1) First, initialize the axis



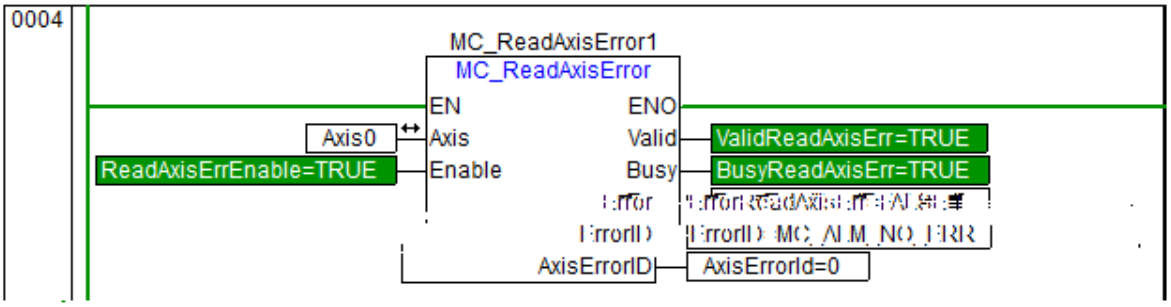
(2) Servo offline error



(3) Reset with MC_Reset



(4) When reading the axis error, it indicates that the error has been successfully reset.



ST language program

(1) Initialize the axis number and axis type settings

```
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
```

Type			or Not	Value		
	Axis	Axis name	No	-	-	AXIS_REF
IN_OUT	TriggerInput	Latching configuration parameters	No	-	See data structure MC_TRIGGER_REF	MC_TRIGGER_REF

This command does not support imaginary axis mode. When the command is running, if the state of the corresponding axis jumps to ErrorStop, this functional block reports an error and releases the latch channel occupied by the command.

Functions supported by this directive:

Support probe capture of two latched channels;

Support driver DI terminal trigger and Z-phase signal as the trigger source, which needs to be supported by servo driver;

Supporting rising edge, falling edge and double-edge trigger types, which needs to be supported by the servo driver;

The command can be configured as a single shot or continuous shot. When it is configured as single trigger, if the Done signal output is valid, it means that the single position capture is completed and the command execution ends; when it is configured as continuous trigger mode, after the position capture is completed, the Done outputs a valid signal, resets again after an IEC scanning cycle, and the command automatically starts to detect new probe input signals. In the continuous firing mode, if the input frequency of the probe signal is greater than the frequency of the IEC scan cycle, this command will lose part of the probe signal.

Window Mode Description

When the input parameter WindowOnly is FALSE, the window detection function is invalid. As long as the probe input signal is valid, the position of the corresponding axis can be latched.

When the input parameter WindowOnly is TRUE, the window detection function is effective. The probe capture command now checks the probe signal and latches the position only if the current position of the axis is within the window.

(1) Limited stroke

Effective window range: FirstPosition ≤ Window Range ≤ LastPosition, when the input parameter FirstPosition > LastPosition, the function block reports an error.

(2) Cyclic stroke

Both FirstPosition ≤ LastPosition and FirstPosition > LastPosition during cyclic travel can be configured according to actual usage.

When FirstPosition > LastPosition, the setpoint will span the upper and lower limit positions of the cyclic stroke.

The function block reports an error when the set v.

Ž

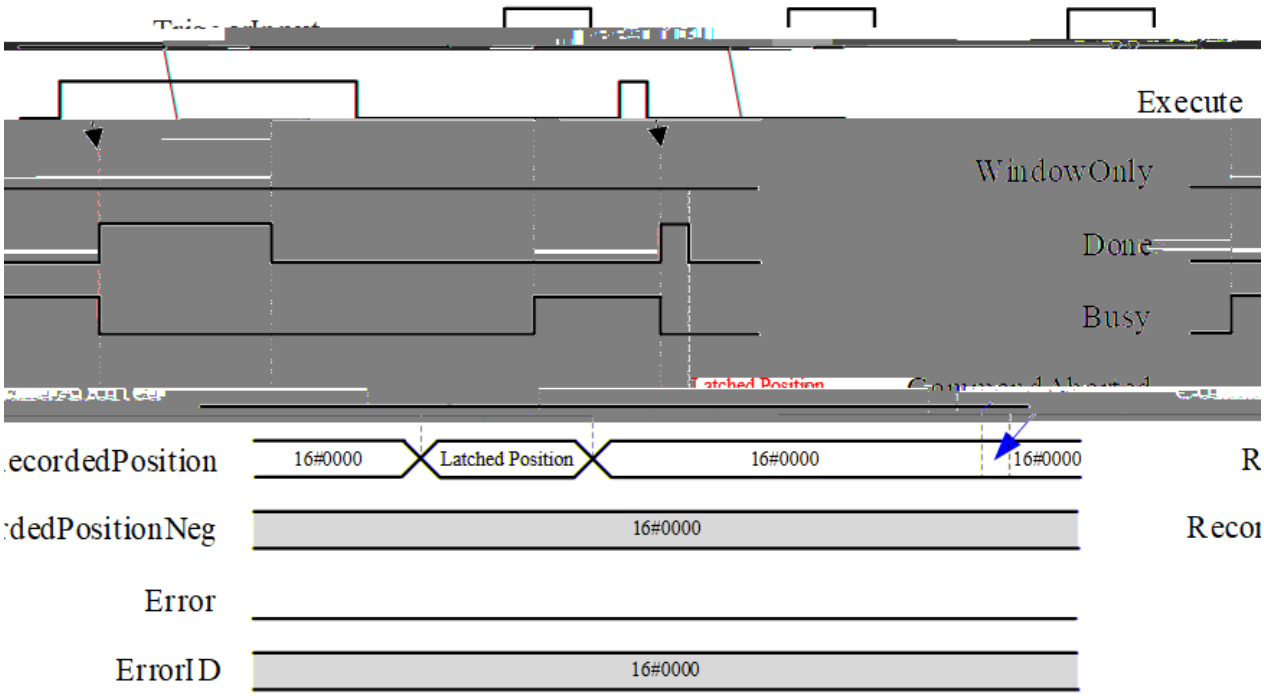
probe commands are defined in the program and the probe IDs are different, the two probe commands will operate independently without affecting each other. However, if the probe IDs are the same, an error will be reported for the probe command to be executed later. If only one probe command is defined in the program, and the rising edge is triggering this command again while it is running, the function block will report an error and abort the running probe command to release the probe ID occupied by the command. If a probe command is called repeatedly, the function block will report an error upon enabling and will not execute the command. This is not a valid use case, and repeating the same function block instance will trigger a warning during configuration compilation.

During normal operation, the MC_TouchProbe command can be aborted through the MC_AbortTrigger command and the occupied probe can be released. When the MC_AbortTrigge output pin Done is TRUE, it indicates that the interruption is successful. At this time, the MC_TouchProb command output pin CommandAborted will be set.

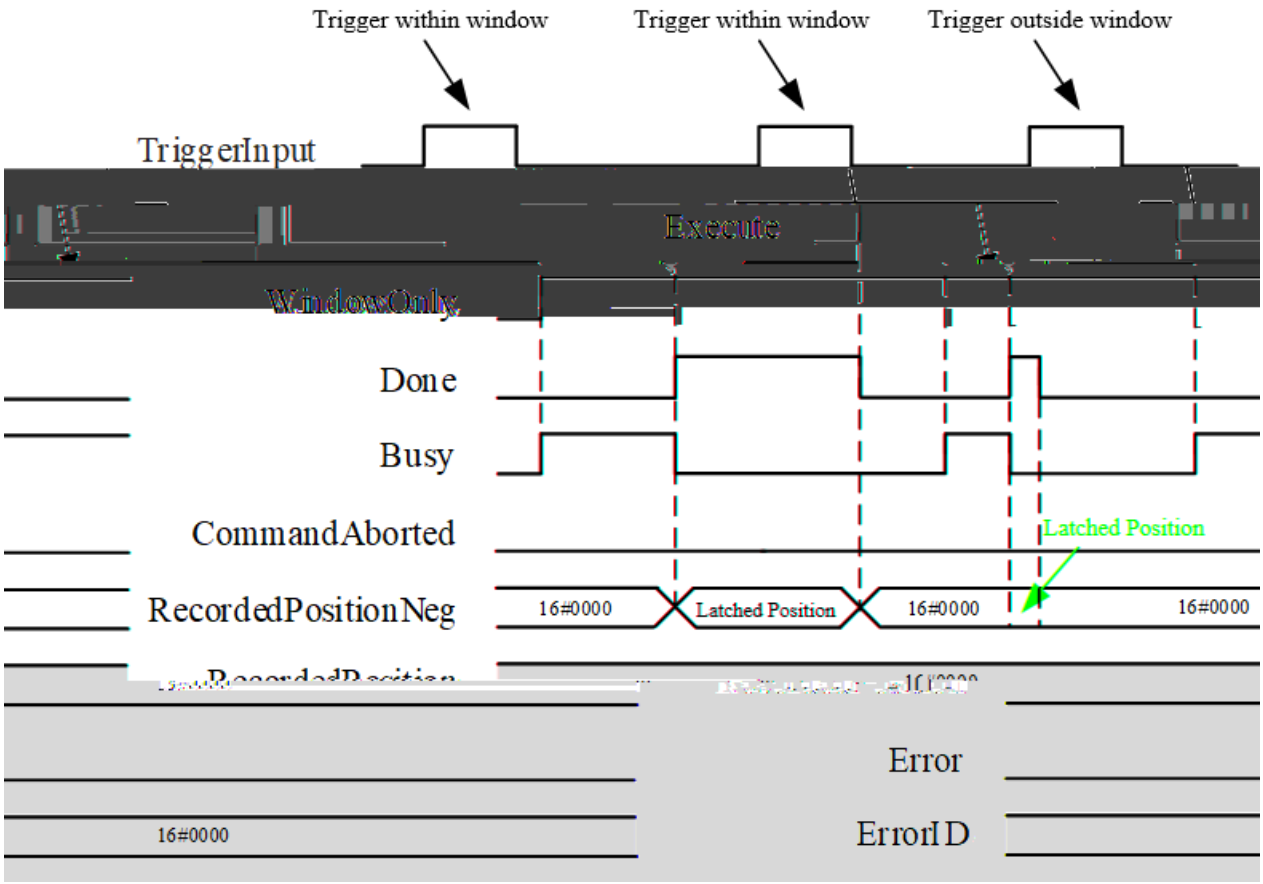
When MC_Home and HMC_GantryInit commands are used, if the homing mode is configured as Z-phase signal correlation, probe 1 will be used to capture Z-phase signals. At this time, the channel of probe 1 of the axis will be occupied, and other commands cannot be used or interrupted. If Probe 1 is already occupied when initiating Z signal-related homing with MC_Home or HMC_GantryInit, an error will be reported.

Function block timing diagram

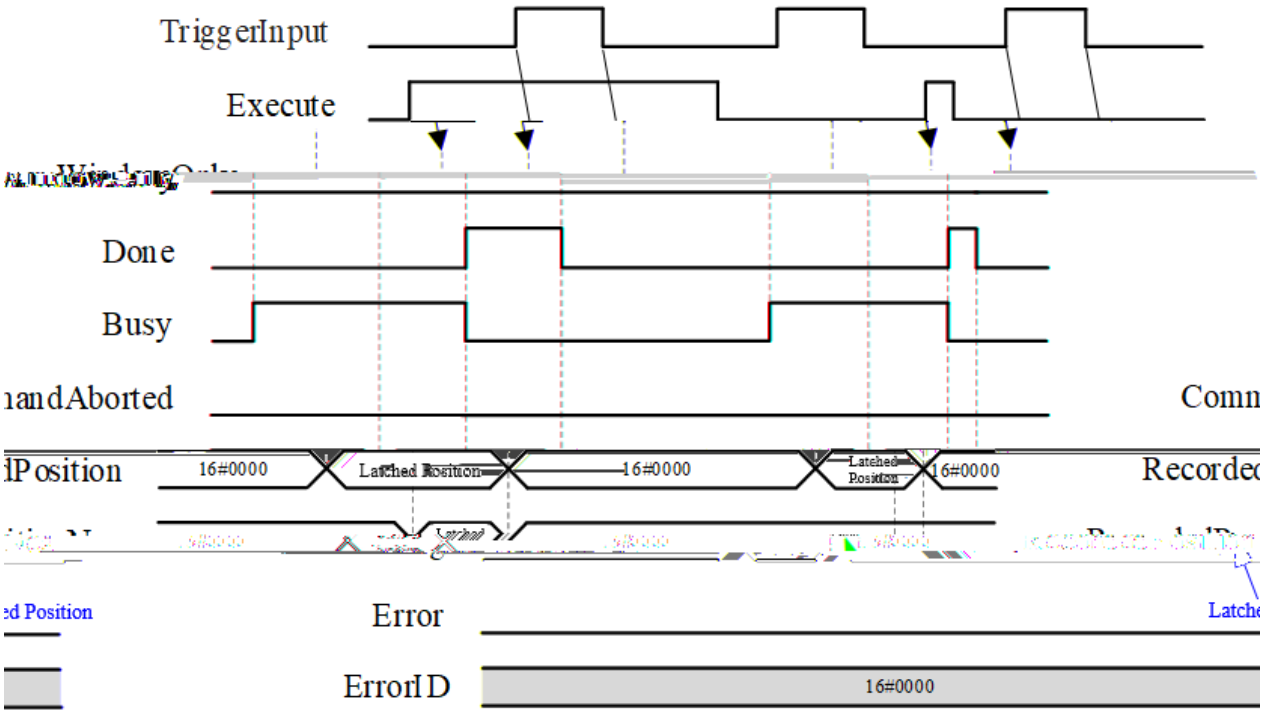
(1) The rising edge of probe 1 is valid, the DI terminal is triggered in single-trigger mode, and the window function is invalid.



(2) The falling edge of the probe 1 is valid, and the DI terminal is triggered in single-trigger mode. The window function is valid.



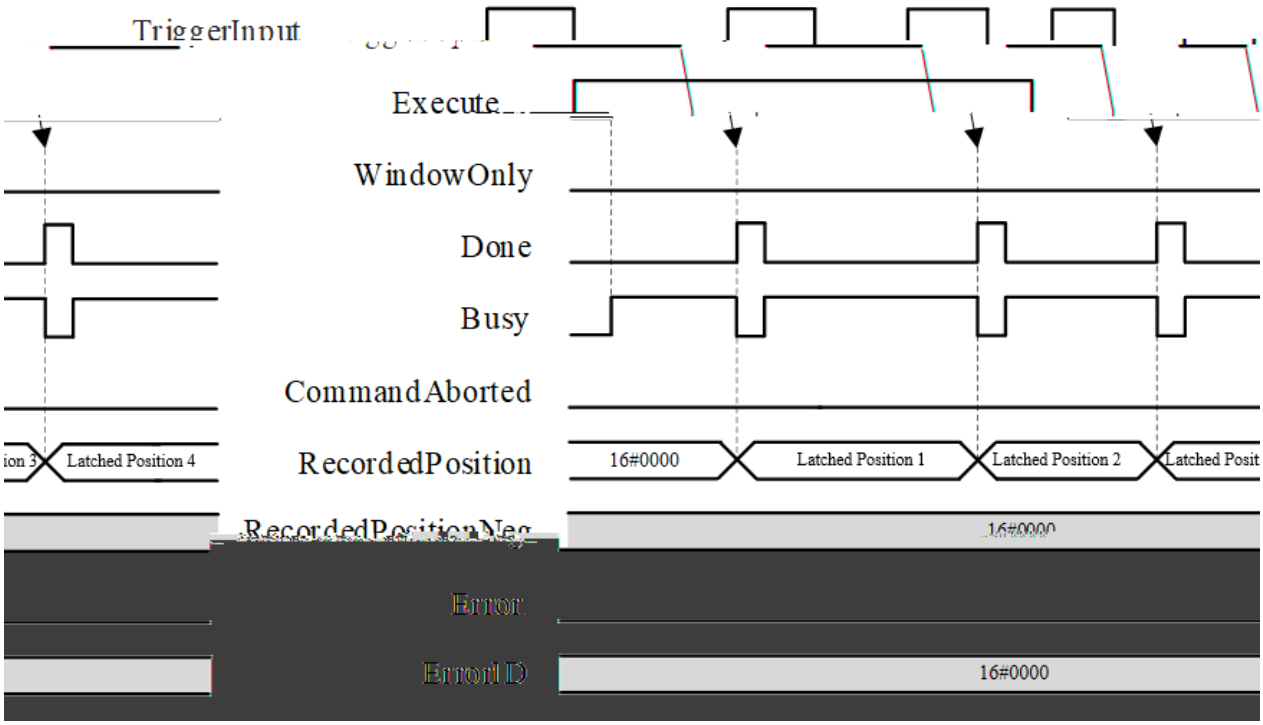
(3) The rising and falling edges of probe 1 are valid, the DI terminal is triggered in single-trigger mode, and the window function is invalid.



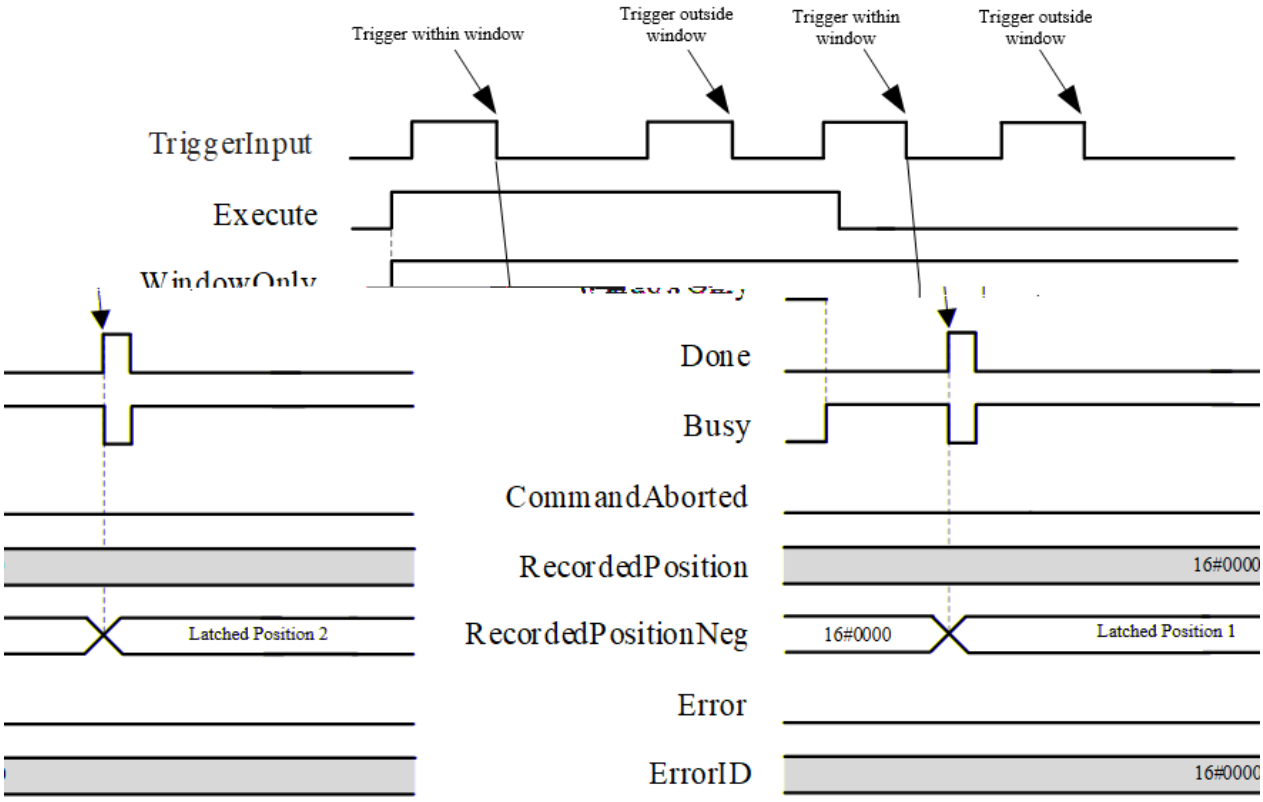
(4) The rising and falling edges of probe 1 are valid, the DI terminal is triggered in single-trigger mode, and the window function is valid.

(5) The rising edge of probe 1 is valid, the DI terminal is triggered in continuous trigger mode, and

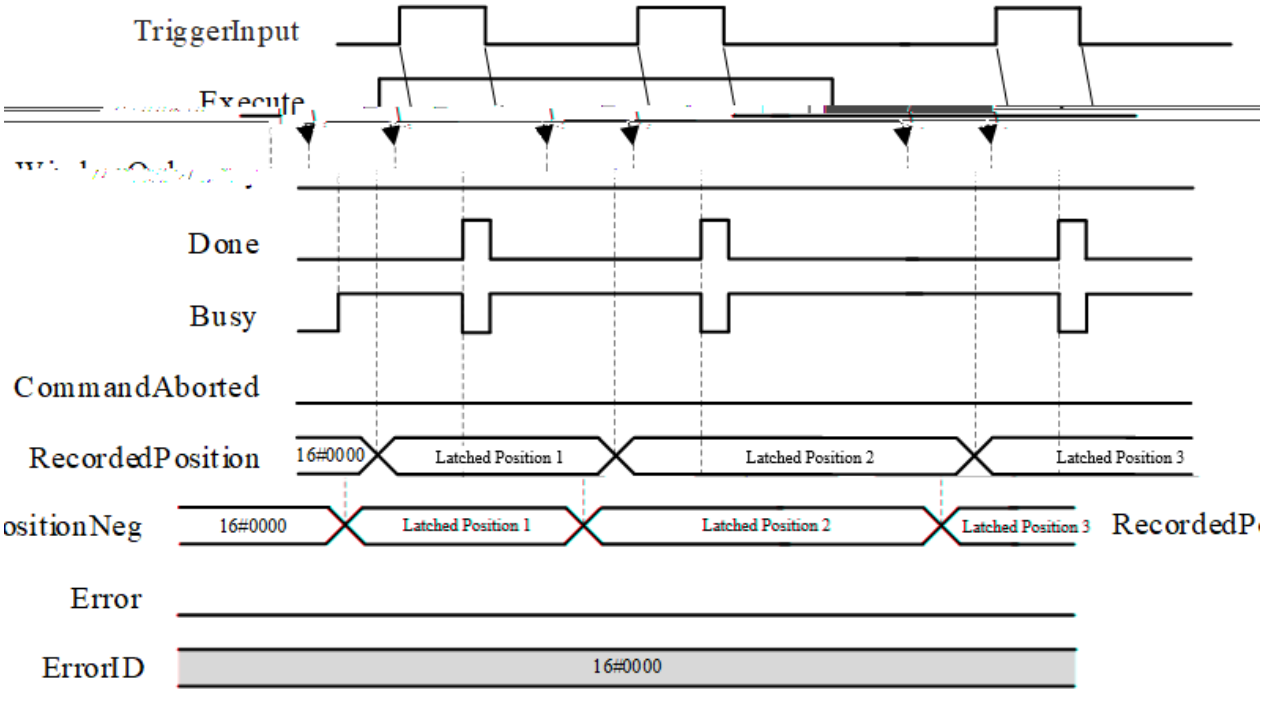
the window function is invalid.



(6) The falling edge of probe 1 is valid, the DI terminal is triggered in continuous trigger mode, and the window function is valid.

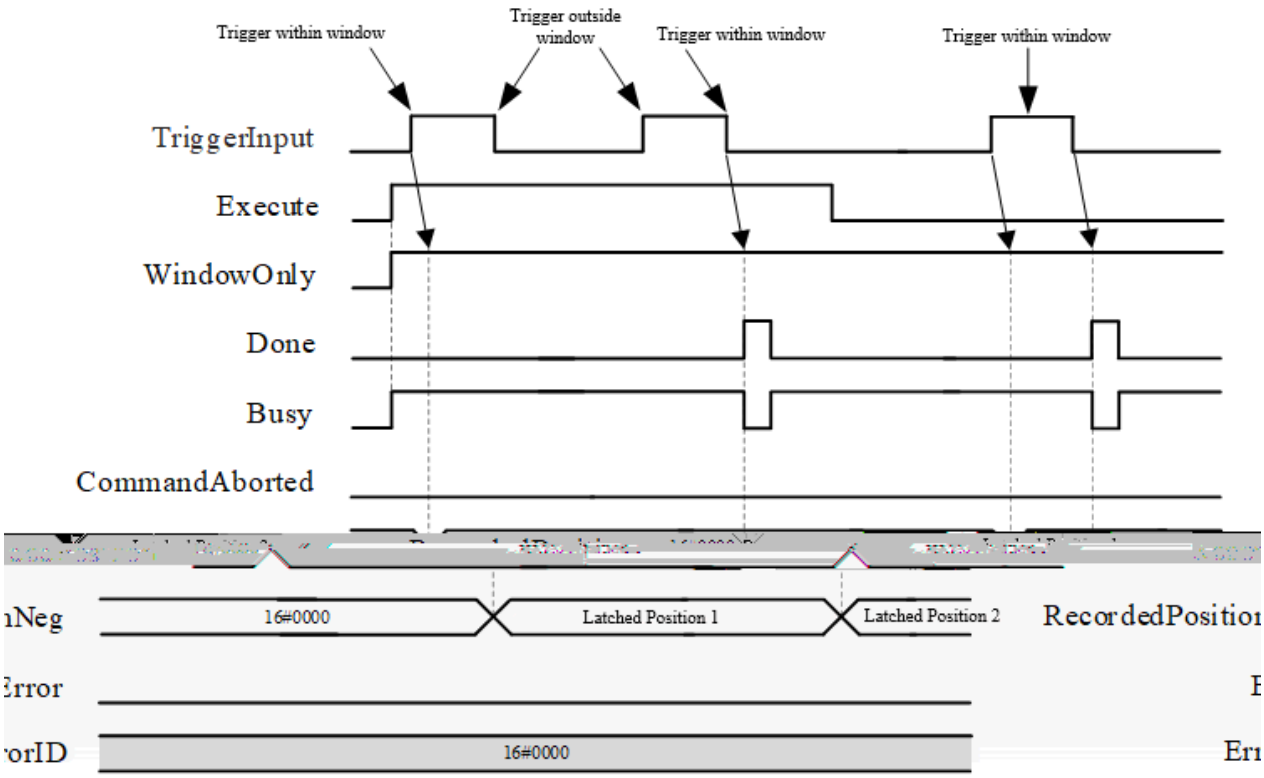


(7) The rising and falling edges of probe 1 are valid, the DI terminal is triggered in continuous trigger mode, and the window function is invalid.

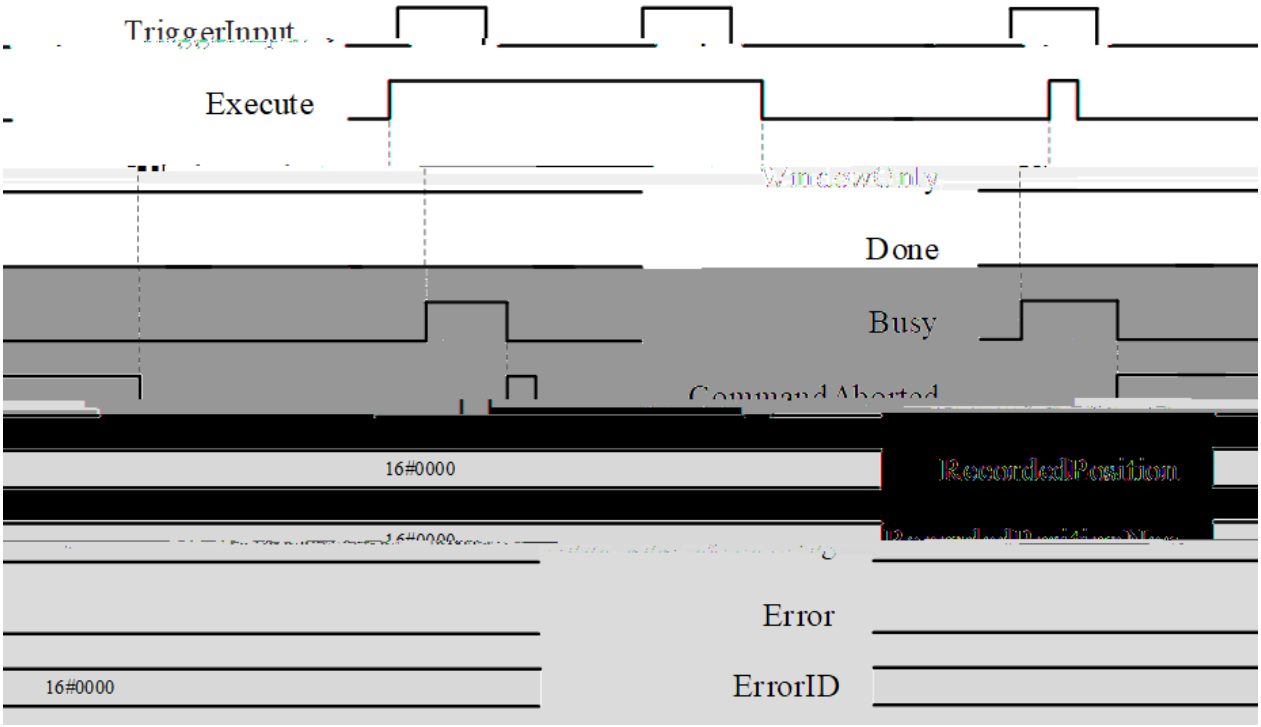


(8) The rising and falling edges of probe 1 are valid, the DI terminal is triggered in continuous

trigger mode, and the window function is valid.



(9) The rising edge of probe 1 is valid, the DI terminal is triggered, and the window function is invalid, which is interrupted by MC_AbortTrigger command.



(10) The rising edge of the probe 1 is valid, the DI terminal is triggered, the window function is invalid, and the command fails.

5.25.3 Examples

This example uses the function block MC_TouchProbe to grab EtherCAT servo probe signals and record positions.

The trigger conditions are set as follows:

Use probe 1;

Triggered by rising edge;

The trigger signal is the driver IO terminal;

Continuous triggering;

No window trigger is used.

Before using the probe capture function, you need to use the SMC_AxisInit command for axis type setting and user unit configuration, and also configure the relevant PDO mapping.

The DI function needs to be configured on the servo driver side as the trigger signal of Probe 1. Please refer to the servo related manual for details.

Variable №a 123.96 2 (ab)1.7 (l)1.9 (e T)68.4 (y)53.3 (p)1.7 (e)]TJD121.8 8esc Ty

No.	Variable Name	Variable Type	Initial Value	

4 5	TouchProbe_Window	BOOL	FALSE	Enable window capture

	TriggerInput	Latching configuration parameters	No	-	See data structure MC_TRIGGER_REF	MC_TRIGGER_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
	Done					

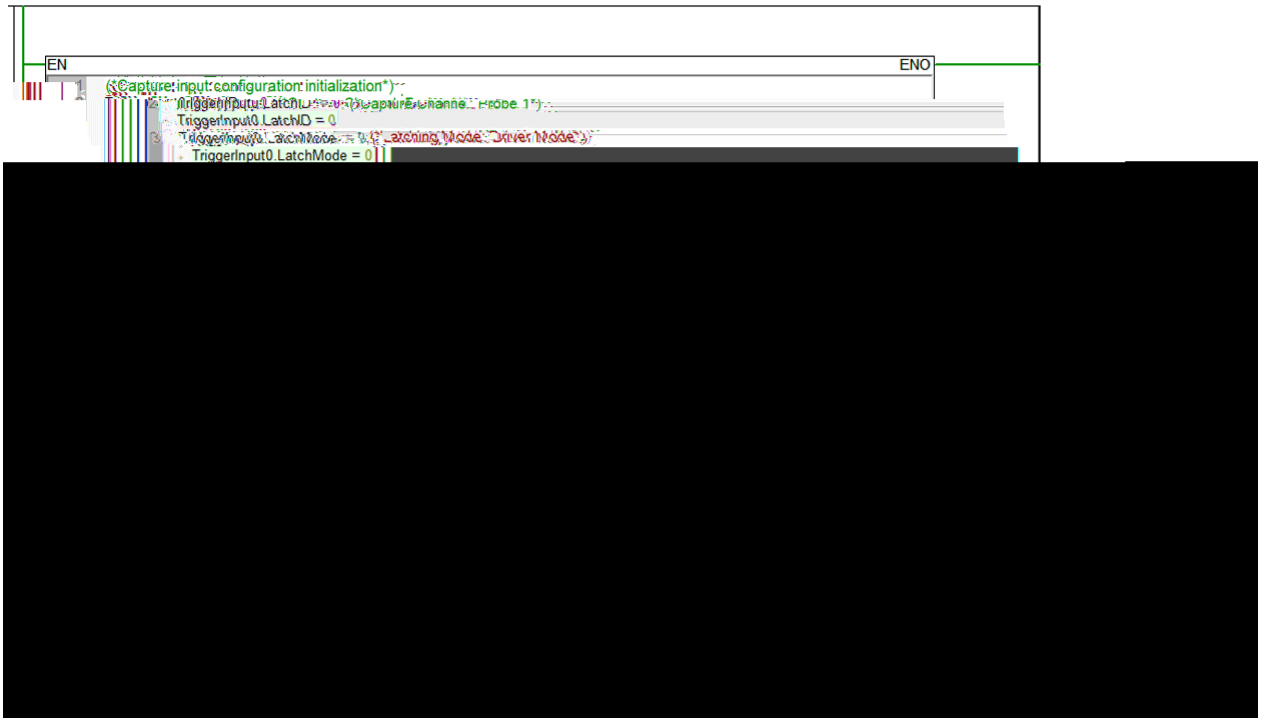
OUTPUT

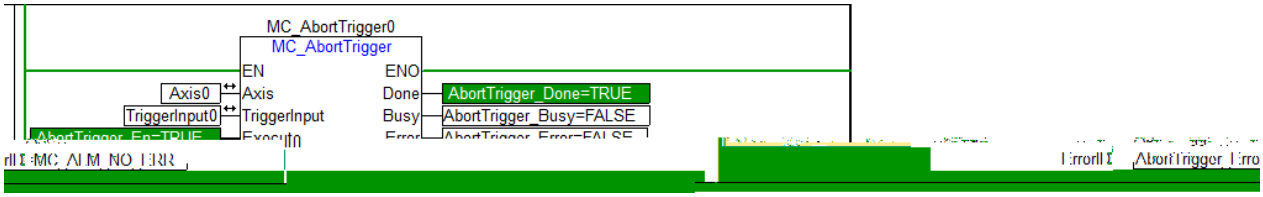
initiated by the MC_TouchProbe command. After interruption, the CommandAborted pin of the MC_TouchProbe command is set to TRUE, and the Done pin of the MC_AbortTrigger command is set to TRUE.

The example Description of MC_TouchProbe is consistent with that in the chapter of MC_TouchProbe command, so it will not be repeated in this chapter.

Variable

No.	Variable Name	Variable Type	Initial Value	Variable Description





ST program implementation

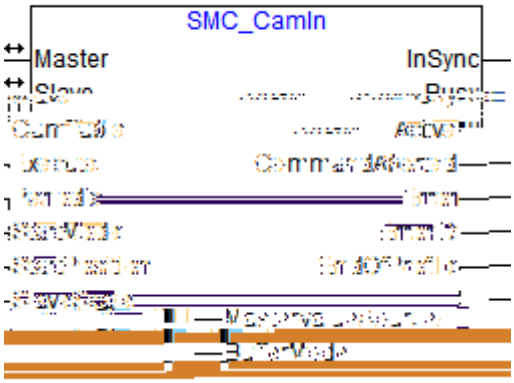
(1) Call the MC_AxisInit command to initialize axis-related information.

```
MC_ReadAxisInfo1(
Enable := ReadAxisInfo_En,
Axis := Axis0,
Valid=>ReadAxisInfo_Valid,
Busy=>ReadAxisInfo_Busy,
Error=>ReadAxisInfo_Error,
ErrorID=>ReadAxisInfo_ErrorID,
HomeAbsSwitch=>ReadAxisInfo_HomeAbsSwitch,
LimitSwitchPos=>ReadAxisInfo_LimitSwitchPos,
LimitSwitchNeg=>ReadAxisInfo_LimitSwitchNeg,
Simulation=>ReadAxisInfo_Simulation,
CommunicationReady=>ReadAxisInfo_ComRdy,
ReadyForPowerOn=>ReadAxisInfo_RdyPowerON,
PowerOn=>ReadAxisInfo_PowerON,
IsHomed=>ReadAxisInfo_Homed,
AxisWarning=>ReadAxisInfo_Warning,
AxisFault=>ReadAxisInfo_Fault);
```

Chapter 6 Synchronous Motion Instructions

6.1 SMC_CamIn (Electronic CAM)

This command realizes the electronic cam function.



6.1.1 Parameter

Command Parameters

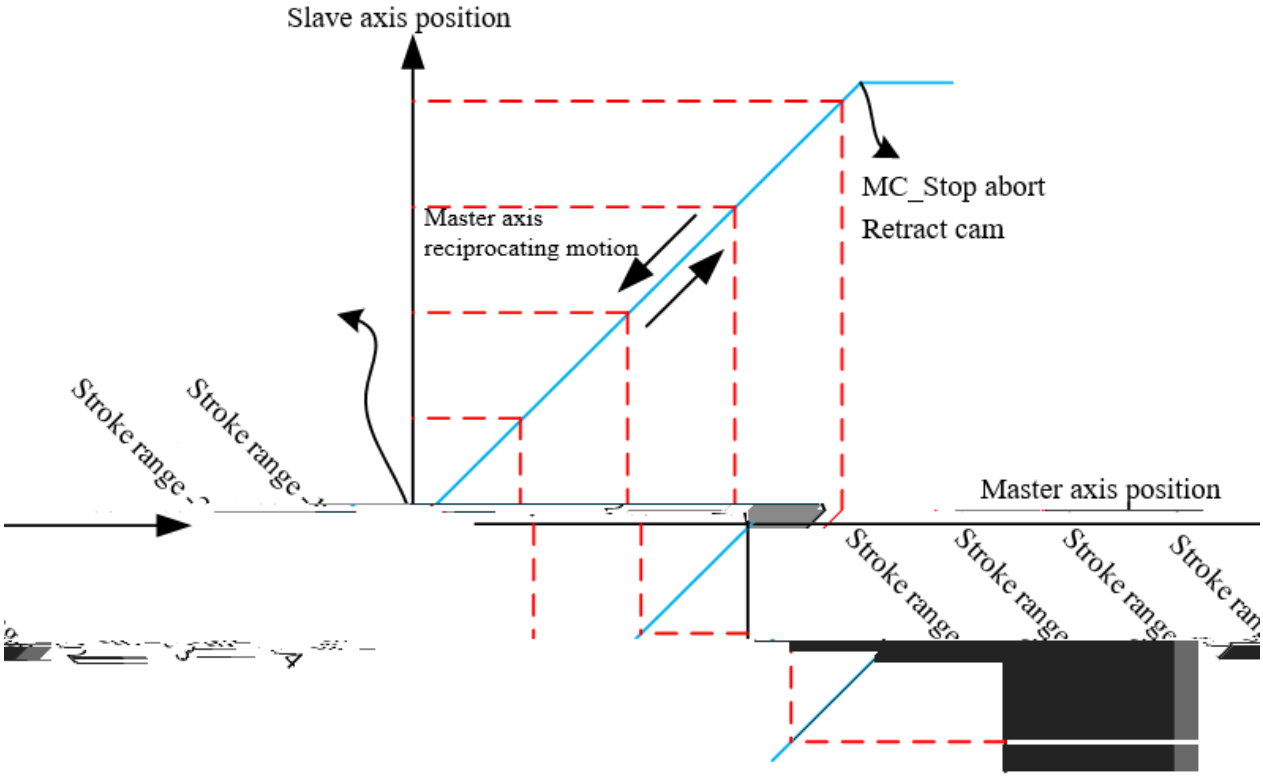
Parameter Type	Name	Description	Empty or Not	Default Value	Scope	Data Type
	Master	Master Axis name	No	-	-	AXIS_REF
IN_OUT	S re14>ve (er)]TJ E		No		-	AXIS_REF

		When DI trigger is started, it is the bit offset mapped by DI point in Area I. For example, the bit offset of %IX100.1 in area I is 100×8+1.			0 ~ (field I byte capacity * 8)-1	

Cam follower

SlaveScale

position start, the left boundary is set to the specified absolute position; The right boundary of the command= left boundary + length of cam stroke section. The command will be canceled if the master moves in the negative direction across the left boundary or in the positive direction across



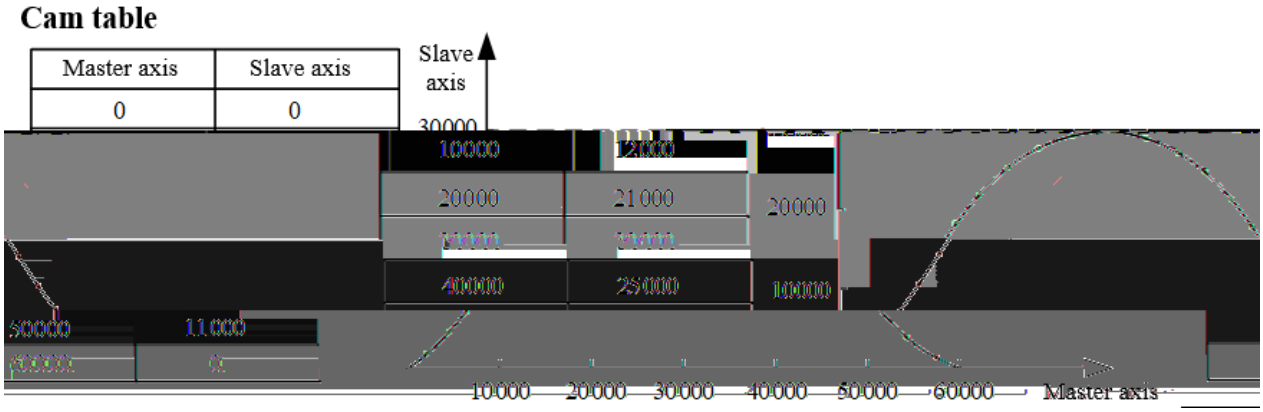
Cyclic immediate start of cam -synchronized Master and slave axis motion trajectory

(3) Cam activation position

After the command activation, wait for the master to reach the StartPosition.

When the master passes through StartPosition, the starting point of the cam position array is executed and the cam command output variable InSync (in cam synchronization) becomes TRUE.

The master and slave positions of a cam position array are specified in relative quantities from the starting point. Interpolate the two cam position data according to the set cam curve interpolation method, and calculate the slave axis position corresponding to the master position, as shown in the following figure.



(4) Abnormal cam position array

An exception is detected triggering this command when the specified cam position array does not exist in the Controller.

(5) Use of cam position array

The same array of cam positions can be specified for several axes. Note: During the execution of this command, when the user rewrites the position of the master in the cam position array to non-incremental data, an error exception will occur when running this command.

(6) Cam activation

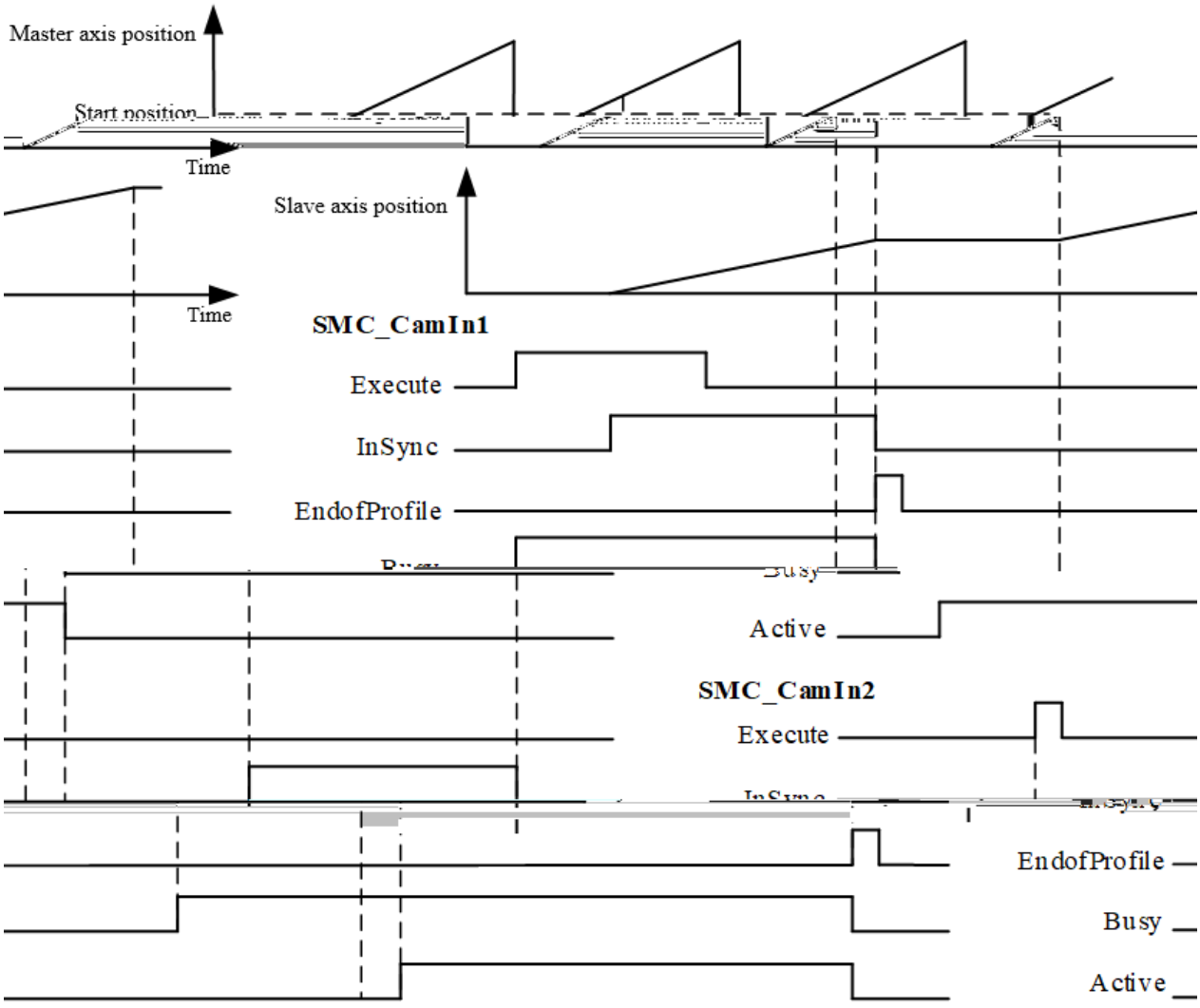
This command can be activated in any state of StandStill, position control, speed control and synchronous control.

The cam start mode includes 0: fixed position start, 10: immediate start and 11: DI trigger start.

Ž Fixed position activation

Cam activation mode is 0: start at fixed position. After the start command, wait for the master camshaft Mpos to cross the cam start position StartPosition from the negative direction and execute the starting point of the cam table. The output variable InSync becomes TRUE.

A cam position array specifies the positions of the master and slave in relative terms, i.e., from the absolute position of each axis at the beginning of the cam position array. For example, in the cyclic stroke mode with a master movement stroke of [0 to 360), when StartPosition is 100. As shown below, the absolute position of a master is the value of the master position from the cam position array plus StartPosition, and the absolute position of a slave axis is the value of the slave axis displacement from the cam position array plus the slave absolute position at the beginning of the cam position array.



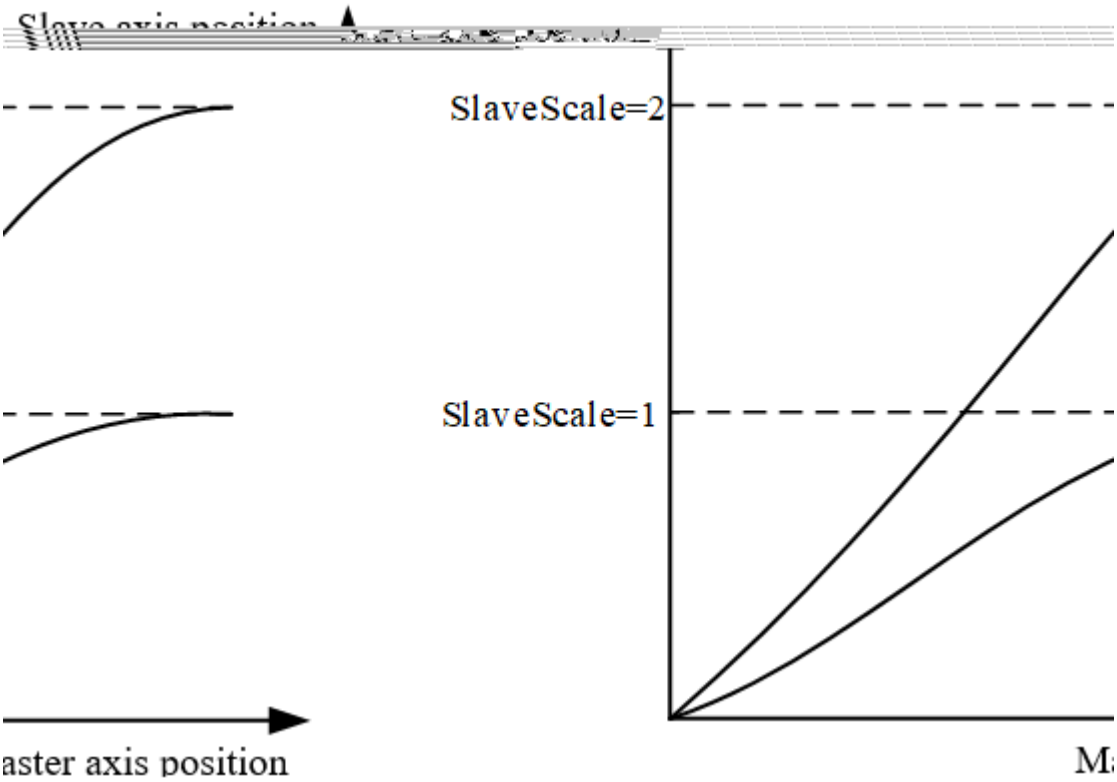
(9) Buffer mode selection

Specify the connection mode between the previous axis action and this axis action.

This command only supports 0: Aborting and 1: Buffered. When this commandn is sent, if BufferMode is 0: Aborting, the currently executing command will be immediately aborted and switched to this command. If BufferMode is 1:Buffered, the buffered command will be automatically started from the cycle when the currently executing command ends normally.

(10) Slave camshaft position data scale factor (scaling factor)

According to the relationship between master phase and slave axis displacement of the specified cam array, modifying the value of scale factor SlaveScale of cam and slave axis position data can increase or decrease the slave axis displacement by a specified ratio (SlaveScale).

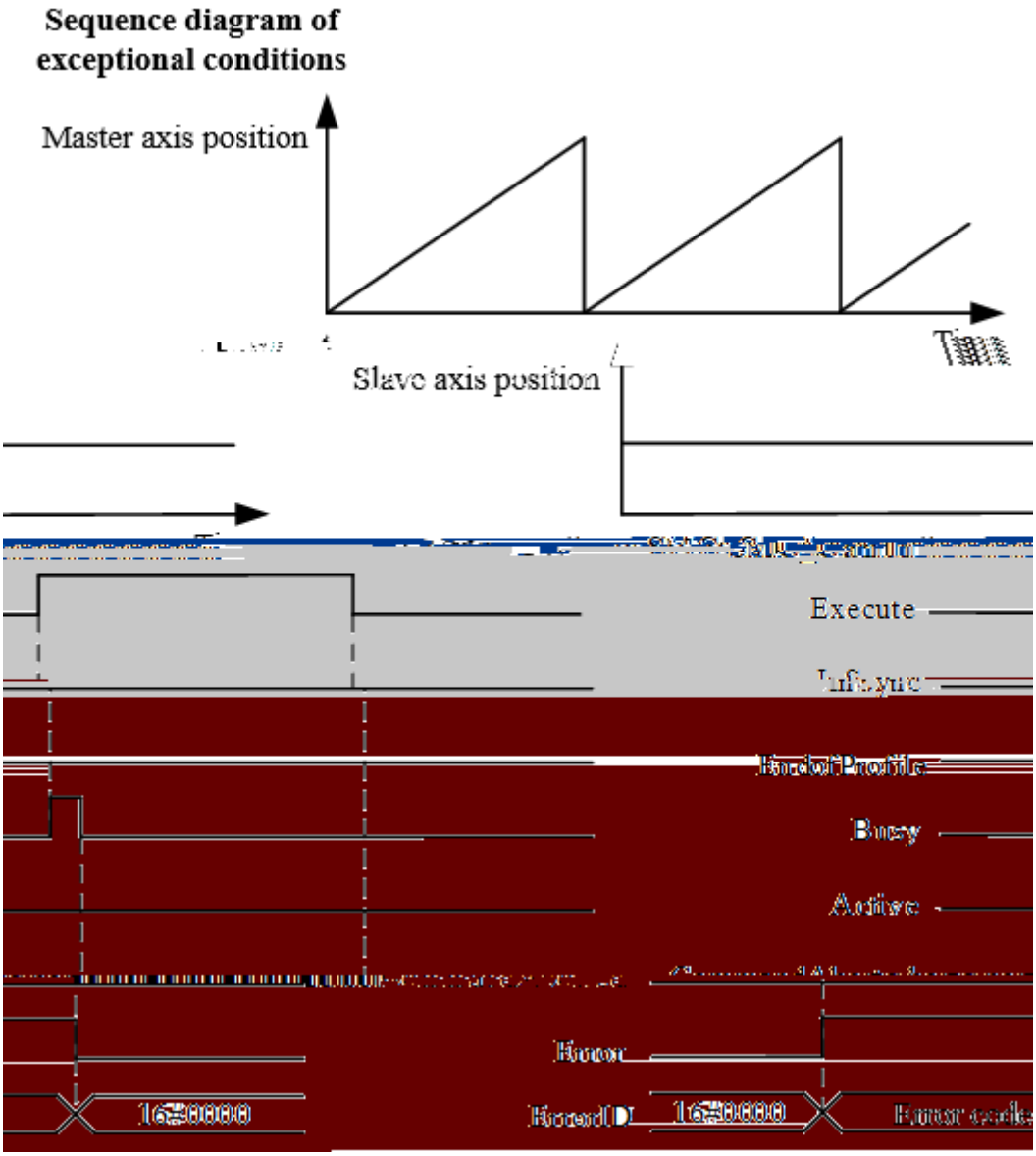


Slave camshaft position data scale factor (SlaveScale)

(11) EndofProfile

When the cam command is in single mode, the cam cycle is completed and the command output pin EndofProfile is set to TRUE and held until the input pin Execute becomes FALSE.

When the cam command is in cycle mode, the first cam stroke is completed, and the command output pin EndofProfile is set to TRUE for one cycle. The command enters the next cam stroke,



6.1.3 Example

In this example, MasterAxis is defined as the main axis, SlaveAxis as the slave axis, and the Variable Type is AXIS_REF.

The master executes the MC_MoveVelocity command, and the slave executes the MC_CamIn command. Retracting the cam command is stopped with MC_Stop command.

LD program implementation

In this example, the periodic mode is established and the cam command is started immediately. Set Periodic (Repeat Mode) = TRUE to repeat the action.

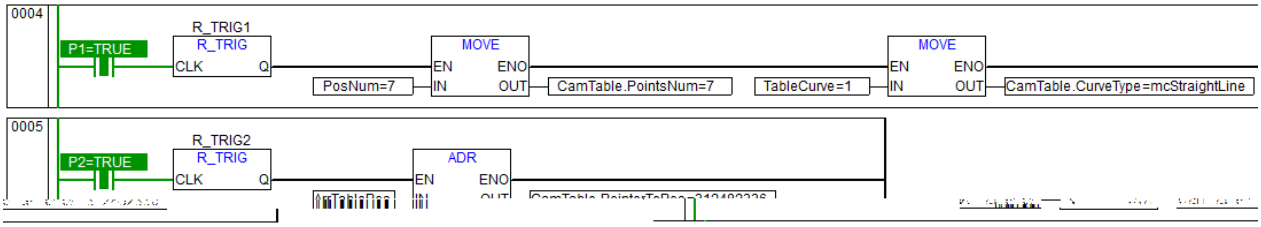
(1) Primary variables of the example program

position arrays. The initialization of the cam data structure variable CamTable is completed.

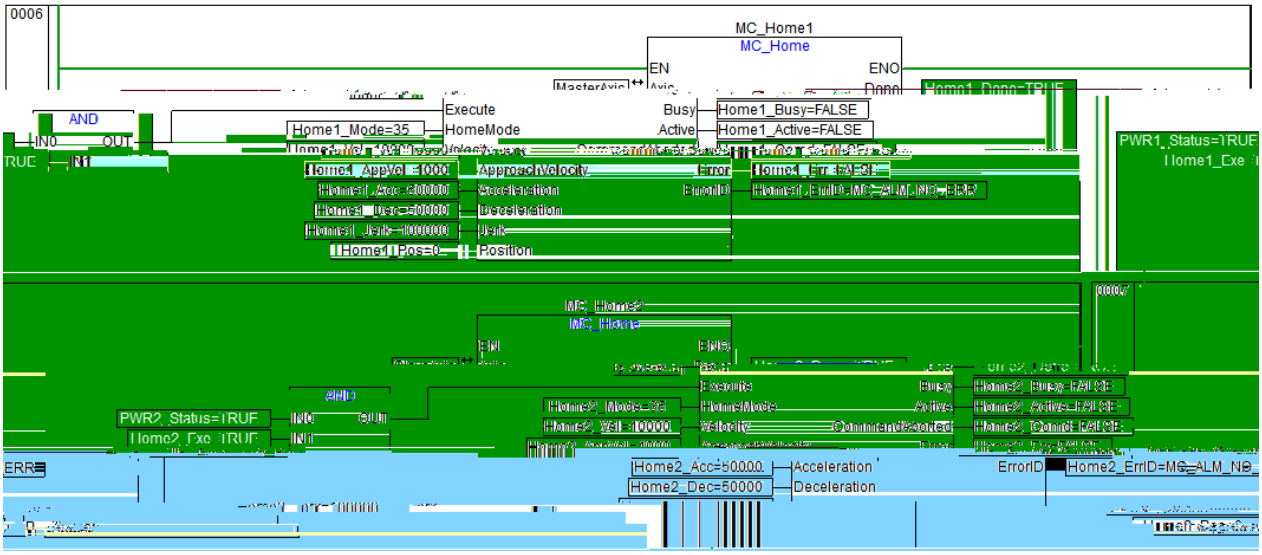
ArrTablePos [0][PosNum] ={0,60,120,180,240,300,360};

ArrTablePos [1][PosNum] ={0,1000, 2000,3000,2500,2000,1500};

PosNum=7;



(5) The master and slave axes are enabled successfully. If the origin is not determined, execute the origin return command of the master and slave axes to determine the origin. See Sections 0006 and 0007 for origin return. For the usage of MC_Home command, see its introduction section.



(6) See Section 0008. After the homing is completed, execute MC_MoveVelocity command. The master starts to move.

	OF LREAL		Pos[2][PosNum]. PosNum is the number of positions.
Cam_InSync	BOOL	FALSE	Synchronization variable of cam command, which becomes TRUE when the cam command is synchronized from an axis.
Cam_EndProFile	BOOL	FALSE	Variable commanded by the cam to complete the cam cycle, which becomes TRUE when the cam commanded cam cycle is completed.
Switch1	BOOL	FALSE	Axis ready state variable. After axis initialization, Switch to TRUE and the axis is ready.
InitFlag	BOOL	FALSE	If this variable is TRUE, the initialization parameter setting is completed.

(2) Axis initialization

Set the AxisID of MasterAxis to 0 and that of SlaveAxis to 1. Call the SMC_AxisInit command to complete the initialization configuration of master and slave axes. The type of both master and slave axes is 50: EtherCAT_Position, and call the SMC_AxisInit command to complete the initialization configuration of master and slave axes.

(3) Axis enabling

After the initialization of the master and slave axes is completed, and the axis ready status

```
ArrTablePos[0, PointsNum-1]:=60000; (*initialization position cam array*)
ArrTablePos[1,0]:=0; (*initialization position cam array*)
ArrTablePos[1,1]:=5000; (*initialization position cam array*)
…
…
ArrTablePos[1, PointsNum-1]:=10000; (*initialization position cam array *)
CamTable.PointsNum := CurveType;(*cam curve interpolation method*)
CamTable.PointerToPos := ADR(ArrTablePos); (*cam position array take address*)
InitFlag:=TRUE;
END_IF
```

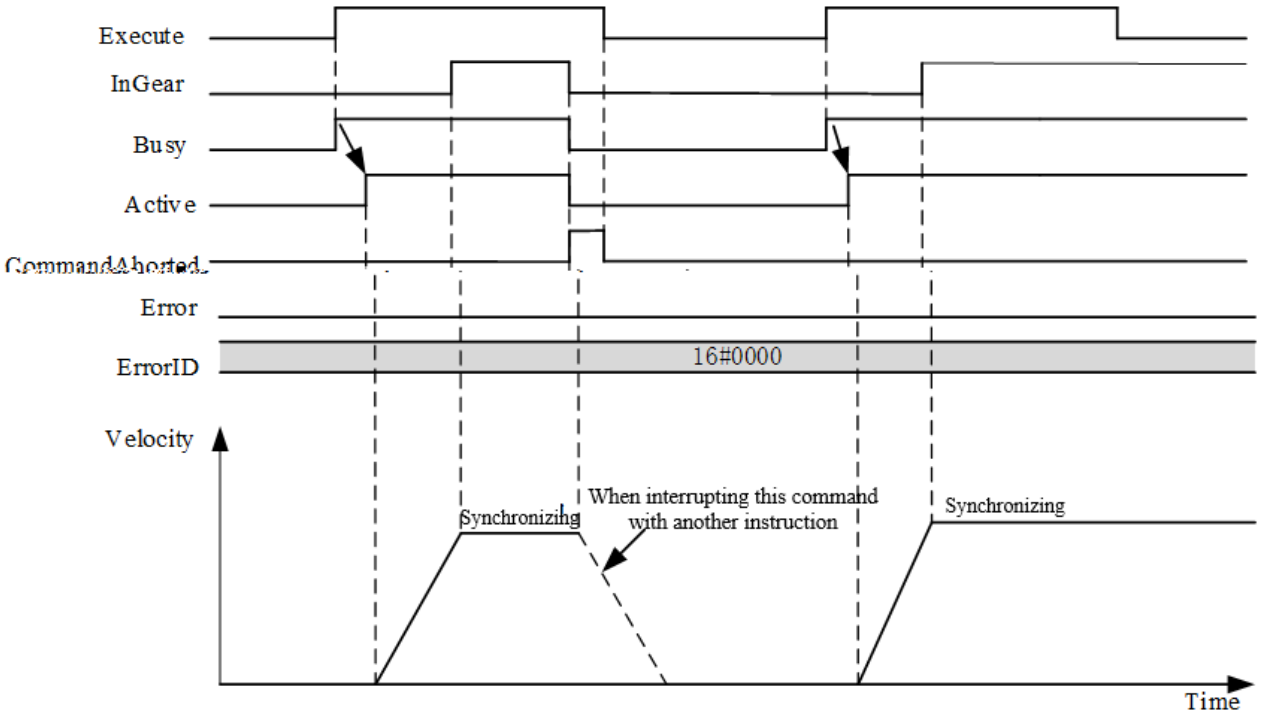
(5) Execute the origin return command of master and slave axis to determine the homing. For the MC_Home command, see its introduction chapter.

```
IF  PWR1_Status THEN
   MC_Home1(
   Execute := Home1_Exe,
   HomeMode := Home1_Mode,
   Velocity := Home1_Vel,
   ApproachVelocity := Home1_AppVel,
   Acceleration := Home1_Acc,
   Deceleration := Home1_Dec,
   Jerk := Home1_Jerk,
   Position := Home1_Pos,
   Axis := MasterAxis,
   Done=> Home1_Done,
   Busy=> Home1_Busy,
   Active=> Home1_Active,
   CommandAborted=> Home1_Comd,
   Error=> Home1_Err,
   ErrorID=> Home1_ErrID);
   MC_Home2(
   Execute := Home2_Exe,
   HomeMode := Home2_Mode,
   Velocity := Home2_Vel,
   ApproachVelocity := Home2_AppVel,
   Acceleration := Home2_Acc,
   Deceleration := Home2_Dec,
   Jerk := Home2_Jerk,
   Position := Home2_Pos,
   Axis := SlaveAxis,
   Done=> Home2_Done,
   Busy=> Home2_Busy,
   Active=> Home2_Active,
   CommandAborted=> Home2_Comd,
   Error=> Home2_Err,
   ErrorID=> Home2_ErrID);
```

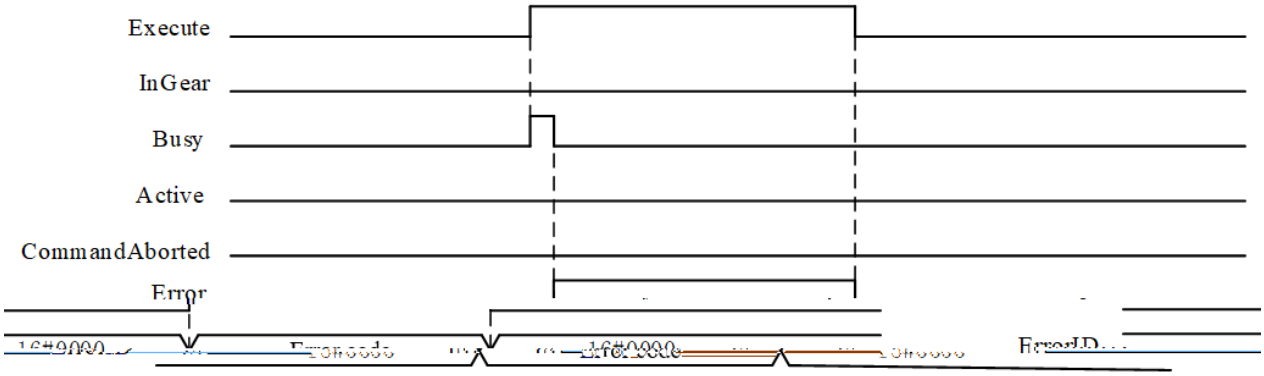
(6) To stop the electronic gear action halfway, use other commands to interrupt or use the MC_Stop command.

Timing diagram

(1) Normal timing diagram



(2) Error timing diagram



6.2.3 Examples

Set up MC_GearIn example program to run electronic gear action commands.

Variable

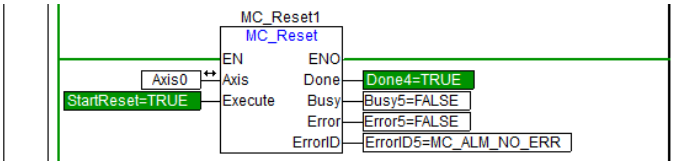
Name	Data Type	Initial Value	Notes
Axis0	AXIS_REF		Axis variable for axis 0

StartInit	BOOL	FALSE	Change to TRUE at the start of initialization
DoneInit	BOOL	FALSE	Change to TRUE upon completion of initialization
PowerEnable	BOOL	FALSE	Change to TRUE at the beginning of enabling
PowerStatus	BOOL	FALSE	Change to TRUE when enabled
StartGear	BOOL	FALSE	TRUE on start command
RatioNumerator	DINT	1	Gear Ratio Numerator
RatioDenominator	UDINT	1	Gear Ratio Denominator
MasterValueSource	MC_SOURCE	0	Master position source selection
InGear	BOOL	FALSE	Change to TRUE during command synchronization
BusyGear	BOOL	FALSE	Change to TRUE when the command is delivered successfully
ActiveGear	BOOL	FALSE	The command changes to TRUE during operation.
ComGear	BOOL	FALSE	Change to TRUE when the command is interrupted
ErrorGear	BOOL	FALSE	Change to TRUE in case of command error

LD program implementation

Use the ladder diagram to bu0.0a6a>BDC 59 a program, determine that the master is axiBDC 5s 1 and tslave a

processing.



ST language program

(1) Initialize the axis

```
SMC_AxisInit1(
Execute := StartInit,
AxisType := 50,
PlusePerCycle := 10000,
GearNumerator := 1,
GearDenominator := 1,
DistancePerCycle := 10000,
LimitSwitchNegCfg := -1,
LimitSwitchPosCfg := -1,
HomeSwitchCfg := -1,
LimitSwNegDIChan := 0,
LimitSwPosDIChan := 1,
HomeDIChannel := 2,
Axis := Axis0,
Done=> DoneInit,
Busy=>Busy1,
Error=>Error1,
ErrorID=>ErrorID1);
```

(2) Enable the axis

```
IF DoneInit THEN
   MC_Power1(
   Enable := PowerEnable,
   Axis := Axis0,
   Status=> PowerStatus,
   Busy=>Busy2,
   Error=>Error2,
   ErrorID=>ErrorID2);
END_IF
```

(3) Send the electronic gear command to carry out electronic gear action

```
MC_GearIn1(
Execute := StartGear,
RatioNumerator := RatioNumerator,
RatioDenominator := RatioDenominator,
MasterValueSource := MasterValueSource,
Acceleration := Acceleration_Gearn1,
```

```
Deceleration := Deceleration_Gearn1,
BufferMode := Buffer_Gearn1,
Master := Axis1,
Slave := Axis0,
InGear=>InGear,
Busy=>BusyGear,
Active=>ActiveGear,
CommandAborted=>ComGear,
Error=>ErrorGear,
ErrorID=>ErrorIDGear);
Reset in case of error
MC_Reset1(
Execute := StartReset,
Axis := Axis0,
Done=>Done4,
Busy=>Busy5,
Error=>Error5,
ErrorID=>ErrorID5);
```

6.2.4 Precautions

The action of the electronic gear takes effect on the slave axis. The acceleration and deceleration in the electronic gear take effect from the acceleration or deceleration of the slave axis during catching up. If it is synchronized, the slave axis moves completely with the main shaft, and the acceleration and deceleration do not take effect.

The main shaft and slave axis cannot be set to the same shaft, and the denominator of gear ratio cannot be 0.

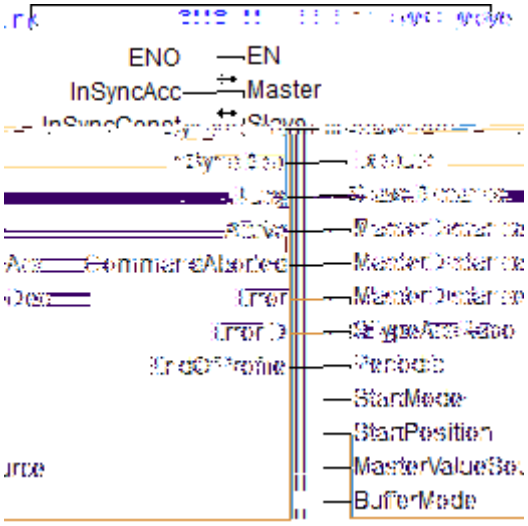
6.2.5 Reference

For the use of masters, see other single-axis motion chapters.

In case of any error, please refer to the appendix Function Block Error Code Description.

6.3 SMC_MoveLink (Flying Shear)

This command implements the pursuit process action.



6.3.1 Parameter

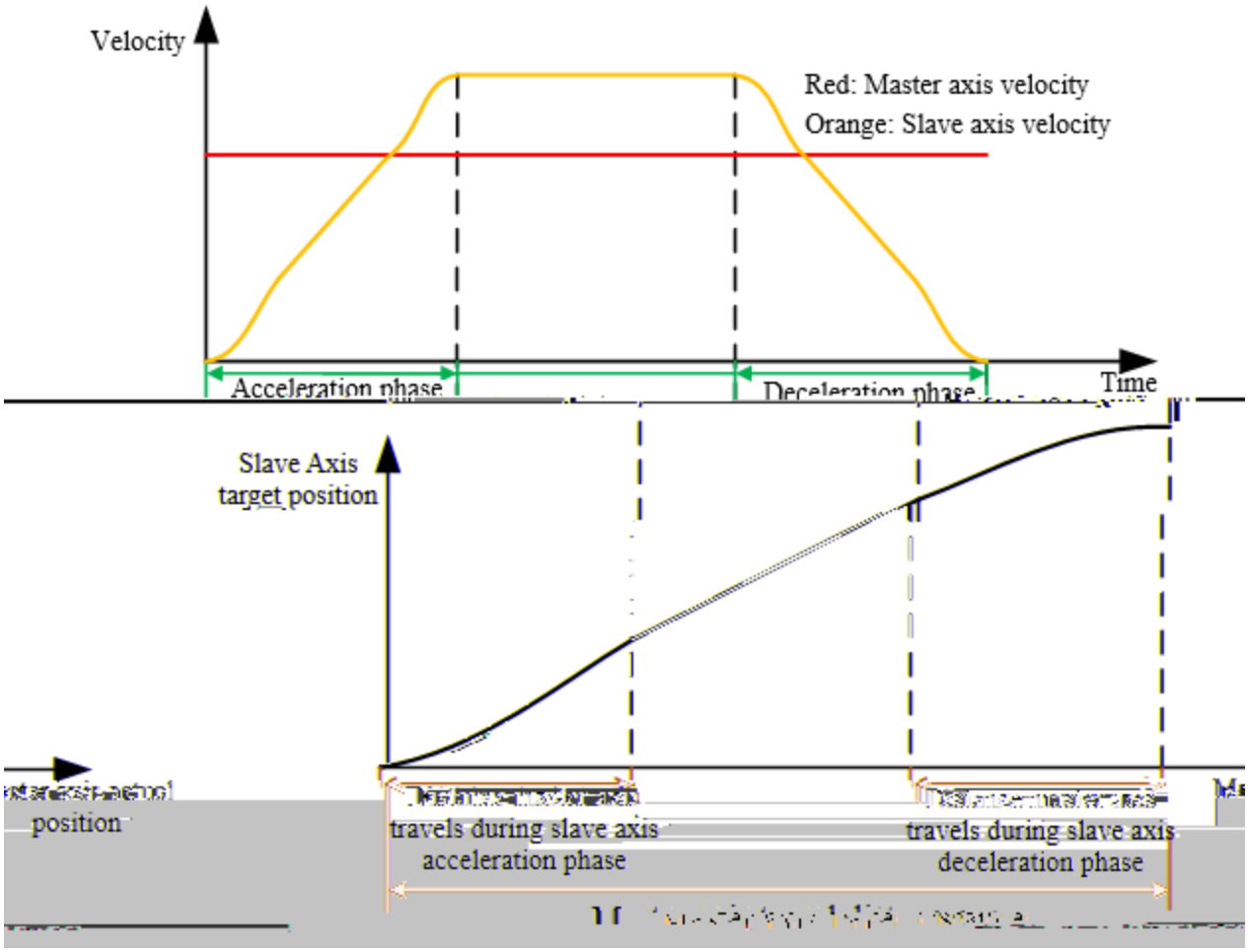
Parameter Type	Title	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	Master	Master axis name	No	-	-	AXIS_REF
	Slave	Slave axis Name	No	-	-	AXIS_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
	SlaveDistance	Slave axis motion distance	No	-	Positive/negative/0	LREAL
	MasterDistance	Master movement distance	No	-	Positive value	LREAL
	MasterDistanceAcc	Distance traveled by master in slave axis acceleration phase	No	-	Positive/0	LREAL
	MasterDistanceDec	Distance moved by the master during deceleration stage of slave axis	No	-	Positive/0	LREAL
	STypeAccRatio	S-shaped acceleration and deceleration percentage, in 1%	YES	0	0 ~ 100	LREAL
	Periodic	FALSE: Single mode				

		When DI trigger is started, it is the bit offset mapped by DI point in Area I. For example, the bit offset of %IX100.1 in area I is 100×8+1.			DI trigger start: 0 ~ (field I byte capacity * 8)-1	
	MasterValueSource	Master Position Source 1: Synchronize with Master feedback position	YES	1	1: mcActualValue	MC_SOURCE
	BufferMode	Buffer mode 0: Interrupt 1: Buffer	YES	1	0: Aborting 1: Buffered	MC_BUFFER_MODE
OUTPUT	InSyncAcc	The slave axis is in the acceleration synchronization section	YES	FALSE	TRUE/FALSE	BOOL
	InSyncConst	The slave axis is in the constant-speed synchronization section	YES	FALSE	TRUE/FALSE	BOOL
	InSyncDec	The slave is in the deceleration synchronous section	YES	FALSE	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Active	In control, TRUE when the command is executed	YES			

- Single mode. After MoveLink is started: For immediate start and DI triggered start, the command left boundary is at the position of cam start; for absolute position start, the command left boundary is at the set absolute position; command right boundary = left boundary + master stroke length. The order is cancelled when the master moves negatively across the left boundary or positively across or reaches the right boundary.
- For single shearing motion, when the master is within the effective stroke range, the master can perform unidirectional and reciprocating motion. When the master reciprocates within the effective stroke range, the slave shaft will also perform reciprocating linkage action.
- Cycle mode. When the master position crosses a master stroke length, it will enter the next flying shearing stroke. At this time, the slave axis will take the slave axis position at the end of the previous flying shearing stroke as the starting point, and make incremental displacement calculation on this basis. Repeat the linkage action of the previous flying shearing stroke in turn. To cancel the cyclic flying shear motion, execute the MC_Stop command. This will stop and cancel the flying shear operation.

The linkage motion track of the master and slave for motion track is consistent with that of the cam. For details of the motion track diagram, please refer to the linkage motion track diagram of the master and slave in the Chapter of Cam Command Description (see the introduction part of cam stroke in SMC_CamIn).

(5)

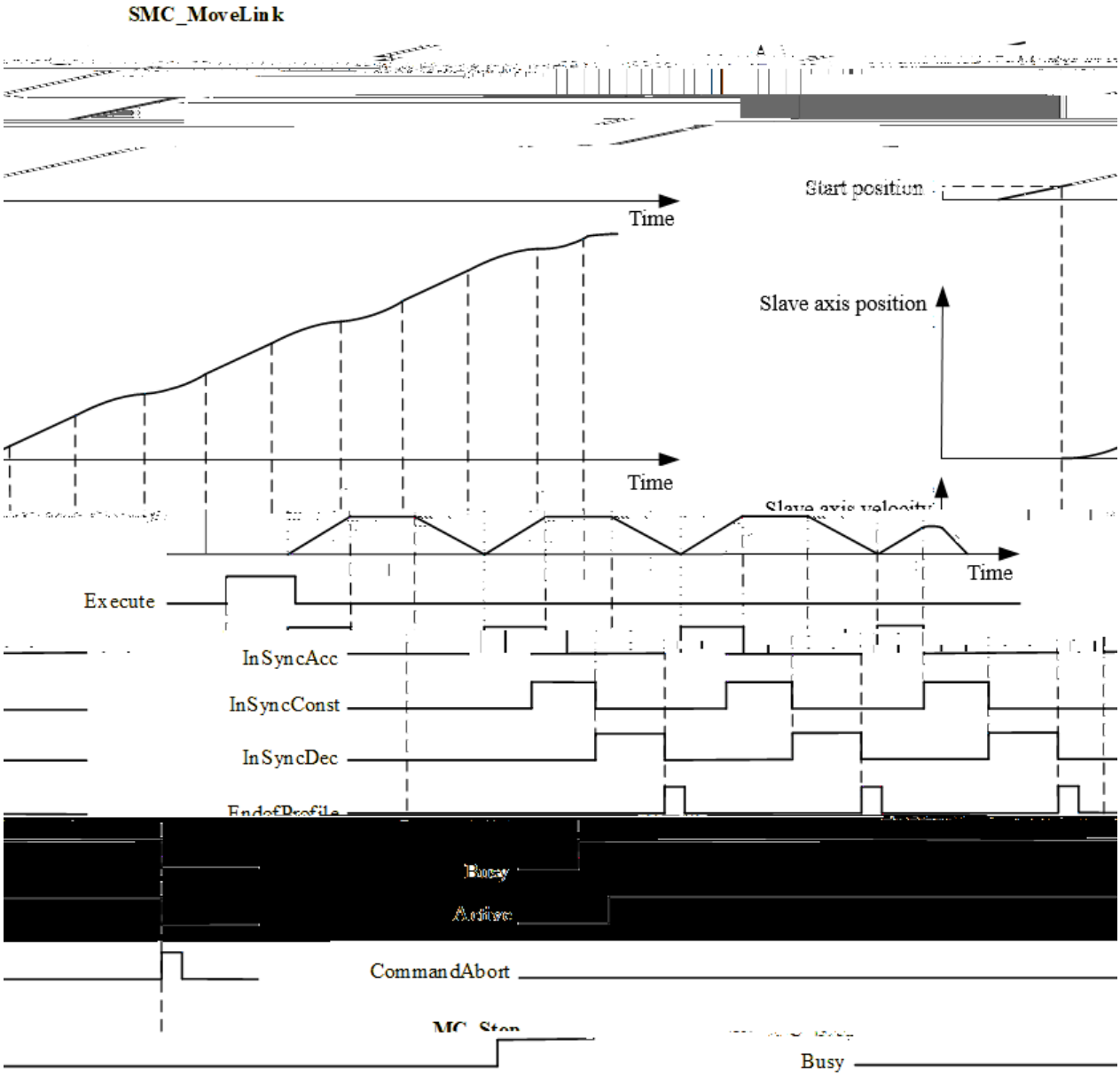

Velocity
Red: Master axis velocity
Orange: Slave axis velocity
Acceleration phase
Deceleration phase
Time
Slave Axis
target position
position
travels during slave axis
acceleration phase
travels during slave axis
deceleration phase

(2) Cyclic flying shearing

For cyclic trailing motion, there is no boundary range limit for master position. If the corresponding stop cancellation command is not executed after startup, the trailing function will be always valid.

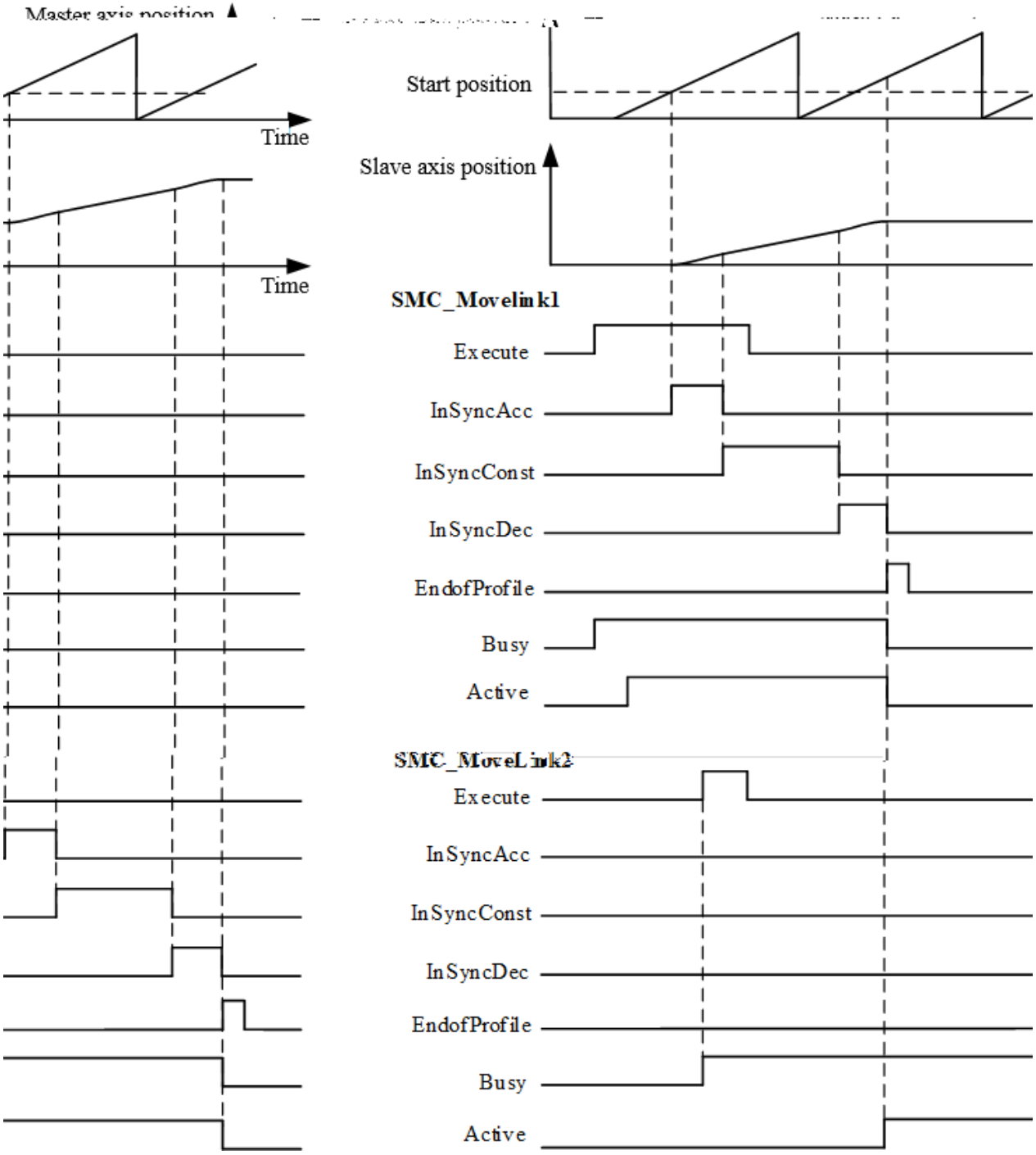
The following figure shows the timing of cut command pins in cycle mode and fixed position startup mode. The slave axis performs corresponding linkage actions with the master. When the master finishes a flying shearing stroke, it enters the next stroke. Incremental calculation is performed from the end point of the previous flying shearing stroke as the starting point to execute periodic linkage actions. At the end of each clipping pass, the output pin EndofProfile is set high for one cycle.

When the MC_Stop command is used to cancel the abort command of cyclic flying shearing, the flying shearing command output pin CommandAborted is set as TRUE and other output pins are reset.



BufferMode (buffer mode selection)

Specify the connection mode between the previous axis action and this action.



EndofProfile (End of periodic flying chasing)

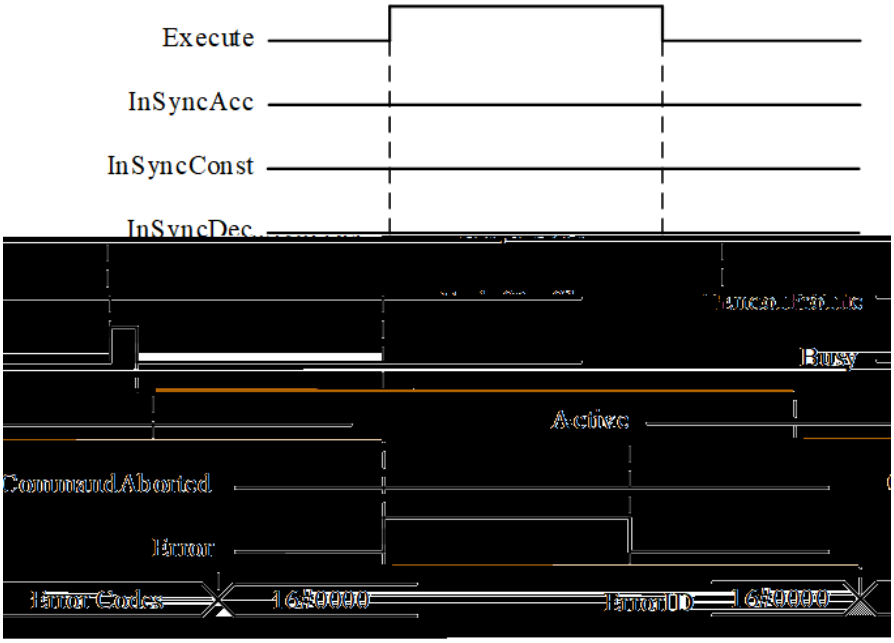
When the chase command is in single mode, the command cycle is completed, and the command output pin EndofProfile is set to TRUE and held until the input pin Execute becomes FALSE.

When the cut command is in periodic mode, the first cut stroke is completed, and the command output pin EndofProfile is set to TRUE for only one cycle. The command enters the next cut stroke,

Error

When starting the flying shearing motion command, if an error occurs, the Error pin is set to True, and the error code ErrorID reports an error value. Refer to the appendix Description of Error Codes to obtain the cause of the error. When the command input pin Execute goes low, the output pin returns to its Default Value.

SMC_MoveLink abnormal situation sequence diagram



6.3.3 Examples

In this example, MasterAxis is defined as the master axis, SlaveAxis as the slave axis, and the variable type is AXIS_REF.

The MasterAxis executes the MC_MoveVelocity command, and the SlaveAxis executes the MC_MoveLink command. Use the MC_Stop command to stop revoking the posthumous motion command.

LD program implementation

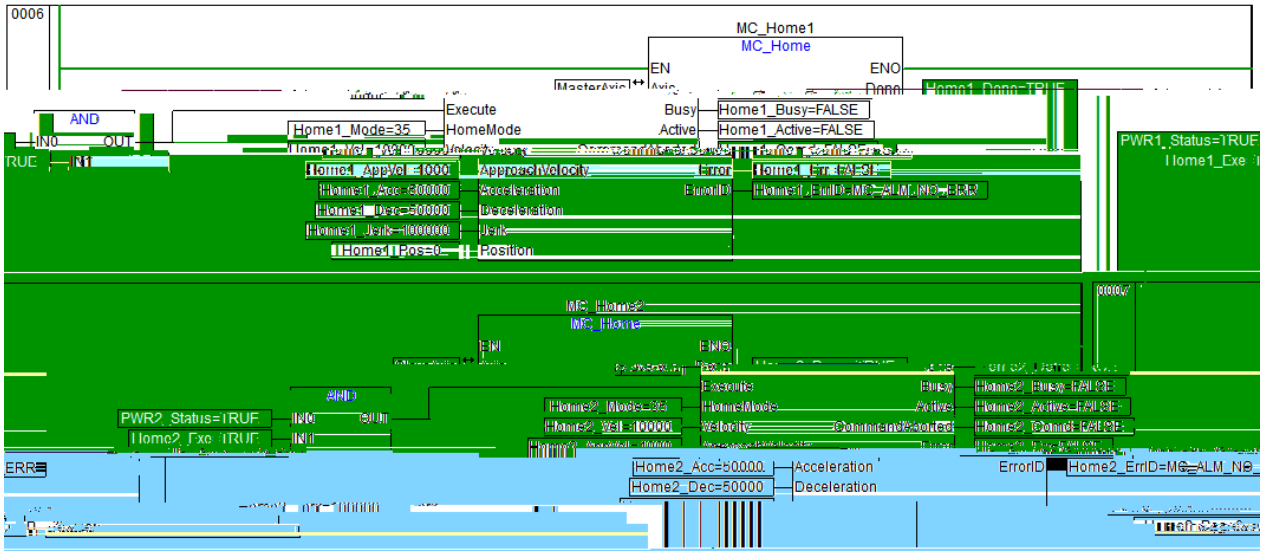
This example sets up a periodic mode, immediately starting the flying shearing command.

Example Program Primary Variables

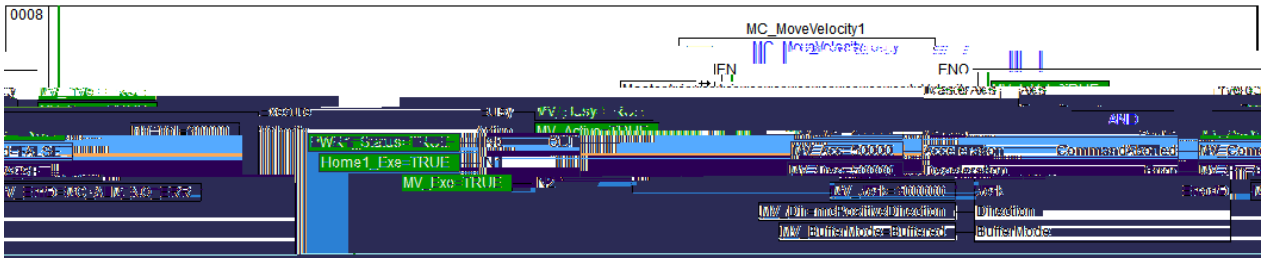
Variable Name	Variable Type	Initial Value	Notes
MasterAxis	AXIS_REF	-	Axis variables of the master axis
SlaveAxis	AXIS_REF	-	Axis variables of the slave axis
Power1_Status	BOOL	FALSE	Variable for successful master enable. Thi7 (e.)2 (T)15 (hi)12.7 (7 24 (i)-0.716 1

Switch1	BOOL	FALSE	The state variable of axis readiness. Switch to TRUE, indicating that the axis is ready.
MV_Exe	BOOL	FALSE	The variable triggered by the rising edge of the speed control command for an axis.
MV_InVel	BOOL	FALSE	Identification variable for the axis speed control command to reach the target speed. This variable becomes TRUE and the speed commanded reaches the target speed.
ML_SlaveDis	LREAL	-	Slave axis motion distance for delivering a flying shearing command from an axis.
ML_MasterDis	LREAL	-	Master axis motion distance for delivering a flying shearing command from an axis.
ML_DisAcc	LREAL	-	The master axis motion distance during the slave axis acceleration phase when issuing a flying shear command to the slave axis.
ML_DisDec	LREAL	-	The master axis motion distance during the slave axis deceleration phase when issuing a flying shear command to the slave axis.

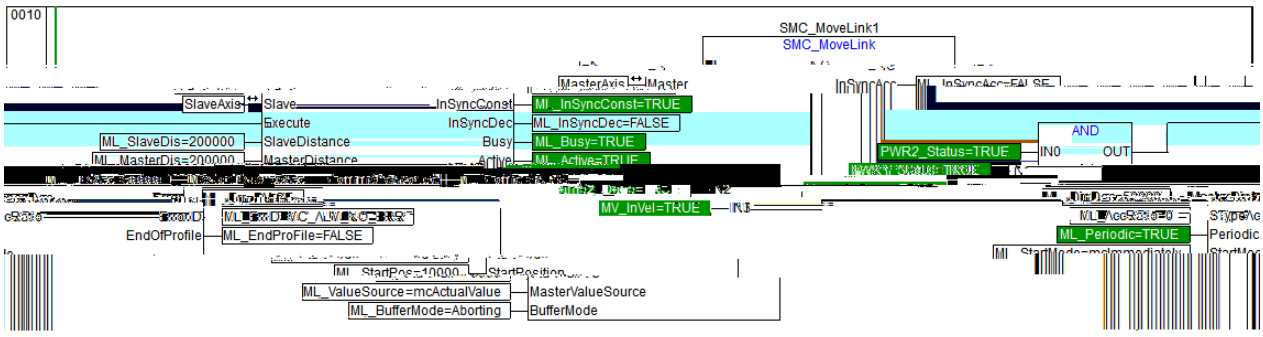
(1) Axis initialization. Set the AxisID of MasterAxis to 0, the AxisID of SlaveAxis to 1, and the axis 2.3.7 (x)-8 (i13.3 (p)-1)-12.2 (.2 (i)2 5195)r s 2.3 3.1 (a)-eo 1,tae Ax w -0.241 -1.446 Td [m116.7 u3.1 (



(4) See Section 0008. After the homing is completed, execute MC_MoveVelocity command. The master starts to move.

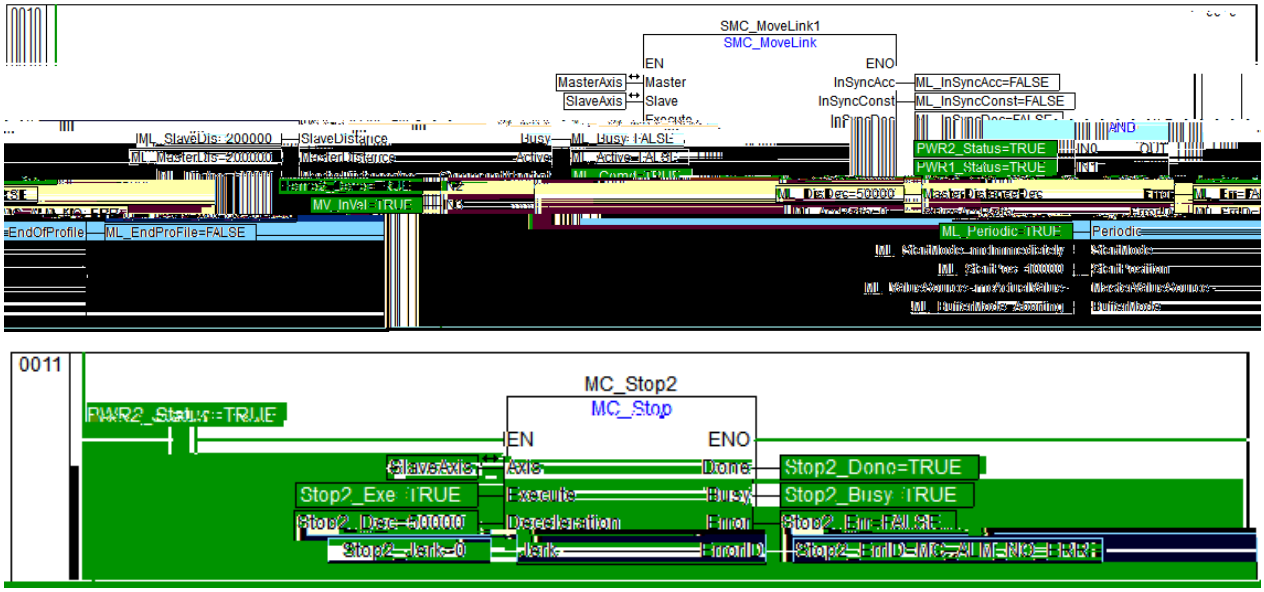


(5) See Section 0009. When the MV_InVel pin of master speed command MC_MoveVelocity becomes TRUE, start the SMC_MoveLink1 command of the slave axis. The command is to start the cycle immediately. The ML_InSyncAcc pin is TRUE, and the slave axis enters the acceleration synchronization section of flying shearing motion (ML_InSyncAcc=TRUE). The slave axis starts moving following the master, and then enters the uniform synchronization section of flying shearing motion (ML_InSyncConst =TRUE) and deceleration synchronization section of flying shearing motion (ML_InSyncDec =TRUE) in turn.

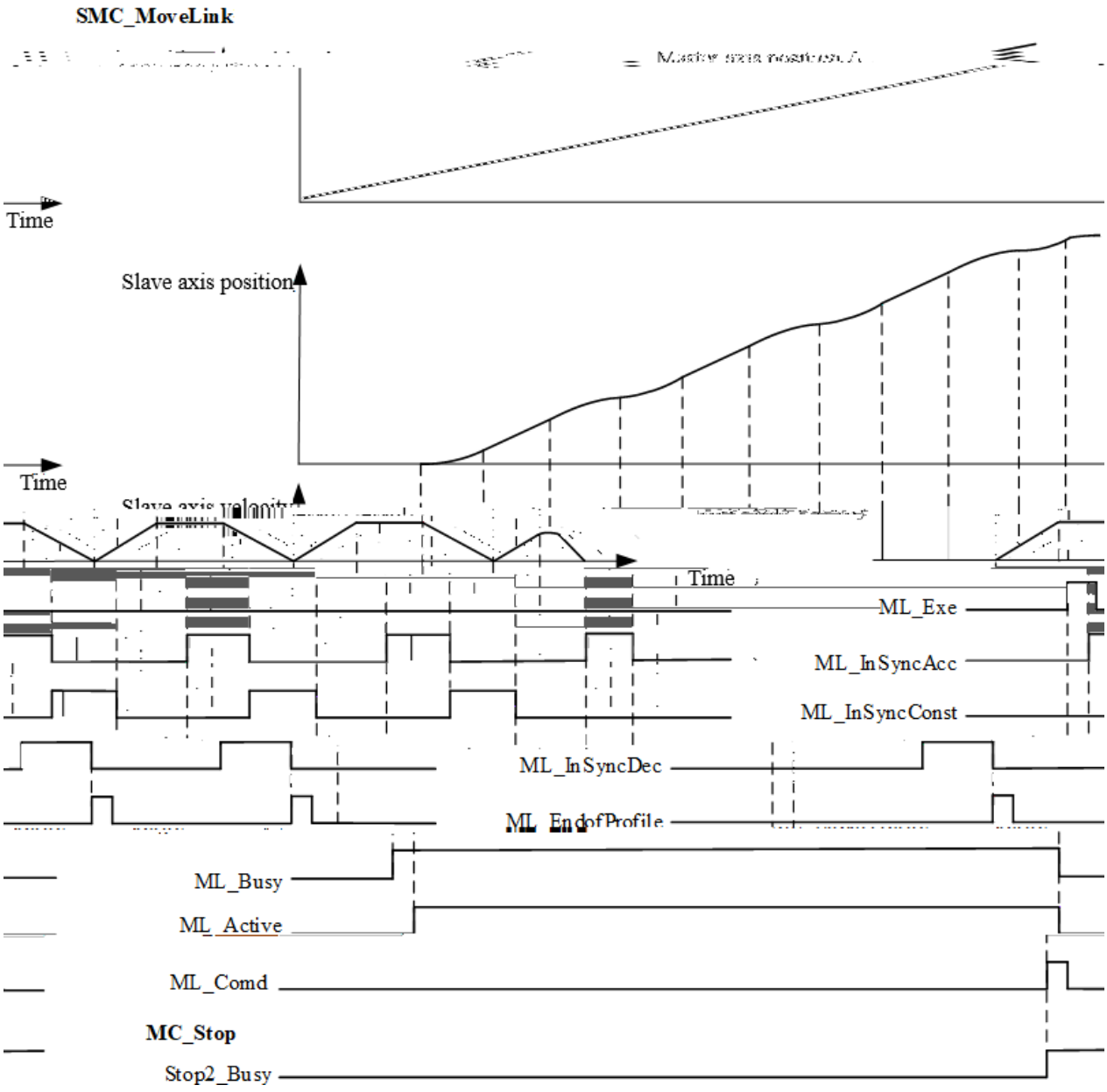


(6) For a cyclic flying shearing command, you can use MC_Stop to cancel and abort the flying shearing motion command of the slave axis. In the following section 0011, the MC_Stop2 is executed to abort the flying shearing command of the slave axis, and the ML_Comd of the trace

command becomes TRUE.



The timing diagram of this example is as follows:



ST language program.

Command is set to single mode, absolute position start and the fixed position of cam is at 100. The master position crosses the cam start position 100 in the negative direction, and the cam action begins.

Primary Variables

			axis is enabled successfully.
			command for an axis.
Switch1	BOOL	FALSE	The state variable of axis readiness. Switch to TRUE, indicating that the axis is ready.
InitFlag	BOOL	FALSE	If this variable is TRUE, the initialization parameter setting is completed.
ML_SlaveDis	LREAL	-	Slave axis motion distance for delivering a flying shearing command from an axis.
ML_MasterDis	LREAL	-	Master axis motion distance for delivering a flying shearing command from an axis.
ML_DisAcc	LREAL	-	The master axis motion distance during the slave axis acceleration phase when issuing a flying shear command to the slave axis.
ML_DisDec	LREAL	-	The master axis motion distance during the slave axis deceleration phase when issuing a flying shear command to the slave axis.

(1) Axis initialization

Set the AxisID of MasterAxis to 0 and that of SlaveAxis to 1. Call SMC_AxisInit to complete the initialization configuration of master and slave axes. The type of both master and slave axes is 50. Call SMC_AxisInit to complete the initialization configuration of master and slave axes. For the usage of SMC_AxisInit, please refer to its introduction.

```
IF FALSE = InitFlag THEN
  ML_SlaveDis:=200;
  ML_MasterDis:=200;
  ML_DisAcc:= 10;
  ML_ DisDec:=10;
  ML_Periodic:=FALSE;
  ML_STartMode:= mcAb5lt;
```

```
   Busy=> PWR2_Busy,
   Error=> PWR2_Err
   ErrorID=> PWR2_ErrID);
END_IF
```

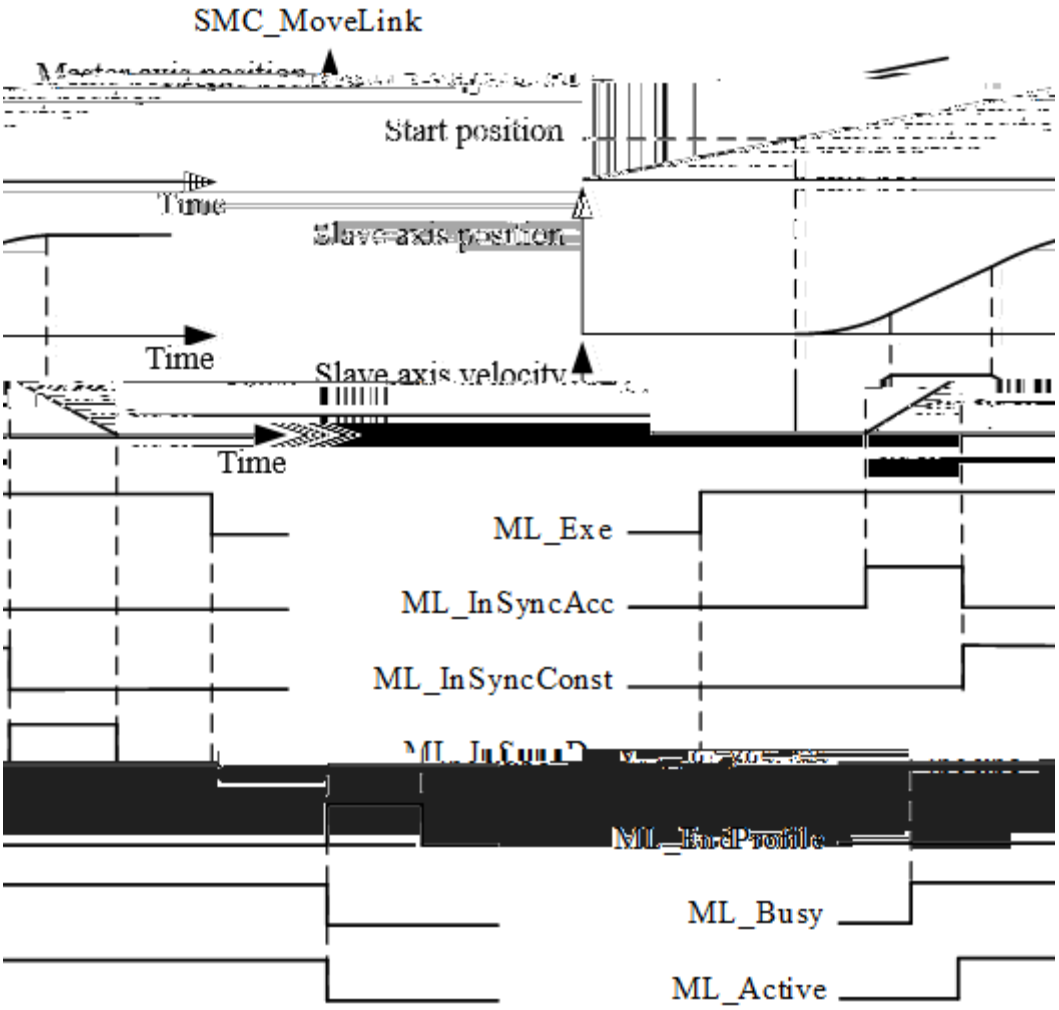
(4) Execute the homing command of master and slave axes to determine the home. For the usage of MC_Home command, see its introduction chapter.

```
IF  PWR1_Status THEN
 MC_Home1(
 Execute := Home1_Exe,
 HomeMode := Home1_Mode,
 Velocity := Home1_Vel,
 ApproachVelocity := Home1_AppVel,
 Acceleration := Home1_Acc,
 Deceleration := Home1_Dec,
 Jerk := Home1_Jerk,
 Position := Home1_Pos,
 Axis := MasterAxis,
 Done=> Home1_Done,
 Busy=> Home1_Busy,
 Active=> Home1_Active,
 CommandAborted=> Home1_Comd,
 Error=> Home1_Err,
 ErrorID=> Home1_ErrID);
 MC_Home2(
 Execute := Home2_Exe,
 HomeMode := Home2_Mode,
 Velocity := Home2_Vel,
 ApproachVelocity := Home2_AppVel,
 Acceleration := Home2_Acc,
 Deceleration := Home2_Dec,
 Jerk := Home2_Jerk,
 Position := Home2_Pos,
 Axis := SlaveAxis,
 Done=> Home2_Done,
 Busy=> Home2_Busy,
 Active=> Home2_Active,
 CommandAborted=> Home2_Comd,
 Error=> Home2_Err,
 ErrorID=> Home2_ErrID);
END_IF
```

(5) The flying shearing command is sent from the axis, and a single fixed position starts. When the master moves from the negative direction of the fixed position to the positive direction, the flying shearing action starts.

```
IF  PWR1_Status AND PWR2_Status AND Home1_Done THEN
 SMC_MoveLink1(
```

```
Execute := ML_Exe,
SlaveDistance := ML_SlaveDis,
```



6.3.4 Precautions

The motion distance of the master needs to be positive.

For a single trailing motion, the slave axis follows the master movement within the travel boundaries and is revoked after the master has crossed the left boundary in negative direction or the right boundary in positive direction.

When flying shearing is started at absolute position, the master passes through the absolute position shr-12.3 (w12.3 (eard3.1 (on7 (w)9.2 (i)3.2 (r))]TJ [(c)-8 (t)-1.1 (i)-8s)-8 (t)-1)0.6 (t)-1.1eariggsher m p m on.

v

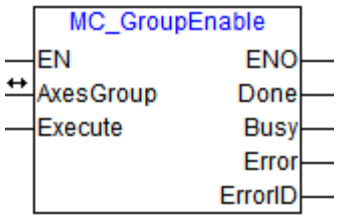
When the slave axis exceeds the soft limit during flying shearing, the command is interrupted and the slave axis aborts running immediately.

When the counter mode of the master is rotation mode, please specify MasterDistance (master movement distance) as the value within 1 revolution of the ring counter of the master.

Chapter 7 Axis Group Motion Instruction

7.1 MC_GroupEnable (Enable Axes Group)

Axis group enable.



7.1.1 Parameter

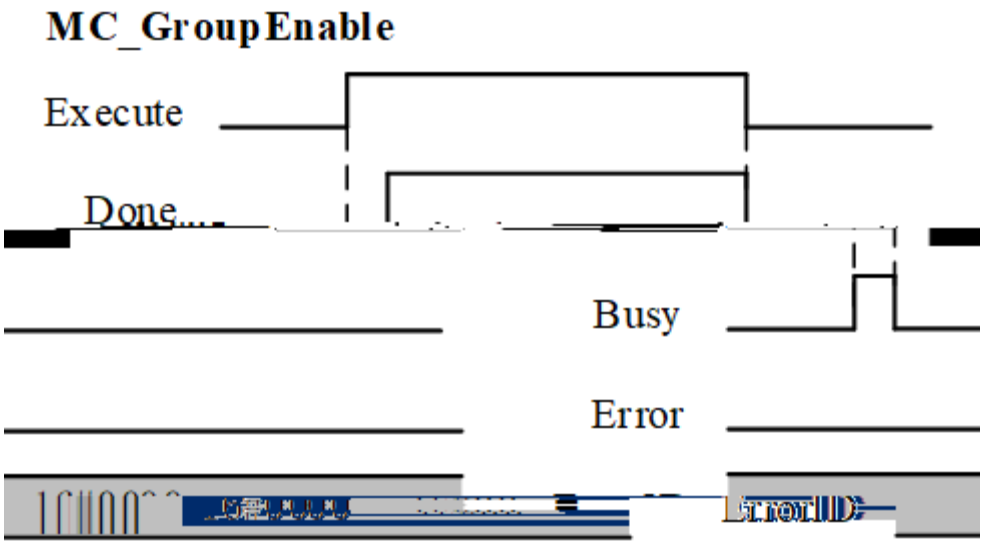
Pin Parameters

Pin Parameter Description

Parameter Type	Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	AxesGroup	Axis group name	No	-	-	AXES_GROUP_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
OUTPUT	Done	Completed	YES	FALSE	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR

Structural parameters of axis group

Currently, only two-



(2) Error pin timing diagram

When this command is executed, there are abnormal conditions such as illegal axis ID and illegal axis type in the axis group, and this command fails to be called. At this time, the command output pin Error=True, and the error code ErrorID reports an abnormal error value. Please refer to the appendix Description of error codes to understand the cause of the error. When the command input pin Execute is low, all output pins are restored to their Default Values. aC 20 Tw 365.1998602 0 0 45 (aC

Description of Primary Variables

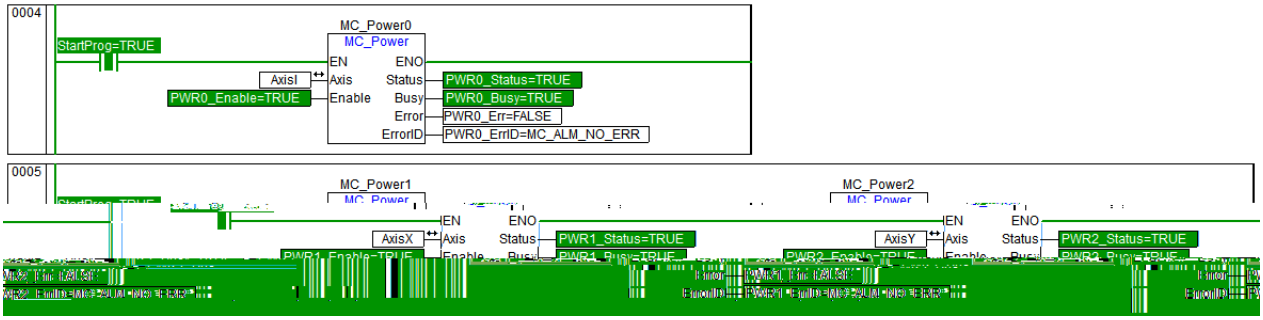
Variable Name	Variable Type	Initia l Value	Notes
AxesGroup	AXES_GROUP_REF	-	Axis group name
AxisI	AXIS_REF	-	Axis variables of combined axis in axis group
AxisX	AXIS_REF	-	Axis variables of sub-axis X in axis group
AxisY	AXIS_REF	-	Axis variables of sub-axis Y in axis group
StartProg	BOOL	FALSE	If this variable is TRUE, the axes in the axis group are ready to perform the axis enabling operation.
Power0_Status	BOOL	FALSE	Variable for successful combination of axes enable. This variable becomes TRUE when the combined axis is enabled successfully.
Power1_Status	BOOL	FALSE	Variables whose sub-axis-axis X enable is successful. When the sub-axis X is enabled successfully, this variable becomes TRUE.
Power2_Status	BOOL	FALSE	Variables whose sub-axisY enable is successful. When the sub-axis Y is enabled successfully, this variable becomes TRUE.
GrpEnable1_Exe	BOOL	FALSE	Axis group enable rising edge trigger variable.

(1) Set the IDs of joint and sub-axes in the axis group. When defining an axis group, you need to define three axes in the axis group first, and match the IDs of the defined axis numbers of the three axes with those of the combined and sub-axes in the axis group. See sections 0001 and 0002 below.

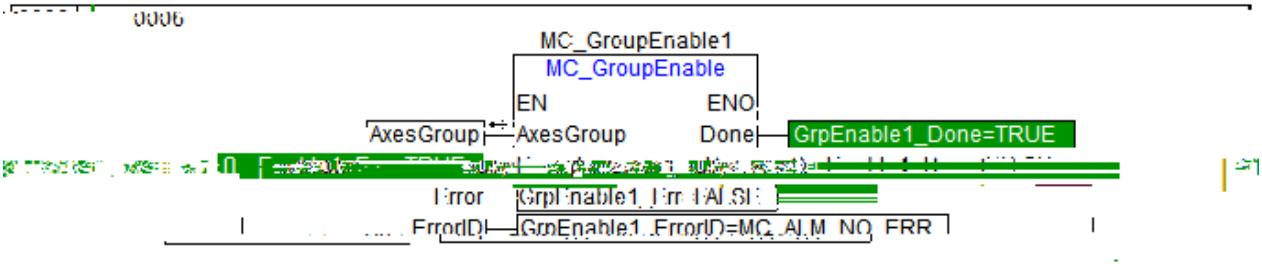
(2) Axis initialization. Call SMC_AxisInit command to complete the axis initialization configuration of three axes in the axis group respectively, and set the combined axis as 0: Virtual and the sub-axis axis type as 50: EtherCAT_Position. For the usage of SMC_AxisInit command, please refer to their respective commands.

Þ Note: If the axis type for a combined axis is not explicitly set and remains as the default 'unused' state, the MC_GroupEnable command will automatically configure the combined axis as a virtual axis and enable it. In this example, the combined axis is defined as a virtual axis, and the MC_Power command must be invoked to complete the combined axis enable.

(3) Axis enable. After the initialization of each axis in the axis group is completed and the axis ready state StartProg is TRUE, execute the axis enable command to complete the axis enable. See Sections 0004 to 0005.



(4) Axis group enable. Call MC_GroupEnable to enable the axis group. If AxesGroupDisabled in the axis group variables is FALSE, the axis group is enabled successfully. See Section 0006.



ST language construction program

Description of Primary Variables

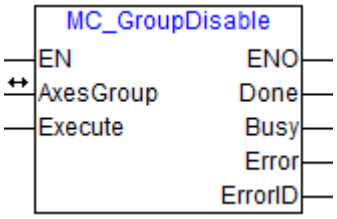
Variable Name	Variable Type	

(2) Call SMC_AxisInit command to complete the axis initialization configuration of three axes in the axis group. Set the axis type of axis combination to 0: Virtual; set the axis types of axis group AxisX/AxisY to 50: EtherCAT_Position. For the usage of SMC_AxisInit command, please refer

After the axis group is enabled, it is forbidden to manually switch the axis type to an axis type not supported by the axis group.

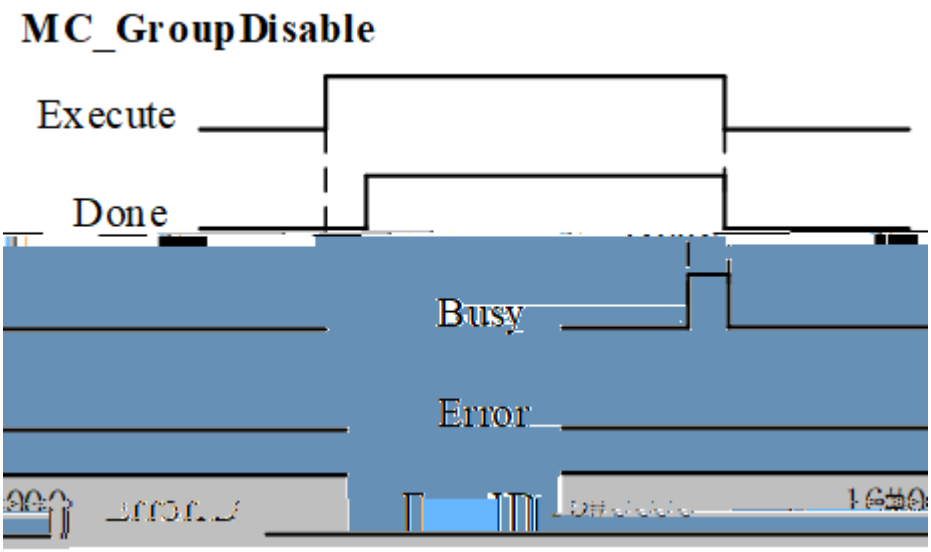
7.2 MC_GroupDisable (Disable Axes Group)

Disable the axis group function.



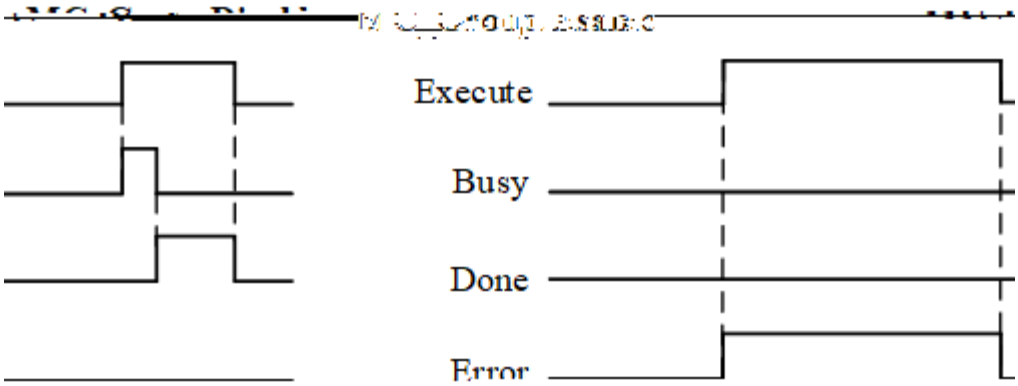
7.2.1 Parameter

Parameter Type	Title	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	AxesGroup	Axis group name	No	-	-	AXES_GROUP_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
OUTPUT	Done	Completed	YES	FALSE	TRUE/FALSE	BOOL
	Busy	Busy flag				



(2) Error pin timing diagram

When this command is executed, there are abnormal conditions such as illegal axis ID and illegal axis type in the axis group, and this command fails to be called. At this time, the command output pin Error=True, and the error code ErrorID reports an abnormal error value. Please refer to the appendix Description of Error Codes to understand the cause of the error. When the command input pin Execute is low, all output pins are restored to their Default Values.



þ After the abnormal condition of the command is solved, re-execute the MC_GroupDisable function block to disable the axis group enable.

7.2.3 Examples

Variable

to its comand section.

(3) After the initialization of each axis in the axis group is completed and the ready state StartPro of the axis is TRUE, execute the axis enable command to complete the axis enable.

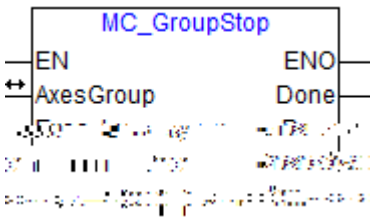
```
IF StartPro THEN
MC_Power0(*Combined axis enable*)
   Enable := PWR0_Enable,
   Axis := AxisI,
   Status=> PWR0_Status,
   Busy=> PWR0_Busy,
   Error=> PWR0_Err,
   ErrorID=> PWR0_ErrID);
MC_Power1(*Sub-axis AxisX Enable*)
   Enable := PWR1_Enable,
   Axis := AxisX,
   Status=> PWR1_Status,
   Busy=> PWR1_Busy,
   Error=> PWR1_Err,
   ErrorID=> PWR1_ErrID);
MC_Power2 (*Sub-axis AxisY Enable*)
   Enable := PWR2_Enable,
   Axis := AxisY,
   Status=> PWR2_Status,
   Busy=> PWR2_Busy,
   Error=> PWR2_Err,
   ErrorID=> PWR2_ErrID);
END_IF
```

(4) Execute axis group enable

```
MC_GroupEnable1(
Execute := GrpEnable_Exe,
AxesGroup := AxesGroup,
```

7.3 MC_GroupStop (AxesGroup Motion Stop)

Control the axis group to decelerate and stop, suspend any motion control command being executed on the axis group, and cancel all motion buffer command on the combined axis of axis group.



7.3.1 Parameter

Parameter Type	Name	Description	Empty or Not	Default Value	Scope	Data Type
IN_OUT	AxesGroup	Axis group name	No	-	-	AXES_GROUP_REF
	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
	Deceleration	Deceleration	No	-	Positive value	LREAL

INPUT

calling MC_GroupStop, only the axis group are in Stopping status.

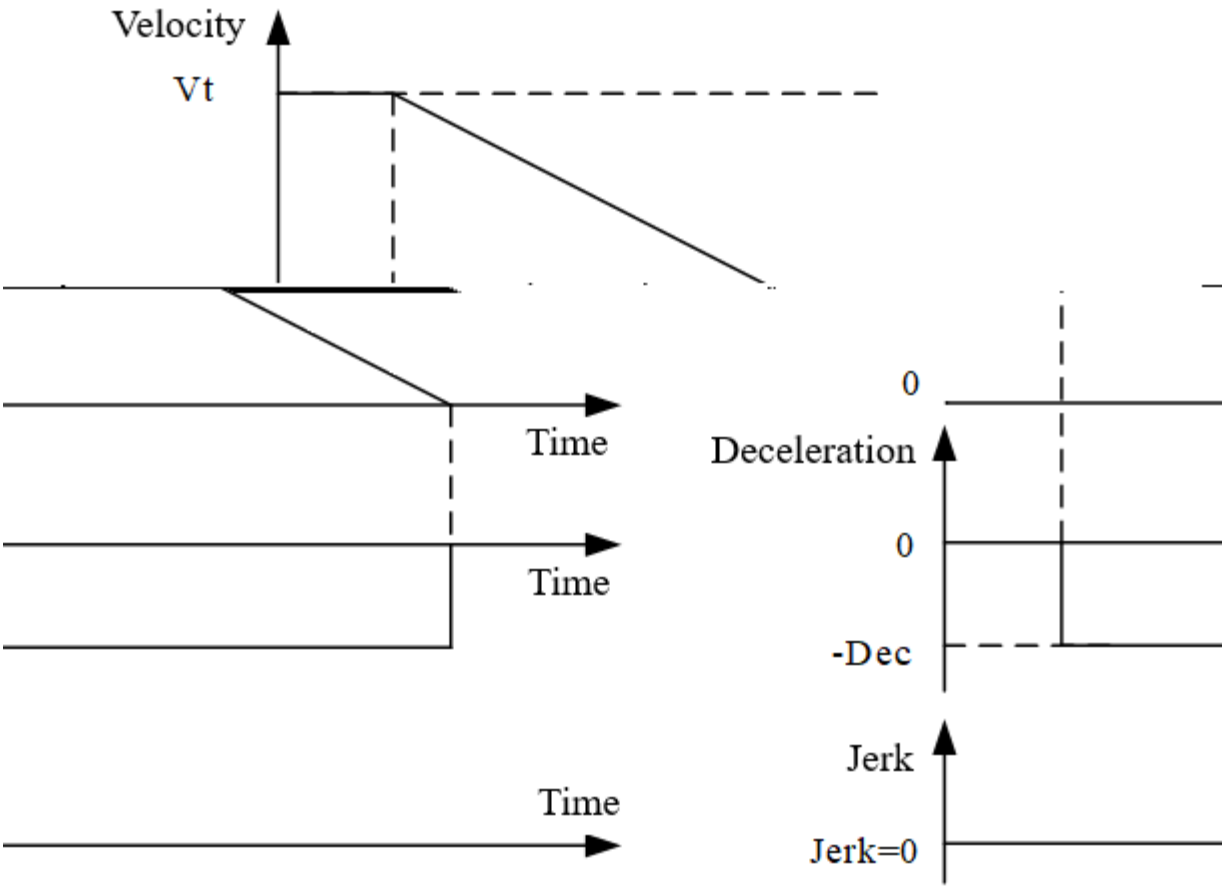
(3) Input parameter latch

When the rising edge of this command is triggered, each axis ID, input parameter Deceleration and Jerk value of the input and output parameter axis group will be latched. During the execution of the command, the modification of latched parameters will be invalid until this command is repeatedly triggered. In addition, after the AxesGroup is enabled, it is forbidden to modify parameters such as combined axis ID, sub-axis ID and axis type of AxesGroup.

(4) Acceleration and deceleration curve

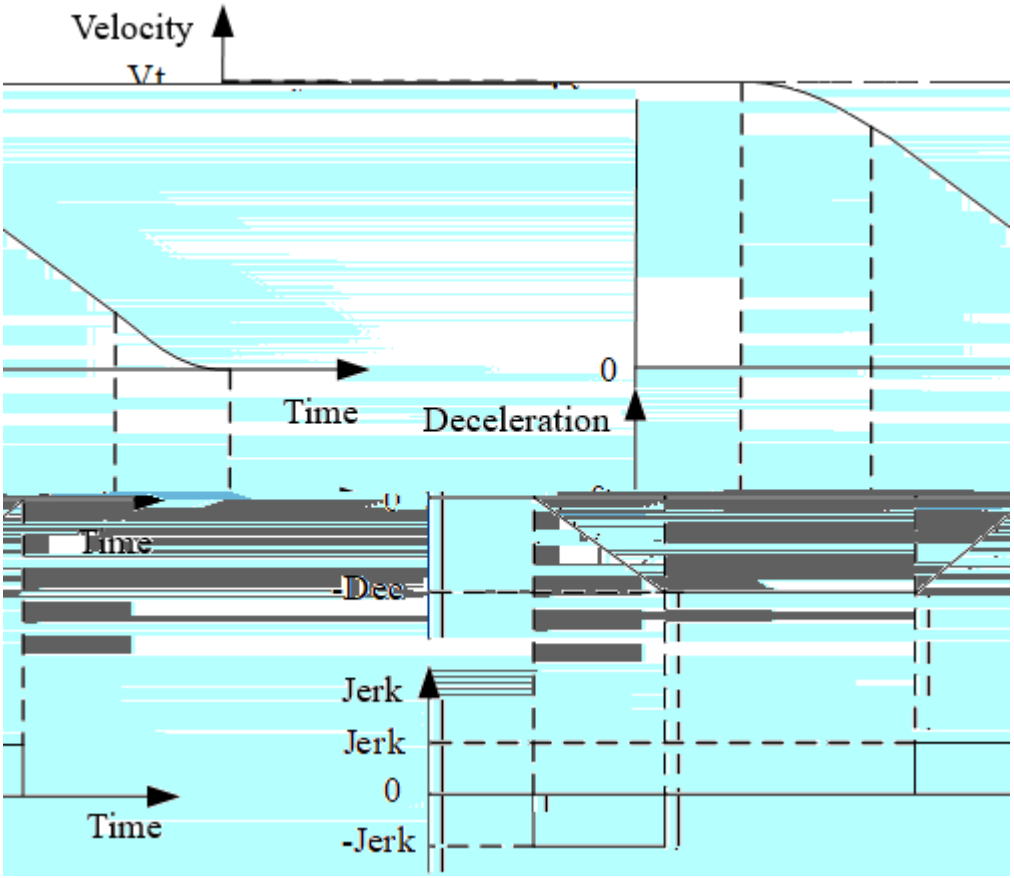
If the input parameter Jerk=0, the deceleration curve is a trapezoidal acceleration/deceleration curve; if the input parameter Jerk>0, the deceleration curve is an S-shaped acceleration/deceleration curve.

Ž When Jerk=0, the motion deceleration curve of axis group is a trapezoidal acceleration and deceleration curve as follows:



Vt: speed when deceleration starts, Dec: set value of deceleration, Jerk: set value of jerk

Ž When Jerk>0, the command value of speed is generated by deceleration. The axis group motion deceleration curve is an S-shaped acceleration and deceleration curve as follows:



Vt: speed when deceleration starts, Dec: set value of deceleration, Jerk: set value of jerk

When the MC_GroupStop command is executed, it is possible to decelerate and stop the SMC_MoveLinearAbsolute (linear interpolation of absolute value of 2 axes), SMC_MoveLinearRelative (linear interpolation of relative value of 2 axes) and SMC_MoveCircular2D (circular interpolation of 2 axes) commands with the set deceleration parameters.

If the MC_GroupStop command is executed during the execution of the interpolation command, the axis group will decelerate and stop on the track of linear or circular interpolation. The interpolation command in the axis group is interrupted.

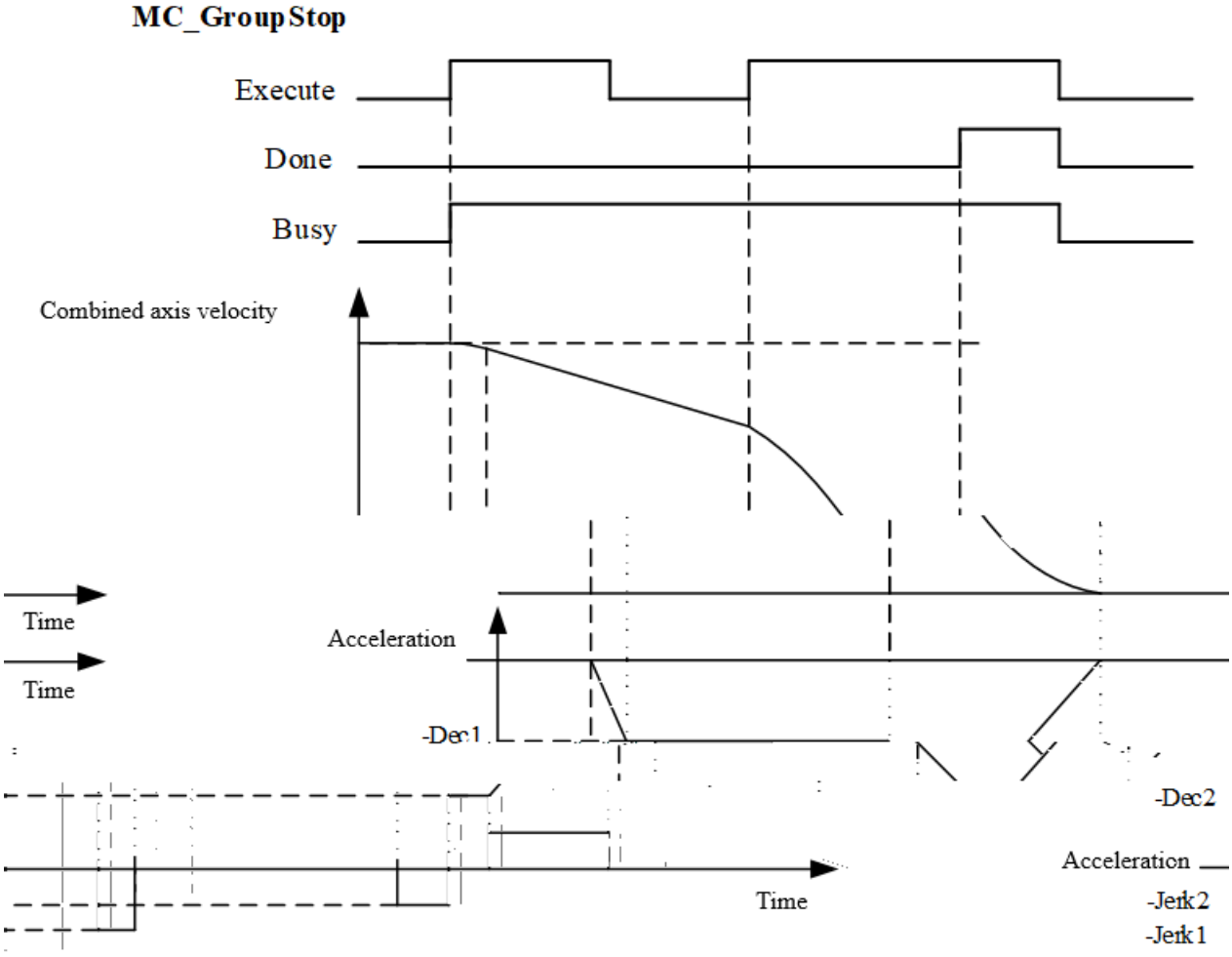
Timing diagram

(1) Normal timing diagram

The rising edge triggering of this command is valid. Execute the rising edge of the input, if there is no error, the Busy pin is set to TRUE. When the command is executed, the axis group is in Stopping status, and the motion of the axis group decelerates and stops. After the deceleration of axis group motion is completed, set "Done" to TRUE. If Execute remains at high level, the Busy and Done pins remain TRUE, and the axis group is continuously in Stopping status; if Execute is set at low level, the Busy and Done pins are set to FALSE, and the combined axis is switched to StandStill status.

(2) Repeated delivery command pin timing diagram

When multiple MC_GroupStop functional blocks are triggered repeatedly, the deceleration (deceleration) of the last delivery command is used for deceleration and stop control, and the command changes the deceleration and jerk.



Dec1: deceleration commanded at first delivery, Dec2: deceleration at seconec1:ec1:lc2:ic2:v2.3 (ec-)5.8t41EMC

þ When the error command is solved, it is necessary to trigger Execute again on the rising edge and start the MC_GroupStop function block to execute the stop action of the control axis group according to the specified deceleration.

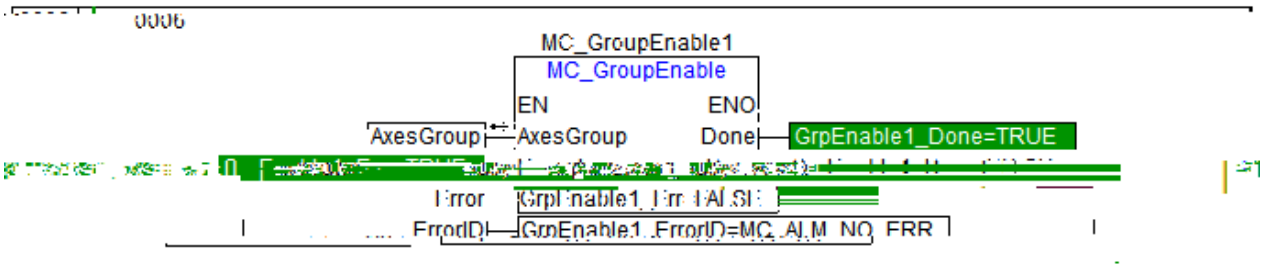
7.3.3 Examples

In this example, the MC_GroupStop command is used to stop the linear interpolation motion command for relative position of 2 axes, and the MC_ReadStatus command is used to read the axis status of the axis group.

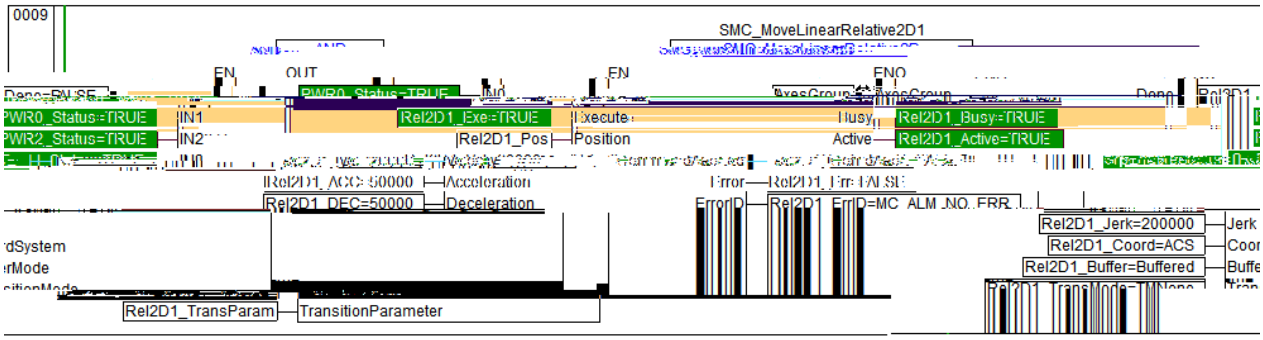
LD diagram program

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
AxesGroup	AXES_GROUP_REF	-	Axis group name
AxisI	AXIS_REF	-	Axis variables of combined axis in axis group
AxisX	AXIS_REF	-	Axis variables of sub-axis X in axis group
AxisY	AXIS_REF	-	Axis variables of sub-axis Y in axis group
StartProg	BOOL	FALSE	If this variable is TRUE, the axes in the axis group are ready to perform the axis enabling operation.
Power0_Status	BOOL	FALSE	Variable for successful combination of axes enabling.(AL)-4 (SE)]TJ

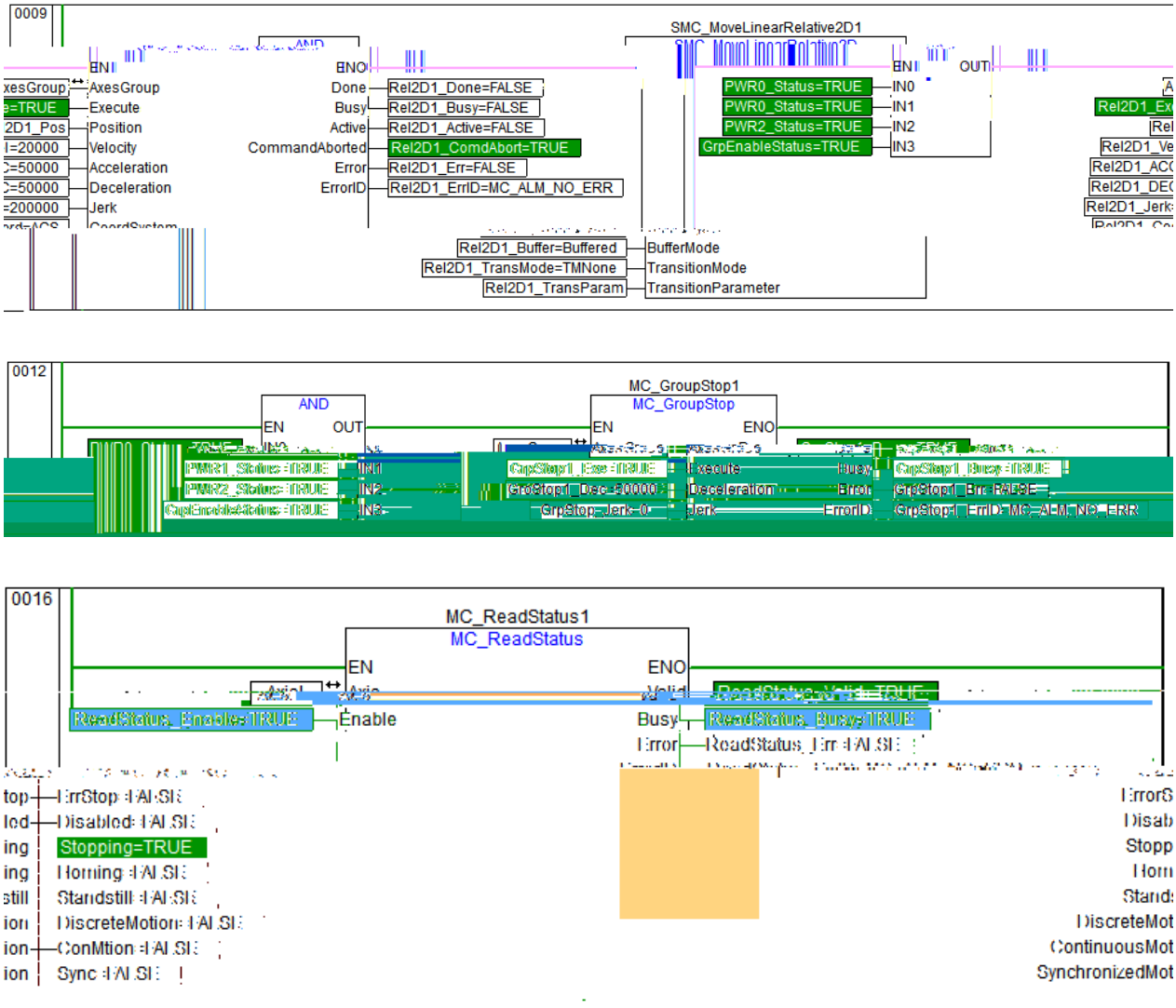


(5) In Section 0009, after each axis in the axis group and the axis group are enabled, trigger the relative position interpolation motion command to control the action of the axis group. For details about how to use the SMC_MoveLinearRelative2D command, see its description document.



(6) Section 0012 calls MC_GroupStop command to stop the interpolation motion of axis group, with deceleration set at 50000 and Jerk at 0. The trapezoidal curve is used for deceleration.

The output pin CommandAborted of the interpolation motion command SMC_MoveLinearRelative2D1 is set to TRUE when the MC_GroupStop command controls the deceleration stop of the axis group, and the interpolation command is interrupted as described in Section 0009 below. At this point, look at the output pin of MC_ReadStatus1 in section 0016 to determine the status of the combined axis of axis group.



ST language

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
AxesGroup	AXES_GROUP_REF	-	Axis group name
AxisI	AXIS_REF	-	Axis variables of combined axis in axis group
AxisX	AXIS_REF	-	Axis variables of sub-axis X in axis group
AxisY	AXIS_REF	-	Axis variables of sub-axis Y in axis group
StartProg	BOOL	FALSE	If this variable is TRUE, the axes in the axis group are ready to perform the axis enable.
InitFlag	BOOL	FALSE	If the variable is TRUE, initialization is complete.
Power0_Status	BOOL	FALSE	Variable for successful combination of axes enabling. This variable becomes TRUE when the combined axis is enabled successfully.

Power1_Status

```
END_IF
```

(4) Execute axis group enable

```
MC_GroupEnable1(
Execute := GrpEnable_Exe,
AxesGroup := AxesGroup,
Done=> GrpEnable_Done,
cutblone20ouprutonuta 1erisbleee 1p8ie grs 1ee20l23. (es)-7.8 ( gr)-6.3 (e)-12. (ou))]T.1.001 TJ 2615 1iutrpblutblcp22 1
```

7.3.4 Precautions

When MC_GroupStop is called, the axis group needs to be enabled.

When MC_GroupStop is called, after the axis group motion is successfully stopped, if the input pin Execute of MC_GroupStop is at high level, the combined axis of axis group will remain in Stopping state. At this time, the interpolation command fails to be sent. When the input pin Execute is low level, the combined axis switches to StandStill status and can normally send interpolation commands.

7.4 SMC_MoveLinearAbsolute2D (Absolute Linear Interpolation on Two Axes)

This command realizes the linear interpolation of 2 axes absolute position.

	Data type
	AXES_GROUP_REF
	BOOL
tive/0	ARRAY[0..1] OF LREAL -
	LREAL
	LREAL

[redacted]

	TransitionParameter	Retained parameters	YES	0	-	ARRAY[0..1] OF LREAL
	Done	Interpolation motion finished	YES	FALSE	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Active					

OUTPUT

(2) Error pin timing diagram

During the execution of this command, if there are errors such as illegal parameters, 0: Unused (unused axis) for an axis in the axis group, and illegal ID of each axis in the axis group, the call of this command fails, with command output pin Error=True, and the error code ErrorID reports an error value. Please refer to the appendix Error Code Description to obtain the cause of the error. When the command input pin Execute goes low, all output pins revert to their default values.

	TransitionParameter	Retained parameters	YES	0	-	ARRAY[0..1] OF LREAL
OUTPUT	Done	Interpolation motion finished	YES	FALSE	TRUE/FALSE	BOOL
	Busy	Busy flag	YES	FALSE	TRUE/FALSE	BOOL
	Active	In control, TRUE when the command is executed	YES	FALSE	TRUE/FALSE	BOOL
	CommandAborted	Termination of Execution	YES	FALSE	TRUE/FALSE	BOOL
	Error	Command error flag	YES	FALSE	TRUE/FALSE	BOOL
	ErrorID	Command error code	YES	0	-	SMC_ERROR

7.5.2 Functions

This command realizes the linear interpolation function of 2-axis relative position.

Interpolation steps

(1) Determine the interpolated axis group. The AxesGroup variable type is AXES_GROUP_REF.

(2) Specify the axis number in the axis group variable, corresponding to the axis number of the combined axis and sub-axis.

(3) Complete the enable of each axis and axis group.

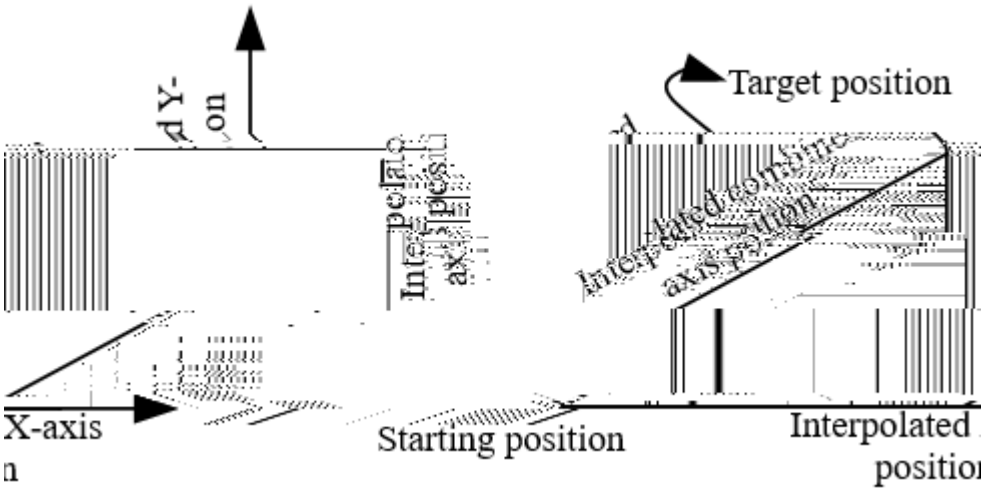
(4) Call the relative position linear interpolation command to execute the linear interpolation action.

Basic functions

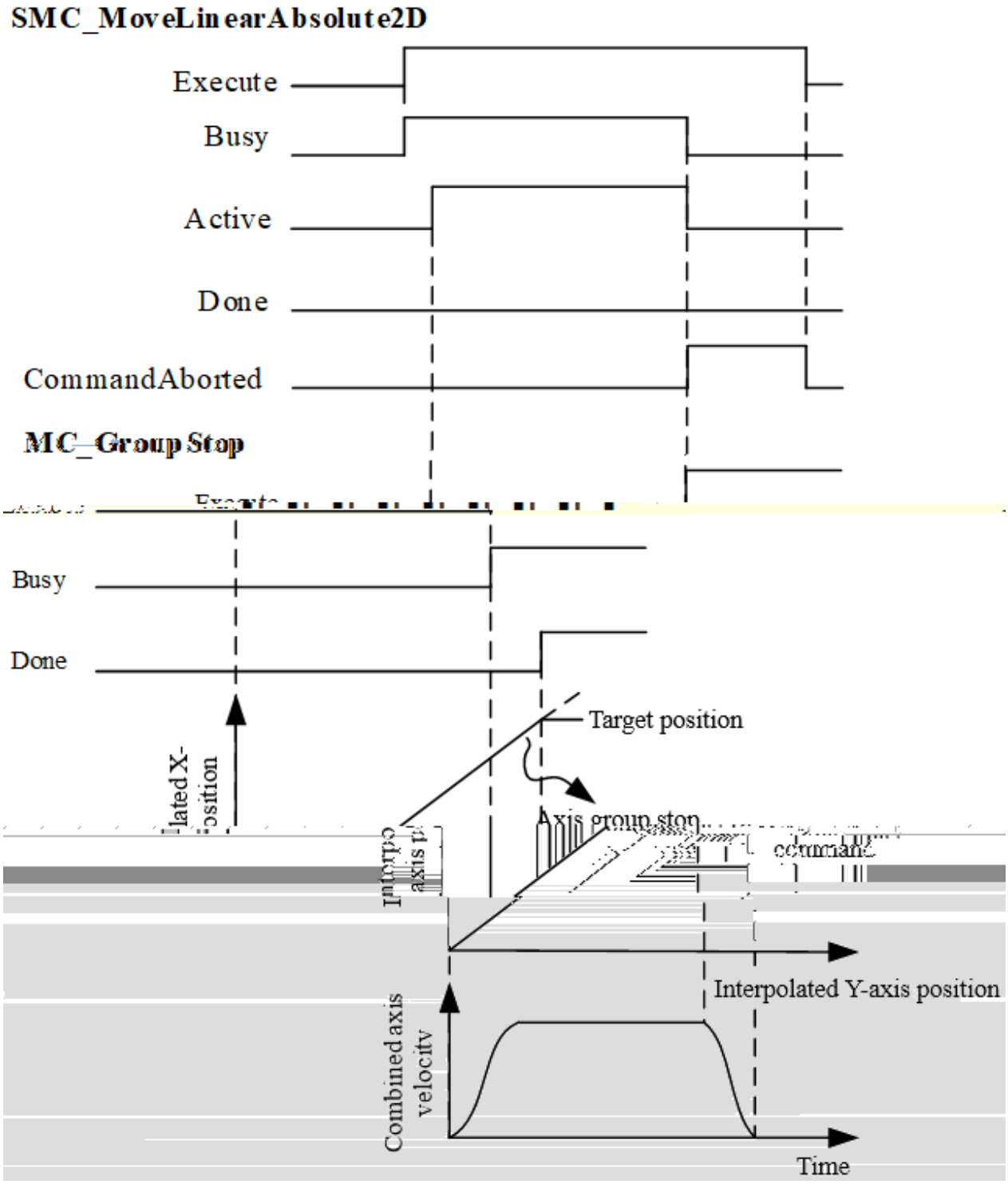
Target location

With the start time of this command as the starting point and the relative position of the two sub-axes set by the command input parameters as the endpoint, the connecting line between the starting point and the ending point is the motion track of the command.

The target position of this command is incremental motion according to the endpoint of the previous motion, meaning each step operates in incremental mode. The target position of the interpolation combined axis is based on the position at the time when the command starts to be executed, following the length of the hypotenuse of a right triangle, with the incremental positions of the two individual axes.



Input parameter latching



CoordSystem

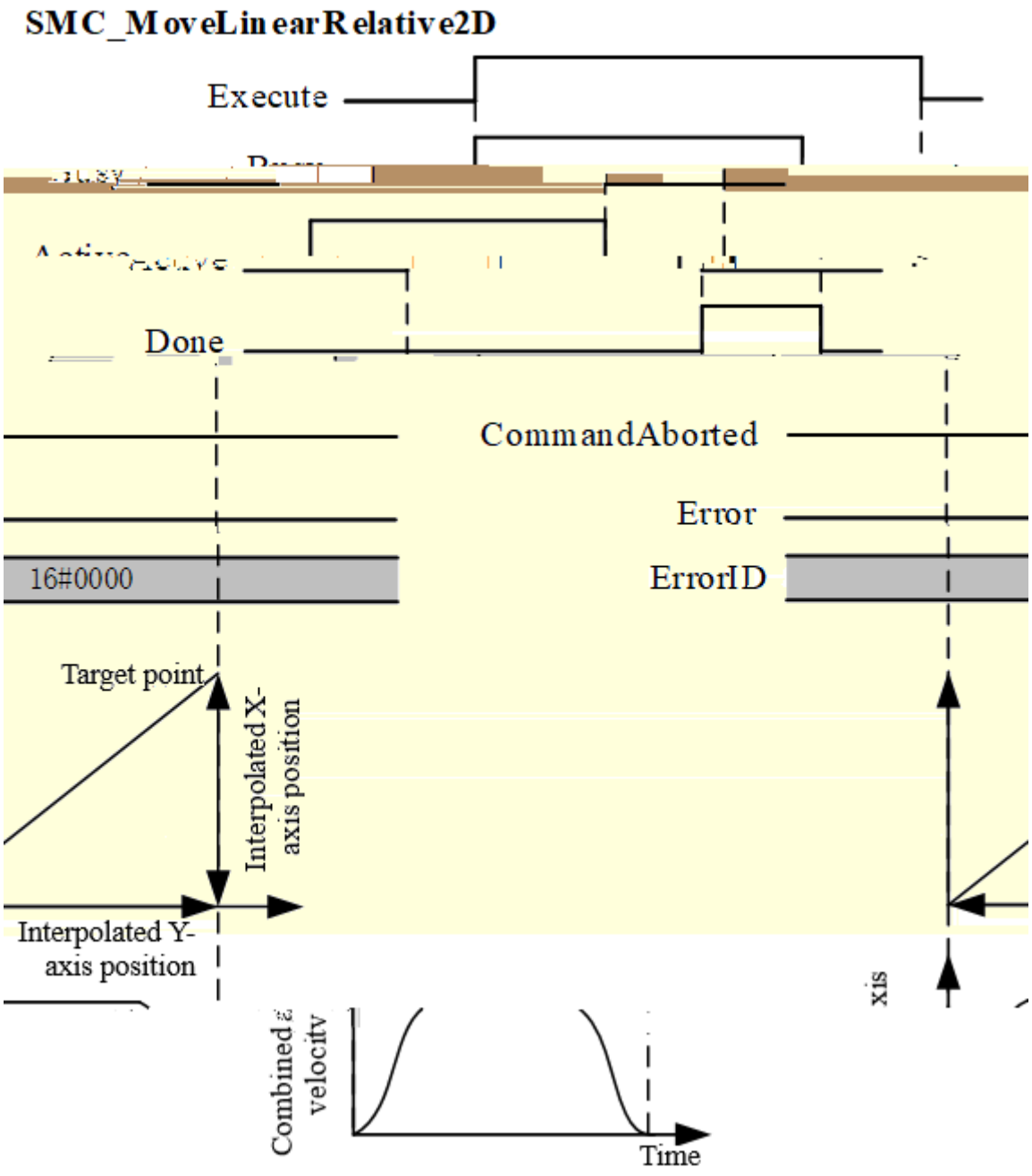
Specify the coordinate system for linear interpolation of relative position.

Use the axis coordinate system (ACS).

Timing diagram

The following is a normal and error timing diagram when the relative position interpolation command is executed.

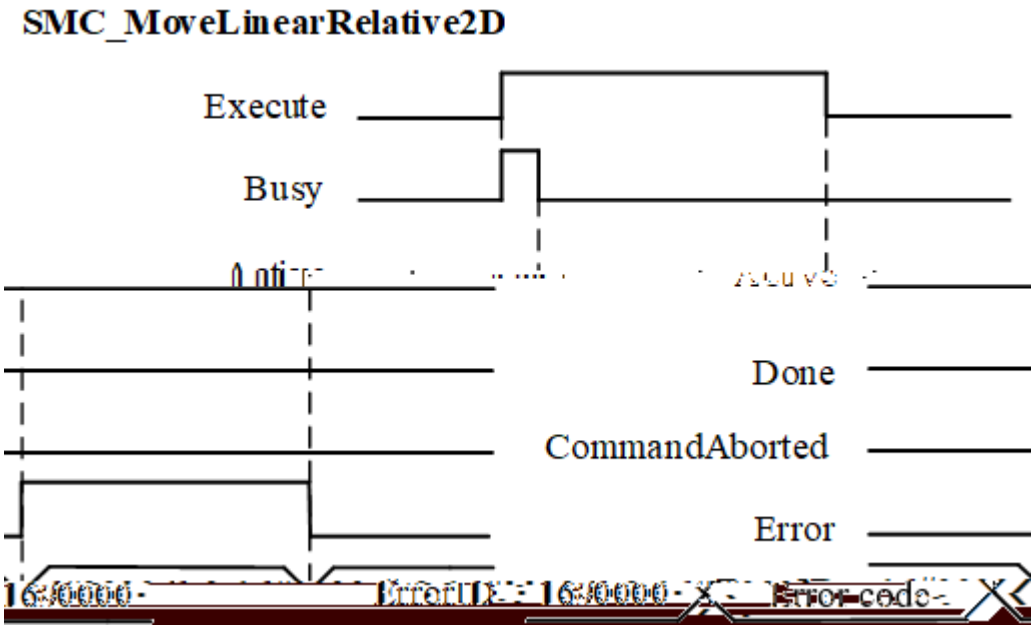
(1) Normal timing diagram



(2) Error pin timing diagram

When this command is executed, if there are abnormalities such as illegal parameters, the axis type of an axis in the axis group being 0: Unused (unused axis), and illegal ID of each axis in the axis group, the call of this command fails, with command output pin Error=True, and the error code ErrorID reports an abnormal error value. Please refer to the appendix Error Code Description to obtain the cause of the error. When the command input pin Execute goes low, all

output pins revert to their Default Values.



þ When the abnormal situation of the command is solved, it is necessary to trigger Execute again on the rising edge to send the interpolation command of the axis group and carry out the interpolation action of the relative position of the axis group.

7.5.3 Examples

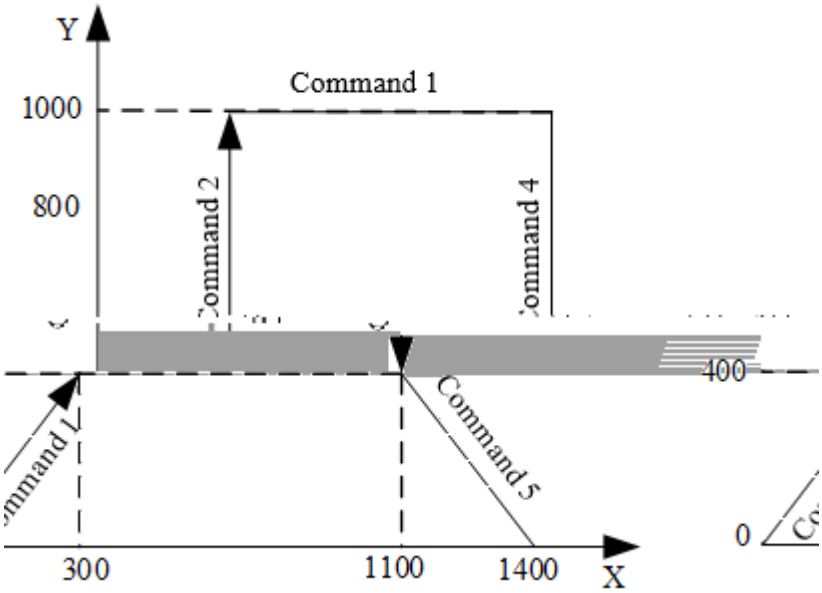
Action example

Action introduction

In the following example, 5 linear interpolation commands for relative positions are executed and will move to the target position after executing the interpolation command.

Set the BufferMode selection of interpolation commands as Buffered to buffer multiple motion commands.

In t t4.19 -.1 (ol)-8.9 (l)3.1 (o)ring r earroverrine2 (pl)6 (nt)-1.1 (e(ons)-8.1t)-1.1 (er) (r)-9 (nt)-1.1 (r)-6.4 4 (m)-24.4 (v)-11.8rrlreirat



Sequentially execute the linear interpolation command 1 to (300, 400) position, the linear interpolation command 2 to (300, 1000) position, the linear interpolation command 3 to (1100, 1000) position, and the linear interpolation command 4 to (1100, 400) position. Execute the linear interpolation command 3 to (1400, 0) position.

LD diagram program

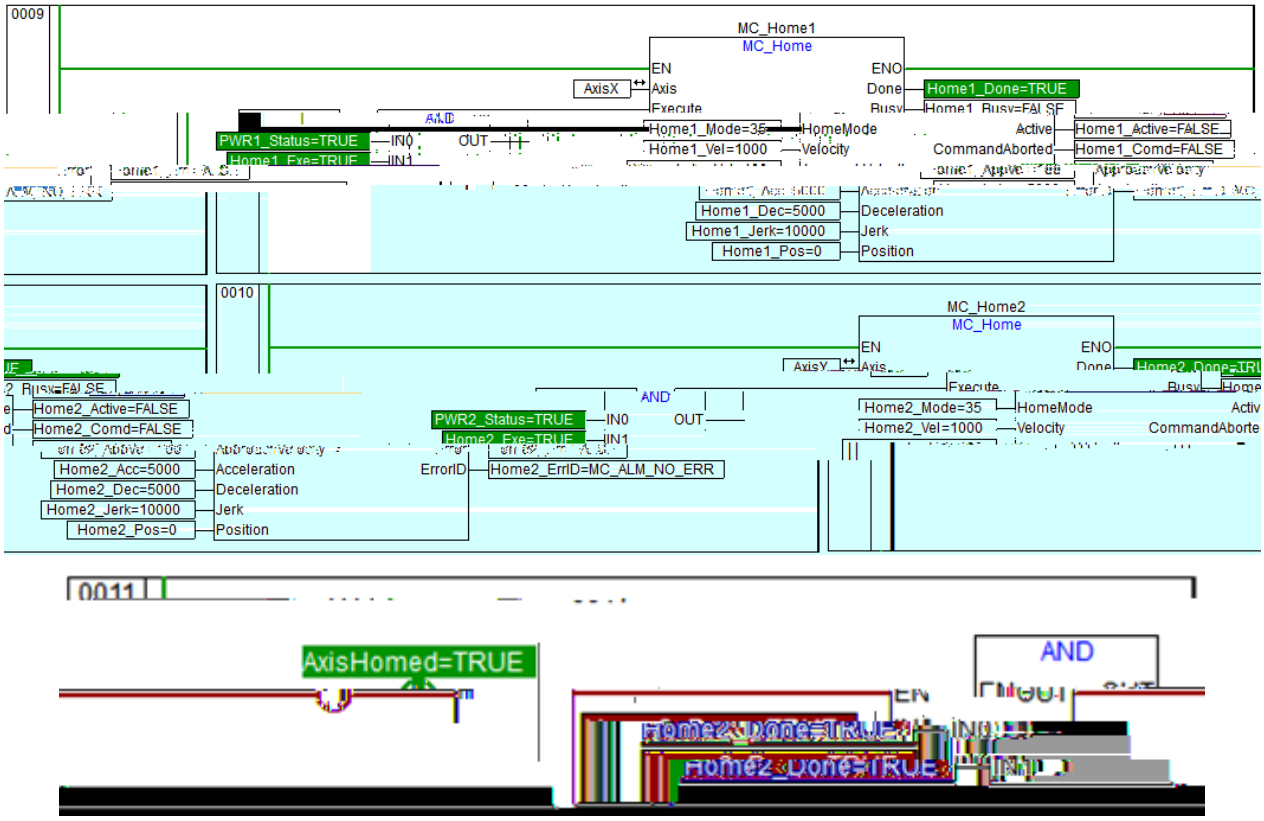
Set the speed of each interpolation command to 100, acceleration and deceleration to 200, acceleration to 5000, and buffer mode to Buffered.

Description of Primary Variables

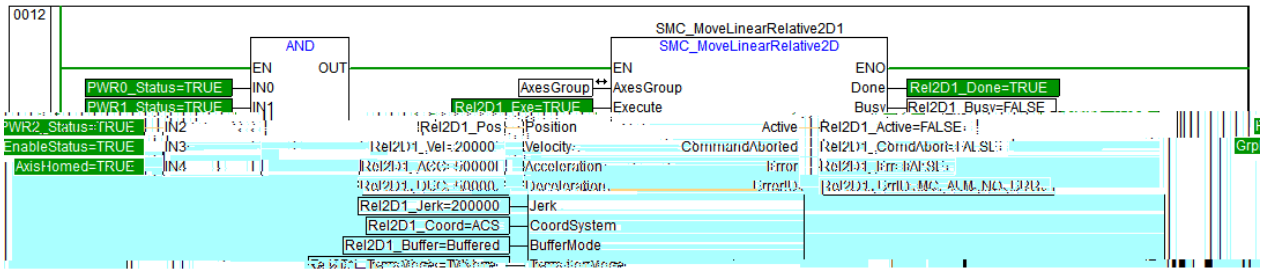
Variable name	Variable type	Initial value	Notes
AxesGroup	AXES_GROUP_REF	-	axis group name

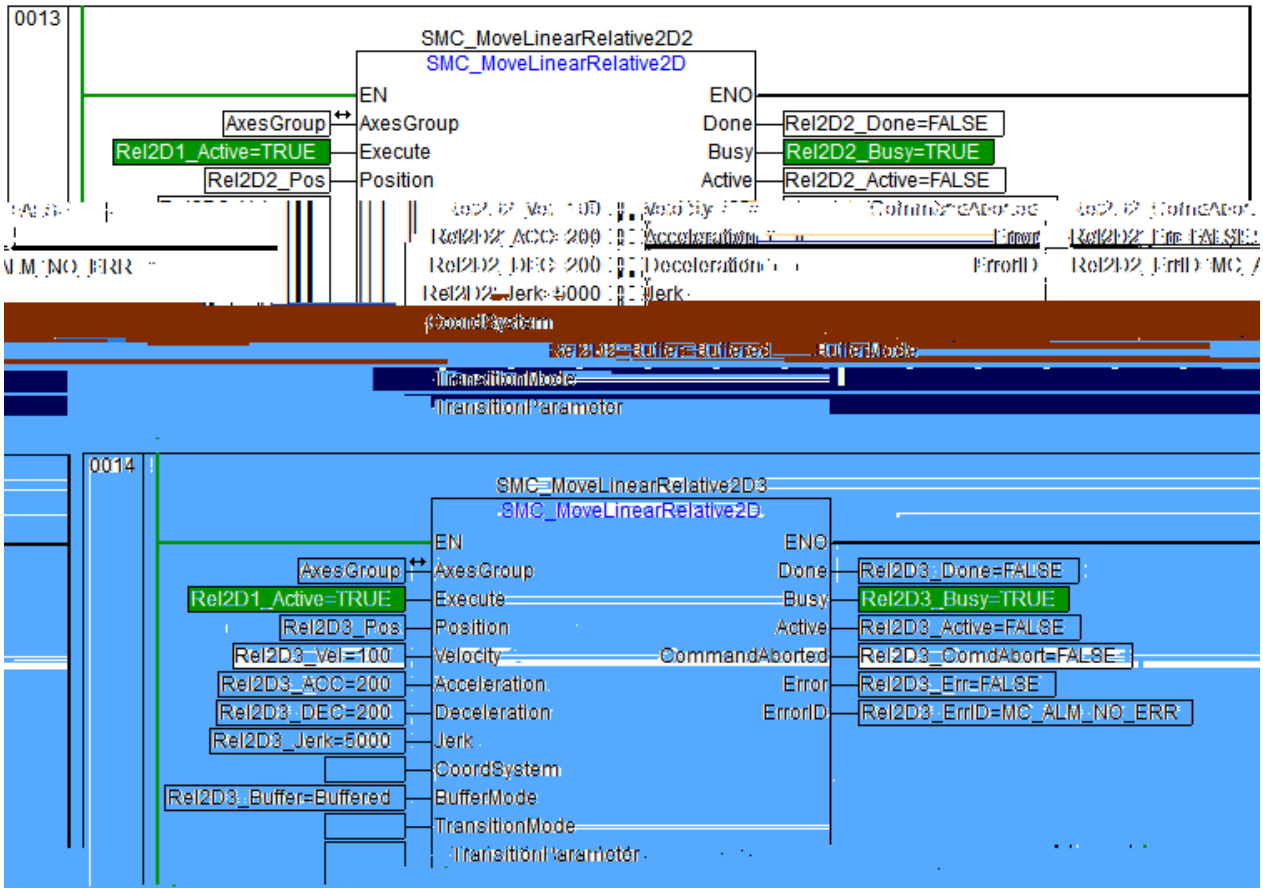
Sett1 06 T(ax)-8 (i)3.1 Cs 06 T(l)-13.2 CD -2.9 Cs 06 T(of)-13.1 C -1.59 (c)-8 (o)1.58 (m -244 r(bi)3.1 Cned)-123 (1 06 T(a)-123 (1n

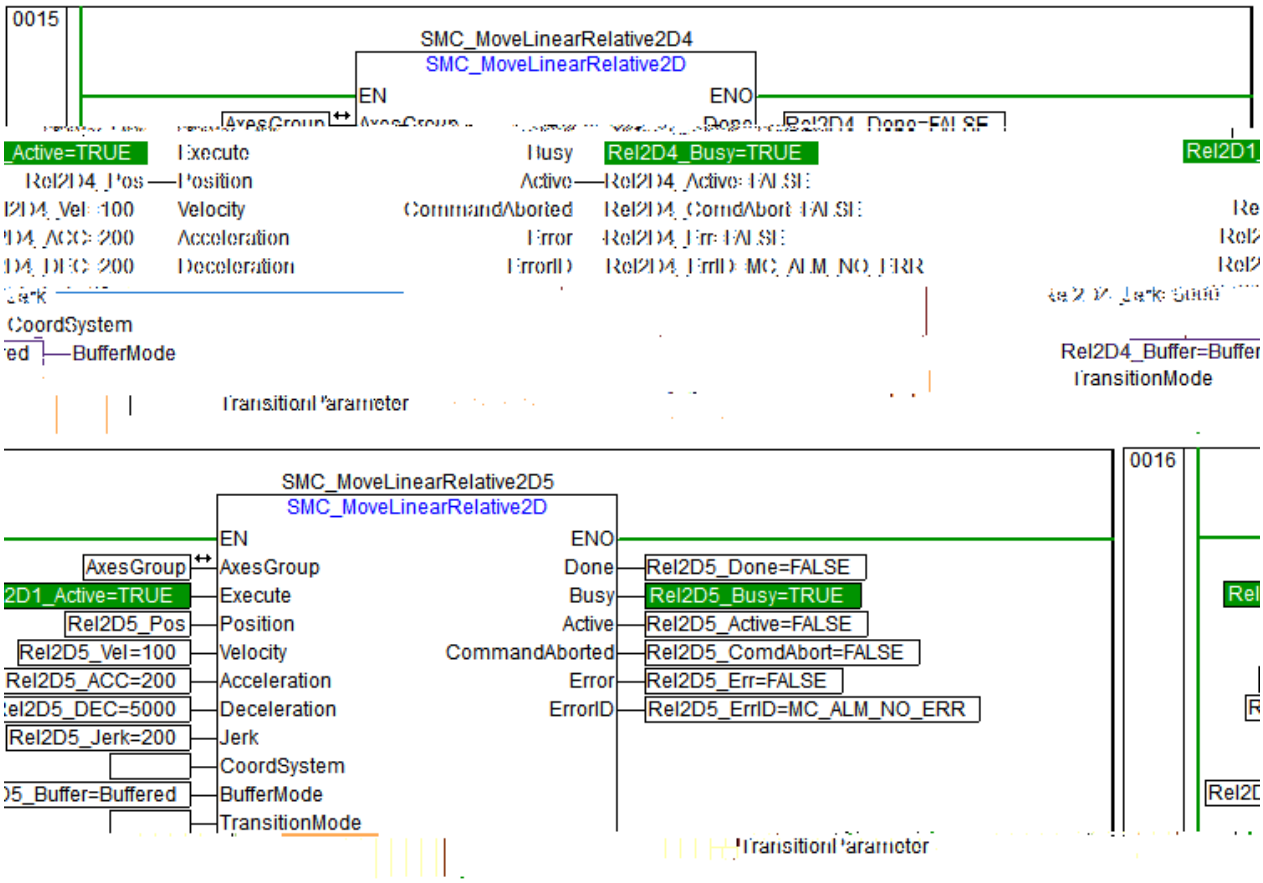
the homing operation. After the homing is completed, set AxisHomed to TRUE. See Section 0009-0011.



(6) In Section 0011, after the axis homing is completed, the interpolation motion command begin to execute. Execute command 1, and deliver commands 2-5 in sequence when Rel2D1_Active of command 1 is TRUE. See section 0012-0016 below.


0012
AND
EN
OUT
PWR0_Status=TRUE
IN0
PWR1_Status=TRUE
IN1
PWR2_Status=TRUE
IN2
EnableStatus=TRUE
IN3
AxisHomed=TRUE
IN4
SMC_MoveLinearRelative2D1
SMC_MoveLinearRelative2D
EN
AxesGroup
Execute
Position
Velocity
Acceleration
Deceleration
Jerk
CoordSystem
BufferMode
TransitionMode
TransitionParameter
ENO
Done
Busy
Active
CommandAborted
Error
ErrorID
AxesGroup
Rel2D1_Exe=TRUE
Rel2D1_Pos
Rel2D1_Vel=20000
Rel2D1_Jerk=200000
Rel2D1_Coord=ACS
Rel2D1_Buffer=Buffered
Rel2D1_Done=TRUE
Rel2D1_Busy=FALSE
Rel2D1_Active=FALSE
Rel2D1_ComdAbort=FALSE
Rel2D1_Err=FALSE
Rel2D1_ErrID=MC_ALM_NO_ERR


0013
SMC_MoveLinearRelative2D2
SMC_MoveLinearRelative2D
EN
AxesGroup
Execute
Position
Velocity
Acceleration
Deceleration
Jerk
CoordSystem
BufferMode
TransitionMode
TransitionParameter
ENO
Done
Busy
Active
CommandAborted
Error
ErrorID
AxesGroup
Rel2D1_Active=TRUE
Rel2D2_Pos
Rel2D2_Done=FALSE
Rel2D2_Busy=TRUE
Rel2D2_Active=FALSE
0014
SMC_MoveLinearRelative2D3
SMC_MoveLinearRelative2D
AxesGroup
Rel2D1_Active=TRUE
Rel2D3_Pos
Rel2D3_Vel=100
Rel2D3_ACC=200
Rel2D3_DEC=200
Rel2D3_Jerk=5000
Rel2D3_Buffer=Buffered
Rel2D3_Done=FALSE
Rel2D3_Busy=TRUE
Rel2D3_Active=FALSE
Rel2D3_ComdAbort=FALSE
Rel2D3_Err=FALSE
Rel2D3_ErrID=MC_ALM_NO_ERR



ST language program

Description of Primary Variables

Variable name	Variable type	Initial value	Notes

			delivered by the axis group. When this variable becomes TRUE, the axis group executes the interpolation command.

(1) Set the axis IDs of combined and split axes in the axis group. Set the defined axis IDs of Axis I, Axis X and Axis Y.

```
IF Switch1 THEN
AxisI.AxisID:=0;
AxisX.AxisID:=1;
AxisY.AxisID:=2;
AxesGroup.AxisInterp_ID:=AxisI.AxisID;
AxesGroup.AxisX_ID:=AxisX.AxisID;
AxesGroup.AxisY_ID:=AxisY.AxisID;
END_IF
```

(2) Set command motion parameters.

```
IF FALSE = InitFlag THEN
(* Parameter setting of relative position linear interpolation command 1. Default values are used for input parameters that have not been set.
Rel2D1_Pos[0]:=300;
Rel2D1_Pos[1]:=400;
Rel2D1_Vel:=100;
Rel2D1_ACC:=200;
Rel2D1_DEC:=200;
Rel2D1_Jerk:=5000;
Rel2D1_Buffer:=Buffered;
(* Parameter setting of relative position linear interpolation command 2. Default values are used for input parameters that have not been set.
Rel2D2_Pos[0]:= 0;
Rel2D2_Pos[1]:=400;
Rel2D2_Vel:=100;
Rel2D2_ACC:=200;
Rel2D2_DEC:=200;
Rel2D2_Jerk:=5000;
Rel2D2_Buffer:=Buffered;
(* Parameter setting of relative position linear interpolation command 3. Default values are used for input parameters that have not been set.
Rel2D3_Pos[0]:=300;
Rel2D3_Pos[1]:=400;
Rel2D3_Vel:=100;
Rel2D3_ACC:=200;
Rel2D3_DEC:=200;
Rel2D3_Jerk:=5000;
Rel2D3_Buffer:=Buffered;
(* Parameter setting of relative position linear interpolation command 4. Default values are used for input parameters that have not been set.
Rel2D4_Pos[0]:= 0;
Rel2D4_Pos[1]:=400;
```

```
Rel2D4_Vel:=100;
Rel2D4_ACC:=200;
Rel2D4_DEC:=200;
Rel2D4_Jerk:=5000;
Rel2D4_Buffer:=Buffered;
(* Parameter setting of linear interpolation command 5 for relative position. Default values are used for input parameters that have not been set.
Rel2D5_Pos[0]:=300;
Rel2D5_Pos[1]:=400;
Rel2D5_Vel:=100;
Rel2D5_ACC:=200;
Rel2D5_DEC:=200;
Rel2D5_Jerk:=5000;
Rel2D5_Buffer:=Buffered;
InitFlag:=TRUE;
END_IF
```

(3) Call the SMC_AxisInit command to complete the axis initialization configuration of three axes in the axis group. The axis type of the axis combination is set to 0: Virtual; the axis types of AxisX/AxisY are both set to 50: EtherCAT_Position. The SMC_AxisInit command is described in the commands section.

(4) Execute the MC_Power command to complete the axis enable of three axes in the axis group.

```
IF StartPro THEN
MC_Power0 (*Combined axis enable *)
Enable := PWR0_Enable,
Axis := AxisI,
Status=> PWR0_Status,
Busy=> PWR0_Busy,
Error=> PWR0_Err,
ErrorID=> PWR0_ErrID);
MC_Power1 (*Sub-axis AxisX enable *)
Enable := PWR1_Enable,
Axis := AxisX,
Status=> PWR1_Status,
Busy=> PWR1_Busy,
Error=> PWR1_Err,
ErrorID=> PWR1_ErrID);
MC_Power2 (*Sub-axis AxisY Enable *)
Enable := PWR2_Enable,
Axis := AxisY,
Status=> PWR2_Status,
Busy=> PWR2_Busy,
Error=> PWR2_Err,
ErrorID=> PWR2_ErrID);
END_IF
```

(5) Enable axis group

```
MC_GroupEnable1(
Execute := GrpEnable_Exe,
AxesGroup := AxesGroup,
Done=> GrpEnable_Done,
Busy=> GrpEnable_Busy,
Error=> GrpEnable_Err,
ErrorID=> GrpEnable_ErrorID);
GrpEnableStatus:= AxesGroup. AxesGroupDisabled;
```

(6) Execute the homing command of the split axis in the axis group to determine the home. See its command section for how to use the MC_Home command.

```
IF PWR1_Status a1g30.5 HRE
```

C_Hom1(()]TJ 0 Tc 0 Tw51.687 0 Td ( )Tj EMC ET /Artifact BMC 0.937 g 66.65190.72 473.52 14.4 re f* EMC BT /P <<

```
AxesGroup := AxesGroup,
Done=> Rel2D5_Done,
Busy=> Rel2D5_Busy,
Active=> Rel2D5_Active,
CommandAborted=> Rel2D5_ComdAbort,
```

		trigger				
	CircMode	Mode selection 0: Three o'clock 1: center/end point	YES	0	0:BORDER 1:CENTER	MC_CIRC_MODE
	AuxPoint	When CircMode is BORDER, AuxPoint is the intermediate point position; When CircMode is CENTER, AuxPoint is the position of center point.	No	-	Positive/negative/0	ARRAY[0..1] OF LREAL
	EndPoint	End position	No			

		parameters				
OUTPUT	Done	Intelation motion finished	YE	FAL	TRUE/FAL	B
	Buy	Busy flag	YE	FAL	TRUE/FAL	B
	Ati	In ontrol, TRUE when the command is executed	YE	FAL	TRUE/FAL	B
[redacted]	CommandAbed	Tmination of Executio	YE	FAL	TRUE/FAL	B E

Er f 1 q 1.72 21.36 re TRUE/FA4 612.24 0.7 ET Q BT 9 -0 0 9 381.48 620.04 Tm ( )Tj ET

Circular interpolation mode (CircMode)

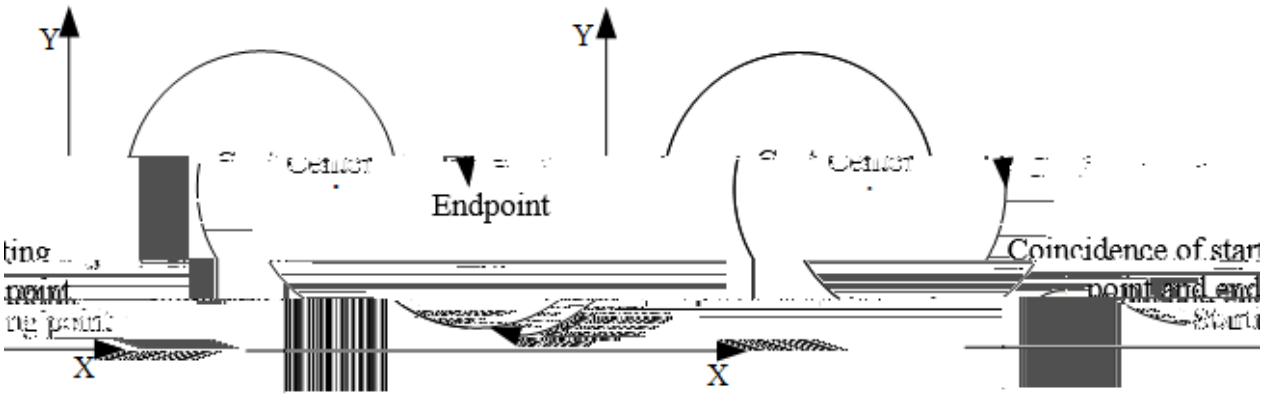
There are two circular interpolation modes: three-point mode and center/endpoint mode, which can be specified by CircMode. When CircMode is BORDER, AuxPoint is the middle point; when CircMode is CENTER, AuxPoint is the center point.

(a) Three-point mode (CircMode=BORDER)

Starting from the current position, perform circular interpolation that connects the middle point AuxPoint(X, Y) and the endpoint EndPoint(X, Y). Linear interpolation from the starting point to the endpoint is performed when thoes points are on the same line, or the middle point and endpoint are at the same point, and the starting point and middle point are at the same point.

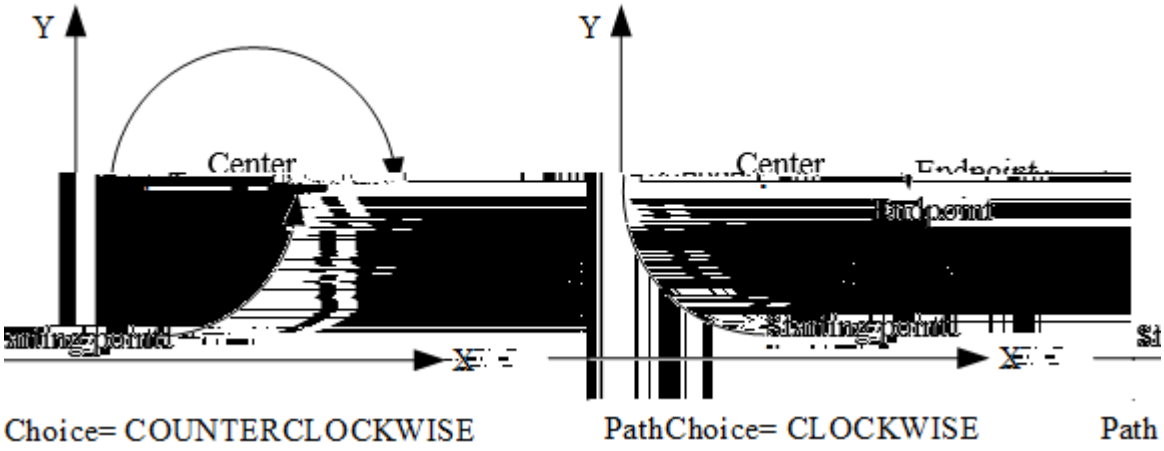
(b) Circle mode/endpoint mode (CircMode=CENTER)

Starting from the current position, perform circular interpolation by connecting the EndPoint(X, Y) with the arc specified by the center position AuxPoint(X, Y). The direction of rotation of the



Motion direction of circular interpolation (PathChoice)

This function is effective when CircMode is CENTER, and the direction of motion PathChoice is 0: CLOCKWISE and 1: COUNTERCLOCKWISE. The following figure shows the interpolation trajectory where AuxPoint and EndPoint of two interpolated circular arcs are consistent, but the direction of circular interpolation motion is opposite.



Handling of repeated triggering of function block

When a function block command is waiting for repeated triggering, the repeatedly triggered command will be put into the motion buffer. The monitoring status of the function block is the running state of the next command, and the running state of the previous command will not be monitored. During the execution of the previous interpolation command, the next interpolation command starts to be executed after the end of the previous one.

BufferMode (buffer mode selection)

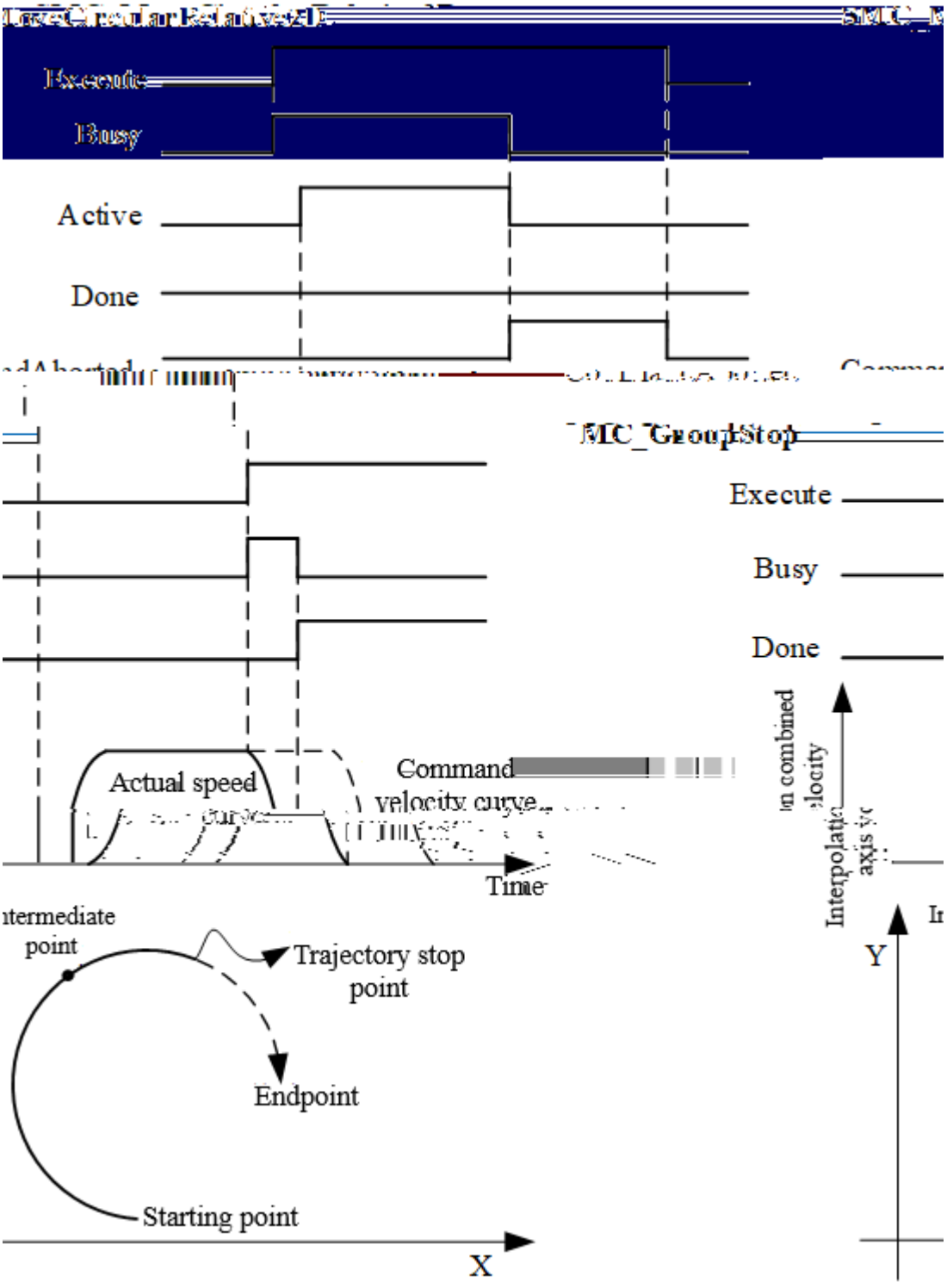
The 2-axis circular interpolation command supports BufferMode, including 1: Buffered, 2: BlendingLow, 3: BlendingPrevious, 4: BlendingNext and 5: BlendingHigh.

When this command is delivered, if BufferMode is 1:Buffered, the bufferd command will be automatically started from the cycle when the currently executing command ends normally. If BufferMode is 2:BlendingLow, the adjacent commands are blended at a low speed. If BufferMode is 3: BlendingPrevious, the speed of the previous command will be blended. If BufferMode is 4:

BlendingNext, the speed of the following command will be blended. If BufferMode is 5: BlendingHigh, it indicates high-speed fusion between adjacent commands.

For details about the buffer mode of interpolation commands, please refer to Buffer Mode Description in the appendix.

Command stop



CoordSystem

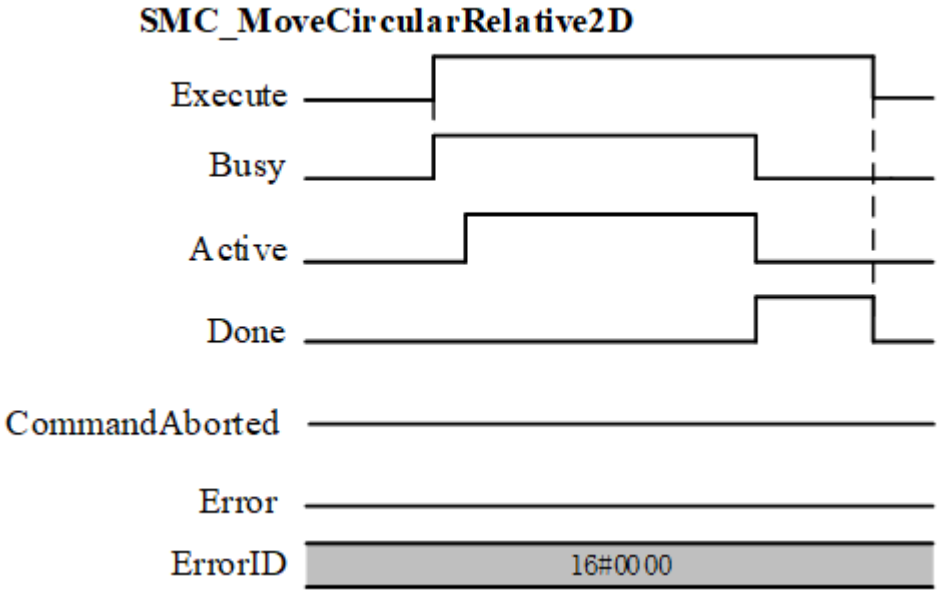
Specify the coordinate system for circular interpolation.

Use the axis coordinate system (ACS).

Timing diagram

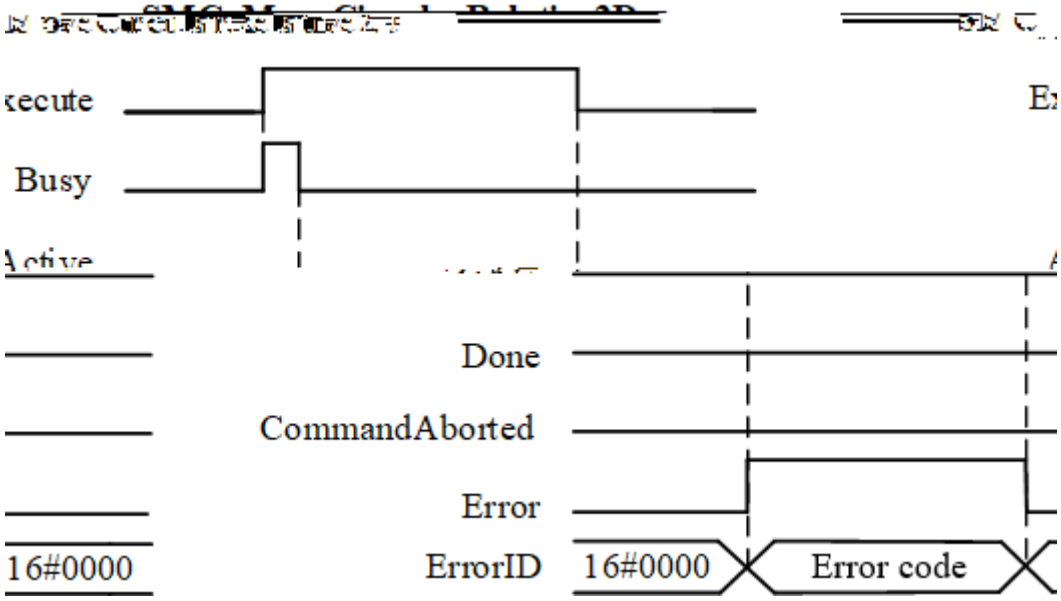
The following is a normal and error timing diagram when the circular interpolation command is executed.

(1) Normal timing diagram

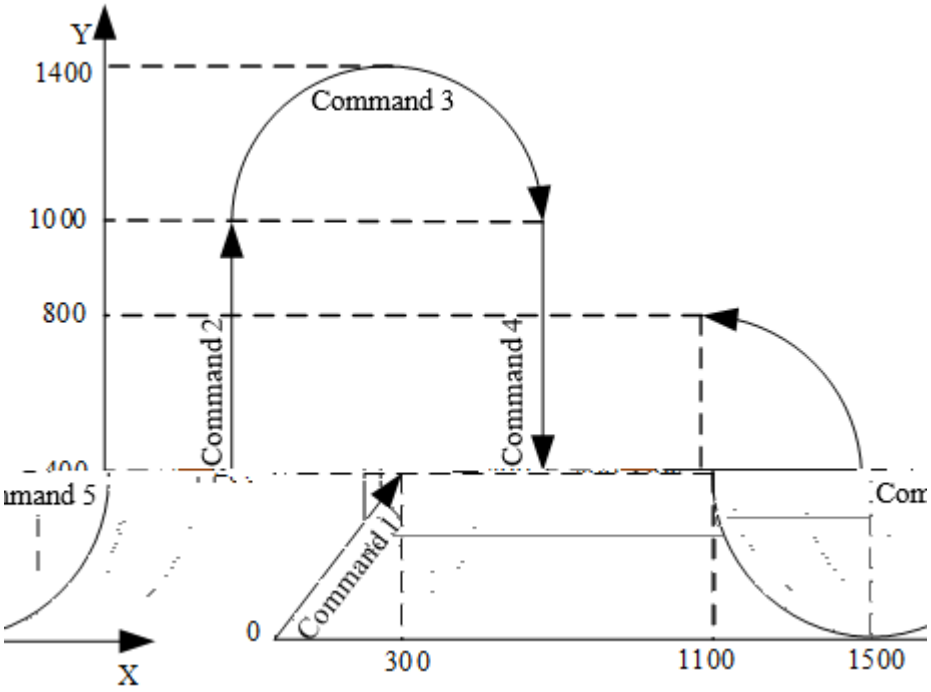


(2) Error pin timing diagram

During the execution of this command, if there are errors such as illegal parameters, 0: Unused (unused axis) for an axis in the axis group, and illegal ID of each axis in the axis group, the call of this command fails, with command output pin Error=True, and the error code ErrorID reports an abnormal error value. Please refer to the appendix Error Code Description to obtain the cause of the error. When the command input pin Execute goes low, all output pins revert to their Default Values.


Busy
Done
CommandAborted
Error
16#0000
ErrorID
16#0000
Error code

þ



The linear interpolation command 1 is executed to the position (300, 400), the linear interpolation command 2 is executed to the position (300, 1000), the circular interpolation command 3 is executed to the position (1100, 1000), the linear interpolation command 4 is executed to the position (1100, 400), and the circular interpolation command 5 is executed to the position (1500, 800) in sequence.

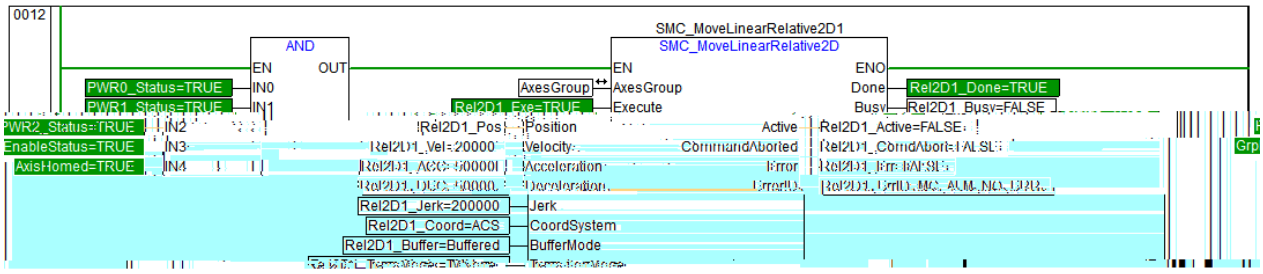
LD diagram program

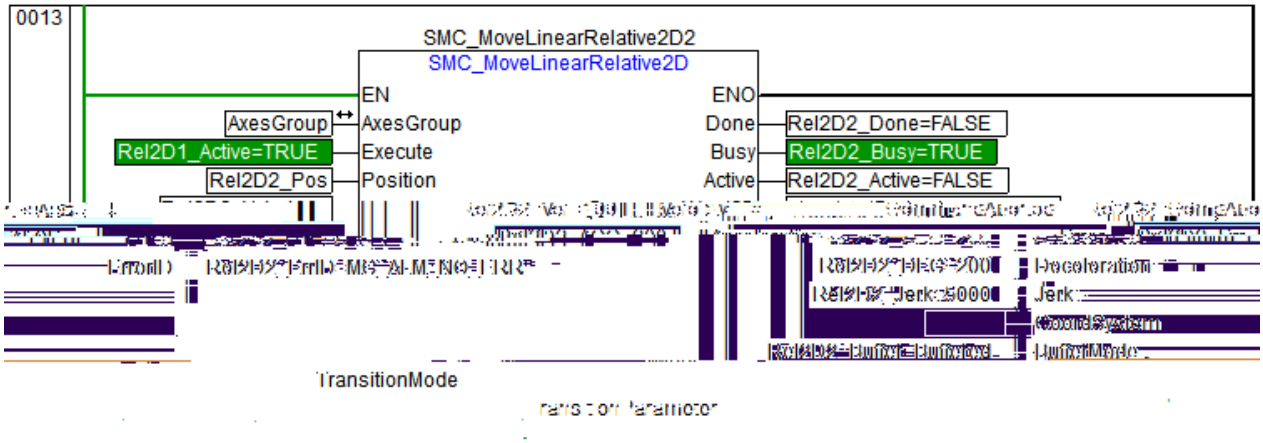
The speed of each interpolation command is set to 100, the acceleration and deceleration are all 200, the jerk is 5000, and the buffer mode is Buffered.

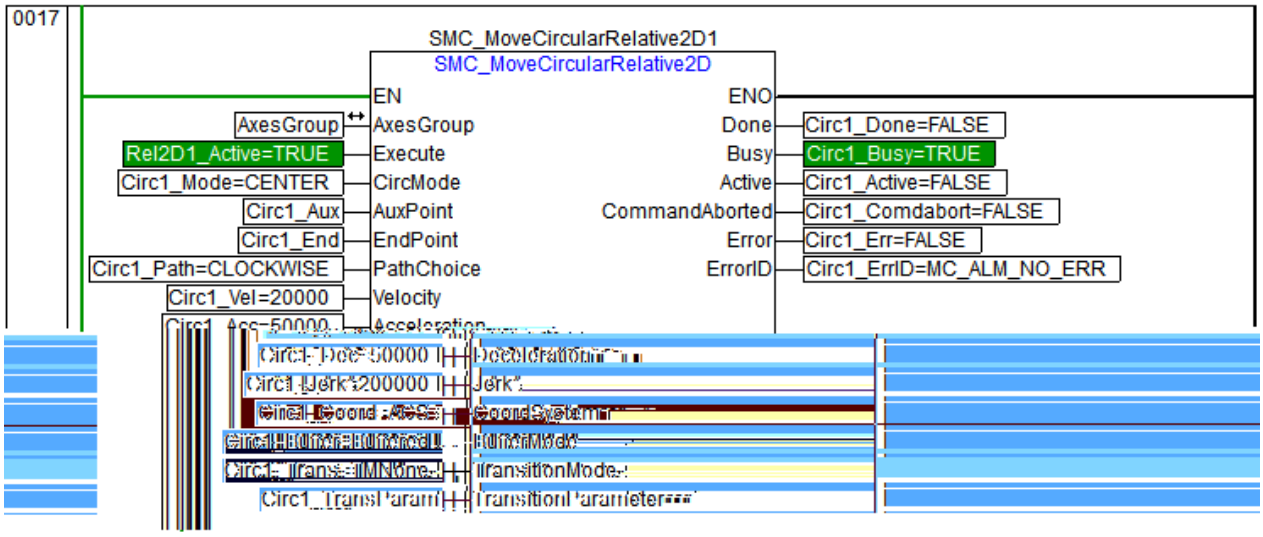
Description of Primary Variables

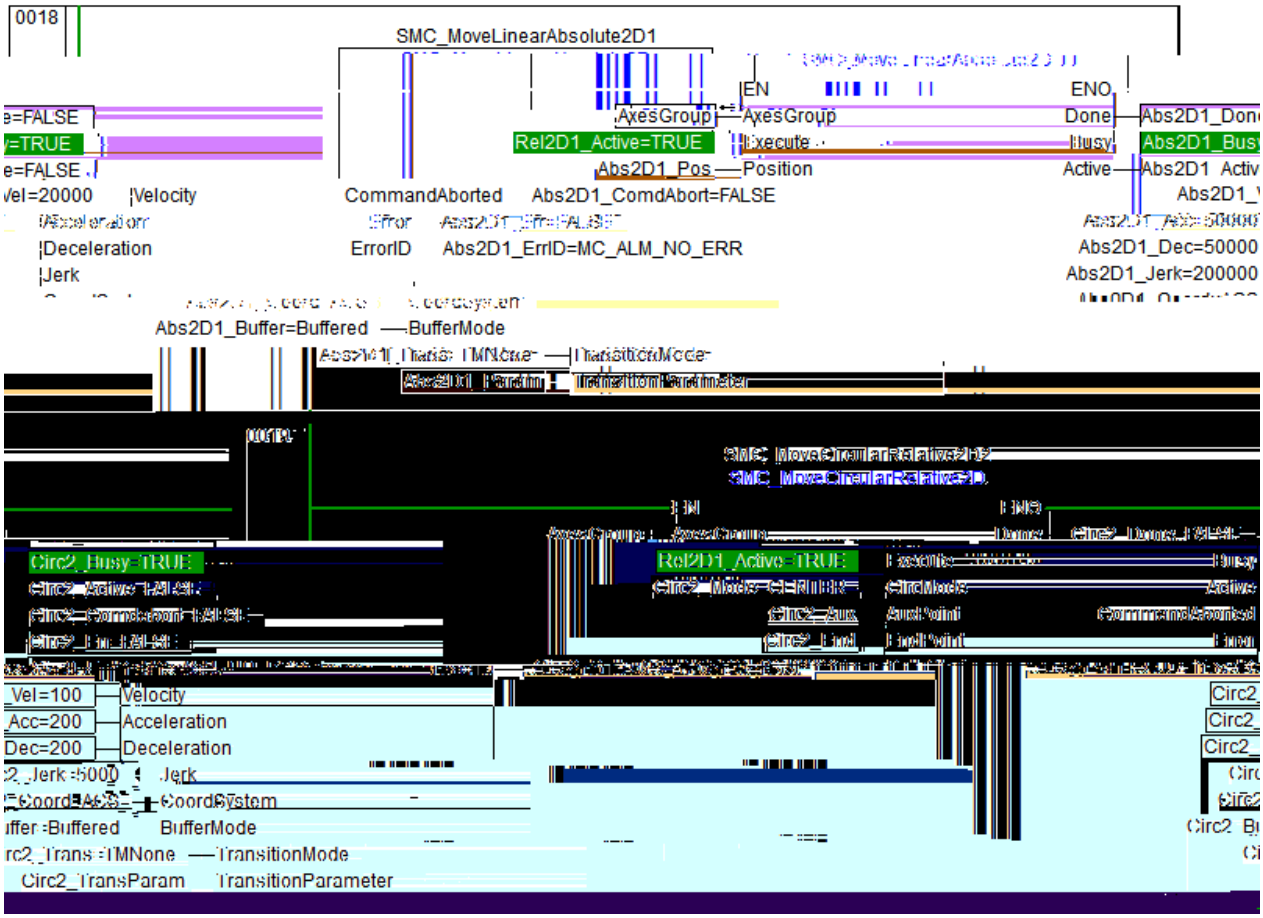
Variable Name	Variable T m1 >>BD(

AxisHomed	BOOL	FALSE	If this variable is TRUE, the return to the homing of each axis in the axis group is completed.
Rel2D1_Exe	BOOL	FALSE	


0012
AND
EN
OUT
IN0
IN1
IN2
IN3
IN4
PWR0_Status=TRUE
PWR1_Status=TRUE
PWR2_Status=TRUE
EnableStatus=TRUE
AxisHomed=TRUE
SMC_MoveLinearRelative2D1
SMC_MoveLinearRelative2D
EN
ENO
AxesGroup
Execute
Position
Velocity
Acceleration
Jerk
CoordSystem
BufferMode
Active
CommandAborted
Error
Done
Busy
AxesGroup
Rel2D1_Exe=TRUE
Rel2D1_Pos
Rel2D1_Vel=20000
Rel2D1_Jerk=200000
Rel2D1_Coord=ACS
Rel2D1_Buffer=Buffered
Rel2D1_Done=TRUE
Rel2D1_Busy=FALSE
Rel2D1_Active=FALSE


0013
SMC_MoveLinearRelative2D2
SMC_MoveLinearRelative2D
EN
ENO
AxesGroup
Execute
Position
Done
Busy
Active
AxesGroup
Rel2D1_Active=TRUE
Rel2D2_Pos
Rel2D2_Done=FALSE
Rel2D2_Busy=TRUE
Rel2D2_Active=FALSE
TransitionMode


0017
SMC_MoveCircularRelative2D1
SMC_MoveCircularRelative2D
EN
ENO
AxesGroup
Execute
CircMode
AuxPoint
EndPoint
PathChoice
Velocity
Acceleration
Jerk
CoordSystem
BufferMode
TransitionMode
Done
Busy
Active
CommandAborted
Error
ErrorID
AxesGroup
Rel2D1_Active=TRUE
Circ1_Mode=CENTER
Circ1_Aux
Circ1_End
Circ1_Path=CLOCKWISE
Circ1_Vel=20000
Circ1_Jerk=200000
Circ1_Done=FALSE
Circ1_Busy=TRUE
Circ1_Active=FALSE
Circ1_Comdabort=FALSE
Circ1_Err=FALSE
Circ1_ErrID=MC_ALM_NO_ERR



ST language program

Description of Primary Variables

Rel2D1_Exe	BOOL	FALSE	The rising-edge trigger variable of interpolation command 1 sent by the axis group.
Rel2D1_Active	BOOL	FALSE	The variable of the Active pin of interpolation command 1 sent by the axis group. When this variable becomes TRUE, the axis group executes the interpolation command.

(1) Set the IDs of joint and sub-axes in the axis group. Set the axis ID of the defined combined axis

```
    Circ1_Buffer:=Buffered;
(* Parameter setting of absolute position linear interpolation command 4. Default Values for unset input
parameters *)
    Abs2D1_Pos[0]:= 1100;
    Abs2D1_Pos[1]:=400;
    Abs2D1_Vel:=100;
    Abs2D1_ACC:=200;
    Abs2D1_DEC:=200;
    Abs2D1_Jerk:=5000;
    AbsD1_Buffer:=Buffered;
(* Parameter setting of circular interpolation command 5. Default Values for unset input parameters *)
    Circ1_Mode:= CENTER;
    Circ1_Aux [0] :=400;
    Circ1_Aux [1] := 0;
    Circ1_End [0] :=400;
    Circ1_End [1] :=400;
    Circ1_Path:= COUNTERCLOCKWISE;
    Circ1_Vel:=100;
    Circ1_ACC:=200;
    Circ1_DEC:=200;
    Circ1_Jerk:=5000;
    Circ1_Buffer:=Buffered;
    InitFlag:=TRUE;
```

(3) Call SMC_AxisInit command to complete the axis initialization configuration of three axes in the axis group. Set the axis type of axis combination to 0: Virtual; set the axis types of axis group AxisX/AxisY to 50: EtherCAT_Position. For the usage of SMC_AxisInit command, please refer to its command chapter.

(4) Execute the MC_Power command to enable the three axes in thh t001 Tofp-6.3 (.)]TJ 0 Tc 0 205 7.181 0 Td

```
Enable := PWR2_Enable,
Axis := AxisY,
Status=> PWR2_Status,
Busy=> PWR2_Busy,
Error=> PWR2_Err,
```

```
    Done=> Home2_Done,
    Busy=> Home2_Busy,
    Active=> Home2_Active,
    CommandAborted=> Home2_Comd,
    Error=> Home2_Err,
    ErrorID=> Home2_ErrID);
END_IF
IF  Home1_Done  AND Home2_Done THEN
    AxisHomed:=TRUE;
END_IF
```

(7) After the axis homing is completed, start executing the interpolation motion command. Execute command 1. When Rel2D1_Active of command 1 is TRUE, commands 2-5 are sent sequentially.

```
IF PWR0_Status  AND PWR1_Status  AND PWR2_Status  AND GrpEnableStatus AND AxisHomed THEN
SMC_MoveLinearRelative2D1(*interpolation command 1*)
    Execute := Rel2D1_Exe,
    Position := Rel2D1_Pos,
    Velocity := Rel2D1_Vel,
    Acceleration := Rel2D1_ACC,
    Deceleration := Rel2D1_DEC,
    Jerk := Rel2D1_Jerk,
    CoordSystem := Rel2D1_Coord,
    BufferMode := Rel2D1_Buffer,
    TransitionMode := Rel2D1_Trans,
    TransitionParameter := Rel2D1_Param,
    AxesGroup := AxesGroup,
    Done=> Rel2D1_Done,
    Busy=> Rel2D1_Busy,
    Active=> Rel2D1_Active,
    CommandAborted=> Rel2D1_ComdAbort,
    Error=> Rel2D1_Err,
    ErrorID=> Rel2D1_ErrID);
SMC_MoveLinearRelative2D2(*interpolation command 2*)
    Execute := Rel2D1_Active,
    Position := Rel2D2_Pos,
    Velocity := Rel2D2_Vel,
    Acceleration := Rel2D2_ACC,
    Deceleration := Rel2D2_DEC,
    Jerk := Rel2D2_Jerk,
    CoordSystem := Rel2D2_Coord,
    BufferMode := Rel2D2_Buffer,
    TransitionMode := Rel2D2_Trans,
    TransitionParameter := Rel2D2_Param,
    AxesGroup := AxesGroup,
    Done=> Rel2D2_Done,
    Busy=> Rel2D2_Busy,
```

```
    Active=> Rel2D2_Active,
    CommandAborted=> Rel2D2_ComdAbort,
    Error=> Rel2D2_Err,
    ErrorID=> Rel2D2_ErrID);
SMC_MoveCircularRelative2D1(*interpolation command 3*)
    Execute := Rel2D1_Active,
    CircMode := Circ1_Mode ,
    AuxPoint := Circ1_Aux,
    EndPoint := Circ1_End,
    PathChoice:= Circ1_Path,
    Velocity := Circ1_Vel,
    Acceleration := Circ1_ACC,
    Deceleration := Circ1_DEC,
    Jerk := Circ1_Jerk,
    CoordSystem := Circ1_Coord,
    BufferMode := Circ1_Buffer,
    TransitionMode := Circ1_ Trans,
    TransitionParameter := Circ1_Param,
    AxesGroup := AxesGroup,
    Done=> Circ1_Done,
    Busy=> Circ1_Busy,
    Active=> Circ1_Active,
    CommandAborted=> Circ1_Abort,
    Error=> Circ1_Err,
    ErrorID=> Circ1_ErrID);
SMC_MoveLinearAbsolute2D1(*interpolation command 4*)
    Execute := Rel2D1_Active,
    Position := Abs2D1_Pos,
    Velocity := Abs2D1_Vel,
    Acceleration := Abs2D1_ACC,
    Deceleration := Abs2D1_DEC,
    Jerk := Abs2D1_Jerk,
    CoordSystem := Abs2D1_Coord,
    BufferMode := Abs2D1_Buffer,
    TransitionMode := Abs2D1_Trans,
    TransitionParameter := Abs2D1_Param,
    AxesGroup := AxesGroup,
    Done=> Abs2D1_Done,
    Busy=> Abs2D1_Busy,
    Active=> Abs2D1_Active,
    CommandAborted=> Abs2D1_ComdAbort,
    Error=> Abs2D1_Err,
    ErrorID=> Abs2D1_ErrID);
SMC_MoveCircularRelative2D2(*interpolation command 5*)
    Execute := Rel2D1_Active,
```

```
    CircMode := Circ2_Mode ,
    AuxPoint := Circ2_Aux,
    EndPoint := Circ2_End,
    PathChoice:= Circ2_Path,
    Velocity := Circ2_Vel,
    Acceleration := Circ2_ACC,
    Deceleration := Circ2_DEC,
    Jerk := Circ2_Jerk,
    CoordSystem := Circ2_Coord,
    BufferMode := Circ2_Buffer,
    TransitionMode := Circ2_ Trans,
    TransitionParameter := Circ2_Param,
    AxesGroup := AxesGroup,
    Done=> Circ2_Done,
    Busy=> Circ2_Busy,
    Active=> Circ2_Active,
    CommandAborted=> Circ2_Abort,
    Error=> Circ2_Err,
    ErrorID=> Circ2_ErrID);
END_IF
```

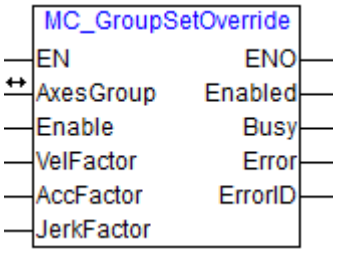
7.6.4 Precautions

When the starting point and endpoint are the same, an exception will occur that the starting point and end point of circular interpolation are the same.

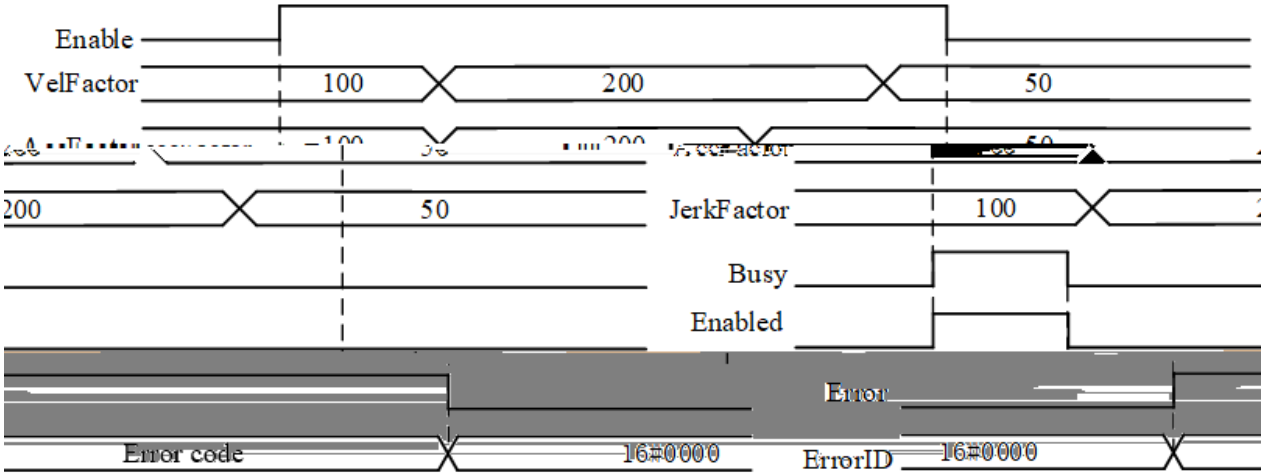
BufferMode (buffer mode selection) in this command does not support 0: Aborting.

7.7 MC_GroupSetOverride (Set Axes Group Override Factors)

This command sets the speed, acceleration/deceleration and jerk scale factor of the combined axis in the axis group, which can increase or decrease the running speed, acceleration/deceleration and jerk of the combined axis according to the set ratio.

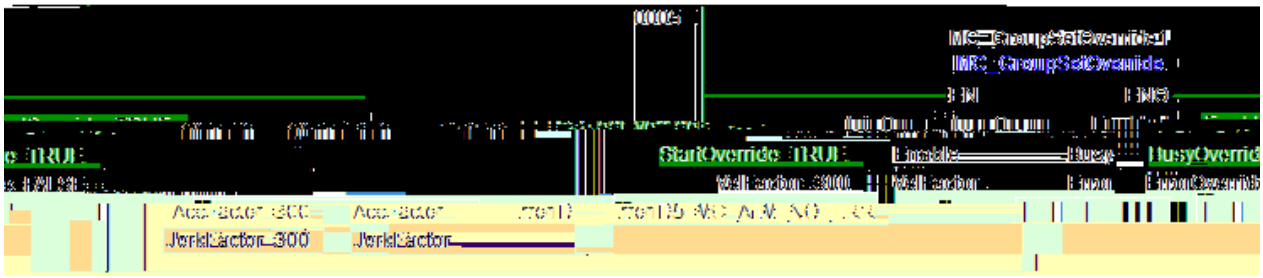


7.7.1 Parameter



(3) Restart and multiple start commands

The input of this command is Enable type. When Enable is TRUE, the function block runs all the time. When there is an MC_GroupSetOverride command being executed and other MC_GroupSetOverride commes Eartdee.ng (es)-8 (t)h3.1 (de)-13.2 (pl)a-6.4 (t)3r (ands)]TJ 0 Tc41650616.3



ST language program

(1) Initialize and enable the axis group. Set the IDs and types of X, Y and combined axes respectively. Set the combined axis type as a virtual axis. Then enable the axis group. After enabling, send SMC_MoveLinearAbsolute2D command to the axis group. The rising edge triggers the control axis group to run with 2-axis absolute position interpolation command. For specific usage, please refer to the section of SMC_MoveLinearAbsolute2D command.

(2) Keep MC_GroupSetOverride at high level to set the override of motion parameters.

```
MC_GroupSetOverride1(
Enable := StartOverride,
VelFactor := 300,
AccFactor := 300,
JerkFactor := 300,
Axis := Axis0,
Enabled=>EnabledOverride,
Busy=>BusyOverride,
Error=>ErrorOverride,
ErrorID=>ErrorID);
```

7.7.4 Precautions

When using this command for override setting, if the maximum speed of the axis group is exceeded, the maximum speed will be taken as the running speed.

This command has no effect on the encoder axis.

For a command that this command is invalid, Enabled will remain TRUE even if the "Enabled" of the command is set to TRUE.

This command is only valid for the combined axis in the axis group.

7.7.5 Reference

For axis group motion commands, refer to the section of SMC_MoveLinearAbsolute2D command.

For error codes, please refer to the appendix Functional Block Error Codes.

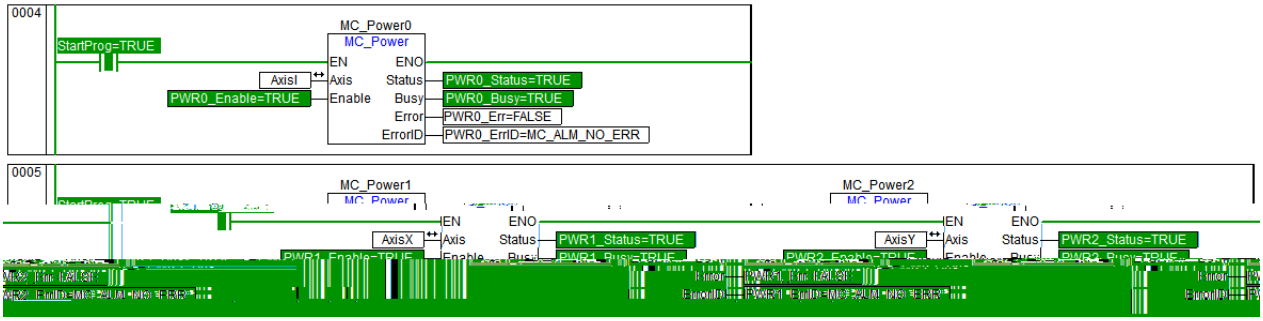
7.8 SMC_GroupSetDynamicLimits (Set Axes Group Dynamic Parameter Limits)

MaxVelocity is greater than the speed V0 of the axis group motion command. The axis group motion command takes the self-set command speed as the actual running speed, and the axis group motion command increases. Meanwhile, Max_Acc2, Max_Dec2 and Max_Jerk2 are the set limit values of other motion parameters respectively.

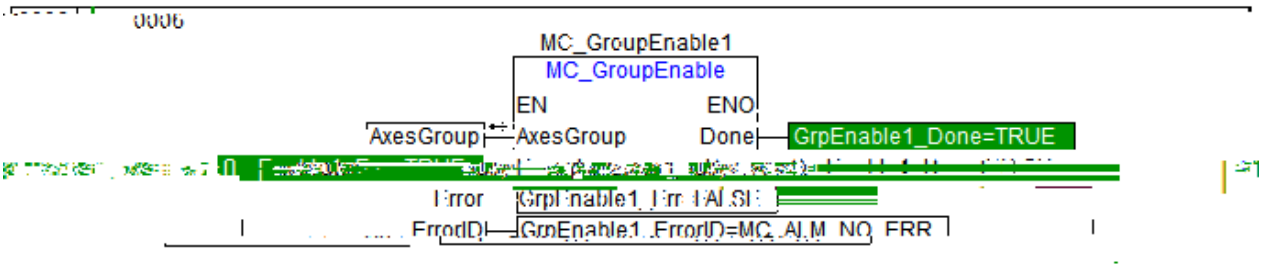
T Ed(2r)/4.46[-8)(-2)(n)c23mallhl2l 9 (e,)-1( )]TJ 0(er87 0 Td ( )Tj 12.6>Tj <</MCID 1 >>BDC /C2_0 1 T70.146 Tw 9.96

(2) Axis initialization. Call SMC_AxisInit command to complete the axis initialization configuration of three axes in the axis group respectively, and set the combined axis as 0: Virtual and the Sub-axis type as 50: EtherCAT_Position. Please refer to relevant chapters for the usage of SMC_AxisInit command.

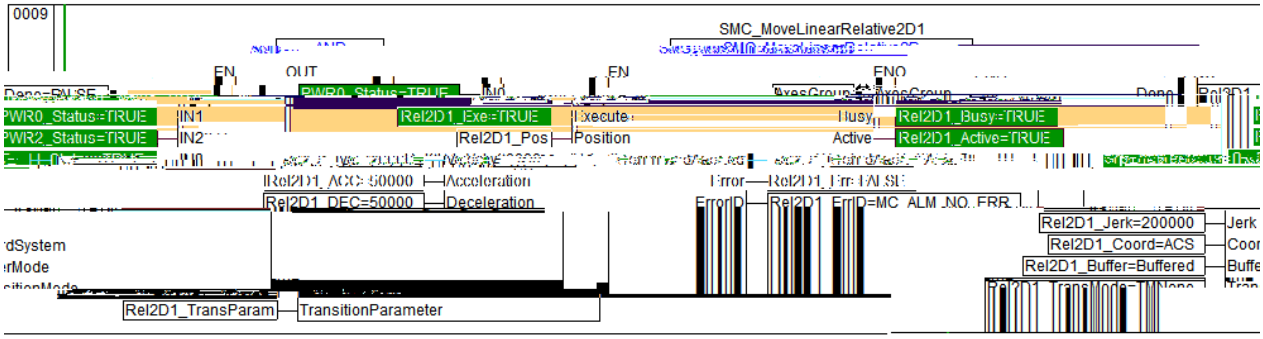
(3) Axis enable. After each axis in the axis group is ready (StartProg =TRUE), call MC_Power to enable the axes of the combined axis and two sub-axes of the axis group. See Sections 0004 to 0005.



(4) Axis group enable. Call MC_GroupEnable to enable the axis group. If AxesGroupDisabled in the axis group variables is FALSE, the axis group is enabled successfully. See Section 0006. After the axis group is enabled successfully, reverse AxesGroupDisabled and assign it to GrpEnableStatus variable.



(5) In Section 0009, after each axis in the axis group and the axis group are enabled, trigger the relative position interpolation motion command to control the movement of the axis group. Please refer to relevant chapters for the usage of SMC_MoveLinearRelative2D command.


0009
SMC_MoveLinearRelative2D1
EN
OUT
EN
ENO
IN1
IN2
Rel2D1_Exe=TRUE
Rel2D1_Pos
Execute
Position
Busy
Active
Rel2D1_Busy=TRUE
Rel2D1_Active=TRUE
Rel2D1_ACC=50000
Acceleration
Rel2D1_DEC=50000
Deceleration
Error
ErrorID
Rel2D1_ErrID=MC_ALM_NO_ERR
Rel2D1_Jerk=200000
Jerk
Rel2D1_Coord=ACS
Rel2D1_Buffer=Buffered
Rel2D1_TransParam
TransitionParameter

(6)

of the defined

ree axes in the
of axis group
13-8 ( )0.5 (c)-8 (1 (A)28 (o

```
MC_GroupEnable1(
Execute := GrpEnable_Exe,
AxesGroup := AxesGroup,
Done=> GrpEnable_Done,
Busy=> GrpEnable_Busy,
Error=> GrpEnable_Err,
ErrorID=> GrpEnable_ErrorID);
GrpEna1097 (a)-12.2 tG= AxesGrou.: Axesrouè¡#¸•  1B ìëÍH
```

parameters of the interpolation command are not limited, and the actual running speed of the interpolation command reaches the set command speed; after t1, the motion parameter limit set by the SMC_GroupSetDynamicLimits command takes effect, and the interpolation command speed is decelerated to the speed limit value. The diagram of limited interpolation closing axis motion parameters is

7.9 SMC_GroupCornerSpeedLimits (Set Axes Group Corner Speed Limits)

This command sets the corner speed limit parameters during the movement of the axis group to reduce the motion impact at non-smooth corner joints.

7.9.1 Parameter

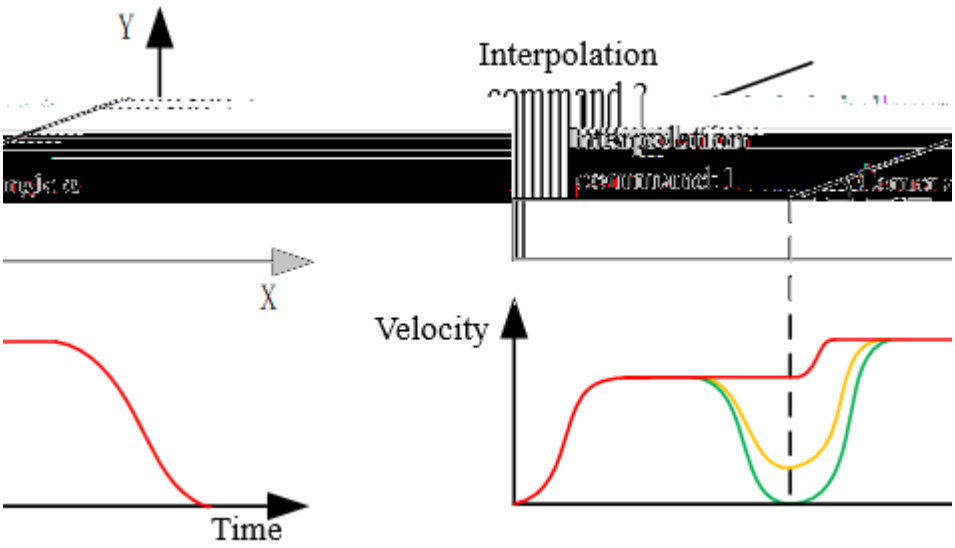
Parameter Type	Title	Description	Empty or Not	Default Value	Scope	Data Type

The critical deceleration angle DecAngle can limit the velocity at the junction of axis group motion commands, and the critical stop angle DecAngle limits the velocity at the junction of axis group motion commands to 0. The critical deceleration angle DecAngle is less than or equal to the critical stop angle DecAngle.

(3) Function description

Set the corner velocity limit parameters during the movement of the axis group, and the velocity at the joint of the interpolation motion command of the axis group is limited to reduce the motion impact at the non-smooth corner joint.

As shown in the following figure, for the speed curve at the junction of two interpolation commands, the speed of the first interpolation command is taken as the velocity at the junction of the two commands.



In the above figure, interpolation motion command 2 is set to be fused at the velocity of interpolation command 1, i.e. the starting point speed of the second interpolation command is set as the command speed of interpolation command 1.

If the set critical deceleration angle and critical stop angle are greater than the turning angle α between the two commands, the connecting speed between the two commands is not limited, and the velocity curve is shown as the red speed curve;

If the set critical deceleration angle is less than the turning angle α between the two commands, and the critical stop angle is greater than the turning angle α between the two commands, the connection velocity between the two commands is limited, and the velocity curve is shown as the orange velocity curve;

If the set critical deceleration angle and critical stop angle are both smaller than the turning angle α between the two commands, the connection velocity between the two commands is limited, and the velocity at the connection point is 0. The green velocity curve is shown in the velocity curve.

Timing diagram

(1) Normal timing diagram

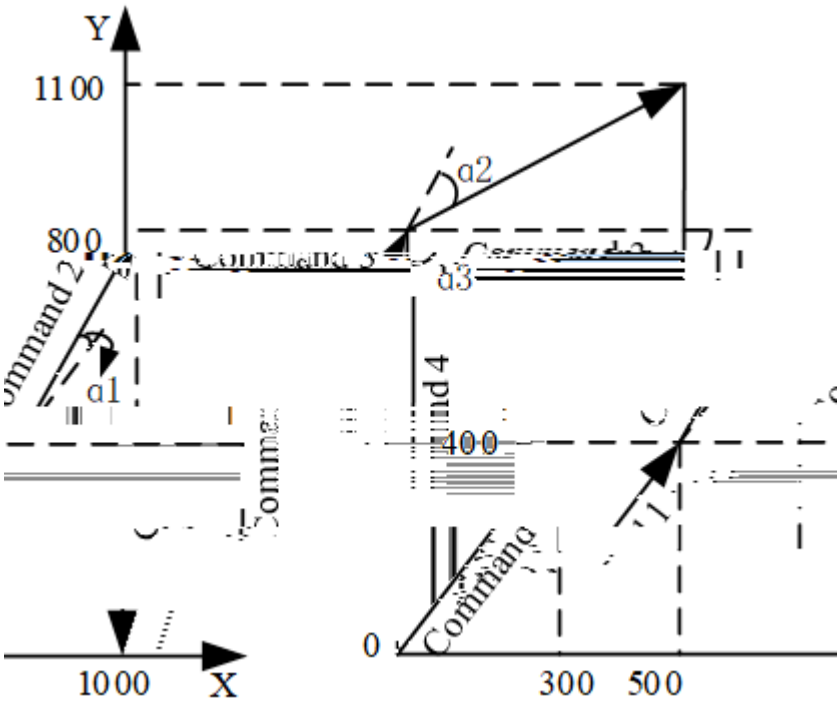
the axis group.

The following example executes 4 linear interpolation commands.

In this example, the buffer mode BufferMode of the linear interpolation command 1 is set to Buffered. The buffer modes of the remaining three interpolation commands are 3: velocity fusion of the previous command, 4: velocity fusion of the next command and 5: high-velocity fusion between adjacent commands.

In this example, when the output Active of the linear interpolation command 1 becomes TRUE, the remaining interpolation commands 2-4 are executed.

■ Schematic Diagram of Action



The linear interpolation command 1 reaches the position (300, 400), the linear interpolation command 2 reaches the position (500, 800), the linear interpolation command 3 reaches the position (1000, 1100), and the linear interpolation command 4 reaches the position (1000, 0) in this order.

The break angle α1 of interpolation command 1 and 2 is about 10°, the break angle of interpolation command 2 and 3 is 32°, and the break angle of interpolation command 3 and 4 is 121°.

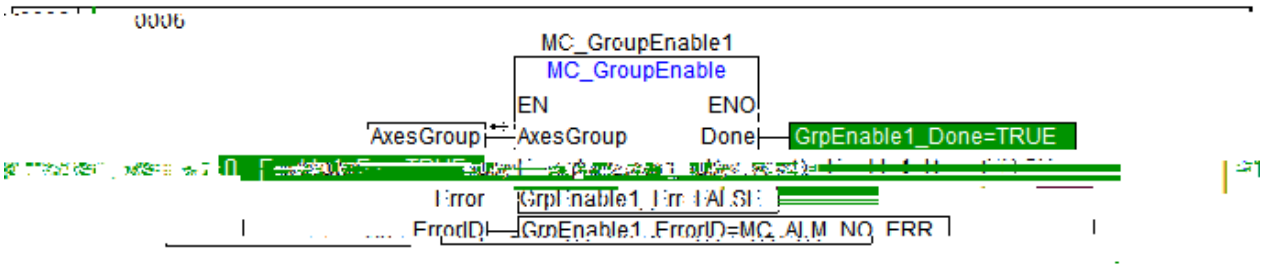
Run SMC_GroupCornerSpeedLimits to set the corner velocity limit parameters of the axis group. Set the deceleration angle DecAngle to 20° and the stop angle StopAngle to 60°.

LD diagram program

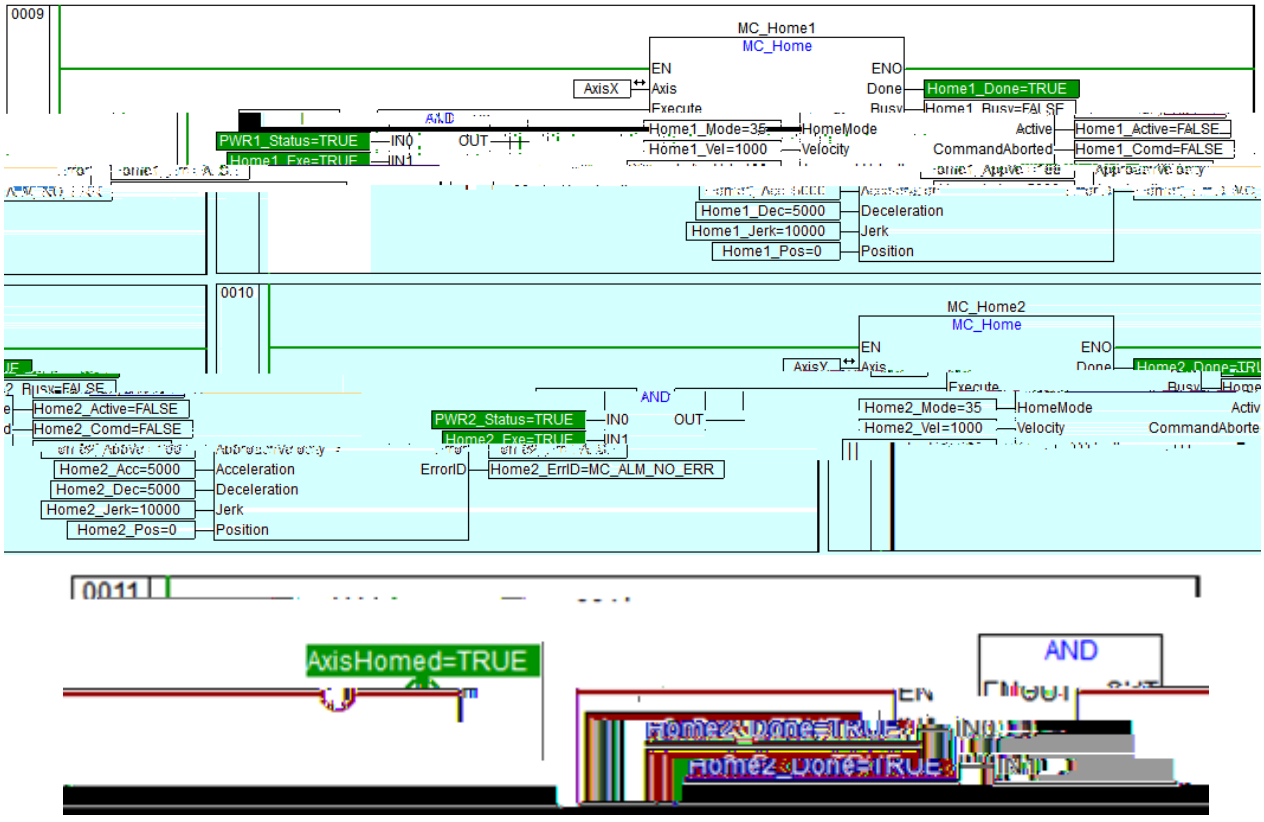
Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
AxesGroup	AXES_GROUP_REF	-	Axis group name
AxisI	AXIS_REF	-	Axis variables of combined axis in axis group
AxisX	AXIS_REF	-	Axis variables of sub-axis X in axis group

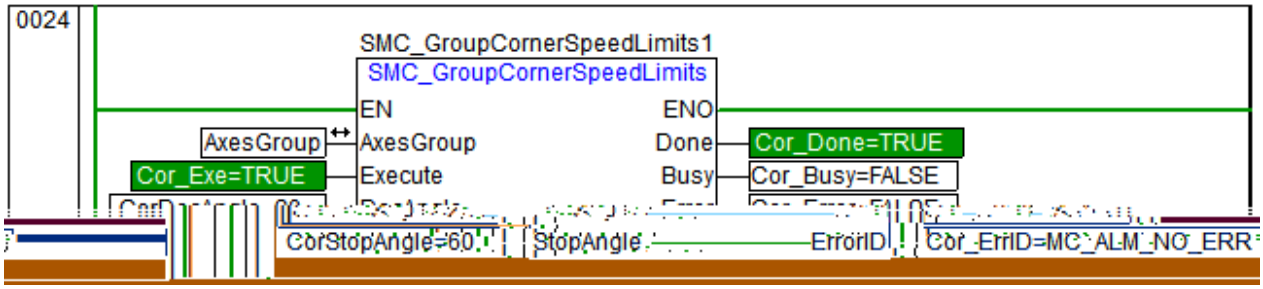
AxisY	AXIS_REF	-	Axis variables of sub-axis Y in axis group
StartProg	BOOL	FALSE	If this variable is TRUE, the axes in the axis group are ready to perform the axis enabling operation.



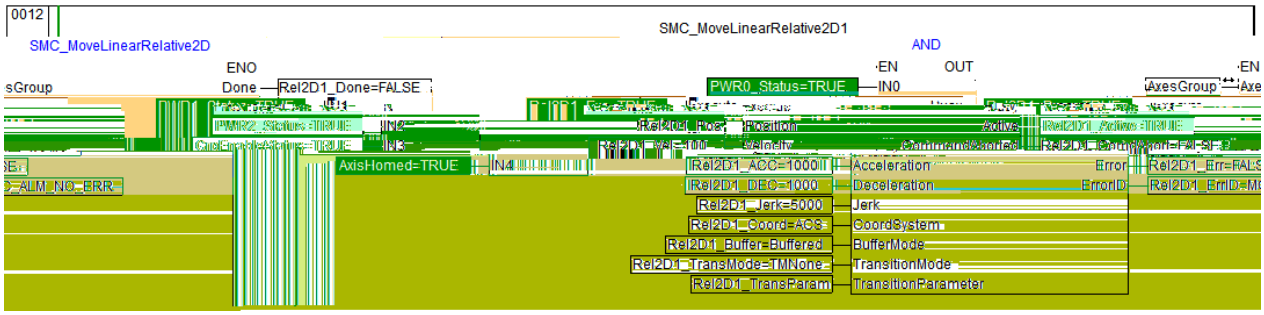
(5) After the two sub-axes in the axis group are enabled, if the home is not determined, perform homing action. When homing is complete, the homing completed variable AxisHomed is set to TRUE. See Section 0009-0011.

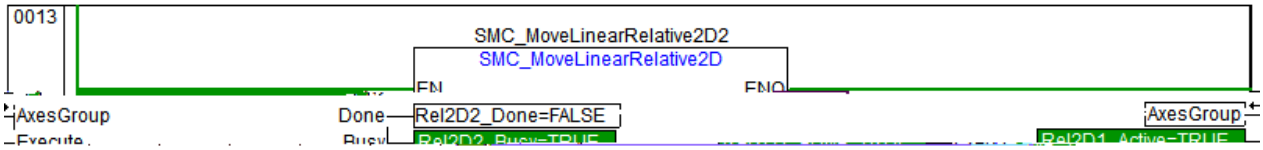


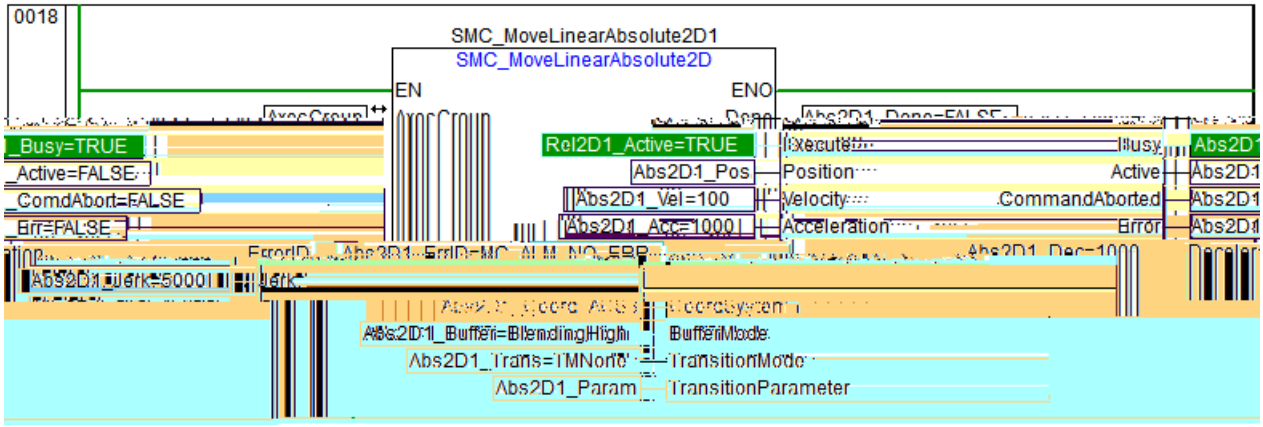
(6) In section 0024, execute the SMC_GroupCornerSpeedLimits command to set the corner velocity limit parameters.



(7) In section 0011, after the axis homing is completed, start executing the interpolation motion command. Execute command 1. When Rel2D1_Active of command 1 is TRUE, commands 2-4 are sent in sequence. See paragraphs 0012-0014 and 0018 below.







ST language program

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
AxesGroup	AXES_GROUP_REF	-	

```
    BufferMode := Rel2D1_Buffer,
    TransitionMode := Rel2D1_Trans,
    TransitionParameter := Rel2D1_Param,
    AxesGroup := AxesGroup,
    Done=> Rel2D1_Done,
    Busy=> Rel2D1_Busy,
    Active=> Rel2D1_Active,
    CommandAborted=> Rel2D1_ComdAbort,
    Error=> Rel2D1_Err,
    ErrorID=> Rel2D1_ErrID);
SMC_MoveLinearRelative2D2(*interpolation command 2*)
    Execute := Rel2D1_Active,
    Position := Rel2D2_Pos .5 3 /P <</M30ve,
```

```
    ErrorID=> Rel2D3_ErrID);
SMC_MoveLinearAbsolute2D1(*interpolation command 4*)
    Execute := Rel2D1_Active,
    Position := Abs2D1_Pos,
    Velocity := Abs2D1_Vel,
    Acceleration := Abs2D1_ACC,
    Deceleration := Abs2D1_DEC,
    Jerk := Abs2D1_Jerk,
    CoordSystem := Abs2D1_Coord,
    BufferMode := Abs2D1_Buffer,
    TransitionMode := Abs2D1_Trans,
    TransitionParameter := Abs2D1_Param,
    AxesGroup := AxesGroup,
    Done=> Abs2D1_Done,
    Busy=> Abs2D1_Busy,
    Active=> Abs2D1_Active,
    CommandAborted=> Abs2D1_ComdAbort,
    Error=> Abs2D1_Err,
    ErrorID=> Abs2D1_ErrID);
END_IF
```

The set corner deceleration angle is 20° and the stop angle is 60°. The rotation angles of interpolation command 1 and command 2 are less than the deceleration angle, and the velocity at the junction of the two commanded rotation angles (see connecting point 1 in the figure below) is not limited; The rotation angles of interpolation command 2 and command 3 are greater than the deceleration angle and less than the stop angle, and the speed at the junction of the two commanded rotation angles (see connecting point 2 in the figure below) is decelerated; The rotation angles of interpolation command 3 and command 4 are greater than the stop angle, and the speed at the junction of the two commanded rotation angles (see the connecting point 3 in the figure below) is reduced to 0;

The combined-axis speed curve of the interpolation motion command is as follows:

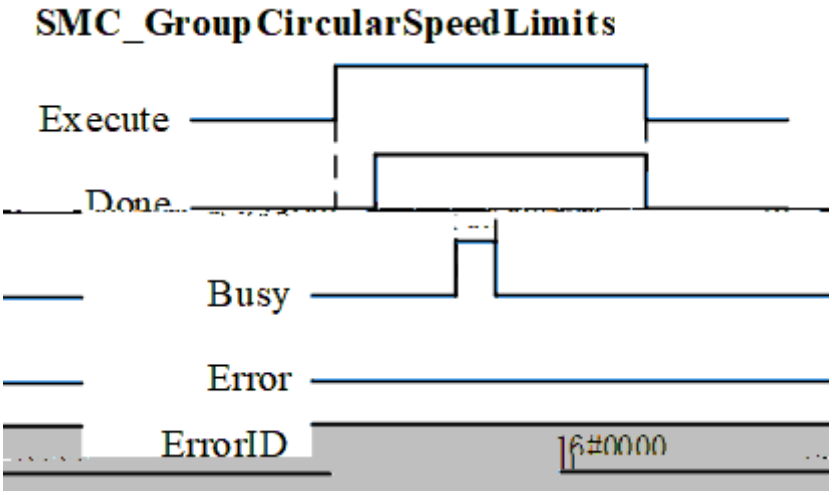
IN_OUT	AxesGroup	Axis group name	No	-	-	AXES_GROUP_REF
INPUT	Execute	Rising edge trigger	No	-	TRUE/FALSE	BOOL
	Option	Parameter setting mode: 0: Speed limit by setting the maximum impact value at arc turning; 1: The speed is limited by setting the maximum speed of the specified arc radius.	YES	0	0/1	USINT
	MaxImpulse	Valid when Option is 0: Maximum impact value, in the same unit as jerk	YES	1E100	Positive value	LREAL

Rvo2re W n

- When the parameter setting method Option is 1, recalculate MaxImpulse according to the maximum interpolation velocity Velocity and curvature of the specified arc radius. MaxImpulse =Curvature² * dSpeed³. Curvature=1/Radius.
- The speed limit parameters entered are all positive values.

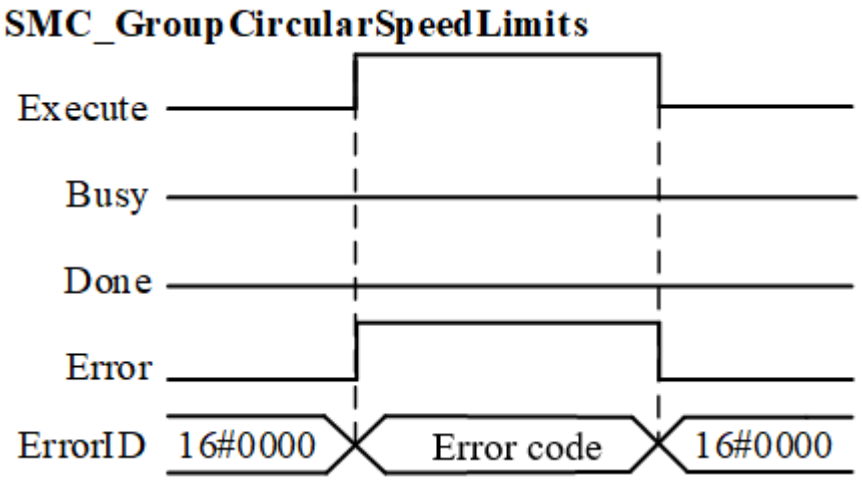
Timing diagram

(1) Normal timing diagram



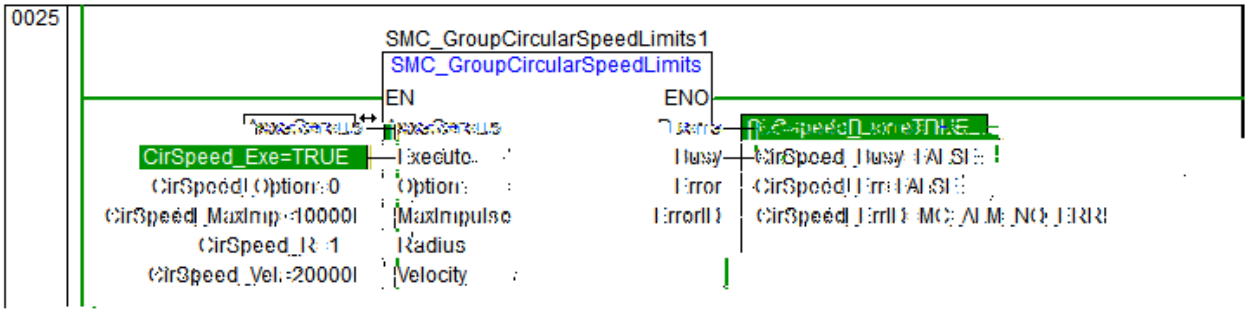
(2) Error pin timing

During the execution of this command, if there are error such as illegal parameter, 0: Unused (unused axis) for sub-axis and illegal ID of each axis in the axis group, the call of this command fails, with command output pin Error=True and error code ErrorID reporting an error value. Please refer to the appendix Error Code Description to obtain the cause of the error. When the command input pin Execute goes low, all output pins revert to their Default Values.

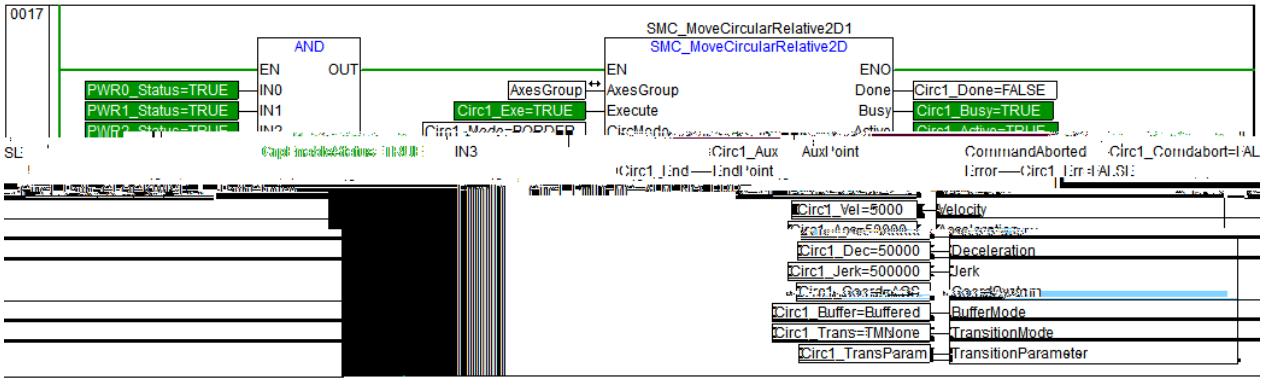


- When the abnormal situation of the command is solved, it is necessary to trigger Execute again on the rising edge and start the SMC_GroupCircularLimits functional block to complete the setting of arc

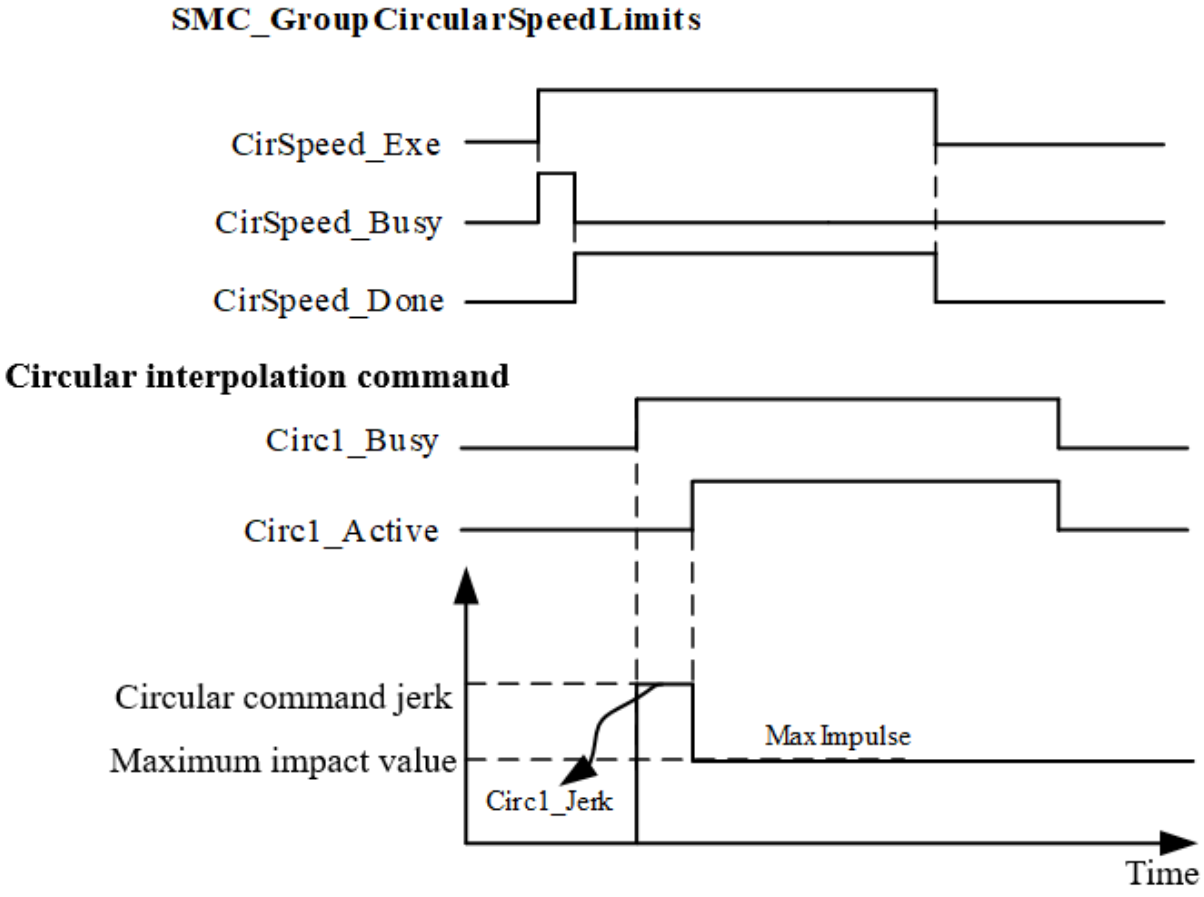
axis axis type as 50: EtherCAT_Position. For the usage of



(6) In section 0017, execute the circular interpolation command.



After the parameters are set, the jerk parameter of circular interpolation command is limited as follows:



ST language

Execute the SMC_GroupCircularSpeedLimits command, set the input parameter Option to 1, the arc radius to 1000, and the maximum interpolation velocity of the arc to 1000. Calculate MaxImpulse = (1/1000)2 * 10003 = 1000.

Execute the circular interpolation command SMC_MoveCircularRelative2D. In this command, the arc radius is 1000, the command speed is set to 5000, both acceleration and deceleration are 50000, and jerk is 500000. The speed and jerk of the axis group interpolation motion are limited. The speed of the axis combination axis is 1000, and the jerk is the limit value 1000 (MaxImpulse).

Description of Primary Variables

Variable Name	Variable Type	Initial Value	Notes
AxesGroup	AXES_GROUP_REF	-	Axis group name
AxisI	AXIS_REF	-	Axis variables of combined axis in axis group
AxisX	AXIS_REF	-	Axis variables of sub-axis X in axis group
AxisY	AXIS_REF	-	Axis variables of sub-axis Y in axis group
StartProg	BOOL	FALSE	If this variable is TRUE, the axes in the axis group are ready to perform the axis enabling operation.
Power0_Status	BOOL	FALSE	Variables whose sub-axis X enable is successful. When the sub-axis X is enabled successfully, this variable becomes TRUE.
Power1_Status	BOOL	FALSE	Variables whose sub-axis Y enable is successful. When the sub-axis Y is enabled successfully, this variable becomes TRUE.

Power2_Status	BOOL	FALSE	Variable for successful combination of axes enabling. This variable turns TRUE when the combined axis is enabled successfully.
InitFlag	BOOL	FALSE	If this variable is TRUE, the initialization parameter setting is completed.
GrpEnableStatus	BOOL	TRUE	

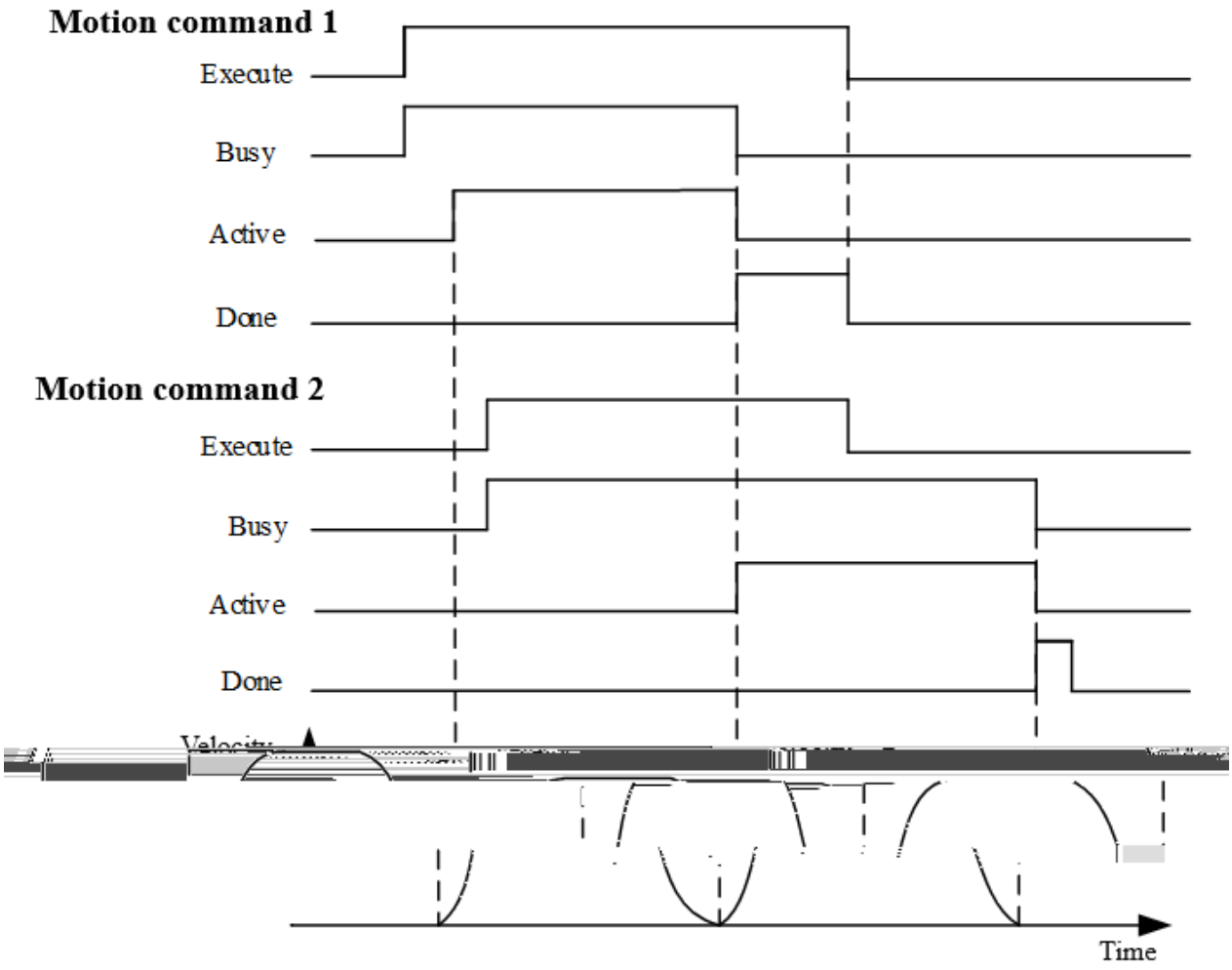
2203	MC_ERR_TRANSITION_MODE_INVALID	Illegal transition curve selection parameter
2204	MC_ERR_AXES_GROUP_ENABLED_FAILURE	axis group enable failed
2950	MC_ERR_PDO_UNMAPPED_FOR_STATUS_WORD	Bus driver PDO not mapped Statusword
2951	MC_ERR_PDO_UNMAPPED_FOR_ACTUAL_POS	Bus driver PDO is not mapped to Position actual value
2952	MC_ERR_PDO_UNMAPPED_FOR_TOUCH_PROBE_STATUS	Touch probe status not mapped by bus driver PDO
2953	MC_ERR_PDO_UNMAPPED_FOR_TOUCH_PROBE_POS_POS1	Touch probe pos1 pos value is not mapped by bus driver PDO
2954	MC_ERR_PDO_UNMAPPED_FOR_TOUCH_PROBE_NEG_POS1	Touch probe pos1 neg value is not mapped by bus driver PDO
2955	MC_ERR_PDO_UNMAPPED_FOR_TOUCH_PROBE_POS_POS2	Touch probe pos2 pos value is not mapped by bus driver PDO
2956	MC_ERR_PDO_UNMAPPED_FOR_TOUCH_PROBE_NEG_POS2	Touch probe pos2 neg value is not mapped by bus driver PDO
2957	MC_ERR_PDO_UNMAPPED_FOR_DIGITAL_INPU	

5001	MC_ERR_GEAR_MASTER_SOURCE_INVALID	Illegal master position source parameter
5040	MC_ERR_CAM_POINTER_TO_ARRARYPOS_ISNULL	Pointer to cam position array is null
5041	MC_ERR_CAM_TABLE_POSNUM_INVALID	Illegal number of positions in cam position array (less than 3)
5042	MC_ERR_CAM_MASTER_SOURCE_INVALID	Illegal electronic camshaft position source parameter
5043	MC_ERR_CAM_SLAVESCALE_INVALID	Electronic camshaft slave shaft position data scale factor illegal
5044	MC_ERR_CAM_DI_STARTPOSITION_INVALID	Electronic cam DI triggering starting position illegal
5045	MC_ERR_CAM_START_MODE_INVALID	Electronic cam start mode error
5046	MC_ERR_CAM_SLAVE_STANDIN_POSITION_LIMIT1	The slave shaft is in the limit status when the electronic cam starts

	NT	group are different
5153	MC_ERR_GANTRY_AXIS_TYPE_SET_ERR	Gantry synchronizing shaft type error

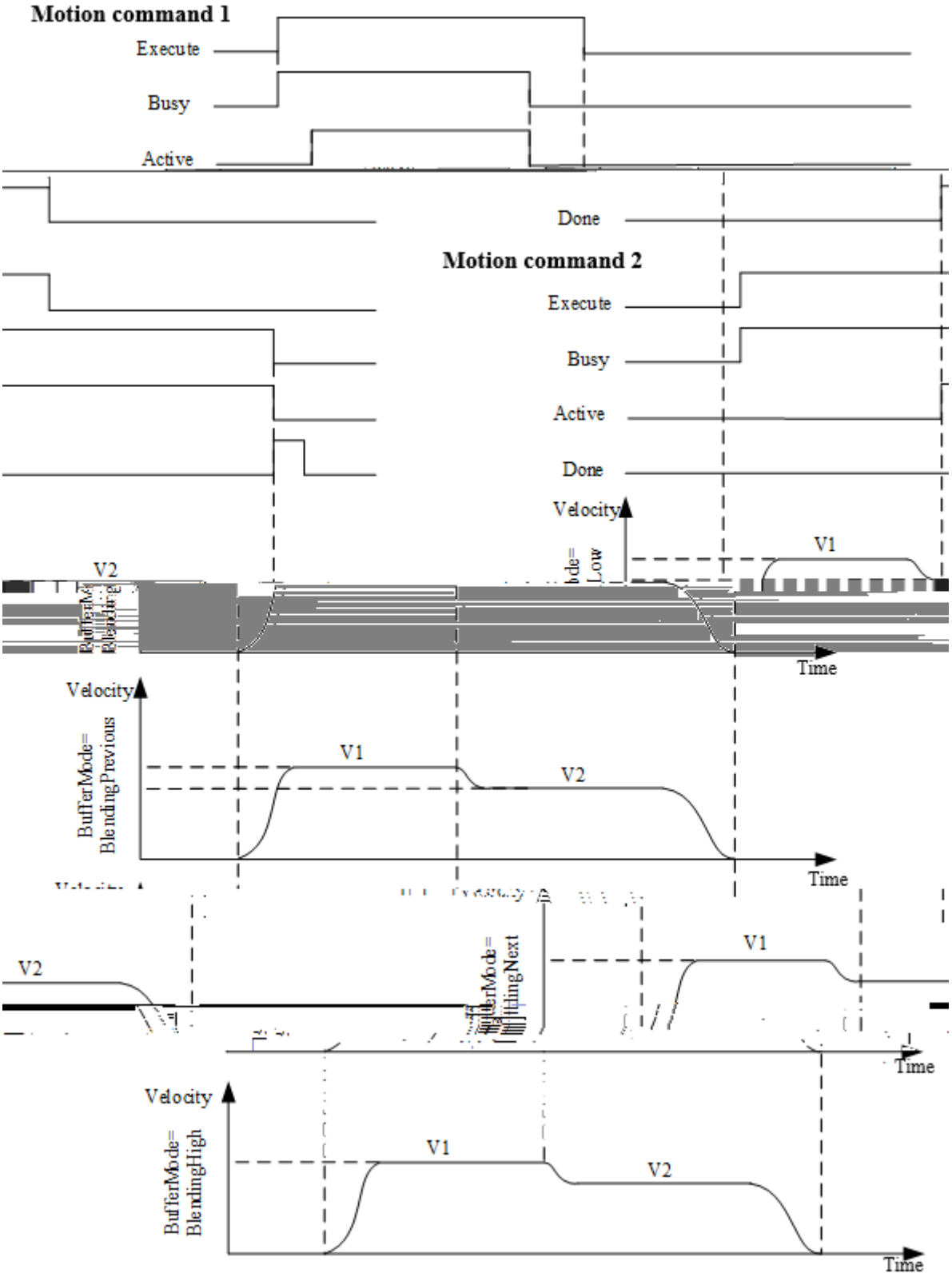
5154

- S2: starting position of the next motion command
- E2: Target position of the next motion command



8.2.1.3 Fusion

The previous and next commands are speed fused in the set fusion mode, and the specific speed fusion mode is determined by the fusion mode.


Motion command 1
Execute
Busy
Active
Done
Motion command 2
Execute
Busy
Active
Done
Velocity
V1
V2
Time
Velocity
BufferMode=
BlendingPrevious
V1
V2
Time
V2
V1
Time
Velocity
BufferMode=
BlendingHigh
V1
V2
Time