M-Series Controller

Motion Control

Instruction Manual

January 2024 V1.0
Version：ATC/MMXC022310

Contents

Preface

Thank you very much for purchasing the M Series Motion Controller. This manual introduces motion controller motion instructions, such as run/unrun operation instructions, speed instructions, relative displacement commands, electronic gears, electronic cams, and axis groups.

Intended audience

This manual is intended for technicians who program and debug the M200 series, M311, M312, M500S series, and M500 series motion controllers. Readers need to have some basic knowledge and programming mindset related to programmable controllers.

Manual revision information

Version	Date	Content
V1.00	2023/09/07	First edition

Other information

- The content of this manual is edited based on product information and customer requirements, users who have questions or find errors in the contents of the manual are welcome to call HCFA or send an email to 400@hcfa.cn and follow the version number marked on the front cover to assist in the clarification.
- The contents of this manual, including text, pictures, logos, forms, etc., may not be reproduced or disseminated in any form without the authorization of the company. Otherwise, the company shall pursue the violator's legal responsibility in accordance with the law.

Chapter1 Motion control instructions basics

1.1 Motion control instructions composition

The motion instructions in the ladder diagram are shown below. The motion control instructions consist of instantiation names, instruction names, input variables, and output variables. The instantiation name is used to assign memory to the corresponding instructions, and when used, a different instantiation name is assigned to each instruction.

Input parameters are used to set the values needed for motion control instructions, such as target position, target speed, etc., to the input variables. When an input parameter is omitted, the value of the input variable is the initial value. Input parameters can be omitted for reserved input variables or input variables that have no effect on the execution of the instruction during use.

Output variables are used to output the state of the instruction, and the value of the output variable is passed to the output parameter when an output variable is available.

Input and output variables can be used by instantiating the name, as shown below, by assigning a value to the input variable Position via the input parameter Pos, or by assigning the value of the output parameter Done to the output parameter Abs_Done.

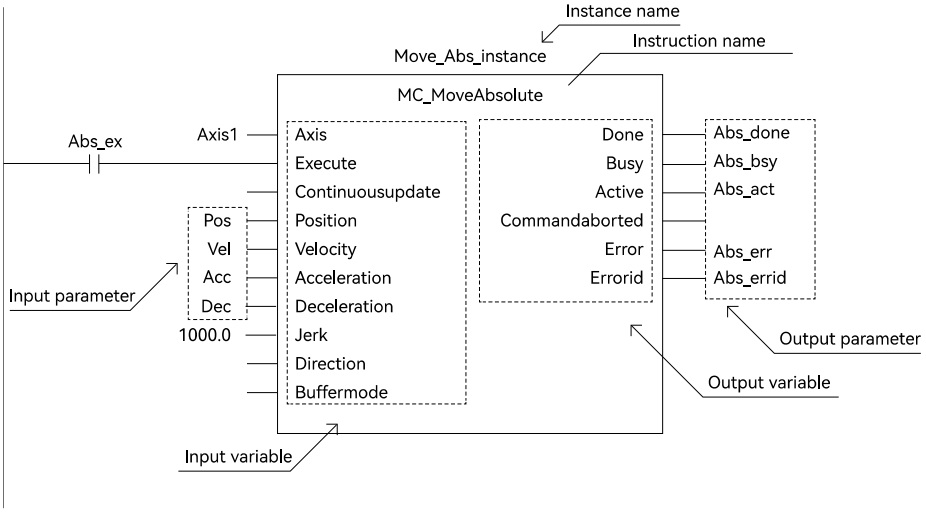


Figure 1-1 Motion control instructions composition diagram

1.2 Supported languages for motion control instructions

Motion control instructions can be edited in the software in both Ladder Diagram (LD) and Structured Text (ST) programming languages.

- Ladder diagram（LD）
- Structured text（ST）

The following figure shows an example of the MC_MoveAbsolute instruction edited in the LD.

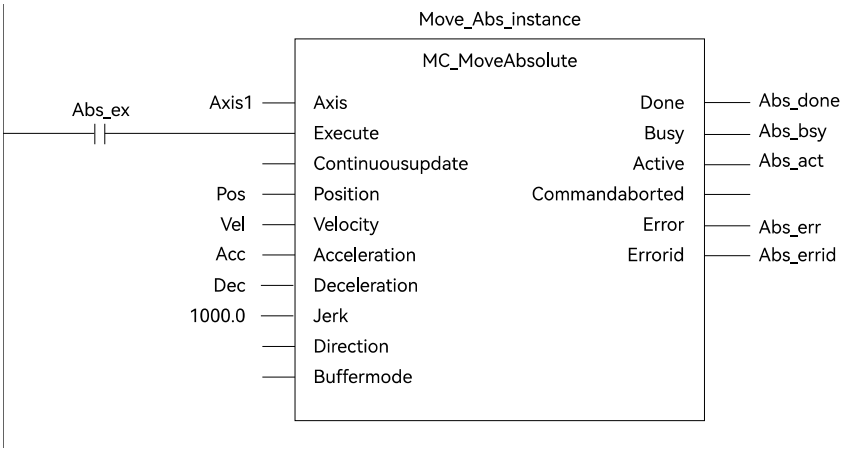


Figure 1-2 Example diagram of MC_MoveAbsolute instruction in LD

The following is an example of a MC_MoveAbsolute (absolute displacement) instruction edited in the ST language: The symbol between an input variable and an input parameter is: =, and the symbol between an output variable and an output parameter is =>.

```
Move_Abs_instance(
 Axis:=axis1 ,
 Execute:= abs_ex,
 Position:=pos ,
 Velocity:=vel ,
 Acceleration:= acc,
 Deceleration:= dec,
 Jerk:=1000.0 ,
 Done=>abs_done ,
 Busy=> abs_bsy,
 Active=> abs_act,
 Error=> abs_err,
 ErrorID=>abs_errID );
```

1.3 Execution of motion control instructions

Motion control instructions can be executed only when added to an event-triggered (motion event) task, and cannot be executed properly when added to other tasks:

Task type		Whether or not to execute
Event triggering	Motion event	Yes
	Non-motion event	No
Fixed period		No
Freewheeling		No

1.4 Parameter unit of motion control instructions

The units of the Position and Distance parameters in the motion control instructions are the same as the units of the "working travel per revolution" of the mechanism set in the software.

The settings in the software are located in "Motion Control" → "Axis Settings" → "Basic Settings".

For the mechanism shown in the figure below, one revolution of the motor corresponds to a 10 mm working travel of the screw. Set up in the software according to the figure below, when the screw is controlled to run 10 mm by the MC_MoveRelative (relative displacement) instruction, the input variable Position (absolute position) value of the instruction is in millimeters, Velocity (target velocity) is in millimeters per second, Acceleration and Deceleration in millimeters per second2 and Jerk in millimeters per second3.

The parameter units of motion control instructions, such as Distance, Velocity, Acceleration, Deceleration, Jerk, etc. are related to the actual mechanism and software settings, and these position-related units are called travel units.

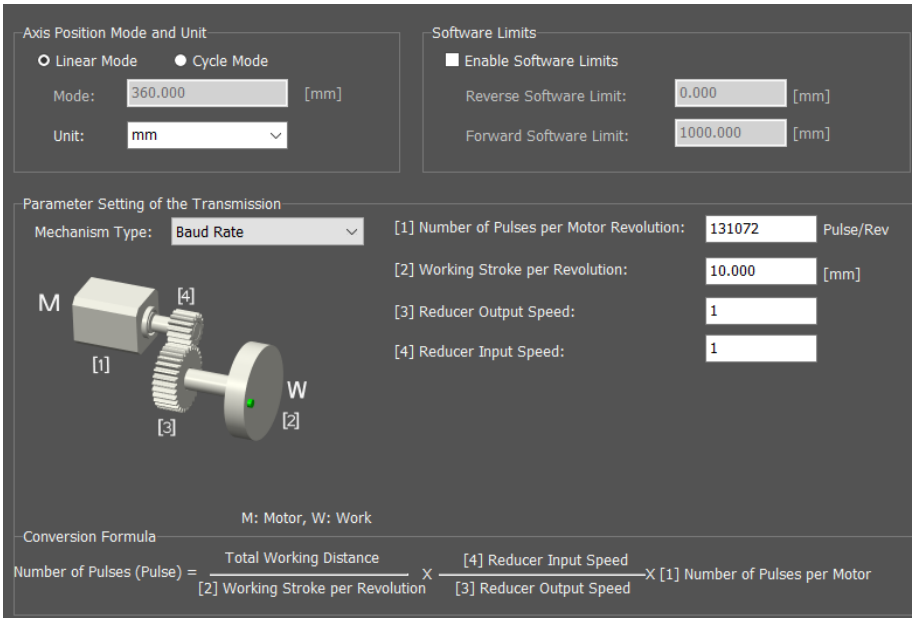


Figure 1-3 Axis Setting Parameter Diagram

The following table shows the MC_MoveRelative instruction's parameter units after the axis has been set via the "Basic Settings".

Name	Meaning	Unit
Distance	Travel distance	millimeter
Velocity	Target velocity	mm/sec
Acceleration	Acceleration	mm/sec^2
Deceleration	Deceleration	mm/sec^2
Jerk	Jerk	mm/sec^3

1.5 Parameter descriptions related to motion control instructions

The speed control of the controller adopts an S-type speed curve, which can effectively reduce the impact of motion control. The parameter values of Velocity, Acceleration, Deceleration, and Jerk need to be set in the motion control instructions when the S-type speed curve is operating.

The abbreviations for the corresponding position, velocity, acceleration, and jerk during axis motion are shown in the table below.

Position	Velocity	Acceleration/Deceleration	Jerk
p	v	a	j

The relations among the corresponding positions, velocities, accelerations, and jerk degrees during the axis motion are as follows:

When the acceleration does not reach the set acceleration, a = jt; when the acceleration reaches the set acceleration, a is the value of the set acceleration;

When acceleration is greater than 0 and acceleration decreases, a = set acceleration + jt × -1.

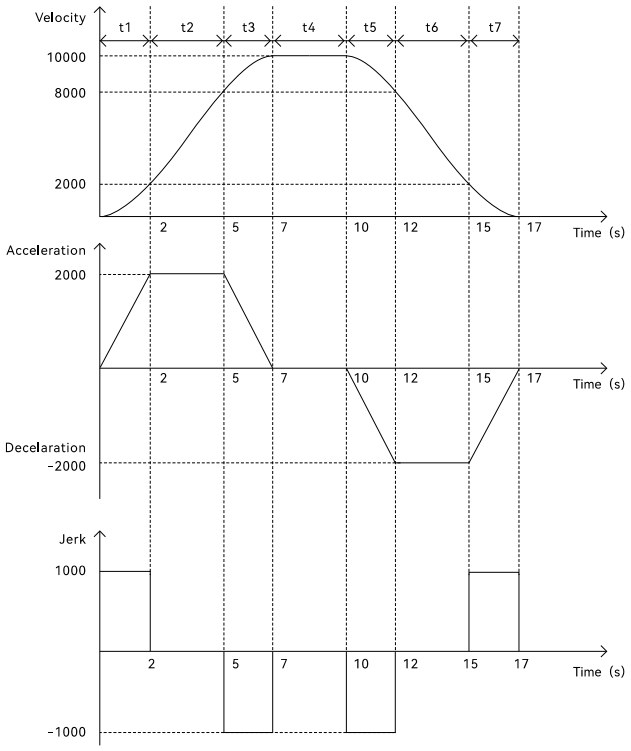
The derivative of velocity with respect to time is the acceleration: a = dv/dt.

The derivative of position with respect to time is velocity: v=dp/dt.

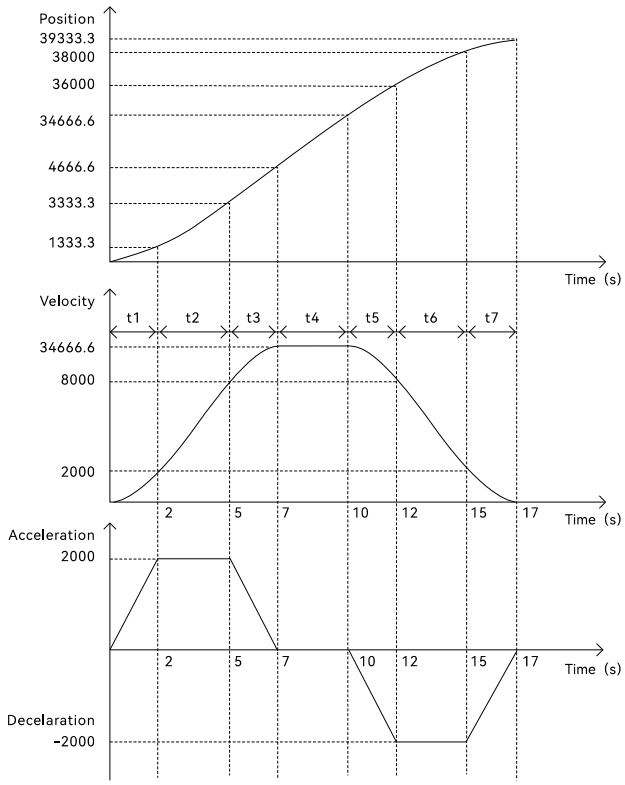
The following figure shows the correspondence between position, velocity, acceleration, jerk and time when the

current position of the axis is 0 and the MC_MoveRelative instruction Velocity=10000, Acceleration=2000, Deceleration=2000, Jerk=1000.

Correspondence among velocity, acceleration/deceleration and jerk



Correspondence among position, velocity and acceleration/deceleration



Phase	Name	Time (second)	Jerk (j)	Acceleration/Deceleration (a)	Velocity (v)	Position (p)
t1	Increasing acceleration phase	2	1000	0~2000	0~2000	0~1333.3
t2	Constant acceleration phase	3	0	2000	2000~8000	1333.3~3333.3
t3	Decreasing acceleration phase	2	-1000	2000~0	8000~10000	3333.3~4666.6
t4	Constant velocity phase	3	0	0	10000	4666.6~34666.6
t5	Increasing deceleration phase	2	-1000	0~-2000	10000~8000	34666.6~36000
t6	Constant deceleration phase	3	0	-2000	8000~2000	36000~38000
t7	Decreasing deceleration phase	2	1000	2000~0	2000~0	38000~39333.3

As shown in the above figure and the above table, the jerk, acceleration, velocity, and position of the stages t1~t4 are calculated as shown in the table below.

Phase	Name	Time (second)	Jerk (j)	Acceleration/Deceleration (a)	Velocity (v)
t1	2	1000	j×t1	$1/2\times j\times t1^2=2000$	$1/6\times j\times t1^3$
t2	3	0	2000	$2000+a\times t2=8000$	$1/6\times j\times t1^3+1/2\times a\times t2^2$
t3	2	-1000	$2000+j\times t3\times -1$	$8000+1/2\times j\times t3^2$	$1/6\times j\times t1^3+1/2\times a\times t2^2+1/6\times j\times t3^3$
t4	3	0	0	10000	$1/6\times j\times t1^3+1/2\times a\times t2^2+1/6\times j\times t3^3+10000\times t4$

Chapter2 Variable

2.1 Axis variable

Axis variables are system-defined, array-type reference global variables, which is named as Axis. Different axes can be accessed by Axis[corresponding axis number]. When used in the project, the variables of the axis variable members are used directly without declaration... When getting the actual position of axis 1, it can be accessed through Axis[1].ActPos. The detailed description of the axis variable is shown in the following table.

Axis variable member		Data type	Meaning	Unit
Axis[corresponding axis number]	CmdPos	LREAL	Axis Instruction Position	Travel unit
	CmdVel	LREAL	Axis Instruction Velocity	Travel unit/second
	CmdAcc	LREAL	Axis instruction acceleration	Travel unit/sec2
	CmdTrq	INT	Axis Instruction Torque	[0.1%]
	ActPos	LREAL	Axis actual position	Travel unit
	ActVel	LREAL	Axis actual velocity	Travel unit/second
	ActTrq	INT	Axis actual torque	[0.1%]
	AxState	USINT	Axis state (Refer to "Meaning of the values corresponding to the axis state in the axis variables" below.)	
	ErrPos	LREAL	Difference between the axis instruction position and the actual position in the same period	Travel unit
	AxErrCode	UINT	Axis error code	
	AxType	USINT	Axis position type 1: Linear mode 0: Periodic mode	
	AxModulo	LREAL	Mode (valid when the axis position mode is periodic mode)	
	ScaleDen	LREAL	Working gear reduction ratio denominator	
	ScaleNum	LREAL	Working gear reduction ratio numerator	
	UnitPerTunr	LREAL	Working travel per revolution of work	
	VelFactor	LREAL	Multiple of target speed	
	EnableSoftLimit	BOOL	Software limit activated or not 1: activated 0: not activated	
	PositivePosLimit	LREAL	Positive software limit	Travel unit
	NegtivePosLimit	LREAL	Negative software limit	Travel unit

Meaning of the corresponding value of the axis state in the axis variable

Axis variable member	Corresponding value	Meaning of corresponding value
AxState	0	Disabled (disable operation)
	1	Standstill（stop）
	2	ErrorStop（stop error）
	3	Stopping（stop decreasing）
	4	Homing（origin searching）
	5	DiscreteMotion（Positioning）
	10	Rotary cutting status
	11	Electronic gear is engaging in
	12	Electronic cam is engaging in
	14	Torque status

Axis variable member	Corresponding value	Meaning of corresponding value
AxState	15	ContinuousMotion
	16	Electronic cam synchronization
	17	Electronic Gear Synchronization
	18	Electronic gear synchronization of two spin

Chapter3 Related to Motion Control

3.1 Axis state machine

The specified axis executes different motion instructions and the status of the corresponding axis is shown below. The status of the corresponding axis can be checked through the MC_ReadStatus instruction or the axis state variable AxState in the axis structure.

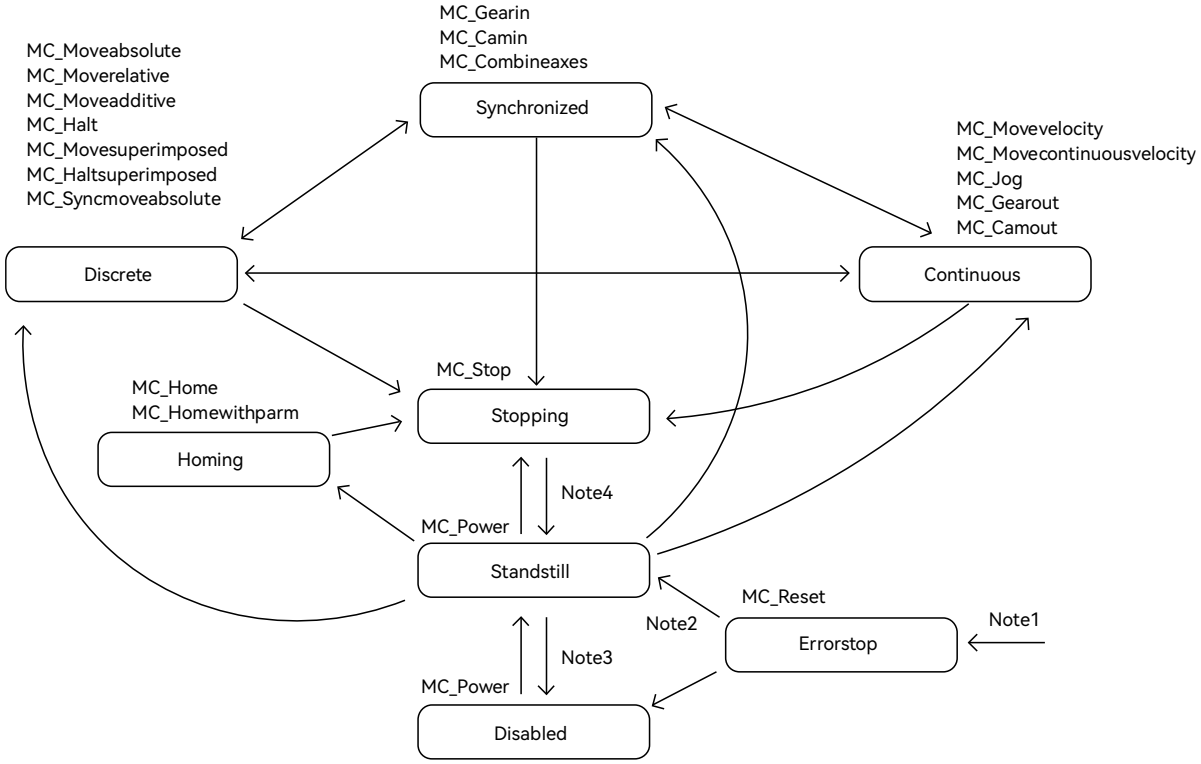


Figure 3-1 Axis state transfer diagram

Note 1: When an abnormality occurs during axis operation or the status word error bit is TRUE, the axis state enters the ErrorStop state (error stop). If the error bit of the status word of the axis is TRUE, regardless of whether the axis is in the enable state or not, the axis state enters the ErrorStop state (error stop). During the axis is operating, the axis state enters the ErrorStop (error stop) state when the axis is deactivated by the MC_Power instruction.

Note 2: After MC_Reset is executed and the axis fault is reset, when the Enable of MC_Power instruction is TRUE, the axis state changes from ErrorStop to StandStill; when the Enable of MC_Power instruction is FALSE, the axis state changes from ErrorStop to Disabled. Stop) status to Disabled status; when Enable of MC_Power instruction is FALSE, the axis state changes from ErrorStop to StandStill status.

Note 3: When the Enable of MC_Power instruction is TRUE and the Status bit is TRUE, the axis state becomes StandStill status. When the axis state is StandStill, all motion instructions can be executed.

Note 4: When Done is TRUE and Execute is FALSE in the MC_Stop instruction, the axis state changes from Stopping to StandStill.

Note 5: When MC_MoveSuperImposed and MC_HaltSuperImposed instructions are executed separately, the axis state is Discrete (positioning).When MC_MoveSuperImposed and MC_HaltSuperImposed instructions are executed simultaneously with other instructions, the axis state is the axis state corresponding to the execution of other instructions.

3.2 Buffer mode during multi-starting of the same axis

When the current motion control instruction controls an axis, another motion control instruction (buffer instruction) is started for the same axis, which is called multi-starting. The speed transition between the buffer instruction and the current instruction is determined by the BufferMode value, which is called the buffer mode. The buffer mode can be used to realize that when two instructions are executed, the speed of the axis is not reduced to zero, which improves the work efficiency, such as in the case of multi-stage speed. The meanings of the terms related to multiple starting are shown in the following table.

Name	Meaning
Current instruction	A motion control instruction that is currently controlling the axis
Buffer Instruction	A motion control instruction that is started for the same axis and is in the wait state when the axis is controlled by other instructions.
Buffer Mode	The mode of the buffer instruction control axis when the current instruction target position is reached
Buffer Speed	The speed used when the current instruction target position is reached

When buffer mode selection is aborted, the buffered instruction is executed immediately without any buffer. When another mode is selected for buffer mode, the buffer instruction can only have one level of buffer. If another instruction is started for the buffer after the buffer instruction is started for the current instruction execution, an error will be reported during execution.

With the exception of mcAborting mode, buffer instructions should be started as far in advance as possible; if the buffer instruction is started when the current instruction is about to complete, the buffer speed cannot be reached. It is recommended that the output variable Active (in control) of the current instruction be used to trigger the start of the buffer instruction.

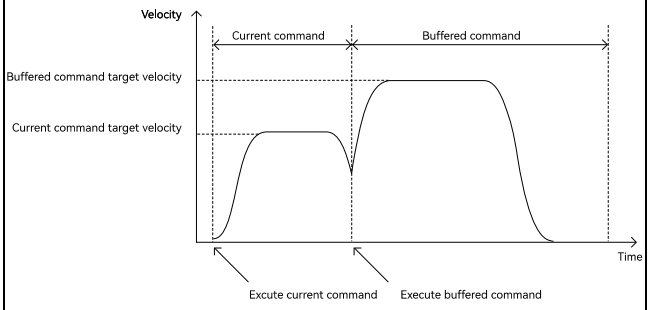
The meanings of the different buffer modes are shown in the following table.

BufferMode		Meaning
mcAborting (abortion)		Aborts the instruction that is currently controlling the axis and switches to buffer instruction control
mcBuffered (buffer)		Wait for the current instruction execution to complete (e.g., target position or target speed is reached) and switch to the buffer instruction control axis.
Buffer		Switches to the buffer instruction control axis when the current instruction target position is reached. The target speed at the time of the buffer instruction switches to the speed selected through the buffer mode. The axis speed does not stop when the current instruction and buffer are switched.
	mcBlendingLow (low-speed buffer)	Compare the target speed of the current instruction with the target speed of the instruction that has been buffered. When the target position of the current instruction is reached, the lower speed is used as the buffer speed.
	mcBlendingPrevious (buffer at the target speed of the current instruction)	When the target position of the current instruction is reached, the target speed of the current instruction is used as the buffer speed.
	mcBlendingNext (buffer at the target speed of the buffered instruction)	When the target position of the current instruction is reached, the target speed of the buffer instruction is used as the buffer speed.
	mcBlendingHigh (high-speed buffer)	Compare the target speed of the current instruction with the target speed of the instruction that has been buffered. When the target position of the current instruction is reached, the higher speed is used as the buffer speed.

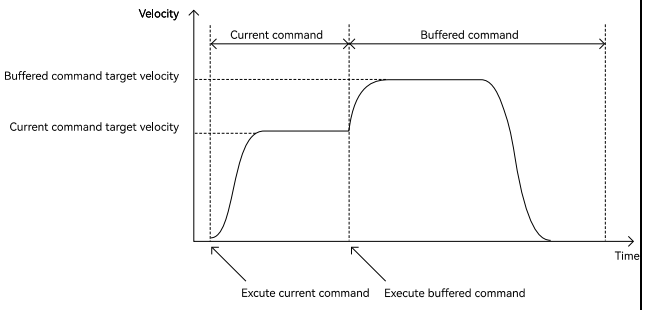
■ mcAborting (abortion)

Default mode for buffer mode, multiple starting with a buffer instruction to control the axis immediately, no buffer.

Buffer instruction aborts the current instruction when the current instruction decelerate



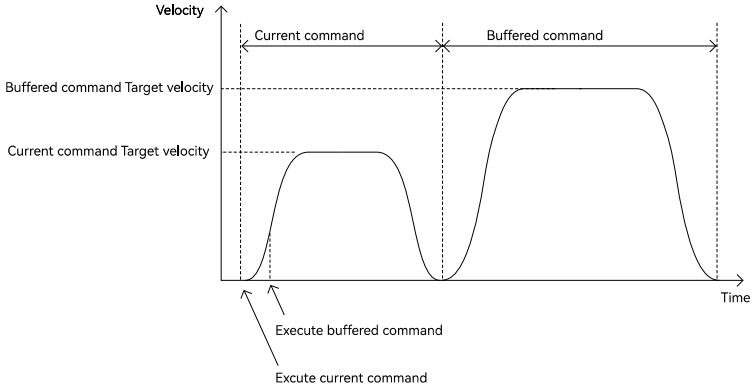
Buffer instruction aborts the current instruction when the current instruction is evenly spaced



■ mcBuffered (buffered)

Wait for the currently executed instruction to complete (e.g., target position or target speed is reached) and switch to the buffer instruction control axis.

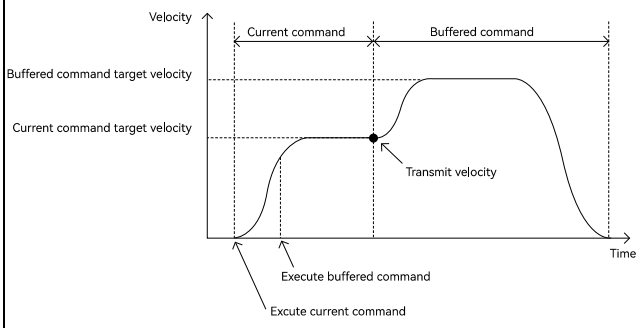
The buffer instruction is executed when the current instruction is accelerated, and the buffer instruction waits until the target position of the current instruction is reached and the velocity decreases to zero.



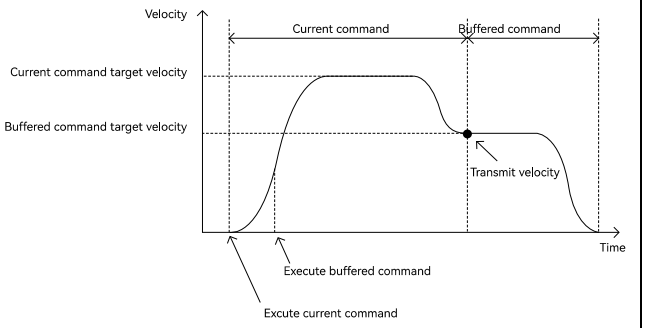
■ mcBlendingLow (low-speed buffer)

Compare the target speed of the current instruction with the target speed of the instruction that has been buffered. When the target position of the current instruction is reached, the lower speed is used as the buffer speed.

When the current instruction target speed is low and the buffer instruction target speed is high



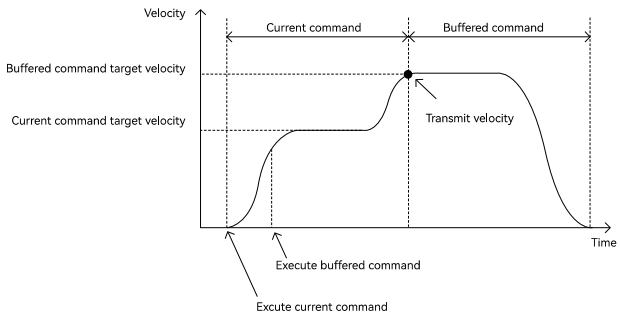
When the current instruction target speed is high and the buffer instruction target speed is low



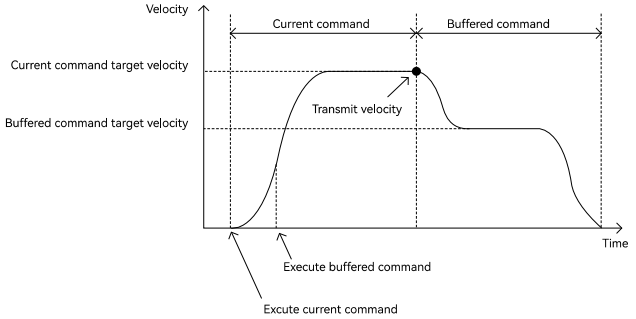
■ mcBlendingHigh (high-speed buffer)

Compare the target speed of the current instruction with the target speed of the instruction that has been buffered. When the target position of the current instruction is reached, the higher speed is used as the buffer speed.

When the current instruction target speed is low and the buffer instruction target speed is high



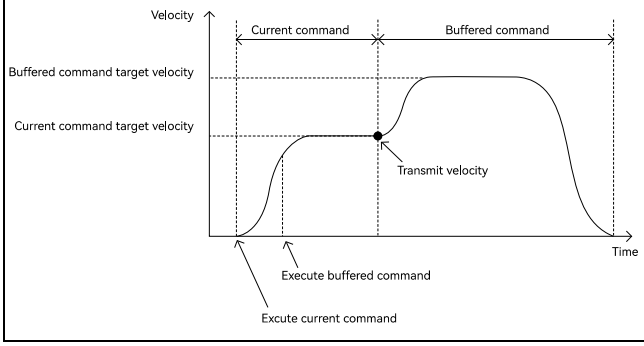
When the current instruction target speed is high and the buffer instruction target speed is low



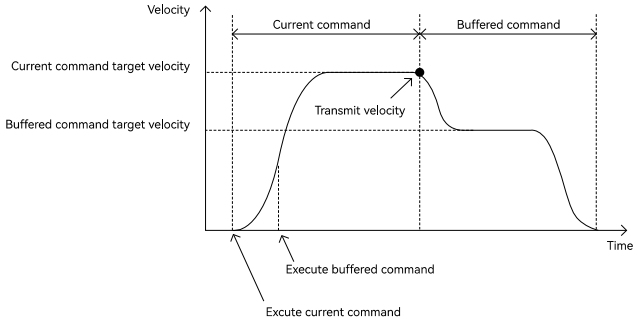
■ mcBlendingPrevious (buffer at the target speed of the current instruction)

When the target position of the current instruction is reached, the target speed of the current instruction is used as the buffer speed.

When the current instruction target speed is low and the buffer instruction target speed is high



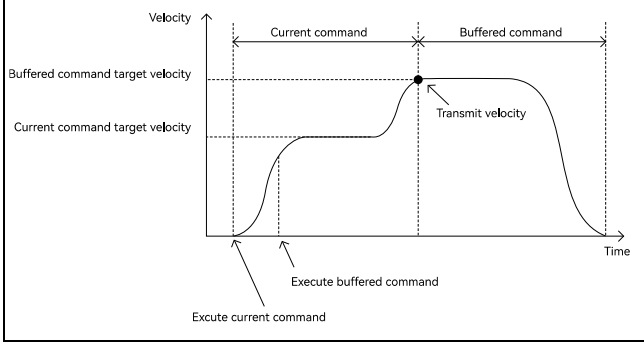
When the current instruction target speed is high and the buffer instruction target speed is low



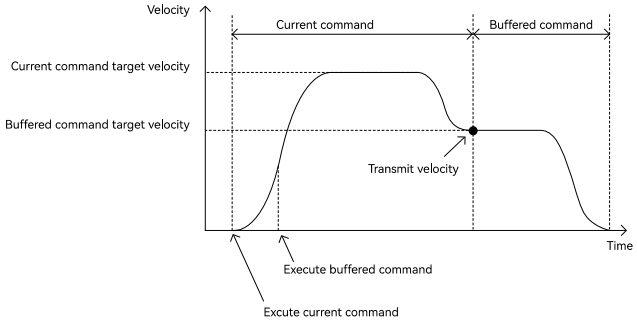
■ mcBlendingNext (buffer at the target speed of the buffered instruction)

The target velocity of the buffer instruction is used as the buffer velocity when the target position of the current instruction is reached.

When the current instruction target speed is low and the buffer instruction target speed is high.



When the current instruction target speed is high and the buffer instruction target speed is low.



Introduction to the interpolation instruction BufferMode

BufferMode value	TransitionMode value	Description
1：mcBuffered（buffer）	0：mcTMNone(no transition curve)	Wait for the execution of the current interpolation instruction to complete (the position is reached and the velocity is reduced to 0) and execute the buffer interpolation instruction.
3：mcBlending-Previous（buffer at the target speed of the current instruction）	2：mcTMConstantVelocity(transition at a set speed)	Wait for the execution of the current interpolation instruction to complete and execute the buffer interpolation instruction with a handover speed of the synthesis speed of the current interpolation instruction.
	3：mcTMCornerDistance(transition with additional angle)	If the current interpolation instruction is completed and the speed is not reduced to 0, the buffer interpolation instruction is executed, and there is a circular transition between the end position of the current interpolation instruction and the start position of the buffer interpolation instruction, and the interpolation speed is not reduced to 0 during the transition.

■ Supported buffer modes for each instruction

When buffer instruction and current instruction are buffered, the buffer mode is set through the pin of BufferMode, and the value that can be set for BufferMode of the buffer instruction is related to both the current instruction and the buffer instruction.

- Case 1: If the MC_MoveAbsoluteis the current instruction and the MC_MoveVelocity instruction is a buffer instruction, the BufferMode pin of the MC_MoveVelocity instruction can be set with the buffer modes mcAborting, mcBuffered, mcBlendingLow, mcBlendingPrevious, mcBlendingNext, and mcBlendingHigh.
- Case 2: If the MC_MoveVelocity instruction is the current instruction and the MC_Absolute instruction is a buffer instruction, the buffer modes that can be set by the BufferMode pin of the MC_MoveRelative instruction are mcAborting and mcBuffered The BufferMode pin of the MC_MoveRelative instruction can be set to mcAborting and mcBuffered.

The list of BufferMode that can be selected for the buffer instruction according to the current instruction is as follows.

Current instruction	BufferMode available for the buffer instruction
MC_MoveAbsolute	【mcAborting, mcBuffered, mcBlendingLow, mcBlendingPrevious, mcBlendingNext, mcBlendingHigh】*1
MC_MoveRelative	【mcAborting, mcBuffered, mcBlendingLow, mcBlendingPrevious, mcBlendingNext, mcBlendingHigh】*1
MC_MoveAdditive	【mcAborting, mcBuffered, mcBlendingLow, mcBlendingPrevious, mcBlendingNext, mcBlendingHigh】*1
MC_MoveSuperimposed	mcAborting
MC_HaltSuperimposed	mcAborting
MC_MoveVelocity	mcAborting, mcBuffered
MC_Home	Only the MC_Stop instruction can abort the MC_Home instruction
MC_Halt	mcAborting, mcBuffered
MC_ GearIn	mcAborting, mcBuffered
MC_ GearOut	mcAborting, mcBuffered
MC_CombineAxes	mcAborting, mcBuffered
MC_ CamIn	mcAborting, mcBuffered
MC_ CamOut	mcAborting, mcBuffered

*1: When the buffer instruction is MC_GearIn, MC_CamIn, or MC_CombineAxes, only mcAborting and mcBuffered can be selected for BufferMode.

The completion status of the current instruction is determined by the completion bits of each instruction. A completion bit of TRUE indicates that the execution of the instruction is completed and the Buffer instruction is started. The following table notes the completion bits of each instruction to facilitate judgment when selecting the BufferMode mode.

Instruction	Whether or not to buffer the instruction	Whether or not a buffer instruction can be followed	Completion bit
MC_MoveAbsolute	Yes	Yes	Done
MC_MoveRelative	Yes	Yes	Done
MC_MoveAdditive	Yes	Yes	Done
MC_Home	No	No	Done
MC_Stop	No	No	Done
MC_Halt	Yes	Yes	Done
MC_MoveSuperimposed	No	No	——
MC_HaltSuperimposed	No	No	——
MC_MoveVelocity	Yes	Yes	InVelocity
MC_CamIn	Yes	Yes	EndOfProfile
MC_CamOut	No	Yes	Done
MC_GearIn	Yes	Yes	InGear
MC_GearOut	No	Yes	Done
MC_CombineAxes	Yes	Yes	InSync

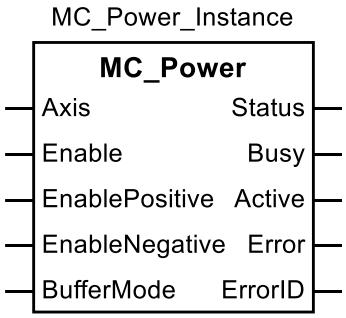
3.3 Supported axis specifications and motion control functions for different models

Model name	Supported axis number range and number of single-axis							Electronic cam numbering range	Cam point numbering range	Rotary cut numbering range	Supported maximum number of axis group	CNC function
	Axis number range	Number of all axes	Number of virtual encoder axis	Number of servo axis	Number of pulse axis	Number of encoder axis	Number of virtual encoder axis					
HCM511S-32MT4-D	1~16	16	16	8	4	2	16	1~16	1~32	1~2	1	Not supported
HCM501S-16MT4-D	1~16	16	16	8	4	2	16	1~16	1~32	1~2	1	Not supported
HCM301-16MT4-D	1~16	16	16	Not supported	4	2	16	Not supported	Not supported	Not supported	Not supported	Not supported
HCM302-16MT4-D	1~16	16	16	Not Supported	4	2	16	1~16	1~32	1~2	1	Not supported
HCM310-16MT4-D	1~16	16	16	Not Supported	4	2	16	Not supported	Not supported	Not supported	Not supported	Not supported
HCM311-16MT4-D	1~16	16	16	Not Supported	4	2	16	1~16	1~32	1~2	1	Not supported
HCM312-32MT6-D	1~16	16	16	Not Supported	6	2	16	1~16	1~32	1~2	1	Not supported

Chapter4 Single-axis Instructions

4.1 MC_Power (enable/disable operation)

The specified axis is used for controlling the power stage. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_Power	Enable/ disable operation	FB		MC_Power_Instance (Axis:=parameter , Enable :=parameter, EnablePositive:=parameter , EnableNegative :=parameter, BufferMode :=parameter , Status=> parameter, Busy => parameter, Active=> parameter , Error=> parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Enable	Effective	BOOL	TRUE or FALSE	FALSE	Set to TRUE, control axis enters operating state Set to FALSE, release the axis's operating status
EnablePositive	Positive effective	BOOL	TRUE or FALSE	FALSE	Set to TRUE, allow axis forward rotation Set to FALSE, prohibit axis forward rotation
EnableNegative	Negative effective	BOOL	TRUE or FALSE	FALSE	Set to TRUE , allow axis reverse rotation Set to FALSE , prohibit axis reverse rotation
BufferMode	Buffer mode	MC_Buffer_Mode	0: mcAborting 1: mcBuffered	0	When Enable changes from TRUE to FALSE, the specified axis is released from operating state

■ Output variable

Name	Meaning	Data type	Valid range	Description
Status	Operable	BOOL	TRUE or FALSE	TRUE when the axis state has entered the operating state FALSE when the axis state is in the released operating state
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Status	When the specified axis enters the operating state	◆ When the specified axis is released from operating state ◆ When Error becomes TRUE
Busy	When Enable is TRUE	◆ When Error is TRUE
Active	When the instruction is started	◆ When Error is TRUE
Error	The value of the command input variable is not within the allowed range	◆ When Enable changes from TRUE to FALSE

■ Function description

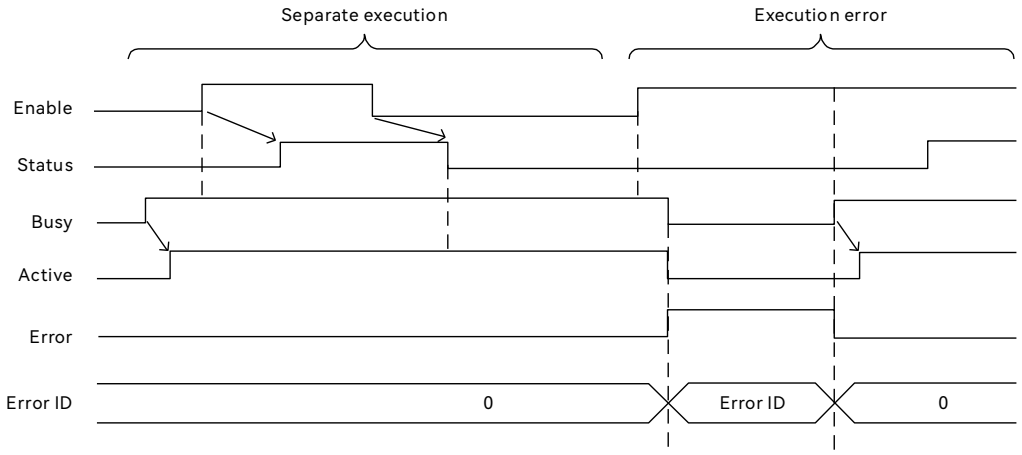
- This command is used to control the specified axis to enable or disable the operating state. This command can operate on the servo axis, virtual servo axis, and pulse axis, but cannot operate on the encoder axis. The virtual servo axis has no released operating status. When executing this command on the virtual servo axis, regardless of whether the Enable of the command is TRUE or FALSE, the Status bit of the command is TRUE.
- When Enable is set to TRUE, the control axis enters the operating state. When the Status is TRUE, the axis is in an operating state, receiving action commands, and can be controlled.
- When Enable is set to FALSE, release the operating status of the axis; When the Status is FALSE, the axis is in the released operating state, does not receive action commands, and cannot be controlled, MC_ Reset (axis abnormal reset) can still be executed.
- When EnablePositive is TRUE, the axis is allowed to rotate forward, and the motion command can control the axis to rotate forward. When EnablePositive is FALSE, the axis cannot be controlled to rotate forward. When the motion command controls the axis to rotate forward, an error will be reported. When the motion command controls the axis from reverse to forward rotation, the executing motion command will be aborted, and the axis will stop according to the deceleration and jerk set by the aborted command. After the axis stops, it enters the Standstill status.
- EnablePositive changes from TRUE to FALSE when the motion command controls the forward rotation of the axis. The executing motion command will be aborted, and the axis will stop at the deceleration and jerk set by the aborted command. After the axis stops, it enters the Standstill status.
- When EnableNegative is TRUE, axis reversal is allowed, and motion commands can control axis reversal. When EnableNegative is FALSE, axis reversal cannot be controlled, and an error will be reported when the motion command controls axis reversal. When the motion command controls the axis from forward to reverse, the executing motion command will be aborted, and the axis will stop according to the deceleration and jerk set by the aborted command. After the axis stops, it enters the Standstill status.
- Enabling Negative changes from TRUE to FALSE when the motion command controls the axis to reverse. The executing motion command will be aborted, and the axis will stop at the deceleration and jerk set by the aborted command. After the axis stops, it enters the Standstill status.

- BufferMode

  When the input variable Enable changes from TRUE to FALSE, different values can be set through BufferMode to choose the way to release the operating status.

BufferMode value	Meaning
mcAborting (Aborted)	When the axis is in a standstill status and Enable changes from TRUE to FALSE, the axis operation status is released and the axis status enters the Disabled status. When the axis is operating and Enable changes from TRUE to FALSE, the axis operation status is released, and the axis status enters the errorstop state. Users need to execute MC_Reset command to reset before controlling the axis again. **Note**: When the axis is operating, Enable changes from TRUE to FALSE to control the axis to disengage, which may cause equipment collision or personal injury. When using, please be sure to understand whether the on-site equipment can perform this operation.
mcBuffered (buffered)	When the axis is in a standstill status and Enable changes from TRUE to FALSE, the axis will release the operating status and enter the Disabled status. When the axis is in operation and Enable changes from TRUE to FALSE, the axis operation status will not be released until it becomes standstill, and the axis operation status will enter the Disabled status.

■ Output variable timing diagram description



- **Separate execution**

  This instruction will be executed when Enable is either FALSE or TRUE, as it has two functions: enable and disable operation. After the task is executed, Busy (executing) will become TRUE, and the next period Active (under control) will become TRUE. After Enable is set to TRUE, control the specified axis to run. When entering the operating status, Status becomes TRUE, while Busy and Active remain TRUE. After Enable becomes FALSE, the specified axis is released from operation. After releasing, the Status becomes FALSE, while Busy and Active remain TRUE.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range (such as the axis number exceeding the range), the instruction Error will become TRUE, and the corresponding error code will be output. The cause of the problem can be found by the value of the ErrorID, and Busy (executing) and Active (under control) will become FALSE. When Enable changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID (error code) becomes 0.

4.2 MC_Reset（abnormal axis reset）

When there is an alarm, abnormal state machine, or abnormal position error on the specified axis, reset it through this instruction. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_Reset	Abnormal axis reset	FB	MC_Reset_Instance MC_Reset Axis — Done Execute — Busy Error ErrorID	MC_Reset_Instance (Axis:=parameter , Execute:=parameter , Done=> parameter , Busy => parameter, Error=> parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs

■ Output variable refreshing timing

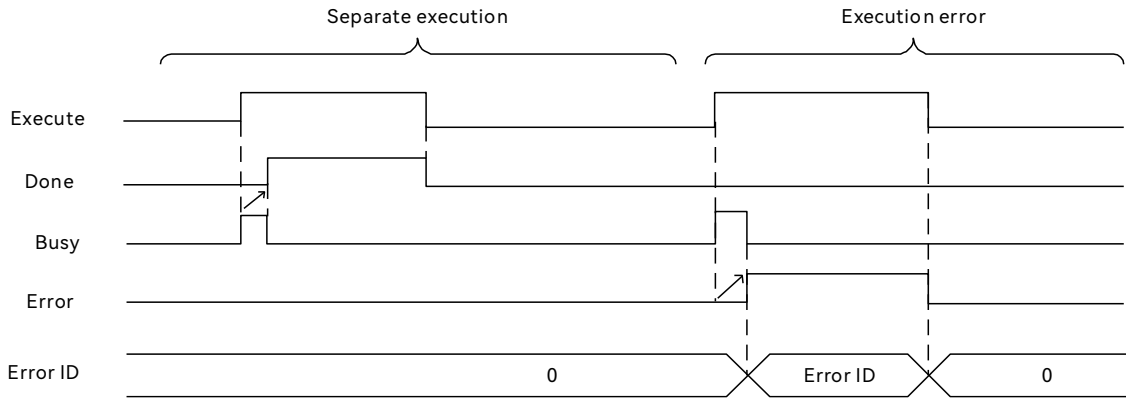
Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the axis abnormality is relieved	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one period, it becomes FALSE
Busy	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE
Error	When the input variable value of the command is not within the allowed range or the axis is abnormal and cannot be reset normally	◆ When Error is TRUE and Execute changes from TRUE to FALSE

■ Function description

- This command is used to reset axis abnormalities. When there is an alarm, abnormal state machine, or position error exceeding the set value on the specified axis, it can be reset through this command.
- Only when there is an abnormality in the specified axis can it be reset through this command. When Execute changes from FALSE to TRUE, start resetting exceptions for the specified axis. When resetting axis anomalies with this command, sometimes multiple periods are required to complete the reset.
- When there is an alarm or error in the drive, it is recommended to first eliminate the alarm or error in the drive before executing the command to reset.

- This command can reset the servo axis, virtual servo axis, pulse axis, and encoder axis.
- When this instruction is executed, it can be combined with MC_ ReadStatus(read axis status) command. MC_ ReadStatus instruction reads the status of the axis and enters the ErrorStop status, it executes the instruction. A common exception is that the difference between the specified axis command position and the feedback position exceeds the set value in the software, causing the axis to enter an ErrorStop status.

■ Output variable timing diagram description



- **Separate execution**

  When there is an abnormality in the axis, when Execute (start) changes from FALSE to TRUE, Busy (executing) also becomes TRUE. When the axis abnormality is eliminated, Done becomes TRUE, and Busy (executing) becomes FALSE. When Execute (start) becomes FALSE, Done (complete) also becomes FALSE.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy (executing) becomes TRUE, Error in the next period becomes TRUE, Busy (executing) becomes FALSE, and ErrorID (error code) outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID (error code). When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

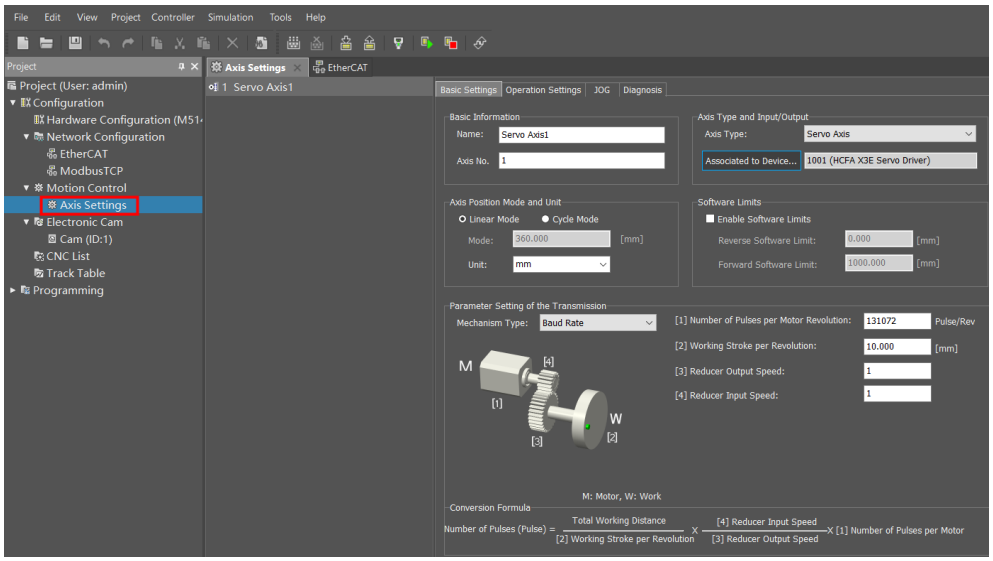
■ Example program

- **Functionality**

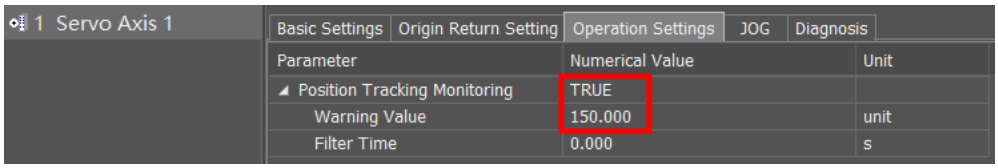
  When the axis encounters an alarm or other error, it can be executed by executing MC_ Reset (abnormal axis reset) resets the axis alarm or internal controller alarm.

- **Axis parameter setting**

  The axis parameter setting for axis1 is shown below.



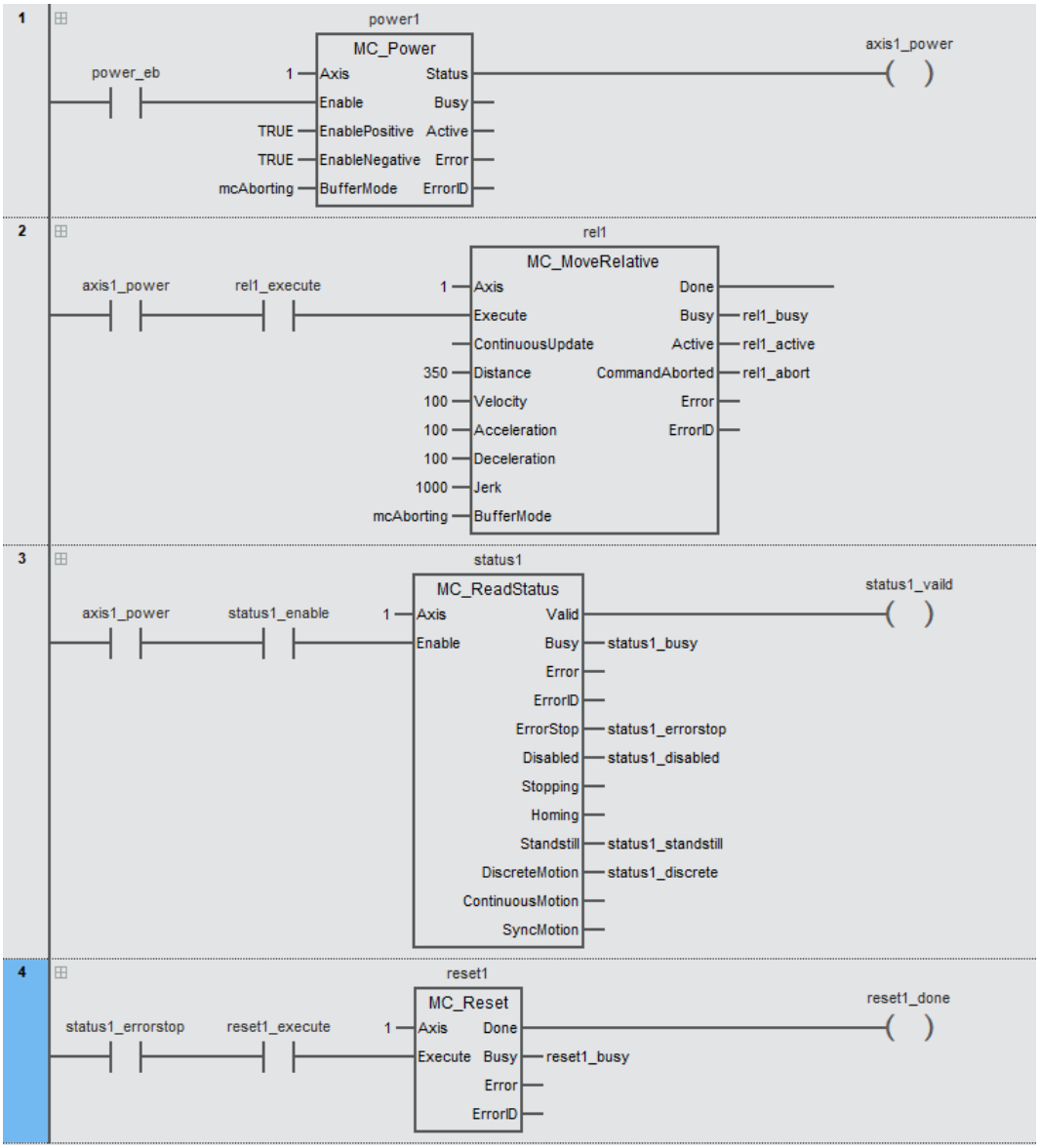
The allowable position error for the axis command position and feedback position is shown in the red box in the following figure, with a set value of 150. The "position tracking monitoring" at the red box in the following figure is set to TRUE.



■ Variable table

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_execute		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	reset1_execute		BOOL		
VAR	reset1		MC_Reset		
VAR	reset1_done		BOOL		
VAR	reset1_busy		BOOL		
VAR	status1		MC_ReadStatus		
VAR	status1_enable		BOOL		
VAR	status1_vaild		BOOL		
VAR	status1_busy		BOOL		
VAR	status1_disabled		BOOL		
VAR	status1_standstill		BOOL		
VAR	status1_discrete		BOOL		
VAR	status1_errorstop		BOOL		

- LD



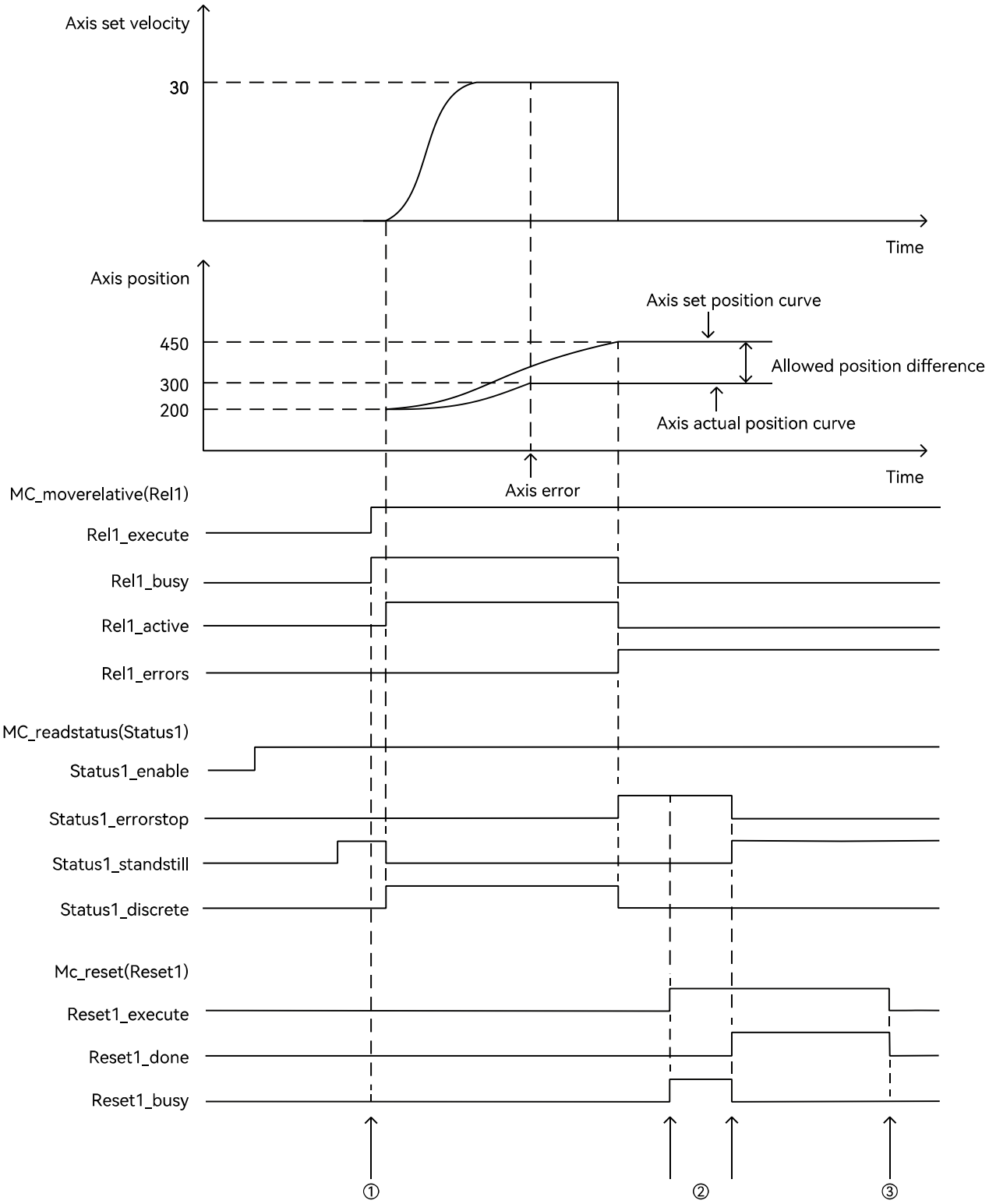
- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE
    BufferMode:=mcAborting ,
    Status=>axis1_power);

rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_execute ,
    Distance:=350 ,
    Velocity:=100 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcAborting ,
    Done=>rel1_done ,
    Busy=>rel1_busy ,
    Active=>rel1_active ,
```

```
        CommandAborted=>rel1_abort );

    status1(
        Axis:=1 ,
        Enable:= axis1_power AND status1_enable ,
        Valid=> status1_vaild,
        Busy=> status1_busy ,
        ErrorStop=> status1_errorstop,
        Disabled=> status1_disabled ,
        Standstill=> status1_standstill ,
        DiscreteMotion=> status1_discrete);

    reset1(
        Axis:=1 ,
        Execute:=status1_errorstop AND reset1_execute ,
        Done=>reset1_done ,
        Busy=>reset1_busy);
```

■ Program description

- After enabling the axis, re1_execute becomes TRUE, the relative displacement instruction (rel1) begins to execute, and the second period in which re1_execute becomes TRUE begins to control the axis.
- When limit alarms occur during the operation of the axis, the command position of the axis continues to increase, and the actual position of the axis no longer increases. When the difference between the command position and the actual position of the axis exceeds the allowable position error set in the software 【Axis Settings】 →【Operation Settings】, the axis state enters the errorstop status. When reset1_execute becomes TRUE, a reset command is executed. After the reset command is completed, the axis status changes from errorstop to standstill, and the axis command position becomes the actual value of the axis position.
- When reset1_execute becomes FALSE, the Done also becomes FALSE, preparing for the next execution of the reset command.



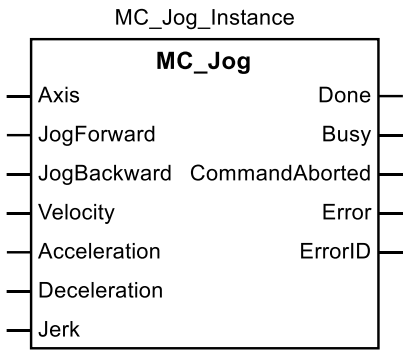
① The relative command (rel1) starts to execute, and control the axis in the second cycle

② When the set position and actual position of the axis exceed the allowable position difference error, the axis status enters errorstop, and when reset1_execute becomes True, MC_Reset is executed to reset the error, and after the execution is completed, the axis status changes to standstill.

③ When reset1_execute becomes FALSE after reset completion, Done becomes FALSE.

4.3 MC_Jog（JOG）

Control the specified axis for JOG. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_Jog	JOG	FB		MC_Jog_Instance (Axis :=parameter, JogForward :=parameter, JogBackward :=parameter , Velocity :=parameter , Acceleration :=parameter , Deceleration :=parameter , Jerk :=parameter , Done => parameter , Busy => parameter, CommandAborted => parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
JogForward	Jog forward	BOOL	TRUE or FALSE	FALSE	Set to TRUE, control the axis to move forward Set to FALSE, stop forward movement
JogBackward	Jog backward	BOOL	TRUE or FALSE	FALSE	Set to TRUE, control the axis to move backward Set to FALSE, stop reverse movement
Velocity	Target velocity	LREAL	Positive number	Required field	Specify target velocity *1 (unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Specify acceleration *1 (unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (unit: travel unit/second^3) *2

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions"*.

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions"*.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when jogging stops
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when an instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

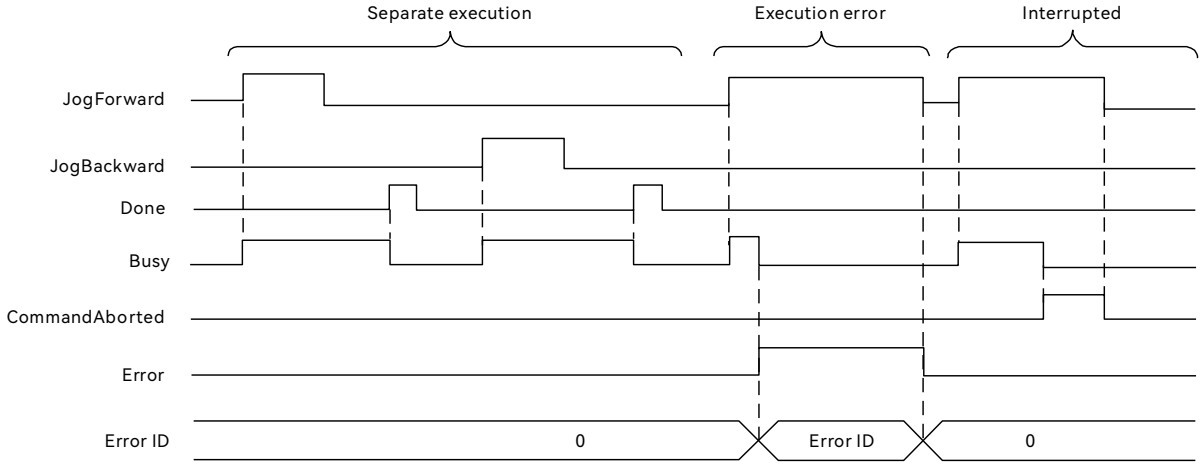
Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After the jog ends, Done becomes TRUE for one period	◆ After the specified axis stops, Done becomes TRUE for one period and then becomes FALSE
Busy	The rising edge of JogForward or JogBackward	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the execution of instructions is aborted	◆ When CommandAborted is TRUE and JogForward or JogBackward changes from TRUE to FALSE ◆ When JogForward or JogBackward is FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period, and then becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encountered an exception during the instruction execution process.	◆ When Execute changes from TRUE to FALSE

■ Function description

● Basic function description

This instruction is used to control the specified axis to perform forward or reverse jogs based on input variables. When JogForward is TRUE, the axis runs in the forward direction, and when JogForward is FALSE, the axis stops operating. When JogBackward is TRUE, the axis runs in reverse, and when JogBackward is FALSE, the axis stops operating. JogForward and JogBackward cannot both be TRUE.

■ Output variable timing diagram description



● Separate execution

When JogForward or JogBackward changes from FALSE to TRUE, Busy also becomes TRUE during execution; When JogForward or JogBackward changes from TRUE to FALSE, the axis decelerates and stops according to the input variable setting value. After the axis stops and the axis state becomes stand still, Done changes from FALSE to TRUE for one period and then returns to FALSE.

● Interrupted

When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy, and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

● Execution error

When the input variable value of this instruction is not within the allowable range, the instruction JogForward or JogBackward changes from FALSE to TTRUE, while Busy becomes TRUE, and the next period

Error becomes TRUE. Busy becomes FALSE, and the corresponding error code is output. The cause of the problem can be found by the value of ErrorID. When the instruction JogForward or JogBackward changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

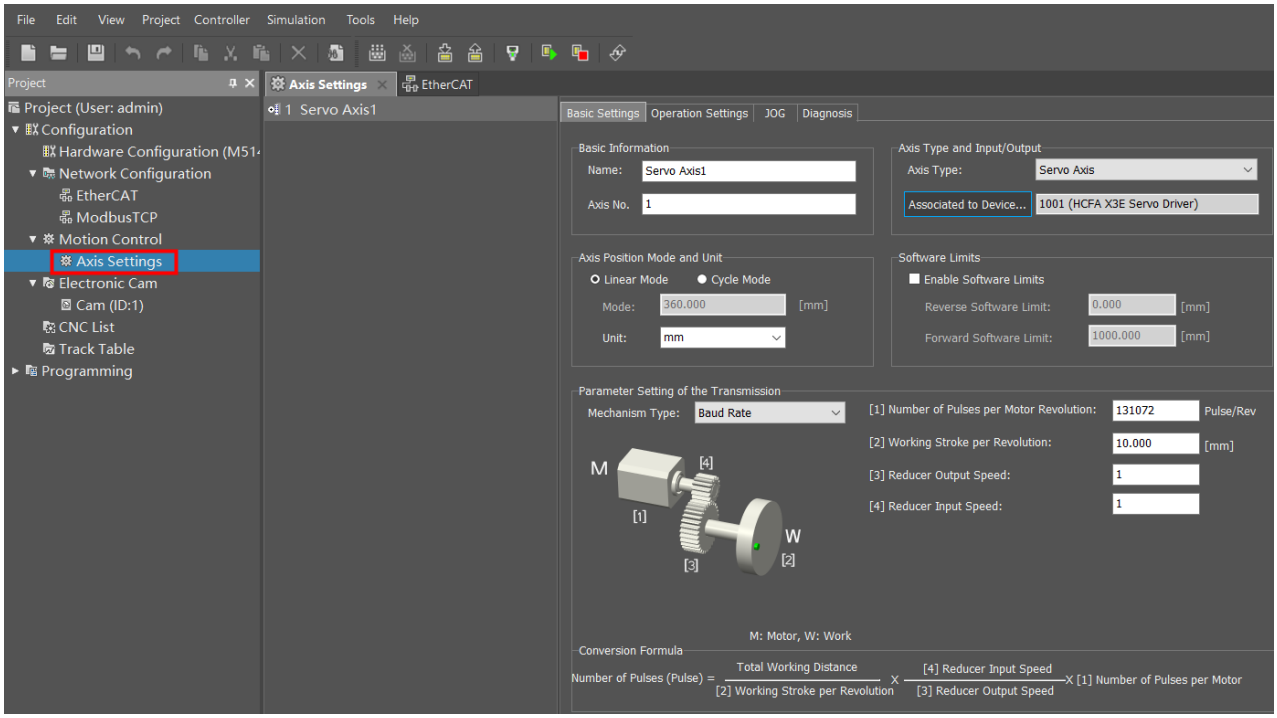
■ Example program

● Functionality

After enabling the axis, MC_ JOG command controls the forward and reverse jogs of the axis. When the forward rotation button is pressed, the axis rotates forward; When the forward button is released, the axis deceleration stops. When the reverse button is pressed, the axis rotates backward. When the reverse button is released, the axis deceleration stops.

● Axis parameter setting

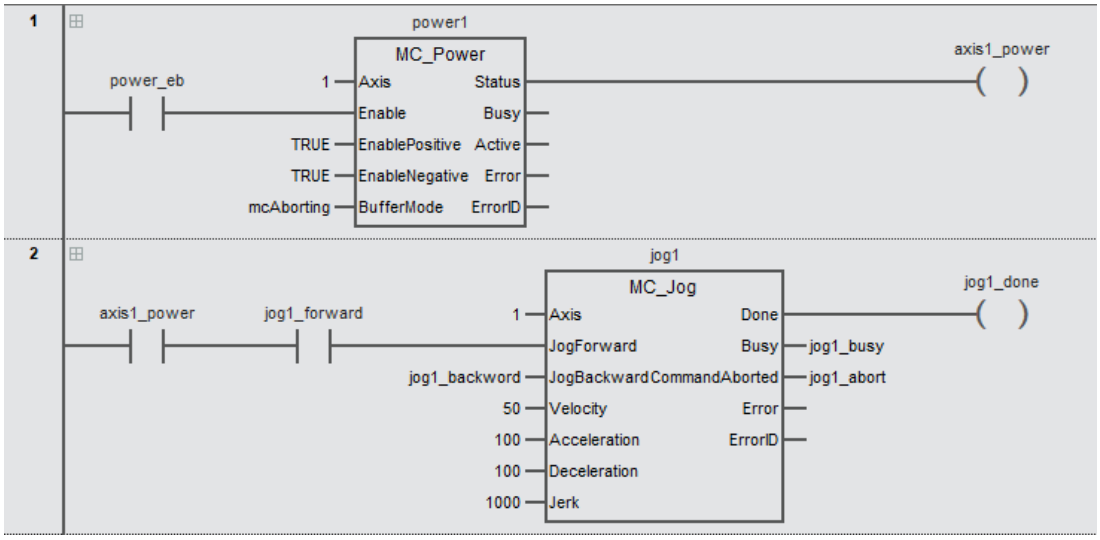
The axis parameter setting for axis1 is shown below.



● Variable table

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	jog1_forward	%IX0.0	BOOL		
VAR	jog1_backward	%IX0.1	BOOL		
VAR	jog1		MC_Jog		
VAR	jog1_done		BOOL		
VAR	jog1_busy		BOOL		
VAR	jog1_abort		BOOL		

- LD

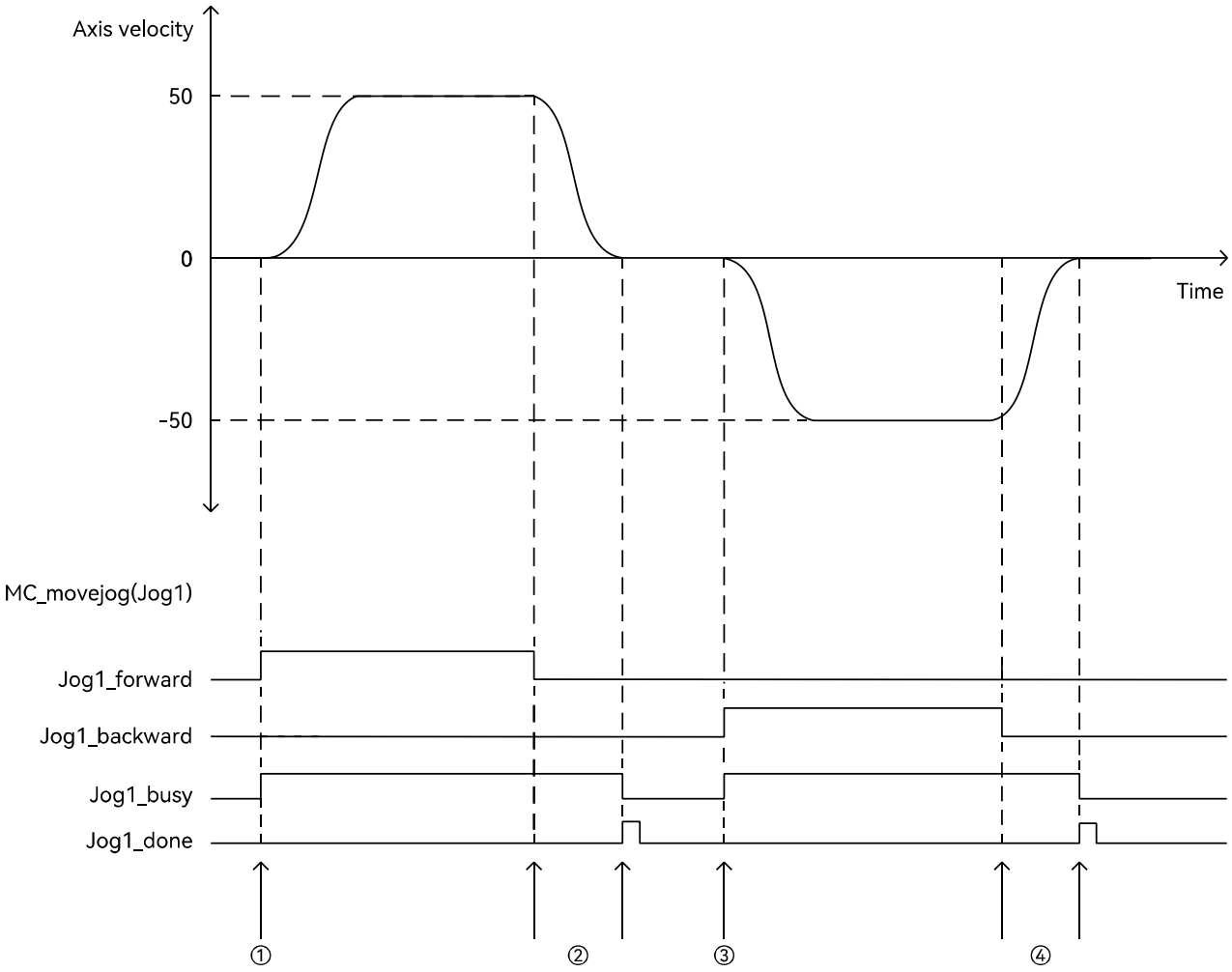


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);

jog1(
    Axis:=1 ,
    JogForward:=axis1_power AND jog1_forward ,
    JogBackward:=jog1_backward ,
    Velocity:=50 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Done=>jog1_done ,
    Busy=>jog1_busy ,
    CommandAborted=>jog1_abort);
```

- Program description

  - After enabling the axis, control the forward and reverse jogs of the axis through external buttons. The external forward button is connected to the input channel % IX0.0 of the controller, and the corresponding variable is jog1_ forward; the external reverse button is connected to the input channel %IX0.1, and the corresponding variable is jog1_ backward.
  - When the forward button is pressed, the axis rotates forward and accelerates to the target velocity according to the input parameters set by MC_Jog; when the forward button is released, the axis deceleration stops.
  - When the reverse button is pressed, the axis reverses and accelerates to the target velocity according to the input parameters set by MC_Jog; when the reverse button is released, the axis deceleration stops.



① When jog1_forward is TRUE, the axis rotates positively and accelerates to the target speed according to the input parameter set by the jog command.

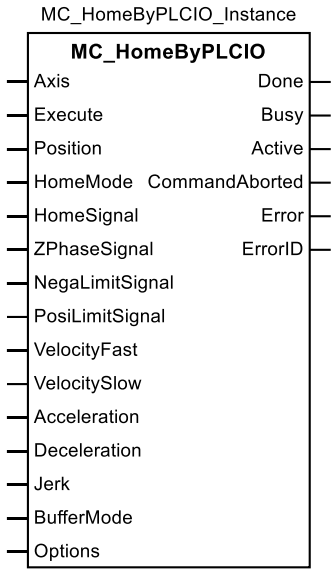
② When jog1_forward is FALSE, the axis stops and decelerates to 0 according to the input parameter set by the jog command. after the axis speed is decelerated to 0, the completion bit of the jog command becomes TRUE, and one cycle later it becomes FALSE.

③ When jog1_backward is TRUE, the axis reverses and accelerates to the target speed according to the input parameter set by the jog command.

④ When jog1_backward is FALSE, the axis stops and decelerates to 0 according to the input parameter set by the jog command. after the axis speed is decelerated to 0, the completion bit of the jog command becomes TRUE, and one cycle later it becomes FALSE.

4.4 MC_HomeByPLCIO (home setting via input signals)

Use the controller's homing modes, homing signals, limit signals determine the mechanical origin for the specified axis. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_HomeByPLCIO	Home setting via input signals	FB		MC_HomeByPLCIO_Instance(Axis :=parameter , Execute :=parameter , Position :=parameter , HomeMode :=parameter , HomeSignal :=parameter , ZPhaseSignal :=parameter , NegaLimitSignal :=parameter , PosiLimitSignal :=parameter, VelocityFast :=parameter, VelocitySlow :=parameter, Acceleration :=parameter , Deceleration :=parameter , Jerk :=parameter, BufferMode :=parameter, Options :=parameter, Done => parameter , Busy => parameter , Active => parameter , CommandAborted => parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
Position	Home position	LREAL	Negative number, positive number, 0	0	The set position of the axis after homing (unit: travel unit)
HomeMode	Home mode	INT	17~30、35	0	Different modes of homing For details, please refer to Appendix 1 Homing modes
HomeSignal	Home signal	BOOL	TRUE or FALSE	FALSE	When the rising edge of this parameter is detected, the axis executes corresponding actions based on different homing modes (this signal needs to be connected to the controller)
ZPhaseSignal	Z-phase signal	BOOL	TRUE or FALSE	FALSE	Not available for now
NegaLimitSignal	Negative limit signal	BOOL	TRUE or FALSE	FALSE	When the rising edge of this parameter is detected, the axis executes corresponding actions based on different homing modes (this signal needs to be connected to the controller)
PosiLimitSignal	Positive limit signal	BOOL	TRUE or FALSE	FALSE	When the rising edge of this parameter is detected, the axis executes corresponding actions based on different homing modes (this signal needs to be connected to the controller)
VelocityFast	Starting velocity of homing	LREAL	Negative number, positive number, 0	0	Set the velocity when searching for the origin
VelocityLow	Homing approach velocity	LREAL	Negative number, positive number, 0	0	The velocity at which deceleration stops after encountering the home signal
Acceleration	Acceleration	LREAL	Negative number, positive number, 0	0	Specify acceleration *1 (Unit: travel unit/second^2) *2

Deceleration	Deceleration	LREAL	Negative number, positive number, 0	0	Specify deceleration *1 (Unit: travel unit/second²) *2
Jerk	Jerk	LREAL	Negative number, positive number, 0	0	Specify jerk *1 (Unit: travel unit/second³) *2
BuffMode	Buffer mode	MC_Buffer_Mode	0：mcAborting	0	Reserved
Options	Reserved	Reserved	Reserved	Reserved	Reserved

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions ".*

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions".*

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when jogging stops
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when an instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After instruction execution is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one period, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Active	When the instruction is started	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the execution of instructions is aborted	◆ When Execute changes from TRUE to FALSE ◆ When an instruction is executed and Execute is FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE and after one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from TRUE to FALSE

■ Function description

● Basic Function description

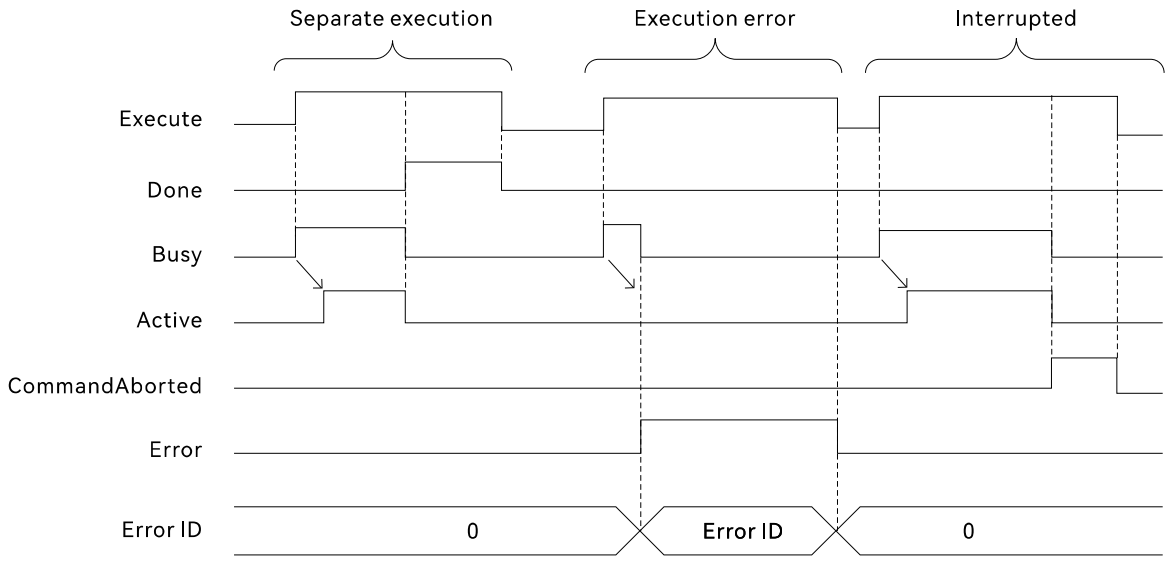
➢ Use the controller's homing mode, homing signals, and limit signals to determine the mechanical home for the specified axis. Connect the home switch, forward limit switch, or reverse limit switch to the main

input channel of the controller to achieve the homing function. (The axis specified by this instruction can be a servo axis, virtual servo axis, pulse axis, or encoder axis.)

- The encoder axis can only be set to the homing mode 35.
- This instruction can only be executed when the axis is in StandStill status. When executed in other status, this instruction will report an error.

- **Instruction completion timing**
  - After the origin is set, the instruction is completed and Done changes from FALSE to TRUE.
- **Re-execute the instruction**
  - When the instruction execution is completed and Execute changes from FALSE to TRUE again, the instruction can be re executed; When an instruction is being executed and Execute changes from FALSE to TRUE again, it does not affect the execution of the instruction, and it still executes the instruction according to the input variable that has not been completed.
- **Execute this instruction while other instructions are in processing**
  - When other motion instructions are executed, starting this instruction will result in an error, and can only be executed when the axis is in StandStill status.
- **Execute other instructions while this instruction is in processing**
  - When this instruction is executed, other motion instructions are initiated, and how they are buffered is determined by the BufferMode parameter values of other motion instructions. When other motion instructions and this instruction are buffered, the timing for the execution of other instructions is when the instruction Done becomes TRUE. If there is no BufferMode for other motion instructions, it is generally aborted.
- **Abnormality elimination**
  - When the instruction is executed, if the input variable value of the instruction is not within the allowed range or does not meet the execution conditions of the instruction, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed, if an exception is encountered (such as an axis alarm), the instruction will also report an error and the axis will immediately stop.

■ **Output variable timing diagram description**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. When the instruction is completed, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy becomes TRUE, Error becomes TRUE, Busy becomes FALSE, and ErrorID outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

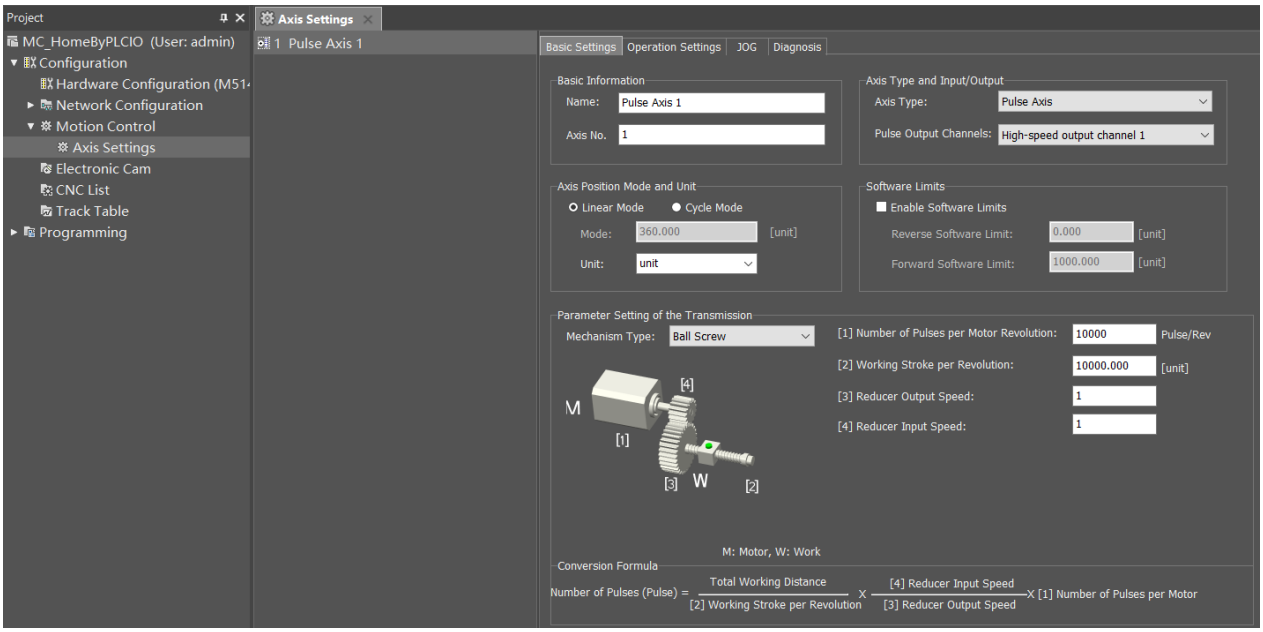
- **Interrupted**

  When the input variable value of this instruction is not within the allowable range, when the Execute command changes from FALSE to TRUE, Busy (executing) also becomes TRUE, and the next period Error (error) becomes TRUE. The corresponding error code is outputted by the ErrorID (error code), which can be used to find the cause of the problem. When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

■ Example program

- **Parameter configuration**

  The axis parameter setting for axis1 is shown below.



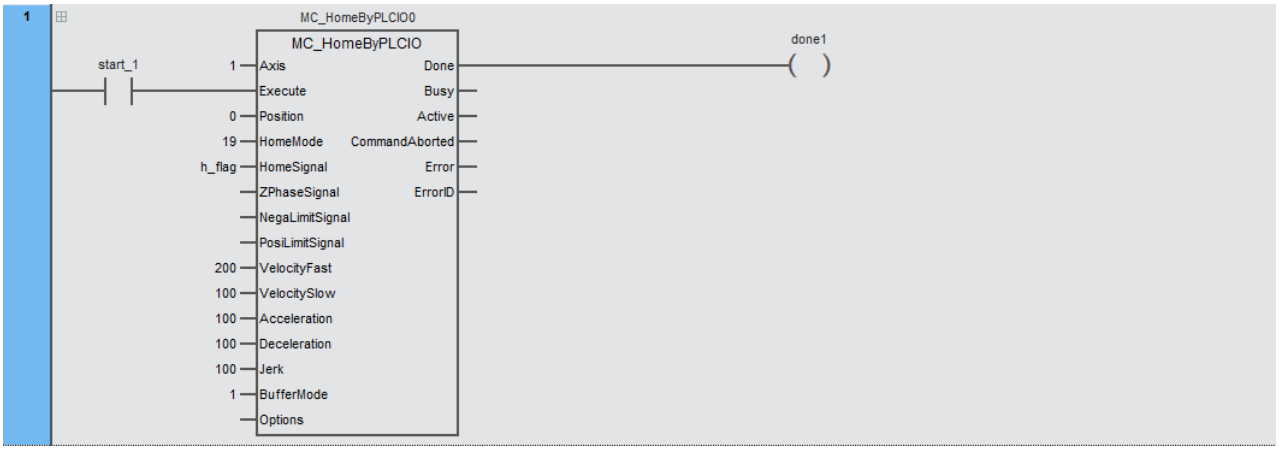
- **Take homing mode 19 as an example to illustrate the homing function**

  Set the trigger position of the external sensor signal to the user-specified home position. The homing mode and velocity are set through MC_ HomeByPLCIO instruction, and the external sensor signal is connected to the input channel of the controller. After the instruction is executed, homing is executed according to the homing mode set by the instruction.

- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	start_1		BOOL		
VAR	MC_HomeByPLCIO		MC_HomeByPLCIO		
VAR	h_flag		BOOL		
VAR	done1		BOOL		

- **LD**



- **ST**

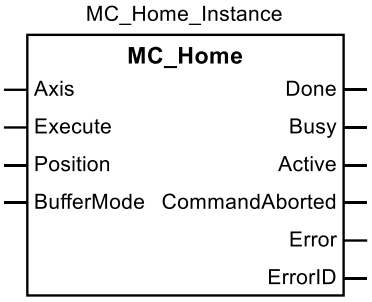
```
MC_HomeByPLCIO1(
Axis:= 1,
Execute:= start_1,
Position:= 0,
HomeMode:= 19,
HomeSignal:= h_flag,
VelocityFast:= 200,
VelocitySlow:= 100,
Acceleration:= 100,
Deceleration:= 100,
Jerk:= 100,
BufferMode:= 1,
Done=> done1 );
```

- **Program Description**

  - When Start_ 1 change from FALSE to TRUE, the homing is executed according to MC_ parameter set by HomeByPLCIO. The controller input detected the home signal from ON to OFF, and the home setting is completed. After completing the home setting, set the trigger position of the external sensor signal to the user-specified home position. The completion flag of the home setting will change to TRUE, and the output parameter Done of the MC_HomeByPLCIO instruction will become TRUE.
  - When the home signal is invalid during start, run at high velocity in the forward direction. When encountering the home signal from OFF to ON during forward operation, slow down and stop, then change to low-velocity and run in the negative direction. When encountering the home signal state from ON to OFF during low velocity negative operation, decelerate and stop, using the stop position as the home.
  - When the home signal is valid during start, run at high velocity in the negative direction. When operating in the negative direction, when encountering the home signal from ON to OF, slow down and stop, then quickly retreat to the position where the home signal is valid before slowing down and stopping, and then change to a low velocity to run in the negative direction. When encountering the home signal from ON to OF during low velocity negative operation, decelerate and stop, using the stop position as the home.
  - Regardless of whether encountering a negative limit or a positive limit ON status, the process of homing is stopped and an alarm is triggered.

4.5 MC_Home（home setting）

The specify axis uses the homing mode, homing signals, limit signals of the drive to determine the mechanical origin. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_Home	Home setting	FB		MC_Home_Instance (Axis :=parameter , Execute :=parameter, Position :=parameter, BufferMode :=parameter, Done => parameter , Busy => parameter , Active => parameter , CommandAborted => parameter, Error => parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
Position	Home position	LREAL	Negative number, positive number, 0	0	The set position of the axis after finding the origin (unit: travel unit) *1
BufferMode	Buffer mode	MC_Buffer_Mode	0: Reserved	0	Reserved

*1: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions"*.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when an instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When instruction execution is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one period, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE

Active	When the instruction is started	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the execution of instructions is aborted	◆ When Execute changes from TRUE to FALSE ◆ When an instruction is executed and Execute is FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from TRUE to FALSE

■ Function description

● Basic function description

- ➢ When the instruction controls the axis as a servo axis, it is used to determine the mechanical origin using the drive's homing mode, home signals, and limit signals. Connect the origin, forward limit, or reverse limit to the external input of the servo drive to achieve the homing function.
- ➢ The virtual servo axis and encoder axis homing mode can only be set to 35.
- ➢ Set the homing mode, homing start velocity, and homing approach velocity in the 【Motion Control】→【Axis Settings】→【Homing Settings】 section of the software. Please refer to the example program for the setting. Please refer to the instructions of the drive corresponding to the specified axis for homing mode description. For instructions on the homing mode of the HCFA servo drive, please refer to the last chapter " Homing modes".
- ➢ This instruction can only be executed when the axis is in StandStill status. When executed in other status, this instruction will report an error.

● PDO mapping

The M500S series controllers do not require user mapped data objects. The M500 series controller requires users to map data objects, as shown in the table below.

PDO reception (master=>slave)(hexadecimal)	Meaning of the mapped data	PDO sending (slave => master)	Meaning of mapped data
6040_ 0 (index_sub index)	Control word	6041_ 0 (index_sub index)	Control word
6060_ 0 (index_sub index)	Control model	6061_ 0 (index_sub index)	Feedback mode

● Instruction completion timing

After the home is set, the command is completed, and Done changes from FALSE to TRUE.

● Re-execute the instruction

When the instruction execution is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; When an instruction is being executed and Execute changes from FALSE to TRUE again, the execution of the instruction will not be effected and continue according to the input variable that has not been completed.

● Execute this instruction while other instructions are in processing

When other motion instructions are executed, starting this instruction will cause an error and can only be executed when the axis is in StandStill status.

● Execute other instructions while this instruction is in processing

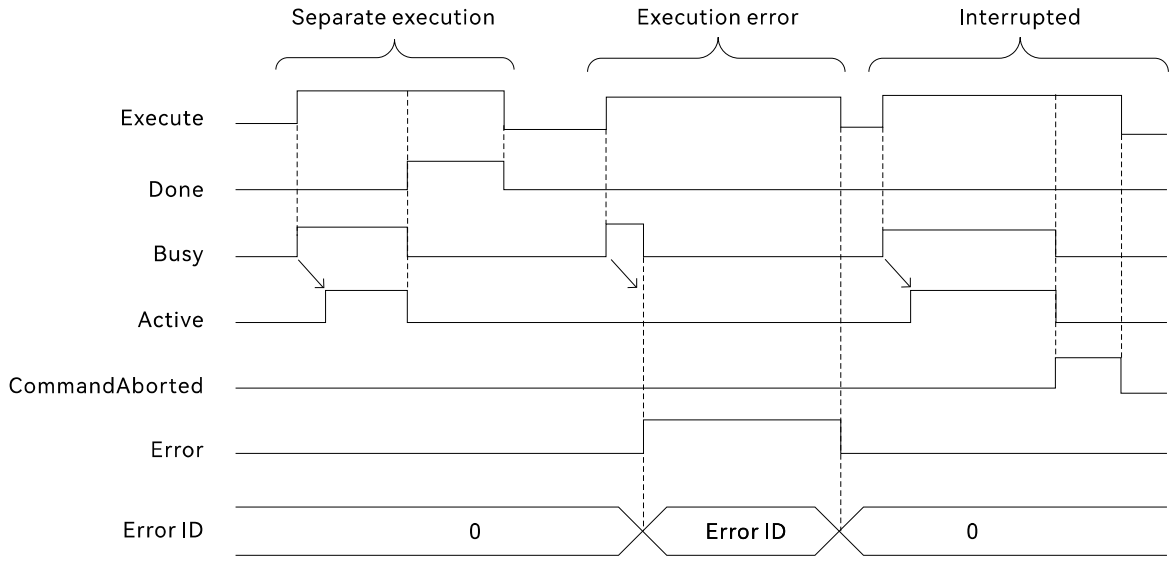
When other motion instructions are initiated while this instruction is executed, the ways other instructions are buffered are determined by the BufferMode parameter values of other motion instructions.

When other motion instructions and this instruction are buffered, the timing for the execution of other instructions is when the instruction Done becomes TRUE. If other motion instructions do not have the BufferMode parameter, the instruction is usually aborted.

- **Abnormality elimination**

  When the instruction is executed and the input variable value of the instruction is not within the allowed range or does not meet the execution conditions of the instruction, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed and an exception is encountered (such as an axis alarm), the command will also report an error and the axis will immediately stop.

■ Output variable timing diagram description



- **Separate execution**

  When Execute changes from FALSE to TRUE, and Busy also becomes TRUE, and Active becomes TRUE in the next period. When the instruction is completed, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range and Execute instruction changes from FALSE to TRUE, Busy becomes TRUE, Error becomes TRUE, Busy becomes FALSE, and ErrorID outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID (in error code). When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

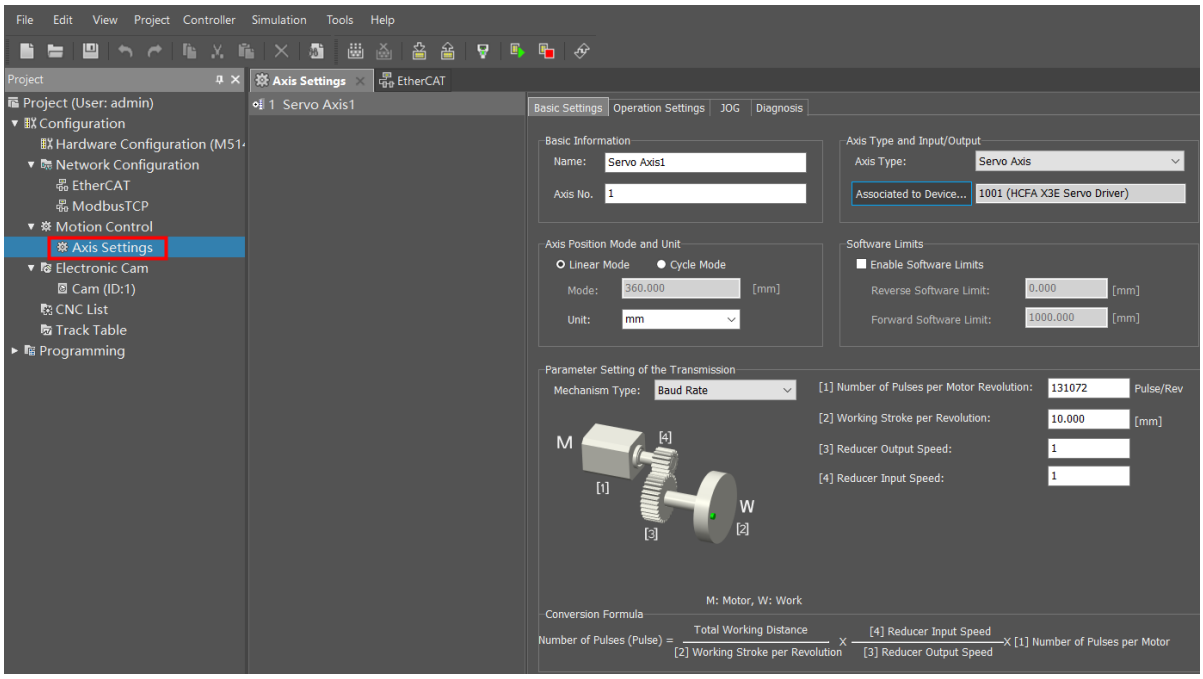
■ Example program

- **Functionality**

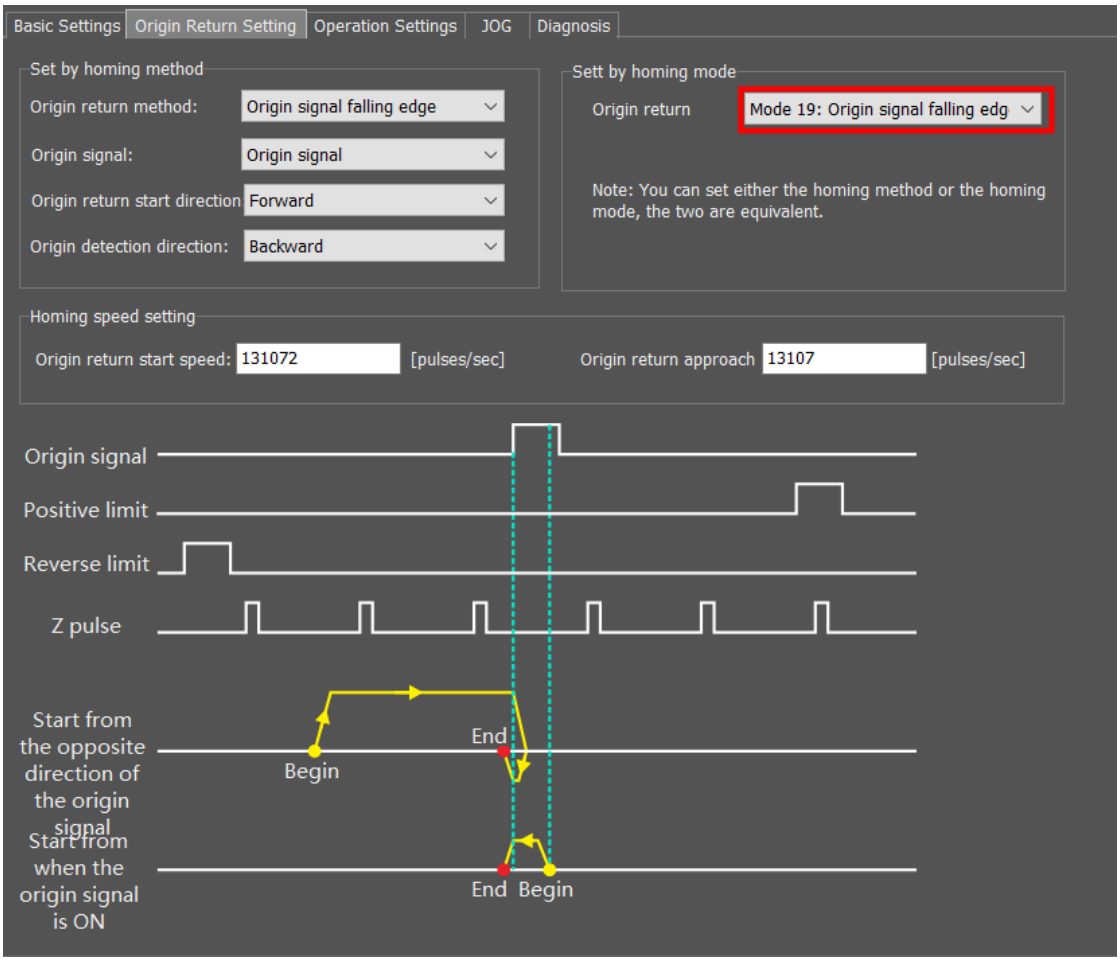
  Set the trigger position of the external sensor signal to the user specified home position. The homing mode and homing velocity are set through software. The external sensor signal is connected to the input of the servo drive. After the home setting command is executed, the homing mode is set in the software axis parameter "Homing setting" interface to execute the homing action.

- **Axis parameter setting**

  The parameter settings and homing mode settings for Axis 1 are shown below.



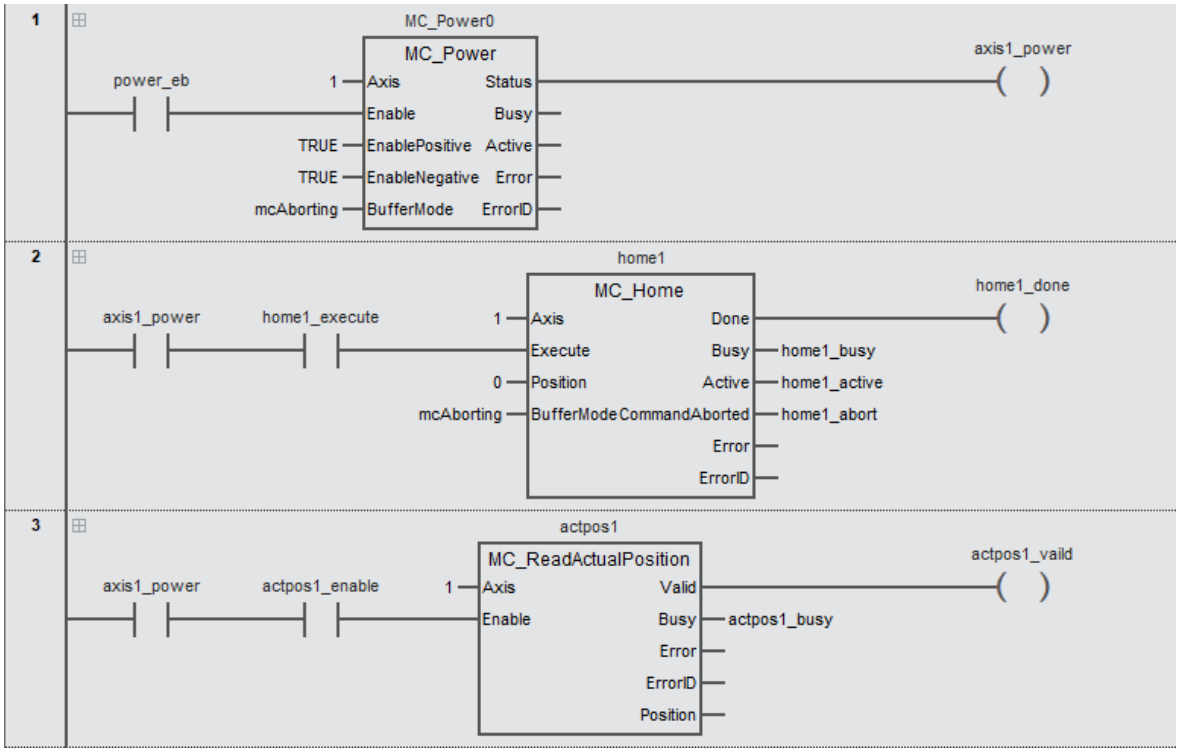
The homing mode is set as shown in the red box below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	home1_execute		BOOL		
VAR	home1		MC_Home		
VAR	home1_done		BOOL		
VAR	home1_busy		BOOL		
VAR	home1_active		BOOL		
VAR	actpos1_enable		BOOL		
VAR	actpos1		MC_ReadActualPosition		
VAR	actpos1_vaild		BOOL		
VAR	actpos1_busy		BOOL		

- **LD**

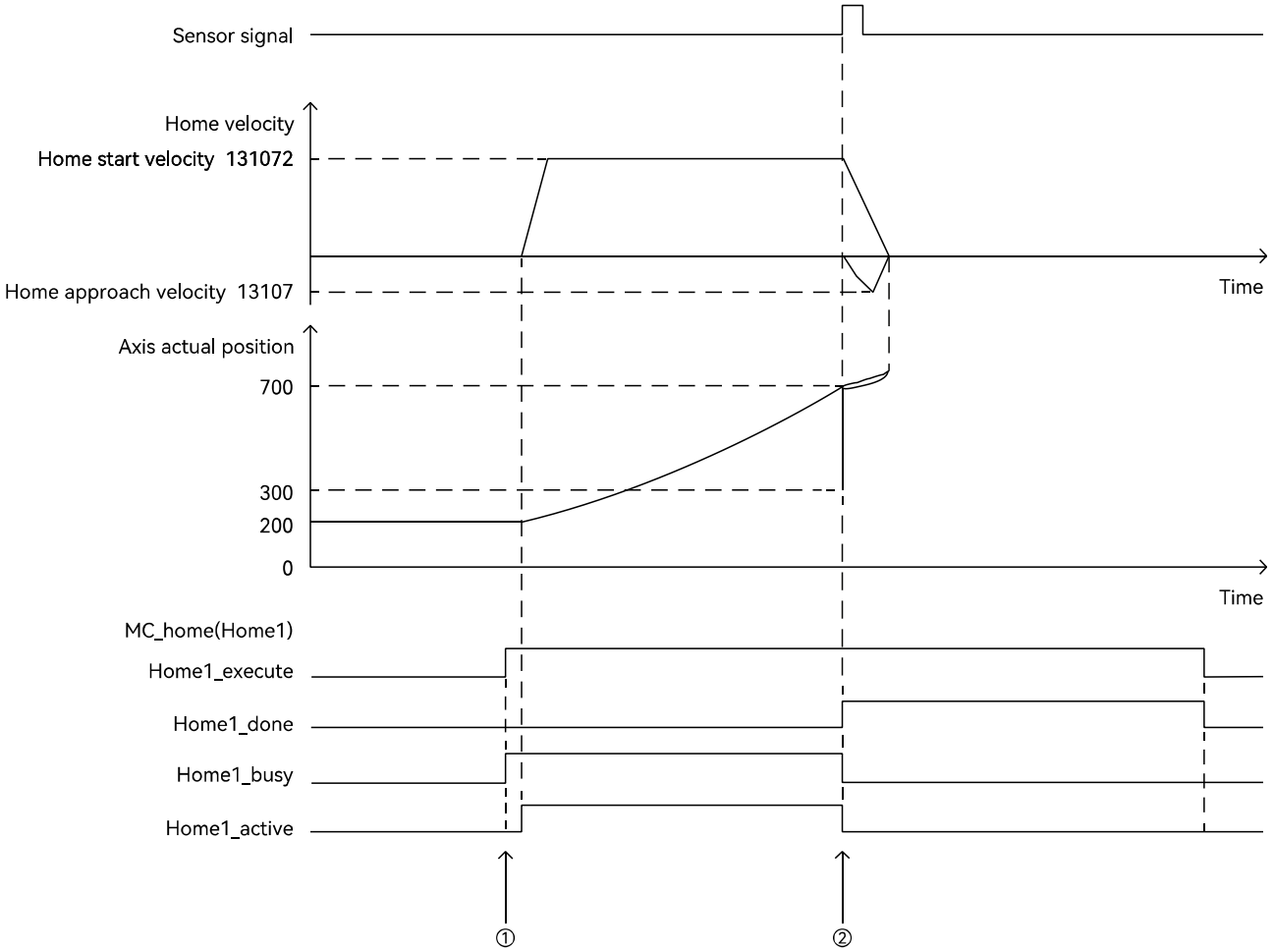


- **ST**

```
power1 (
      Axis:=1 ,
      Enable:=power_eb ,
      EnablePositive:=TRUE ,
      EnableNegative:=TRUE ,
      BufferMode:=mcAborting ,
      Status=>axis1_power);

home1(
      Axis:=1 ,
      Execute:= axis1_power AND home1_execute ,
      Position:=0 ,
      BufferMode:=mcAborting ,
```

```
Done=>home1_done ,
Busy=>home1_busy ,
Active=>home1_active ,
CommandAborted=>home1_abort );

actpos1(
    Axis:=1 ,
    Enable:=axis1_power AND actpos1_enable ,
    Valid=>actpos1_vaild ,
    Busy=>actpos1_busy );
```

- **Program description**
  - After enabling the axis, home1_execute becomes TRUE, and the home setting instruction (home1) begins to be executed. After the command is executed, the servo drive works in the home mode and uses the home function of the servo drive to control the axis to return to the home.
  - During the process of returning the axis to the home, when encountering the home signal (external sensor signal is connected to the input of the servo drive), the axis decelerates and stops, then reverses. When encountering the home signal changing from TRUE to FALSE during reversal, the current position of the axis is set to the home position set by the home setting instruction (home1).



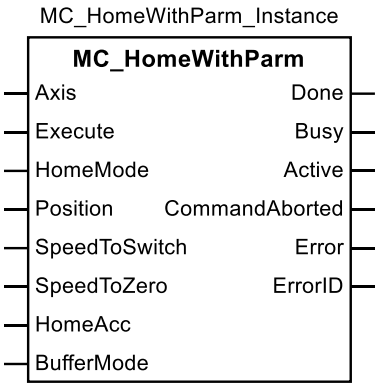
① When home1_execute becomes TRUE, the home instruction (home 1) starts to execute, controlling the servo drive to work in the home mode utilizing the servo drive's home function control.

② When the axis encounters the home signal and decelerates and stops, then reverses after stopping, and when the home signal changes from TRUE to FALSE after reversing, the current position of the axis is set to the home position set in the home instruction.

4.6 MC_HomeWithParm（home setting with parameters）

The specify axis uses the drive's homing mode, home signals, and limit signals to determine the mechanical origin. The homing mode, velocity acceleration can be set through this instruction. Library: MotionControl_ Part 2

Applicable devices: M500S series, M500 series

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_HomeWithParm	Home setting with parameters	FB		MC_HomeWithParm_Instance (Axis :=parameter , Execute :=parameter , HomeMode :=parameter, Position :=parameter, SpeedToSwitch :=parameter , SpeedToZero :=parameter, HomeAcc :=parameter, BufferMode :=parameter , Done => parameter, Busy => parameter , Active => parameter , CommandAborted => parameter, Error => parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
HomeMode	Homing mode	USINT	1~35	Required field	Control the way the specified axis finds the origin
Position	Home position	LREAL	Negative number, positive number, 0	0	The set position of the axis after finding the origin (unit: travel unit) *1
SpeedToSwitch	Starting speed of homing	UDINT	Positive number, 0		Set the speed when searching for the origin
SpeedToZero	Homing approach speed	UDINT	Positive number, 0		The speed at which deceleration stops after encountering the home signal
HomeAcc	Acceleration when searching for the origin	UDINT	Positive number, 0		Acceleration from stationary to reaching the origin and returning to the starting velocity
BufferMode	Buffer mode	MC_Buffer_Mode	0: Reserved	0	Reserved

*1: For a detailed of instruction units, please refer to the *"Parameter unit of motion control instructions".*

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TURE when the instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when an instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After instruction execution is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one period, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Active	When the axis is under control	◆ When Done becomes TRUE ◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the execution of instructions is aborted	◆ Execute changes from TRUE to FALSE ◆ When an instruction is executed and Execute is FALSE and the instruction is aborted by another instruction, CommandAborted becomes TRUE and after one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ Execute changed from TRUE to FALSE

■ Function description

● Basic function description

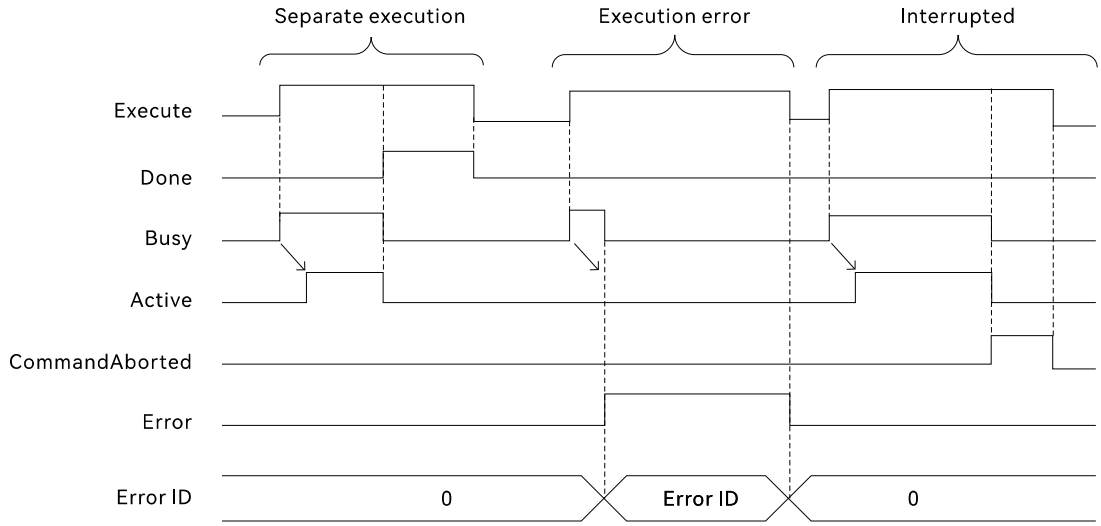
- This instruction is used to control the axis to use the drive's homing mode, home signals, and limit signals to determine the mechanical origin. The homing mode and velocity can be set through this instruction. Connect the origin, forward limit, or reverse limit switch to the external input of the servo drive to achieve the homing function. The homing mode, home position, homing starting velocity, and homing approach velocity can be changed before the execution.
- The homing mode is set by the input variable HomeMode, the starting velocity of homing is set by the input variable SpeedToSwitch, and the approaching velocity of homing is set by the input variable SpeedToZero. Please refer to the instructions of the drive corresponding to the specified axis for the homing mode description. For instructions on the homing mode of the HCFA servo drive, please refer to the last chapter " Homing modes".
- The virtual servo axis and encoder axis can only be set to the homing mode 35.
- This instruction can only be executed when the axis is in StandStill status. When executed in other status, this instruction will report an error.

● PDO mapping

The M500S series controllers do not require user mapped data objects. The M500 series controller requires users to map data objects, as shown in the table below.

PDO reception (master => slave) (hexadecimal)	Meaning of the mapped data	PDO sending (slave => master)	Meaning of the mapped data
6040_ 0 (index_sub index)	Control word	6041_ 0 (index_sub index)	Status word
6060_ 0 (index_sub index)	Control model	6061_ 0 (index_sub index)	Feedback mode

■ Output variable timing diagram description



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. When the instruction is completed, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy becomes TRUE, Error becomes TRUE, Busy becomes FALSE, and ErrorID outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, and Busy and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE

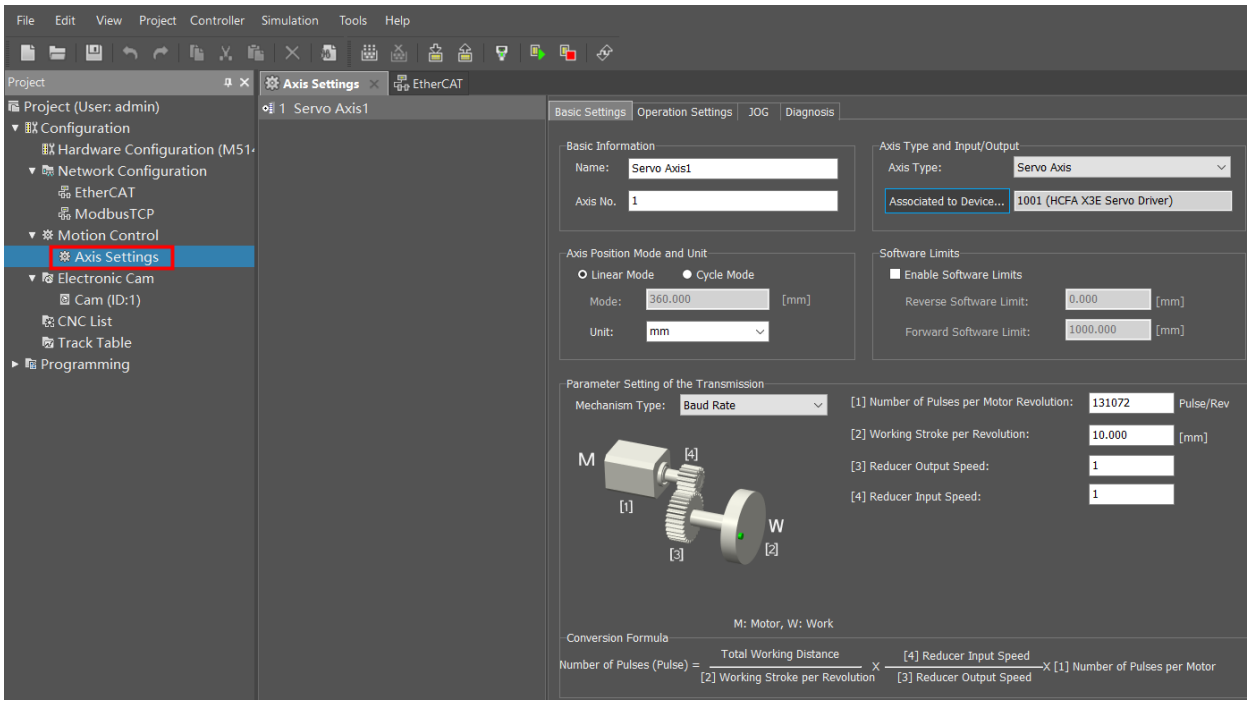
■ Example program

- **Functionality**

  Set the trigger position of the external sensor signal to the user specified home position. The homing mode and homing velocity are controlled by the instruction MC_ HomeWithParm, and external sensor signals are connected to the input of the servo drive. After the home setting instruction is executed, the homing is executed according to the instruction.

- **Axis parameter setting**

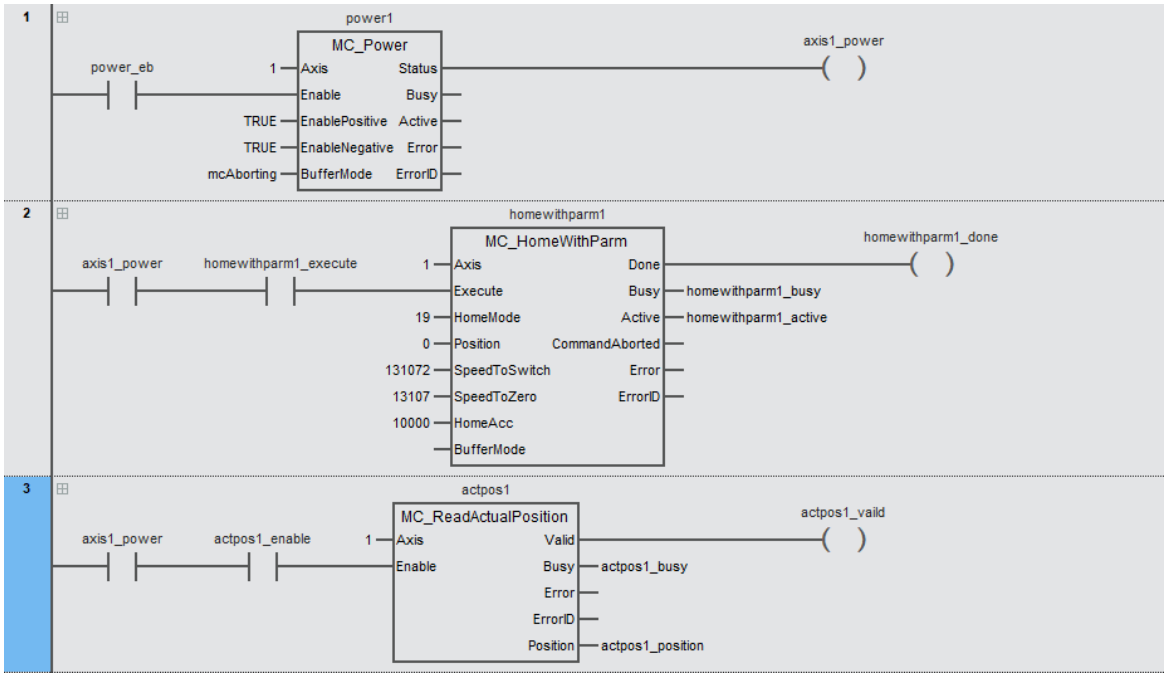
  The parameter settings and homing mode settings for Axis 1 are shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	homewithparm1_execute		BOOL		
VAR	homewithparm1		MC_HomeWithParm		
VAR	homewithparm1_done		BOOL		
VAR	homewithparm1_busy		BOOL		
VAR	homewithparm1_active		BOOL		
VAR	actpos1_enable		BOOL		
VAR	actpos1		MC_ReadActualPosition		
VAR	actpos1_vaild		BOOL		
VAR	actpos1_busy		BOOL		
VAR	actpos1_position		LREAL		

- **LD**



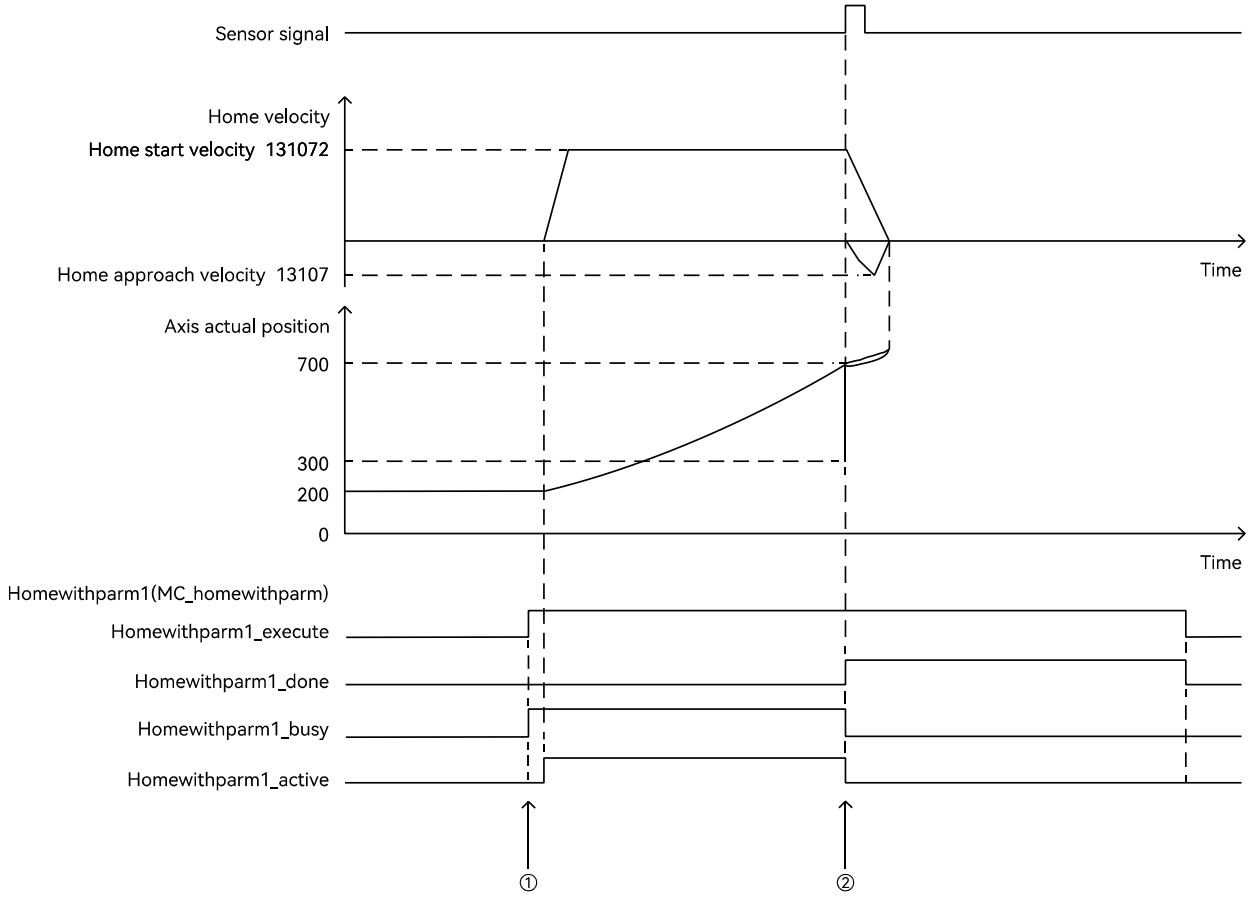
- **ST**

```
MC_Power0(
        Axis:=1 ,
        Enable:=power_eb ,
        EnablePositive:=TRUE ,
        EnableNegative:=TRUE ,
        BufferMode:=mcAborting ,
        Status=>axis1_power);

 homewithparm1 (
        Axis:=1 ,
        Execute:= axis1_power AND homewithparm1_execute ,
        HomeMode:=19,
        Position:=0 ,
        SpeedToSwitch:=131072,
        SpeedToZero:=13107,
        HomeAcc:=1000,
        Done=> homewithparm1_done ,
        Busy=> homewithparm1_busy ,
        Active=> homewithparm1_active );

actpos1(
        Axis:=1 ,
        Enable:=axis1_power AND actpos1_enable ,
        Valid=>actpos1_vaild ,
        Busy=>actpos1_busy ,
        Position=>actpos1_position);
```

- **Program Description**

  - After enabling the axis, HomeWithParm1_ execute becomes TRUE and is executed. Then, the servo drive works in the home mode and uses the home function to control the homing of the axis.

- During the process of axis homing, when encountering the home signals (external sensor signal connected to the input of the servo drive), the axis decelerates and stops, then reverses. When encountering the home signal changing from TRUE to FALSE during reversal, the current position of the axis is set to the home position set by the parameter specified home setting command (HomeWithParm1).

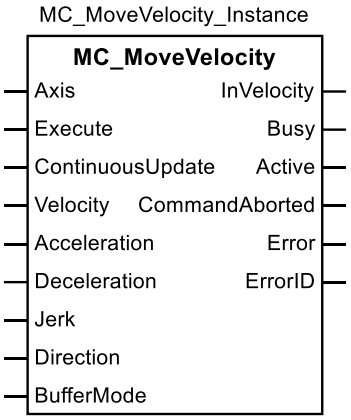


① When homewithparm1 execute becomes TRUE, the home setting instruction (homewithparm 1) starts to execute, controlling the servo drive to work in the home mode utilizing the servo drive's home function control.

② When the axis encounters the home signal and decelerates and stops, then reverses after stopping, and when the home signal changes from TRUE to FALSE after reversing, the current position of the axis is set to the home position set in the home instruction.

4.7 MC_MoveVelocity (velocity)

Using the position control mode of the drive, the controller transmits the position to the drive at a set period based on input variables, simulating speed control. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveVelocity	Velocity	FB		MC_MoveVelocity_Instance (Axis :=parameter, Execute :=parameter, ContinuousUpdate :=parameter , Velocity :=parameter, Acceleration :=parameter , Deceleration :=parameter, Jerk :=parameter , Direction :=parameter, BufferMode :=parameter , InVelocity => parameter, Busy => parameter , Active => parameter , CommandAborted => parameter, Error => parameter, ErrorID => parameter);

■ **Input variable**

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
ContinuousUpdate	Continuous updates	BOOL	TRUE or FALSE	FALSE	Reserved
Velocity	Target velocity	LREAL	0, positive numbers	Required field	Specify target velocity *1 (Unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive numbers	Required field	Specify acceleration*1 (Unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive numbers	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive numbers	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
Direction	Direction	MC_Direction	1：mcPositiveDirection 3：mcNegativeDirection 4：mcCurrentDirection	1	Set the instruction to control the direction of axis operation 1: Positive direction 3: Reverse direction 4: Operate in the current direction (when the axis is stationary, it moves in the positive direction)
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered 2：mcBlendingLow 3：mcBlendingPrevious 4：mcBlendingNext 5：mcBlendingHigh	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered 2: buffer at low velocity 3: buffer at the previous velocity 4: buffer at the next velocity 5: buffer at low velocity

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions ".*

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions"*.
*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis"*.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Invelocity	Target velocity reached	BOOL	TRUE or FALSE	TRUE when the axis instruction velocity reaches the target velocity
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Invelocity	When the axis velocity reaches the target velocity	◆ When Error becomes TRUE ◆ When CommandAborted become TRUE
Busy	When Execute changes to TRUE	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Active	When the instruction is started	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When an instruction is executed and Execute is FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE and after one period, it becomes FALSE
Error	The variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from TRUE to FALSE

■ Function description

● Basic function description

Using the position control mode of the drive, the controller transmits the position to the drive at a set period based on input variables, simulating velocity control. This instruction starts executing when Execute changes from FALSE to TRUE; The instruction will change the velocity of the axis based on the set value of the input variable, accelerating or decelerating from the current velocity to the set velocity.

● Direction selection

This instruction controls the direction of the axis during operation, which is specified by the Direction. When the value of Direction is 1, it moves in the positive direction; When the value of Direction is 3, the axis moves in the opposite direction; When the value of Direction is 4, the axis continues to move in the current operating direction. When the value of Direction is 4, if other instructions control the axis to move in the positive direction, the velocity instruction interrupts other instructions and still moves in the positive direction; When other instructions control the axis to move in the opposite direction, the velocity

instruction interrupts other commands and still moves in the opposite direction; When the axis is in a stopped state, it moves in the positive axial direction.

- **Re-execute the instruction**

  When an instruction is being executed and Execute changes from FALSE to TRUE, the instruction can be re-executed. The parameters that can be re-executed at this time include Velocity, Acceleration, Deceleration, Jerk, and Direction, while other parameters will not take effect. When there is a BufferMode relation between the velocity instruction and other motion instructions, changing the parameters of the velocity instruction will take effect after being re-executed. The original relay relation will still be maintained, and the handover velocity will be recalculated.

- **Execute this instruction while other instructions are in processing**

  When other motion instructions are executed, starting this instruction can switch or buffer to it. The buffer of this instruction and other motion instructions is determined by the value of the input variable BufferMode, and the values that can be set for the input variable BufferMode of this instruction are related to the executing motion instruction.

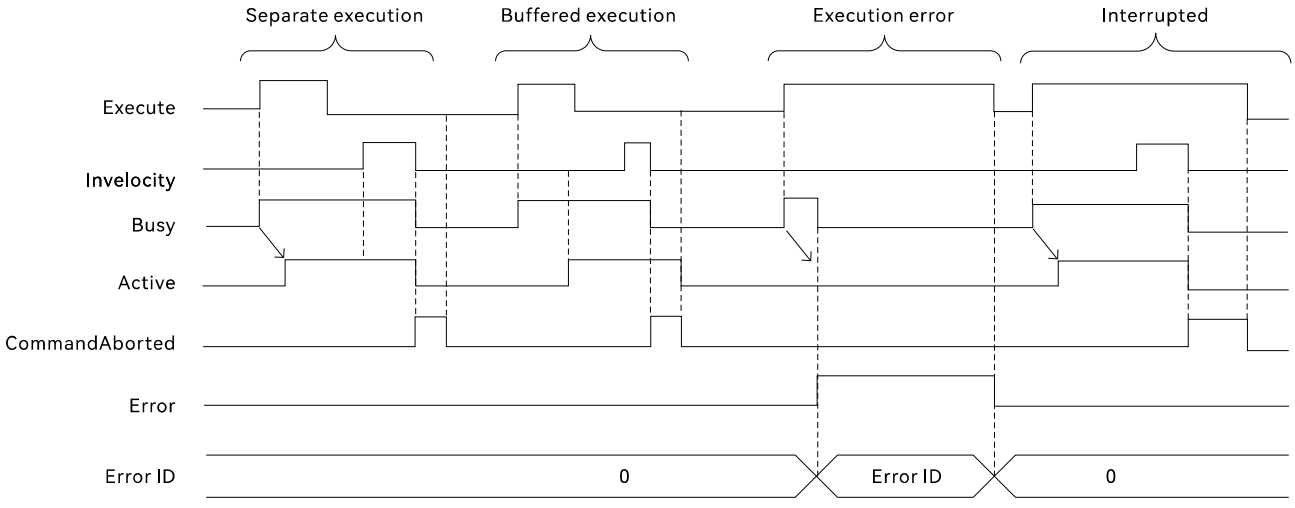
- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, it starts other motion instructions. The bufferMode parameter values of other motion instructions and how the instruction is buffered are determined. The bufferMode values of other motion instructions can only be selected as aborted or wait; If other motion commands do not have the BufferMode parameter, the instruction is usually aborted. When other motion instructions and the instruction are buffered, the timing for the execution of other instructions is when InVelocity becomes TRUE.

- **Abnormality elimination**

  When the instruction is executed, if the input variable is illegal, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this command is executed, if an exception is encountered (such as an axis alarm), the command will also report an error and the axis will immediately stop.

■ **Output variable timing diagram description**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. When the axis velocity reaches the target velocity, Inelocity becomes TRUE, Busy and Active remain TRUE.

- **Buffered execution**

  When other instructions control the axis, execute this instruction (BufferMode value is not mcAborting). When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE when the previous instruction is completed.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, and the Execute instruction changes from FALSE to TRUE, Busy becomes TRUE, Error becomes TRUE, Busy becomes FALSE, and ErrorID outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

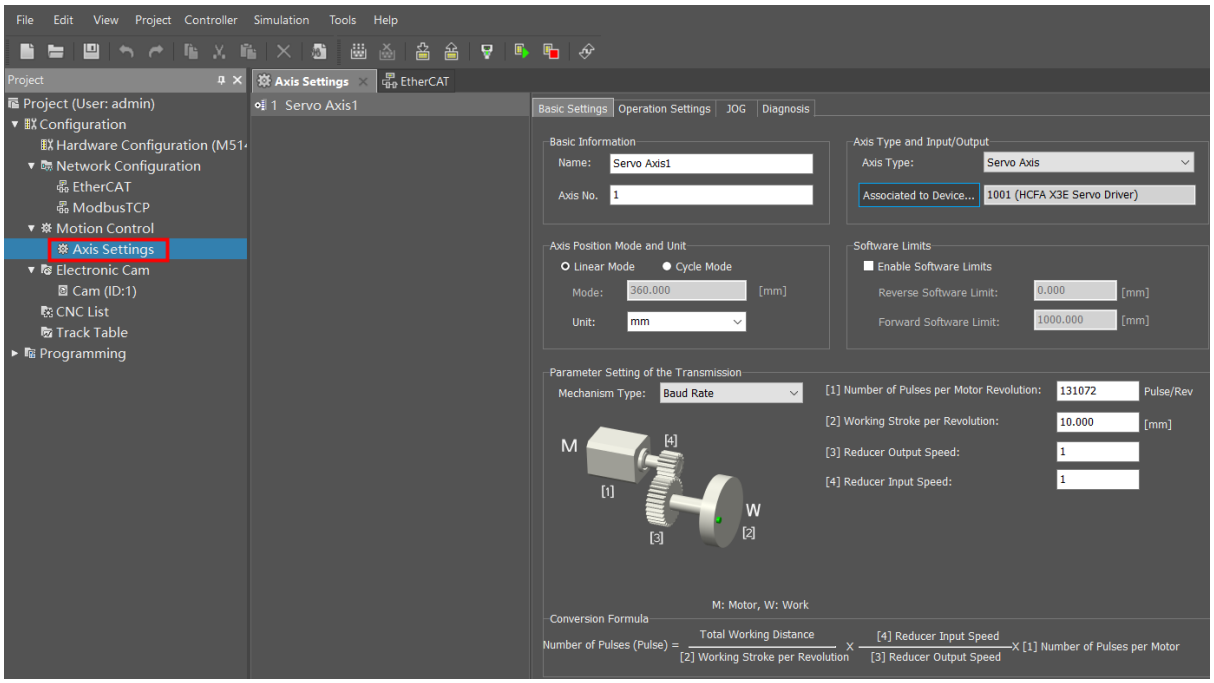
■ Example program

- **Functionality**

  Real time change of the target velocity of the axis. When both the velocity instruction and velocity ratio setting instruction are executed, the target velocity of the axis can be modified by changing the target velocity ratio. The function implemented by this instruction can also be achieved by executing MC_ The MoveContinuusVelocity instruction to modify the target velocity of the axis in real time.

- **Axis parameter setting**

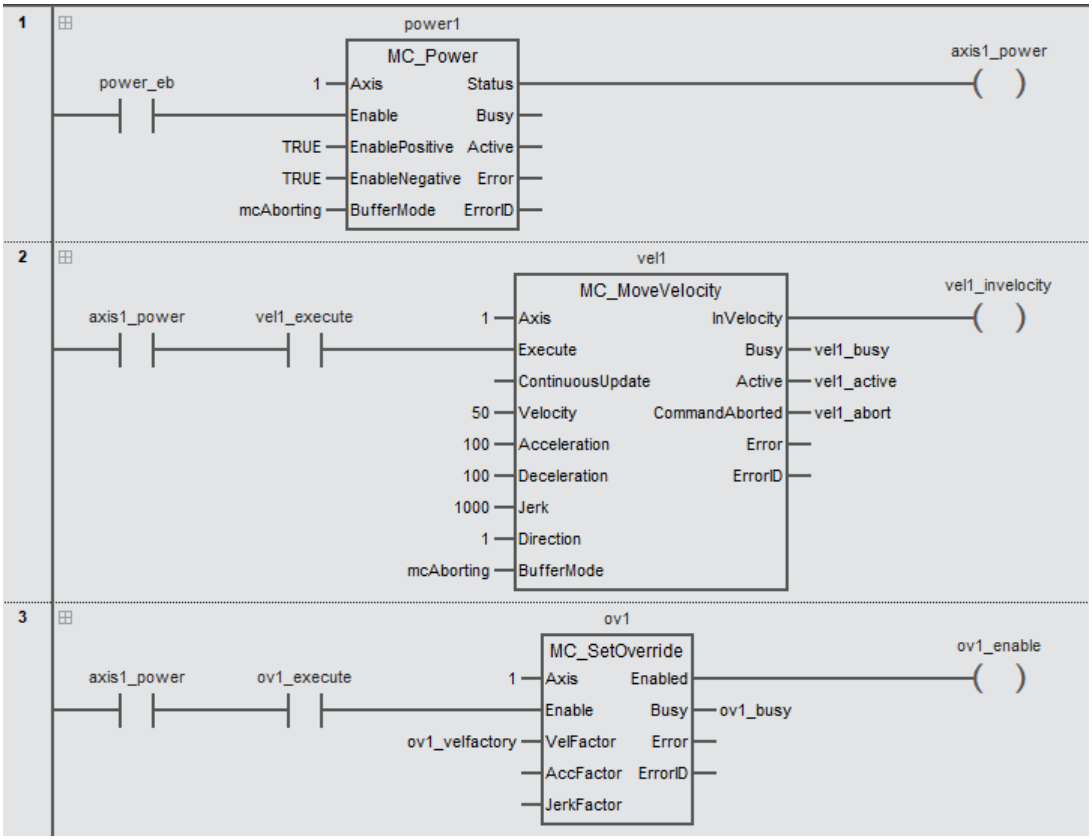
  The parameter settings for axis 1 are shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	vel1_execute		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_ invelocity		BOOL		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	ov1		MC_SetOverride		
VAR	ov1_execute		BOOL		
VAR	ov1_enable		BOOL		
VAR	ov1_velfactory		LREAL	100	
VAR	ov1_busy		BOOL		

- **LD**

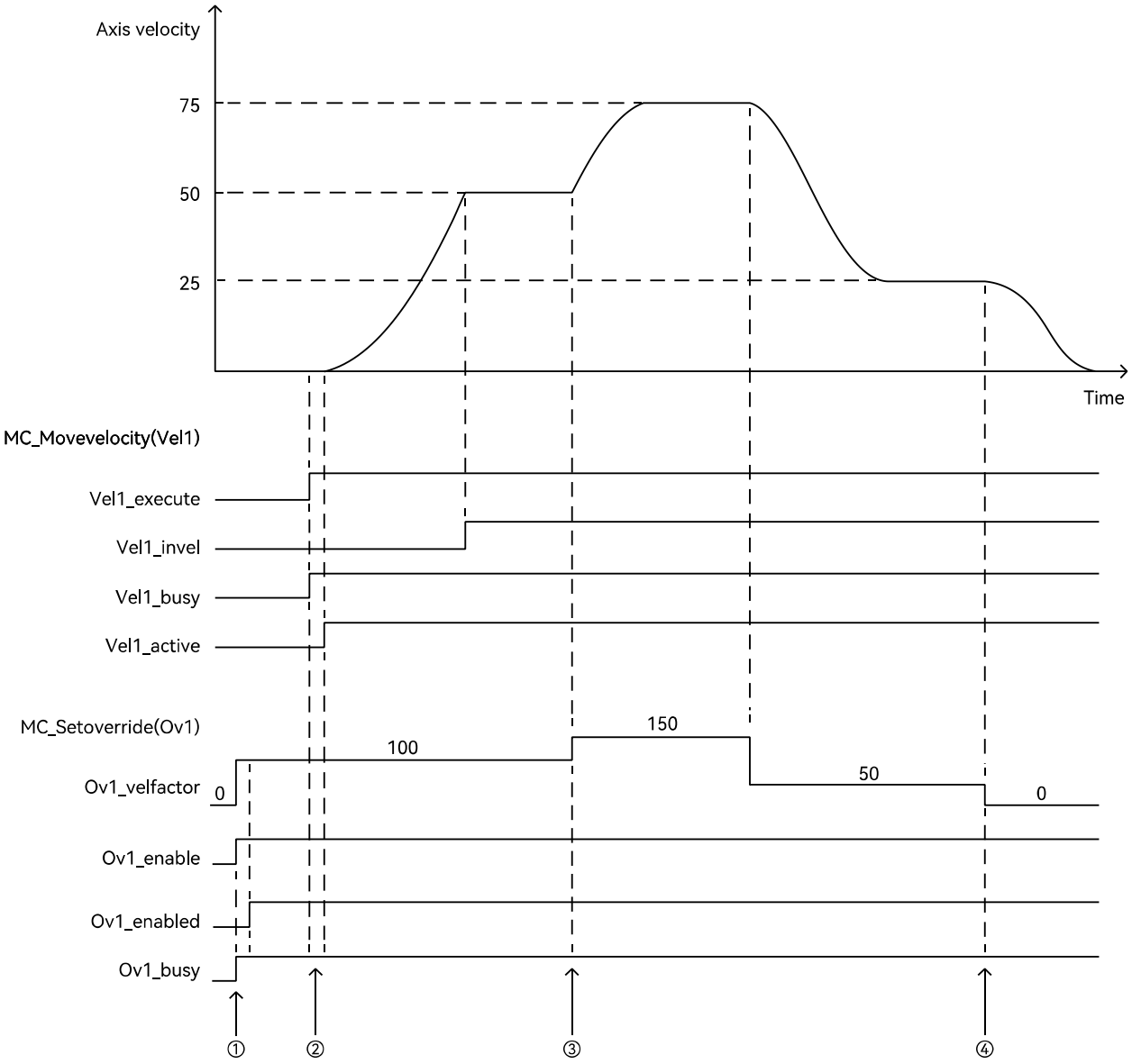


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);

vel1(
    Axis:=1 ,
    Execute:= axis1_power AND vel1_execute ,
    Velocity:=50 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Direction:=1 ,
    BufferMode:=mcAborting ,
    InVelocity=> vel1_invelocity ,
    Busy=> vel1_busy ,
    Active=> vel1_active ,
    CommandAborted=> vel1_abort );

ov1(
    Axis:=1 ,
    Enable:=axis1_power AND vel1_execute ,
    VelFactor:=ov1_velfactory ,
    Enabled=>ov1_enable ,
    Busy=>ov1_busy);
```

- Program description

  - After enabling the axis, ov1_ Enable to TRUE, execute the velocity ratio setting instruction (ov1) first, and the target velocity ratio value is 100%. Vel1_ When execute becomes TRUE, the velocity instruction (vel1) begins to be executed, and the second period of velocity instruction execution begins to control the axis.
  - If users want to change the target velocity of the axis during operation, they can change the target velocity ratio value setting instruction. As shown in the figure below, when the target velocity ratio value changes from 100 to 150, the target velocity of the axis becomes 75. The calculation method is: the target velocity of the velocity command (50) * 150%=75. After the target velocity ratio value changes, the axis accelerates or decelerates according to the input parameters set by the velocity command. When the target velocity ratio value becomes 0, the axis target velocity becomes 0, and the axis decelerates to 0 according to the input parameter set by the velocity command.



① The speed ratio setting command is executed first after the axis is enabled, and the target velocity ratio is 100.

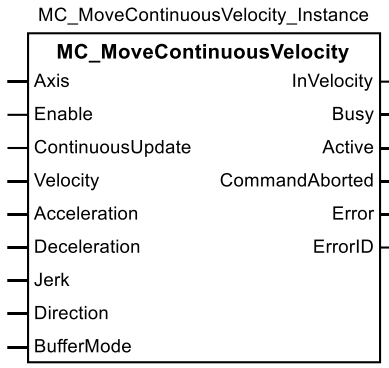
② The velocity command is executed and will control the axis in the 2nd cycle.

③ Changing the target velocity ratio to 150 which changes the target velocity of the axis to the target velocity of the speed command multiplied by 150%. The axis accelerates to the new target velocity with the input parameters set by the velocity command.

④ Changing the target velocity ratio to 0 which changes the target velocity of the axis to 0 and the axis decelerates to 0 with the input parameter set by the velocity command.

4.8 MC_MoveContinuousVelocity（real-time velocity modification）

This instruction is used to control the specified axis to accelerate or decelerate to a set velocity and perform uniform motion according to the set acceleration rate. Library: MotionControl_ Part 2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveContinuousVelocity	Real-time velocity modification	FB		MC_MoveContinuousVelocity _Instance (Axis :=parameter, Enable :=parameter , ContinuousUpdate :=parameter , Velocity :=parameter , Acceleration :=parameter, Deceleration :=parameter, Jerk :=parameter , Direction :=parameter, BufferMode :=parameter , Invelocity => parameter , Busy => parameter, Active => parameter , CommandAborted => parameter, Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Enable	Effective	BOOL	TRUE or FALSE	FALSE	Execute this instruction when Enable is TRUE
ContinuousUpdate	Continuous updates	BOOL	TRUE or FALSE	FALSE	Reserved
Velocity	Target velocity	LREAL	Positive numbers	Required field	Specify target velocity *1 (Unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive numbers	Required field	Specify acceleration*1 (Unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive numbers	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive numbers	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
Direction	Direction	MC_Direction	1：mcPositiveDirection 3：mcNegativeDirection 4：mcCurrentDirection	1	Set the instruction to control the direction of axis operation 1: Positive direction 3: Reverse direction 4: Operate in the current direction (when the axis is stationary, it moves in the positive direction)
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered 2：mcBlendingLow 3：mcBlendingPrevious 4：mcBlendingNext 5：mcBlendingHigh	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered 2: buffer at low velocity 3: buffer at the previous velocity 4: buffer at the next velocity 5: buffer at low velocity

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions "*.

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions"*.
*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis"*.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Invelocity	Target velocity reached	BOOL	TRUE or FALSE	TRUE when the axis command velocity reaches the target velocity
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Invelocity	When the axis velocity reaches the target velocity	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Busy	When Execute changes to TRUE	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Active	When the instruction is started	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When an instruction is executed and Execute is FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE and after one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from TRUE to FALSE

■ Function description

● Basic function description

Using the position control mode of the drive, the controller transmits the position to the drive at a set period based on input variables to simulate velocity control. This instruction is executed when Enable is TRUE. Based on the set value of the input variable, the instruction controls the axis to accelerate or decelerate from the current velocity to the set target velocity for uniform motion. When Enable is TRUE, changing the values of the input variables Velocity, Acceleration, Deceleration, Jerk , or Direction/ This instruction will immediately take effect without the need of re-execution.

● Direction selection

This instruction controls the direction of the axis during operation, which is specified by the Direction. When the value of Direction is 1, it moves in the positive axial direction; when the value of Direction is 3, the axis moves in the opposite direction; when the value of Direction is 4, the axis continues to move in the current operating direction. When the value of Direction is 4, if other instructions control the axis to move in the positive direction, the velocity instruction interrupts other instructions and still moves in the positive direction; When other instructions control the axis to move in the opposite direction, the velocity

instruction interrupts other instructions and still moves in the opposite direction; When the axis is in a stopped state, it moves in the positive axial direction.

- **Re-execute the instruction**

  When Enable is TRUE, execute the instruction; when Enable is FALSE, execute the instruction according to the input variable before Enable changes from TRUE to FALSE.

- **Execute this instruction while other instructions are in processing**

  When other motion instructions are executed, starting this instruction can switch or buffer it. The buffer of this instruction and other motion instructions is determined by the value of the input variable BufferMode, and the values that can be set for the input variable BufferMode of this instruction are related to the executing motion instruction.

- **Execute other instructions while this instruction is in processing**

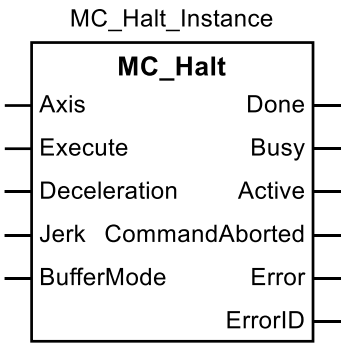
  When this instruction is executed, it starts other motion instructions. The bufferMode parameter values of other motion instructions and how the instruction is buffered are determined. The bufferMode values of other motion instructions can only be selected as interrupt or wait; If other motion instructions do not have the BufferMode parameter, the instruction is usually aborted. When other motion instructions and the instruction are buffered, the timing for the execution of other instructions is when InVelocity (target velocity reached) becomes TRUE.

- **Abnormality elimination**

  When the instruction is executed, if the input variable is illegal, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this command is executed, if an exception is encountered (such as an axis alarm), the command will also report an error and the axis will immediately stop.

4.9 MC_Halt（halt）

Control the specified axis to decelerate and stop from the current velocity. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_Halt	Halt	FB		MC_Halt_Instance (Axis :=parameter , Execute :=parameter , Deceleration :=parameter , Jerk :=parameter , BufferMode :=parameter, Done => parameter, Busy => parameter , Active => parameter , CommandAborted => parameter, Error => parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions ".*

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions".*

*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis".*

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	Deceleration completed, when the specified axis velocity becomes 0	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period, it becomes FALSE
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the axis is under control	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When the instruction is aborted by another instruction, CommandAborted becomes TRUE, and after one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When an exception is encountered during the execution of the instruction, Error becomes TRUE. After one period, it becomes FALSE

■ Function description

● Basic function description

- ➢ Control the specified axis to decelerate and stop from the current velocity, with the deceleration and jump set by the command input variables.
- ➢ When MC_ Halt instruction begins to execute, and the state machine enters DiscreteMotion. When the axis velocity drops to 0, Done becomes TRUE, and the state machine becomes Standstill status.
- ➢ Compared to the MC_Stop instruction, the MC_Halt instruction does not latch the axis, and the controller can execute other motion instructions. During the execution of the MC_Halt instruction, the axis can execute other motion instructions to interrupt the MC_Halt instruction while decelerating; After the MC_Halt instruction is executed and the axis has stopped, the controller can execute other motion instructions to restart the axis.

● Re-execute the instruction

When the instruction execution is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when an instruction is being executed and Execute changes from FALSE to TRUE again, it will not affect the execution of the instruction and will continue to be executed in the previous way.

● Execute this instruction while other instructions are in processing

When other motion instructions are executed, the instruction is initiated, and the way it and other motion instructions are buffered are determined by the value of the input variable BufferMode. The value that can be set for the input variable BufferMode in this instruction can only be set to mcAborting or mcBuffered.

● Execute other instructions while this instruction is in processing

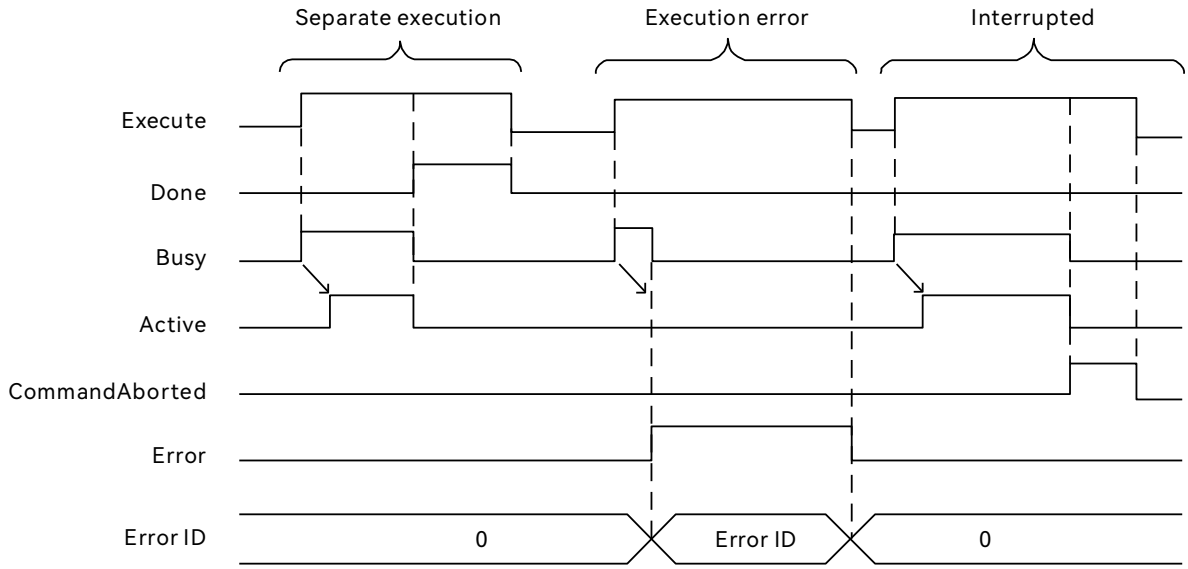
When this instruction is executed, other motion instructions are initiated, and how they are buffered is determined by the BufferMode parameter values of other motion instructions. When other motion

instructions and this instruction are buffered, the timing for the execution of other instructions is when Done becomes TRUE. If other motion commands do not have the BufferMode parameter, the instruction is usually aborted.

- **Abnormality elimination**

  When the instruction is executed, if the input variable of the instruction is not within the allowed range or does not meet the execution conditions of the instruction, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed, if an exception is encountered (such as an axis alarm), the instruction will also report an error and the axis will immediately stop.

■ Output variable timing diagram description



- **Separate execution**

  When Execute changes from FALSE to TRUE, and Busy also becomes TRUE, and Active becomes TRUE in the next period. After the instruction controls the axis to stop, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy (executing) becomes TRUE, Error becomes TRUE in the next period, Busy (executing) becomes FALSE, and ErrorID (executing) outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID (in error code). When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy, and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

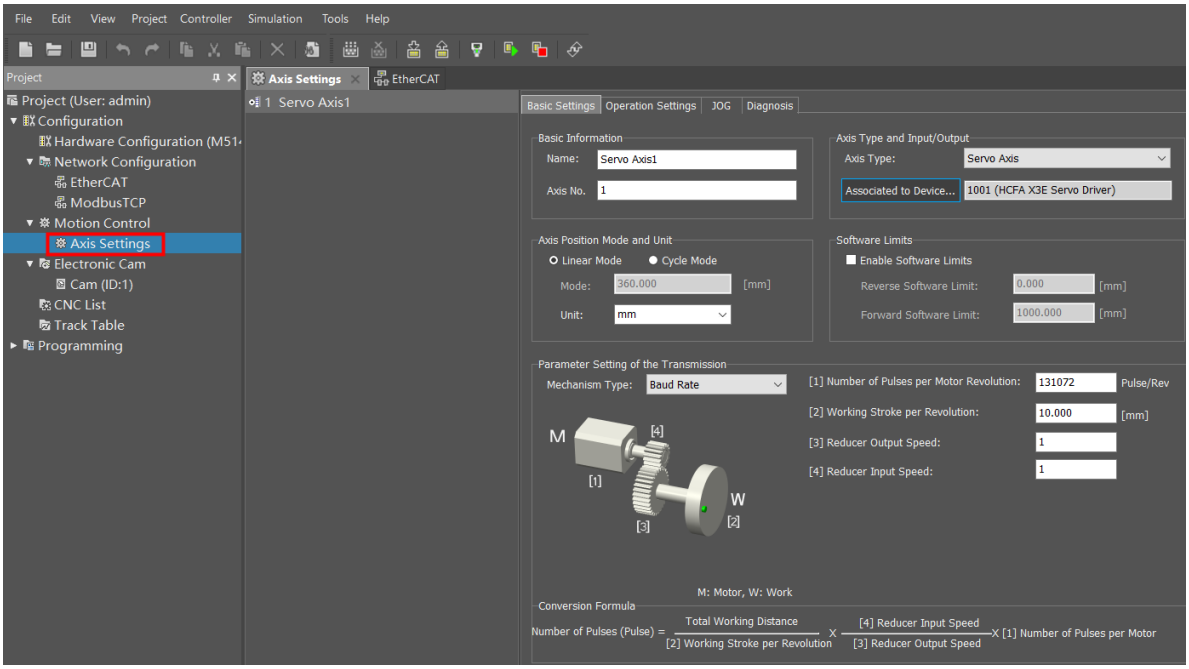
■ Example program

- **Functionality**

  When the velocity instruction controls the operation of the axis, execute MC_ Halt instruction controls the axis to decelerate and stop according to the input parameters set in this instruction.

- **Axis parameter setting**

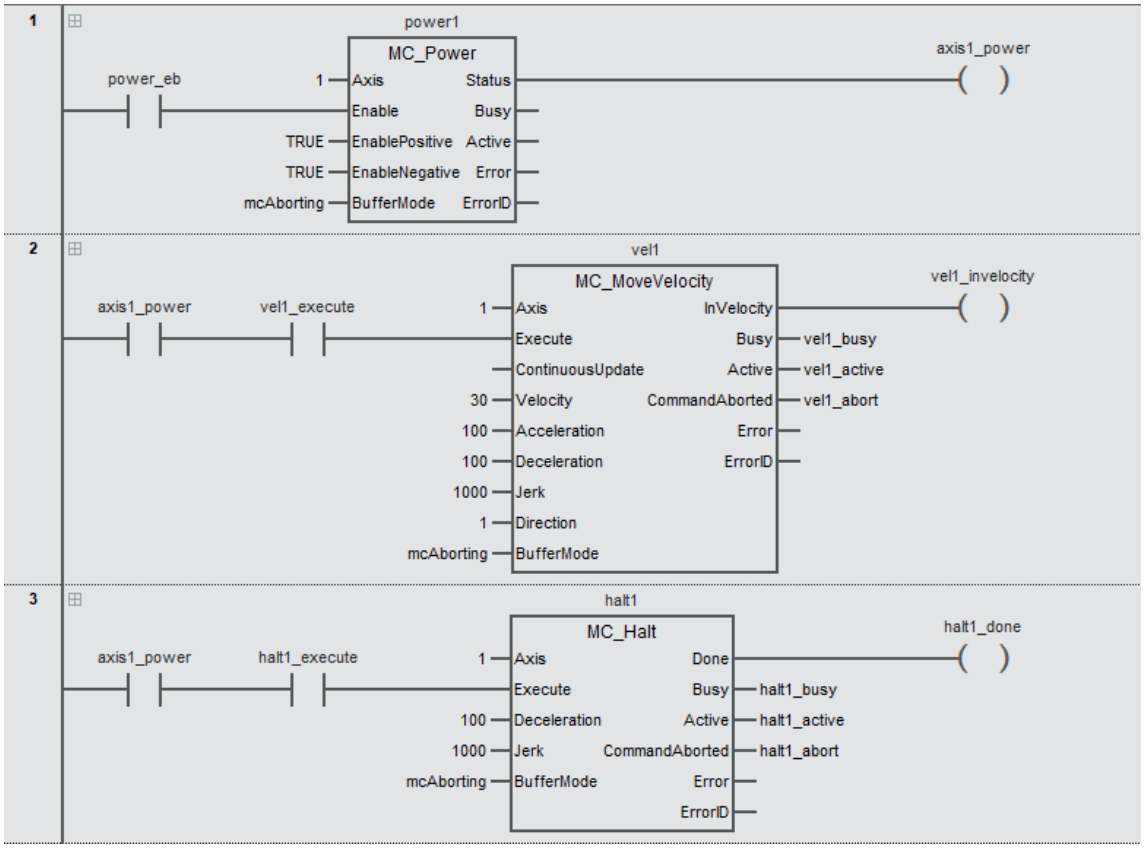
  The parameter setting for axis 1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	vel1_execute		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_invelocity		BOOL		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	halt1_execute		BOOL		
VAR	halt1		MC_Halt		
VAR	halt1_done		BOOL		
VAR	halt1_busy		BOOL		
VAR	halt1_active		BOOL		
VAR	halt1_abort		BOOL		

- LD



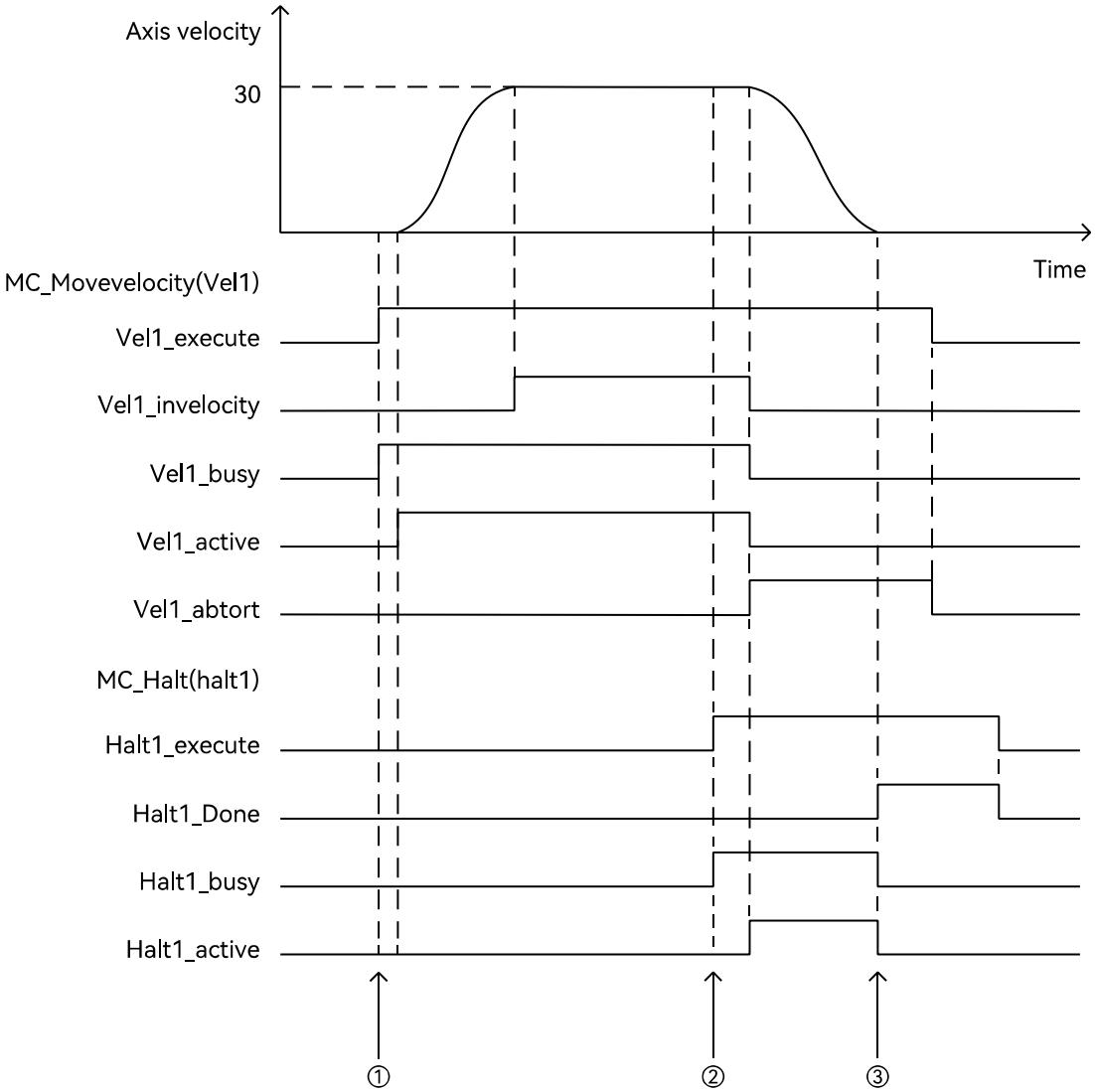
- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis 1_power);

vel1(
    Axis:=1 ,
    Execute:= axis 1_power AND vel1_execute ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Direction:=1 ,
    BufferMode:=mcAborting ,
    InVelocity=> vel1_invelocity ,
    Busy=> vel1_busy ,
    Active=> vel1_active ,
    CommandAborted=> vel1_abort );

halt1(
    Axis:=1 ,
    Execute:= axis 1_power AND halt_execute ,
    Deceleration:=100 ,
    Jerk:=1000 ,
```

```
BufferMode:= mcAborting ,
Done=> halt1_done ,
Busy=> halt1_busy ,
Active=> halt1_active ,
CommandAborted=> halt1_abort );
```

- **Program Description**

  - After enabling the axis, vel1_ execute becomes TRUE, execute the velocity instruction to control the axis operation. When the axis needs to stop operating, set halt1_ execute to TRUE and execute MC_ Halt instruction, control the axis to decelerate and stop according to the input parameters set by the Halt1 instruction, Halt1 instruction's Halt1_ When done becomes TRUE, the instruction velocity of the axis decreases to 0.



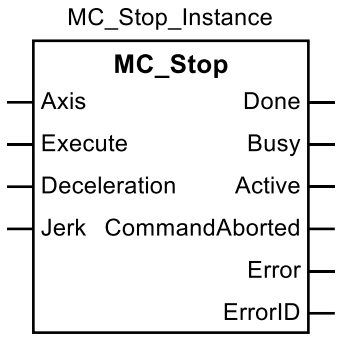
① The velocity command (vel1) starts execution and controls the axis from the 2nd cycle.

② The stop command (halt1) starts execution, and controls the axis to decelerate and stop according to the set input parameters in the 2nd cycle, which interrupts the execution of the velocity command.

③ At the completion of the stop command (halt1) (Done is TRUE), while Active and Busy become FALSE, the axis command velocity is reduced to 0.

4.10 MC_Stop（stop and latch）

Control the specified axis to decelerate and stop from the current velocity, and after stopping, it will be latched in a stopped state. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_Stop	Stop and latch	FB		MC_Stop_Instance (Axis :=parameter, Execute :=parameter, Deceleration :=parameter, Jerk :=parameter, Done => parameter , Busy => parameter , Active => parameter, CommandAborted => parameter, Error => parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute this instruction when Execute is TRUE
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions ".*

*2: For details of the instruction units, please refer to *"Parameter unit of motion control instructions".*

■ Output variable

Name	Meaning	Data type	Valid range	Default
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	When an instruction is aborted, it becomes TRUE
Error	Error	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	Deceleration completed, when the specified axis velocity becomes 0	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one period, it becomes FALSE
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE

Active	When the axis is under control	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the execution of an instruction is aborted	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When the instruction is aborted by another instruction, CommandAborted becomes TRUE, and after one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When an exception is encountered during the execution of the instruction, Error becomes TRUE. After one period, it becomes FALSE

■ Function description

● Basic function description

- ➢ Control the designated axis to decelerate and stop from the current velocity. The deceleration and jump are set by the command input variables. After stopping, the axis state will be locked in the stopped state.
- ➢ After the instruction is executed, as long as the input variable Excel is set to TRUE, the axis state will remain in the Stopping state. When this instruction is executed, only another instantiated MC is present_ Stop can interrupt this command, and executing other motion commands will report an error.
- ➢ After the execution of this instruction, when the axis stops and the input variable Excel is FALSE, the axis state becomes the standstill state, and other motion instructions can be executed at this time.

● Re-execute the instruction

When the instruction is in execution and Execute changes from FALSE to TRUE again, it will not affect the execution of the instruction and it will continue to be executed in the previous way.

● Execute this instruction while other instructions are in processing

When other motion instructions are executed, starting this instruction will interrupt the other motion instructions that are being executed.

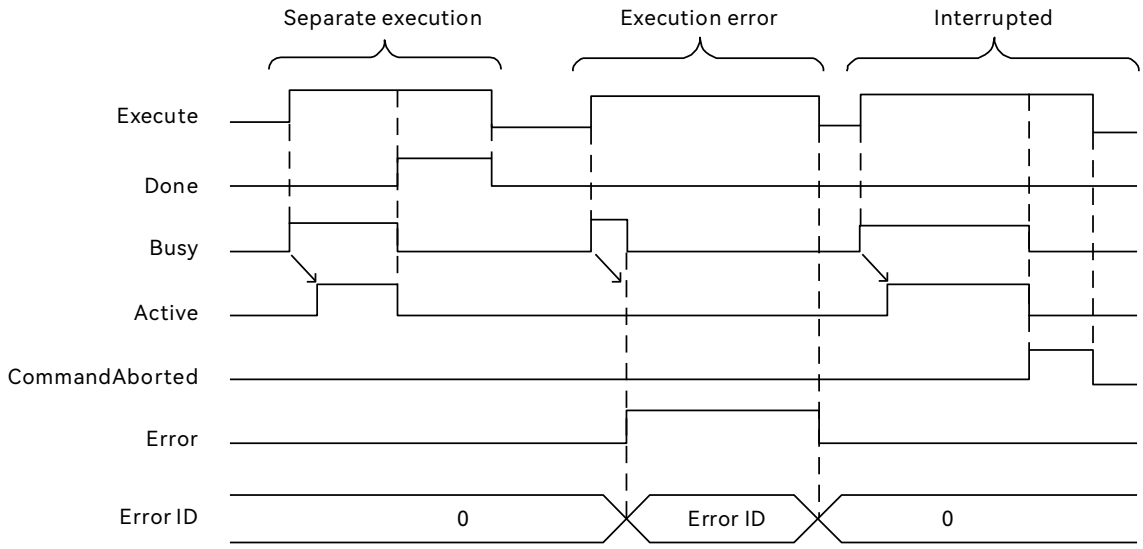
● Execute other instructions while this instruction is in processing

After the execution of this instruction, as long as the input variable Excel is TRUE, the axis state will remain in the Stopping state. When this instruction is executed, only another instantiated MC_Stop can interrupt this instruction, and executing other motion commands will report an error.

● Abnormality elimination

When the instruction is executed, if the input variable value of the instruction is not within the allowed range or does not meet the execution conditions, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed, if an exception is encountered (such as an axis alarm), the command will also report an error and the axis will immediately stop.

■ Output variable timing diagram description



- **Separate executions**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. After the command controls the axis to stop, Done becomes TRUE, Busy and Active remain TRUE. When Execute becomes FALSE, Done, Busy, and Active become FALSE simultaneously.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute command changes from FALSE to TRUE, Busy (executing) also becomes TRUE, and the next period Error (error) becomes TRUE. The corresponding error code is outputted by the ErrorID (error code), which can be used to find the cause of the problem.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

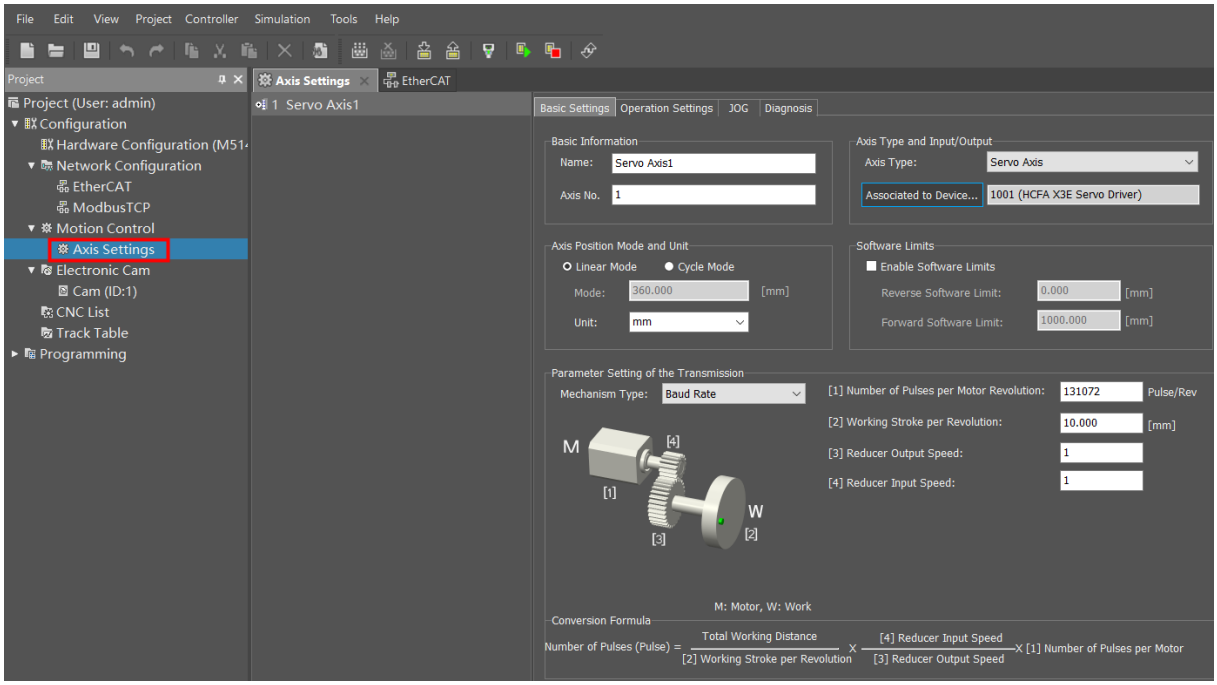
■ Example program

- **Functionality**

  When the velocity instruction controls the operation of the axis, execute MC_ Stop instruction controls the axis to decelerate and stop according to the input parameters set in this command. When MC_ Stop instruction is executed, executing other motion instructions will result in an error. If users need to execute other motion instructions (not MC_Stop commands), they need to set execute of the MC _ Stop instruction becomes FALSE before executing other motion instructions.

- **Axis parameter setting**

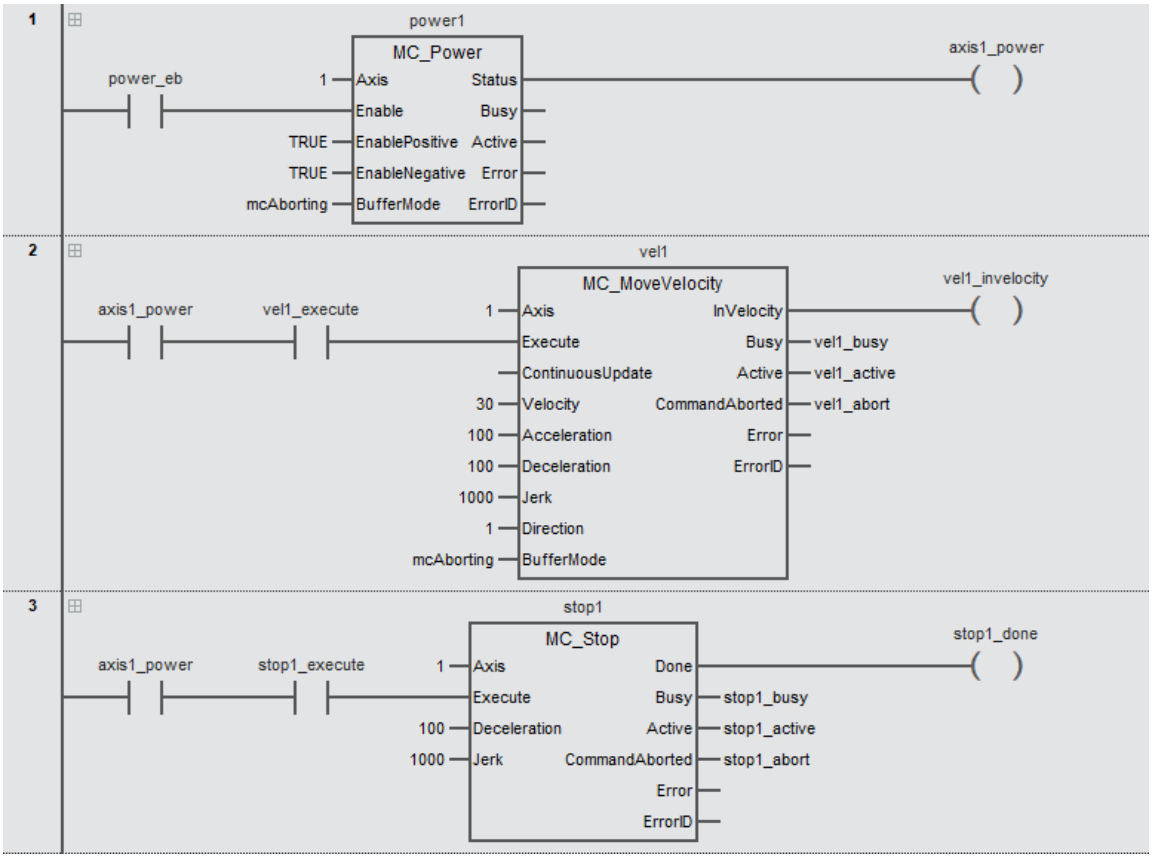
  The parameter setting for axis 1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	vel1_execute		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_ invelocity		BOOL		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	stop1_execute		BOOL		
VAR	stop1		MC_Stop		
VAR	stop1_done		BOOL		
VAR	stop1_busy		BOOL		
VAR	stop1_active		BOOL		
VAR	stop1_abort		BOOL		

- LD

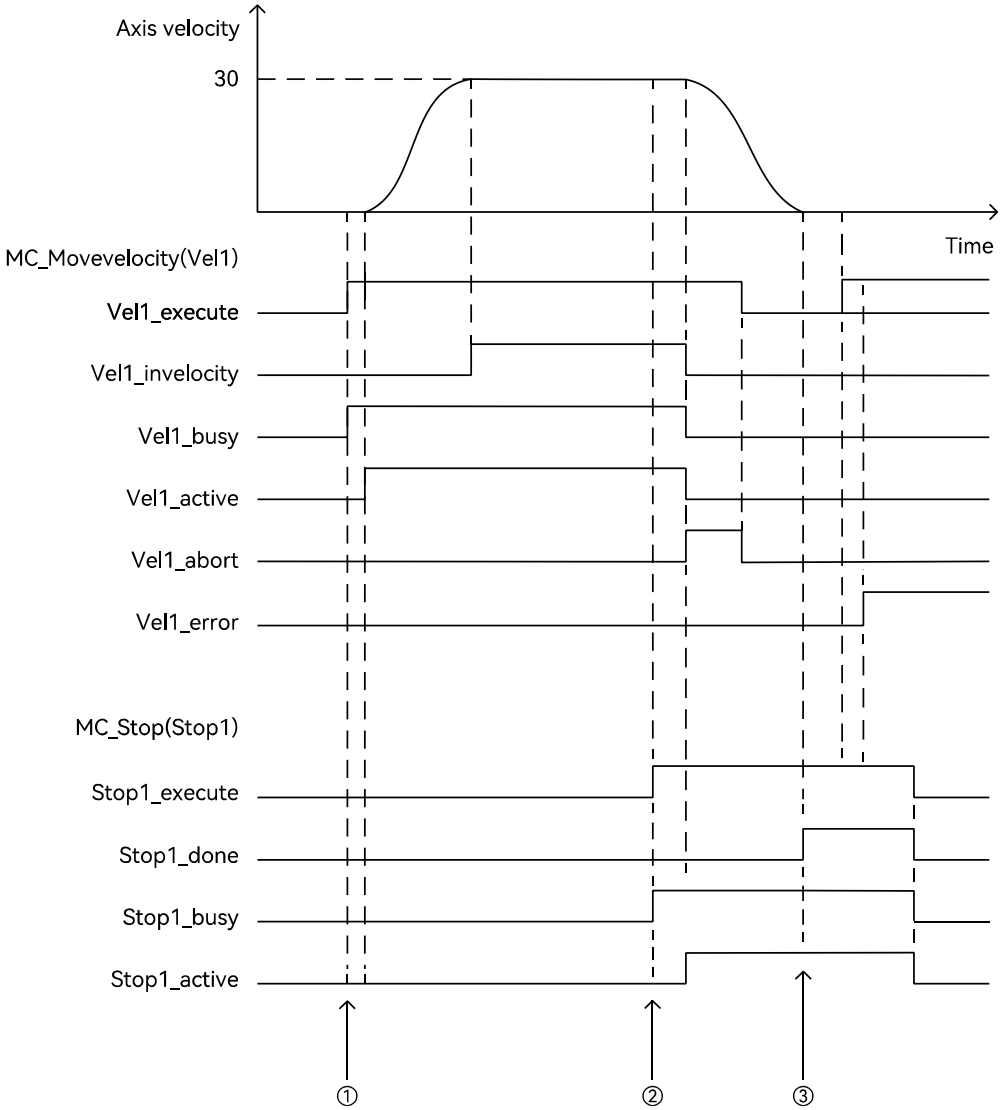


- ST

```
power1 (
        Axis:=1 ,
        Enable:=power_eb ,
        EnablePositive:=TRUE ,
        EnableNegative:=TRUE ,
        BufferMode:=mcAborting ,
        Status=>axis 1_power);

vel1 (
        Axis:=1 ,
        Execute:= axis 1_power AND vel1_execute ,
        Velocity:=30 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        Direction:=1 ,
        BufferMode:=mcAborting ,
        InVelocity=> vel1_invelocity ,
        Busy=> vel1_busy ,
        Active=> vel1_active ,
        CommandAborted=> vel1_abort);

stop1(
        Axis:=1 ,
        Execute:= axis 1_power AND stop1_execute ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        Done=> halt1_done ,
        Busy=> halt1_busy ,
        Active=> halt1_active ,
```

CommandAborted=> halt1_abort);

- **Program description**
  - After enabling the axis and vel1_ execute becomes TRUE, execute the velocity instruction to control the axis operation. When the axis needs to be stopped, set stop 1_ execute to TRUE to execute MC_ Stop instruction, the control axis decelerates and stops according to the input parameters set by the stop1 instruction, and the stop_done of the stop1 instruction becomes TRUE, the instruction velocity of the axis drops to 0, and the Busy and active of the instruction remain TRUE.
  - When the MC_ Stop instruction is executed, only another MC can be executed. The Stop instruction interrupts the current MC_ Stop instruction; If other motion instructions are executed (not MC_Stop), other motion instructions will report an error. If other motion instructions need to be executed, it is necessary to change the execution of the MC_Stop instruction to FALSE before executing other motion instructions.



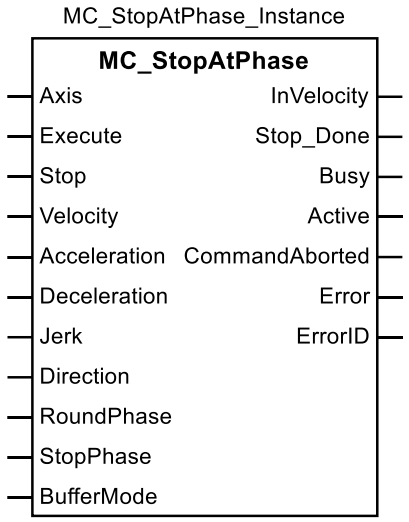
① The velocity command (vel1) is executed and controls the axis in the 2nd cycle.

② The stop instruction (stop1) is executed, and in the 2nd cycle, the control axis decelerates and stops according to the set input parameters, and interrupts the execution of the velocity command.

③ When the stop instruction (stop1) is completed (Done is TRUE), Active and Busy are still TRUE; if the velocity command is executed at this time, an error will be reported, and it is necessary to change the execute bit of the stop command to FALSE before executing other motion commands.

4.11 MC_StopAtPhase（stop at the specified phase）

Stop at the specified phase when controlling the operation of the specified axis. Library: MotionControl_ Part 2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_StopAtPhase	Stop at the specified phase	FB		MC_StopAtPhase_Instance (Axis :=parameter , Execute :=parameter , Stop :=parameter, Velocity :=parameter, Acceleration :=parameter , Deceleration :=parameter , Jerk :=parameter , Direction :=parameter , RoudPhase :=parameter , StopPhase :=parameter , BufferMode :=parameter , InVelocity => parameter , Stop_Done => parameter , Busy => parameter, Active => parameter , CommandAborted => parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	When the rising edge of this parameter is detected, control the specified axis to accelerate to the target velocity set by Velocity according to the set input variable and continue to operate
Stop	Stop	BOOL	TRUE or FALSE	FALSE	When the rising edge of this parameter is detected, the specified axis begins to decelerate and stop, with the final stop position being the phase specified by StopPhase
Velocity	Target velocity	LREAL	Positive number	Required field	Specify target velocity *1 (Unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Specify acceleration*1 (Unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
Direction	Direction	MC_Direction	1: mcPositiveDirection 3: mcNegativeDirection	1	When the rising edge of the Execute parameter is detected, the direction of axis operation 1: Positive direction 3: Negative direction
RoundPhase	Round phase	LREAL	Positive number	Required field	The value of StopPhase, which is used to calculate the period of StopPhase, is calculated by dividing the axis command position by RoundPhase

StopPhase	Stop phase	LREAL	0~RoundPhase setting value	0	The value of the stopping phase of the specified axis is calculated by dividing the axis command position by the RoundPhase (period)
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered 2：mcBlendingLow 3：mcBlendingPrevious 4：mcBlendingNext 5：mcBlendingHigh	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered 2: buffer at low velocity 3: buffer at the previous velocity 4: buffer at the next velocity 5: buffer at low velocity

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions"*.

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions"*.

*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis"*.

■ Output variable

Name	Meaning	Data type	Valid range	Default
Invelocity	Target velocity reached	BOOL	TRUE or FALSE	TRUE when the axis instruction velocity reaches the target velocity
Stop_Done	Reaching the specified phase	BOOL	TRUE or FALSE	TRUE when the specified axis position reaches the specified phase
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the instruction controls the axis
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when an instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Invelocity	When the target velocity is reached	◆ When CommandAborted is TRUE ◆ When Error is TRUE ◆ When Execute changes from FALSE to TRUE
Stop_Done	When the specified stop phase reaches	◆ When CommandAborted is TRUE ◆ When Error is TRUE ◆ When Stop is FALSE
Busy	When Execute changes to TRUE	◆ When Stop_Donne changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period, and then becomes FALSE

Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE

■ Function description

● Basic function description

➢ This instruction is used to control the specified axis to stop at the specified phase during operation. When the execution changes from FALSE to TRUE, this instruction takes the set value of Velocity as the target velocity, accelerates or decelerates to the target velocity according to the value set by the input variable. After reaching the target velocity, the output variable Invercity is set to TRUE and runs at a constant velocity. When Stop changes from FALSE to TRUE, it decelerates according to the value set by the input variable and ultimately stops at the phase set by StopPhase. But the specified axis may stop at the set phase after one revolution, or it may stop at the set phase after two revolutions. This instruction ensures an accurate phase, and the number of revolutions to reach the set phase and the velocity and input variables of the specified axis when Stop is TRUE are related.

➢ The input variables for this instruction, StopPhase and RoundPhase, are measured in travel unit, which are the same as the unit for 【working travel per revolution】 in the software. The value of StopPhase after the specified axis stops is equal to the remainder of the axis command position for RoundPhase. When using this instruction, the RoundPhase and the "working travel per revolution" in the software are generally set to the same value.

➢ As shown in the figure below, the value of StopPhase is 180 and the value of RoundPhase is 360. After executing this instruction, if the axis position is between 0 and 180, Stop _ Done will change from FALSE to TRUE. If the axis stops at position 180 and the axis position is between 180 and 540, Stop_ Done will change from FALSE to TRUE, and the stop position will be 540.

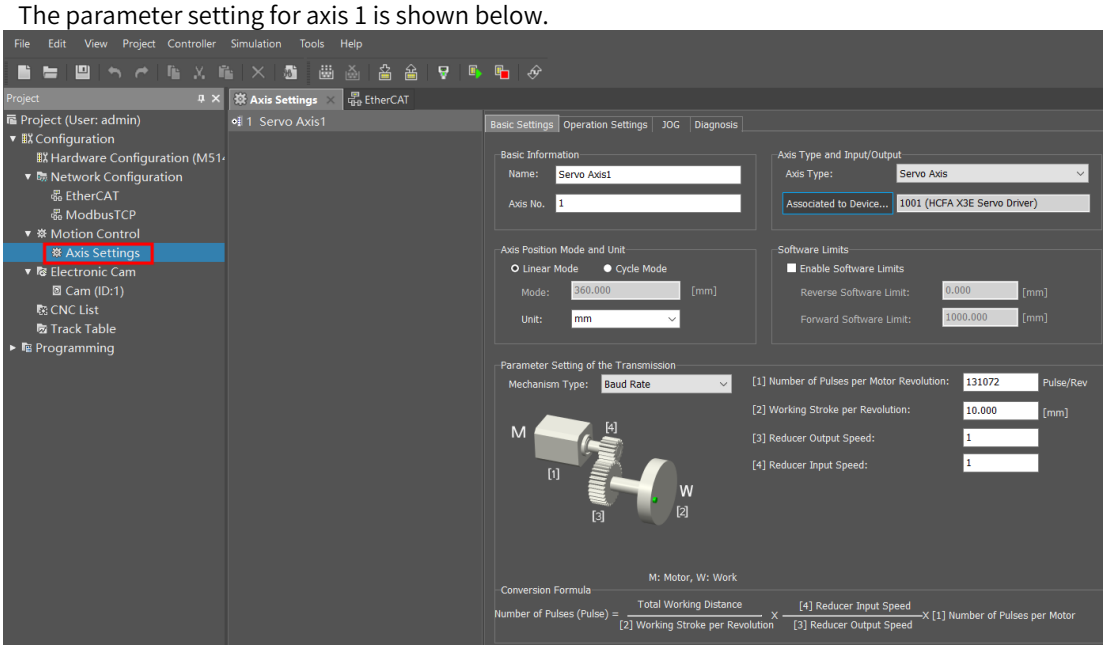
■ Example program

● Functionality

Execute MC_ StopAtPhase instruction controls the axis to run at a certain velocity and stop at the set phase.

● Axis parameter setting

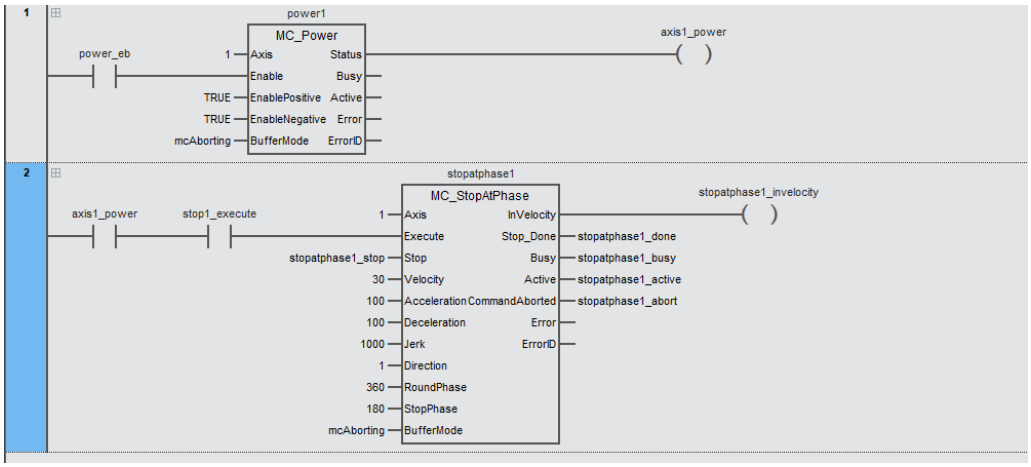
The parameter setting for axis 1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	stopatphase1_execute		BOOL		
VAR	stopatphase1		MC_StopAtPhase		
VAR	stopatphase1_invelocity		BOOL		
VAR	stopatphase1_stop		BOOL		
VAR	stopatphase1_done		BOOL		
VAR	stopatphase1_busy		BOOL		
VAR	stopatphase1_active		BOOL		
VAR	stopatphase1_abort		BOOL		

- **LD**



- **ST**

```
power1
    (
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);

stopatphase1
    (
    Axis:=1 ,
    Execute:= axis1_power AND stopatphase1_execute ,
    Stop:=stopatphase1_stop
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Direction:=1 ,
    RoundPhase:=360,
    StopPhase:=180,
```

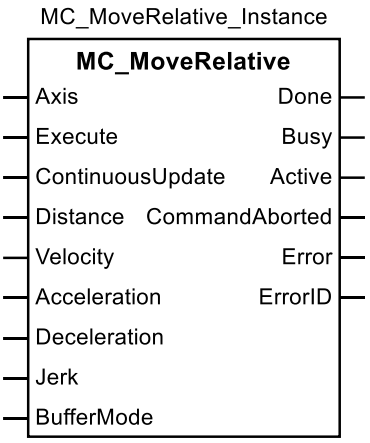
```
BufferMode:=mcAborting ,
InVelocity=> stopatphase1_invelocity,
Done=> stopatphase1_done
Busy=> stopatphase1_busy ,
Active=> stopatphase1_active ,
CommandAborted=> stopatphase1_abort);
```

- **Program description**

After enabling the axis, control the axis operation when stopathhase1_ execute becomes TRUE. When the axis needs to stop at the set phase, set stopathhase1_ stop to TRUE, the control axis decelerates and stops according to the input parameters set by the stopathase1 instruction, the stopathase1_done of halt1 becomes TRUE, and the instruction velocity of the axis decreases to 0.

4.12 MC_MoveRelative（relative movement）

Control the specified axis to move the specified distance with the current position as the reference point. The movement distance is specified by Distance. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveRelative	Relative movement	FB		MC_MoveRelative_Instance (Axis :=parameter , Execute :=parameter , ContinuousUpdate :=parameter, Distance :=parameter , Velocity :=parameter, Acceleration :=parameter , Deceleration :=parameter , Jerk :=parameter , BufferMode :=parameter, Done => parameter , Busy => parameter , Active => parameter , CommandAborted => parameter , Error => parameter, ErrorID => parameter);

■ **Input variable**

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
ContinuousUpdate	Continuous update	BOOL	TRUE or FALSE	FALSE	Reserved
Distance	Moving distance	LREAL	Positive number, negative number, 0	0	Specify the movement distance based on the current position as the reference point (Unit: travel unit) *2
Velocity	Target velocity	LREAL	Positive number	Required field	Specify target velocity *1 (Unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Specify acceleration*1 (Unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered 2：mcBlendingLow 3：mcBlendingPrevious 4：mcBlendingNext 5：mcBlendingHigh	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered 2: buffer at low velocity 3: buffer at the previous velocity 4: buffer at the next velocity 5: buffer at low velocity

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions".*

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions".*

*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis".*

■ Output variable

Name	Meaning	Data type	Valid range	Default
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period, and then becomes FALSE
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period, and then becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When an exception is encountered during the execution of the instruction, Error becomes TRUE for one period, and then becomes FALSE

■ Function description

● Basic function description

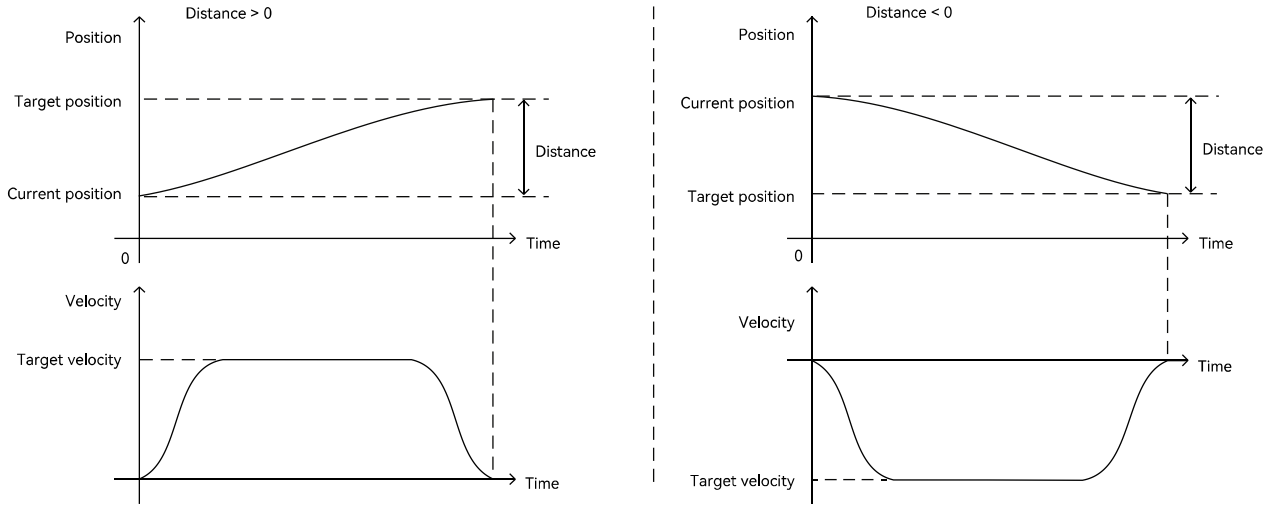
This instruction takes the current position of the axis as a reference and moves the specified distance. Distance, Velocity, Acceleration, Deceleration, and Jerk can be specified in the input variables. This instruction moves according to the values set by the input variables when Execute changes from FALSE to TRUE.

● Distance

- The target position of the axis is determined by the current position of the axis and the value of Distance, that is, the target position=current position+Distance. The value of Distance can be positive, 0, or negative. The values of Velocity, Acceleration, Deceleration, and Jerk can only be positive.
- When the axis is stationary, execute this instruction. The position and velocity curves when the Distance value is greater than 0 and less than 0 are shown below. When the distance value is greater

than 0, the axis runs in the forward direction; When the distance value is less than 0, the axis runs in the opposite direction.

- When the distance value is 0, the instruction does not control the axis operation, and the Done of the instruction becomes TRUE after executing the instruction for one scan period.



- **Instruction completion timing**

When the instruction position of the axis reaches the calculated position, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

When the instruction execution is completed and Execute changes from FALSE to TRUE again, the instruction can be re executed; When an instruction is being executed and Execute changes from FALSE to TRUE again, it does not affect the execution of the instruction, and it still executes the instruction according to the input variable that has not been completed.

- **Execute this instruction while other instructions are in processing**

When other motion instructions are executed, starting this instruction can switch or buffer it. The buffer of this instruction and other motion instructions is determined by the value of the input variable BufferMode, and the values that can be set for the input variable BufferMode of this instruction are related to the executing motion instruction. When other instructions are executed, this instruction is initiated, and the reference point for the input variable Distance is the instruction position of the axis when the instruction Active becomes TRUE.

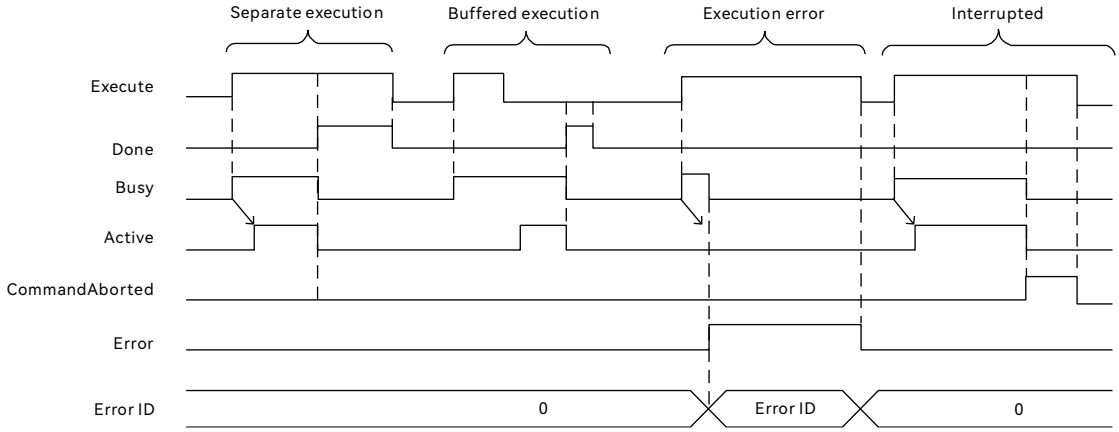
- **Execute other instructions while this instruction is in processing**

When this instruction is executed, other motion instructions are initiated, and the ways they are buffered is determined by the BufferMode pin parameter values of other motion instructions. When other motion instructions and this instruction are buffered, the timing for the execution of other instructions is when Done becomes TRUE. If there is no BufferMode pin for other motion instructions, it is generally aborted.

- **Abnormality elimination**

When the instruction is executed, if the input variable value of the instruction is not within the allowed range or does not meet the execution conditions of the instruction, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed, if an exception is encountered (such as an axis alarm), the command will also report an error and the axis will immediately stop.

■ Output variable timing diagram description



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. When Distance is reached and positioning is completed, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Buffered execution**

  When other instructions control the axis, execute this instruction (BufferMode value is not mcAborting). When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE when the previous instruction is completed.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy (executing) becomes TRUE, Error (in the next period) becomes TRUE, Busy (executing) becomes FALSE, and ErrorID (executing) outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID (in error code). When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy and Active becomes FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

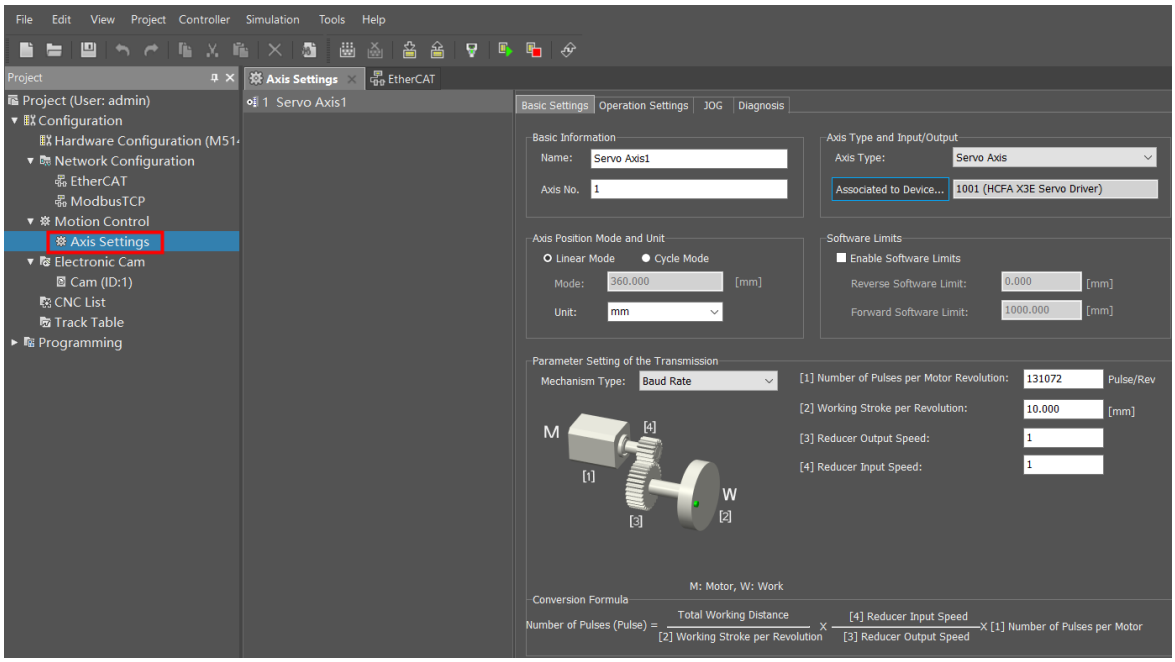
■ Example program

- **Functionality**

  By using the BufferMode function of the relative displacement command (MC_MoveRelative), the control axis moves through two segments of displacement. When the first segment of displacement is completed, the velocity does not need to decrease to 0 and directly moves through the second segment of displacement.

- **Axis parameter setting**

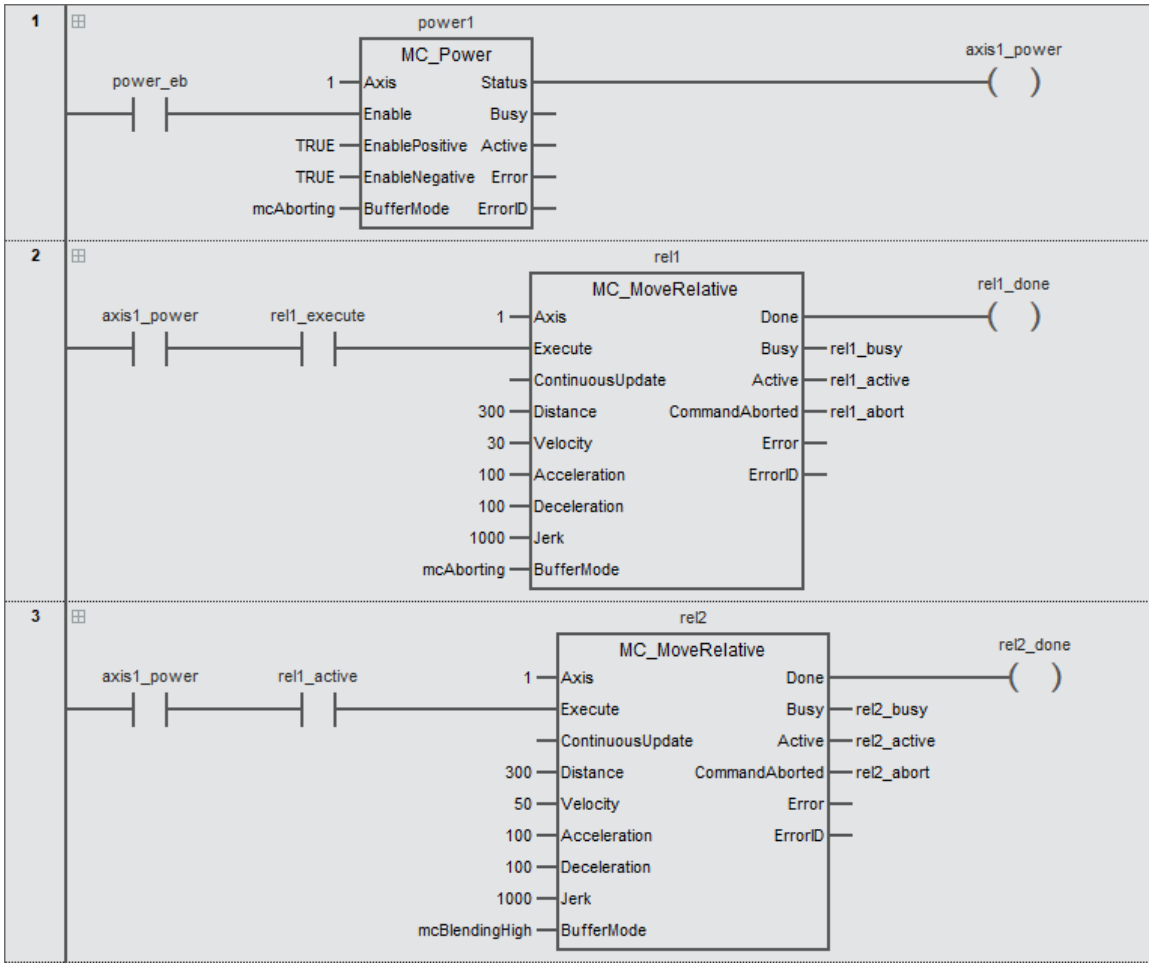
  The parameter settings for axis 1 are shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_execute		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	rel2		MC_MoveRelative		
VAR	rel2_done		BOOL		
VAR	rel2_busy		BOOL		
VAR	rel2_active		BOOL		
VAR	rel2_abort		BOOL		

- LD

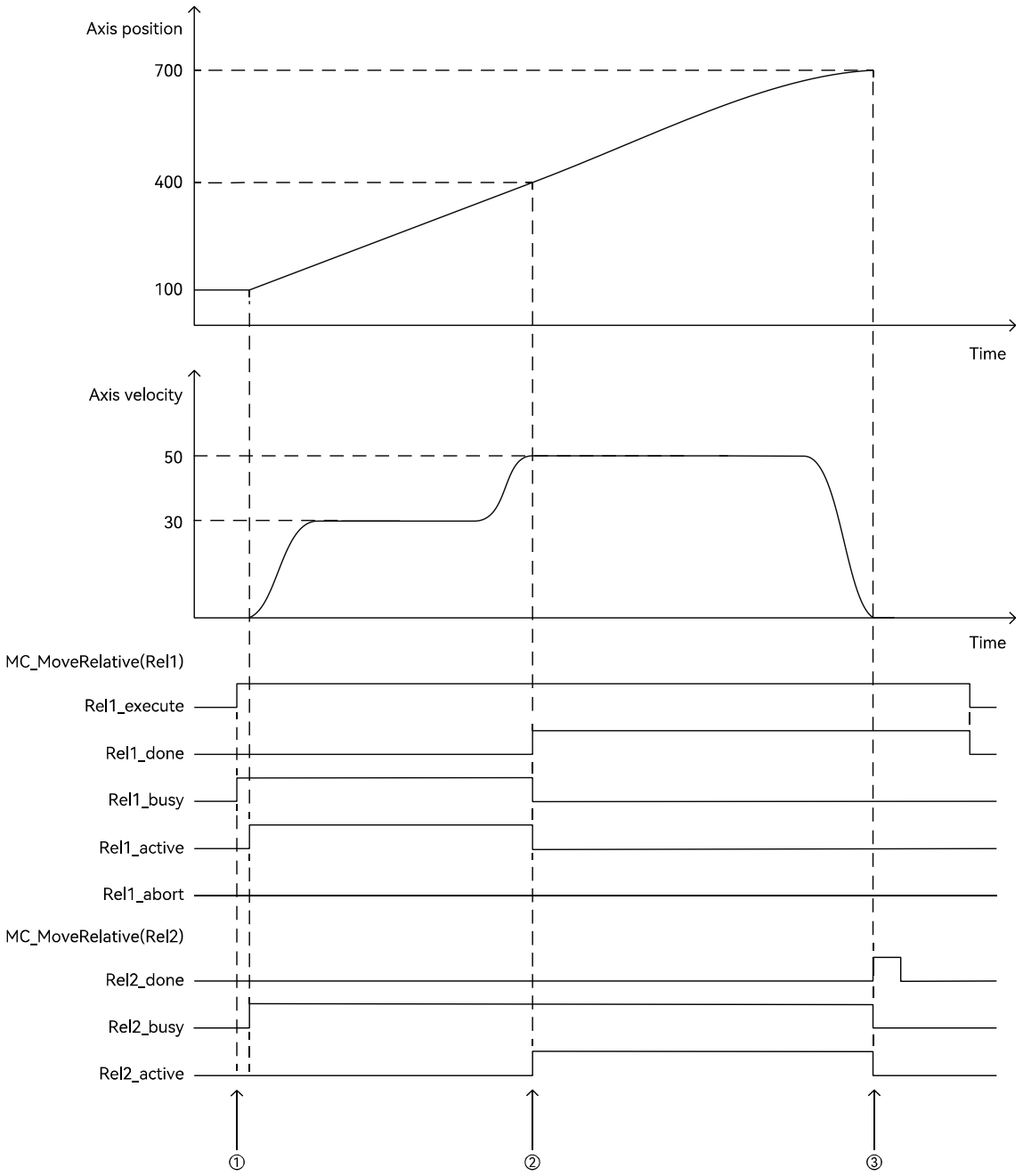


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);


rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_execute ,
    Distance:=300 ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcAborting ,
    Done=>rel1_done ,
    Busy=>rel1_busy ,
    Active=>rel1_active ,
    CommandAborted=>rel1_abort);
```

```
rel2(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_active ,
    Distance:=300 ,
    Velocity:=50 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcBlendingHigh ,
    Done=>rel2_done ,
    Busy=>rel2_busy ,
    Active=>rel2_active ,
    CommandAborted=>rel2_abort );
```

- **Program Description**
  - After enabling the axis, When re1_ execute becomes TRUE, the first relative displacement instruction (rel1) starts executing, the second period in which re1_execute becomes TRUE begins to control the axis, and the Active trigger of this instruction triggers the execution of the second relative displacement instruction (rel2) and does not control the axis until the completion of the first relative displacement instruction (rel1) (Done is TRUE).
  - When the second relative displacement instruction (rel2) controls the axis, the instruction velocity of the axis is 50 (determined by the value of the BufferMode input variable in the rel2 instruction), and the instruction velocity of the axis does not need to be reduced to 0. When the second relative displacement instruction (rel1) is completed (Done is TRUE), because the execution of the instruction is FALSE, Done becomes TRUE after one period and then becomes FALSE.



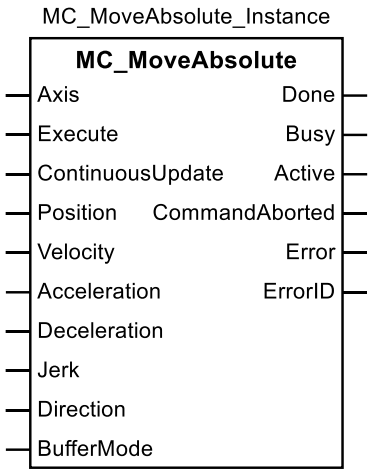
① The first relative instruction (rel1) is executed and the axis is controlled in the 2nd cycle, while the second relative instruction (rel2) is executed but does not control the axis until the Done signal of the first relative instruction is TRUE.

② When the first relative instruction (rel1) is completed (Done is TRUE), the active of the 2nd relative instruction (rel2) becomes TRUE and control of the axis starts.

③ When the 2nd relative instruction (rel2) is completed (Done is TRUE), Active and Busy change to FALSE at the same time, and the Done bit is TRUE for one cycle and then changes to FALSE.

4.13 MC_MoveAbsolute（absolute movement）

Control the specified axis to move to the absolute target position with the 0 point position as the reference point. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveAbsolute	Absolute movement	FB		MC_MoveAbsolute_Instance (Axis :=parameter , Execute :=parameter , ContinuousUpdate :=parameter , Position :=parameter , Velocity :=parameter , Acceleration :=parameter, Deceleration :=parameter, Jerk :=parameter, Direction :=parameter, BufferMode :=parameter , Jerk :=parameter, Done => parameter, Busy => parameter , Active => parameter , CommandAborted => parameter , Error => parameter, ErrorID => parameter);

■ **Input variable**

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the instruction when the rising edge of the parameter is detected
ContinuousUpdate	Continuousupdate	BOOL	TRUE or FALSE	FALSE	Reserved
Position	Absolute position	LREAL	Positive number, negative number, 0	0	Specify the absolute position with the 0 point position as the reference point (Unit: travel unit) *2
Velocity	Target velocity	LREAL	Positive number	Required field	Specify target velocity *1 (Unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Specify acceleration*1 (Unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
Direction	Direction	MC_Direction	1：mcPositiveDirection 2：mcShortestWay 3：mcNegativeDirection 4：mcCurrentDirection	1	Set the direction of axis operation 1: Positive direction 2: The shortest movement distance 3: Negative direction 4: Operate in the current direction and move in the positive direction when the axis is stationary

BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered 2：mcBlendingLow 3：mcBlendingPrevious 4：mcBlendingNext 5：mcBlendingHigh	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered 2: buffer at low velocity 3: buffer at the previous velocity 4: buffer at the next velocity 5: buffer at low velocity

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions".*

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions".*

*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis".*

■ Output variable

Name	Meaning	Data type	Valid range	Default
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period and then becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE

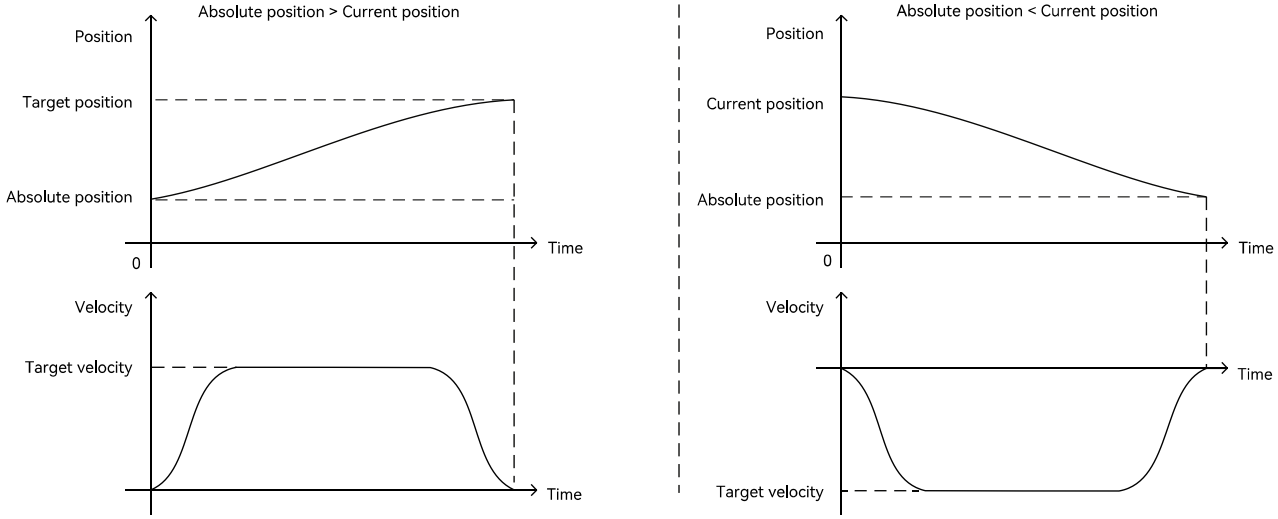
■ Function description

● Basic function description

This instruction is used to control the axis to move to the absolute position with the 0 point position as the reference point. Position, Velocity, Acceleration, Deceleration, and Jerk can be specified in the input variables. This instruction moves according to the value set by the input variable when Execute changes from FALSE to TRUE.

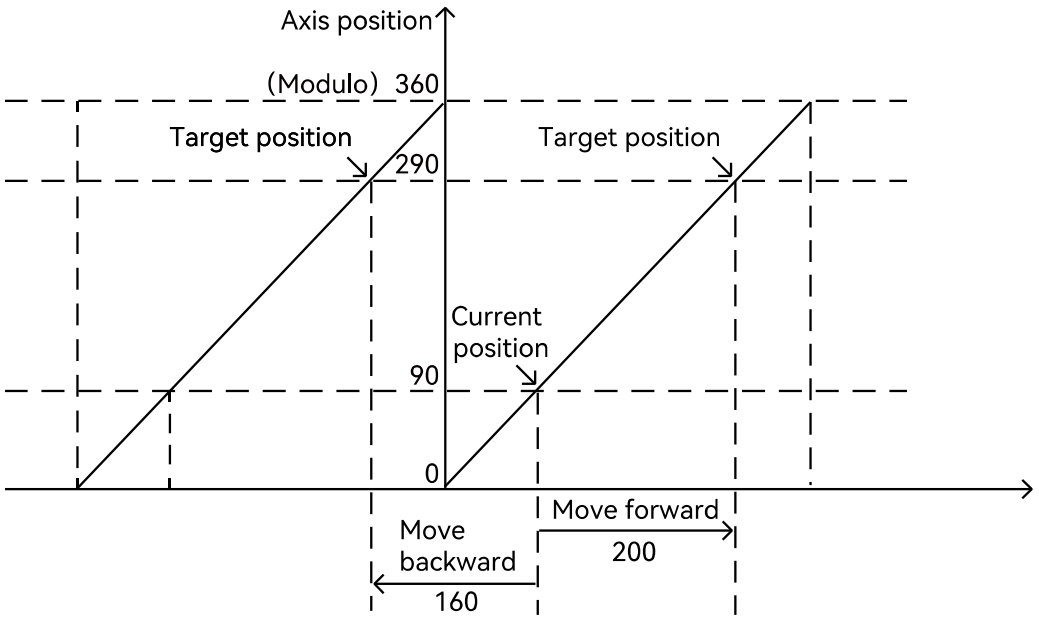
● Absolute position

- The value of Position can be positive, 0, or negative. The values of Velocity, Acceleration, Deceleration, and Jerk can only be positive. When the axis is configured as a rotation axis in the software 【 Axis Configuration】 →【Basic Settings】, the value of Position must be less than the value of the mold.
- When the axis is configured as a linear axis in the software [Axis Configuration] → [Basic Settings], the instruction is executed when the axis is stationary. When the position value is greater than the current position and less than the current position, the position curve and velocity curve are shown below. When the value of Position is greater than the current position, the axis runs in the positive direction; When the value of Position is less than the current position, the axis rotates in the negative direction.
- When the value of Position is equal to the current position, the instruction does not control the axis operation, and the Done of the command becomes TRUE after executing the instruction for one scan period.



● Direction

- The input variable Direction is only effective when the axis is configured in period mode in the software 【Axis Configuration】→【Basic Settings】. This parameter is used to set the direction of axis operation.
- The mold setting for the axis is 360 (configured in [Axis Configuration] → [Basic Settings]). As shown in the figure below, the current position of the axis is 90, the target position is 290, and the Direction selects different values. When the axis is stationary, this instruction is executed, and the corresponding axis's operating direction is shown in the table below. When the value of Direction is 2 (the shortest movement distance), the distance when moving in reverse is 160=360-290+90, and the distance when moving forward is 200=290-90. When moving reversely, the distance is short, indicating that the axis is actually moving in the opposite direction.



Parameter	Direction			
Parameter values	mcPositiveDirection	mcShortestWay	mcNegativeDirection	mcCurrentDirection
Meaning	Positive direction	the shortest movement distance	Reverse direction	Operating in the current direction and moving in the positive direction when the axis is stationary
axis operating direction	Forward movement	Reverse movement	Reverse movement	Forward movement

- When the value of Direction is mcCurrentDirection (moving in the current operating direction and moving in the positive direction when the axis is stationary), execute this instruction, and the axis movement direction will move according to the direction of movement triggered by this instruction when executed. If the axis moves forward before executing this instruction, and when using this instruction to abort other instructions, the axis still moves forward to reach the target position; If the front axis is stationary before executing this instruction, the control axis will run in the forward direction after executing this instruction.

- **Instruction completion timing**

  When the instruction position of the axis reaches the calculated position, the instruction is completed, and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

  When the instruction execution is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed. When an instruction is being executed and Execute changes from FALSE to TRUE again, it does not affect the execution of the instruction, and it still executes the instruction according to the input variable that has not been completed.

- **Execute this instruction while other instructions are in processing**

  When other motion instructions are executed, starting this instruction can switch or buffer it. The buffer of this instruction and other motion instructions is determined by the value of the input variable BufferMode, and the values that can be set for the input variable BufferModee are related to the executing motion instruction. When other instructions are executed, the instruction is initiated, and the reference point for the input variable Position is the instruction position of the axis when the instruction Active becomes TRUE.

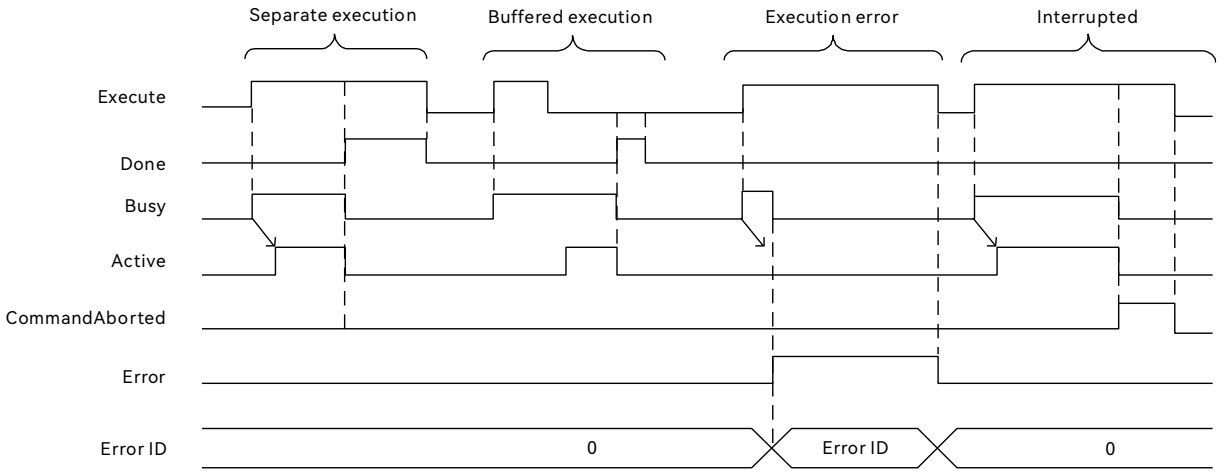
- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, other motion instructions are initiated, and the way they are buffered is determined by the BufferMode pin parameter values of other motion instructions. When other motion instructions and this instruction are buffered, the timing for the execution of other instructions is when the instruction "Done" becomes TRUE. If there is no BufferMode pin for other motion commands, it is generally aborted.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the allowed range or does not meet the execution conditions of the instruction, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed, if an exception is encountered (such as an axis alarm), the instruction will also report an error and the axis will immediately stop.

■ **Output variable timing diagram description**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. When Distance is reached and positioning is completed, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Buffered execution**

  When other instructions control the axis, execute this instruction (BufferMode value is not mcAborting). When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE when the previous instruction is completed.

- **Execution error**

  When the input variable value of this instruction is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy (executing) becomes TRUE, Error (in the next period) becomes TRUE, Busy (executing) becomes FALSE, and ErrorID (executing) outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID (in error code). When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, CommandAborted also becomes FALSE.

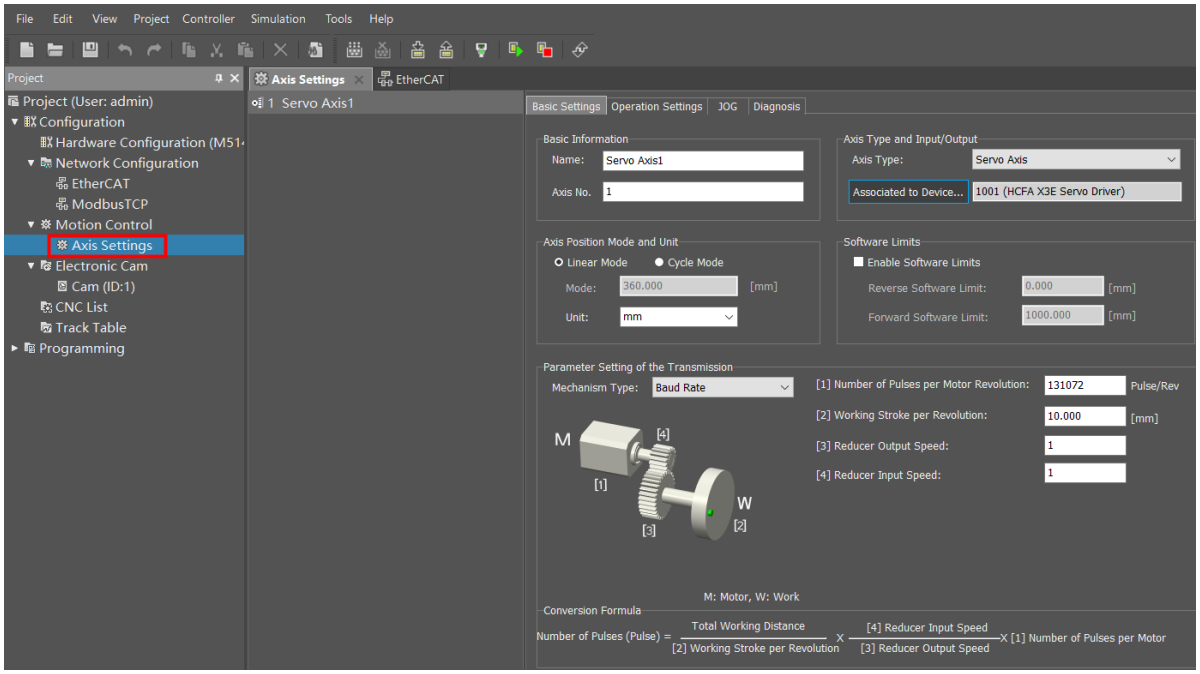
■ Example program

- **Functionality**

  The BufferMode function of the MC_MoveAbsolute displacement instruction is used to control the movement of the axis in two segments. When the first segment of displacement is completed, the velocity does not need to decrease to 0 and directly moves to the second segment.

- **Axis parameter setting**

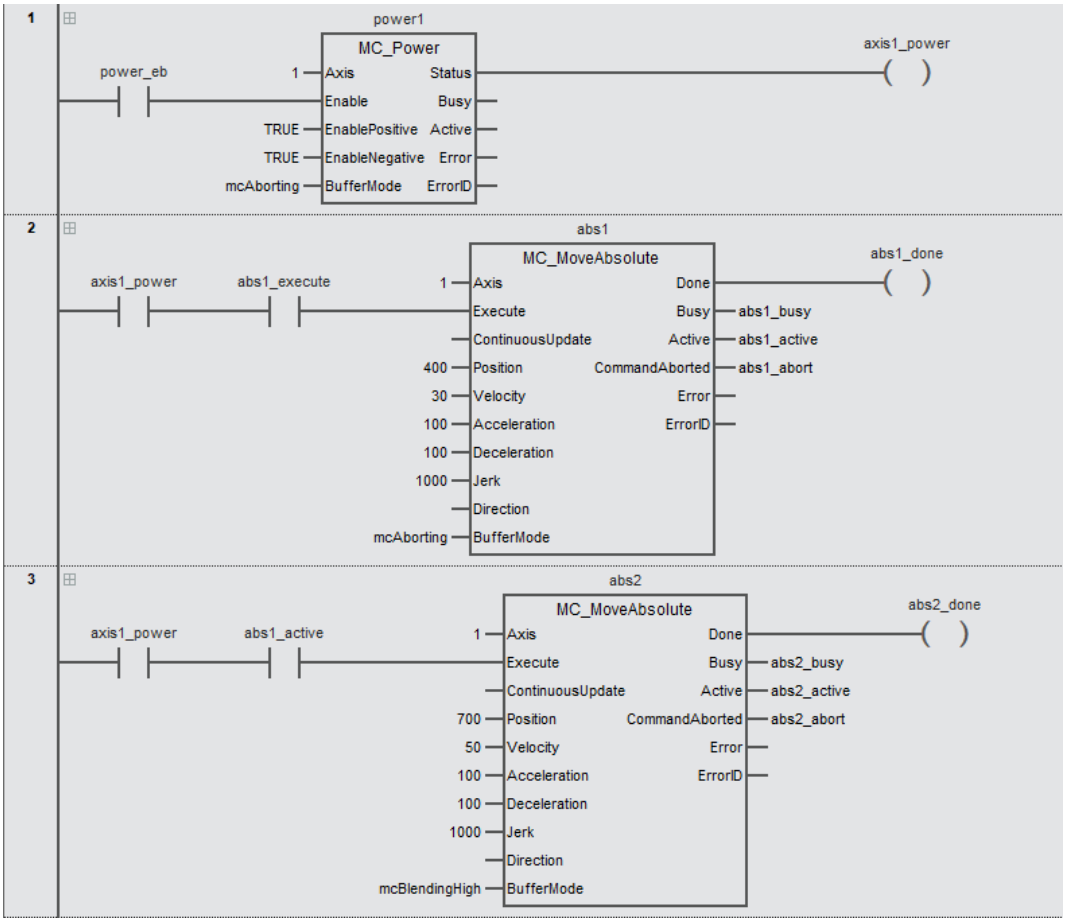
  The parameter setting for axis 1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	abs1_execute		BOOL		
VAR	abs1		MC_MoveAbsolute		
VAR	abs1_done		BOOL		
VAR	abs1_busy		BOOL		
VAR	abs1_active		BOOL		
VAR	abs1_abort		BOOL		
VAR	abs2		MC_MoveAbsolute		
VAR	abs2_done		BOOL		
VAR	abs2_busy		BOOL		
VAR	abs2_active		BOOL		
VAR	abs2_abort		BOOL		

- **LD**

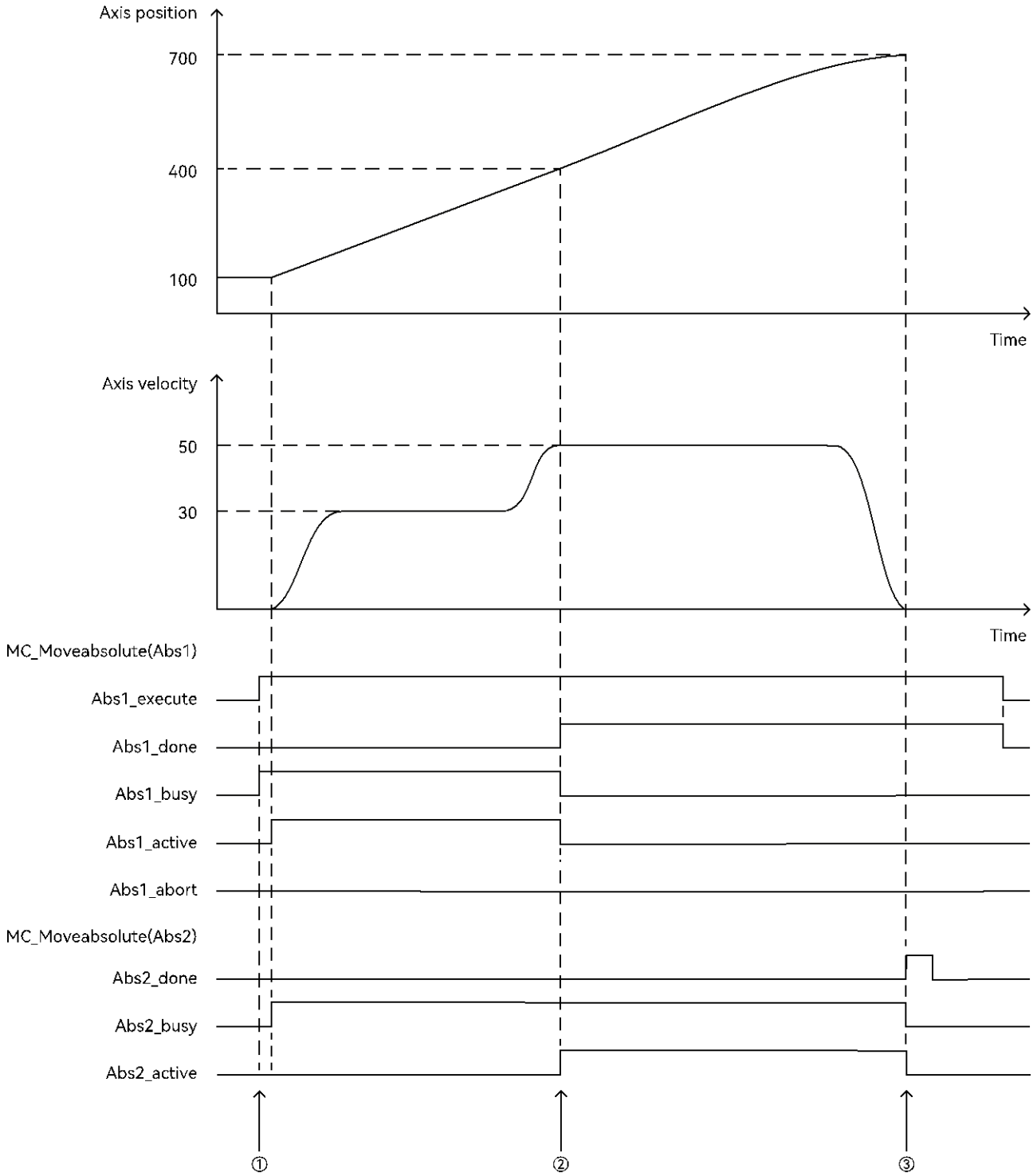


- **ST**

```
power1 (
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
```

```
        BufferMode:=mcAborting ,
        Status=>axis1_power);


    abs1(
        Axis:=1 ,
        Execute:=axis1_power AND abs1_execute ,
        Position:=400 ,
        Velocity:=30 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:=mcAborting ,
        Done=>abs1_done ,
        Busy=>abs1_busy ,
        Active=>abs1_active ,
        CommandAborted=>abs1_abort );


    abs2(
        Axis:=1 ,
        Execute:=axis1_power AND abs1_active ,
        Position:=700 ,
        Velocity:=50 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:=mcBlendingHigh ,
        Done=>abs2_done ,
        Busy=>abs2_busy ,
        Active=>abs2_active ,
        CommandAborted=>abs2_abort );
```

- **Program description**
  - After enabling the axis, the current position of the axis is 100. When abs1_execute becomes TRUE, the first absolute displacement instruction (abs1) begins to execute. The axis is under control when abs1_execute is true for the second period, and the Active of this instruction triggers the execution of the second absolute displacement instruction (abs2) without controlling the axis until the first absolute displacement command (abs1) is completed (Done is TRUE).
  - When the second absolute displacement instruction (abs2) controls the axis, the instruction velocity of the axis is 50 (determined by the value of the BufferMode input variable in the abs2 instruction), and the instruction velocity of the axis does not need to be reduced to 0. When the second absolute displacement instruction (abs2) is completed (Done is TRUE), because the execution of the instruction is FALSE, Done becomes TRUE after one period and then becomes FALSE.



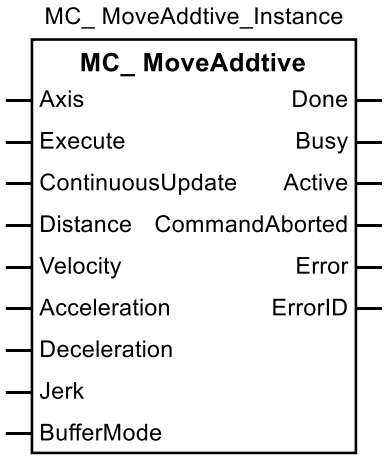
① The 1st absolute instruction (abs1) starts to execute, and starts to control the axis in the 2nd cycle, while the 2nd absolute instruction (abs2) executes but does not control the axis until the 1st absolute instruction Done signal is TRUE.

② When the 1st absolute instruction (abs1) is completed (Done is TRUE), the active of the 2nd absolute instruction (abs2) is changed to TRUE, and start to control the axis.

③ At the completion of the 2nd absolute instruction abs2 (Done is TRUE), Active and Busy change to FALSE at the same time. the Done bit is TRUE for one cycle and then becomes FALSE.

4.14 MC_MoveAdditive（additional movement）

The specified axis to move an additional distance, and the final target position is the sum of the target position of other instructions and the additional distance specified by that instruction. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveAddtive	Additional movement	FB		MC_ MoveAddtive _Instance (Axis :=parameter , Execute :=parameter , ContinuousUpdate , Distance :=parameter , Velocity :=parameter , Acceleration :=parameter , Deceleration :=parameter , Jerk :=parameter, BufferMode :=parameter, Done => parameter , Busy => parameter , Active => parameter , CommandAborted => parameter, Error => parameter, ErrorID => parameter);

■ **Input variable**

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Start	BOOL	TRUE or FALSE	FALSE	Execute the command when the rising edge of the parameter is detected
ContinuousUpdate	Continuous update	BOOL	TRUE or FALSE	FALSE	Reserve
Distance	Moving distance	LREAL	Positive number, negative number, 0	0	Specify the movement distance based on the current position as the reference point (Unit: travel unit) *2
Velocity	Target velocity	LREAL	Positive number	Required field	Specify target velocity *1 (Unit: travel unit/second) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Specify acceleration*1 (Unit: travel unit/second^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Specify deceleration *1 (Unit: travel unit/second^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Specify jerk *1 (Unit: travel unit/second^3) *2
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered 2：mcBlendingLow 3：mcBlendingPrevious 4：mcBlendingNext 5：mcBlendingHigh	0	Set the buffer mode between two instructions*3 0: aborted 1: buffered 2: buffer at low velocity 3: buffer at the previous velocity 4: buffer at the next velocity 5: buffer at low velocity

*1: For the relationship among Velocity, Acceleration, Deceleration, and Jerk, please refer to the *"Parameter description of motion control instructions".*

*2: For a detailed introduction to instruction units, please refer to the *"Parameter unit of motion control instructions".*

*3: For a detailed introduction to BufferMode, please refer to the *"Buffer mode during multi-starting of the same axis".*

■ Output variable

Name	Meaning	Data type	Valid range	Default
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one period, it becomes FALSE
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When the instruction is aborted by another instruction, CommandAborted becomes TRUE, and after one period, it becomes FALSE
Error	The input variable value of the instruction is not within the allowed range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ The instruction has been executed and Execute has become FALSE. When an exception is encountered during the execution of the instruction, Error becomes TRUE. After one period, it becomes FALSE

■ Function description

● Basic function description

➢ The specified axis to move an additional distance, and the final target position is the sum of the target position of other instructions and the additional distance specified by that instruction. Distance, Velocity, Acceleration, Deceleration, and Jerk can be specified in the input variables. This instruction moves according to the values set by the input variables when Execute changes from FALSE to TRUE. The value of Distance can be positive, 0, or negative. The values of Velocity, Acceleration, Deceleration, and Jerk can only be positive.

➢ When there is no other instruction to control the operation of the axis, execute this instruction separately and the relative displacement instruction function of MC_ MoveRelative is the same, used to control the axis to move the set distance according to the set velocity, acceleration and deceleration, and jump. The reference starting point of this distance is the axis position at the beginning of the instruction execution.

- When the instruction aborts the displacement-related instruction, after the instruction is completed, the distance the terminal execution mechanism moves is the sum of the remaining distance of other instructions and the additional distance set by the instruction. When this instruction aborts non displacement related instructions (such as velocity commands, etc.), it will abort the execution of other instructions and stop after moving at the set velocity, accelerating or decelerating for a set distance. The reference starting point for the additional distance is the current position of the axis when the instruction starts to be executed.
- When MC_ MoveSuperior is Separate execution, the instruction is executed. If the value of the input variable BufferMode of the instruction is mcAborting, the instruction will abort the executing MC_ MoveSuperior instruction, the final target position is current position of the axis before the execution of the MC_ MoveSuperior instruction, the additional distance specified by the distance specified by the command, and MC_ MoveSuperposed is the sum of distances specified by Distance; If the value of the input variable BufferMode in this instruction is not mcAborting, executing this instruction will result in an error.
- MC_ MoveSuperior is executed simultaneously with other displacement instructions, the instruction is executed. If the value of the input variable BufferMode of the instruction is mcAborting, the instruction will abort all currently executing displacement instructions. The final target position is the additional distance specified by the instruction through Distance, MC_ MoveSuperior is the sum of the distance specified by Distance and the target position of other displacement instructions. If the value of the input variable BufferMode in this instruction is not mcAborting, then the instruction is based on the value of BufferMode and MC_ MoveSuperior buffers other displacement instructions executed simultaneously, and executes the instruction after the target position of the other instructions is reached.

- **Instruction completion timing**

  When the instruction position of the axis reaches the calculated position, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

  When the instruction execution is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; When an instruction is being executed and Execute changes from FALSE to TRUE again, it does not affect the execution of the instruction and still executes the instruction according to the input variable that has not been completed.

- **Execute this instruction while other instructions are in processing**

  When other motion instructions are executed, starting this instruction can switch or buffer it. The buffer of this instruction and other motion instructions is determined by the value of the input variable BufferMode, and the values that can be set for the input variable BufferMode of this instruction are related to the executing motion instruction. When other instructions are executed, this instruction is initiated, and the reference point for the input variable Distance is the instruction position of the axis when the instruction Active becomes TRUE.

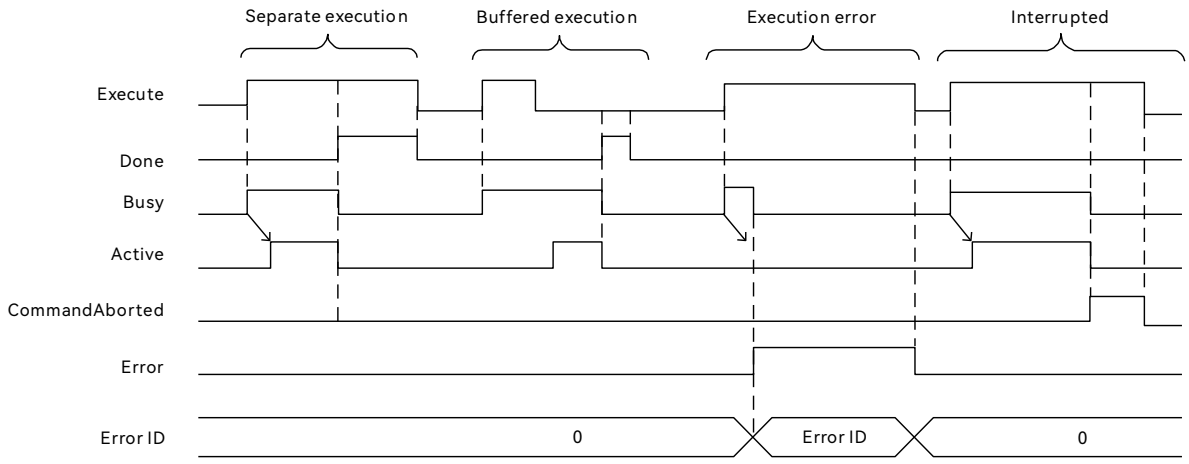
- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, other motion instructions are initiated, and the way they are buffered is determined by the BufferMode pin parameter values of other motion instructions. When other motion instructions and this instruction are buffered, the timing for the execution of other instructions is when the instruction Done becomes TRUE. If there is no BufferMode pin for other motion instructions, it is generally aborted.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the allowed range or does not meet the execution conditions of the instruction, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. When this instruction is executed, if an exception is encountered (such as an axis alarm), the instruction will also report an error and the axis will immediately stop.

Output variable timing diagram description



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE in the next period. When Distance is reached and positioning is completed, Done becomes TRUE, Busy and Active become FALSE simultaneously. When Execute becomes FALSE, Done also becomes FALSE.

- **Buffered execution**

  When other instructions control the axis, execute this instruction (BufferMode value is not mcAborting). When Execute changes from FALSE to TRUE, Busy also becomes TRUE, and Active becomes TRUE when the previous instruction is completed.

- **Execution error**

  When the input variable value is not within the allowable range, when the Execute instruction changes from FALSE to TRUE, Busy becomes TRUE, Error in the next period becomes TRUE, Busy becomes FALSE, and ErrorID outputs the corresponding error code. The cause of the problem can be found by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, while Error becomes FALSE, the value of ErrorID becomes 0.

- **Interrupted**

  When the instruction is aborted by another instruction after execution, CommandAborted becomes TRUE, Busy and Active become FALSE simultaneously; When Execute becomes FALSE, CommandAborted also becomes FALSE.

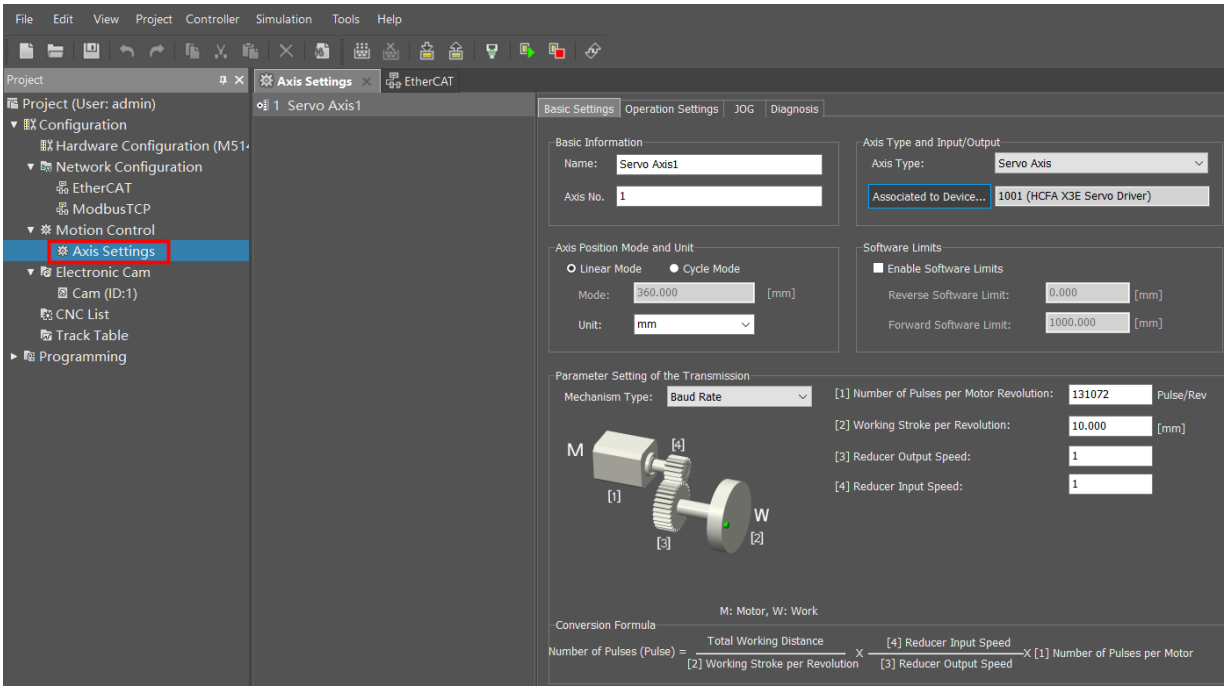
Example program

- **Functionality**

  When the relative displacement instruction (MC_MoveRelative) controls the operation of the axis, it is aborted by the additional displacement instruction (MC_MoveAdditive). After the abortion, the axis takes the velocity set by the additional displacement instruction (MC_MoveAdditive) as the target velocity. After the instruction is completed, the distance the axis moves is the sum of the distance set by the relative displacement instruction and the additional displacement instruction.

- **Axis parameter setting**

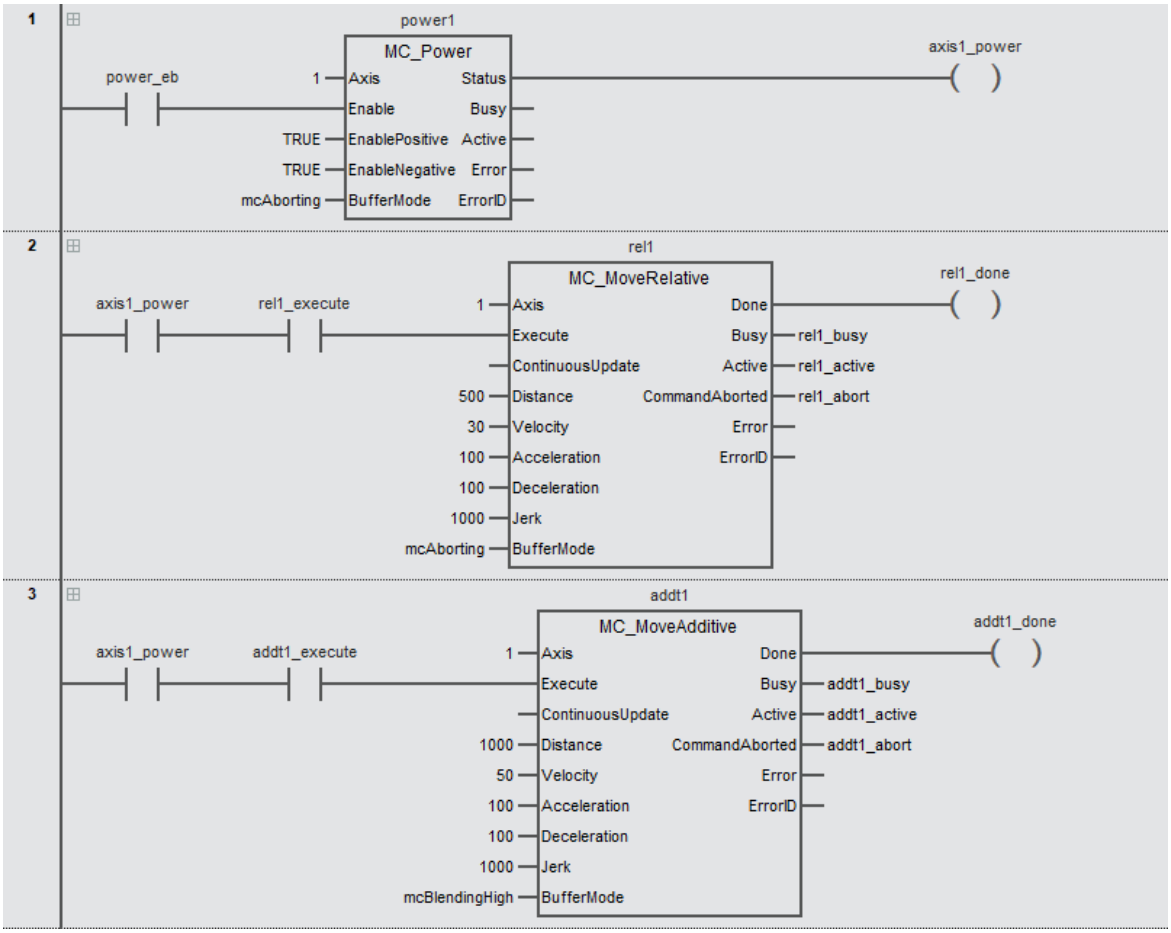
  The parameter setting for axis 1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_execute		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	addt1_execute		BOOL		
VAR	addt1		MC_MoveAdditive		
VAR	addt1_done		BOOL		
VAR	addt1_busy		BOOL		
VAR	addt1_active		BOOL		
VAR	addt1_abort		BOOL		

- LD

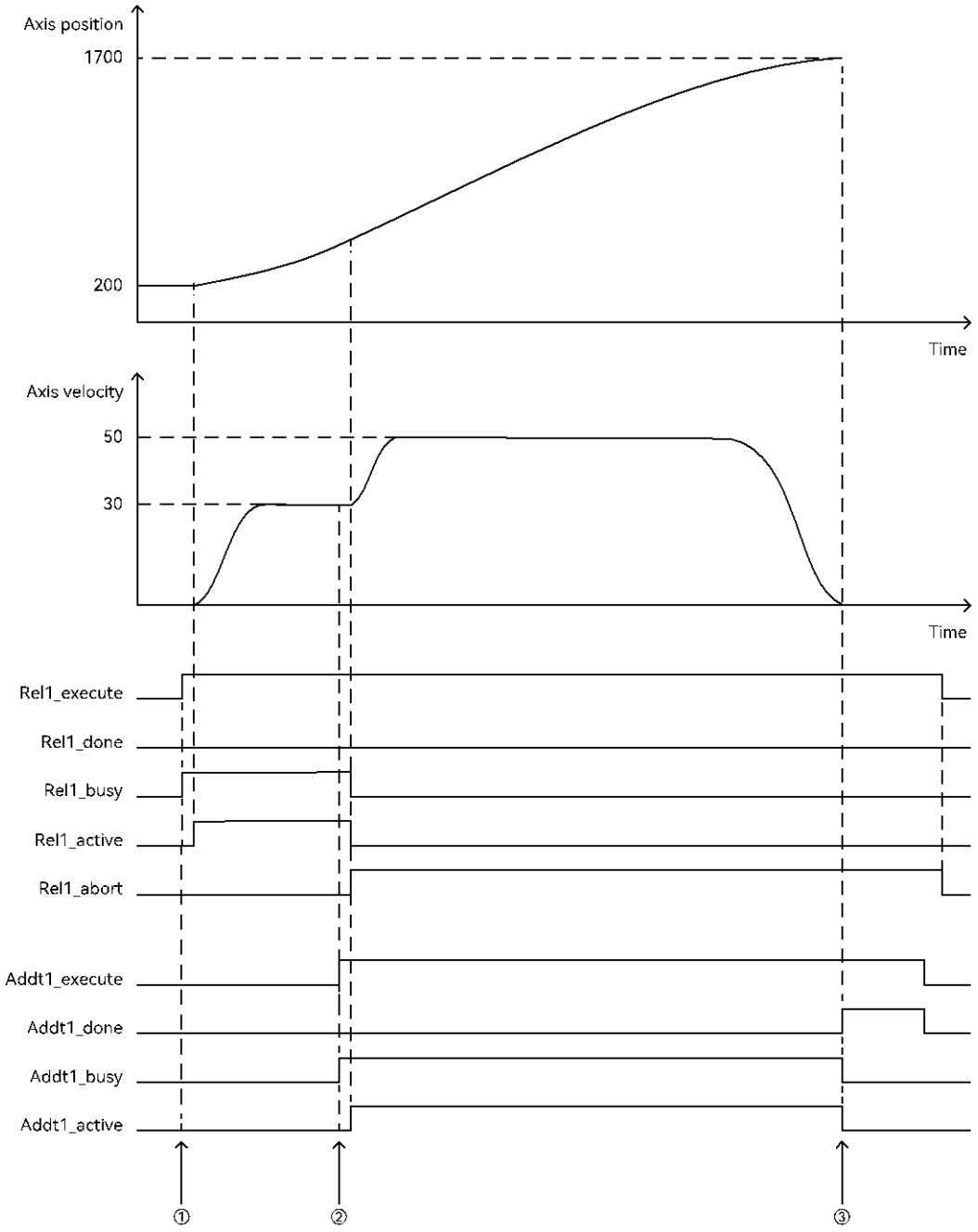


- ST

```
power1 (
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);

rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_execute ,
    Distance:=500 ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=100 ,
    BufferMode:=mcAborting ,
    Done=>rel1_done ,
    Busy=>rel1_busy ,
    Active=>rel1_active ,
    CommandAborted=>rel1_abort);

addt1 (
    Axis:=1 ,
```

```
Execute:= axis1_power AND addt1_execute ,
Distance:=1000 ,
Velocity:=50 ,
Acceleration:=100 ,
Deceleration:=100 ,
Jerk:=1000 ,
BufferMode:= mcBlendingHigh ,
Done=> addt1_done ,
Busy=> addt1_busy ,
Active=>addt1_active ,
CommandAborted=> addt1_abort);
```

- **Program description**
  - After enabling the axis, re1_ When execute becomes TRUE, the relative displacement instruction (rel1) begins to execute. When the relative displacement instruction (rel1) controls the operation of the axis, addt1_ execute becomes TRUE, and the additional displacement command is executed (addt1), which aborts the execution of the relative displacement instruction after controlling the axis.
  - After the execution of the relative displacement instruction is aborted by the additional displacement instruction, the axis takes the velocity set by the additional displacement instruction (addt1) as the target velocity. As shown in the figure below, when the additional displacement instruction (addt1) is not executed, the velocity of the axis is 30. After execution, the target velocity of the axis becomes 50. After the execution of the additional displacement instruction is completed, the distance of axis movement is the sum of the distance set by the relative displacement instruction and the additional displacement instruction. As shown in the figure below, the initial position of the axis is 200. After the completion of the additional displacement instruction, the position of the axis is 1700, and the movement distance is 1700-200=1500. The sum of the distances set by the relative displacement instruction and the additional displacement instruction is 500, and the movement distance set by the relative displacement instruction is 1000.



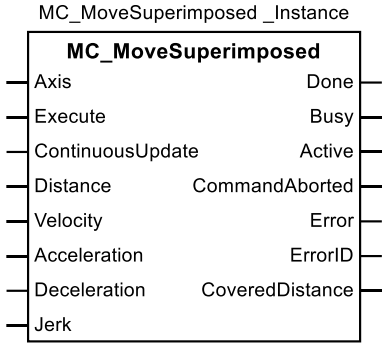
① The relative instruction (rel1) starts execution, and the axis is controlled in the 2nd cycle.

② The additional instruction (addt1) is executed, and the 2nd cycle starts to control the axis and interrupts the execution of the relative instruction.

③ When the additive instruction (addt1) is completed (Done is TRUE), and at the same time Active and Busy become FALSE. the distance of two instructions is the sum of the distance set by the two instructions.

4.15 MC_MoveSuperimposed（superimpose relative displacement）

This instruction superimposes the distance, velocity, acceleration generated for the specified axis with the original instruction. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveSuperImposed	Superimpose relative displacement	FB		MC_MoveSuperimposed _Instance (Axis :=parameter , Execute :=parameter , ContinuousUpdate :=parameter, Distance :=parameter, Velocity :=parameter, Acceleration :=parameter, Deceleration :=parameter, Jerk :=parameter, Done => parameter , Busy => parameter, Active => parameter , CommandAborted => parameter , Error => parameter, ErrorID => parameter , CoveredDistance=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
ContinuousUpdate	Continuous Update	BOOL	TRUE or FALSE	FALSE	Reserved
Distance	Additional distance	LREAL	Positive number, Negative number, 0	0	Specify the travel distance from the current position (Unit: Travel unit)*2
Velocity	Target velocity	LREAL	Positive number	Required field	Specify the target velocity *1 (Unit: Travel unit/s)*2
Acceleration	Acceleration rate	LREAL	Positive number	Required field	Specify the acceleration rate *1 (Unit: Travel unit/s^2)*2
Deceleration	Deceleration rate	LREAL	Positive number	Required field	Specify the deceleration rate *1 (Unit: Travel units/s^2)*2
Jerk	Jerk	LREAL	Positive number	Required field	Specify the jerk*1 (Unit: Travel units/s^3)*2

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions ".*

*2: For details of the instruction units, please refer to *"Parameter unit of motion control instructions"*.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error

ErrorID	Error code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.
CoveredDistance	Covered distance	LREA	Positive number，Negative number，0	The distance that the command control axis has moved after this instruction is executed.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done becomes TRUE and then FALSE one period later
Busy	When Execute becomes TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute becomes FALSE and the instruction is aborted by another instruction, CommandAborted becomes TRUE and then FALSE one period later
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction.	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute becomes FALSE and an exception is encountered during the execution of the instruction, Error becomes TRUE and then FALSE one period later

■ Function description

● Basic function description

- ➢ This instruction superimposes the distance, velocity, and acceleration generated for the specified axis with the original instruction, which is often used for error correction when the actual value does not match the theoretical value. This instruction can be operated on the slave or master axis in a multi-axis relationship and is normally executed on the slave axis.
- ➢ This instruction takes the current position of the axis as the reference point and moves the distance specified by Distance. Distance, Velocity, Acceleration, Deceleration, and Jerk can be specified as input variables. The axis moves according to the input variables when Execute changes from FALSE to TRUE.
- ➢ In the standstill state of the axis, MC_MoveSuperImposed behaves like MC_MoveRelative.
- ➢ This instruction does not interrupt the executing instruction but is executed together with other instructions.

● Re-execute the instruction

When the execution of the instruction is complete and Execute changes from FALSE to TRUE again, When this instruction is being executed and Execute changes from FALSE to TRUE again, the execution of instructions will not be affected, the instruction continues to be executed in a previous way.

● Execute this instruction while other instructions are in processing

- ➢ When MC_MoveSuperimposed instruction is executed to control the same axis while other instructions are being executed. This instruction does not interrupt the instruction being executed, but

is executed in parallel with the other instructions. The distance and velocity generated by this instruction are superimposed on those of other instructions.

- If another MC_MoveSuperimposed instruction (controlling the same axis) is executed when other instructions and MC_MoveSuperimposed are executed (controlling the same axis), the MC_MoveSuperimposed instruction executed later will interrupt the MC_MoveSuperimposed instruction executed earlier and has no effect on the other instructions originally executed.

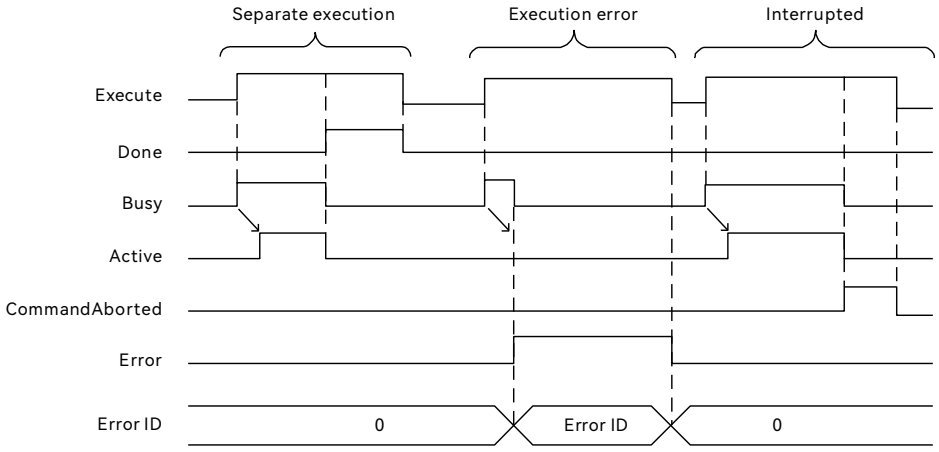
- **Execute other instructions while this instruction is in processing**

  - When MC_MoveSuperimposed instruction is executed separately, this instruction will be aborted if the MC_HaltSuperimposed instruction (controlling the same axis) is executed.
  - When the MC_MoveSuperimposed instruction is executed alone, if another MC_MoveSuperimposed instruction (controlling the same axis) is executed, the MC_MoveSuperimposed instruction executed later will interrupt the instruction executed earlier.
  - When other instructions and the MC_MoveSuperimposed instruction are both executed (controlling the same axis) if the MC_HaltSuperimposed instruction is executed (controlling the same axis), the MC_MoveSuperimposed instruction will be aborted, and there is no effect on the other instructions originally executed.
  - When other instructions and the MC_MoveSuperimposed instruction are both executed (controlling the same axis), if the MC_MoveAdditive instruction is executed (controlling the same axis), refer to the function description of the MC_MoveAdditive instruction for details.
  - When both other instructions and MC_MoveSuperimposed instruction are executed (controlling the same axis), if other motion instructions are executed (controlling the same axis, excluding MC_MoveSuperimposed, MC_HaltSuperimposed, and MC_MoveAdditive instructions), and the later executed motion instruction without BufferMode parameter, other instructions and MC_MoveSuperimposed instruction will be aborted; if the BufferMode parameter of the later executed motion instruction selects mcAborting (interrupt), other instructions and MC_MoveSuperimposed instruction will be aborted; if the BufferMode parameter of the later executed motion instruction selects not mcAborting (interrupt), MC_MoveSuperimposed instruction will be aborted and other instructions will still be executed normally.

- **Abnormality elimination**

When this instruction is executed, if the input variable is illegal, the Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g., an axis error), the instruction will also report an error and the axis stop immediately.

■ **Output variable timing diagram description**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When Distance is reached and positioning is complete, Done becomes TRUE, and Busy and Active change to FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy becomes TRUE. In the next period, Error becomes TRUE while Busy becomes FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error becomes FALSE and the value of ErrorID becomes zero.

- **Interrupted**

  When the execution of this instruction is aborted by another instruction, CommandAborted becomes TRUE, Busy, and Active change to FALSE at the same time. When Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

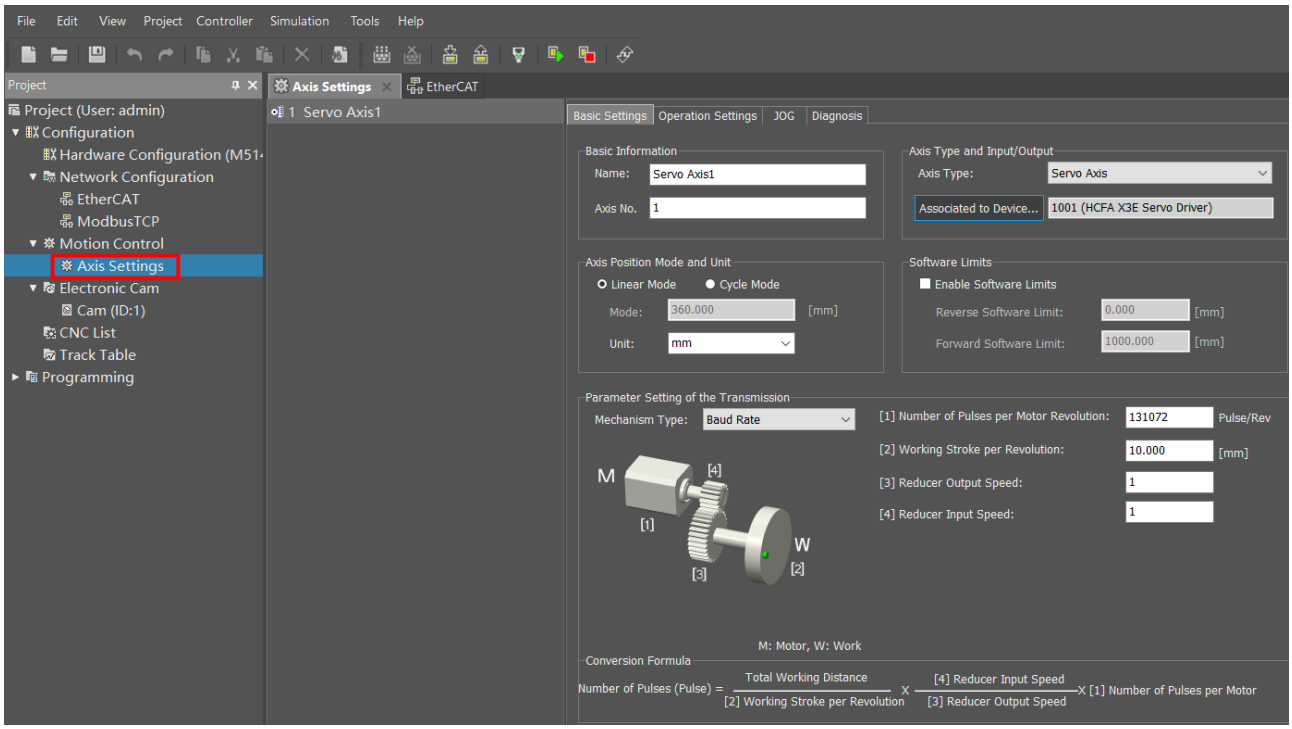
Example program

- **Functionality**

  When MC_MoveRelative and MC_MoveSuperimposed control the same axis together, the velocity, acceleration, and distance of the two instructions can be superimposed and applied to the same axis. This instruction is generally used to correct the position when the actual value does not match the theoretical value.

- **Axis parameter setting**

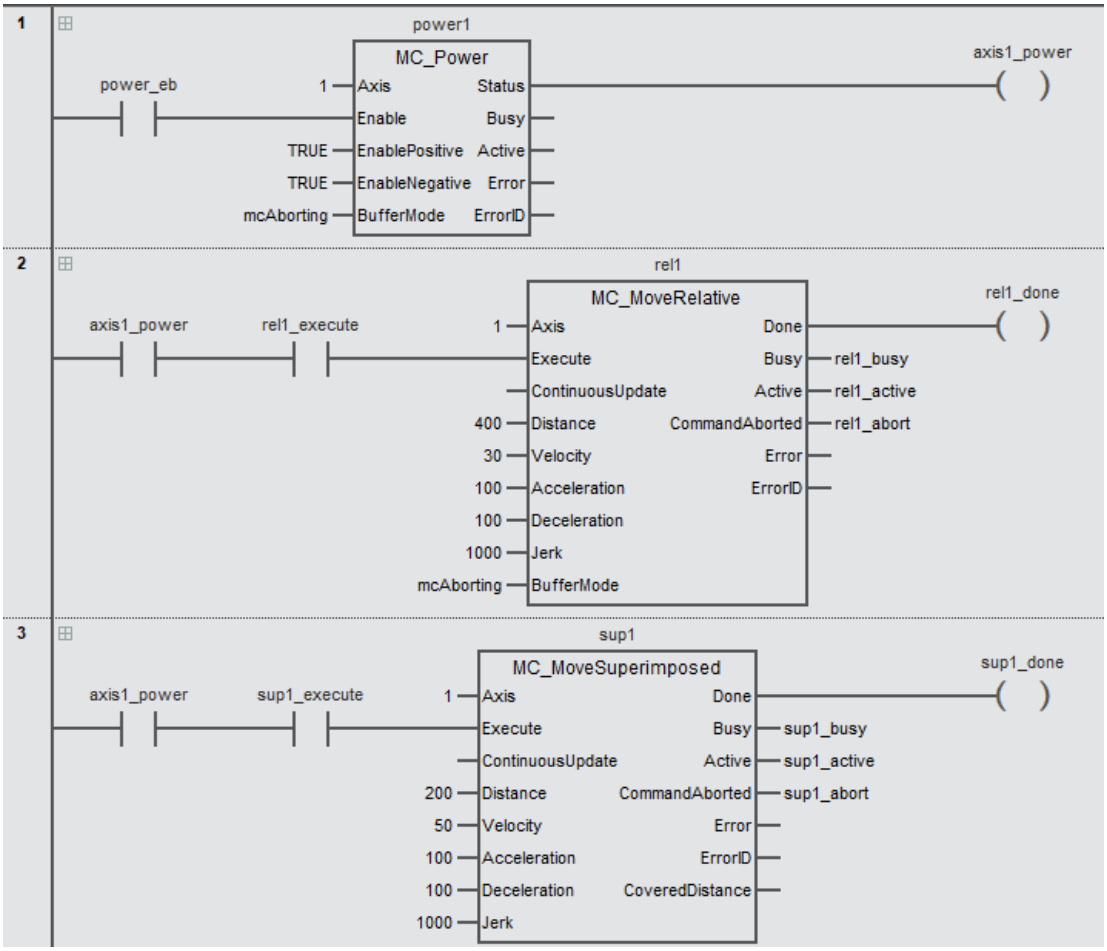
  The axis parameter settings for axis1 are shown below.



- Variable table

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_execute		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	sup1_execute		BOOL		
VAR	sup1		MC_MoveSuperimposed		
VAR	sup1_done		BOOL		
VAR	sup1_busy		BOOL		
VAR	sup1_active		BOOL		
VAR	sup1_abort		BOOL		

- LD

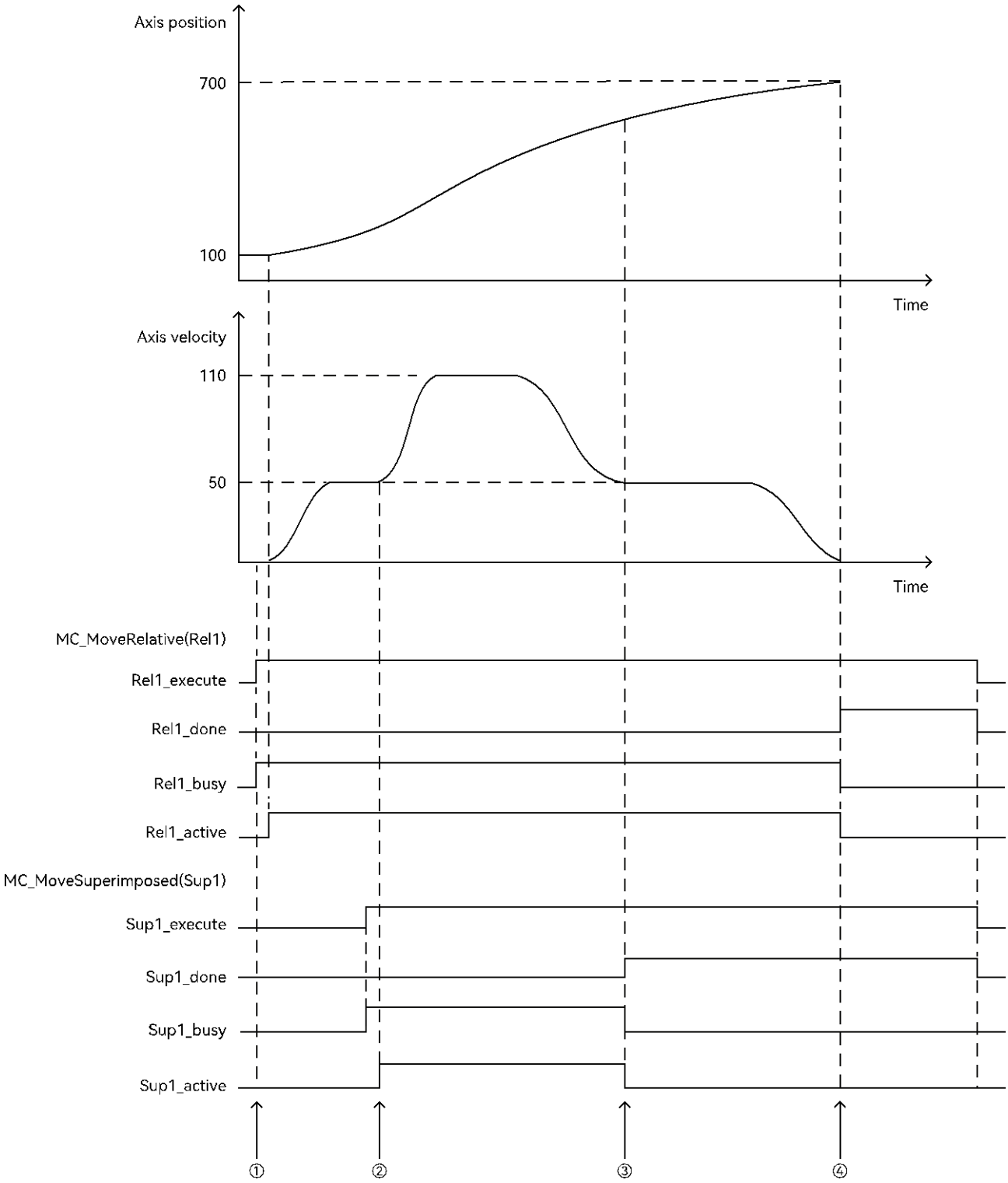


- ST

```
power1 (
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);


rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_execute ,
    Distance:=400 ,
    Velocity:=50 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=100 ,
    BufferMode:=mcAborting ,
    Done=>rel1_done ,
    Busy=>rel1_busy ,
    Active=>rel1_active ,
    CommandAborted=>rel1_abort );
```

```
sup1 (
    Axis:=1 ,
    Execute:= axis1_power AND sup1_execute ,
    Distance:=200 ,
    Velocity:=60 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Done=> sup1_done ,
    Busy=> sup1_busy ,
    Active=>sup1_active ,
    CommandAborted=> sup1_abort );
```

- **Program description**
  - After the axis is enabled, rel1 starts to execute once re1_execute becomes TRUE. When rel1 controls axis operation and sup1_execute becomes TRUE, sup1 is executed. After sup1 controls the axis, the velocity, acceleration, and distance generated by the two instructions are superimposed and then acted on one axis.
  - As shown in the figure below, when sup1 is not executed, the velocity of the axis is 50 (target velocity of rel1). After sup1 is executed and the set velocity is reached, the velocity of the axis is 110, which is the superposition of the target velocity of rel1 (50) and the target velocity of sup1(60). When sup1 is completed (the set target distance is traveled), rel1 is not completed, rel1 continues to control the axis, and the axis velocity becomes the target velocity set by rel1.
  - After rel1 is completed, the moving distance of the axis is the sum of the distances set by rel1 and sup1. As shown in the following figure, the initial position of the axis is 100, and after rel1 is completed, the position of the axis is 700, and the moving distance is 700-100=600, which is the sum of the distances set by rel1 and sup1, and the moving distance set by rel1 is 400, and the moving distance set by sup1 is 200.



① The relative instruction (rel1) starts execution, and the axis is controlled in the 2nd cycle.

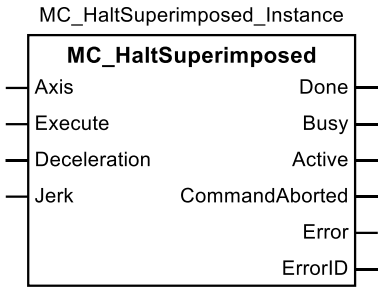
② The superimposed instruction (sup1) starts controlling the axis, and the velocity and acceleration of the relative instruction and the velocity and acceleration of the superimposed instruction are summed.

③ When the superimposed instruction (sup1) is completed (Done is TRUE), the relative instruction (rel1) has not been completed and is still controlling the axis.

④ When the relative instruction (rel1) is completed (Done is TRUE) at the same time Active and Busy become FALSE.

4.16 MC_HaltSuperimposed (halt superimposing displacement)

This instruction controls the specified axis to halt the superimposed displacement in accordance with the set deceleration, jerk. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_HaltSuperImposed	Halt superimposing displacement	FB		MC_HaltSuperImposed _Instance (Axis:=parameter , Execute :=parameter , Deceleration:=parameter , Jerk :=parameter, Done=> parameter , Busy=> parameter , Active=> parameter , CommandAborted => parameter, Error=> parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
Deceleration	Deceleration rate	LREAL	Positive number	Required field	Specify the deceleration rate *1 (Unit: Travel unit/s²)*2
Jerk	Jerk	LREAL	Positive number	Required field	Specify the jerk*1 (Unit: Travel unit/s³)*2

*1: For the relation among Deceleration and Jerk, please refer to "Parameter description of motion control instructions ".

*2: For details of the instruction units, please refer to "Parameter unit of motion control instructions".

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is being controlled
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when an error occurs
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the positioning is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE. ◆ When the instruction is executed and Execute is FALSE, Done becomes TRUE for one period and then to FALSE.
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE

Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute becomes FALSE and the instruction is aborted by another instruction, CommandAborted becomes TRUE and then FALSE one period later
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction.	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute becomes FALSE and an exception is encountered during the execution of the instruction, Error becomes TRUE and then FALSE one period later

■ Function description

● Basic function description

- ➢ This instruction controls the specified axis to halt the superimposed displacement in accordance with the set deceleration and jerk.
- ➢ If the MC_HaltSuperimposed instruction is executed while the MC_MoveSuperimposed instruction is executing, the MC_HaltSuperimposed instruction will interrupt the MC_MoveSuperimposed instruction; if the MC_MoveSuperimposed instruction is executed while the MC_MoveSuperimposed instruction is not executing, the MC_HaltSuperimposed instruction will report an error.
- ➢ If MC_MoveSuperimposed and other motion instructions are executed at the same time and control the same axis, the MC_HaltSuperimposed instruction is executed, which will interrupt the MC_MoveSuperimposed instruction and other motion instructions are still executed normally.

● Re-execute the instruction

The MC_HaltSuperimposed instruction cannot be executed alone; the MC_HaltSuperimposed instruction can be executed only when the MC_MoveSuperimposed instruction is executed. When the MC_HaltSuperimposed instruction is being executed and Execute becomes TRUE from FALSE again, the execution of this instruction will not be effected, and MC_HaltSuperimposed instruction will continue to be executed as before.

● Execute this instruction while other instructions are in processing

The MC_HaltSuperimposed instruction can be executed only when the MC_MoveSuperimposed instruction is executed. When the MC_HaltSuperimposed instruction is executed while other instructions are executing, this instruction will report an error.

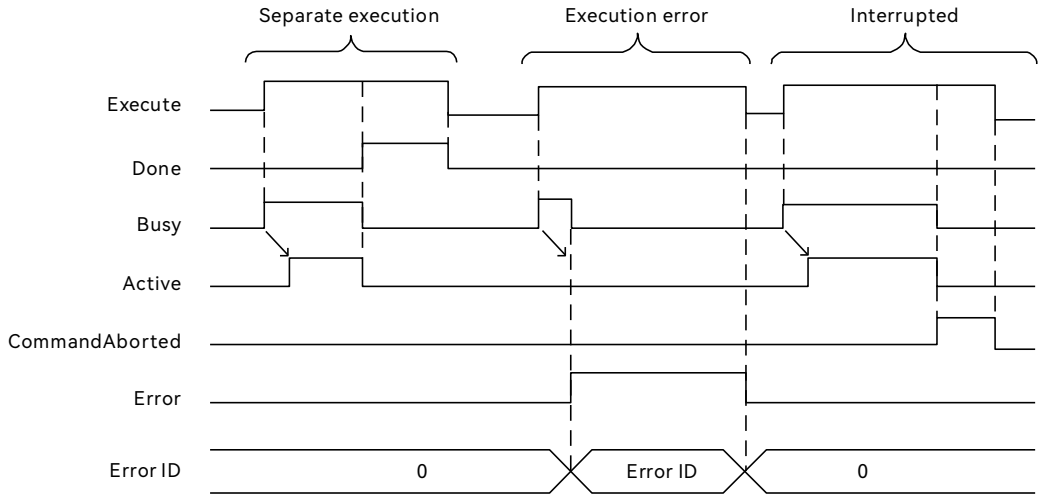
● Execute other instructions while this instruction is in processing

During the execution of the instruction, other instructions are executed, and how the other instructions and the instruction are transitioned is determined by the value of the BufferMode pin parameter of the other instructions, if the other instructions do not have BufferMode pins, it usually aborts the instruction.

● Abnormality elimination

When this instruction is executed, if the input variable is illegal, the Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g., an axis error), the instruction will also report an error and the axis will stop immediately.

- **Output variable timing diagram description**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When this instruction completes, Done becomes TRUE, and Busy and Active change to FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy becomes TRUE. In the next period, Error becomes TRUE while Busy becomes FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error becomes FALSE and the value of ErrorID becomes zero.

- **Interrupted**

  When the execution of this instruction is aborted by another instruction, CommandAborted becomes TRUE, Busy, and Active change to FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

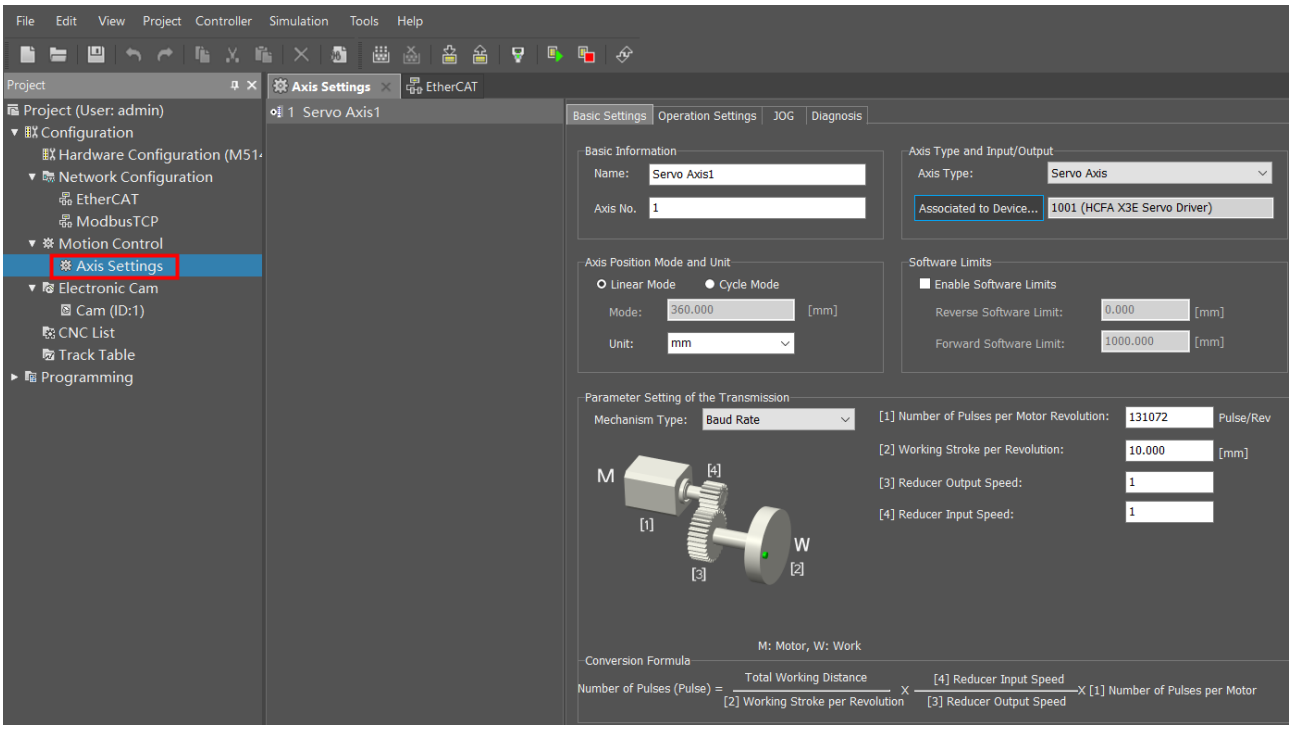
■ Example program

- **Functionality**

  When the actual value does not match the theoretical value the position can be corrected by the MC_MoveSuperImposed instruction. If the corrected position is found to be unreasonable or incorrect after starting the MC_MoveSuperImposed instruction, the position correction can be aborted by the MC_HaltSuperImposed instruction.

- **Axis parameter setting**

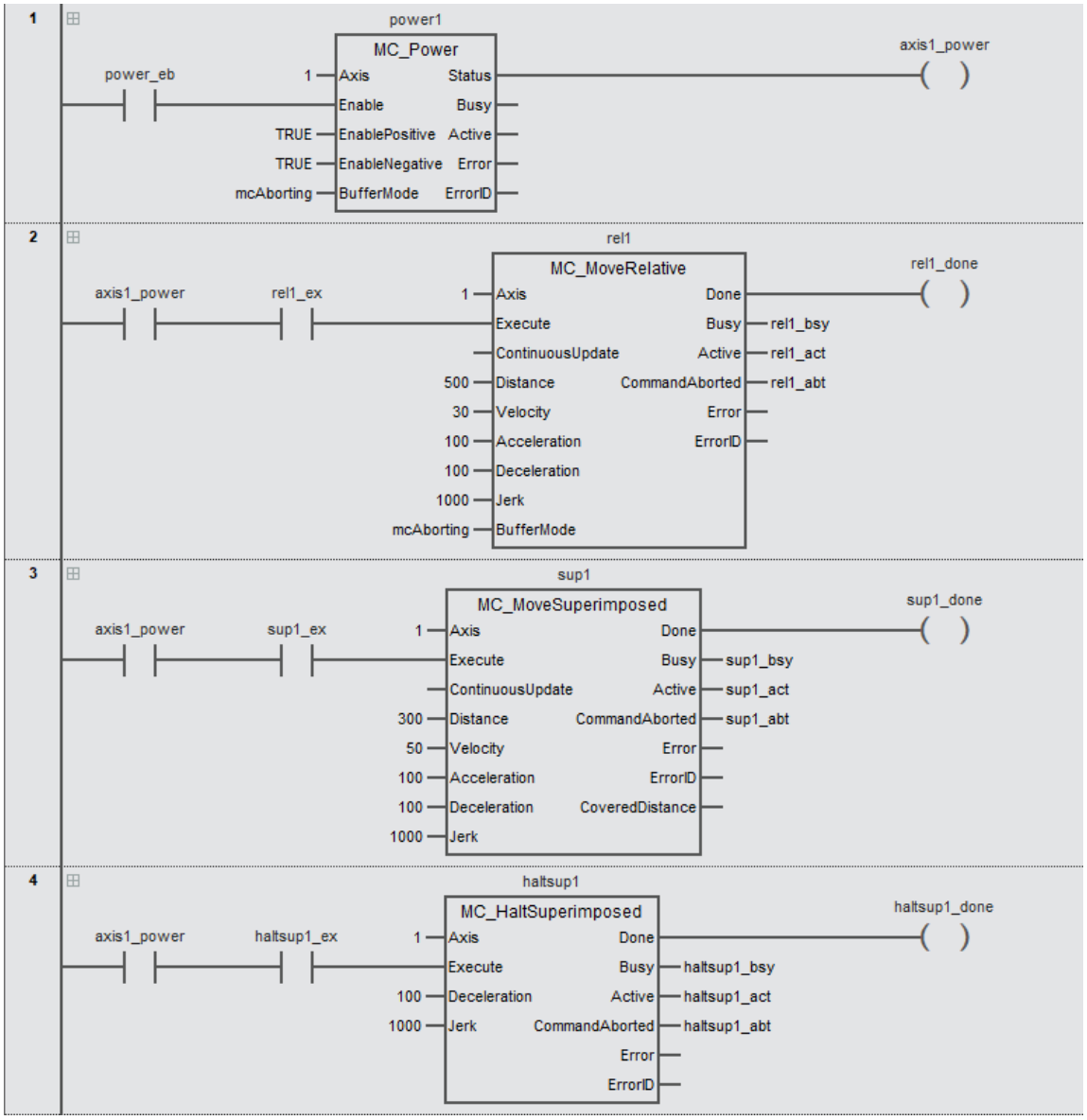
  The axis parameter setting for axis1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_ex		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_bsy		BOOL		
VAR	rel1_act		BOOL		
VAR	rel1_abt		BOOL		
VAR	sup1_ex		BOOL		
VAR	sup1		MC_MoveSuperimposed		
VAR	sup1_done		BOOL		
VAR	sup1_bsy		BOOL		
VAR	sup1_act		BOOL		
VAR	sup1_abt		BOOL		
VAR	hatsup1_ex		BOOL		
VAR	hatsup1		MC_HaltSuperimposed		
VAR	hatsup1_done		BOOL		
VAR	hatsup1_bsy		BOOL		
VAR	hatsup1_act		BOOL		
VAR	hatsup1_abt		BOOL		

- LD

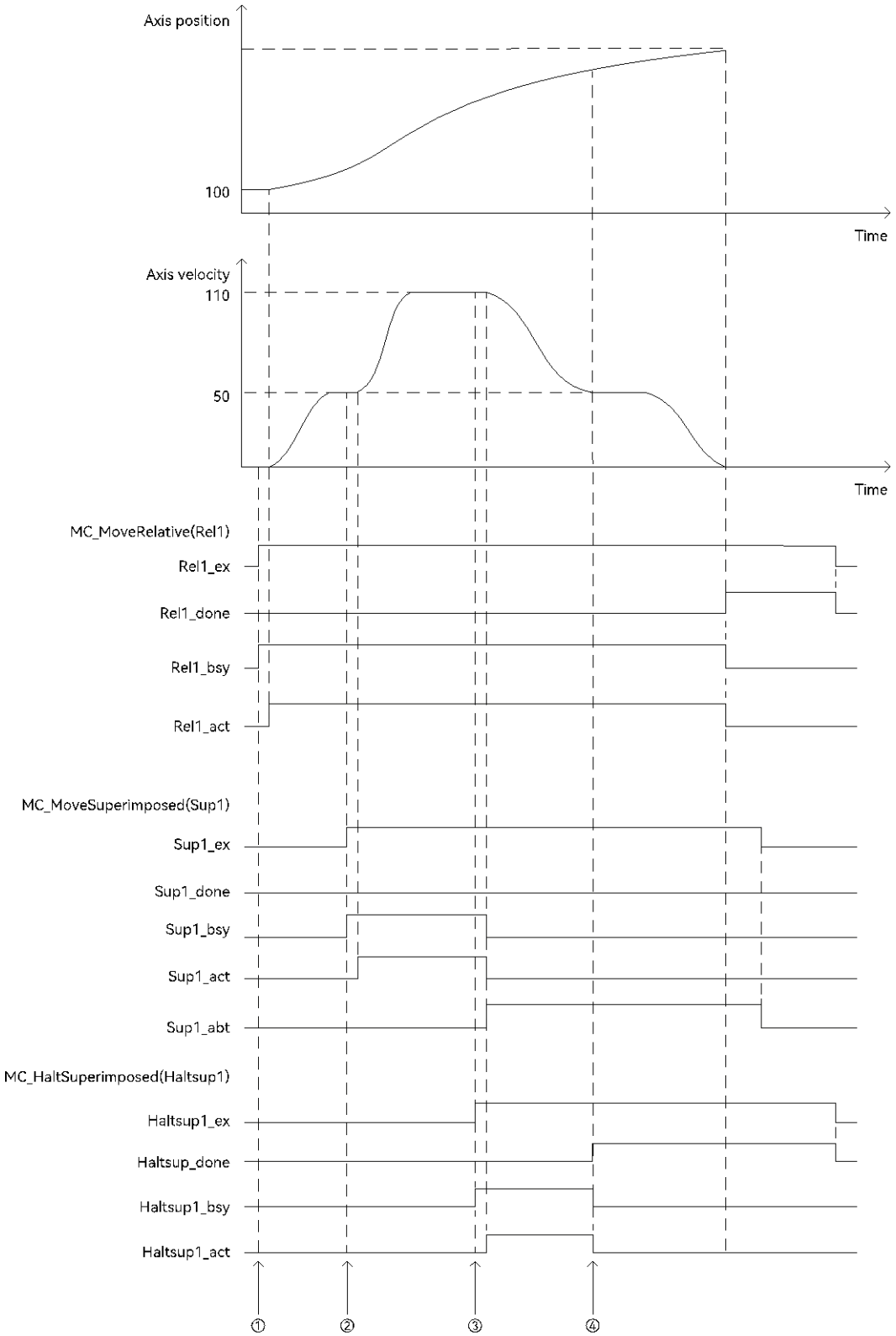


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);


rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_ex ,
    Distance:=500 ,
    Velocity:=50 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=100 ,
```

```
        BufferMode:=mcAborting ,
        Done=>rel1_done ,
        Busy=>rel1_bsy ,
        Active=>rel1_act ,
        CommandAborted=>rel1_abort );


    sup1 (
        Axis:=1 ,
        Execute:= axis1_power AND sup1_ex ,
        Distance:=300 ,
        Velocity:=60 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        Done=> sup1_done ,
        Busy=> sup1_bsy ,
        Active=>sup1_act ,
        CommandAborted=> sup1_abt
        );


    hatsup1 (
        Axis:=1 ,
        Execute:= axis1_power hatsup1_ex
        Deceleration:=100 ,
        Jerk:=1000 ,
        Done=> hatsup1_done ,
        Busy=> hatsup1_bsy ,
        Active=> hatsup1_act ,
        CommandAborted=> hatsup1_abt );
```

- **Program description**
  - After the axis is enabled, rel1 starts to execute when re1_execute becomes TRUE. When rel1 controls axis operation and sup1_execute becomes TRUE, sup1 is executed. After sup1 controls the axis, the velocity, acceleration, and distance generated by the two instructions are superimposed and then acted on one axis.
  - If the corrected position is found to be unreasonable or incorrect after starting the MC_MoveSuperImposed instruction, the position correction can be stopped by the MC_HaltSuperImposed instruction.
  - As shown in the following figure, haltsup1 is executed when haltsup1_execute becomes TRUE. The execution of this instruction only interrupts the execution of sup1 and has no effect on the execution of rel1. When haltsup1 is completed, the rel1 is still executing and the velocity of the axis becomes 50 (the target velocity of rel1) from 110 (the superimposed velocity of rel1 and sup1).



① The relative instruction (rel1) starts execution, and the axis is controlled in the 2nd cycle.

② The superimposed instruction (sup1) starts execution, and start to control the axis in the 2nd cycle. the velocity and acceleration of the relative instruction and the of the superimposed instruction are superimposed.

③ Stop the execution of the superimposed instruction (haltsup1).

④ When the superimposed instruction (sup 1) is completed (Done is TRUE), the relative instruction (rel1) has not been completed and is still controlling the axis.

4.17 MC_SetOverride (velocity factor setting)

This instruction is used to set the global velocity ratio of the specified axis. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetOverride	Velocity factor setting	FB	MC_SetOverride_Instance MC_SetOverride Axis Enable VelFactor AccFactor JerkFactor Enabled Busy Error ErrorID	MC_SetOverride _Instance (Axis:=parameter , Enable:=parameter , VelFactor:=parameter , AccFactor:=parameter , JerkFactor :=parameter, Enabled => parameter, Busy=> parameter , Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Factor in effect for target velocity FALSE: The factor of the target velocity becomes "100%".
VelFactor	Factor of target velocity	LREAL	0~500	100	Factor of target velocity (Unit: %)
AccFactor	Factor of acceleration	LREAL	Positive number	Required field	Reserved
JerkFactor	Factor of jerk	LREAL	Positive number	Required field	Reserved

■ Output variable

Name	Meaning	Data type	Valid range	Description
Enabled	Enabled	BOOL	TRUE or FALSE	TRUE when the axis is under control.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error Code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Enabled	When Enable becomes TRUE	◆ When Enable becomes FALSE ◆ When Error becomes TRUE
Busy	When Enable becomes TRUE	◆ When Enable becomes FALSE ◆ When Error becomes TRUE
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction	◆ When Error is TRUE and Enable changes from TRUE to FALSE

■ Function description

● Basic function description

- The MC_SetOverride instruction changes override factors related to the target velocity of the axis. When the Enable of this instruction is TRUE, the target velocity of the specified axis can be changed in real-time by modifying the value of the input variable. VelFactor, whose unit is percentage, such as "50" means "50%", the valid range of VelFactor is 0~500. If it is out of the range, the instruction will report an error.
- The override factors apply only to the following instructions.

MC_MoveAbsolute	MC_MoveRelative
MC_MoveAdditive	MC_MoveVelocity
MC_MoveSuperimposed	MC_JOG

- After this instruction is executed, the new target velocity is as follows:

The changed target velocity = the target velocity of the current instruction × VelFactor

- When the Enable of this instruction is TRUE, the value of the input variable VelFactor is changed to accelerate or decelerate to the target velocity according to the input variables of the current axis instruction. If the value of the input variable VelFactor is changed when the MC_MoveVelocity instruction is executed, acceleration or deceleration to the target velocity is performed according to the set values of the input variables Acceleration, Deceleration, and Jerk of the MC_MoveVelocity instruction.
- When VelFactor is set to 0, the target velocity will decrease to 0. If users want to pause the axis motion while maintaining the operation status, set the VelFactor to 0. Axis state in the Axis Variable does not change at this time and will be determined by the motion instruction of the control axis.
- When the Enable of this instruction changes from TRUE to FALSE, the specified axis changes from the current velocity to the target velocity of the current control axis instruction, which is equivalent to the case where the value of VelFactor is 100. The axis velocity change is accelerated or decelerated to the target velocity according to the input variable of the current control axis instruction. If the value of VelFactor is 200 and Enable is TRUE when the MC_MoveVelocity instruction is executed and Enable becomes FALSE from TRUE, the axis velocity is accelerated or decelerated to the target velocity according to the set values of Acceleration, Deceleration, and Jerk of MC_MoveVelocity instruction input variables.
- After the MC_MoveVelocity instruction is executed and the input variable InVelocity becomes TRUE, change the value of VelFactor and the target velocity of the axis will change, but the output variable InVelocity of the MC_MoveVelocity instruction remains TRUE.

● Re-execute the instruction

This instruction is executed constantly when Enable is TRUE.

● Abnormality elimination

When this instruction is executed, If the value of the instruction input variable is out of valid range or does not satisfy the execution conditions of the instruction, Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

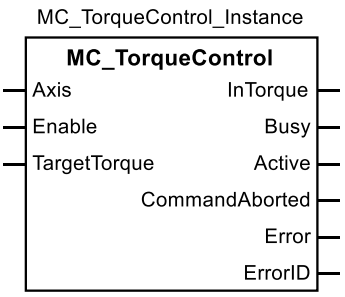
● Example program

For example, programs, please refer to the MC_MoveVelocity instruction.

4.18 MC_TorqueControl (torque control)

The MC_TorqueControl instruction controls the output torque of the Servo Drive in Torque Control Mode.
Library: MotionControl

Applicable models: M500S Series, M500 Series

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_TorqueControl	Torque control	FB		MC_TorqueControl_Instance (Axis:=parameter , Enable:=parameter , TargetTorque:=parameter , InTorque => parameter, Busy => parameter, Active=> parameter , CommandAborted => parameter, Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: The Servo Drive enters torque mode and transmits the target torque to the Servo Drive FALSE: Servo Drive exits torque mode and enters CSP mode
TargetTorque	Target torque	INT	Positive number, Negative number, 0	0	Specify the target torque transmitted to the Servo Drive in increments of［0.1%］, specify a percentage of the rated torque, i.e., the rated torque is 100.0%. If the target torque is set to 30, it means that the target torque is set to three percent of the rated torque, and if Enable is set to TRUE, the value of this parameter is transmitted to the drive in real time.

■ Output variable

Name	Meaning	Data type	Valid range	Description
InTorque	Target torque reached	BOOL	TRUE or FALSE	TRUE when the controller's instruction torque to the drive reaches the target torque
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an ins truction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
InTorque	When target torque is output	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Busy	When Enable becomes TRUE	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE

Active	When the instruction is started	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction	◆ When Enable changes from TRUE to FALSE

■ Function description

● Basic function description

➢ When the Enable of this instruction is TRUE, the axis works in torque mode, and the controller transmits the set TargetTorque to the specified axis in real-time via EtherCAT bus, and the controller does not participate in the calculation. If users need the torque of the specified axis to change slowly, it is recommended to use the MC_TorqueControlWithVelocity instruction or write a program to control the process of changing the TargetTorque value, or set the parameter of the torque change rate of the specified axis to realize it.

➢ When Enable is FALSE, the axis immediately exits torque mode and enters CSP mode.

● PDO mapping

PDO is not required for M500S series controllers but for M500 series，the mapping objects are shown in the following table.

PDO reception (master => slave) (hexadecimal)	Meaning of the mapped data	PDO sending (slave => master)	Meaning of the mapped data
6040_0 (index_subindex)	Control Word	6041_0 (index_subindex)	Status word
6060_0 (index_subindex)	Control Mode	6061_0 (index_subindex)	Actual mode
6071_0 (index_subindex)	Target torque		

● Execute this instruction while other instructions are in processing

This instruction will report an error if it is started when another motion instruction is executed.

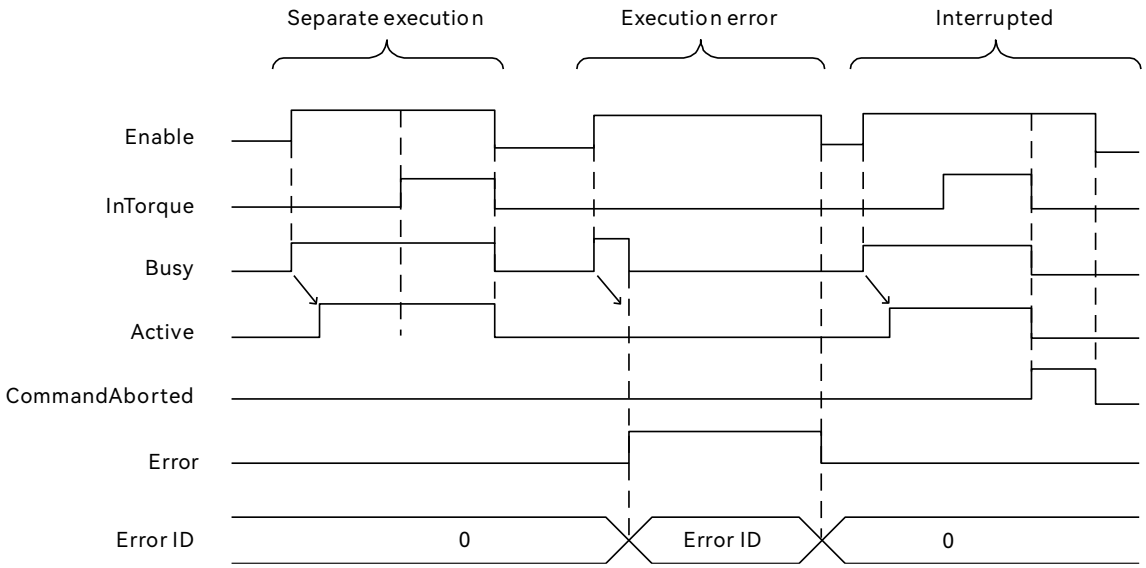
● Execute other instructions while this instruction is in processing

When this instruction is executed, you can execute the MC_Stop instruction to interrupt this instruction If other instructions are executed, the other instructions report an error.

● Abnormality elimination

When this instruction is executed, if the value of the instruction input variable is out of valid range, Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

■ Output variable timing diagram description



- **Separate execution**

  When Enable changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When target torque is output, InTorque becomes TRUE, and Busy and Active change to FALSE at the same time.

- **Execution error**

  When the value of the input variable is not within the allowable range, Enable changes from FALSE to TRUE while Busy becomes TRUE. The next period Error becomes TRUE while Busy becomes FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error becomes FALSE, and the value of ErrorID becomes 0.

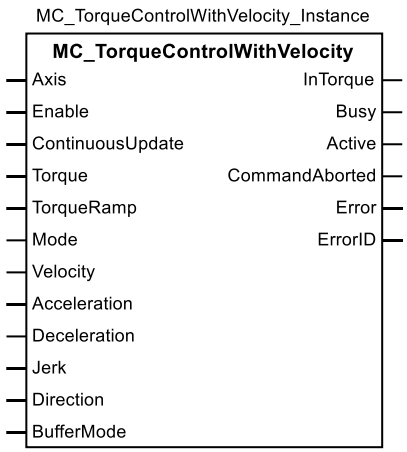
- **Interrupted**

  When the execution of this instruction is aborted by another instruction, CommandAborted becomes TRUE, Busy and Active change to FALSE at the same time; when Enable becomes FALSE, CommandAborted becomes FALSE at the same time.

4.19 MC_TorqueControlWithVelocity (torque control with velocity)

The MC_TorqueControlWithVelocity instruction controls the torque of the axis in real-time and limits its velocity in Torque Control Mode. Library: MotionControl

Applicable models: M500S Series, M500 Series

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_TorqueControlWithVelocity	Torque control with velocity	FB		MC_TorqueControlWithVelocity_Instance (Axis :=parameter, Enable :=parameter , ContinuousUpdate :=parameter , Torque:=parameter , TorqueRamp:=parameter , Mode:=parameter , Velocity:=parameter , Acceleration :=parameter, Deceleration :=parameter, Jerk:=parameter , Direction:=parameter , BufferMode :=parameter, InTorque => parameter, Busy => parameter, Active=> parameter , CommandAborted=> parameter , Error => parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: The Servo Drive enters torque mode and transmits the target torque to the Servo Drive FALSE: Servo Drive exits torque mode and enters CSP mode
ContinuousUpdate	Continuous update	BOOL	TRUE or FALSE	FALSE	Reserved
Torque	Target torque	INT	Positive number, Negative number, 0	0	Specify the target torque to output to the Servo Drive in increments of［0.1%］, specify a percentage of the rated torque, i.e., the rated torque is 100.0%. If the target torque is set to 30, it means that the target torque is set to three percent of the rated torque, and if Enable is set to TRUE, the value of this parameter is transmitted to the drive in real time
TorqueRamp	Torque ramp	LREAL	Positive number, Negative number	Required field	Rate of change from the current set torque to the changed set torque (unit: ‰/s)
Mode	Limiting speed parameter selection	INT	0, 1	0	0: Velocity is associated with the Servo internal parameter 16#6080; 1: Velocity is associated with Servo internal parameter 16#607F;

Velocity	Velocity limit	LREAL	Positive number	Required field	When the MC_TorqueControlWithVelocity instruction controls an axis, the velocity of the limit axis cannot exceed this value. When the value of Mode is different, the servo parameter associated with this parameter is different, no matter which servo parameter is selected, the unit of this parameter is: Travel unit/s.
Acceleration	Acceleration rate	LREAL	Positive number	Reserved	Reserved
Deceleration	Deceleration rate	LREAL	Positive number	Reserved	Reserved
Jerk	Jerk	LREAL	Positive number	Reserved	Reserved
Direction	Direction	MC_Direction	1：mcPositiveDirection 3：mcNegativeDirection	1	Specify the direction of rotation of the axis. 1: Positive direction 3: Negative direction
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting	Reserved	Reserved

■ Output variable

Name	Meaning	Data type	Valid range	Description
InTorque	Target Torque Reached	BOOL	TRUE or FALSE	TRUE when the target torque is reached
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
InTorque	When target torque is output	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Busy	When Enable becomes TRUE	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
Active	When the instruction is started	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction	◆ When Enable changes from TRUE to FALSE

■ Function description

● Basic function description

➢ When the Enable of this instruction is TURE, the axis works in torque mode, and the controller transmits the limiting velocity (Velocity) for torque mode operation to the specified axis. The controller targets the set target torque, and varies the torque value according to the set torque change rate

(TorqueRamp), then transmits the calculated torque value to the specified axis in real time. The controller maintains the target torque output after the torque to the specified axis reaches the target torque. When the target torque value is changed, the controller changes the torque value according to the torque rate of change (TorqueRamp) and transmits the torque to the specified axis in real-time.

- TorqueRamp in units of ‰/s, the target torque (Torque) is a thousandth of the rated torque. The time required to change from the current torque to the target torque is calculated as follows:

  Time for current torque to reach target torque: = (Target torque - Current given torque) / TorqueRamp

- When Enable is FALSE, the axis immediately exits torque mode and enters CSP mode.

- **PDO mapping**

  PDO is not required for M500S series controllers but for M500 series, the mapping objects are shown in the following table.

PDO reception (master => slave) (hexadecimal)	Meaning of the mapped data	PDO sending (slave => master)	Meaning of the mapped data
6040_0 (index_subindex)	Control Word	6041_0 (index_subindex)	Status word
6060_0 (index_subindex)	Control Mode	6061_0 (index_subindex)	actual mode
6071_0 (index_subindex)	Target torque		
607F_0 （index _subindex） or 6080_0，configure the corresponding parameters according to the Mode value of this instruction	Velocity limit		

- **Execute this instruction while other instructions are in processing**

  This instruction will report an error if it is started when another motion instruction is executed.

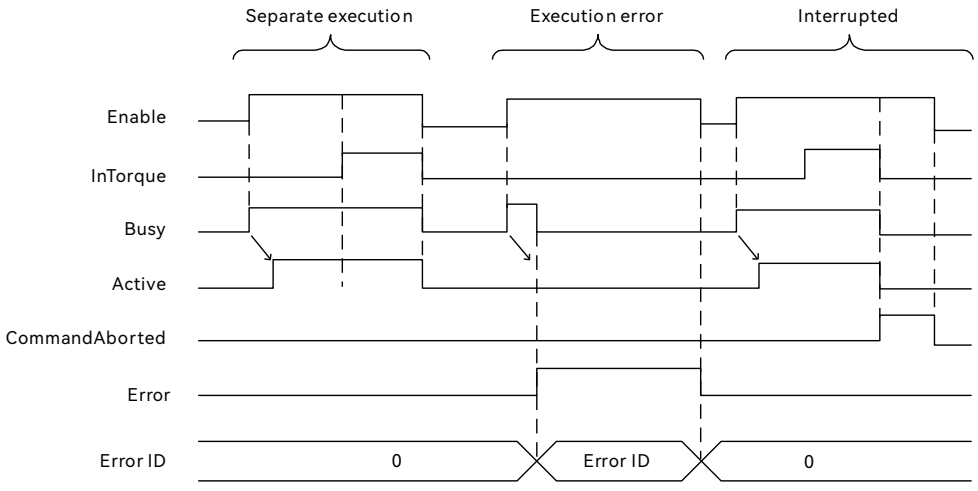
- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, users can execute the MC_Stop instruction to interrupt this instruction. If other instructions are executed, the other instructions will report an error.

- **Abnormality elimination**

  When this instruction is executed, If the value of the instruction input variable is out of valid range, the Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

■ **Output variable timing diagram description**



- **Separate execution**

  When Enable changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When target torque is output, InTorque becomes TRUE, and Busy and Active change to FALSE at the same time.

- **Execution error**

  When the value of the input variable is not within the allowable range, Enable changes from FALSE to TRUE while Busy becomes TRUE. In the next period Error becomes TRUE while Busy becomes FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error becomes FALSE, and the value of ErrorID becomes 0.

- **Interrupted**

  When the execution of this instruction is aborted by another instruction, CommandAborted becomes TRUE, Busy, and Active change to FALSE at the same time; when Enable becomes FALSE, CommandAborted becomes FALSE at the same time.

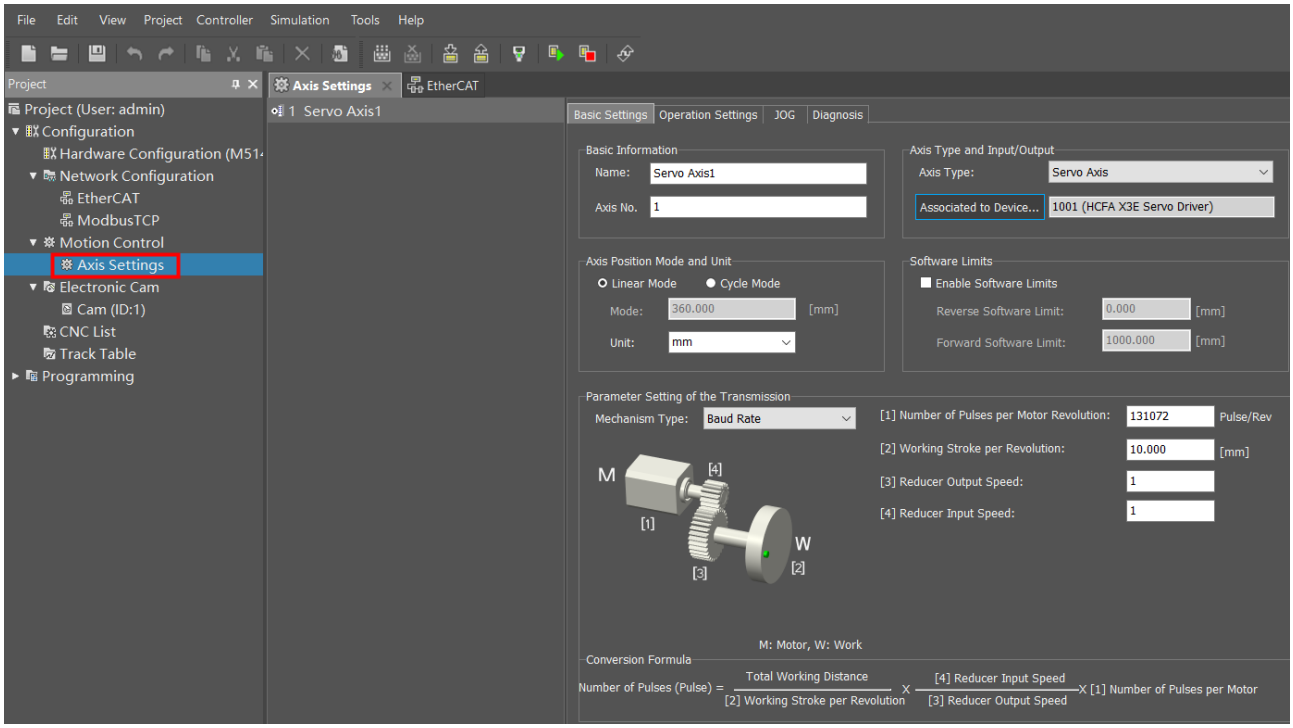
■ Example program

- **Functionality**

  The specified axis is controlled to operate in torque mode to reach the target torque, the axis reaches the target torque at TorqueRamp and limits the maximum velocity of the axis.

- **Axis parameter setting**

  The axis parameter setting for axis1 is shown below.

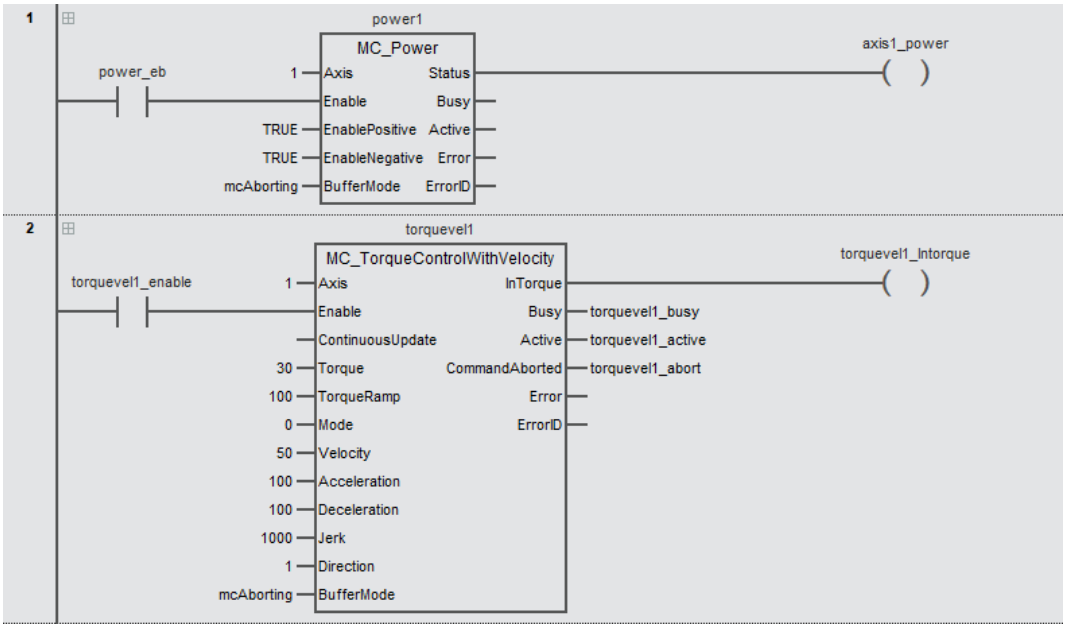


- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	torquevel1		MC_TorqueControlWithVelocity		
VAR	torquevel1_enable		BOOL		
VAR	torquevel1_intorque		BOOL		

VAR	torquevel1_busy		BOOL		
VAR	torquevel1_active		BOOL		
VAR	torquevel1_abort		BOOL		

- LD



- ST

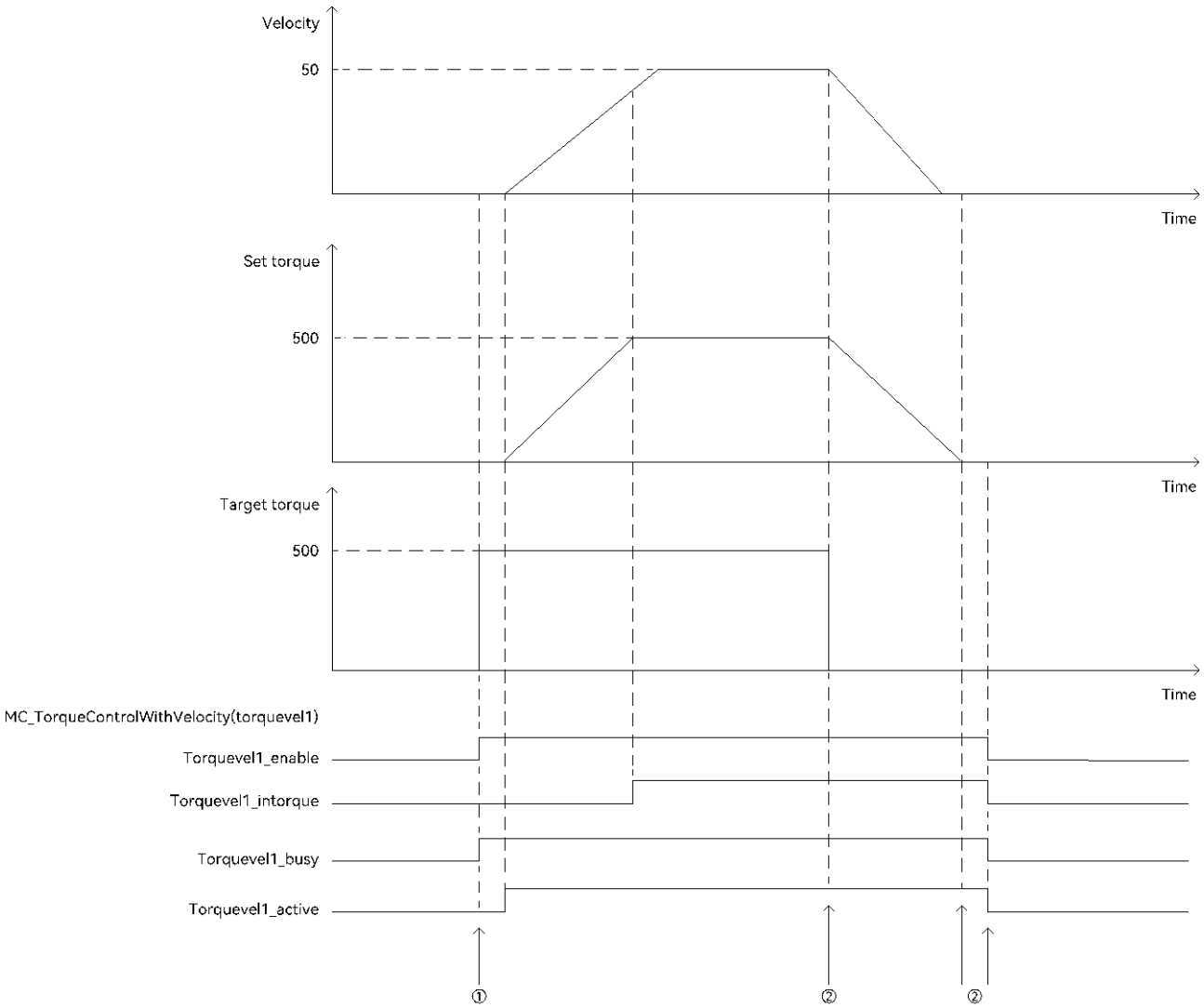
```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );

torquevel1(
    Axis:=1 ,
    Enable:=torquevel1_enable ,
    Torque:=30 ,
    TorqueRamp:=100 ,
    Mode:=0 ,
    Velocity:=50 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Direction:=1 ,
    BufferMode:=mcAborting ,
    InTorque=>torquevel1_intorque ,
    Busy=>torquevel1_busy ,
    Active=>torquevel1_active ,
    CommandAborted=>torquevel1_abort );
```

- Program description

  ➢ After the axis is enabled, the torquevel1 instruction starts to execute when torquevel1_enable

becomes TRUE. This instruction controls axis 1 to work in torque mode to reach the target torque according to the set TorqueRamp and limits the maximum velocity of the axis to not exceed Velocity. The command torque change process transmitted by the controller to axis 1 is shown in the figure below. After the command torque reaches the set torque, torque1_intorque becomes TRUE and maintains the target torque output.

- After setting the target torque to 0, the command torque becomes 0 according to TorqueRamp. The change process is shown below.
- After the command torque becomes 0 and torquevel1_enable is set to FALSE, torquevel1_intorque, torquevel1_busy, and torquevel1_active all change to FALSE at the same time, and axis 1 exits the torque mode and enters the CSP mode, limit velocity (Velocity) becomes the value before Enable changes from FALSE to TRUE.

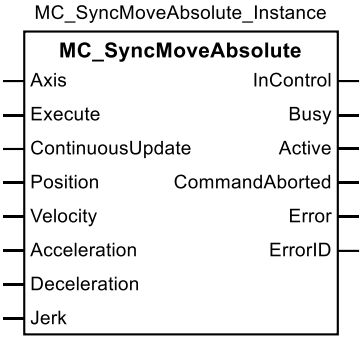


① The torque command with velocity limit (torquevel1) starts execution, and the command torque changes to the target torque according to the set value of the torque change rate.

② Setting the target torque to 0, the command torque changes to 0 in accordance with the set value of the torque change rate, and after the command torque becomes 0, the drive exits the torque mode by setting torquevel1_enable to FALSE again.

4.20 MC_SyncMoveAbsolute（periodic synchronous absolute positioning）

The MC_SyncMoveAbsolute instruction outputs the specified target position for the axis periodically. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SyncMoveAbsolute	Periodic synchronous absolute positioning	FB		MC_SyncMoveAbsolute_Instance (Axis :=parameter, Execute:=parameter , ContinuousUpdate:=parameter , Position:=parameter , Velocity :=parameter, Acceleration:=parameter , Deceleration:=parameter , Jerk :=parameter, InControl=> parameter , Busy=> parameter , Active=> parameter , CommandAborted => parameter, Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
ContinuousUpdate	Continuous Update	BOOL	TRUE or FALSE	FALSE	Reserved
Position	Absolute position	LREAL	Positive number, Negative number, 0	0	Specify the absolute position to move from the current position to the set position in one period. (Unit: Travel unit)*1
Velocity	Target velocity	LREAL	Positive number	Required field	Reserved
Acceleration	Acceleration rate	LREAL	Positive number	Required field	Reserved
Deceleration	Deceleration rate	LREAL	Positive number	Required field	Reserved
Jerk	Target velocity	LREAL	Positive number	Required field	Reserved

*1:For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "*Parameter description of motion control instruction*".

■ Output variable

Name	Meaning	Data type	Valid range	Description
InControl	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error

ErrorID	Error Code	WORD	0~65535	Refer to "*instruction error code description*" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
InControl	When the instruction controls an axis	◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Busy	When Execute becomes TRUE	◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE, CommandAborted becomes TRUE and Execute becomes FALSE. ◆ When the instruction has been executed and Execute becomes FALSE, and the instruction is aborted by other instructions, CommandAborted becomes TRUE, and after one period, it becomes FALSE.
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction	◆ When Error is TRUE and Execute changes from TRUE to FALSE. ◆ The instruction has been executed and Execute becomes FALSE. If an exception is encountered during the execution of the instruction, Error becomes TRUE, and after one period, it becomes FALSE.

■ Function description

● Basic function description

- This instruction is used to pass the incremental value of the input variable Position for each period plus the axis commanded position for the previous period to the specified axis. The incremental value for each period starts to be calculated when the output variable active becomes TRUE, the first incremental value is based on the value of the input variable Position when Execute becomes TRUE. It is recommended that the axis actual position be moved into the input variable Position when Execute becomes TRUE.
- This instruction is generally used for planning the motion trajectory. The control of the axis is performed by controlling the value of the input variable Position.
- The following table shows the two cases when the axis is controlled by this instruction, and users can understand the control method of this instruction better by these two cases.
- The instruction is activated when Execute becomes TRUE from FALSE, and the instruction can still be executed even if Execute becomes FALSE again. If users want to terminate the instruction, they can execute MC_Halt or MC_Stop instruction to interrupt the instruction.

Case 1:

Execution period	A period where Busy becomes TRUE	Period 1 where Active becomes TRUE	Period 2 where Active becomes TRUE
Position value	500	530	580
Axis command position	500	530	580
Axis actual position	500	530	580

Case 2:

Execution period	A period where Busy becomes TRUE	Period 1 where Active becomes TRUE	Period 2 where Active becomes TRUE
Position value	500	530	580
Axis command position	300	330	380
Axis actual position	300	330	380

- **Instruction completion timing**

  After Execute changes from FALSE to TRUE, this instruction continues to execute, passing the amount of change in Position from one task period to the axis.

- **Re-execute the instruction**

  After Execute changes from FALSE to TRUE, the instruction continues to execute, after which Execute changes again from FALSE to TRUE, with no effect on the execution of the instruction. If users want to re-execute this instruction, they can execute MC_Halt or MC_Stop instruction to interrupt this instruction, and then execute this instruction again to restart.

- **Execute this instruction while other instructions are in processing**

  This instruction will report an error if it is started when another motion instruction is executed.

- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, users can execute MC_Halt or MC_Stop instruction to interrupt this instruction. If other instructions are executed, the other instructions will report an error.

- **Abnormality elimination**

  When this instruction is executed, if the input variable is illegal, the Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g., an axis error), the instruction will also report an error and the axis will stop immediately.

4.21 MC_SetPosition (position setting)

The MC_SetPosition instruction changes the command position or actual position of the specified axis to the set value. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetPosition	Position setting	FB	MC_SetPosition_Instance MC_SetPosition Axis / Execute / Position / Relative / ReferenceType / ExecutionMode Done / Busy / Error / ErrorID	MC_SetPosition _Instance (Axis :=parameter, Execute:=parameter , Distance :=parameter, Relative:=parameter , Reference Type:=parameter , ExecutionMode:=parameter , Done=> parameter , Busy=> parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
Position	Target position	LREAL	Positive number, Negative number, 0	0	Specify the target position （Unit: Travel unit）
Relative	Relative position selection	BOOL	TRUE or FALSE	FALSE	Specifies the relative position type TRUE: Relative FALSE: Absolute
ReferenceType	Position type selection	MC_ ReferenceType	0：mcCommandPosition 1：mcActualPosition	0	Specifies the position type 0: Command position 1: Actual position
ExecutionMode	Execution mode	Reserved	Reserved	Reserved	Reserved (This input pin can be left unfilled with variables or constants)

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error Code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done becomes TRUE and then FALSE one period later
Busy	When Execute becomes TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE

Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Execute changes from TRUE to FALSE

■ Function description

● Basic function description

- ➢ The MC_SetPosition instruction changes the commanded or actual position of the specified axis to the set value. The execution of this instruction does not cause the axis to move or change the physical position of the specified axis, and is generally used to redefine the position coordinates or coordinate offsets. This instruction is realized by the controller's internal arithmetic, and will not change the position of the servo drive when operated on a servo axis.
- ➢ This instruction can be executed for the servo axis, virtual servo axis, pulse axis, and encoder axis. Because the command and actual positions of the virtual servo axis and encoder axis are always the same, there is no difference when selecting the command position or the actual position for operation.
- ➢ After this instruction is executed, the difference between the commanded position of the axis and the actual position is kept unchanged. This instruction follows the command position and actual position of the axis whether it is specified to operate on the command position or actual position of the axis. In absolute mode, this instruction specifies the following formula for calculating the axis position when operating on the command position or actual position:

  This instruction specifies the formula for calculating the actual position of the axis when operating on the command position as follows:

  Actual position：= Position - Difference between the axis command and actual position when this instruction is activated

  This instruction specifies the formula for calculating the commanded position of the axis when operating on the actual position as follows:

  Command position：= Position + Difference between the axis command and actual position when this instruction is activated
- ➢ This instruction can be executed both when the axis is stationary and when it is moving.

● Position, relative, reference type selection

- ➢ When this instruction is executed, the commanded position and the actual position of the axis change according to the input variables of the instruction. The input variable Position sets the position; Reference type is used to select the command position or the actual position, and a setting of 0 indicates the command position operation for the axis, and a setting of 1 indicates the actual position operation for the axis; Relative is used to set whether the position of Position and axis is relative or absolute, Relative is FALSE, which means that the position of Position and axis is absolute, and the position of axis is set to the value of Position, Relative is TRUE, which means that the position of Position and axis is relative, and the position of axis is set to the value of Position. The value of Position becomes the axis position.
- ➢ When the axis is stationary, generally the command position and actual position of the axis are the same, and after executing this instruction, the command position and actual position of the axis remain the same. When the axis moves, generally the command position and the actual position of the axis have a difference. Assuming that the instruction is executed when the difference between the command position and the actual position of the axis is 200, the command position and the actual position obtained by the instruction according to the different set values of the input variables are

shown in the table below.

Parameter	Value			
Relative (Relative position selection)	FALSE：Absolute position	FALSE：Absolute position	TRUE：Relative position	TRUE：Relative position
ReferenceTyp (Position type selection)	1：Actual position	0：Command position	1：Actual position	0：Command position
Position (Target position)	1000	1000	1000	1000
Command position before execution	2000	2000	2000	2000
Actual position before execution	1800	1800	1800	1800
Position difference before execution	200	200	200	200
Command position after execution	1200	1000	3000	3000
Actual position after execution	1000	800	2800	2800
Position difference after execution	200	200	200	200

- **Instruction application description**
  - When MC_SetPosition is executed for a master axis that has established a multi-axis relationship, the change in the position of the master axis caused by MC_SetPosition does not affect the slave axis. When applied to a slave axis, the position of the slave axis changes, but its original synchronization relationship with the master axis is not affected.
  - If the MC_SetPosition instruction is executed during the execution of the MC_Stop instruction (not completed), the MC_SetPosition instruction will report an error; if the MC_SetPosition instruction is executed after the execution of the MC_Stop instruction is completed, the MC_SetPosition instruction can be executed normally.
- **Re-execute the instruction**

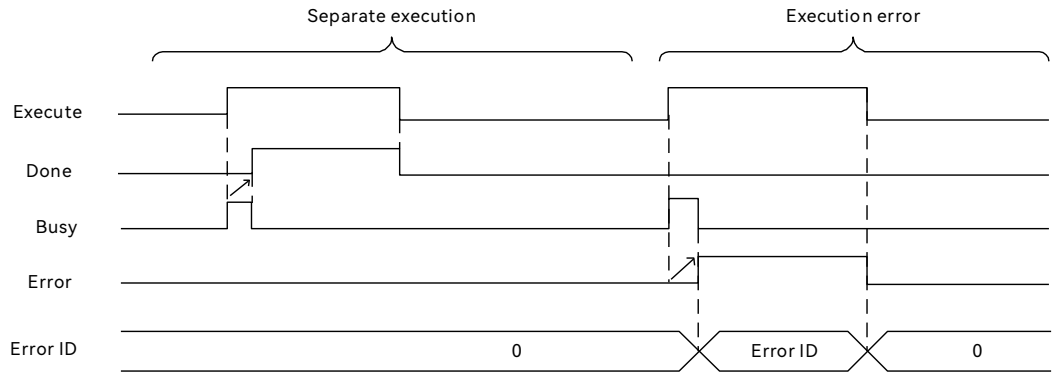
  When the execution of the instruction is complete and Execute changes from FALSE to TRUE again, When this instruction is being executed and Execute changes from FALSE to TRUE again, the execution of instructions will not be affected, the instruction continues to be executed in a previous way.
- **Execute this instruction while other instructions are in processing**

  This instruction is executed when other instructions are executed, and this instruction does not affect the motion of other instructions. However, after this instruction and other motion instructions are executed, because the commanded position and the actual position of the axis change, make sure that the instruction is fully understood before executing other motion instructions.
- **Execute other instructions while this instruction is in processing**

  Other motion instructions can be executed while this instruction is executed. Because the command position and the actual position of the axis that executes this instruction will change, please make sure that the instruction is fully understood before executing other motion instructions.
- **Abnormality elimination**

  When this instruction is executed, if the input variable is illegal, the Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

■ Output variable timing diagram description



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy becomes TRUE. In the next period, Error becomes TRUE while Busy becomes FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error becomes FALSE and the value of ErrorID becomes 0.

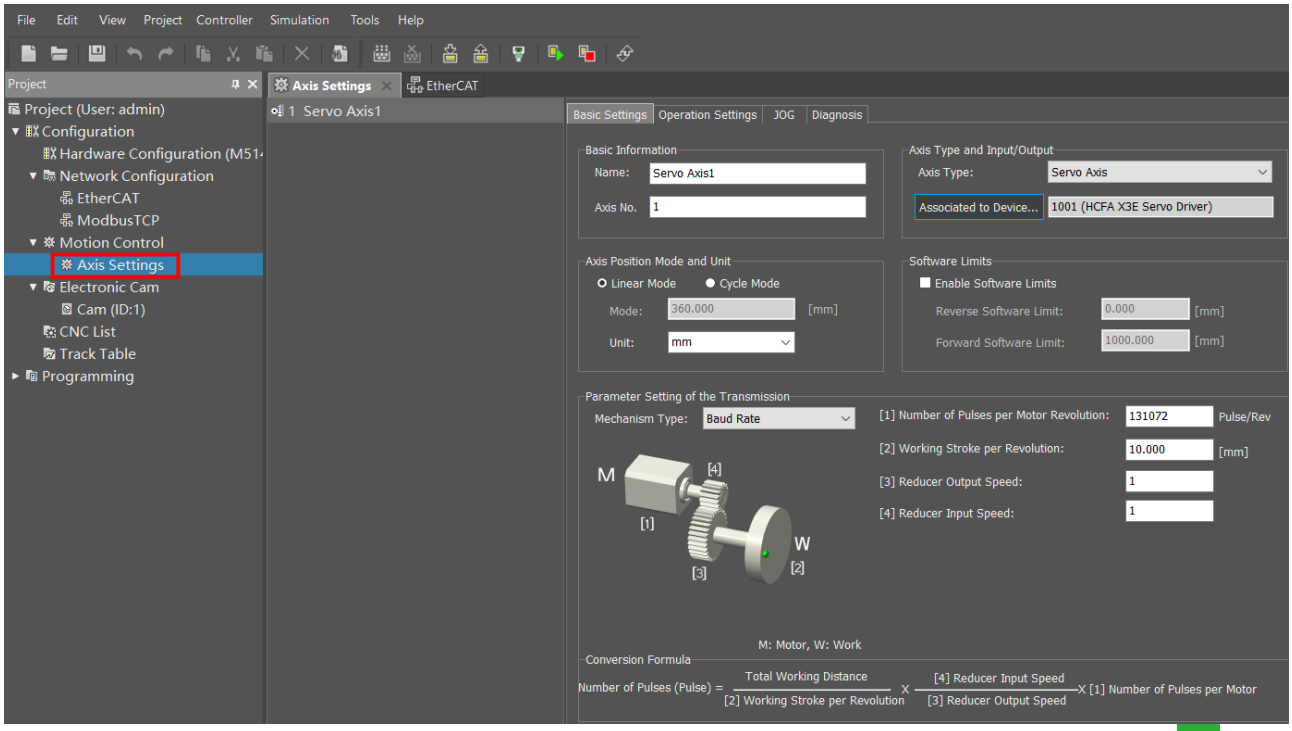
■ Example program

- **Functionality**

  The current position of the axis is set to the position set by the user through the MC_SetPosition instruction. After the position is set, the current position is used as the reference point and then the specified relative position is traveled.

- **Axis parameter setting**

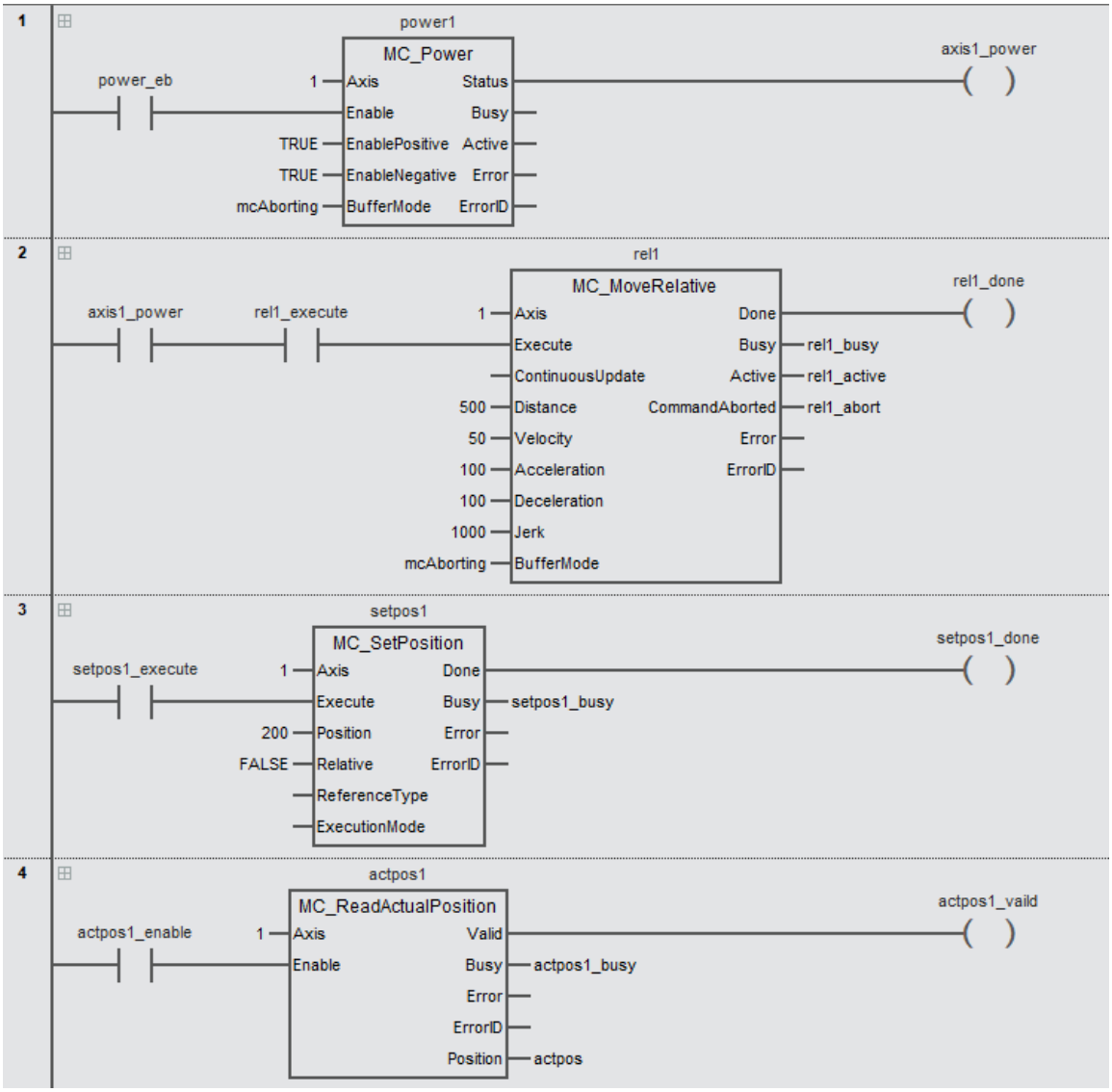
  The axis parameter setting for axis1 is shown below.



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_execute		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	setpos1_execute		BOOL		
VAR	setpos1		MC_SetPosition		
VAR	setpos1_done		BOOL		
VAR	setpos1_busy		BOOL		
VAR	actpos1		MC_ReadActualPosition		
VAR	actpos1_enable		BOOL		
VAR	actpos1_vaild		BOOL		
VAR	actpos1_busy		BOOL		
VAR	actpos		LREAL		

- LD

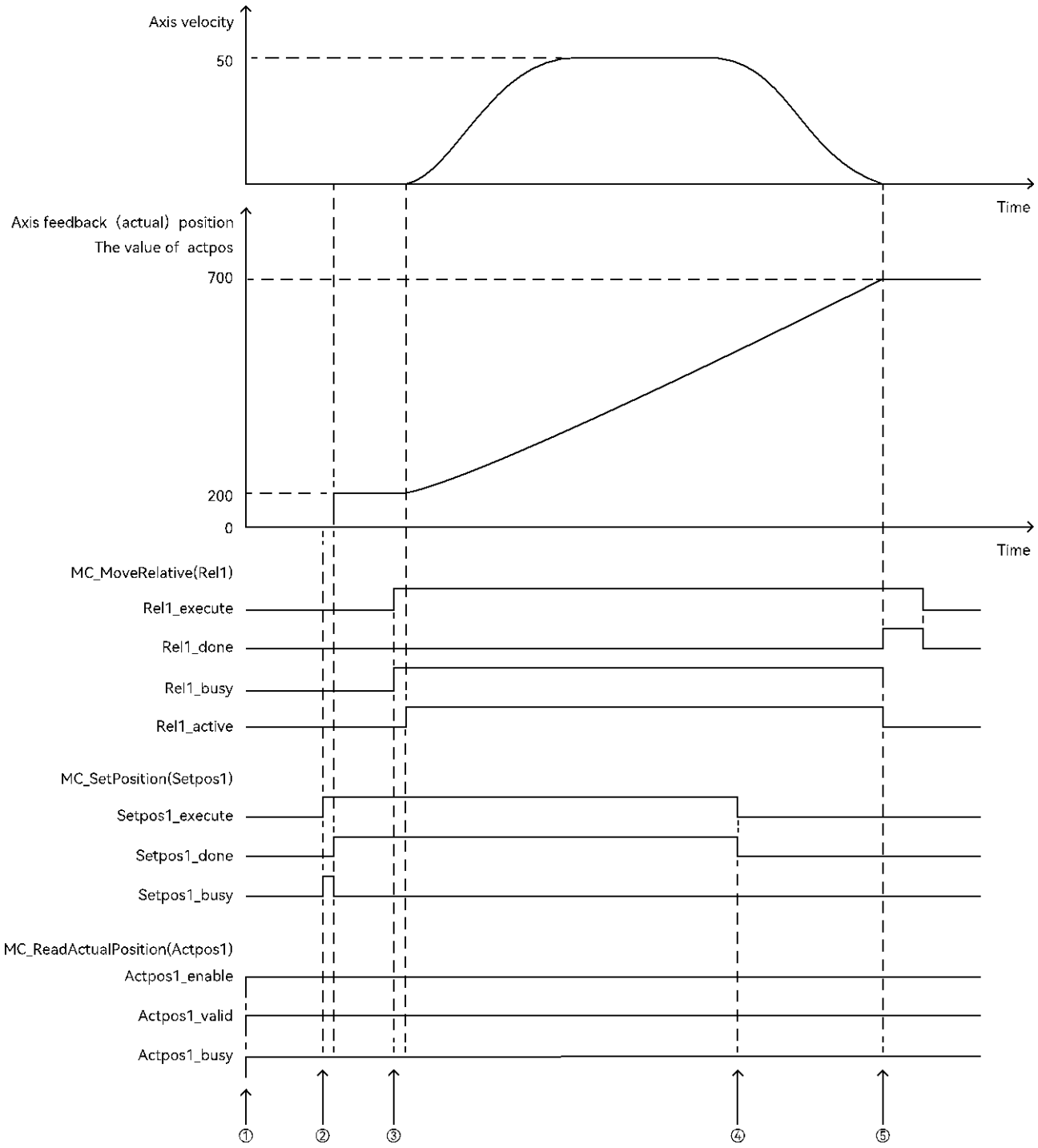


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );

rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_execute ,
    Distance:=500 ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcAborting ,
```

```
        Done=>rel1_done ,
        Busy=>rel1_busy ,
        Active=>rel1_active ,
        CommandAborted=>rel1_abort
        );

    setposition1(
        Axis:=1 ,
        Execute:=axis1_power AND setpos1_execute ,
        Position:=200 ,
        Relative:=FALSE ,
        Done=>setpos1_done ,
        Busy=>setpos1_busy ,
        );

    actposition1(
        Axis:=1 ,
        Enable:=actpos1_enable ,
        Valid=>actpos1_vaild ,
        Busy=>actpos1_busy ,
        Position=>actpos,
        );
```

- **Program description**
  - After the program is executed, actpos1_enable becomes TRUE, and the actual position of the axis is read by the actpos1 instruction, and the read actual position is stored in the variable actpos.
  - When setpos1_execute becomes TRUE, the esposition1 instruction is executed, and in the 2nd period, the actual position of the axis changes from the actual position to the position set by the position setting instruction, and the actual position of the axis changes from 0 to 200. After the execution of the setposition1 instruction completes, setpos1_execute becomes FALSE at the same time that setpos1_done becomes FALSE.
  - After the setposition1 instruction is executed, the rel1 instruction is executed. rel1 instruction takes the set position 200 as the reference point and travels another 500 travel units. Phase rel1 instruction is completed, and the axis position is 700.



① Once the program is run, it starts to read the actual position of the axis.

② When setpos1 execute becomes TRUE, the setposition instruction (setpos1 starts to execute, and on the 2nd cycle, the feedback position of the axis changes from 0 to 200.

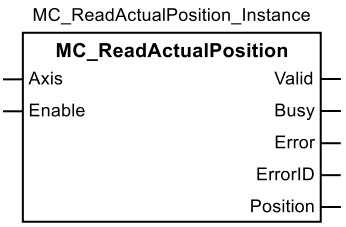
③ The relative instruction (rel1) starts to execute.

④ When the execution of the setposition instruction (setpos1 is completed and execute becomes FALSE, the done bit also becomes FALSE.

⑤ Execution of the relative instruction (rel1) is completed.

4.22 MC_ReadActualPosition (read actual position)

This instruction is used to periodically read the actual position of the specified axis. Library: MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_ReadActualPosition	Read actual position	FB		MC_ReadActualPosition _Instance (Axis :=parameter, Enable :=parameter, Valid => parameter, Busy => parameter, Error=> parameter , ErrorID=> parameter , Position => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
Axis	Axis number	USINT	Depend on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Read the actual position of the specified axis FALSE: Stop reading the actual position of the specified axis.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Valid	Completed	BOOL	TRUE or FALSE	TRUE when read the actual position
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error Code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.
Position	Actual position	LREAL	Positive number, Negative number, 0	The actual position of the specified axis is converted according to the "Parameter setting" of the axis Unit: Travel unit

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Valid	When the value of Position can be read	◆ When Enable becomes FALSE ◆ When Error becomes TRUE
Busy	When Enable becomes TRUE	◆ When Enable becomes FALSE ◆ When Error becomes TRUE
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Enable changes from TRUE to FALSE

■ Function description

● Basic function description

This instruction is used to periodically read the actual position of the specified axis. The specified axis can be a servo axis, a virtual servo axis, a pulse axis, or an encoder axis. The value of Position is updated once per task period when the input variable Enable is TRUE; when Enable is FALSE, the value of Position stops being updated and remains unchanged from the last value.

- **Calculation of Actual Position**

The value of Position is converted by the motor to the position in the mechanism in Travel unit. When the specified axis is a servo axis, the source of this instruction reads the position as the number of pulses that the motor feeds back to the drive; when it is an encoder axis, the source of this instruction reads the position as the number of pulses received by the controller's encoder interface; and when it is a virtual servo axis, the source of this instruction reads the position as the commanded position of the virtual servo axis, which does not need to be converted. When the specified axes are servo axes or encoder axes, the value of Position is obtained through the conversion of the above sources by the mechanism parameters, which can be viewed in the software [Motion Control] -> [Axis Setting] -> [Basic Setting], and the calculation method of the conversion relationship is shown in the figure below, which is calculated by the controller and does not require manual calculation.

$$Actual\ Position = \frac{\begin{matrix}Number\ of\ pulses\ feed\ back\ from\ the\ motor\ to\ the\ drive \\ or\ received\ by\ the\ encoder\end{matrix} * Working\ stroke\ per\ revolution * reducer\ output\ velocity}{Number\ of\ pulses\ per\ revolution\ of\ the\ motor * Input\ velocity\ of\ the\ gearbox}$$

- **Actual position update period**

➢ This instruction reads the actual position of the axis, which is updated once a task period, and the instruction reads the same actual position at different moments within the same task.

➢ The timeliness of this instruction to read the actual position of the axis is not as accurate as the MC_TouchuProbe and MC_TouchuProbeCyc instructions, so if users need to read the accurate position of the axis, please use the MC_TouchuProbe and MC_TouchuProbeCyc instructions.

- **Effect of MC_Home, MC_SetPosition and MC_HomeWithParm on the actual position**

After the MC_Home, MC_SetPosition, and MC_HomeWithParm instructions are executed, the axis commanded position and the actual position follow the position set by these instructions. After these instructions are executed, the position read by this instruction is the same as the actual position in the axis structure (the axis is Finite).

- **Abnormality elimination**

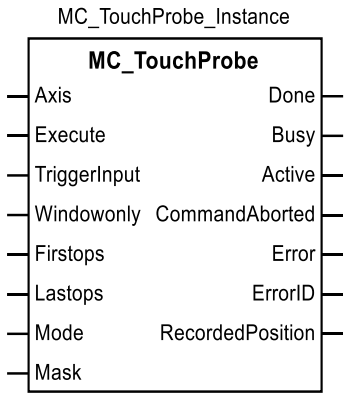
When this instruction is executed, If the value of the instruction input variable is out of valid range, the Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

- **Example program**

For an example program, refer to MC_SetPosition instruction.

4.23 MC_TouchProbe（Recorded position）

The MC_TouchProbe instruction records the position of an axis when a trigger signal occurs. Library : MotionControl

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_TouchProbe	Recorded position	FB		MC_Touchprobe _Instance (Axis:=parameter , Execute :=parameter, TriggerInput:=parameter , Windowonly :=parameter, Firstpos:=parameter , Lastpos:=parameter , Mode:=parameter , Mask:=parameter , Done => parameter, Busy => parameter, Active => parameter, CommandAborted => parameter, Error => parameter, ErrorID => parameter, RecordedPosition=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
TriggerInput	Trigger input condition	MC_Triggerinput	0:mcTriggerinput_I0 1:mcTriggerinput_I1 ... 7: mcTriggerinput_I7 8:mcTriggerinputI_I10 9:mcTriggerinputI_I11 ... 15:mcTriggerinputI_17	0	This input variable is only valid when the encoder axis position is recorded, i.e., when Mode is set to 0 and 1. If this input variable is set to 0, the controller input channel %IX0.0 is used to lock the encoder axis position; if this input variable is set to 1, the controller input channel %IX0.1 is used to lock the encoder axis position
Windowonly	Window only	BOOL	Reserved	Reserved	Reserved
Firstops	First position	LREAL	Reserved	Reserved	Reserved
Lastops	Last position	LREAL	Reserved	Reserved	Reserved
Mode	Mode	INT	0、1、5、6	0	This input variable is used to set the mode for triggering the locking of the axis position. 0: Mode 0*1 The position of the encoder axis is reocorded by the rising edge of the controller's input channel, which input channel is specified by the input variable TriggerInput. RecordedPosition is the position of the controller's encoder interface after receiving the number of pulses converted by the axis. parameters. 1: Mode 1*1 The position of the encoder axis is

					reocorded by the falling edge of the controller's input channel, which input channel is specified by the input variable TriggerInput. RecordedPosition is the position of the controller's encoder interface after receiving the number of pulses converted by the axis 5: Mode 5*2 The position of the servo axis is reocorded by the rising edge of the input channel of the drive, the actual position of the servo axis is determined by the probe function of the drive. RecordedPosition is the motor's actual position converted by the axis parameters. 6: Mode 6*2 The position of the servo axis is reocorded by the falling edge of the input channel of the drive, the actual position of the servo axis is determined by the probe function of the drive. RecordedPosition is the motor's actual position converted by the axis parameters.
Mask	Mask	INT	Reserved	Reserved	Reserved

*1: When multiple MC_TouchProbe instructions use both mode 0 and mode 1, mode 0 and mode 1 cannot be specified as the same input point.

*2: Mode 5 and Mode 6 are only supported by M500S series and M500 series controllers.

■ Output variable

Name	Meaning	Data types	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the trigger signal is executed and the position is recorded
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Active	Controlling	BOOL	TRUE or FALSE	TRUE when the axis is being controlled
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to “Instruction error code”
RecordedPosition	Recorded position	LREAL	Positive number, Negative number, 0	The actual position of the latched axis when the signal is triggered, which is the position of the motor or the position of the controller encoder interface after the number of pulses received and converted by the axis parameters.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the trigger signal is executed and the position is recorded	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE for one period later and then changes to FALSE

Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute changes to FALSE and the instruction is aborted by another instruction, CommandAborted changes to TRUE for one period later and then changes to FALSE
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Execute changes from TRUE to FALSE

■ Function description

● Basic function description

The MC_TouchProbe instruction records the position of an axis when a trigger signal occurs. The specified axis can be the servo axis or encoder axis. When Execute changes to TRUE from FALSE, the instruction is executed according to the set input variable. When the trigger condition of this instruction is triggered, the position of the axis at the trigger moment is output to RecordedPosition, and the execution of this instruction is completed.

● PDO Mapping

The M500S series controllers do not require the user to map data objects. The M500 series controllers require the user to map data objects, which are shown in the following table.

PDO reception (master => slave) (hexadecimal)	Meaning of mapped data	PDO sending (slave => master)	Meaning of mapped data
6040_0（index _subindex）	Control word	6041_0（index _subindex）	Status word
6060_0（index _subindex）	Control mode	6061_0（index _subindex）	Actual mode
60B8_0（index _subindex）	Recorded position	60B9_0（index _subindex）	Recorded position status
		60BA_0（index _subindex）	Rising edge recorded position1
		60BB_0（index _subindex）	Falling edge recorded position1

● Recorded position

➢ The RecordedPosition is recorded when an external trigger condition is established and remains unchanged until the next external trigger condition is established when a new position is recorded again. The value of RecordedPosition is obtained from the data source by converting the axis parameters in travel units. Axis parameters can be viewed in [Motion Control] → [Axis Settings] → [Basic Settings] in the software. When the input variable Axis specifies the axis as servo axis, RecordedPosition is obtained by converting the number of pulses fed back from the motor to the driver through the axis parameters; when the input variable Axis specifies the axis as encoder axis, RecordedPosition is obtained by converting the number of pulses received from the encoder through the axis parameters.

Data sources	Mode	Trigger signal
Number of pulses feedback from the motor to the driver	The value of Mode is set to 5 or 6	The trigger signal is triggered by the rising edge of the servo drive's input channel , which is set by the servo drive.

Number of pulses received by encoder 1*1 The correspondence between the encoder axis and the encoder is specified in the software.	The value of Mode is set to 0 or 1	The trigger signal is triggered by Input variable TriggerInput. M300 and M500S series can be specified as %IX0.2~%IX0.7.
Number of pulses received by encoder 2*1 The correspondence between the encoder axis and the encoder is specified in the software.	The value of Mode is set to 0 or 1	The trigger signal is triggered by Input variable TriggerInput. M300 and M500S series can be specified as %IX0.0~%IX0.1, %IX0.4~ %IX0.7.

*1: The Encoder 1 interface is for input points %IX0.0 and %IX0.1, and the Encoder 2 interface is for input points %IX0.2 and %IX0.3.

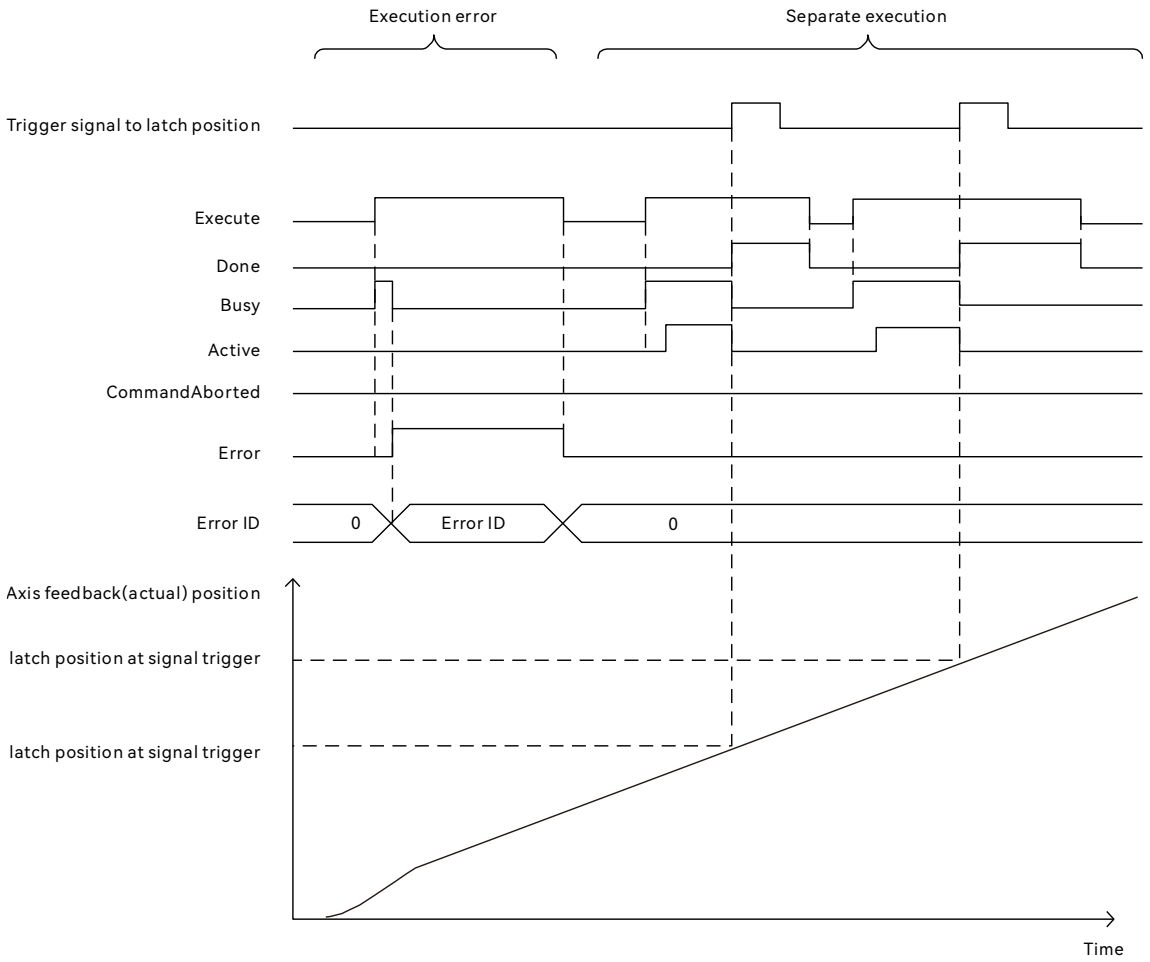
- RecordedPosition is LREAL data type, the position of the data source overstep the limit, the controller will be automatically handled internally, the position will not overstep the limit, will be accumulated. If the drive position type is 32 signed number, when the drive is positively rotated, the position changes to -2147483648 when the drive is positively rotated from 0 to 2147483647 and then positively rotated for 1 pulse, and the position recorded to this instruction is 2147483648.
- When the motor position or the number of pulses received by the coding axis does not exceed the limit, the value of RecordedPosition is calculated as follows:

$$\boldsymbol{RecordedPosition} = \frac{\boldsymbol{Number\ of\ pulses} \times \boldsymbol{Working\ stroke\ per\ revolution} \times \boldsymbol{Reducer\ output\ speed}}{\boldsymbol{Number\ of\ pulses\ per\ motor} \times \boldsymbol{Reducer\ input\ speed}}$$

- **Abnormality elimination**

When this instruction is executed, If the value of the instruction input variable is out of valid range, Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

■ Output variable timing diagram



- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy changes to TRUE. The next period Error changes to TRUE while Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When 0Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Active changes to TRUE in the next period. When an external trigger signal is detected, Done changes to TRUE, and Busy and Active change to FALSE at the same time, the position of the record is stored in the RecordedPosition variable. When Execute changes to FALSE, Done changes to FALSE at the same time.

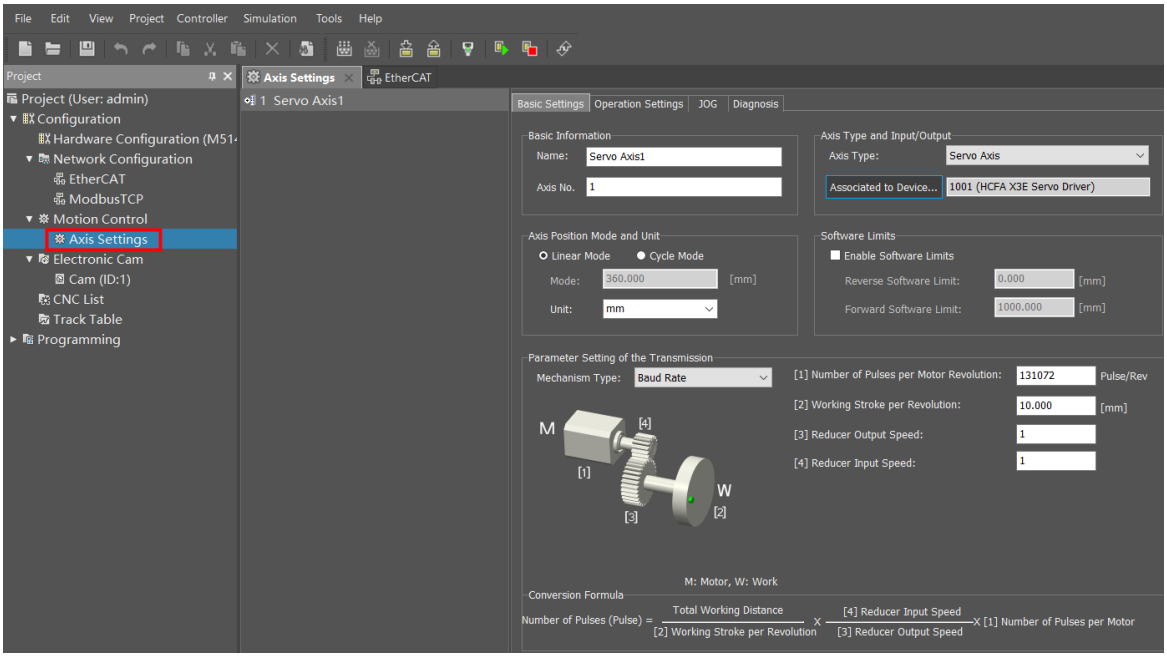
■ Example program

- **Functionality**

  The actual position of the axis is recorded when the MC_MoveVelocity instruction controls the axis operation and the external trigger signal condition is established.

- **Axis Parameter Settings**

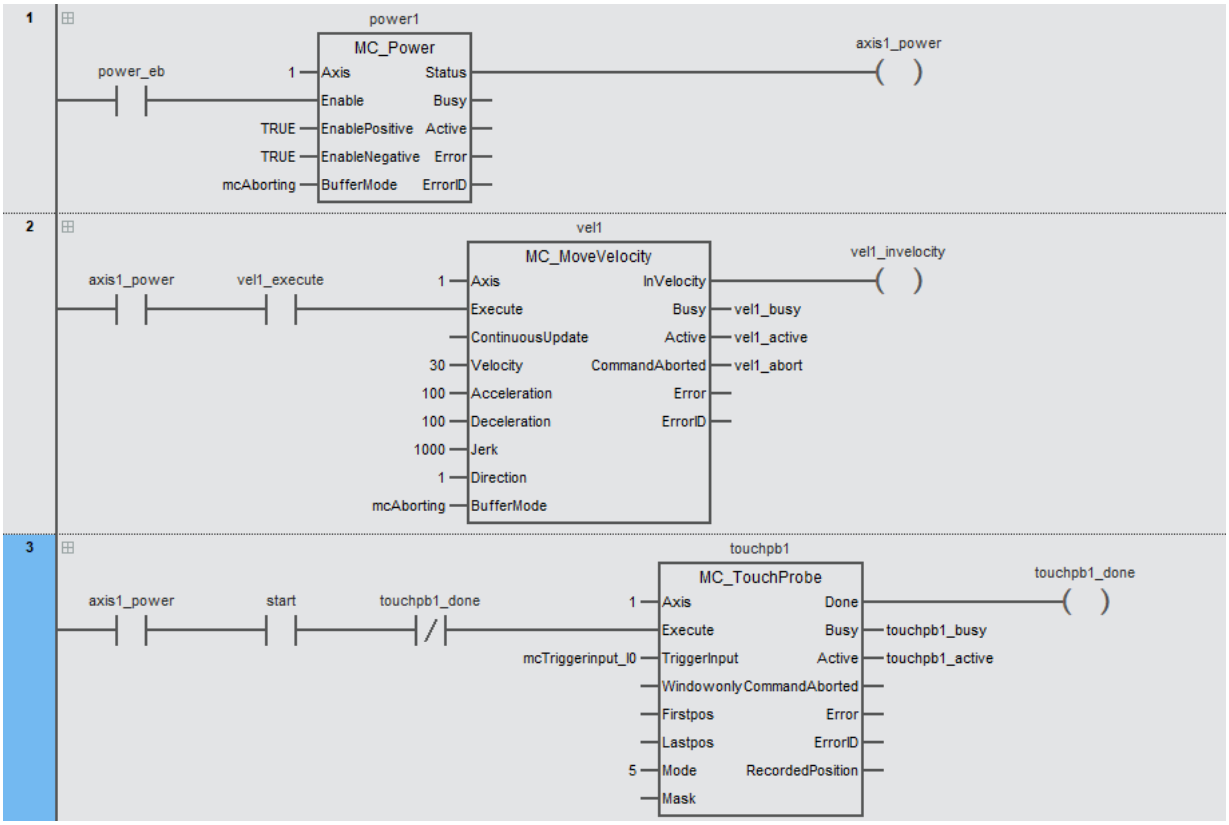
  The axis parameter settings for axis1 are shown below.



- **Variable table**

Category	Name	Assigned to	Data Type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	vel1_execute		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_ invelocity		BOOL		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	touchpb1		MC_TouchProbe		
VAR	touchpb1_done		BOOL		
VAR	touchpb1_busy		BOOL		
VAR	touchpb1_active		BOOL		
VAR	start		BOOL		
VAR	touchpb1_actpos		LREAL		

- LD

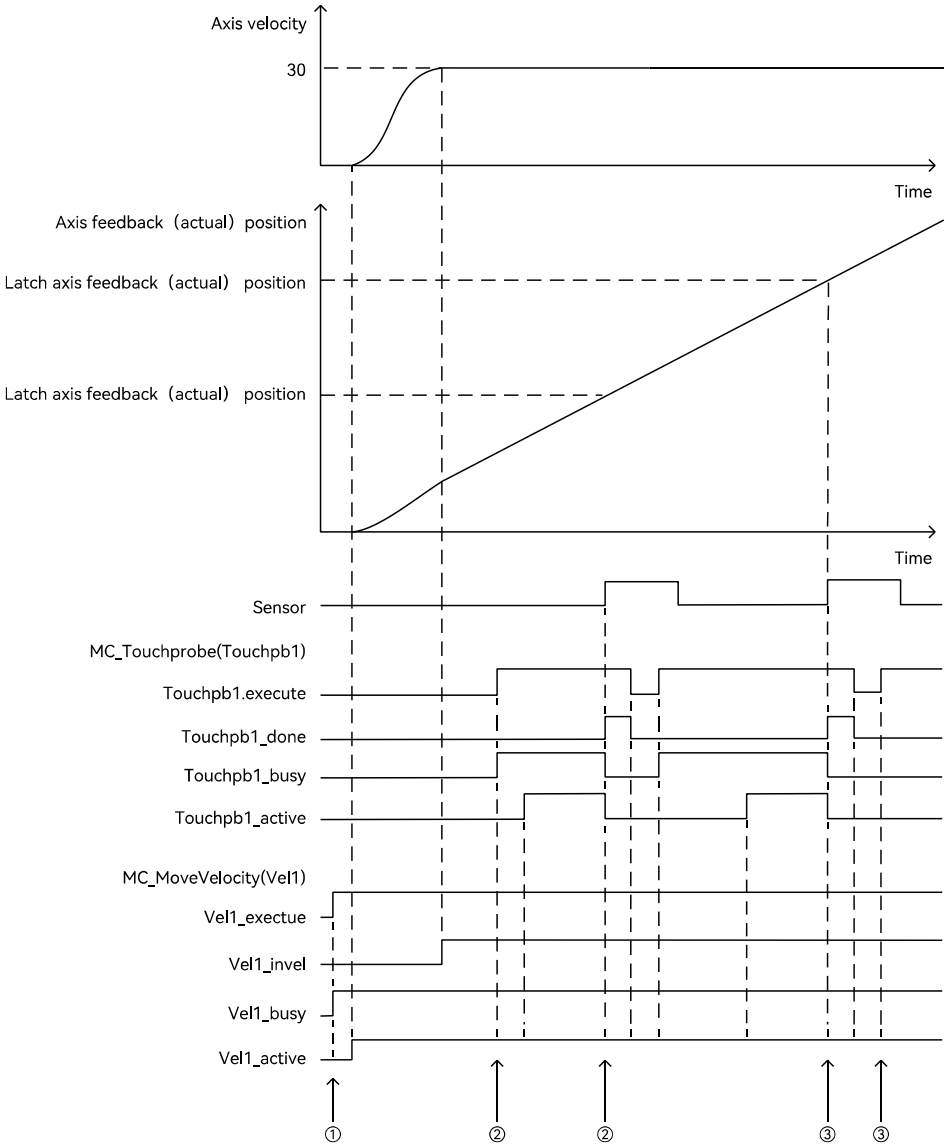


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power);

vel1(
    Axis:=1 ,
    Execute:= axis1_power AND vel1_execute ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    Direction:=1 ,
    BufferMode:=mcAborting ,
    InVelocity=> vel1_invelocity ,
    Busy=> vel1_busy ,
    Active=> vel1_active ,
    CommandAborted=> vel1_abort );

touchpb1(
    Axis:=1 ,
    Execute:=axis1_power AND start and NOT touchpb1_done ,
    TriggerInput:=mcTriggerinput_I0 ,
    Mode:=5
```

```
Done=>touchpb1_done ,
Busy=>touchpb1_busy ,
Active=>touchpb1_active ,
Active=>touchpb1_actpos );
```

- **Program description**
  - When vel1_execute changes to TRUE, the vel1 instruction starts to execute, and in the 2nd period, the vel1 instruction controls the axis operation.
  - When start is TRUE, the touchpb1 instruction is executed, and when the rising edge of the external sensor signal is detected, the actual position of the axis feedback is recorded into the touchpb1_actpos variable, and the execution of this instruction is completed. After the touchpb1 instruction is executed, it will automatically trigger the instruction to be executed again (realized through the program), wait for the rising edge of the external sensor signal, and record the actual position of the axis when the rising edge of the external sensor signal is detected and the execution of the instruction is completed, and so on.



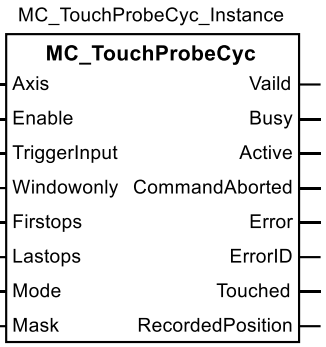
① The movevelocity command (vel1) starts to execute, and control the axis in the second cycle .

② The signal triggers touchprobe instruction (touchpb1) to start execution. After execution, when the rising edge of the external sensor signal is detected, the position of the axis is latching and the execution of this instruction is completed.

③ After touchpb1 execution is completed, the instruction is automatically triggered to be executed again (realized by the program).

4.24 MC_TouchprobeCyc（Recorded position periodically）

The MC_TouchProbeCyc instruction periodically records the position of an axis when a trigger signal occurs.
Library:MotionControl_Part2

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_TouchProbeCyc	Recorded position periodically	FB		MC_TouchprobeCyc _Instance (Axis :=parameter, Enable:=parameter , TriggerInput :=parameter, Windowonly:=parameter , Firstpos:=parameter , Lastpos:=parameter , Mode:=parameter , Mask :=parameter, Vaild=> parameter , Busy=> parameter , Active => parameter, CommandAborted => parameter, Error => parameter, ErrorID => parameter, Touched => parameter, RecordedPosition=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Enable	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
TriggerInput	Trigger input condition	MC_Triggerinput	0:mcTriggerinput_I0 1:mcTriggerinput_I1 ... 7: mcTriggerinput_I7 8:mcTriggerinputI_I10 9:mcTriggerinputI_I11 ... 15:mcTriggerinputI_17	0	This input variable is only valid when the encoder axis position is locked, i.e., when Mode is set to 0 and 1. If this input variable is set to 0, the controller input channel %IX0.0 is used to lock the encoder axis position; if this input variable is set to 1, the controller input channel %IX0.1 is used to lock the encoder axis position
Windowonly	Window only	BOOL	Reserved	Reserved	Reserved
Firstops	First position	LREAL	Reserved	Reserved	Reserved
Lastops	Last position	LREAL	Reserved	Reserved	Reserved
Mode	Mode	INT	0、1、5、6	0	This input variable is used to set the mode in which the recording of the axis position is triggered. By setting the value of this variable, you can select whether the recording of the encoder axis position is triggered by the controller’s input or the recording of the servo axis position is triggered by the driver’s input . 0: Mode 0*1 The position of the encoder axis is recorded by the rising edge of the controller's input channel, which input

					channel is specified by the input variable TriggerInput. RecordedPosition is the position of the controller's encoder interface after receiving the number of pulses converted by the axis. parameters. 1: Mode 1*1 The position of the encoder axis is recorded ed by the falling edge of the controller's input channel, which input channel is specified by the input variable TriggerInput. RecordedPosition is the position of the controller's encoder interface after receiving the number of pulses converted by the axis 5: Mode 5*2 The position of the servo axis is recorded by the rising edge of the input channel of the drive, the actual position of the servo axis is determined by the probe function of the drive. RecordedPosition is the motor's actual position converted by the axis parameters. 6: Mode 6*2 The position of the servo axis is locked by the falling edge of the input channel of the drive, the actual position of the servo axis is determined by the probe function of the drive. RecordedPosition is the motor's actual position converted by the axis parameters.
Mask	Mask	INT	Reserved	Reserved	Reserved

*1: When multiple MC_TouchProbe instructions use both mode 0 and mode 1, mode 0 and mode 1 cannot be specified as the same input point.

*2: Mode 5 and Mode 6 are only supported by M500S series and M500 series controllers.

■ Output variable

Name	Meaning	Data types	Valid range	Description
Vaild	Vaild	BOOL	TRUE or FALSE	TRUE when the instruction executes normally
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Active	Controlling	BOOL	TRUE or FALSE	TRUE when the axis is being controlled
CommandAborted	Command Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to “Instruction error code”
Touched	Record position complete	BOOL	TRUE or FALSE	TRUE when the trigger signal is executed and the position is recorded
RecordedPosition	Recorded position	LREAL	Positive number、Negative number、0	The actual position of the locked axis at the moment of the trigger signal, which is the position of the motor or the position of the controller encoder interface after the number of pulses received and converted by the axis parameters.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Vaild	When Enable changes to TRUE	◆ When Enable changes to FALSE ◆ When Error changes to TRUE
Busy	When Enable changes to TRUE	◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute changes to FALSE and the instruction is interrupted by another instruction, CommandAborted changes to TRUE and then FALSE after one cycle
Error	The value of the input variable is not within the allowed range.	◆ When Error is TRUE and Execute changes from TRUE to FALSE

■ Function description

● Basic function description

The MC_TouchProbeCyc instruction records the position of an axis when a trigger signal occurs, the specified axis can be servo axis or encoder axis. When Enable is TRUE, when the external trigger condition is established, the actual position of the specified axis will be output to RecordedPosition, and the actual position will be recorded once when the external trigger condition is triggered. The difference between this instruction and MC_Touchuprobe instruction is that this instruction is not cyclic, when Enable is TRUE, after recording the position, it will prepare for the next recording of the position, and does not need the program to process it; when MC_Touchuprobe instruction is executed, after recording the position, you need to make the instruction executed again through the program. After the instruction is executed again and the specified trigger condition is established, the position of the next axis can be recorded.

● Recorded Position

The RecordedPosition is recorded when an external trigger condition is established and remains unchanged until the next external trigger condition is established when a new position is recorded again. The value of RecordedPosition is obtained from the data source by converting the axis parameters in travel units. Axis parameters can be viewed in [Motion Control] → [Axis Settings] → [Basic Settings] in the software. When the input variable Axis specifies the axis as servo axis, RecordedPosition is obtained by converting the number of pulses fed back from the motor to the driver through the axis parameters; when the input variable Axis specifies the axis as encoder axis, RecordedPosition is obtained by converting the number of pulses received from the encoder through the axis parameters.

Data sources	Setting Mode	Trigger signal
Number of pulses feedback from the motor to the driver	The value of Mode is set to 5 or 6	The trigger signal is triggered by the rising edge of the servo drive's input channel , which is set by the servo drive.
Number of pulses received by encoder 1*1 The correspondence between the encoder axis and the encoder is specified in the software.	The value of Mode is set to 0 or 1	The trigger signal is triggered by Input variable TriggerInput. M300 and M500S series can be specified as %IX0.2~%IX0.7.
Number of pulses received by encoder 2*1 The correspondence between the encoder axis and the encoder is specified in the software.	The value of Mode is set to 0 or 1	The trigger signal is triggered by Input variable TriggerInput. M300 and M500S series can be specified as %IX0.0~%IX0.1, %IX0.4~ %IX0.7.

*1: M500S series encoder 1 interface as input channels %IX0.0 and %IX0.1, and Encoder 2 interface as input channels %IX0.2 and %IX0.3.

- **PDO Mapping**

The M500S series controllers do not require the user to map data objects. The M500 series controllers require the user to map data objects, which are shown in the following table.

PDO reception (master => slave) (hexadecimal)	Meaning of mapped data	PDO sending (slave => master)	Meaning of mapped data
6040_0（index _subindex）	Control Word	6041_0（index _subindex）	Status Word
6060_0（index _subindex）	Control Mode	6061_0（index _subindex）	Actual mode
60B8_0（index _subindex）	Recorded position	60B9_0（index _subindex）	Recorded position status
		60BA_0（index _subindex）	Rising edge recorded position1
		60BB_0（index _subindex）	Falling edge recorded position1

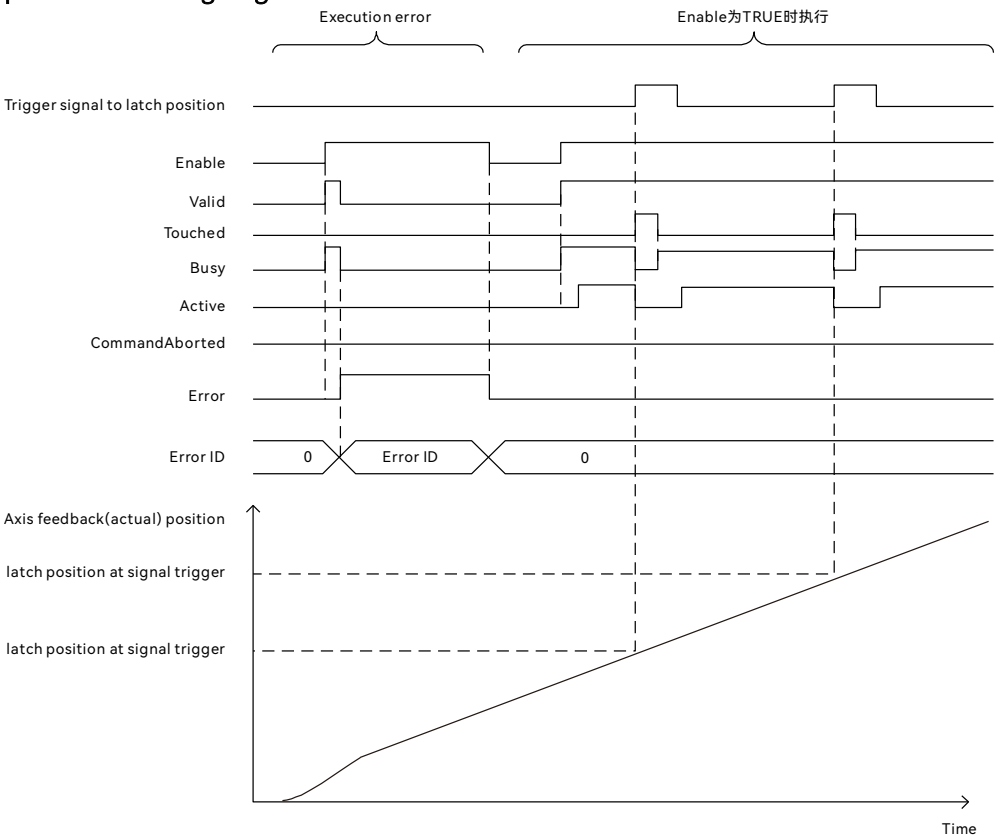
- RecordedPosition is LREAL data type, the position of the data source overstep the limit, the controller will be automatically handled internally, the position will not overstep the limit, will be accumulated. If the drive position type is 32 signed number, when the drive is positively rotated, the position changes to -2147483648 when the drive is positively rotated from 0 to 2147483647 and then positively rotated for 1 pulse, and the position recorded to this instruction is 2147483648.
- When the motor position or the number of pulses received by the coding axis does not exceed the limit, the value of RecordedPosition is calculated as follows:

$$RecordedPosition = \frac{Number\ of\ pulses \times Working\ stroke\ per\ revolution \times Reducer\ output\ speed}{Number\ of\ pulses\ per\ motor \times Reducer\ input\ speed}$$

Abnormality elimination

When this instruction is executed, If the value of the instruction input variable is out of valid range, Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

■ **Output variable timing diagram**



- **Execution error**

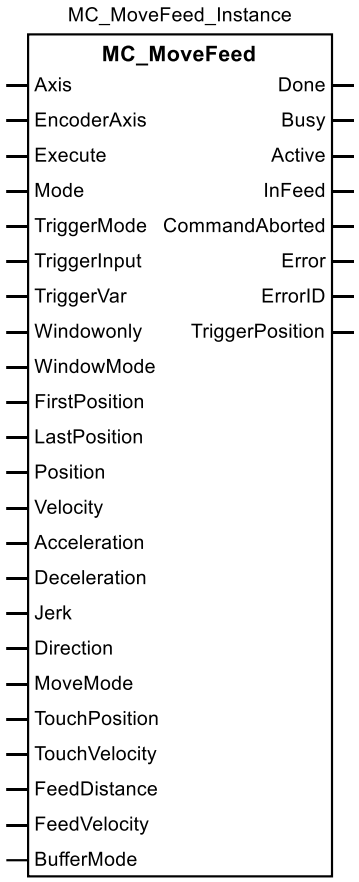
  When the value of the input variable is not within the allowable range, Enable changes from FALSE to TRUE while Valid and Busy changes to TRUE. The next period Error changes to TRUE while Valid and Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Enable changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Separate execution**

  When Enable changes from FALSE to TRUE, Valid and Busy changes to TRUE at the same time, and Active changes to TRUE in the next period. When an external trigger signal is detected, Touched changes to TRUE for one task period, and Busy and Active change to FALSE at the same time. the position of the record is stored in the RecordedPosition variable. Touched changes to TRUE for one task period and then automatically changes to FALSE, Busy changes to TRUE at the same time, and Active changes to TRUE after one task period.

4.25 MC_MoveFeed（Interrupt feeding）

This instruction is activated to control the specified axis for displacement or velocity movement. After this instruction controls the axis to move beyond the specified relative position, the axis movement velocity is switched from high speed to low speed, and the low speed movement is used to search for an external interrupt signal. After detecting the external interrupt input, the actual position of the axis is grasped with the interrupt input signal as the reference point, and then moves the specified relative distance.Library:MotionControl_Part2

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_MoveFeed	Move the specified distance after an external interrupt	FB		MC_MoveFeed_Instance(Axis:=parameter, EncoderAxis:=parameter , Execute :=parameter, Mode:=parameter , TriggerMode:=parameter , TriggerInput :=parameter, TriggerVar:=parameter , Windowonly :=parameter, WindowMode :=parameter, FirstPosition :=parameter, LastPosition :=parameter, Position :=parameter, Velocity:=parameter , Acceleration :=parameter, Deceleration :=parameter, Jerk :=parameter, Direction:=parameter , MoveMode:=parameter , TouchPosition:=parameter , TouchVelocity :=parameter, FeedDistance:=parameter , FeedVelocity:=parameter , BufferMode:=parameter , Done => parameter, Busy=> parameter , Active=> parameter , InFeed => parameter, CommandAborted => parameter, Error=> parameter , ErrorID => parameter, TriggerPosition => parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
EncoderAxis	Encoder axis number	USINT	Depends on model	Required field	Specify the axis number of the encoder axis
Exexute	Execute	BOOL	TRUE or FALSE	FALSE	When the rising edge of this parameter is detected, relative displacement motion or velocity motion is performed according to the value of MoveMode
Mode	Mode	INT	0、1	0	Specify the mode of the input signal 0: After an external interrupt is input to record the axis position, the axis is then moved by the specified relative distance. 1.After detecting the rising edge of the input variable TriggerVar, the axis then

					moves the specified relative distance
TriggerMode	Trigger mode	INT	0、1、5、6	0	This input variable is used to set the mode for triggering the recording of the axis position. 0: Mode 0*1 The position of the encoder axis is recorded by the rising edge of the controller's input channel, which input channel is specified by the input variable TriggerInput. RecordedPosition is the position of the controller's encoder interface after receiving the number of pulses converted by the axis. parameters. 1: Mode 1*1 The position of the encoder axis is recorded by the falling edge of the controller's input channel, which input channel is specified by the input variable TriggerInput. RecordedPosition is the position of the controller's encoder interface after receiving the number of pulses converted by the axis 5: Mode 5*2 The position of the servo axis is locked by the rising edge of the input channel of the drive, the actual position of the servo axis is determined by the probe function of the drive. RecordedPosition is the motor's actual position converted by the axis parameters. 6: Mode 6*2 The position of the servo axis is recorded by the falling edge of the input channel of the drive, the actual position of the servo axis is determined by the probe function of the drive. RecordedPosition is the motor's actual position converted by the axis parameters.
TriggerInput	Trigger input	MC_Triggerinput	0:mcTriggerinput_I0 1:mcTriggerinput_I1 ... 7: mcTriggerinput_I7 8:mcTriggerinputI_I10 9:mcTriggerinputI_I11 ... 15:mcTriggerinputI_17	0	This input variable is only valid when the encoder axis position is recorded, i.e., when Mode is set to 0 and 1. If this input variable is set to 0, the controller input channel %IX0.0 is used to record the encoder axis position; if this input variable is set to 1, the controller input channel %IX0.1 is used to lock the encoder axis position
TriggerVar	Trigger variable	BOOL	TRUE or FALSE	FALSE	When Mode is set to 1, the axis moves the specified relative distance after the rising edge of this input variable is detected.

Windwonly	Window only	BOOL	TRUE or FALSE	FALSE	Specify whether to enable or disable the window FALSE: Invalid TRUE: Valid
WindowMode	Window mode	BOOL	TRUE or FALSE	FALSE	Specify the mode of the wWndowonly FALSE: Absolute mode TRUE: Relative mode
FirstPosition	First position	LREAL	Positive number、Negative number、0	0	Specify the position where latching is enabled Linear mode: First position of Window Cycle mode: 0≤FirstPosition<Cycle
LastPosition	Last position	LREAL	Positive number、Negative number、0	0	Specify the position where latching is disabled Linear mode: Last position of Window Cycle mode: 0≤LastPosition<Cycle
Position	Target position	LREAL	Positive number、Negative number、0	0	The meaning of the target position is determined by the value of the input variable MoveMode. When the value of MoveMode is 1, the value of Position is the relative position, using the command position when Execute changed from FALSE to TRUE as the reference point. When the value of MoveMode is 0, the value of Position is the absolute position. Linear mode: Limitless Cycle mode: 0≤Position<Cycle The axis position mode can be set in "Axis Settings" → "Basic Settings" in the software. When the value of MoveMode is 2, the value of Position is meaningless.
Velocity	Target velocity	LREAL	Positive number	Required field	Specify the target velocity *2 (Unit: Travel units/s)*3
Acceleration	Acceleration rate	LREAL	Positive number	Required field	Specify the acceleration rate *2 (Unit: Travel units /s2)*3
Deceleration	Deceleration rate	LREAL	Positive number	Required field	Specify the deceleration rate *2 (Unit: Travel units /s2)*3
Jerk	Jerk	LREAL	Positive number	Required field	Specify the jerk*2 (Unit: Travel units /s3)*3
Direction	Direction	MC_Direction	1: mcPositiveDirection 2: mcShortestWay 3: mcNegativeDirection 4: mcCurrentDirection	1	Specify the direction of rotation 1: Positive direction 2: Shortest way 3: Negative direction 4: Current direction, If the axis is at standstill, it moves in the positive direction
MoveMode	Move mode	MC_MoveMode	0: mcAbsolute 1: mcRelative 2: mcVelocity		Select the travel method. 0: Absolute positioning 1: Relative positioning 2: Velocity control

TouchPosition	Switch to relative position at low velocity	LREAL	Positive number、Negative number、0	0	This parameter calculates the relative position from the start of instruction execution. When the relative position exceeds the position set by this parameter, the target velocity of the axis changes to the velocity set by TouchVelocity from the velocity set by Velocity. If the position is a relative position, the initial reference position is the commanded position of the axis at the time the instruction is executed.
TouchVelocity	Target velocity after exceeding TouchPosition	LREAL	Positive number、Negative number、0	0	When the actual position exceeds the position set by the TouchPosition parameter, the target velocity of the axis changes to the velocity set by TouchVelocity from the velocity set by Velocity.
FeedDistance	The relative distance that needs to be moved after detecting an external interrupt	LREAL	Positive number、Negative number、0	0	Relative distance to be moved after external interrupt input is detected*2 (Unit: Travel units/s)*3
FeedVelocity	Target speed during relative positioning after detecting external interrupts	LREAL	Positive number	Required field	Specify the target velocity when moving relative to the distance after detecting an external interrupt input *2 (Unit: Travel units/s)*3
BufferMode	Buffer mode	MC_Buffer_Mode	0: mcAborting 1: mcBuffered 2: mcBlendingLow 3: mcBlendingPrevious 4: mcBlendingNext 5: mcBlendingHigh	0	Setting the bufferMode between two instructions*4 0: Aborting 1: Buffered 2: BlendingLow 3: BlendingPrevious 4: BlendingNext 5: BlendingHigh

*1: When multiple MC_TouchProbe instructions use both mode 0 and mode 1, mode 0 and mode 1 cannot be specified as the same input point.

*2: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "*Parameter description of motion control instructions*".

*3:For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

*4:For details of BufferMode, please refer to "*Buffer mode during multi-starting of the same axis*".

*5:Mode 5 and Mode 6 are only supported by M500S series and M500 series controllers.

■ Output variable

Name	Meaning	Data types	Valid range	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when relative positioning completion triggered by an external interrupt; TRUE when positioning completion by MoveMode
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Active	Controlling	BOOL	TRUE or FALSE	TRUE when the axis is being controlled
InFeed	Feeding	BOOL	TRUE or FALSE	TRUE when specified signal be interrupted

CommandAborted	Command Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to "*Instruction error code*"
TriggerPosition	Recorded trigger position	LREAL	Positive number、Negative number、0	Record the position of the axis when the signal is triggered

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When relative positioning completion triggered by an external interrupt; When positioning completion by MoveMode	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then FALSE one period later
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
Active	When the instruction is started	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE ◆ When CommandAborted changes from FALSE to TRUE
InFeed	When the specified interrupt signal is triggered	◆ When InFeed changes to TRUE and Execute changes from TRUE to FALSE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute changes to FALSE and the instruction is interrupted by another instruction, CommandAborted changes to TRUE and then FALSE one period later
Error	The value of the input variable is not within the allowed range, the execution conditions of the instruction are not satisfied, and an error is encountered during the execution of the instruction	◆ When Error is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction has been executed and Execute changes to FALSE and an exception is encountered during the execution of the instruction, Error changes to TRUE and then FALSE one period later

■ Function description

● Basic function description

- After the instruction is started, the specified axis is controlled for displacement or velocity movement. When the axis position exceeds the relative position set by TouchPosition, the axis movement velocity is switched from high speed to low speed to search for an external interrupt signal at low speed. After detecting the external interrupt input, the actual position of the axis is detected with the interrupt input signal as the reference point and then moved the specified relative distance.
- When Execute is changed from FALSE to TRUE, the axis moves in one of Absolute positioning , Relative positioning or Velocity control, depending on the setting of MoveMode. If MoveMode is set to 0, Position is set to absolute position; if MoveMode is set to 1, Position is set to relative position; if MoveMode is set to 2, Position is invalid, and the instruction controls the axis to start moving with Velocity as the target velocity. This instruction controls the target velocity of the axis to switch from Velocity to TouchVelocity when the axis position exceeds the relative position set by TouchPosition (TouchPosition is the relative position with the reference point of the axis position when the instruction Execute changes from FALSE to TRUE), the purpose of the velocity switching is to search for the interrupt signal when the velocity is low. The purpose of the speed switching is to look for the interrupt signal at low speed, so that it stops after encountering the interrupt signal, and the stopping

position is more accurate. This instruction looks for an external interrupt signal after switching to the target velocity set by TouchVelocity. After detecting the external interrupt signal, use the actual position of the gripping axis of the interrupt input signal as the reference point, use the set value of FeedVelocity as the target velocity, and perform the relative displacement movement with the distance specified by FeedDistance. if FeedDistance is a positive number, the axis move the relative distance in the positive direction, and if FeedDistance is a negative number, the axis move the relative distance in the negative direction. If there is no external interrupt signal, when MoveMode selects Absolute Positioning or Relative Positioning, the axis stops when it reaches the target position set by Position; When MoveMode is selected for velocity control, the set value of Velocity is used as the target velocity to control the axis movement first. When the position of the axis controlled by this instruction exceeds the relative position set by TouchPosition, the axis moves with the velocity set by TouchVelocity as the target velocity.

- **PDO Mapping**

The M500S series controllers do not require the user to map data objects. The M500 series controllers require the user to map data objects, which are shown in the following table.

PDO reception (master => slave) (hexadecimal)	Meaning of mapped data	PDO sending (slave => master)	Meaning of mapped data
6040_0（index _subindex）	Control Word	6041_0（index _subindex）	Status Word
6060_0（index _subindex）	Control Mode	6061_0（index _subindex）	Actual mode
60B8_0（index _subindex）	Recorded position	60B9_0（index _subindex）	Recorded position status
		60BA_0（index _subindex）	Rising edge recorded position1
		60BB_0（index _subindex）	Falling edge recorded position1

- **TriggerMode**

➢ When Mode is 0, it is the external interrupt trigger positioning mode, the external interrupt signal triggers and records the axis position, then moves the relative distance specified by FeedDistance. The input variable TriggerMode specifies that the position of the servo axis is recorded via the input channel of the drive or the position of the encoder axis is recorded via the input channel of the controller.

(a) When TriggerMode is 0 or 1, the position of the encoder axis is recorded through the controller input channel and then the servo axis is positioned. The relationship between encoder axis and servo axis is established through hardware wiring (pulse output of the driver is connected to the encoder input of the controller), and this mode can be used to indirectly get the servo axis position by getting the position of the encoder axis. The parameter EncoderAxis sets the axis number of the encoder axis.

The input variable TriggerInput is used to set the input signal for recording the position of the encoded axis, this variable is only valid if TriggerMode is 0 or 1. If this variable is set to 0, the controller input channel %IX0.0 is used to record the position of the encoder axis; if it is set to 1, the controller input channel %IX0.1 is used to record the position of the encoder axis.

(b) When TriggerMode is 5 or 6, the servo axis position is recorded through the trigger drive input channel.

Mode5: The position of the servo axis is locked by the rising edge of the input channel of the drive. RecordedPosition is the motor's actual position converted by the axis parameters.

Mode6: The position of the servo axis is locked by the falling edge of the input channel of the drive, RecordedPosition is the motor's actual position converted by the axis parameters

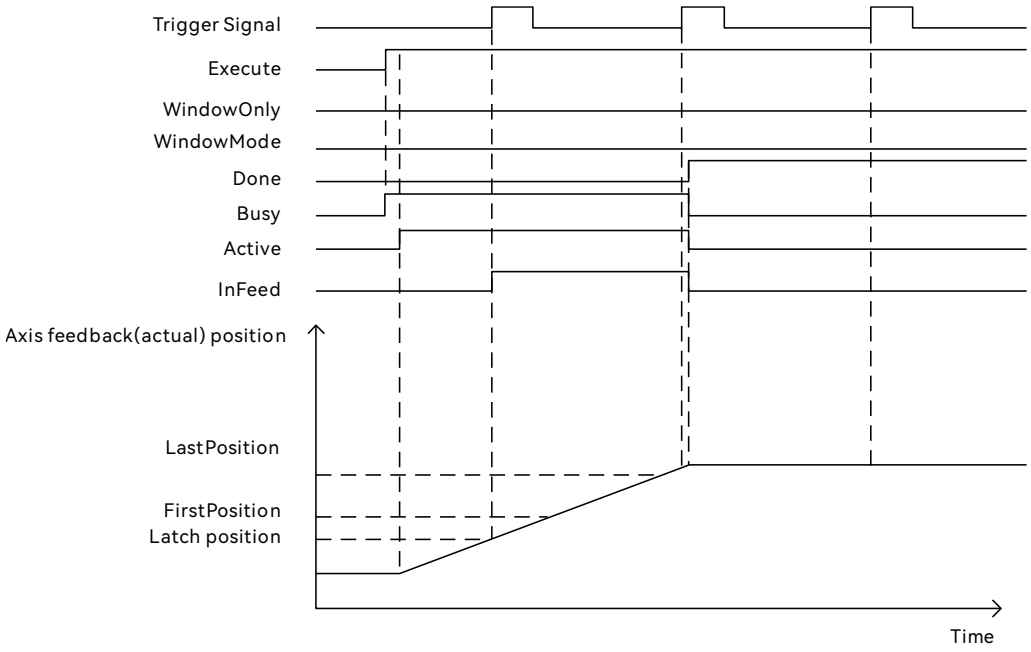
➢ When Mode is 1, it is the variable trigger positioning mode, the controller detects the rising edge of the input variable TriggerVar, and then moves the relative distance specified by FeedDistance. The parameter TriggerVar sets the trigger variable, which can be an internal variable of the controller or an input channel of the controller as an external trigger signal. Triggering using the rising edge of this variable is done by detecting the rising edge of the TriggerVar variable by means of a scanning program, not by means of an interrupt, and because of the delay in IO refreshing, the positioning accuracy with Mode set to 1 is lower than Mode set to 0.

NOTE: When using variable-triggered positioning modes, the trigger variable must be within the motion event task in case the rising edge of the TriggerVar is not detected.

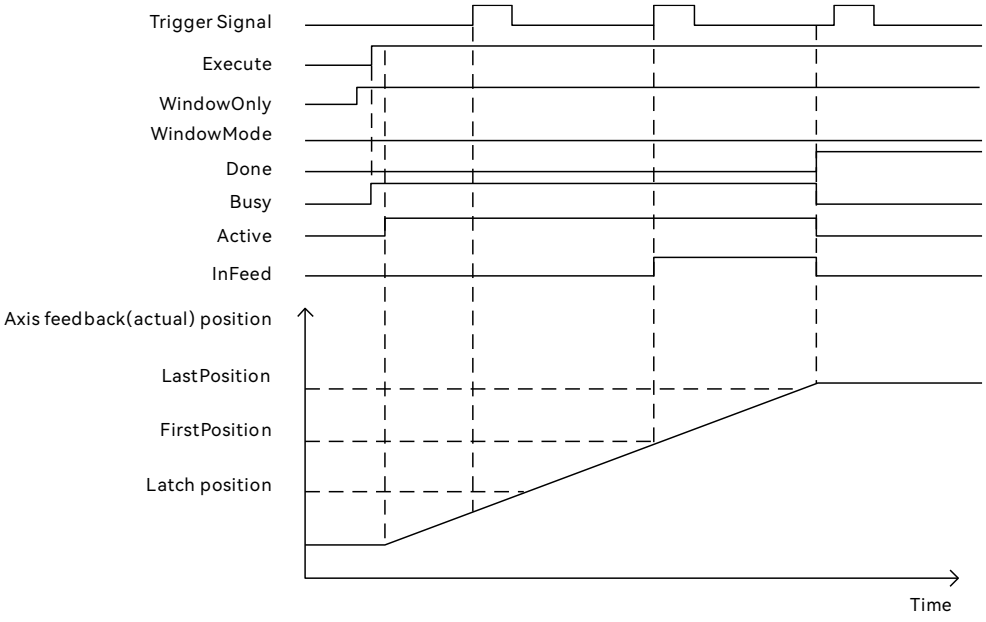
- **Window function**

➢ Windownly sets whether the window function is valid or not

1) When Windownly is FALSE, the window function is invalid, and the relative positioning action can be triggered by the axis detecting an external interrupt signal at any position.



2) When Windownly is TRUE, the window function is valid and the position of the axis can trigger the relative positioning action only when an external interrupt signal is detected within the valid range of the window.



- **WindowMode Sets the window mode**

  1）When WindowMode is FALSE, the window mode is absolute mode.

  When the window mode is absolute mode, if the axis controlled by the instruction is linear mode, the effective range of the window is FirstPosition ⩽ axis position ⩽ LastPosition. The position of an axis can trigger a relative positioning action only if an external interrupt signal is detected within the valid range of the window.

  If the axis controlled by the instruction is a cyclic mode, the valid range of the window is the interval from FirstPositon positive rotation to LastPosition. The instruction reports an error when FirstPositon ⩾ the mode of the axis, or LastPosition ⩾ the mode of the axis.

  2）When WindowMode is TRUE, the window mode is relative.

  When the window mode is relative mode, FirstPosition and LastPosition are relative positions, and the reference point of these two positions is the axis position when this instruction Execute changes to TRUE from FALSE. If the axis position at the time of execution of this instruction is 100 and FirstPosition and LastPosition are set to 500 and 800 respectively, the relative positioning action can be triggered only when the external interrupt signal is detected to be valid when the axis position is between 600 and 900.

  When the window mode is relative mode, if the axis controlled by the instruction is linear mode, the external signal can be triggered to lock the effective range of the axis position: (Axis position when this instruction is executed + FirstPosition) ⩽ axis position ⩽ (Axis position when this instruction is executed + LastPosition).

  When the window mode is relative mode, the window position is calculated as follows if the axis controlled by the instruction is cyclic mode:

  FirstPosition=MODABS(Axis position when this instruction is executed +FirstPosition, Modulo value);
  LastPosition =MODABS(Axis position when this instruction is executed +FirstPosition, Modulo value);
  That is, the newly computed position takes the remainder of the module value.

  Note:

  If the axis controlled by the instruction is a linear axis, the instruction reports an error when FirstPosition ⩾ LastPosition.

  When the axis controlled by the instruction is a cycle axis, the instruction reports an error when FirstPositon ⩾ the mode of the axis or LastPosition ⩾ the mode of the axis.

- **Re-execute the instruction**

  When the execution of the instruction is completed (when Done changes to TRUE) and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed to completion.

- **Execution during Execution of Other Instructions**

  The program can switch or cache to this instruction if it is started while other motion instructions are executing. How this instruction and other motion instructions are relayed is determined by the value of the instruction input variable BufferMode, and the value that can be set for the instruction input variable BufferMode is related to the motion instruction being executed.

- **Execution of Other Instructions during Instruction Execution**

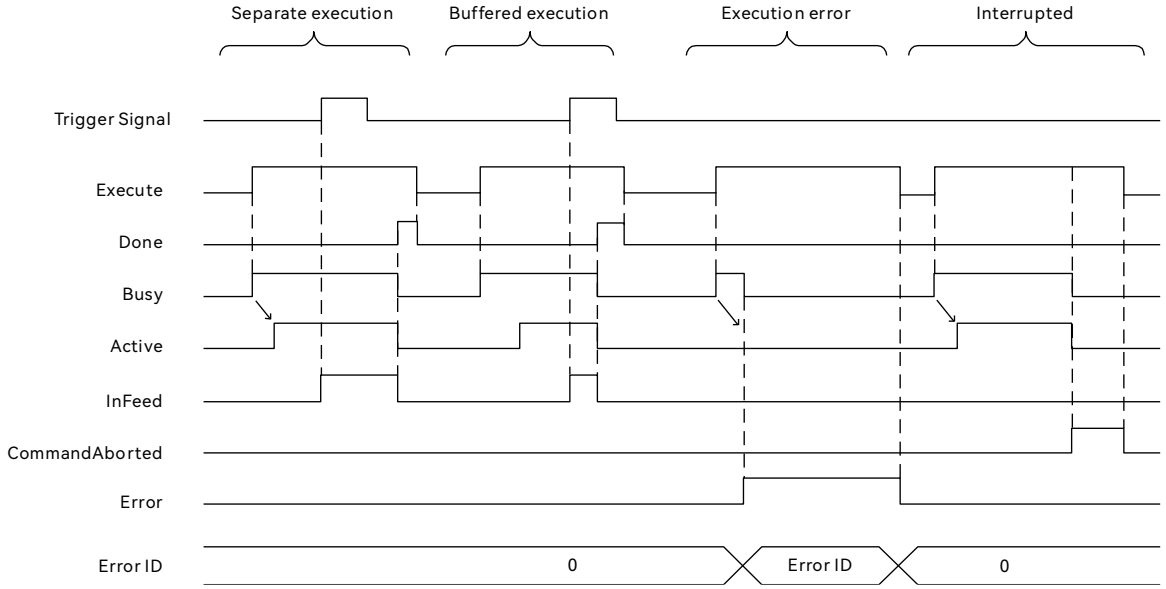
  It is not recommended to execute other instructions when this instruction is executed. If other instructions choose to interrupt this instruction, there is no point in executing this instruction, and if other

instructions and this instruction are intersected, the external trigger signal of this instruction will interrupt the instruction that is executed afterward when it is triggered.

- **Abnormality elimination**

  When this instruction is executed, if the input variable is illegal, Error will change to TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g., an axis error), the instruction will also report an error and the axis will stop immediately.

■ Output variable timing diagram



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Active changes to TRUE in the next period. InFeed changes to TRUE when an external interrupt triggers the recording of the position, When the relative positioning completion is triggered by an external interrupt or the positioning completion specified by MoveMode when there is no external interrupt, Done changes to TRUE, and Busy、 Active、InFeed change to FALSE at the same time. When Execute changes to FALSE, Done changes to FALSE at the same time.

- **Blending**

  Other instructions control the axis when the instruction is executed (the value of BufferMode is not mcAborting). When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Active changes to TRUE at the completion of the previous instruction.

- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy changes to TRUE. The next period Error changes to TRUE while Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Execution interruption**

  When the execution of this instruction is interrupted by another instruction, CommandAborted changes to TRUE, Busy and Active change to FALSE at the same time; when Execute changes to FALSE, CommandAborted changes to FALSE at the same time.

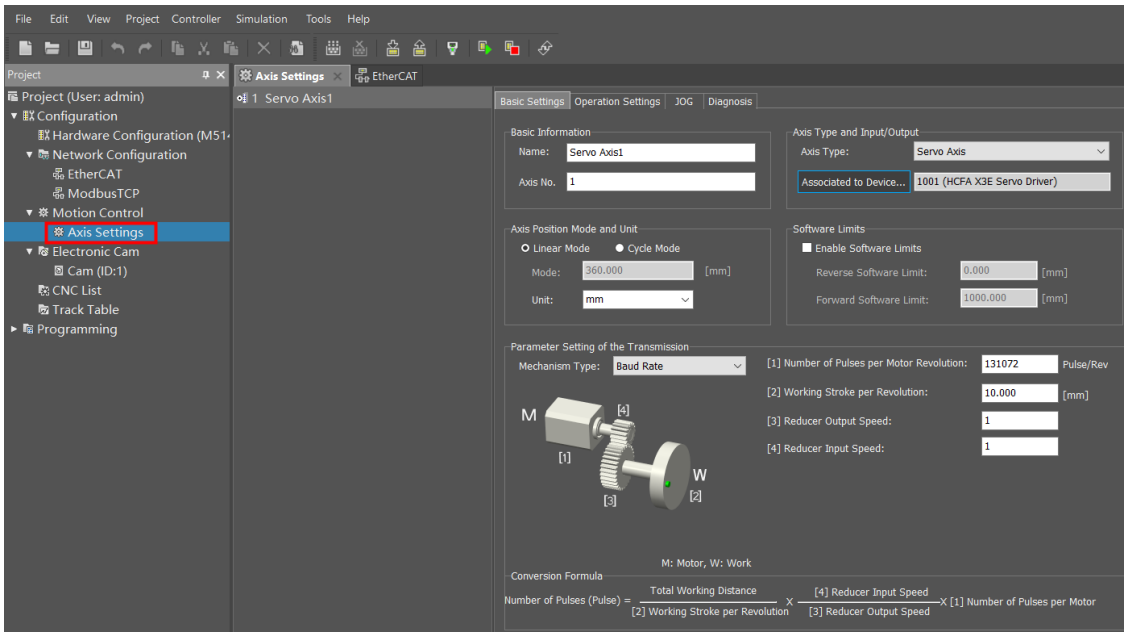
■ Example program

● Functionality

The specified axis first performs absolute value positioning, this instruction controls the axis position over the relative position 500 set by TouchPosition, the axis movement velocity is switched from high speed to low speed, and the external interrupt signal is searched for at low speed (the external interrupt signal is connected to the input channel of the servo driver) at low speed, after the external interrupt input is detected, the actual position of the axis is grasped as the reference point by the interrupt input signal. The axis then moves the relative distance specified by FeedDistance and stops after moving the specified relative distance.

● Axis Parameter Settings

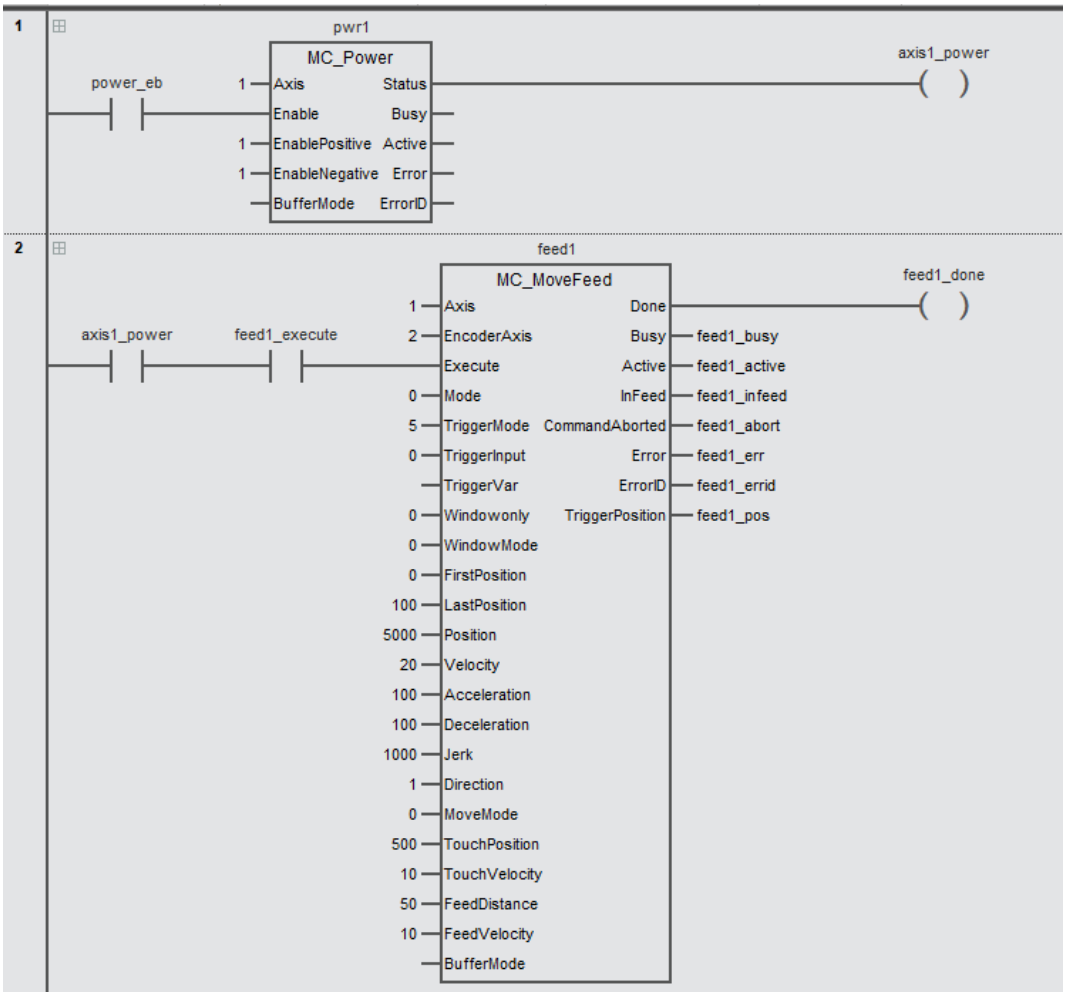
The axis parameter settings for axis1 are shown below.



● Variable table

Category	Name	Assigned to	Data Type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	feed1_execute		BOOL		
VAR	feed1		MC_MoveFeed		
VAR	feed1_done		BOOL		
VAR	feed1_busy		BOOL		
VAR	feed1_active		BOOL		
VAR	feed1_infeed		BOOL		
VAR	feed1_abort		BOOL		
VAR	feed1_err		BOOL		
VAR	feed1_errid		WORD		
VAR	feed1_pos		LREAL		

- LD

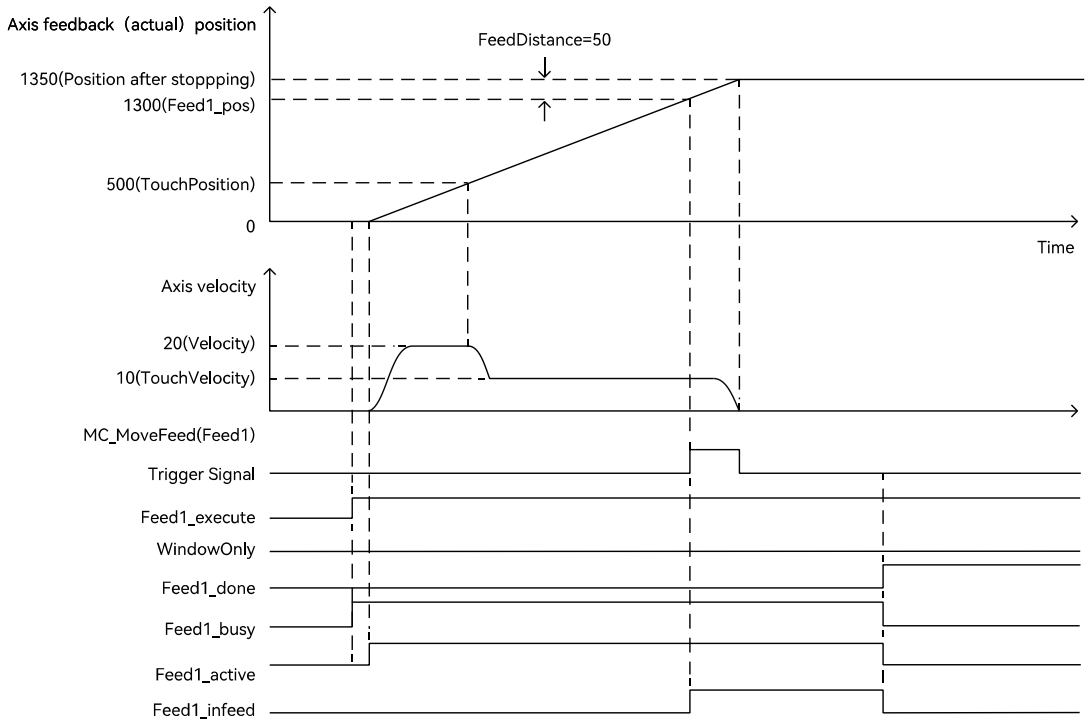


- ST

```
power1 (
        Axis:=1 ,
        Enable:=power_eb ,
        EnablePositive:=TRUE ,
        EnableNegative:=TRUE ,
        BufferMode:=mcAborting ,
        Status=>axis1_power);

feed1(Axis:= 1,
        EncoderAxis:=2 ,
        Execute:= axis1_power AND feed1_execute,
        Mode:=0 ,
        TriggerMode:=5 ,
        TriggerInput:=0 ,
        Windowonly:=0 ,
        WindowMode:=0 ,
        FirstPosition:=0 ,
        LastPosition:=0 ,
        Position:=5000 ,
        Velocity:=20 ,
        Acceleration:=100 ,
        Deceleration:=100,
        Jerk:= 100,
        Direction:= 1,
```

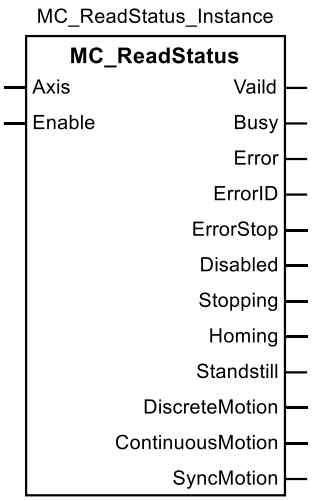
```
MoveMode:= 0,
TouchPosition:= 500,
TouchVelocity:=10 ,
FeedDistance:= 50,
FeedVelocity:=10 ,
BufferMode:= mcAborting ,
Done=> feed1_done ,
Busy=> feed1_busy ,
Active=> feed1_active,
InFeed=> feed1_infeed ,
CommandAborted=> feed1_abort ,
Error=> feed1_err ,
ErrorID=> feed1_errid,
TriggerPosition=> feed1_pos);
```

- **Program description**
  - After the axis is enabled, the current position of the axis is 0. Mode is set to 0 (External interrupt input to record axis position), MoveMode is set to 0 (Absolute positioning), and when feed1_execute changes to TRUE from FALSE, the axis starts to move according to the Position, Velocity, Acceleration. Deceleration, Jerk set parameters to move, and calculate the relative position of the axis traveled by this instruction control.
  - This instruction takes the commanded position of the axis at the start of execution as the reference position, When this instruction controls the axis position to exceed the relative position of 500 set by TouchPosition, the target velocity of the axis decreases from 20 to 10 (the velocity set by Velocity changes to the velocity set by TouchVelocity) and looks for an external interrupt signal in the low-speed state. When the rising edge of the external interrupt signal is detected, the position of the specified axis is recorded, feed1_infeed changes to TRUE, feed1_pos outputs the recorded axis position, and the axes are then relatively positioned according to FeedDistance and FeedVelocity. After positioning is completed, the axis stops moving, feed1_Done changes to TRUE, and feed1_busy, feed1_active, and feed1_infeed all change to FALSE at the same time. After feed1_execute changes to FALSE, feed1_done changes to FALSE in the same period, and feed1_pos maintains the position of the external interrupt signal lock axis and does not clear.



4.26 MC_ReadStatus（Read axis status）

This instruction is used to periodically read the status of the axis. Library: MotionControl

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_ReadStatus	Read axis status	FB		MC_ReadStatus_Instance (Axis :=parameter, Enable :=parameter, Valid=> parameter , Busy => parameter, Error=> parameter , ErrorID=> parameter , ErrorStop=> parameter , Disable=> parameter , Stopping => parameter, Homing=> parameter , Standstill => parameter, DiscreteMotion => parameter, ContinuousMotion=> parameter , SyncMotion=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Reads axis status FALSE: Stop reading axis status

■ Output variable

Name	Meaning	Data types	Valid range	Description
Valid	Valid	BOOL	TRUE or FALSE	TRUE when the instruction executes normally
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to “Instruction error code”
ErrorStop	ErrorStop	BOOL	TRUE or FALSE	When a motion instruction is executed, the axis state changes according to the executed instruction, and these output variables are used to display the current axis state of the specified axis.
Disabled	Disabled	BOOL	TRUE or FALSE	
Stopping	Stopping	BOOL	TRUE or FALSE	
Homing	Homing	BOOL	TRUE or FALSE	
Standstill	Standstill	BOOL	TRUE or FALSE	
DiscreteMotion	Discrete Motion	BOOL	TRUE or FALSE	
ContinuousMotion	Continuous Motion	BOOL	TRUE or FALSE	
SyncMotion	Synchronized Motion	BOOL	TRUE or FALSE	

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Vaild	When Enable changes to TRUE	◆ When Enable changes to FALSE ◆ When Error changes to TRUE
Busy	When Enable changes to TRUE	◆ When Enable changes to FALSE ◆ When Error is TRUE
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Enable changes from TRUE to FALSE
ErrorStop	When AxisState changes to ErrorStop	◆ When AxisState is not ErrorStop
Disabled	When AxisState changes to Disabled	◆ When AxisState is not Disabled
Stopping	When AxisState changes to Stopping	◆ When AxisState is not Stopping
Homing	When AxisState changes to Stopping	◆ When AxisState is not Stopping
Standstill	When AxisState changes to Standstill	◆ When AxisState is not Standstill
DiscreteMotion	When AxisState changes to DiscreteMotion	◆ When AxisState is not DiscreteMotion
ContinuousMotion	When AxisState changes to ContinuousMotion	◆ When AxisState is not ContinuousMotion
SyncMotion	When AxisState changes to SyncMotion	◆ When AxisState is not SyncMotion

■ Function description

● Basic function description

➢ This instruction is used to periodically read the status of the axis and is valid when the input variable Enable is TRUE. When the output variable Busy is TRUE indicating that the instruction is being executed. When Valid is TRUE it means that the axis status corresponding to the output variable is valid.

➢ The axis status is related to the executed motion instruction and the state of the servo drive. For details of the correspondence between motion instructions and axis status, refer to *3.1*.

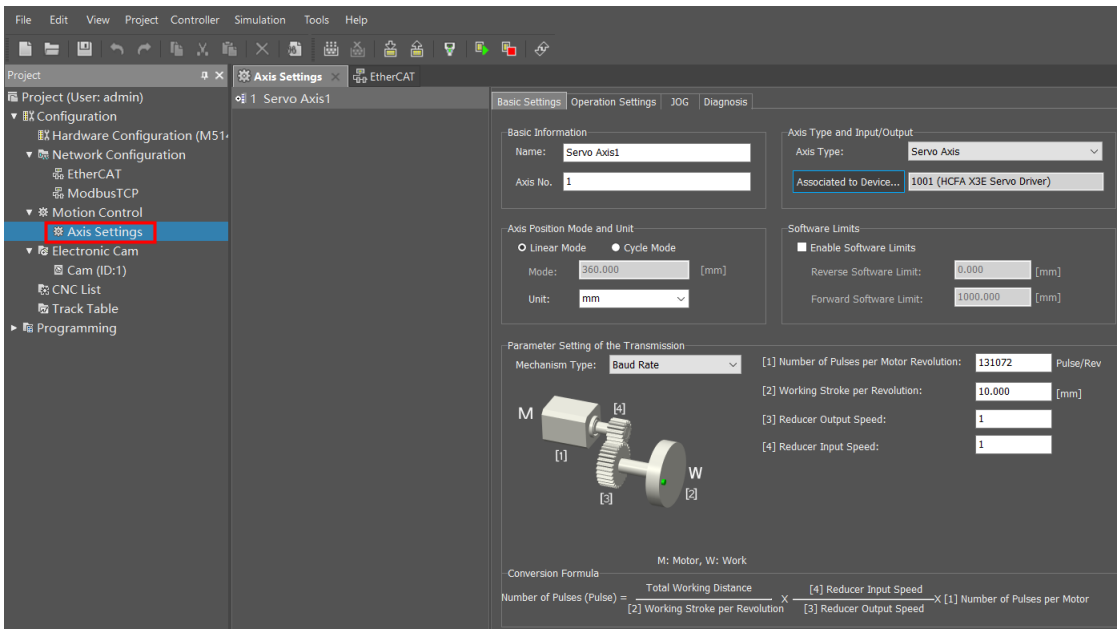
■ Example program

● Functionality

When MC_MoveRelative is executed, the axis status is read with the MC_ReadStatus instruction.

● Axis Parameter Settings

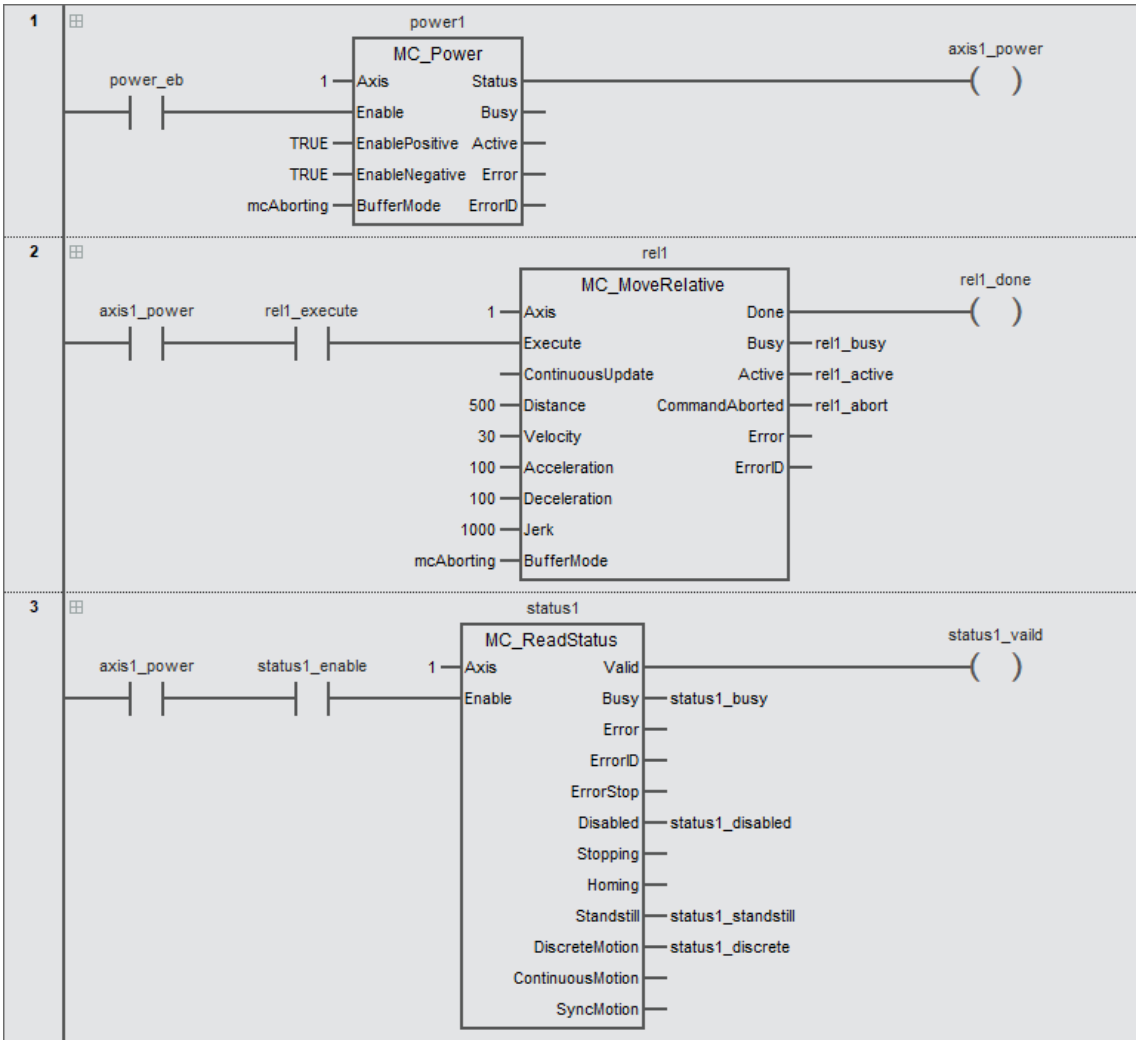
The axis parameter settings for axis1 are shown below:



- **Variable table**

Category	Name	Assigned to	Data Type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	rel1_execute		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	status1		MC_ReadStatus		
VAR	status1_enable		BOOL		
VAR	status1_vaild		BOOL		
VAR	status1_busy		BOOL		
VAR	status1_disabled		BOOL		
VAR	status1_standstill		BOOL		
VAR	status1_discrete		BOOL		

- **LD**

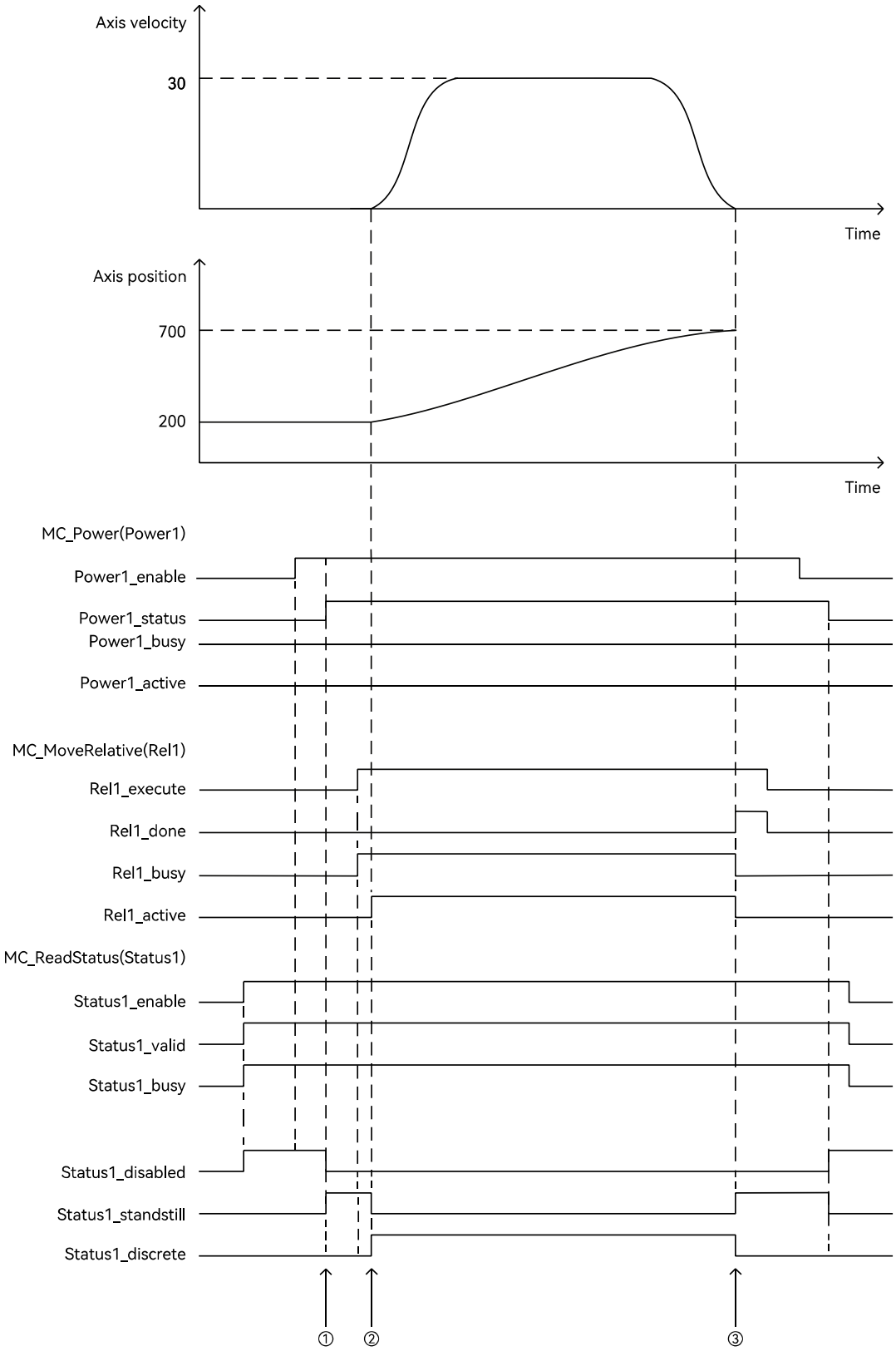


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis 1_power);


rel1(
    Axis:=1 ,
    Execute:=axis1_power AND rel1_execute ,
    Distance:=500 ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcAborting ,
    Done=>rel1_done ,
    Busy=>rel1_busy ,
    Active=>rel1_active ,
    CommandAborted=>rel1_abort );

status(
    Axis:=1 ,
    Enable:= axis1_power AND status_enable ,
    Valid=> status_vaild,
    Busy=> status_busy ,
    Disabled=> status_disabled ,
    Standstill=> status_standstill ,
    DiscreteMotion=> status_discrete );
```

- Program description
  - When status1_enable is TRUE, the status1 instruction starts to read the axis status. When the status1 instruction is executed, the axis is not enabled, the axis status is disabled , the output variable status1_disabled changes to TRUE from FALSE, and the other axis status output variables are FALSE.
  - After power1_enable changes to TRUE, the power1 instruction controls axis enable, the output variable power1_status changes to TRUE, and the axis status changes from disabled to standstill. The output variable status1_disabled changes to FALSE from TRUE and status1_standstill changes to TRUE from FALSE.
  - When rel1_excute changes to TRUE, the rel1 instruction is executed, and when rel1_active changes to TRUE, the rel1 instruction controls the axis operation and the axis status changes from standstill to discrete. The output variable status1_standstill changes to FALSE from TRUE, and the output variable status1 _discrete changes to TRUE from FALSE.
  - When execution of the rel1 instruction is complete (Done changes to TRUE), the axis status changes from DiscreteMotion to standstill. the output variable status1_standstill changes from FALSE to TRUE, and the output variable status1_discrete changes from TRUE to FALSE.
  - The power1 instruction controls axis disable when power1_enable changes to FALSE, and the axis status changes from standstill to disabled when the output variable power1_status changes to FALSE.

The output variable status1_standstill changes to FALSE from TRUE and status1_disabled changes to TRUE from FALSE.



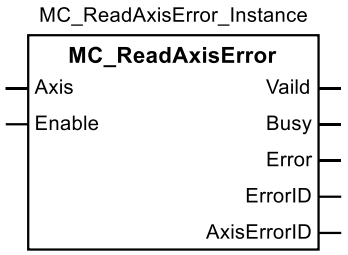
① The MC_Power instruction controls the axis state from Disabled to Standstill when the axis is enabled.

② After the relative instruction (rel1) is executed and rel1_active becomes TRUE, the axis state changes from Standstill to DiscreteMotion .

③ When the relative instruction (rel1) is completed (the Done bit becomes TRUE), the axis state changes from the DiscreteMotion to the Standstill .

4.27 MC_ReadAxisError（Read axis error）

This instruction is used to periodically read the error for the specified axis. Library: MotionControl

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_ReadAxisError	Read axis error	FB		MC_ReadAxisError _Instance (Axis:=parameter , Enable:=parameter , Valid=> parameter , Busy=> parameter , Error => parameter, ErrorID=> parameter , AxisErrorID=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Reads axis error code FALSE: Stop reading axis error code

■ Output variable

Name	Meaning	Data types	Valid range	Description
Valid	Valid	BOOL	TRUE or FALSE	TRUE when the instruction executes normally
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to “Instruction error code”
AxisErrorID	Axis error code	WORD	Positive number、0	Search axis exception code，please refer to Axis Error Code

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Valid	When the specified axis error is readable	◆ When Enable changes to FALSE ◆ When Error changes to TRUE
Busy	When Enable changes to TRUE	◆ When Enable changes to FALSE ◆ When Error changes to TRUE
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Enable changes from TRUE to FALSE

■ Function description

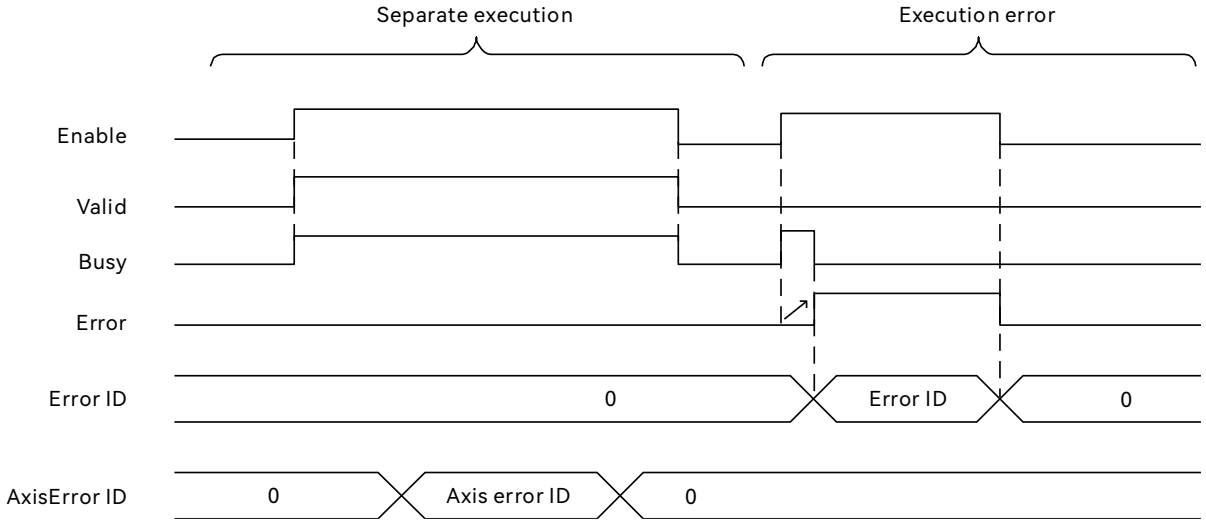
● Basic function description

This instruction is used to periodically read the error for the specified axis, the specified axis can be a servo axis, a virtual servo axis, a pulse axis, or an encoder axis. The instruction is executed when the input variable Enable is TRUE. When the difference between the command position and actual position of the axis exceeds the permissible setting value, the corresponding error code can be read from the output variable AxisErrorID of this instruction, and the cause of the axis abnormality can be queried through this error code.

When AxisErrorID is not 0, the meaning is shown in the following table.

AxisErrorID		Meaning	Processing method
Hexadecimal	Decimalism		
FF01	65281	The difference between the command and actual position of the axis exceeds the permissible setting value.	Check that the warning value for the difference between the commanded and actual position of the axis in the axis settings screen is reasonable Check if the axis has an alarm
FF02	65282	Abnormal EtherCAT network status	Check that the EtherCAT network on which the axis is located is connected to the controller.
FF03	65283	Axis position exceeds the set soft limit position	Execute the MC_Reset instruction first, and then run the axis in the backward direction to run the axis position within the soft limit range
FF04	65284	When the MC_Home instruction is executed, the controller changes to the stop state from the running state	Contact controller technicians to analyze the reason
FF05	65285	Driver error(Bit 3 of 0x6041(error bit)is ON)	Determine the cause of the error based on the drive's error code
FF06	65286	De-enable during drive operation	Contact technicians to analyze the reason
FF07	65287	The difference between the command position and the actual position inside the drive exceeds the allowable setting value	Contact drive technicians to analyze the reason

■ Output variable timing diagram



Separate execution

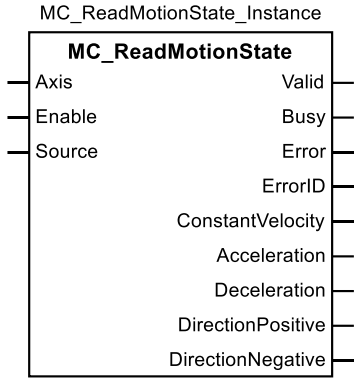
When Enable changes from FALSE to TRUE, Valid and Busy changes to TRUE at the same time. AxisErrorID displays the axis error code when Enable is TRUE and the specified axis has an error. The value of AxisErrorID changes to 0 when the axis error is resolved.

Execution error

When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy changes to TRUE. The next period Error changes to TRUE while Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When 0Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

4.28 MC_ReadMotionState（Read axis motion state）

This instruction is used to read the motion states and running direction of the specified axis. Library: MotionControl

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_ReadMotionState	Read axis motion state	FB		MC_ReadMotionState _Instance (Axis :=parameter, Enable :=parameter, Source:=parameter , Valid=> parameter , Busy=> parameter , Error => parameter, ErrorID => parameter, ConstantVelocity => parameter, Acceleration=> parameter , Deceleration=> parameter , DirectionPositive=> parameter , DirectionNegative=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Reads axis motion state FALSE: Stop reading motion state

■ Output variable

Name	Meaning	Data types	Valid range	Description
Valid	Valid	BOOL	TRUE or FALSE	TRUE when the instruction executes normally
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to “Instruction error code”
ConstantVelocity	Constant velocity	BOOL	TRUE or FALSE	TRUE if the axis motion state is at constant velocity, FALSE if it is not.
Acceleration	Acceleration state	BOOL	TRUE or FALSE	TRUE if the axis motion state is at acceleration state, FALSE if it is not.
Deceleration	Deceleration state	BOOL	TRUE or FALSE	TRUE if the axis motion state is at deceleration state, FALSE if it is not.
DirectionPositive	Positive direction	BOOL	TRUE or FALSE	TRUE if the axis motion state is at positive direction, FALSE if it is not.
DirectionNegative	Negative direction	BOOL	TRUE or FALSE	TRUE if the axis motion state is at negative direction, FALSE if it is not.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Valid	When the specified axis motion state is readable	◆ When Enable changes to FALSE ◆ When Error changes to TRUE
Busy	When Enable changes to TRUE	◆ When Enable changes to FALSE ◆ When Error changes to TRUE
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Enable changes from TRUE to FALSE
ConstantVelocity	When the axis motion state is at constant velocity	◆ When the axis motion state is not at constant

		velocity
Acceleration	When the axis motion state is at acceleration state	◆ When the axis motion state is not at acceleration state
Deceleration	When the axis motion state is at deceleration state	◆ When the axis motion state is not at deceleration state
DirectionPositive	When the axis motion state is at positive direction	◆ When the axis motion state is not at positive direction
DirectionNegative	When the axis motion state is at negative direction	◆ When the axis motion state is not at negative direction

■ Function description

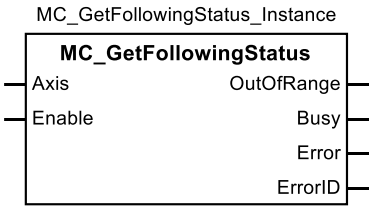
● Basic function description

- ➢ This instruction is used to periodically read the status of the axis and is valid when the input variable Enable is TRUE. When the output variable Busy is TRUE indicating that the instruction is being executed. When Valid is TRUE it means that the axis motion state corresponding to the output variable is valid.
- ➢ The controller judges whether the axis is in ConstantVelocity, Acceleration or Deceleration state by the axis command velocity and command acceleration. The axis can be judged only when it is in the running state, and it can't be judged when the axis is not in the running state.
- ➢ The controller judges whether the axis is in DirectionPositive or DirectionNegative state by the command velocity and command acceleration of the axis. The axis can be judged only when it is in the running state, and it can't be judged when the axis is not in the running state.

4.29 MC_GetFollowingStatus（Read position difference status）

This instruction is used to read in real time whether the position difference between the command position and the actual position of the specified axis exceeds the set value or not. Library:MotionControl_Part2

Applicable models: M500S Series、M500 Series

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_GetFollowingStatus	Read position difference status	FB		MC_GetFollowingStatus_Instance (Axis:=parameter , Enable:=parameter , OutOfRange => parameter, Busy=> parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Read position difference is out of range or not FALSE: Stop reading status

■ Output variable

Name	Meaning	Data types	Valid range	Description
OutOfRange	Out of range	BOOL	TRUE or FALSE	TRUE when the position difference between the command position and the actual position of the specified axis exceeds the set value
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to "*Instruction error code*"

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
OutOfRange	When the position difference between the command position and the actual position of the specified axis exceeds the set value	◆ When the position difference between the commanded and actual axis positions does not exceed the set value
Busy	When Enable changes to TRUE	◆ When Error changes to TRUE
Error	The value of the input variable is not within the allowed range	◆ When Enable changes to FALSE

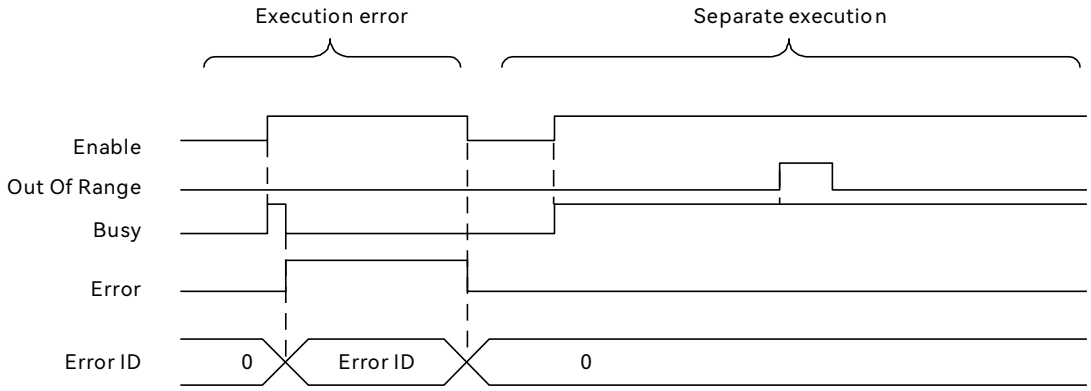
■ Function description

● Basic function description

This instruction is used to read in real time whether the position difference between the command position and the actual position of the specified axis exceeds the set value or not. If the difference between the commanded position and the actual position of the axis exceeds this setting value, the axis enters the errorstop state. After executing MC_Reset, the axis command position and actual position will be the same. The upper limit of the difference between the command and actual position of the axis and the filter time

can be set via the "Operation Settings" in the "Axis Settings" section of the software, or be set by the instruction MC_SetFollowingParm.

■ **Output variable timing diagram**



- **Execution error**

  When the value of the input variable is not within the allowable range, Enable changes from FALSE to TRUE while Busy changes to TRUE. The next period Error changes to TRUE while Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Enable changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

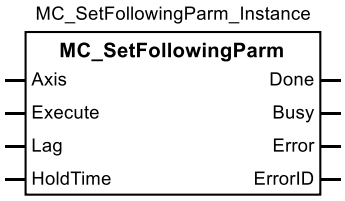
- **Separate execution**

  When Enable changes from FALSE to TRUE, Busy changes to TRUE at the same time. When Enable is TRUE, OutOfRange changes to TRUE when the position difference between the command and actual position of the axis exceeds the allowable range; OutOfRange changes to FALSE when the position difference between the command and actual position of the axis does not exceed the allowable range.

4.30 MC_SetFollowingParm（Set position difference parameters）

This instruction is used to set the allowable position difference between the command position and actual position of the specified axis and the duration of the position difference. Library:MotionControl_Part2

Applicable models: M500S Series、M500 Series

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_SetFollowingParm	Set position difference parameters	FB		MC_SetFollowingParm_Instance (Axis:=parameter , Execute :=parameter, Lag :=parameter, HoldTime:=parameter , Done=> parameter , Busy => parameter, Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
Lag	Position difference	LREAL	Positive number	Required field	Set position difference
HoldTime	Hold time	LREAL	Positive number	Required field	Duration of exceeding the set position difference setting value (unit: s)

■ Output variable

Name	Meaning	Data types	Valid range	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to "*Instruction error code*"

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then FALSE one period later
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE
Error	The value of the input variable is not within the allowed range	◆ When Execute changes from TRUE to FALSE

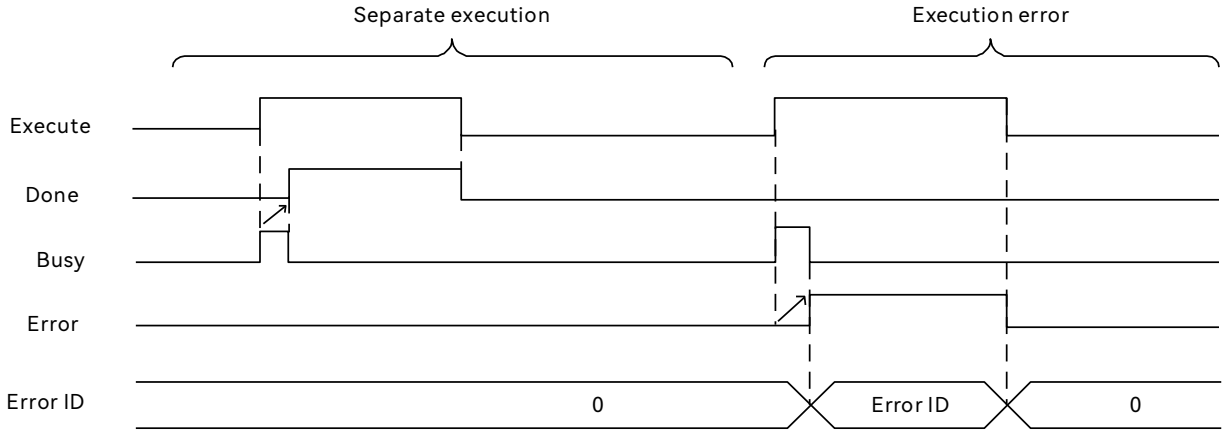
■ Function description

- **Basic function description**

This instruction is used to set the allowable position difference between the command position and actual position of the specified axis and the duration of the position difference. HoldTime and Lag can be set by [Axis Settings] → [Operation Settings] in the software. If the difference between the commanded position and the actual position of the axis exceeds this setting value, the axis enters the errorstop state.

After executing MC_Reset, the axis command position and actual position will be the same. The MC_GetFollowingStatus instruction can be used to read position difference.

■ Output variable timing diagram



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Active changes to TRUE in the next period, Busy changes to FALSE at the same time. When Execute changes to FALSE, Done changes to FALSE at the same time.

- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy changes to TRUE. The next period Error changes to TRUE while Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When 0Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

4.31 MC_EnableSoftLimit（Enable software limit）

This instruction is used to enable the software limit function. Library: MotionControl_Part2

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_EnableSoftLimit	Enable software limit	FB	MC_EnableSoftLimit_Instance MC_EnableSoftLimit Axis — Done Enable — Error EnableLimit — ErrorID	MC_EnableSoftLimit_Instance (Axis :=parameter, Enable :=parameter, EnableLimit :=parameter, Done => parameter, Error => parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Enable	Enable	BOOL	TRUE or FALSE	FALSE	TRUE: Instruction execution FALSE: instruction is not executed.
EnableLimit	Enable software limit	LREAL	TRUE or FALSE	FALSE	TRUE: Enable software limit FALSE: disable software limit

■ Output variable

Name	Meaning	Data types	Valid range	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to "*Instruction error code*"

■ Output variable refreshing timing

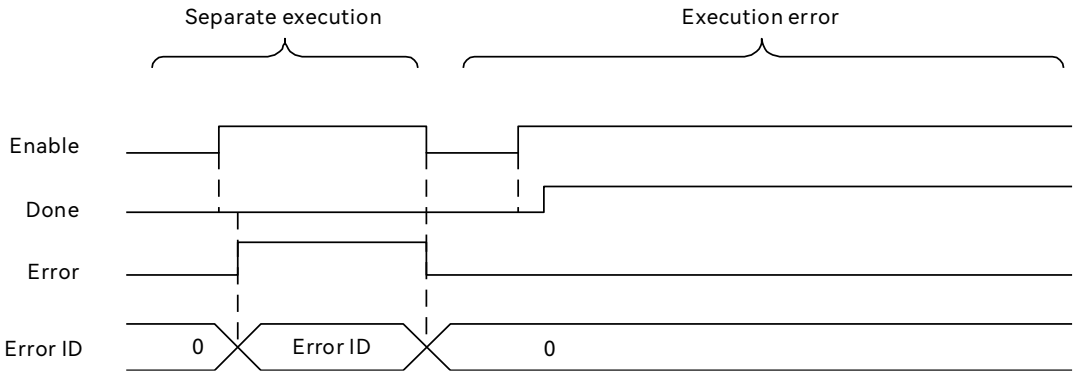
Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction is completed	◆ When Done is TRUE and Enable changes to FALSE from TRUE ◆ When the instruction is executed and Execute is FALSE, Done becomes TRUE and then FALSE after one cycle.
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Enable changes from TRUE to FALSE

■ Function description

● Basic function description

This instruction is used to enable the software limit function. The software limits can be set by the software and can also be set and changed by this instruction. Without this instruction, the software setting will prevail. After enabling the software limit, the position of the axis exceeds the software positive limit or the software negative limit, the axis state will enter the ErrorStop state, which needs to be cleared by executing MC_Reset reset.

■ Output variable timing diagram



- **Execution error**

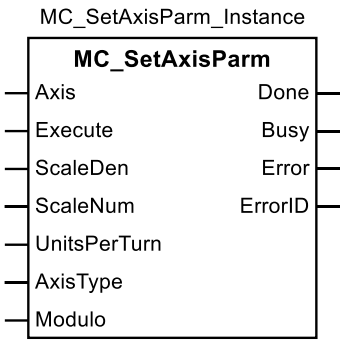
  When the value of the input variable is not within the allowable range, Enable changes from FALSE to TRUE. The next period Error changes to TRUE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Enable changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Separate execution**

  When Enable changes from FALSE to TRUE, Done changes to TRUE in the next period.

4.32 MC_SetAxisParm（Set axis parameters）

This instruction is used to set the axis parameters. Library:MotionControl_Part2

Instructions	Name	FB/FUN	Graphic expression	ST expression
MC_SetAxisParm	Set axis parameters	FB		MC_ChangeAxisParm _Instance (Axis :=parameter, Execute :=parameter, ScaleDen:=parameter , ScaleNum :=parameter, UnitsPerTurn:=parameter, AxisType :=parameter, Modulo:=parameter , Done=> parameter , Busy => parameter, Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data types	Valid range	Default	Description
Axis	Axis number	USINT	Depends on model	Required field	Specify the axis number of the control axis
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected
ScaleDen	Gear ratio numerator	LREAL	Positive integer	Required field	Gear ratio numerator
ScaleNum	Gear ratio denominator	LREAL	Positive integer	Required field	Gear ratio denominator
UnitsPerTurn	Working stroke per revolution	LREAL	Positive number	Required field	Number of travel units moved by one revolution of the working mechanism (Unit: Travel units)
AxisType	Axis type	USINT	0、1	0	Axis type 0: Rotary mode 1: Linear mode
Modulo	Modulo	LREAL	Positive number	Required field	Modulo value

■ Output variable

Name	Meaning	Data types	Valid range	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error
ErrorID	Error Code	WORD	0~65535	Contains the error code when an error occurs For the meaning of the value, please refer to “Instruction error code”

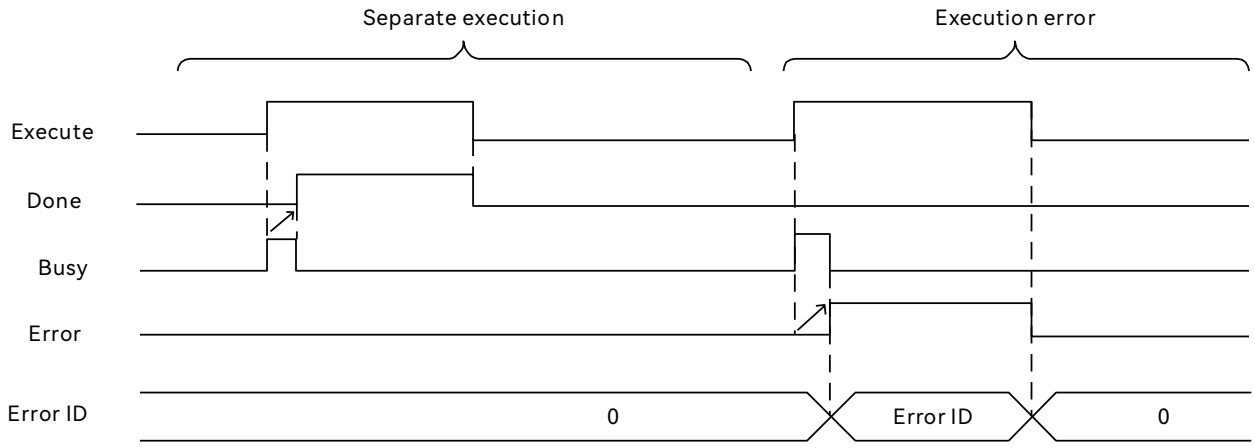
■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then FALSE one period later
Busy	When Execute changes to TRUE	◆ When Done changes from FALSE to TRUE ◆ When Error changes from FALSE to TRUE
Error	The value of the input variable is not within the allowed range	◆ When Error is TRUE and Enable changes from TRUE to FALSE

■ Function description

- This instruction is used to set the axis parameters. When the linkage mechanism of the axis is changed, such as when the parameters of the reducer are changed, this instruction can be used to change the parameters of the axis to be consistent with the parameters of the actual mechanism, which is convenient for the user.
- This instruction must be executed when axis status is Disable or Standstill; If the axis is in any other status, the instruction reports an error.
- After the controller is powered on, the instruction must be re-executed for the set parameters to take effect; after the controller is powered on, if the instruction is not executed, the axis parameters are the same as those in the software 【Axis Setting】→ 【Basic Setting】.
- When using this command, user need to familiarize themselves with the meaning of each parameter of the command. Failure to do so may result in axis speeds that do not match what is expected, causing accidents or hazards.
- After using this instruction, carefully check whether the velocity and position of the motion instruction in the program need to be changed, and then run the program after confirming that there is no error.

■ Output variable timing diagram



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Done changes to TRUE in the next period while Busy changes to FALSE. When Execute changes to FALSE, Done changes to FALSE at the same time.

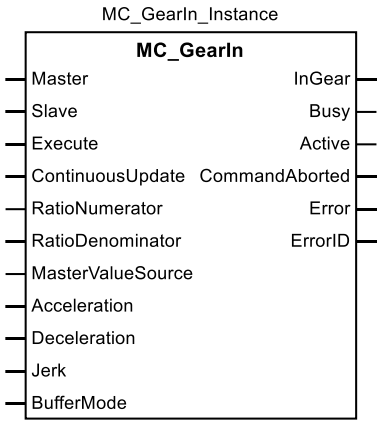
- **Execution error**

  When the value of the input variable is not within the allowable range, Execute changes from FALSE to TRUE while Busy changes to TRUE. The next period Error changes to TRUE while Busy changes to FALSE, ErrorID outputs the corresponding error code, which can be used to find the cause of the problem. When Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

Chapter5 Multi-axis instructions

5.1 MC_GearIn（Electronic gears）

Specifies the gear ratio between the master axis and the slave axis and starts gear operation.
Library: MotionControl

Instruction	Meaning	FB/FUN	Graphic expression	ST expression
MC_GearIn	Electronic gears	FB		MC_GearIn _Instance (Master:=parameter, Slave:=parameter, Execute:=parameter, ContinuousUpdate:=parameter, RatioNumerato:=parameter, RatioDenominator:=parameter , MasterValueSource:=parameter, Acceleration:=parameter, Deceleration:=parameter, Jerk:=parameter , BufferMode:=parameter, InGear=> parameter , Busy=> parameter, Active=> parameter, CommandAborted=> parameter, Error=> parameter, ErrorID=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Master	Master axis number	USINT	Depends on model	Required field	Specify master axis number in the electronic gear.
Slave	Slave axis number	USINT	Depends on model	Required field	Specify slave axis number in the electronic gear.
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected.
ContinuousUpdate	continually updated	BOOL	TRUE or FALSE	FALSE	Reserved
RatioNumerator	Gear ratio numerator	LREAL	Positive or negative number	Required field	Specify the numerator of the electronic gear ratio between the master and slave axes.
RatioDenominator	Gear ratio denominator	LREAL	Positive number	Required field	Specify the denominator of the electronic gear ratio between the master and slave axes.
MasterValueSource	Master position type setting	MC_Source	0: mcSetValue 1: mcActualValue	0	Setting the slave axis to follow the master axis command position or actual position. 0: Setting the slave axis to follow the master axis command position. 1: Setting the slave axis to follow the master axis actual position.
Acceleration	Acceleration	LREAL	Positive number	Required field	When the slave axis is coupling with the master axis, the slave axis acceleration rate. *1 (Unit :travel units/s^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	When the slave axis is coupling with the master axis, the slave axis deceleration rate. *1 (The unit :travel units/s^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	When the slave axis is coupling with the master axis, the slave axis jerk. *1

					(The unit :travel units/s²) *2
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered	0	Specify the behavior when executing between the two motion instruction. *3 0: Aborting 1: Buffered

*1: For the relationship between Velocity, Acceleration, Deceleration and Jerk, please refer to the section ' Motion Control Command Parameter Description '.

*2: Refer to the ' Motion Control Command Parameter Units ' for details on travel units.

*3: For details of BufferMode, please refer to ' Description of Buffer Mode for Multi-Startup of Motion Control Instructions '.

■ Output Variables

Output variable	Meaning	Data type	Default	Description
InGear	Gear ratio achieved	BOOL	TRUE or FALSE	TRUE when the slave axis reaches the target velocity.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged.
Active	Controlling	BOOL	TRUE or FALSE	TRUE when the axis is being controlled.
CommandAborted	Command aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted.
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Meaning	Whether or not to become TRUE	Whether or not to become False
InGear	When the slave axis commanded velocity reaches the target velocity	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
Busy	When Execute changes to TRUE.	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
Active	When the instruction is started.	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
CommandAborted	When the instruction is aborted	◆ When Execute is TRUE and changes to FALSE ◆ After one period when Execute is FALSE or the instruction is aborted by other instruction.
Error	The input variable of the instruction is out of the allowed range, the execution conditions of the instruction are not satisfied, or an exception is encountered during the execution of the instruction.	◆ When Execute is TRUE and changes to FALSE

■ Instruction descriptions

● Basic Instruction descriptions

- When this instruction is executed, the slave axis performs a gear movement with the master axis according to the set value of the input variable for this instruction. The slave axis can choose to follow the command position or the feedback position of the master axis to perform the gearing action by inputting the value of the variable MasterValueSource. The master axis number can be set as servo axis, virtual servo axis, pulse axis, or encoder axis, and the slave axis number can be set as servo axis, virtual servo axis or pulse axis.
- When this instruction is executed, the slave axis state must be in the enable state to be executed, and the master axis state can be executed in the enable or disable state.
- The master and slave axes are decoupled by executing the MC_GearOut instruction, and the slave axis

maintains the slave velocity at the time of decouple after decouple.

- The slave axis can be stopped by MC_Halt or MC_Stop instruction when the master and slave axis are coupled.
- After this instruction is executed, if the slave axis does not reach the target velocity (when InGear of this instruction is FALSE) during the process of establishing the electronic gear relationship between the two axes (e.g. when the instruction is executed while the master axis is running). The slave axis accelerates or decelerates in accordance with Acceleration, Deceleration, Jerk, which are set by this instruction. The target velocity of the slave axis is shown in the following equation. The master and slave axes are not synchronized during this process, and the synchronization error of the master and slave axes caused by the acceleration process is not automatically compensated.
- After this instruction is executed, after the electronic gear relationship between the two axes is established (when the InGear is TRUE), the relationship between the axis velocity, acceleration, deceleration, and master axis velocity, acceleration, and deceleration is as follows:

$$\text{slave axis target velocity} = \text{the master axis velocity} \times \frac{\boldsymbol{Gear\ ratio\ numerator}}{\boldsymbol{Gear\ ratio\ denominator}}$$

$$\text{slave axis acceleration} = \text{the master axis acceleration} \times \frac{\boldsymbol{Gear\ ratio\ numerator}}{\boldsymbol{Gear\ ratio\ denominator}}$$

$$\boldsymbol{Electronic\ Gear\ Ratios} = \frac{\boldsymbol{Gear\ ratio\ numerator}}{\boldsymbol{Gear\ ratio\ denominator}}$$

Deceleration same as the Acceleration

- When the electronic gear ratio is positive, the direction of motion of the slave axis and the master axis are the same. When the electronic gear ratio is negative, the direction of motion of the slave axis and the master axis are opposite.

- **Re-execute the instruction**

When the instruction is being executed , the execute changes from FALSE to TRUE, the instruction can be re-executed; the pin parameters can be re-enacted include RatioNumerator, RatioDenominator, Velocity, Acceleration, Deceleration, and Jerk. The other pin parameters will not take effect. When the instruction creates BufferMode relationship with other motion instructions, change the parameters of this instruction, the parameters of this order will take effect after re-triggering, the original handover relationship will still be maintained, and the handover speed will be recalculated.

- **Execute this instruction while other instructions are in processing**

This instruction is executed when other instructions are activated, and how this instruction is buffered to other motion instructions is determined by the value of the BufferMode input variable for this instruction. The only values that can be set for the instruction input variable BufferMode is mcAborting or mcBuffered.

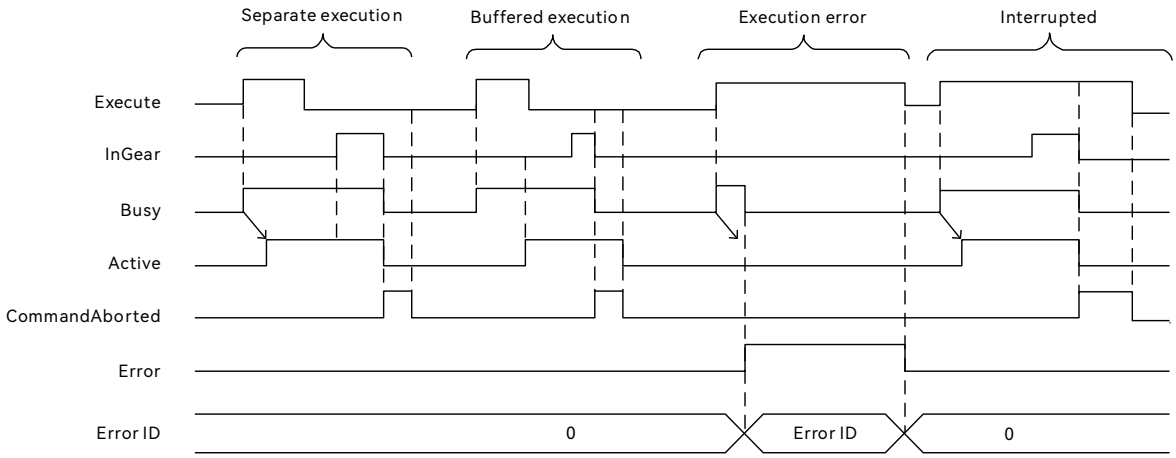
- **Execute other instructions while this instruction are in processing**

Other instructions are executed when this instruction is activated, other motion instructions and this instruction how to buffered is determined by the value of BufferMode of other motion instructions, the value of BufferMode of other motion instructions can only be selected as interrupt or wait; if there is no BufferMode parameter of other motion instructions, it is generally interrupting this instruction. When other motion instructions and this instruction are cached, the timing of execution of other instructions is when the instruction "InGear" becomes TRUE.

- **Abnormality elimination**

When this instruction is executed, if the input variable is illegal, the instruction Error becomes TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g. axis alarm), the instruction will also report error and the axis will stop immediately.

- **The timing diagram of output variables description**



- **Separate execution**

When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Active changes to TRUE for the next period. When the slave axis command velocity reaches the target velocity, InGear changes to TRUE, and Busy and Active remain TRUE.

- **Relay execution**

The instruction is executed when the other instruction controls the axis (the value of BufferMode is not mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction completes.

- **Error execution**

When the value of the input variable of this instruction is not within the allowable range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next period while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that you can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to zero.

- **Execution abortion**

When the execution of this instruction is aborted by other instructions, the instruction CommandAborted becomes TRUE , Busy, Active and InGear become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

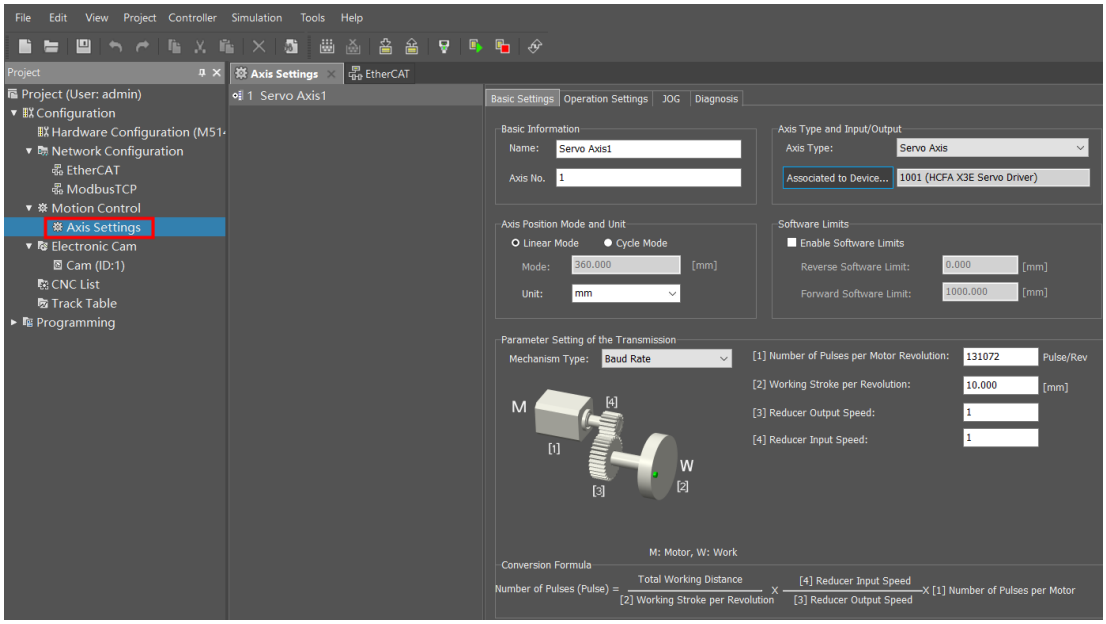
■ **Example program**

- **Functionality**

When the numerator of the electronic gear ratio changes, the value of the numerator of the electronic gear ratio is modified to realize real-time speed regulation of the slave axis speed. Two electronic gear instructions, which are executed alternately when a change in the electronic gear ratio is detected.

- **Axis parameter setting**

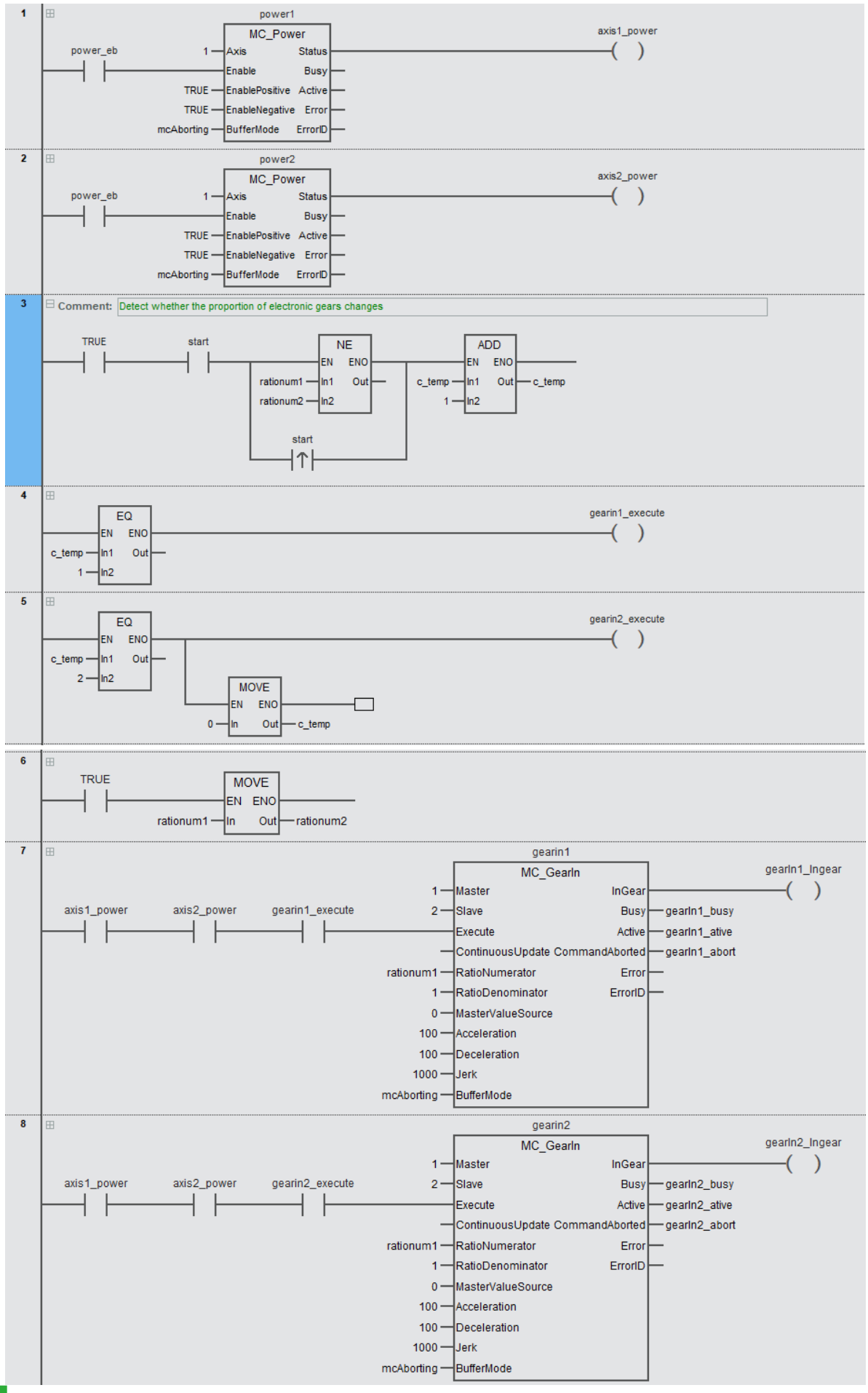
The axis parameters for mater axis 1 and slave axis 2 are the same, and the axis parameter settings for axis 1 are shown below.

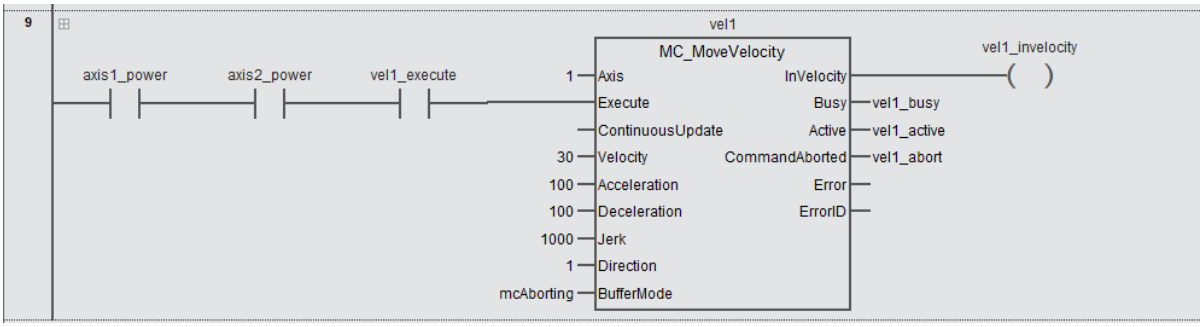


- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	power2		MC_Power		
VAR	axis2_power		BOOL		
VAR	gearin1		MC_GearIn		
VAR	gearIn1_Ingear		BOOL		
VAR	gearIn1_busy		BOOL		
VAR	gearIn1_ative		BOOL		
VAR	gearIn1_abort		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_execute		vel1_invelocity		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	rationum1		LREAL		
VAR	rationum2		LREAL		
VAR	start		BOOL		
VAR	gearin2		MC_GearIn		
VAR	gearIn2_Ingear		BOOL		
VAR	gearIn2_busy		BOOL		
VAR	gearIn2_ative		BOOL		
VAR	gearIn2_abort		BOOL		
VAR	gearin1_execute		BOOL		
VAR	gearin2_execute		BOOL		
VAR	c_temp		UINT		

- **LD**


1
power1
MC_Power
power_eb
1
Axis
Status
Enable
Busy
TRUE
EnablePositive
Active
TRUE
EnableNegative
Error
mcAborting
BufferMode
ErrorID
axis1_power
2
power2
MC_Power
power_eb
1
Axis
Status
Enable
Busy
TRUE
EnablePositive
Active
TRUE
EnableNegative
Error
mcAborting
BufferMode
ErrorID
axis2_power
3
Comment:
Detect whether the proportion of electronic gears changes
TRUE
start
NE
EN
ENO
rationum1
In1
Out
rationum2
In2
ADD
EN
ENO
c_temp
In1
Out
c_temp
1
In2
start
4
EQ
EN
ENO
c_temp
In1
Out
1
In2
gearin1_execute
5
EQ
EN
ENO
c_temp
In1
Out
2
In2
gearin2_execute
MOVE
EN
ENO
0
In
Out
c_temp
6
TRUE
MOVE
EN
ENO
rationum1
In
Out
rationum2
7
gearin1
MC_GearIn
axis1_power
axis2_power
gearin1_execute
1
Master
InGear
gearIn1_Ingear
2
Slave
Busy
gearIn1_busy
Execute
Active
gearIn1_ative
ContinuousUpdate
CommandAborted
gearIn1_abort
rationum1
RatioNumerator
Error
1
RatioDenominator
ErrorID
0
MasterValueSource
100
Acceleration
100
Deceleration
1000
Jerk
mcAborting
BufferMode
8
gearin2
MC_GearIn
axis1_power
axis2_power
gearin2_execute
1
Master
InGear
gearIn2_Ingear
2
Slave
Busy
gearIn2_busy
Execute
Active
gearIn2_ative
ContinuousUpdate
CommandAborted
gearIn2_abort
rationum1
RatioNumerator
Error
1
RatioDenominator
ErrorID
0
MasterValueSource
100
Acceleration
100
Deceleration
1000
Jerk
mcAborting
BufferMode



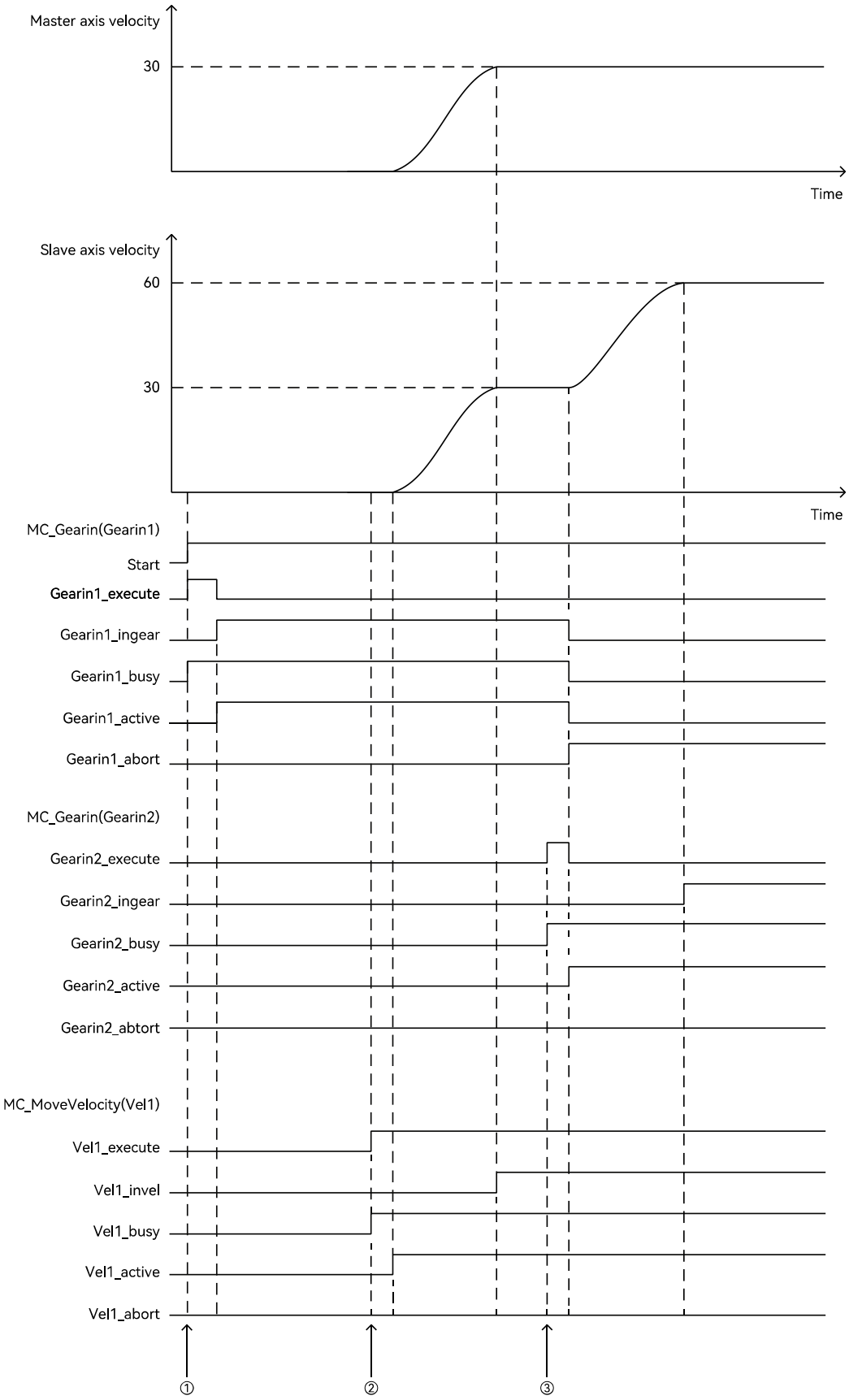
- ST

```
power1(
    Axis:=p1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );


power2(
    Axis:=2 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis2_power
    );


IF   start AND rationum1<>rationum2 THEN
    c_temp:=c_temp+1;
END_IF;

IF c_temp=0 THEN
    gearin1_execute:=TRUE;
ELSE
    gearin1_execute:=FALSE;
END_IF;

IF c_temp=1 THEN
    gearin2_execute:=TRUE;
    c_temp:=0;
ELSE
    gearin1_execute:=FALSE;
END_IF;

rationum2:=rationum1;

gearin1(
    Master:=1 ,
    Slave:=2 ,
    Execute:= axis1_power AND axis2_power AND gearin1_execute ,
    RatioNumerator:=rationum1 ,
```

```
        RatioDenominator:=1 ,
        MasterValueSource:=0 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:=mcAborting ,
        InGear=>gearIn1_Ingear ,
        Busy=>gearIn1_busy ,
        Active=>gearIn1_busy ,
        CommandAborted=>gearIn1_abort
        );

    gearin2(
        Master:=1 ,
        Slave:=2 ,
        Execute:=axis1_power AND axis2_power AND gearin2_execute,
        RatioNumerator:=rationum1 ,
        RatioDenominator:=1 ,
        MasterValueSource:=0 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:=mcAborting ,
        InGear=>gearIn2_Ingear ,
        Busy=>gearIn2_busy ,
        Active=>gearIn2_busy ,
        CommandAborted=>gearIn2_abort
        );

    vel1(
        Axis:=1 ,
        Execute:= axis1_power AND axis2_power AND vel1_execute,
        Velocity:=30 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        Direction:=1 ,
        BufferMode:=mcAborting ,
        InVelocity=> vel1_invelocity ,
        Busy=> vel1_busy ,
        Active=> vel1_active ,
        CommandAborted=> vel1_abort
        );
```

- **Program description**
  - After the axis is enabled, the initial value of rationum1 (electronic gear ratio numerator) is set to 1. When the variable of start changes to TRUE, gearin1_execute changes to TRUE, and the 1st electronic gear instruction (gearin1) starts to be executed, and after the 2nd period, the electronic gear with a gear ratio of 1:1 is established from the axis 2 and the main axis 1. gear relationship.
  - After a gear relationship is established between the slave axis and the master axis, the master axis executes the instruction velocity and the slave axis follows the master.

- To adjust the slave speed, modify the numerator value of the electronic gear ratio in the MC_GearIn instruction. For instance, if the value of rationum1 (the numerator of the electronic gear ratio) changes from 1 to 2, the program will set gearin2_execute to TRUE, triggering the execution of the second electronic gear instruction (gearin2). After gearin2 is executed, the execution of gearin1 will be interrupted. This allows the slave axis speed to be adjusted by changing the gear ratio.
- How the two electronic gear commands can be switched with each other is as follows. After the start variable is TRUE, the value of the c_temp (variable) increased by 1, and it starts to detect whether the values of rationum1 and rationum2 are the same, it detects whether the value of the numerator of the electronic gear ratio has changed or not, and if it has changed, then the value of the c_temp (variable) increased by 1, and the value of the c_temp (variable) changes to 1, and the value of the gearin1_execute changes to TRUE, which triggers the execution of the 1st electronic gear instruction (gearin1), and when the value of the c_temp variable is 2, gearin2_execute changes to TRUE, which triggers the execution of the 2nd electronic gear instruction (gearin2) and sets the value of c_temp to 0, and so on.



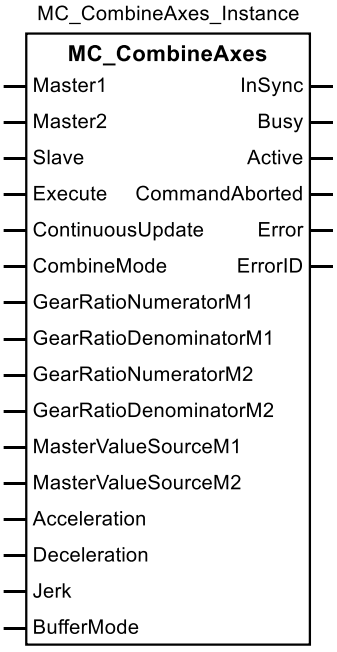
① The 1st gearin instruction (gearin1) starts execution, and the slave axis establishes a gear relationship with the axis.

② After the slave axis and the main axis establish a gear relationship, the main axis executes a velocity command, and the slave axis follows the main axis.

③ When the numerator of the electronic gear changes, the 2nd gearin instruction is executed, interrupting the 1st gearin instruction. The velocity of the slave axis changes according to the gearin ratio.

5.2 MC_CombineAxes（Twin master axis electronic gears）

This instruction is used to establish an electronic gear relationship between two master and one slave axis with set gear ratios. Library：MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_CombineAxes	Twin master axis electronic gears	FB		MC_CombineAxes _Instance (Master1:=parameter, Master2:=parameter, Slave:=parameter, Execute:=parameter , ContinuousUpdate:=parameter , CombineMode :=parameter, GearRatioNumeratorM1:=parameter, GearRatioDenominatorM1 :=parameter, GearRatioNumeratorM2:=parameter , GearRatioDenominatorM2 :=parameter, MasterValueSourceM1:=parameter, MasterValueSourceM2:=parameter , Acceleration :=parameter, Deceleration:=parameter , Jerk:= :=parameter, BufferMode, InSync=> parameter , Busy => parameter , Active=> parameter , CommandAborted=> parameter , Error => parameter , ErrorID=> parameter);

■ Input variable

Input Variable	Meaning	Data type	Valid range	Default	Description
Master1	Master axis 1 number	USINT	Depends on model	Required field	Specify master axis 1 number in the electronic gear.
Master2	Master axis 2 number	USINT	Depends on model	Required field	Specify master axis 2 number in the electronic gear.
Slave	Slave axis number	USINT	Depends on model	Required field	Specify slave axis number in the electronic gear.
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected.
ContinuousUpdate	Continually updated	BOOL	TRUE or FALSE	FALSE	Reserved
CombineMode	Addition and subtraction mode selection	MC_Combine_Mode	0: mcAddAxes 1: mcSubAxes	0	Setting the master 1 and master 2 addition and subtraction modes 0: Addition of master 1 and master 2 change positions 1: Subtraction of master 1 and master 2 change positions
GearRatioNumeratorM1	The master 1 gear ratio numerator	LREAL	Positive or negative number	Required field	Specify the numerator of the electronic gear ratio between the master 1 and slave axes.
GearRatioDenominatorM1	The master 1 gear ratio denominator	LREAL	Positive or negative number	Required field	Specify the denominator of the electronic gear ratio between the master 1 and slave axes.
GearRatioNumeratorM2	The master 2 gear ratio numerator	LREAL	Positive or negative number	Required field	Specify the numerator of the electronic gear ratio between the master 2 and slave axes.
GearRatioDenominatorM2	The master 2	LREAL	Positive or	Required	Specify the denominator of the electronic

	gear ratio denominator		negative number	field	gear ratio between the master 2 and slave axes.
MasterValueSourceM1	Master 1 position type setting	MC_Source	0:mcSetValue 1:mcActualValue	0	Setting the position type of master 1 0: Command position 1: Actual position
MasterValueSourceM2	Master 2 position type setting	MC_Source	0:mcSetValue 1:mcActualValue	0	Setting the position type of master 2 0: Command position 1: Actual position
Acceleration	Acceleration	LREAL	Positive number	Required field	When the slave axis couples the master axis, the slave axis acceleration rate. *1 (The unit :travel units/s2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	When the slave axis couples the master axis, the slave axis deceleration rate. *1 (The unit :travel units/s2) *2
Jerk	Jerk	LREAL	Positive number	Required field	When the slave axis couples the master axis, the slave axis jerk rate. *1 (The unit :travel units/s2) *2
BufferMode	Buffer mode selection	MC_Buffer_Mode	0：mcAborting 1：mcBuffered	0	Specify the behavior when executing between the two motion instruction. *3 0: Aborting 1: Buffered

*1: For the relationship between Velocity, Acceleration, Deceleration and Jerk, please refer to the section ' Motion Control Command Parameter Description '.

*2: Refer to the ' Motion Control Command Parameter Units ' for details on travel units.

*3: For details of BufferMode, please refer to ' Description of Buffer Mode for Multi-Startup of Motion Control Instructions '.

■ Output Variables

Output variable	Meaning	Data type	Default	Description
InSync	Synchronizing	BOOL	TRUE or FALSE	TRUE when the slave axis reaches the target velocity.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is acknowledged.
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is being controlled.
CommandAborted	Abortion	BOOL	TRUE or FALSE	TRUE when the instruction is aborted.
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
InSync	When the instruction velocity from the axis reaches the target velocity	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
Busy	When Execute changes to TRUE.	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
Active	When the instruction is aborted	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
CommandAborted	When the instruction is aborted	◆ After one period when Execute is FALSE or the instruction is aborted by other instruction.
Error	The input variable of the instruction is out of the allowed range, the execution conditions of the	◆ When Execute is TRUE and changes to FALSE

	instruction are not satisfied, or an exception is encountered during the execution of the instruction.	

Instruction descriptions

Basic Instruction descriptions

- After the execution of this instruction, the controller calculates the relative values of the positions generated by master 1 and master 2 for each period. The relative position values generated by each master are then multiplied by the corresponding gear ratios, the positions of the two master after the gear ratios are added or subtracted, finally add the value the current position of the slave axis, the value obtained is passed to the slave axis.
- The relative position values generated by the two masters per period multiplied by the electronic gear ratio can be selected, added or subtracted by entering the value of the variable CombineMode. The slave axes can be selected to follow the commanded or feedback position of the master axis for gearing by entering the values of the variables MasterValueSourceM1 and MasterValueSourceM2. The master axis number can be set as servo axis, virtual servo axis, pulse axis, or encoder axis, and the slave axis number can be set as servo axis, virtual servo axis, or pulse axis.
- When this instruction is executing, the slave axis state must be in the enable state to be executed, and the master axis state can be executed in the enable or disable state.
- After this instruction is executed, if the slave axis does not reach the target velocity (when InGear of this instruction is FALSE) during the process of establishing the electronic gear relationship between the two axes (e.g. when the instruction is executed while the master axis is running). The slave axis accelerates or decelerates in accordance with Acceleration, Deceleration, Jerk, which are set by this instruction. The target velocity of the slave axis is shown in the following equation. The master and slave axes are not synchronized during this process, and the synchronization error of the master and slave axes caused by the acceleration process will not be compensated automatically.
- After this instruction is executed, after the electronic gear relationship between the two axes is established (when the InGear is TRUE), the relationship between the axis velocity, acceleration, deceleration, and master axis velocity, acceleration, and deceleration is as follows:

$$\text{the slave axis target velocity} = \text{the master1 axis velocity} \times \frac{\textbf{Gear ratio numerator}}{\textbf{Gear ratio denominator}} \pm \text{the master2 axis velocity} \times \frac{\textbf{Gear ratio numerator}}{\textbf{Gear ratio denominator}}$$

$$\text{the slave axis acceleration(deceleration)} = \text{the master1 axis acceleration(deceleration)} \times \frac{\textbf{Gear ratio numerator}}{\textbf{Gear ratio denominator}} \pm \text{the master1 axis acceleration(deceleration)} \times \frac{\textbf{Gear ratio numerator}}{\textbf{Gear ratio denominator}}$$

- After this instruction is executed, if user want to decouple the slave axis from the master axis, only decouple it by executing the MC_Stop instruction for the slave axis.

Re-execute the instruction

After this instruction has been executed, it cannot be re-executed. If user want to change the input variables of this instruction, aborted the executing instruction by executing another instruction.

Execute this instruction while other instructions are in processing

This instruction is executed when other instructions are activated, and how this instruction is buffered to other motion instructions is determined by the value of the BufferMode input variable for this instruction. The only values that can be set for the instruction input variable BufferMode are mcAborting or mcBuffered.

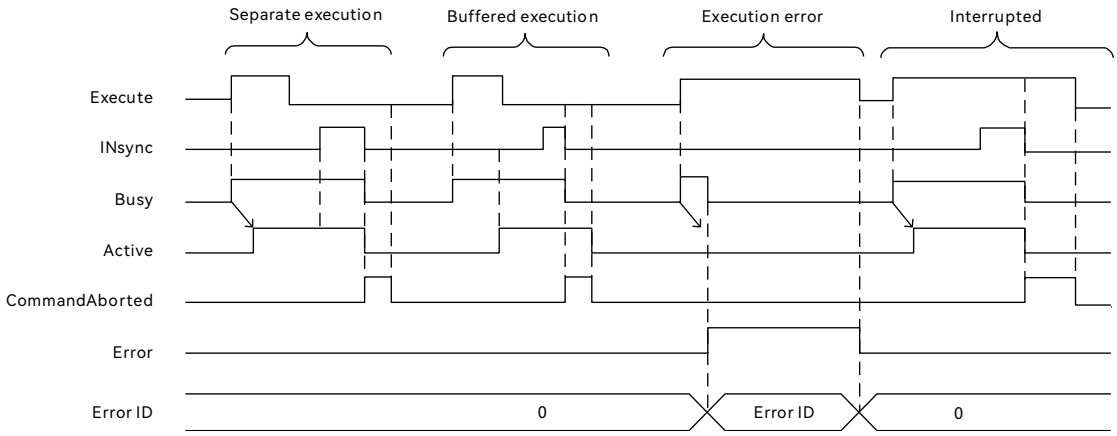
Execute this instruction while other instructions are in processing

Other instructions are executed when this instruction is activated, other motion instructions and this instruction how to buffered is determined by the value of BufferMode of other motion instructions, the value of BufferMode of other motion instructions can only be selected as interrupt or wait; if there is no BufferMode parameter of other motion instructions, it is generally interrupting this instruction. When other motion instructions and this instruction are cached, the timing of execution of other instructions is when the instruction InSync becomes TRUE.

- **Abnormality elimination**

When this instruction is executed, if the input variable is illegal, the instruction Error becomes TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g. axis alarm), the instruction will also report error and the axis will stop immediately.

The timing chart of output variables Description



- **Separate execution**

When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Active changes to TRUE for the next period. When the slave axis command velocity reaches the target velocity, InGear changes to TRUE, and Busy and Active remain TRUE.

- **Relay execution**

The instruction is executed when the other instruction controls the axis (the value of BufferMode is not mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction completes.

- **Error execution**

When the value of the input variable of this instruction is not within the allowable range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next period while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that you can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to zero.

- **Execution abortion**

When the execution of this instruction is aborted by other instructions, the instruction CommandAborted becomes TRUE , Busy, Active and InGear become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

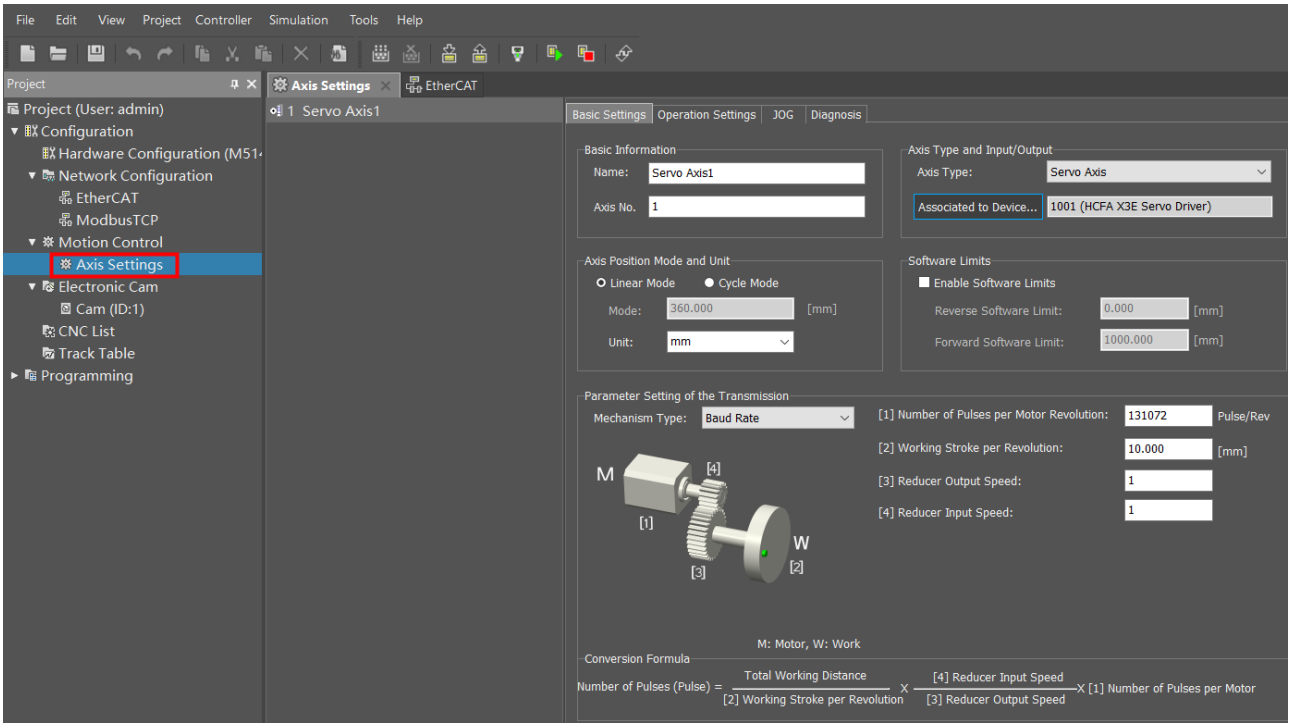
The example program is shown below

- **Functionality**

  The velocity of the two masters are superimposed on the slave axis, and the slave axis velocity is the superimposed sum of the velocity of the two masters.

- **Axis parameter setting**

  The axis parameters for mater axis 1 , mater axis 2 and slave axis 3 are the same, and the axis parameter settings for axis 1 are shown below.

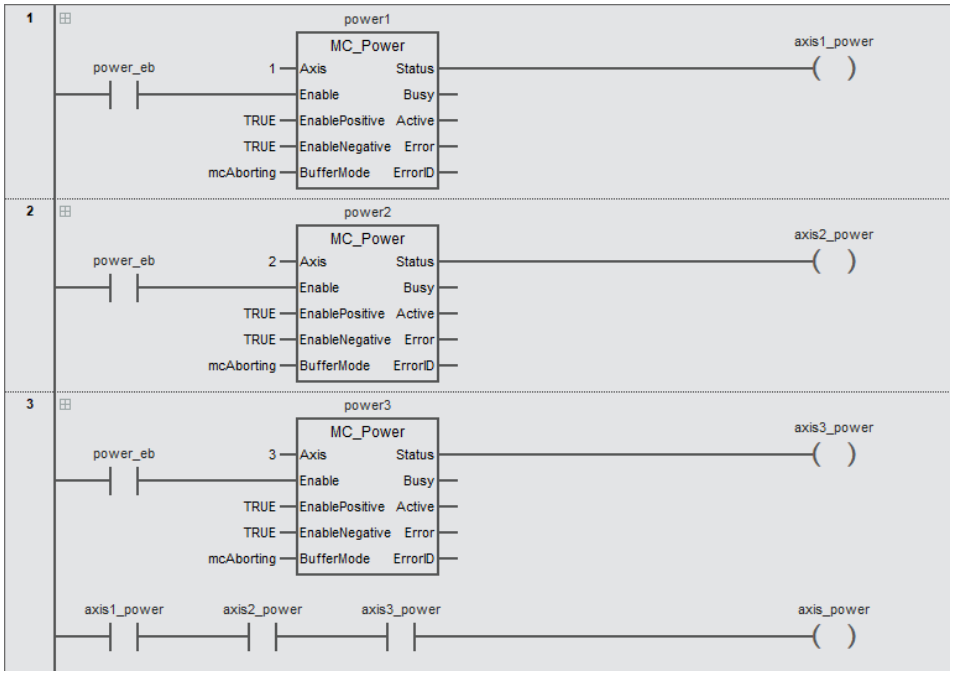


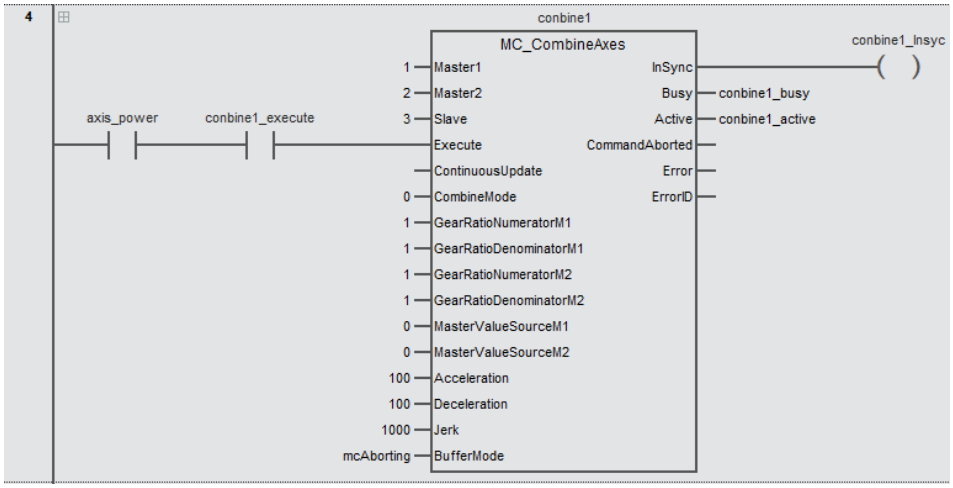
- **Variable table**

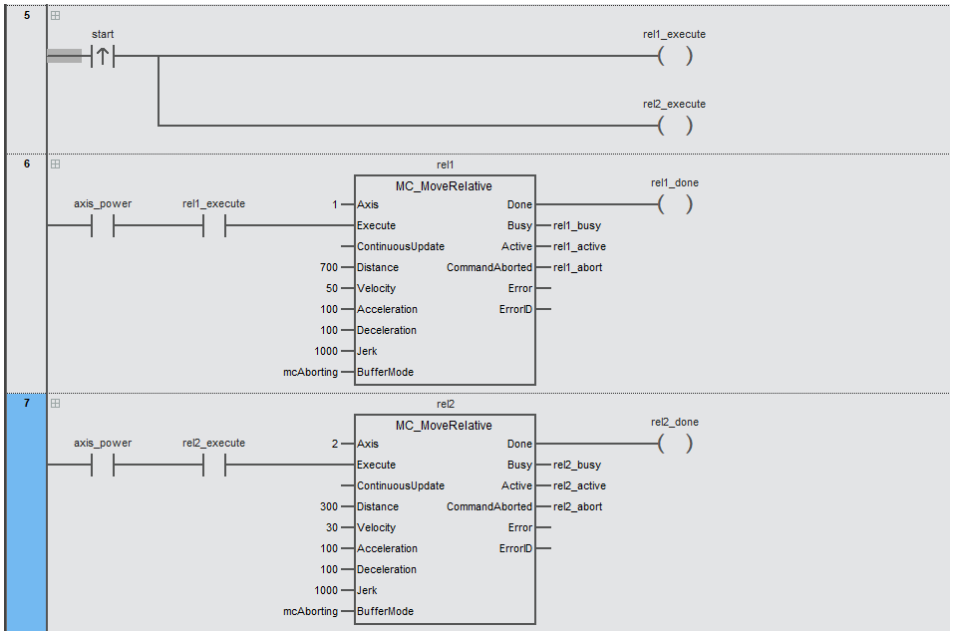
Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	power2		MC_Power		
VAR	axis2_power		BOOL		
VAR	power3		MC_Power		
VAR	axis3_power		BOOL		
VAR	axis_power		BOOL		
VAR	conbin1_execute		BOOL		
VAR	conbin1		MC_CombineAxes		
VAR	conbin1_Insys		BOOL		
VAR	conbin1_busy		BOOL		
VAR	conbin1_ative		BOOL		
VAR	conbin1_abort		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_execute		BOOL		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		
VAR	rel2		MC_MoveRelative		

VAR	rel2_execute		BOOL		
VAR	rel2_done		BOOL		
VAR	rel2_busy		BOOL		
VAR	rel2_active		BOOL		
VAR	rel2_abort		BOOL		
VAR	start		BOOL		

- **LD**







- ST

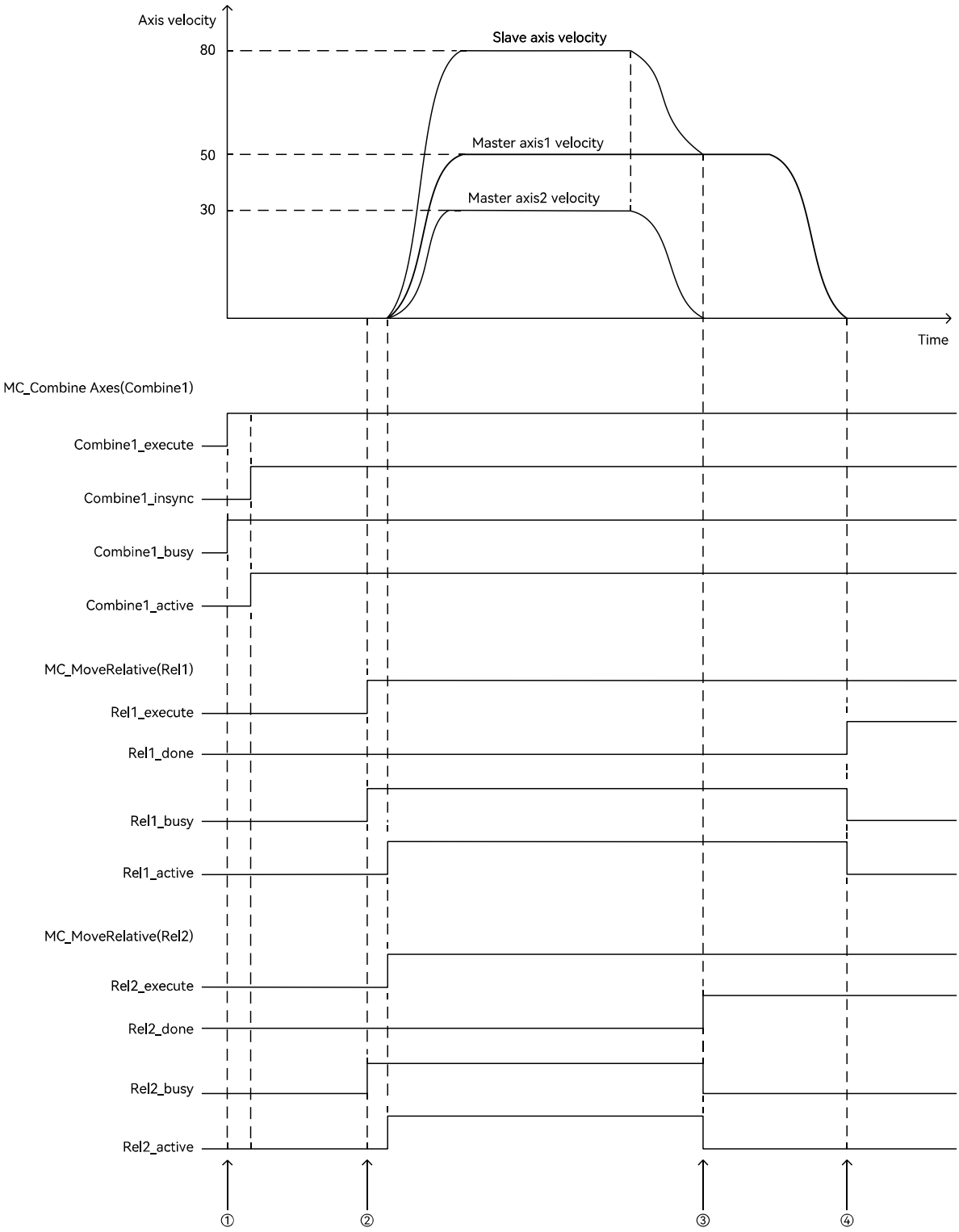
```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );


power2(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );


power3(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );

axis_power:=axis1_power AND axis2_power AND axis3_power;
conbine1(
    Master1:=1 ,
    Master2:=2 ,
    Slave:=3 ,
    Execute:=axis_power AND conbine1_execute ,
```

```
        CombineMode:=0 ,
        GearRatioNumeratorM1:=1 ,
        GearRatioDenominatorM1:=1 ,
        GearRatioNumeratorM2:=1 ,
        GearRatioDenominatorM2:=1 ,
        MasterValueSourceM1:=0 ,
        MasterValueSourceM2:=0 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        InSync=>conbine1_Insyc ,
        Busy=>conbine1_busy ,
        Active=>conbine1_busy ,
        CommandAborted=>conbine1_abort
        );
    rel1_execute:=EDGEPOS(start) ;
    rel2_execute:=EDGEPOS(start) ;

    rel1(
        Axis:=1 ,
        Execute:=axis_power AND rel1_execute ,
        Distance:=700 ,
        Velocity:=50 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:=mcAborting ,
        Done=>rel1_done ,
        Busy=>rel1_busy ,
        Active=>rel1_active ,
        CommandAborted=>rel1_abort
        );


    rel2(
        Axis:=2 ,
        Execute:=axis_power AND rel1_execute ,
        Distance:=300 ,
        Velocity:=30 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:= mcAborting,
        Done=>rel2_done ,
        Busy=>rel2_busy ,
        Active=>rel2_active ,
        CommandAborted=>rel2_abort
        );
```

- **Program description**
  - When conbine1_execute becomes TRUE after axis enable, the double master electronic gear instruction (conbine1) starts execution, and in the 2nd period, an electronic gear relationship with a gear ratio of 1:1 is established between the slave axis 3 and the master axis 1 and master axis 2.
  - After a gear relationship is established between the slave axis and the master axis, both master axes

execute relative displacement commands at the same time, and the slave axis velocity is the sum of the two main axis velocity.

- After the execution of the relative displacement instruction of the master axis 2 is completed, the master axis 1 is still executing the relative displacement instruction, and the slave axis 3 follows the master axis 1, and the velocity of the slave axis 3 and the master axis 1 are the same.
- The master axis 1 relative displacement instruction stops when its execution is completed, and the slave axis 3 follows the master axis 1 to stop.



① The dual master axis electronic gear instruction (combine1) begins execution, and the 2nd cycle establishes a gear relationship between the slave axis and the two master axes.

② Master axis 1 and master axis 2 execute the relative instruction at the same time. After the two instructions control the axes, the positions generated by each of the two instructions are superimposed and act on the slave axis.

③ After the master axis 2 relative instruction is executed, the master axis 1 relative instruction is still being executed, and the slave axis follows the master axis 1 in an electronic gear relationship.

④ After the execution of the master axis 1 relative instruction is completed, the master axis 1 stops and the slave axis follows the master axis to stop.

5.3 MC_GearOut（Decoupling operation）

This instruction stops operation for the MC_GearIn (Start Gear Operation) instruction, the electronic gear relationship between the slave axis and the master axis is also decoupled. Library：MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_GearOut	End gear operation	FB	MC_GearOut _Instance MC_GearOut Slave Execute Done Busy Active CommandAborted Error ErrorID	MC_ GearOut _Instance (Slave :=parameter, Execute :=parameter, Done=> parameter , Busy=> parameter , CommandAborted=> parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Input Variable	Meaning	Data type	Valid range	Default	Description
Slave	Slave axis number	USINT	Depends on model	Required field	Specify slave axis number in the electronic gear.
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected.

■ Output Variables

Output variable	Meaning	Data type	Default	Description
Done	End gear operation	BOOL	TRUE or FALSE	The electronic gear relationship between the slave axis and the master axis is also disengaged.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the axis is being controlled.
CommandAborted	Controlling	BOOL	TRUE or FALSE	TRUE when the instruction is aborted.
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Done	The electronic gear relationship between the slave axis and the master axis is also disengaged.	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
Busy	When Execute changes to TRUE.	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
Active	When the instruction is started.	◆ When Error changes to TRUE. ◆ When CommandAborted changes to TRUE.
CommandAborted	When the instruction is aborted	◆ After one period when Execute is FALSE or the instruction is aborted by other instruction.
Error	The input variable of the instruction is out of the allowed range, the execution conditions of the instruction are not satisfied, or an exception is encountered during the execution of the instruction.	◆ When Execute is TRUE and changes to FALSE

■ Instruction descriptions

- **Basic Instruction descriptions**
  - This instruction is used to abort the execution of the MC_GearIn instruction. When this instruction is executed, the electronic gear relationship established between the slave axis and the master axis through MC_GearIn is released, and the slave axis maintains the slave axis speed at the time of the release after the release. If this instruction is executed while the slave axis is running, the slave axis will continue to run after the instruction is executed. If user want to release the electronic gear relationship and want the slave axis to slow down and stop, execute the MC_Halt or MC_Stop instruction for the slave axis.
  - The execution of this instruction has no effect on the master axis movement.
- **Re-execute the instruction**

  This instruction cannot be restarted. Execution of this instruction reports an error when two axes have not established an electronic gear relationship.
- **Execute this instruction while other instructions are in processing**

  This instruction can be executed only when the MC_GearIn instruction is executed; when other motion instructions are executed, execution of this instruction will report an error

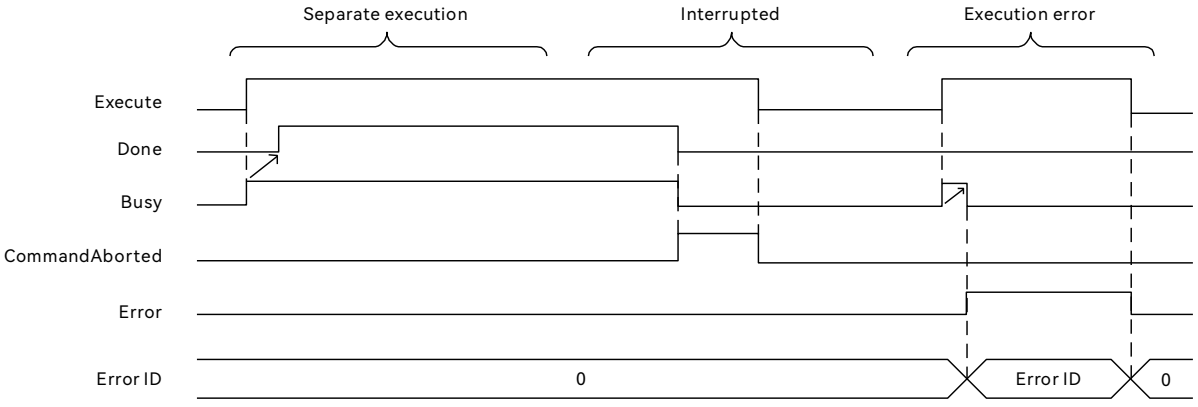
- **Execute this instruction while other instructions are in processing**

Other instructions are executed when this instruction is activated, other motion instructions and this instruction how to buffered is determined by the value of BufferMode of other motion instructions, the value of BufferMode of other motion instructions can only be selected as interrupt or wait; if there is no BufferMode parameter of other motion instructions, it is generally interrupting this instruction. When other motion instructions and this instruction are cached, the timing of execution of other instructions is when the instruction "InGear" becomes TRUE.

- **Abnormality elimination**

When this instruction is executed, if the input variable is illegal, the instruction Error becomes TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g. axis alarm), the instruction will also report error and the axis will stop immediately.

- **The timing chart of output variables description**



- **Separate execution**

When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Done changes to TRUE for the next period. When the slave axis command velocity reaches the target velocity, InGear changes to TRUE, and Busy and Done remain TRUE.

- **Execution abortion**

The instruction is executed when the other instruction controls the axis (the value of BufferMode is not mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction completes.

- **Error execution**

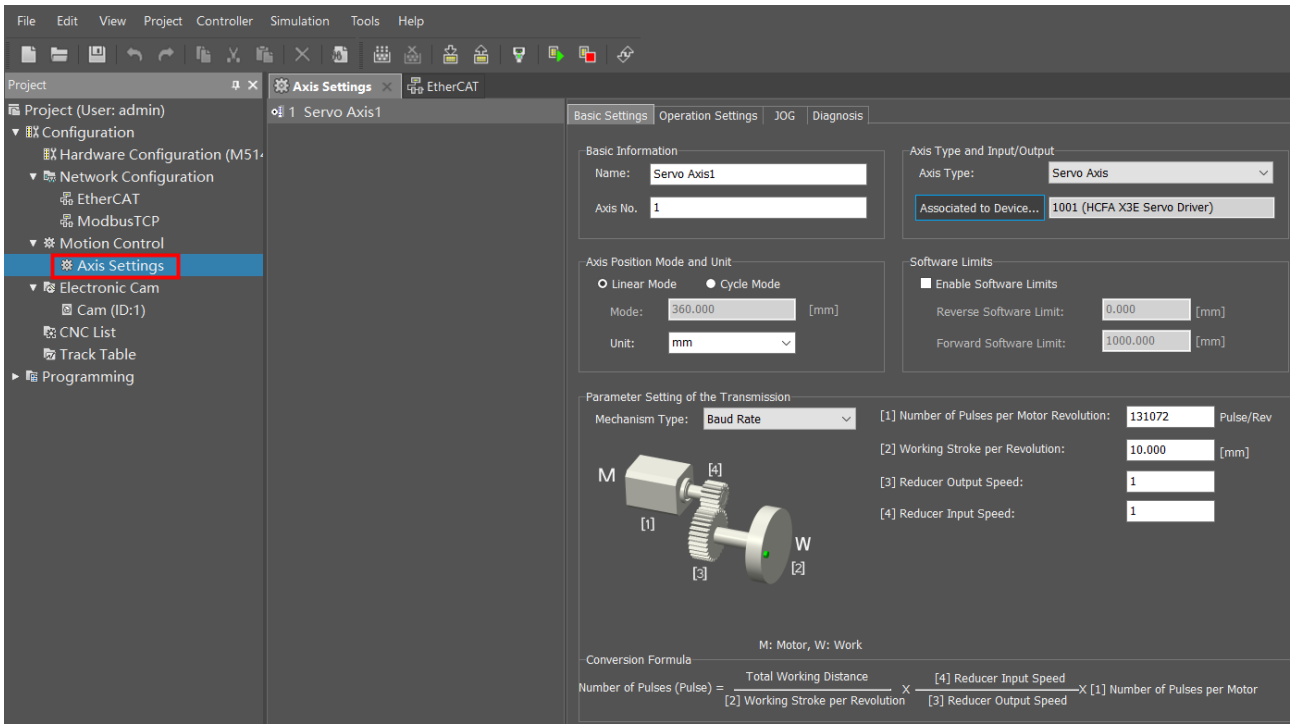
  When the value of the input variable of this instruction is not within the allowable range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next period while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that you can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **The example program is shown below**
  - **Functionality**

    After the electronic gear relationship is established between the slave axis and the master axis, if the electronic gear relationship between the slave axis and the master axis needs to be unlocked and the current speed of the slave axis is maintained to continue operation, this can be realized by executing MC_GearOut (End gear operation).

  - **Axis parameter setting**

    The axis parameters for mater axis 1 and slave axis 2 are the same, and the axis parameter settings for axis 1 are shown below.

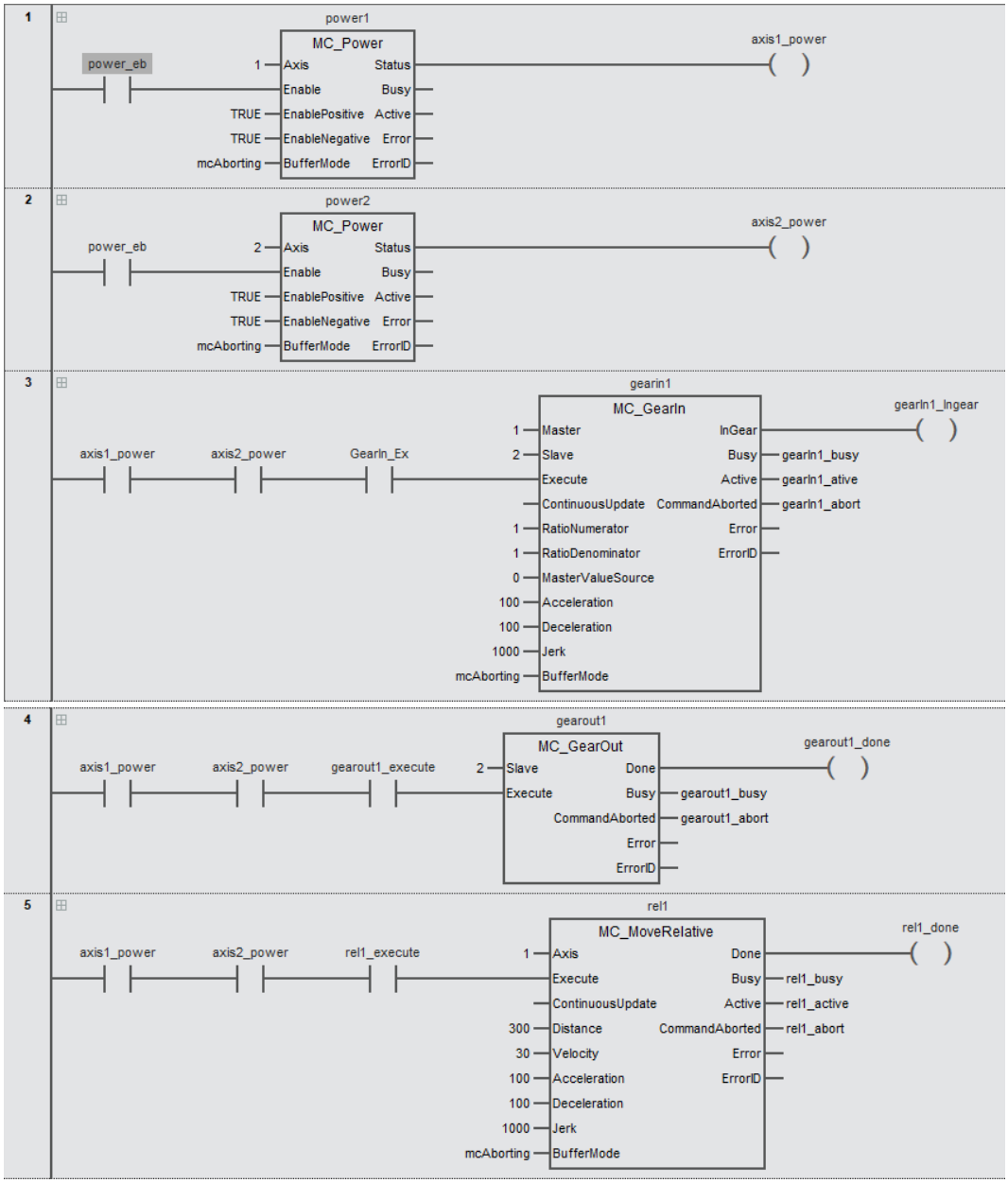


- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	powe1		MC_Power		
VAR	axis1_power		BOOL		
VAR	power2		MC_Power		
VAR	axis2_power		BOOL		
VAR	GearIn_Ex		BOOL		
VAR	gearin1		BOOL		

VAR	gearIn1_Ingear		BOOL		
VAR	gearIn1_busy		BOOL		
VAR	gearIn1_ative		BOOL		
VAR	gearIn1_abort		BOOL		
VAR	gearout1		MC_GearOut		
VAR	gearout1_execute		BOOL		
VAR	gearout1_done		BOOL		
VAR	gearout1_busy		BOOL		
VAR	gearout1_abort		BOOL		
VAR	rel1		MC_MoveRelative		
VAR	rel1_execute		BOOL		
VAR	rel1_done		BOOL		
VAR	rel1_busy		BOOL		
VAR	rel1_active		BOOL		
VAR	rel1_abort		BOOL		

- LD

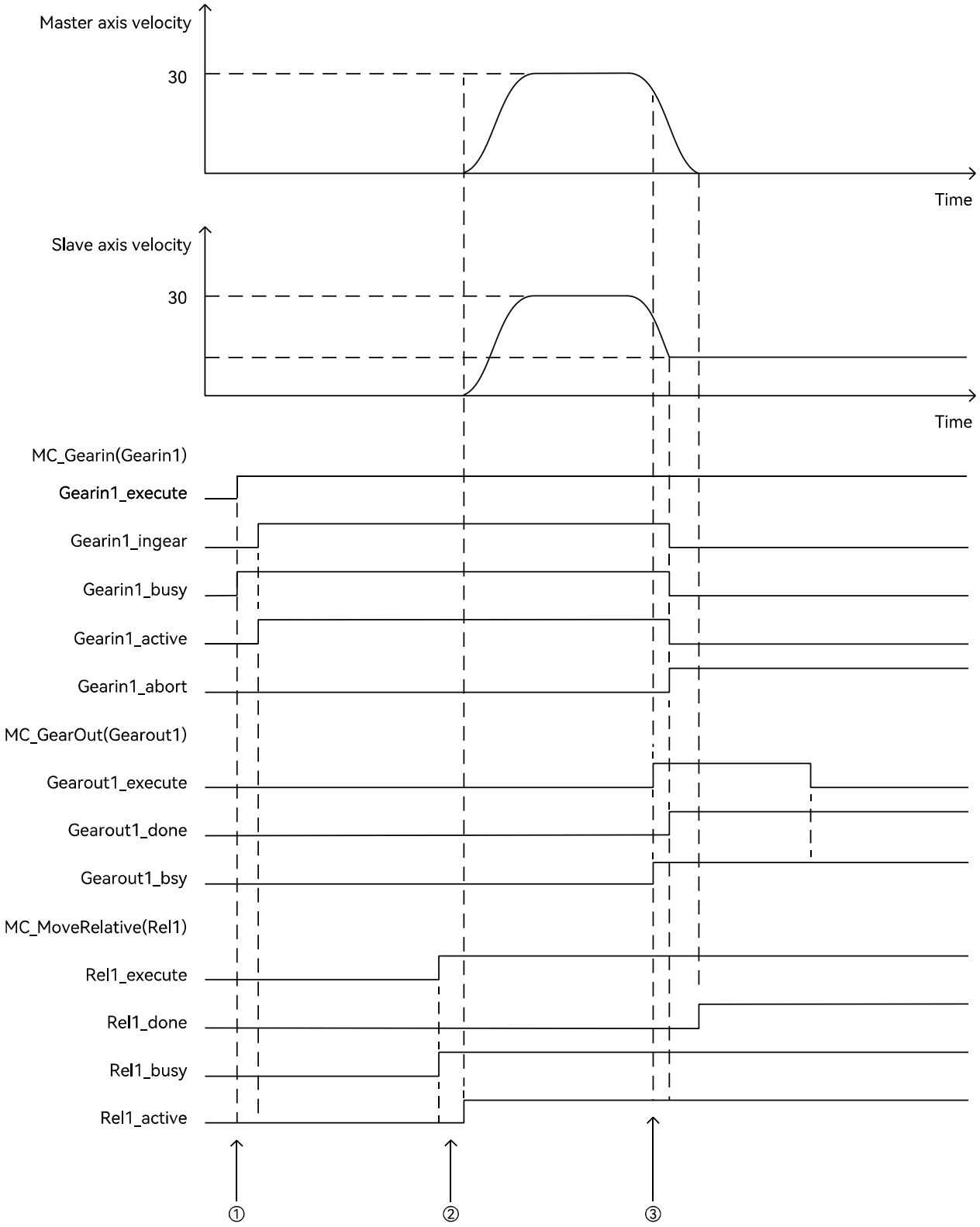


- ST

```
power1 (
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );

power2(
    Axis:=2 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis2_power
    );


gearin1(
    Master:=1 ,
    Slave:=2 ,
    Execute:=axis1_power AND axis2_power AND GearIn_Ex    ,
    RatioNumerator:=1 ,
    RatioDenominator:=1 ,
    MasterValueSource:=0 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcAborting ,
    InGear=>gearIn1_Ingear ,
    Busy=>gearIn1_busy ,
    Active=>gearIn1_busy ,
    CommandAborted=>gearIn1_abort
    );


gearout1(
    Slave:=2 ,
    Execute:=axis1_power AND axis2_power AND gearout1_execute ,
    Done=>gearout1_done ,
    Busy=>gearout1_busy ,
    CommandAborted=>gearout1_abort
    );


rel1(
    Axis:=1 ,
    Execute:=axis1_power AND axis2_power AND rel1_execute ,
    Distance:=300 ,
    Velocity:=30 ,
    Acceleration:=100 ,
    Deceleration:=100 ,
    Jerk:=1000 ,
    BufferMode:=mcAborting ,
```

```
Done=>rel1_done ,
Busy=>rel1_busy ,
Active=>rel1_active ,
CommandAborted=>rel1_abort
);
```

- **Program description**
  - When gearin1_execute becomes TRUE after the axis is enabled, the electronic gear instruction (gearin1) starts to be executed, and after the 2nd period, an electronic gear relationship with a gear ratio of 1:1 is established between the slave axis 2 and the master axis 1.
  - After a gear relationship is established between the slave axis and the master axis, the master axis executes the instruction velocity and the slave axis follows the master.
  - After the slave axis and the mater axis establish the gear relationship, when gearout1_execute becomes TRUE, the electronic gear discouplesment instruction (gearout1) instruction is executed, and in the 2nd period, Done of gearout1 becomes TRUE, the slave axis and the mater axis decouples from the electronic gear relationship, and the slave axis continues to run at the current speed. If you want to decouples the slave axis and the mater axis from the electronic gear relationship and stop, the slave axis can execute the MC_Halt instruction to realize it.



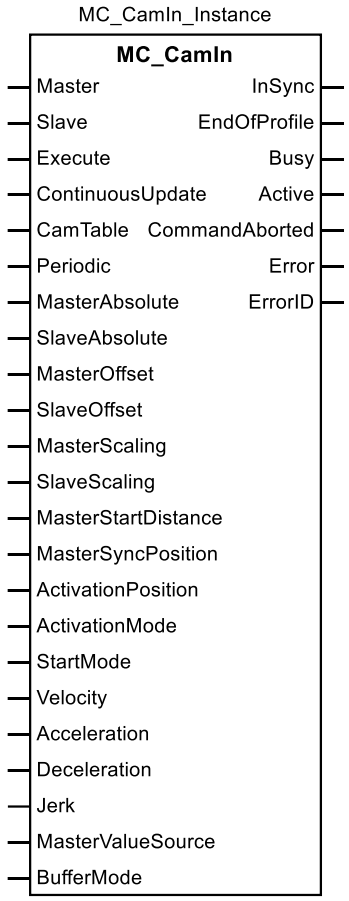
① Gearin instruction (gearin1) starts execution, and the slave axis establishes a gear relationship with the axis.

② After the slave axis and the main axis establish a gear relationship, the main axis executes a velocity command, and the slave axis follows the main axis.

③ Gearout1 is executed and the slave axis continues to run at the current velocity.

5.4 MC_CamIn（Cam coupling operation ）

Controls the synchronized movement of the slave axes in accordance with the planned cam relationship of the master axis. Library：MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_CamIn	Start Cam Operation	FB		MC_CamIn_Instance(Maste :=parameter, Slave:=parameter , Execute:=parameter, ContinuousUpdate:=parameter, CamTable:=parameter , Periodic :=parameter, MasterAbsolute:=parameter , SlaveAbsolute:=parameter , MasterOffset :=parameter, SlaveOffset:=parameter , MasterScaling :=parameter, SlaveScaling :=parameter, MasterStartDistance :=parameter, MasterSyncPosition:=parameter , ActivationPosition:=parameter , ActivationMode:=parameter , StartMode:=parameter , Velocity:=parameter , Acceleration :=parameter, Deceleration :=parameter, Jerk:=parameter , MasterValueSource :=parameter, BufferMode:=parameter , InSync => parameter, EndOfProfile=> parameter , Busy=> parameter , Active => parameter, CommandAborted=> parameter , Error=> parameter , ErrorID=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Master	Master axis number	USINT	Depends on model	Required field	Specify master axis number in the electronic cam.
Slave	Slave axis number	USINT	Depends on model	Required field	Specify slave axis number in the electronic cam
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected.
ContinuousUpdate	continually updated	BOOL	TRUE or FALSE	FALSE	Reserve
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Setting the electronic cam relationship table number
Periodic	Periodic mode	BOOL	TRUE or FALSE	FALSE	Specify whether to execute the specified cam table periodically or only once. TRUE: Periodic FALSE: NonReserveperiodic
MasterAbsolute	Master axis absolute position	BOOL	TRUE or FALSE	FALSE	Sets the correspondence between the master axis position and the cam phase.

					TRUE: Absolute position FALSE: Relative position
SlaveAbsolute	Slave axis absolute position	BOOL	TRUE or FALSE	FALSE	Sets the correspondence between the slave axis position and the cam phase. TRUE: Absolute position FALSE: Relative position
MasterOffset	Master axis offset	LREAL	Positive or negative number,0	0	The phase of the master axis is shifted by using the specified offset value.
SlaveOffset	slave axis offset	LREAL	Positive or negative number,0	0	The displacement of the slave axis is shifted by using the specified offset value.
MasterScaling	Master axis coefficient	LREAL	Positive number	0	The phase of the master axis is extended
SlaveScaling	Slave axis coefficient	LREAL	Positive or negative number	Required field	The displacement of the slave axis is extended
MasterStartDistance	Reserve	Reserve	-	Reserve	Reserve
MasterSyncPosition	Reserve	Reserve	-	Reserve	Reserve
ActivationPosition	CamReservecoupled start position	LREAL	Positive or negative number,0	0	Set the start position of the cam coupling action, when the master passes through this position, the cam coupling will start formally.
ActivationMode	Type of startup position	MC_ACTIVATION_MODE	0：mcRelative 1：mcAbsolute 2：mcPhaseAxis 3：mcPhaseCAM	0	Sets the type of ActivationPosition. 0: Relative position 1: Absolute position 2: Axis phase 3:Cam phase
StartMode	Start Mode	MC_START_MODE	Reserve1、0、1	0	Sets the mode of the coupling action. Reserve1(mcRampInNegative): Inverse 0(mcRampInShortest): shortest distance 1(mcRampInPositive):Positive
Velocity	Coupling compensation velocity	LREAL	Positive number	0	Setting the upper limit value of the compensation velocity during the coupling process.*1 (The unit :travel units/s^2) *2
Acceleration	Coupling compensation acceleration	LREAL	Positive number	0	Setting the upper limit value of the compensation acceleration during the coupling process.*1 (The unit :travel units/s^2) *2
Deceleration	Coupling compensation deceleration	LREAL	Positive number	0	Setting the upper limit value of the compensation deceleration during the coupling process.*1 (The unit :travel units/s^2) *2
Jerk	Jerk	LREAL	Positive number	0	Reserve
MasterValueSource	The source position of master axis	MC_SOURCE	0：mcSetValue 1：mcActualValue	0	Sets the source of the master position. 0: Commanded position 1: Actual position
BufferMode	Buffer mode	MC_Buffer_Mode	0：mcAborting 1：mcBuffered	0	Specify the behavior when executing between the two motion instruction. *3 0: Aborting 1: Buffered

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "Parameter description of motion control instructions ".

*2: For details of the instruction units, please refer to "Parameter unit of motion control instructions".

*3: For details of BufferMode, please refer to "Buffer mode during multiReservestarting of the same axis”.

■ Output variable

Output variable	Meaning	Data type	Default	Description
InSync	Cam synchronization	BOOL	TRUE or FALSE	The slave axis is in cam synchronization with the master axis
EndOfProfile	End of cam cycle	BOOL	TRUE or FALSE	The cam table end point is executed.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Active	Controlling	BOOL	TRUE or FALSE	TRUE when the axis is being controlled.
CommandAborted	Command aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted.
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
InSync	When the slave axis starts cam operation.	◆ When CommandAborted changes to TRUE; ◆ When Error changes to TRUE; ◆ When Periodic changes to FALSE; ◆ When EndOfProfile changes to TRUE.
EndOfProfile	The period where the phase and displacement of the end point of the cam table are output as the command position.	◆ One period after EndOfProfile changes to TRUE
Busy	When Execute changes to TRUE.	◆ When CommandAborted changes to TRUE; ◆ When Error changes to TRUE; ◆ When Periodic changes to FALSE; ◆ When EndOfProfile changes to TRUE.
Active	When the instruction is started.	◆ When CommandAborted changes to TRUE; ◆ When Error changes to TRUE; ◆ When Periodic changes to FALSE; ◆ When EndOfProfile changes to TRUE.
CommandAborted	When the instruction is aborted	◆ After one period when Execute is FALSE or the instruction is aborted by other instruction.
Error	The input variable of the instruction is out of the allowed range, the execution conditions of the instruction are not satisfied, or an exception is encountered during the execution of the instruction.	◆ When Execute changes from TRUE to FALSE ◆ The next program cycle after Execute changes from TRUE to FALSE

■ Function description

- **Basic function descriptions**

The MC_CamIn instruction executes a cam motion that synchronizes the master axis phase and slave axis displacement according to a cam table. This instruction only refers to the master axis position but does not control the master axis, so the master axis can be any type of axis, but the slave axis is only allowed to be servo axis and virtual servo axis.

- **Instruction execution process**

The execution process of this instruction consists of three phases: waiting for coupling start, coupling in progress, and synchronization in progress.

Waiting for coupling to start: the instruction is triggered to execute, but the master axis has not yet passed the specified coupling start position.

Coupling in progress: the slave axis is catching up with the master axis, but has not yet achieved cam synchronization with the master axis.

In synchronization: the slave axis is camReservesynchronized with the master axis and continues to run in synchronization.

- **Coupling start position setting**

In the process of practical application, it is often encountered that it is necessary to start cam coupling at a specific position (phase, angle) of the master, and when the master does not pass through this specific position, the slave axis does not move and is in a waiting state. In the face of this demand, can be realized by setting the coupling start position. The coupling start position means that when the master passes through the position, the coupling action is officially started, after which the slave axis starts to catch up with the master.

The coupling start position is set by the parameter ActivationPosition and ActivationMode combined, the parameter ActivationPosition sets the position and ActivationMode sets the type of the position. There are four types of coupling start positions: relative position, absolute position, axis phase, and cam phase.

ActivationMode=0 (relative position): the coupling activation position is defined as the axis position of the master, in this case, the coupling activation position is the position of the master at the moment of the instruction triggering execution (at the moment of the rising edge of the Execute) plus the ActivationPosition. e.g., the position of the master at the moment of the instruction triggering execution is 1000. For example, if the master axis position at the moment of instruction trigger execution is 1000 and the value of ActivationPosition is 5000, then the coupling start position is 6000 (6000=1000+5000).

ActivationMode=1 (absolute position): the coupling start position is defined as the axis position of the master; in this case the coupling start position is ActivationPosition. e.g. if the master position at the moment of instruction triggering is 1000 and the value of ActivationPosition is 5000, then the coupling start position is 5000.

ActivationMode=2 (axis phase): the coupling start position is defined as the axis phase of the master. For example, the mode of the master is 360 (can be set in the software axis parameter setting interface), the master position at the moment of instruction triggering execution is 500, at this time, the axis phase of the master is 140 (140=500%360), if the ActivationPosition is set to 100, then the coupling start position is 820 (820%360=100) or 460 (460%360=100). 360=100). The coupling action can be started when the master forward motion passes through 820 or reverse motion passes through 460.

ActivationMode=3 (cam phase): The coupling start position is defined as the absolute cam phase of the master. For example, the cam curve planning master cam period is 1000, the master position at the moment of instruction triggering execution is 1500, at this time, the absolute cam phase of the master is 500 (500=1500%1000), if the ActivationPosition is set to 100, then the coupling start position is 2100 (2100%1000=100) or 1100 (1100%1000=100). 1100%1000=100). The coupling action can be initiated when the master forward motion passes through 2100 or reverse motion passes through 1100.

Note: The master may pass through the coupling start position several times, but it is only valid when the coupling start position is encountered for the first time after the instruction is triggered for execution; thereafter the coupling start position will no longer function.

- **Correspondence between axis position and cam phase**

The cam relationship planned in the cam curve is only the correspondence between the master and slave axes in one cam cycle, but during cam execution, it may be necessary to continue for more than one cam cycle, so the synchronization relationship planned in the cam curve is actually the cam phase correspondence between the master and slave axes. The axis positions of the masterReserveslave axes are

linear, and when establishing the cam synchronization relationship, it is first necessary to determine the correspondence between the axis positions and the cam phases.

The parameter MasterAbsolute is used to set the correspondence between the master axis position and the master axis cam phase, and the parameter SlaveAbsolute is used to set the correspondence between the slave axis position and the slave axis cam phase.

MasterAbsolute=FALSE (master axis relative mode): the position of the master axis at the moment of coupling start corresponds to a cam phase of 0 (starting point of the cam curve). For example, if the master position at the moment of coupling start is 1500, the correspondence between the master axis position and its cam phase is based on the fact that the axis position 1500 corresponds to the cam phase 0, and then the cam will be executed from phase 0 (starting point).

MasterAbsolute=TRUE (master absolute mode): The reference for the correspondence between the master axis position and its cam phase is: axis position 0 corresponds to cam phase 0. For example, if the master axis position at the start of coupling is 1500, and the master period in the cam relationship is 1000, then the cam phase of the master axis at the time of the start of coupling is 500, and the cams will start executing from the position where the phase is 500. position 500, then the cam will start executing from the phase position 500.

SlaveAbsolute=FALSE (slave reserve axis relative mode): the cam phase of the slave axis at the moment of coupling start is in accordance with the cam synchronization relation with the cam phase of the main axis. At the moment of coupling start, no matter what position the slave axis is in, its cam phase meets the synchronization relationship with the cam phase of the main axis at that moment. For example, at the moment of coupling start, the cam phase of the main axis is 500, and according to the cam relationship, the corresponding cam phase of the slave axis is 100, if the axis position of the slave axis is 0 at that moment, then the corresponding relationship between the axis position of the slave axis and its cam phase is as follows: the axis position 100 corresponds to the cam phase 0.

SlaveAbsolute=TRUE (slave absolute mode): the correspondence between the slave axis position and its cam phase is based on the following: axis position 0 corresponds to cam phase 0. For example, if the slave axis position at the moment of coupling start is 100, and the slave axis period in the cam relationship is 1000, then the slave axis cam phase at the moment of coupling start is 100.

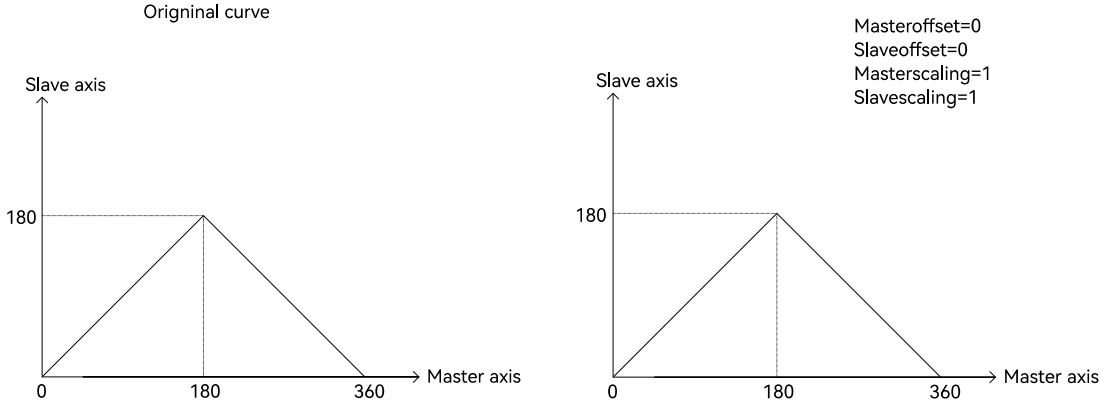
- **Cam curve transformation**

MasterOffset: The original cam relation curve is offset in the direction of the master axis.

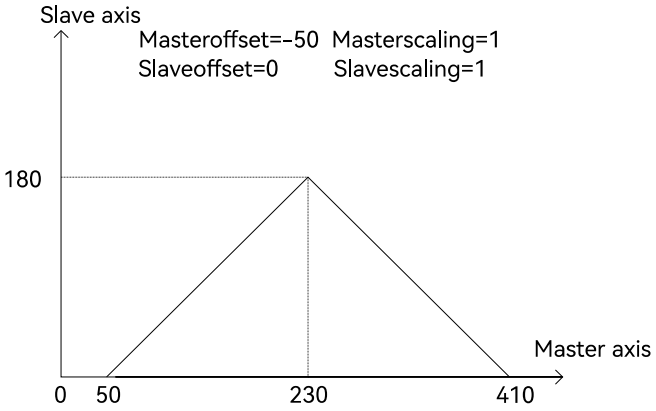
SlaveOffset: The original cam relation curve is offset in the direction of the slave axis (invalid when the slave axis is in relative mode).

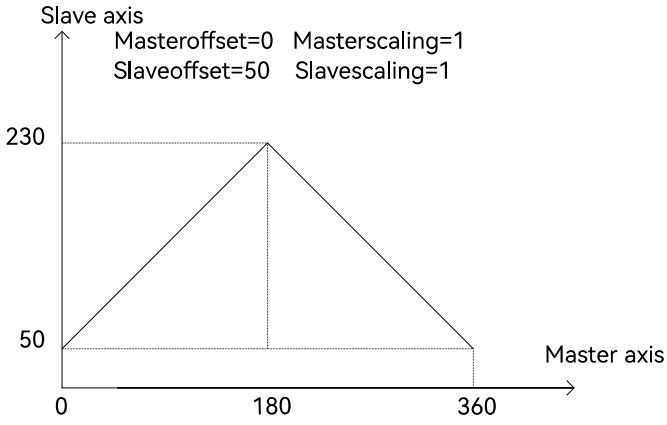
MasterScaling: The original cam relation curve is scaled in the direction of the master axis.

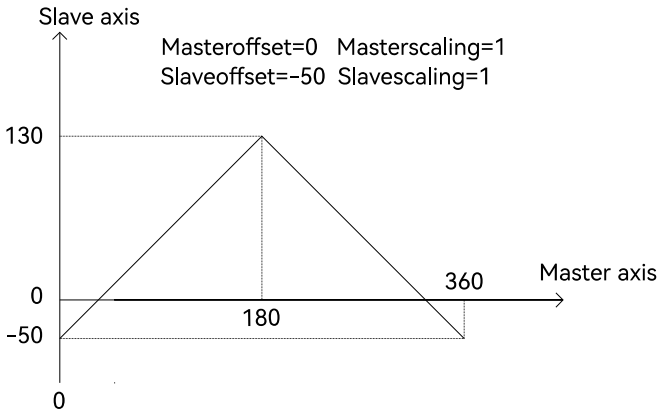
SlaveScaling: The original cam relation curve is scaled in the direction of slave axis.

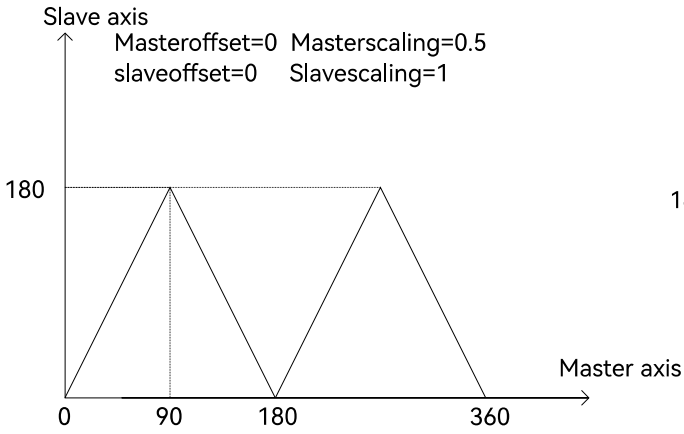


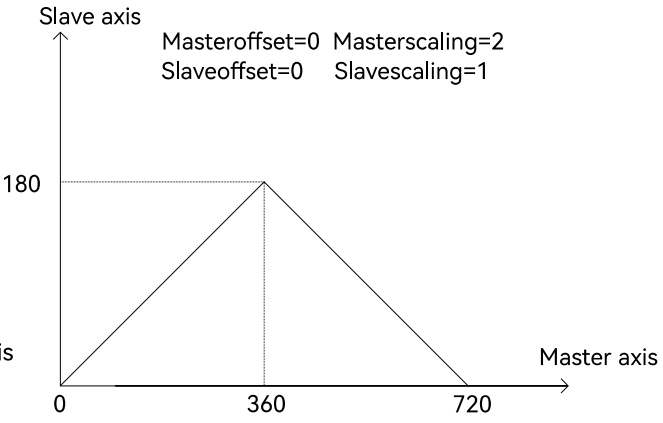

Slave axis
Masteroffset=50 Masterscaling=1
Slaveoffset=0 Slavescaling=1
180
Master axis
-50 0 130 310

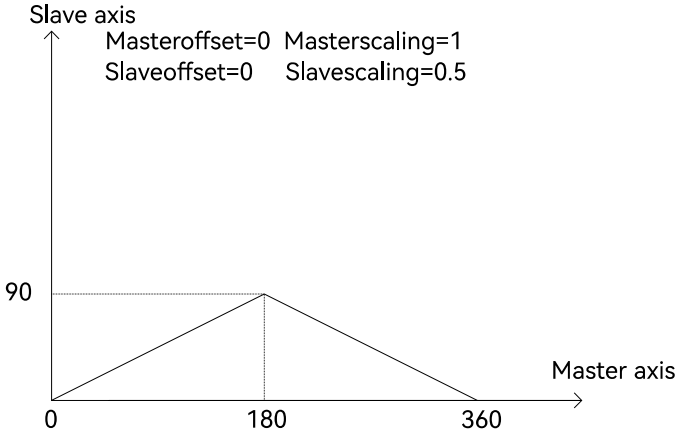

Slave axis
Masteroffset=-50 Masterscaling=1
Slaveoffset=0 Slavescaling=1
180
Master axis
0 50 230 410

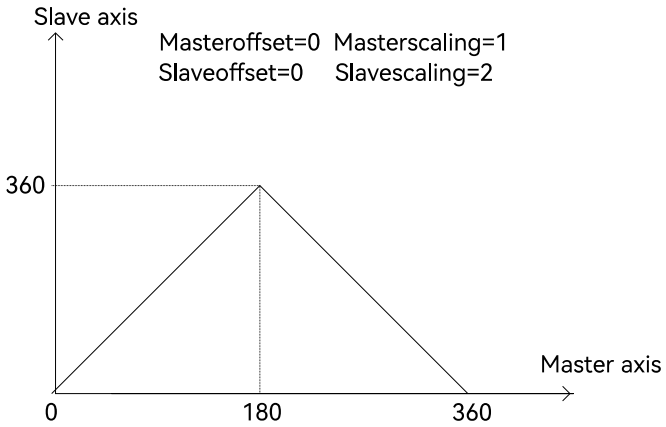

Slave axis
Masteroffset=0 Masterscaling=1
Slaveoffset=50 Slavescaling=1
230
50
Master axis
0 180 360


Slave axis
Masteroffset=0 Masterscaling=1
Slaveoffset=-50 Slavescaling=1
130
0
-50
0
360
180
Master axis


Slave axis
Masteroffset=0 Masterscaling=0.5
slaveoffset=0 Slavescaling=1
180
Master axis
0 90 180 360


Slave axis
Masteroffset=0 Masterscaling=2
Slaveoffset=0 Slavescaling=1
180
Master axis
0 360 720


Slave axis
Masteroffset=0 Masterscaling=1
Slaveoffset=0 Slavescaling=0.5
90
Master axis
0 180 360


Slave axis
Masteroffset=0 Masterscaling=1
Slaveoffset=0 Slavescaling=2
360
Master axis
0 180 360

- **Mode of coupling action**

  Since cam synchronization is cam phase synchronization, the slave axis can move in the forward direction during coupling (from unsynchronized to synchronized) to the phase that meets the synchronization requirements, or in the reverse direction to the phase that meets the synchronization requirements, which can be set using the parameter StartMode.

  StartMode=0 (shortest distance): Determines the direction of movement of the slave axis in the coupling process according to the principle of the smallest moving distance.

  StartMode=1 (positive): The slave axis moves in the forward direction to the phase that meets the synchronization requirements.

  StartMode=Reserve1 (negative): Reverse movement from the slave axis to the phase that meets the synchronization requirements.

- **Coupling action compensation**

  During the coupling process, the speed of the slave axis catching up with the master axis can be adjusted using the compensation parameters. The larger the values of the compensation parameters Velocity, Acceleration and Deceleration, the faster the slave axes catch up.

- **Periodic execution of cams**

  Periodic=TRUE (periodic execution): the cam will execute in an infinite cycle until the cam synchronization is terminated.

  Periodic=FALSE (nonReserveperiodic execution): the cam is terminated after one cycle of execution (after the end of the cam), after which the slave axis is decoupled from the master and no longer follows the movement of the master.

- **Re-execute the instruction**

  A nonReserveperiodic cam, this instruction can be re-executed when a cam cycle ends (after EndOfProfile becomes TRUE). Periodic cam, the instruction cannot be re-executed, another instruction of this instruction can be executed to interrupt the currently executing instruction.

- **Execute this instruction while other instructions are in processing**

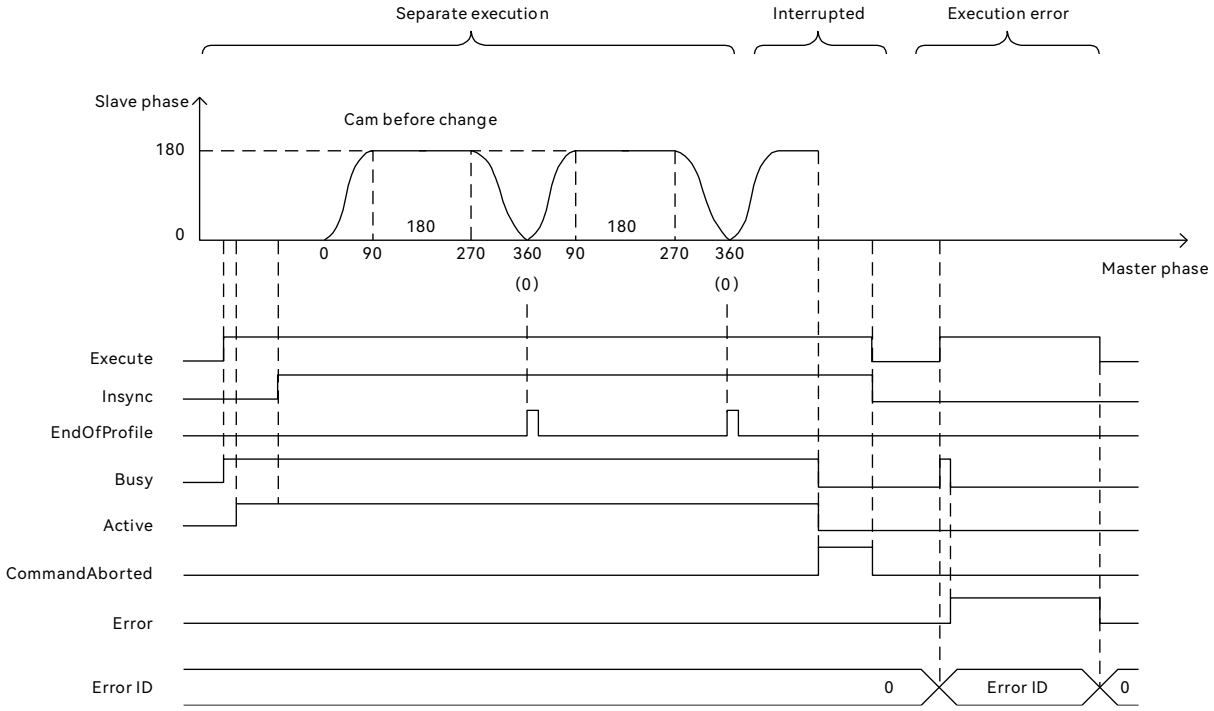
  This instruction is activated when other motion instructions are executed, and how this instruction is buffered to other motion instructions is determined by the value of the BufferMode input variable for this instruction. The only values that can be set for the instruction input variable BufferMode is mcAborting or mcBuffered.

BufferMode	Meaning
mcAborting (Aborted)	Interrupts the command currently controlling the axis and switches to cached command control.
mcBuffered (Buffered)	Waiting for the current instruction to finish execution (e.g., when the target position or target speed is reached), switches to cached instruction control of the axis.

- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, other motion instructions are started, how other motion instructions and this instruction are buffered is determined by the value of the BufferMode pin of the other instruction, and the value of BufferMode of the other motion instruction can only be selected to interrupt or wait; if there is no BufferMode parameter for the other motion instruction, it is usually interrupted. When other motion instructions and this instruction are cached, the timing of other instruction execution is when the instruction EndOfProfile (end of cam cycle) becomes TRUE.

■ Timing description of output variable status



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, Active changes to TRUE in the next cycle, and the InSync bit changes to TRUE when the phases of the slave and master axes conform to the cam relationship. After the execution of this instruction, the InSync bit may change to TRUE at the same time as Active, or it may change to TRUE many cycles after Active changes to TRUE, depending on the input parameters of this instruction. After the execution of this instruction, the InSync bit may become TRUE at the same time as Active, or it may become TRUE many cycles after Active becomes TRUE, depending on the input parameter of the instruction. When Execute changes from TRUE to FALSE, there is no change in the Busy (Executing), Active (Controlling), and InSync (Cam Synchronizing) status bits. At the end of the cam cycle, EndofProfile changes to TRUE for one task cycle.

- **Execution interruption**

  When the execution of this instruction is interrupted by other instructions, the instruction CommandAborted becomes TRUE, and Busy (Execute), Active (Control), and InSync (Cam Synchronization) become FALSE at the same time; when Execute (Execution) becomes FALSE, CommandAborted becomes FALSE at the same time.

- **Execution error**

  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

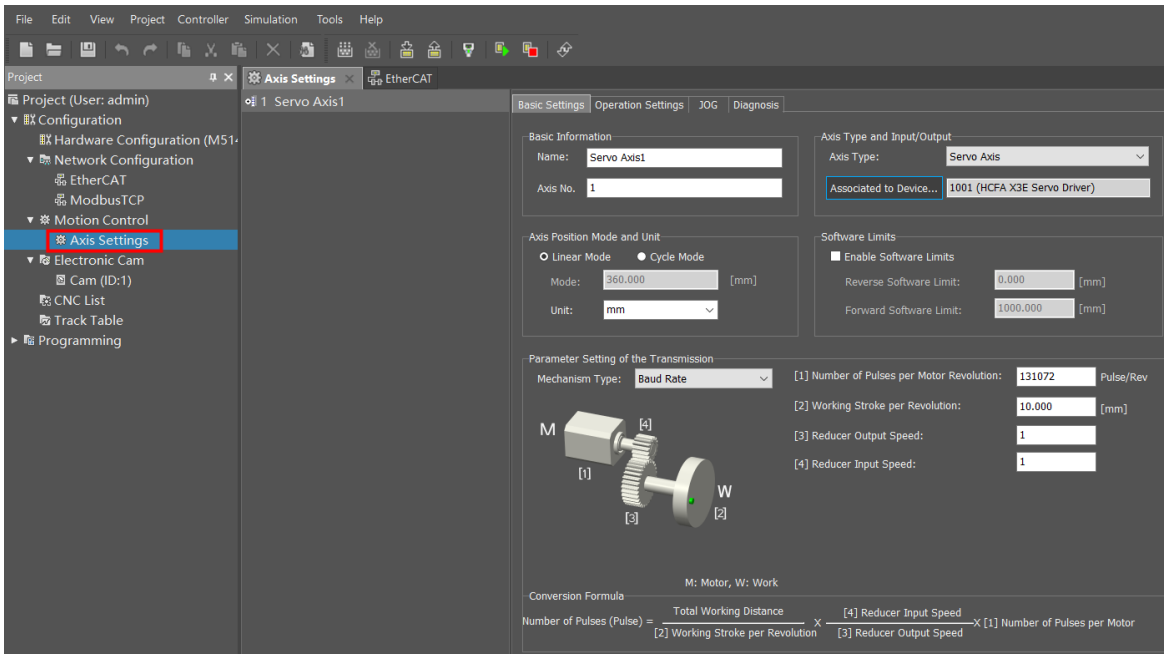
■ The example program is shown below

- **Functionality**

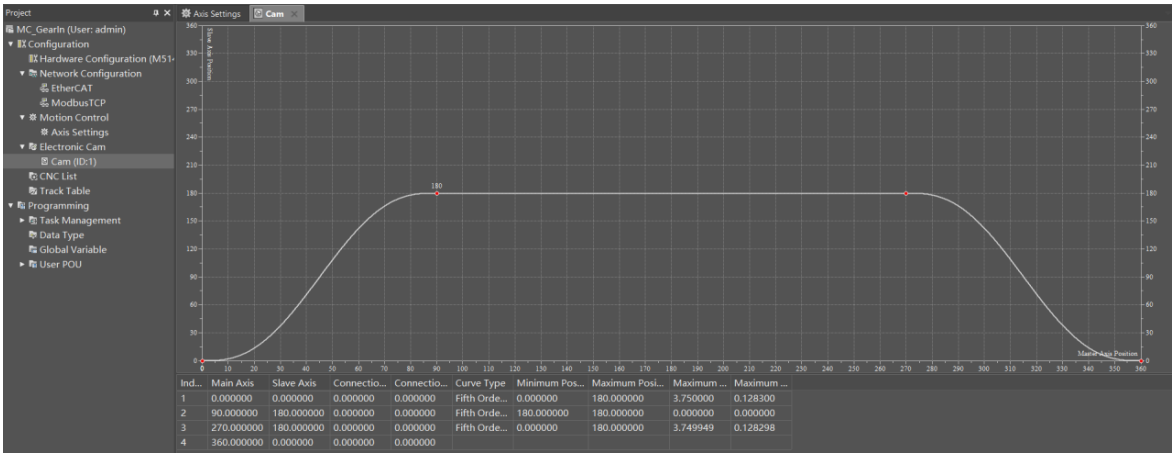
  The two axes establish a cam relationship through the MC_CamIn instruction, and the slave axis follows the master axis according to the cam established in the software. If the slave axis needs to deReservecam with the master axis, the slave axis executes the MC_CamOut instruction to deReservecam.

- **Axis parameters and cam curve setting**

The axis parameters for mater axis 1 and slave axis 2 are the same, and the axis parameter settings for axis 1 are shown below.



The cam curve created in the software is shown below

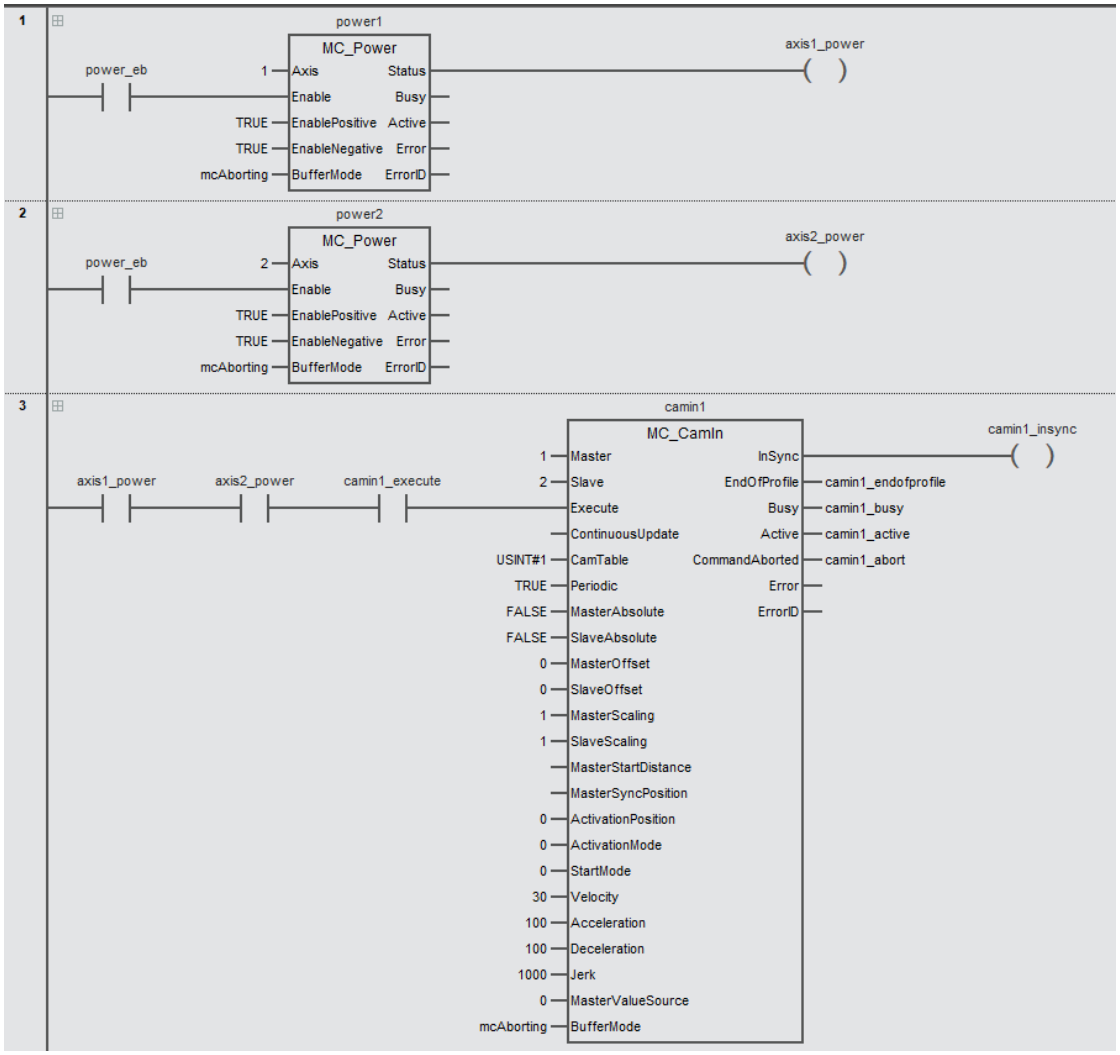


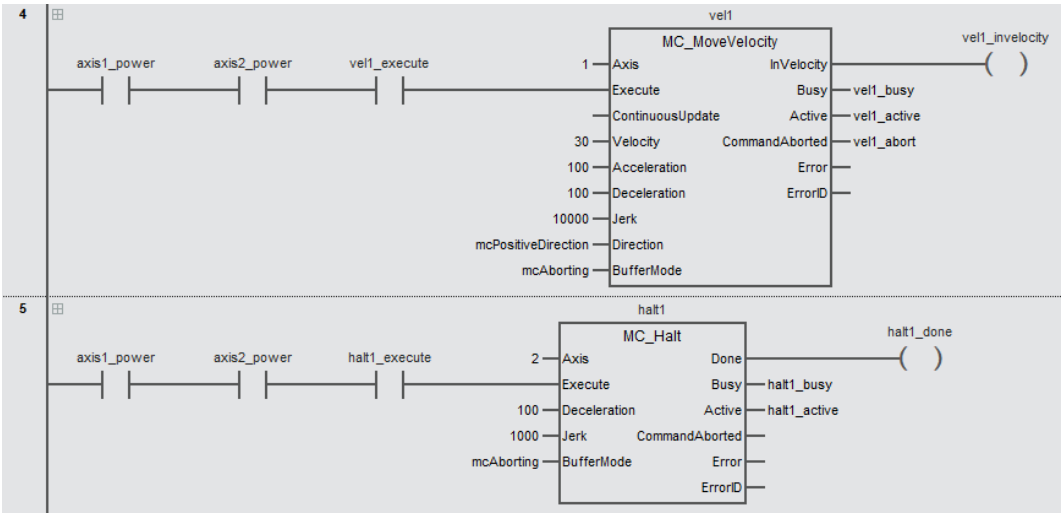
- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	powe1		MC_Power		
VAR	axis1_power		BOOL		
VAR	power2		MC_Power		
VAR	axis2_power		BOOL		
VAR	camin1		MC_CamIn		
VAR	camin1_execute		BOOL		
VAR	camin1_insync		BOOL		
VAR	camin1_busy		BOOL		
VAR	camin1_active		BOOL		
VAR	camin1_abort		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_execute		BOOL		

VAR	vel1_done		BOOL		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	halt1		MC_Halt		
VAR	halt 1_execute		BOOL		
VAR	halt 1_done		BOOL		
VAR	halt 1_busy		BOOL		
VAR	halt 1_active		BOOL		

- LD



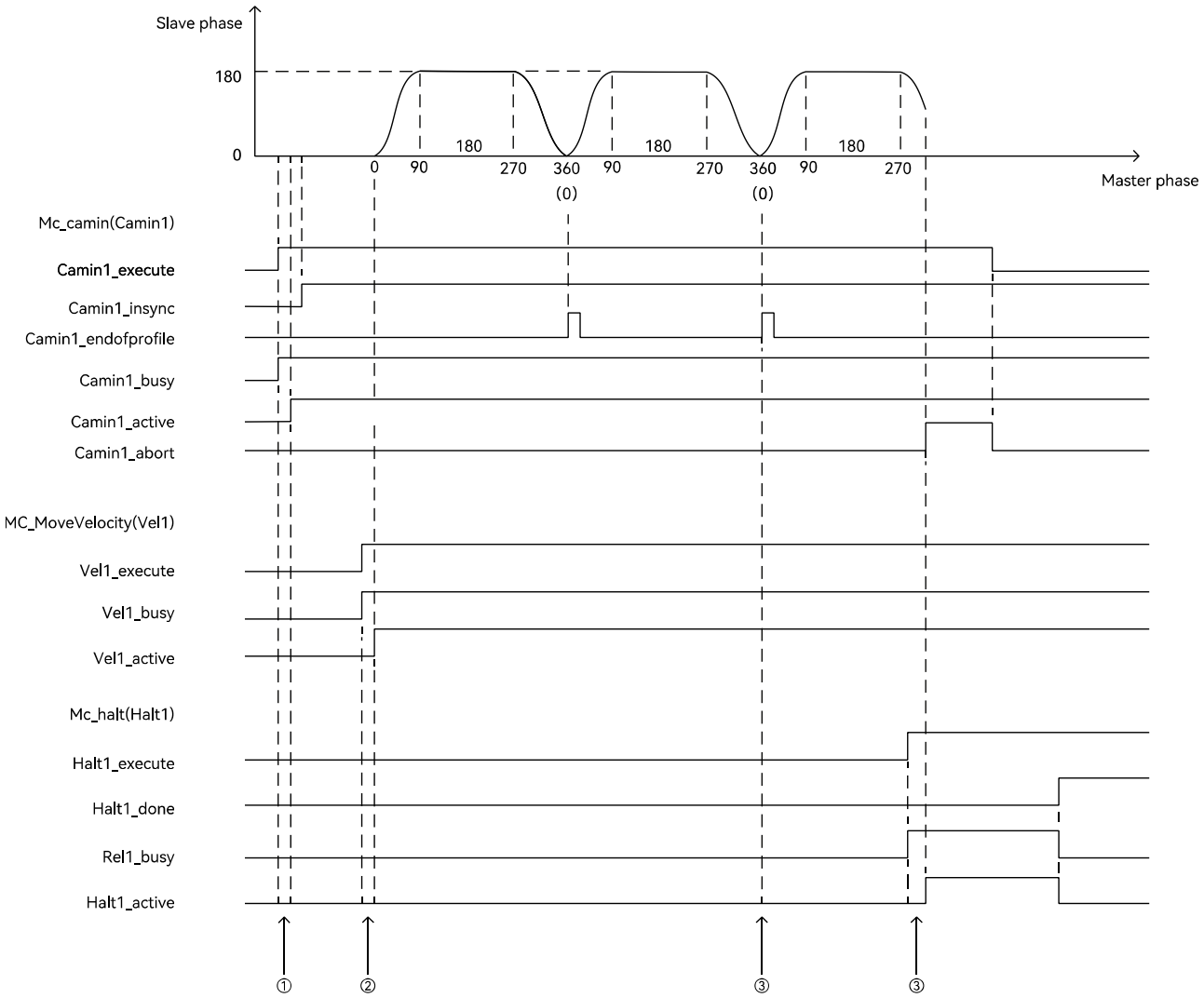


- ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );


power2(
    Axis:=2 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis2_power
    );


camin1(
    Master:=1,
    Slave:=2,
    Execute:= axis1_power AND axis2_power AND camin1_execute ,
    CamTable:= 1,
    Periodic:=TRUE,
    MasterAbsolute:=FALSE,
    SlaveAbsolute:=FALSE,
    MasterOffset:=0,
    SlaveOffset:=0,
    MasterScaling:=1,
    SlaveScaling:=1,
    ActivationPosition:=0,
    ActivationMode:=0,
    StartMode:=0,
    Velocity:=30,
    Acceleration:=100,
    Deceleration:=100,
```

```
        Jerk:=1000,
        MasterValueSource:=0,
        BufferMode:=mcAborting,
        InSync=>camin1_insync,
        Busy=>camin1_busy,
        Active=>camin1_busy,
        CommandAborted=>camin1_abort
        );

    vel1(
        Axis:=1,
        Execute:= axis1_power AND axis2_power AND vel1_execute,
        Velocity:=30,
        Acceleration:=100,
        Deceleration:=100,
        Jerk:=1000,
        Direction:= mcPositiveDirection,
        BufferMode:=mcAborting,
        InVelocity=>vel1_invelocity,
        Busy=>vel1_busy,
        Active=>vel1_active,
        CommandAborted=>vel1_abort
        );
    halt1(
        Axis:=2,
        Execute:=axis1_power AND halt1_execute,
        Deceleration:=100,
        Jerk:=1000,
        BufferMode:=mcAborting,
        Done=>halt1_done,
        Busy=> halt1_busy,
        Active=> halt1_active,
        CommandAborted=> halt1_abort
        );
```

- **Program description**
  - When camIn1_execute changes from FALSE to TRUE after the axis is enabled, the electronic cam coupling instruction (camIn1) starts to execute, and when camIn1_active changes to TRUE and camIn1_insync changes to TRUE in the 2nd cycle, the cam relationship is established between the slave axis 2 and the master axis 1.
  - The cam relationship is established with the master axis in relative mode, the slave axis in relative mode, the master axis ratio is 1, and the slave axis ratio is 1.
  - After the cam relationship is established between the slave axis and the master, and vel1_execute changes from FALSE to TRUE, the master executes the velocity instruction, and the slave axis will follow the master to move according to the planned cam relationship.
  - At the end of one cam cycle, camIn1_endofprofile changes from FALSE to TRUE for one task cycle and then to FALSE again.
  - When the slave axis needs to deReservecam with the master and stop, change halt1_execute from FALSE to TRUE, and the halt1 instruction starts to execute. When halt1_active changes to TRUE, the camIn1 instruction execution is interrupted, the slave axis and the master deReservecam, and the slave axis decelerates and stops.



① The camin instruction (camin1) starts the execution of the slave axis 2 and master axis 1 to establish an electronic cam relationship.

② The master axis executes the velocity command and the slave axis follows the master axis

③ At the end of the cam periodicity, the EndofProfile bit becomes TRUE for one task cycle.

④ After the slave axis executes the halt1 instruction, the camin1 instruction is interrupted when halt1_active becomes TRUE, the slave axis and master axis release the cam relationship, and the slave axis decelerates and stops.

5.5 MC_CamOut（End cam operation）

This instruction is used to release the electronic cam relationship between the slave axis and the master established through the MC_CamIn instruction, and the slave axis continues to run at the current speed after release.
Library：MotionControl

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_CamOut	End cam operation	FB	MC_CamOut_Instance MC_CamOut Slave — Done Execute — Busy CommandAborted Error ErrorID	MC_CamOut _Instance (Slave :=parameter, Execute:=parameter , Done=> parameter, Busy=> parameter, CommandAborted=> parameter, Error=> parameter, ErrorID=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Slave	Slave axis number	USINT	Depends on model	Required field	Specify slave axis number in the electronic cam
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected.

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Cam has been disengaged.	BOOL	TRUE or FALSE	TRUE when the electronic cam relationship between the slave axis and the master axis is released.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
CommandAborted	Command aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted.
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Done	TRUE when the electronic cam relationship between the slave axis and the master axis is released.	◆ When CommandAborted changes to TRUE; ◆ When Error changes to TRUE;
Busy	When Execute changes to TRUE.	◆ When CommandAborted changes to TRUE; ◆ When Error changes to TRUE;
CommandAborted	When the instruction is aborted	◆ After one period when Execute is FALSE or the instruction is aborted by other instruction.
Error	The input variable of the instruction is out of the allowed range, the execution conditions of the instruction are not satisfied, or an exception is encountered during the execution of the instruction.	◆ When Execute is TRUE and changes to FALSE

■ Function description

- **Basic function descriptions**
  - This instruction is used to abort the execution of the MC_CamIn instruction. When this instruction is executed, the electronic cam relationship established between the slave axis and the master axis through MC_CamIn is released, and the slave axis maintains the slave axis speed at the time of the release after the release. If this instruction is executed while the slave axis is running, the slave axis will continue to run after the instruction is executed. If user want to release the electronic cam relationship and want the slave axis to decelerate and stop, user can directly execute the MC_Halt or MC_Stop instruction for the slave axis (it is not necessary to execute the MC_CamOut instruction first and then execute the MC_Halt or MC_Stop instruction).
  - The execution of this instruction has no effect on the movement of the master.
- **Re-execute the instruction**

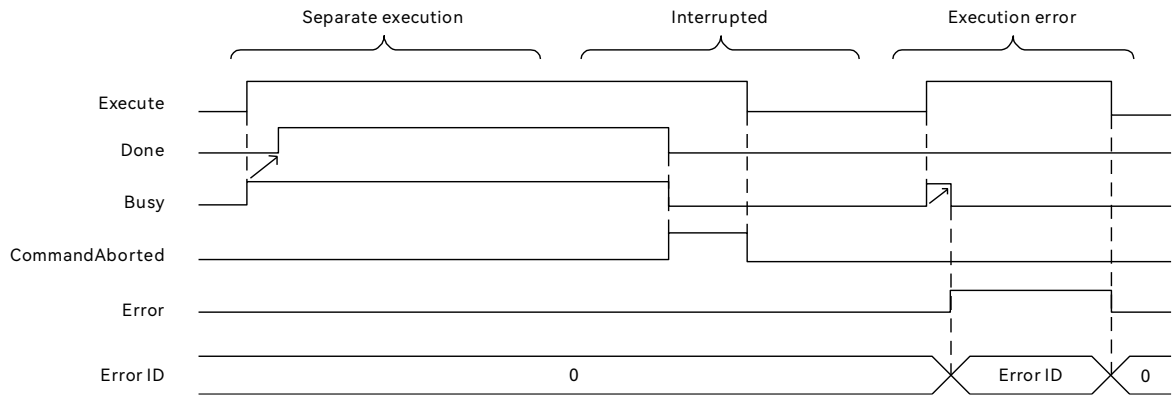
  This instruction cannot be restarted. Execution of this instruction reports an error when two axes have not established an electronic cam relationship.
- **Execute this instruction while other instructions are in processing**

  This instruction can be executed only when the MC_CamIn instruction is executed; execution of this instruction will report an error when other motion instructions are executed.
- **Execute other instructions while this instructions is in processing**

  When this instruction is executed, other motion instructions are started, and how other motion instructions and this instruction are buffered is determined by the value of BufferMode parameter of other instructions, and the value of BufferMode of other motion instructions can only be selected to interrupt and wait; if there is no BufferMode parameter of other motion instructions, it is generally interrupted. When other motion instructions and this instruction are cached, the timing of execution of other instructions is when the instruction Done (Cam has been disengaged) becomes TRUE.
- **Execution error**

  When this instruction is executed, if the input variable is illegal, the instruction Error becomes TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g. axis alarm), the instruction will also report error and the axis will stop immediately.

■ **Timing description of output variable status**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy (Execution in progress) changes to TRUE at the same time, Done (Electronic cam relationship release) changes to TRUE in the next cycle, Busy (Execution in progress) remains TRUE, and the commanded speed of the axis remains unchanged from that before

decoupling. When Execute becomes FALSE, Busy (in execution) and Done (electronic cam relationship decoupling) remain TRUE.

- Execution interruption

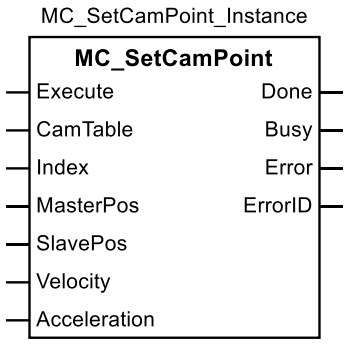
  When the execution of this instruction is interrupted by other instructions, the instruction CommandAborted becomes TRUE, and Busy (Execute), Active (Control), and InSync (Cam Synchronization) become FALSE at the same time; when Execute (Execution) becomes FALSE, CommandAborted becomes FALSE at the same time.

- Execution error

  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

5.6 MC_SetCamPoint（Changing the data of a specified cam point）

This instruction is used to set the data for the specified cam point in the specified cam table. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetCamPoint	Changing the data of a specified cam point	FB		MC_SetCamPoint _Instance (Execute:=parameter, CamTable :=parameter, Index :=parameter, MasterPos :=parameter, SlavePos :=parameter, Velocity :=parameter, Acceleration:=parameter , Done=> parameter , Busy=> parameter , Error=> parameter , ErrorID => parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes
Index	Cam point number	UINT	Depends on model	Required field	Cam point number in the cam table
MasterPos	Master phase	LREAL	Positive number,0	0	Cam point in cam table for master phase
SlavePos	slave phase	LREAL	Positive or negative number,0	0	Cam point in cam table for slave phase
Velocity	Connection velocity	LREAL	Positive or negative number,0	0	Proportional value of slave axis velocity and master velocity at cam point
Acceleration	Connection acceleration	LREAL	Positive or negative number,0	0	Proportional value of follower acceleration and master acceleration at cam point

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False

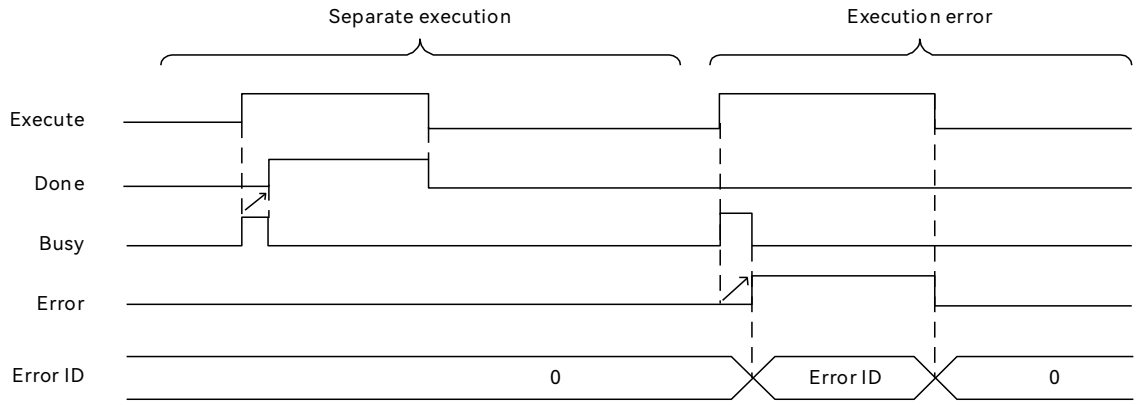
Done	When execution is complete	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one cycle later.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

● Basic function descriptions

This instruction is used to change the data of the specified cam point in the specified cam table. The data of the cam point includes the master axis phase, slave axis phase, connection speed and connection acceleration of the cam point. After this instruction is executed, the changed cam point data does not take effect immediately and must be used in conjunction with the MC_ChangeCamCurve instruction and MC_CamIn instruction. After the execution of this instruction is completed, the MC_ChangeCamCurve instruction is executed first, and then the MC_CamIn instruction is executed, and the cam point information changed by this instruction takes effect immediately after the execution of the MC_CamIn instruction; after the execution of the MC_CamIn instruction is executed, the instruction is executed first, and then the MC_ChangeCamCurve instruction is executed, and the cam point information changed by the instruction takes effect immediately after the execution of the current cam cycle. information takes effect after the current cam cycle is executed (the axis phase takes effect after passing through the end phase of the cam main axis).

■ Timing description of output variable status



● Separate execution

When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

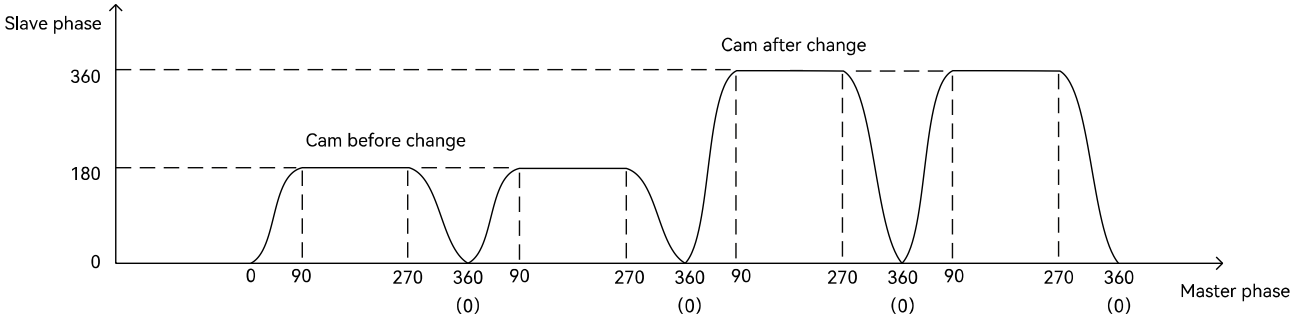
● Execution error

When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

■ The example program is shown below

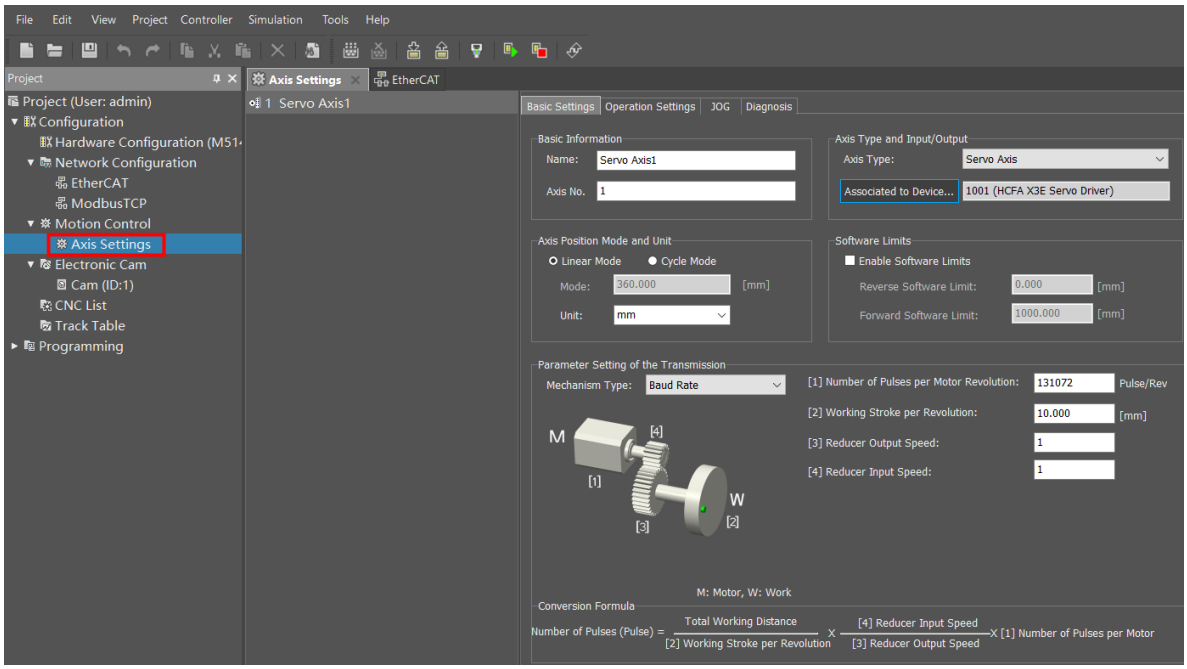
- **Functionality**

  Cam key point data is changed during cam operation, and the newly changed cam data takes effect at the end of the current cam cycle. The cam curve is changed by changing the cam key point data after the electronic cam relationship is established between the slave axis and the master axis. The cam key point data can be changed by using this instruction in conjunction with the MC_ChangeCamCurve instruction.

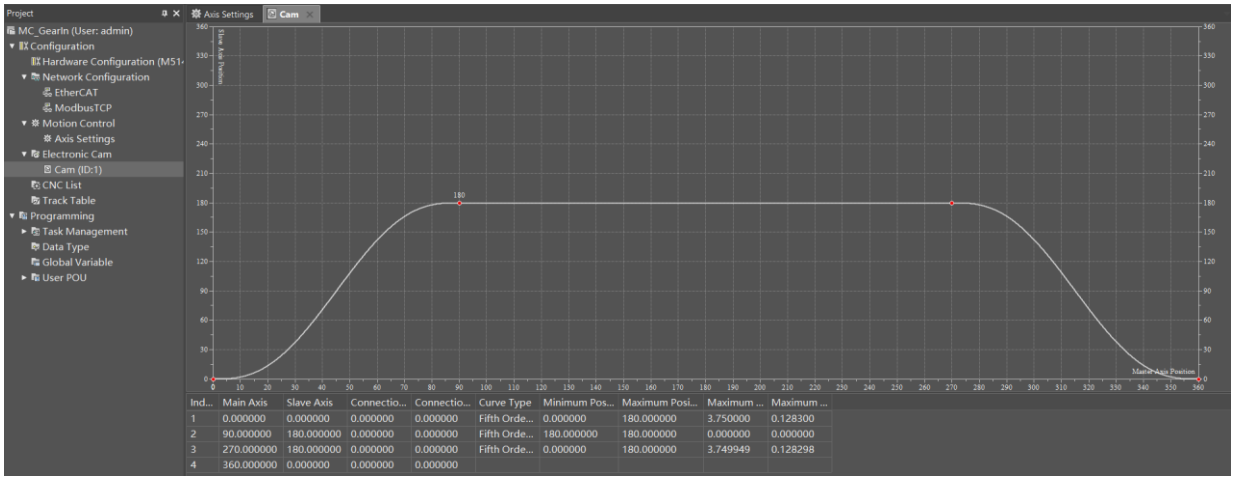


- **Axis parameters and cam curve setting**

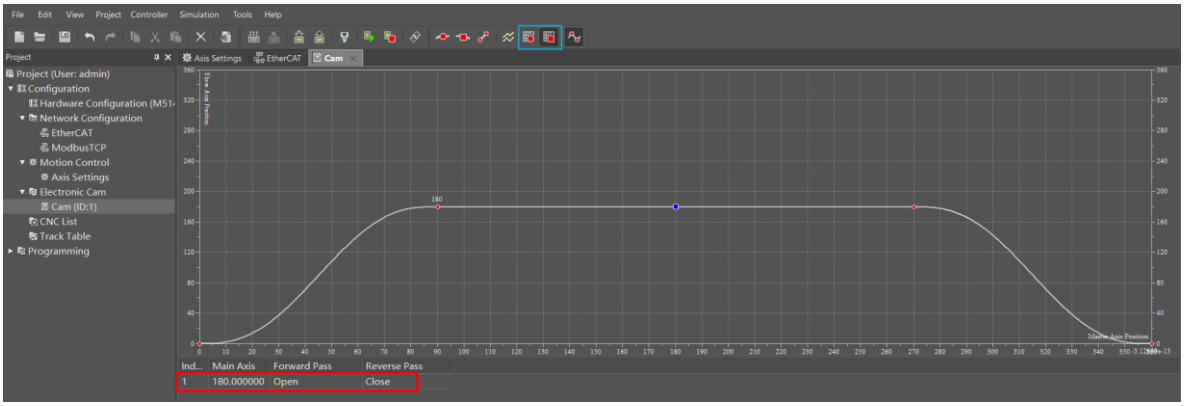
  The axis parameters for mater axis 1 and slave axis 2 are the same, and the axis parameter settings for axis 1 are shown below.



The cam curve created in the software is shown below



The tappet points added to the cam curve are shown in the red box below. Tappet points can be added and displayed by clicking on the blue box below.

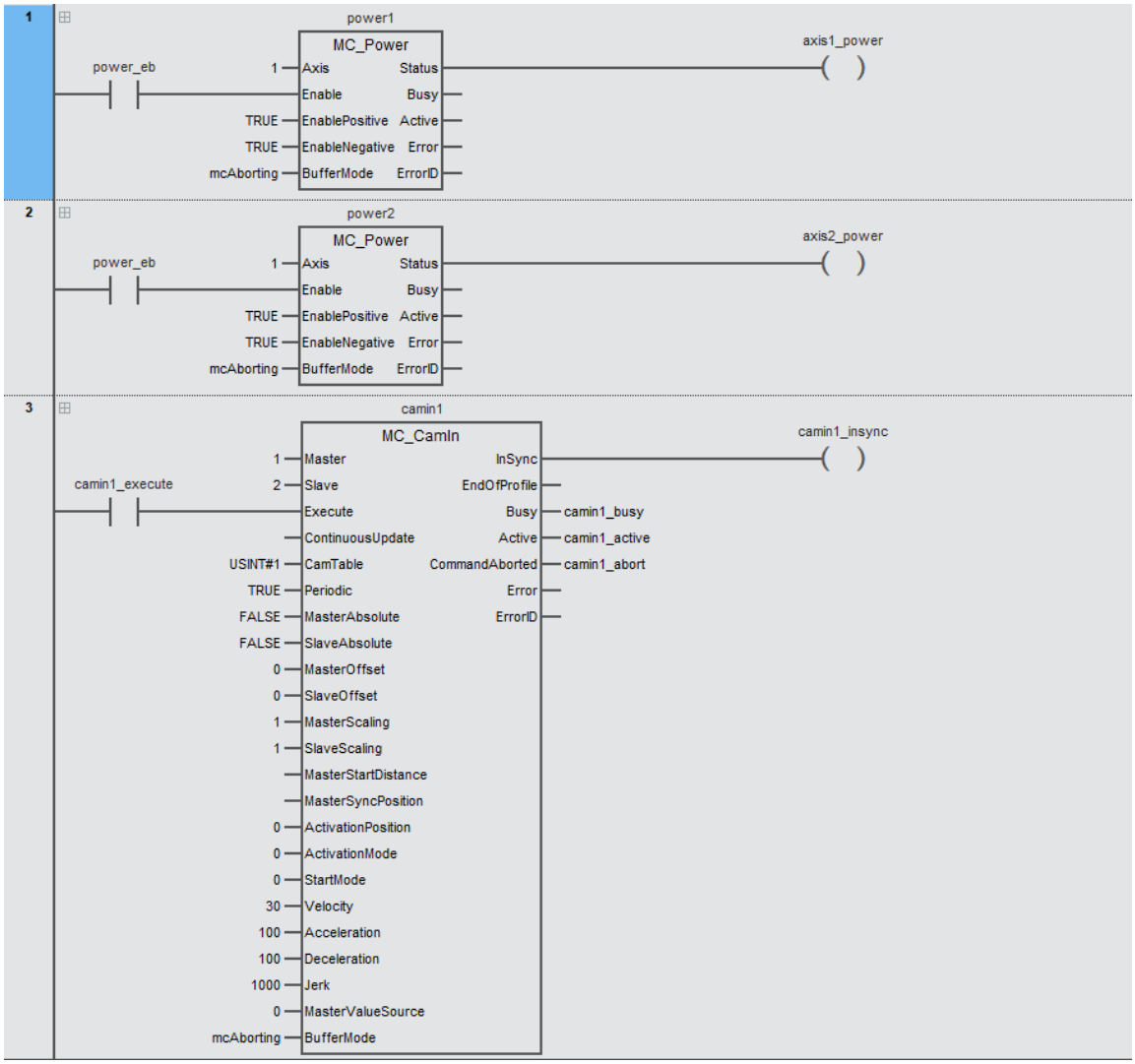


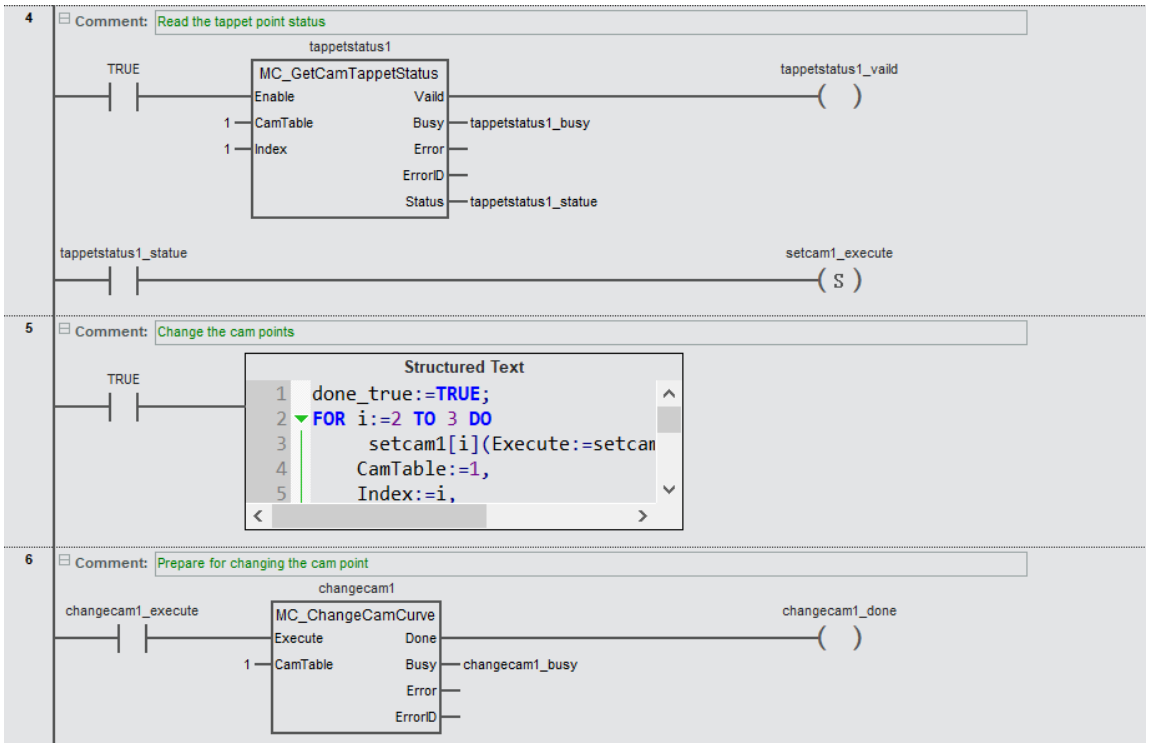
- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	powe1		MC_Power		
VAR	axis1_power		BOOL		
VAR	power2		MC_Power		
VAR	axis2_power		BOOL		
VAR	camin1		MC_CamIn		
VAR	camin1_execute		BOOL		
VAR	gearIn1_Ingear		BOOL		
VAR	camin1_insync		BOOL		
VAR	camin1_busy		BOOL		
VAR	camin1_active		BOOL		
VAR	camin1_abort		BOOL		
VAR	changecam1		MC_ChangeCamCurve		
VAR	changecam1_execute		BOOL		
VAR	changecam1_done		BOOL		
VAR	changecam1_busy		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_execute		BOOL		
VAR	vel1_done		BOOL		
VAR	vel1_busy		BOOL		

VAR	vel1_active		BOOL		
VAR	vel1_abort		BOOL		
VAR	setcam1		ARRAY[1..4] OF MC_SetCamPoint		
VAR	setcam1_execute		BOOL		
VAR	masterpos		ARRAY[1..4] OF LREAL	[0,90,270,360]	
VAR	slavepos		ARRAY[1..4] OF LREAL	[0,360,360, 0]	
VAR	done_true		BOOL		
VAR	i		INT		
VAR	tappetstatus1		MC_GetCamTappetStatus		
VAR	tappetstatus1_vaild		BOOL		
VAR	tappetstatus1_busy		BOOL		
VAR	tappetstatus1_statue		BOOL		
VAR	start		BOOL		

- LD





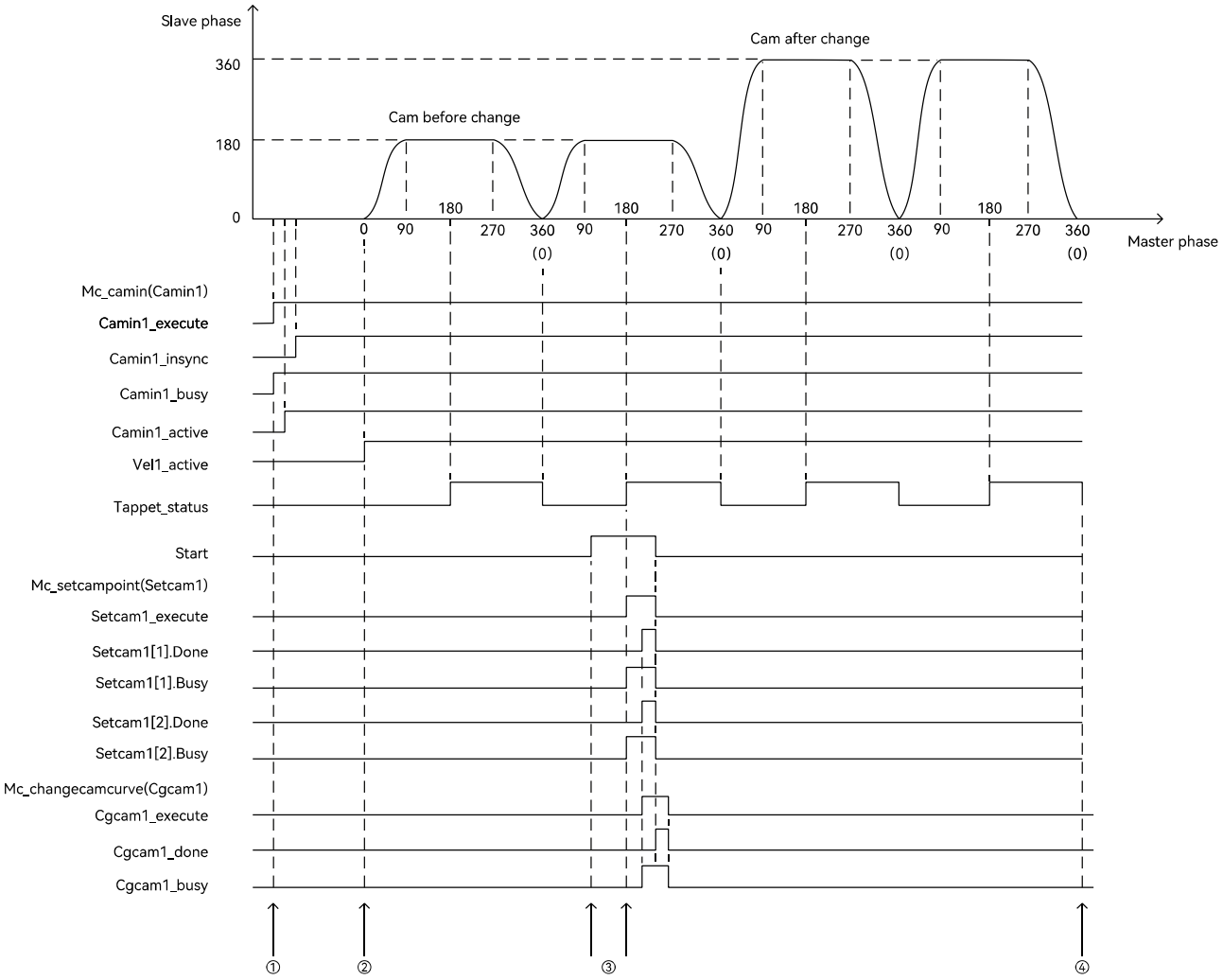
- ST

```
power1(
    Axis:=1,
    Enable:=power_eb,
    EnablePositive:=TRUE,
    EnableNegative:=TRUE,
    BufferMode:=mcAborting,
    Status=>axis1_power
    );
power2(
    Axis:=2,
    Enable:=power_eb,
    EnablePositive:=TRUE,
    EnableNegative:=TRUE,
    BufferMode:=mcAborting,
    Status=>axis2_power
    );


camin1(
    Master:=1,
    Slave:=2,
    Execute:=camin1_execute,
    CamTable:=USINT#1,
    Periodic:=TRUE,
    MasterAbsolute:=FALSE,
    SlaveAbsolute:=FALSE,
    MasterOffset:=0,
    SlaveOffset:=0,
    MasterScaling:=1,
    SlaveScaling:=1,
```

```
                ActivationPosition:=0,
                ActivationMode:=0,
                StartMode:=0,
                Velocity:=30,
                Acceleration:=100,
                Deceleration:=100,
                Jerk:=1000,
                MasterValueSource:=0,
                BufferMode:=mcAborting,
                InSync=>camin1_insync,
                Busy=>camin1_busy,
                Active=>camin1_busy,
                CommandAborted=>camin1_abort
                );
//Read tappet point status
        tappetstatus1(
                Enable:=TRUE ,
                CamTable:=1 ,
                Index:=1 ,
                Vaild=>tappetstatus1_vaild ,
                Busy=>tappetstatus1_busy ,
                Status=>tappetstatus1_statue
                );
        IF tappetstatus1_statue THEN
                setcam1_execute:=TURE;
        END_IF;
//Change cam point
        done_true:=TRUE;
        FOR i:=2 TO 3 DO
        setcam1[i](
                Execute:=setcam1_execute AND start,
                CamTable:=1,
                Index:=i,
                MasterPos:=masterpos[i],
                SlavePos:=slavepos[i],
                Velocity:=0,
                Acceleration:=0
                );
        done_true:=done_trueandsetcam1[i].Done;
        END_FOR;
        IFEDGEPOS(done_true)THEN
                setcam1_execute:=FALSE;
                changecam1_execute:=TRUE;
        END_IF;
//Preparing for cam point changes
        changecam1(
                Execute:=changecam1_execute,
                CamTable:=1,
                Done=>changecam1_done,
                Busy=>changecam1_busy
                );
        IF changecam1_done THEN
                changecam1_execute:=FALSE;
```

```
        start:=FALSE;
    END_IF;

    vel1(
        Axis:=1,
        Execute:=axis1_power AND vel1_execute,
        Velocity:=30,
        Acceleration:=100,
        Deceleration:=100,
        Jerk:=1000,
        Direction:=1,
        BufferMode:=mcAborting,
        InVelocity=>vel1_invelocity,
        Busy=>vel1_busy,
        Active=>vel1_active,
        CommandAborted=>vel1_abort
        );
```

- **Program description**
  - When camin1_execute becomes TRUE after the slave axis and the master are enabled, the electronic cam coupling instruction (camin1) is executed, and the slave axis and the master establish an electronic cam relationship through the electronic cam coupling instruction (camin1).
  - After the slave axis and the master establish the electronic cam relationship, the master executes the speed command and the slave axis follows the master.
  - When start is TRUE, the cam tappet status1 command (tappetstatus1) detects that the master phase is greater than or equal to 180, and the tappet status1_statue variable changes from FALSE to TRUE, then the MC_SetCamPoint command is triggered. Change the cam point data. After all the cam point data to be changed have been changed, trigger the MC_ChangeCamCurve instruction to execute, and the newly changed cam point data will take effect at the end of the current cam cycle.
  - In this example program, the 2nd and 3rd cam points are changed from 180 to 360, and the data of the cam points to be changed are assigned to the initial values of the masterpos and slavepos array variables, refer to the variable table for details.
  - When it is necessary to change the data of multiple cam points, user can instantiate the MC_SetCamPoint instruction as an array and change the cam point data through the FOR loop.For example, the data type of SetCam1 in this sample program is ARRAY[1..4] OF MC_SetCamPoint.



① The camin instruction (camin1) starts the execution of the slave axis 2 and master axis 1 to establish an electronic cam relationship.

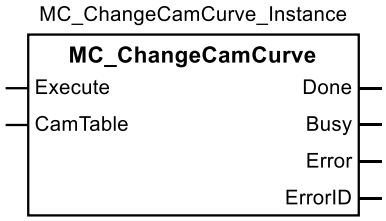
② The master axis executes the velocity command and the slave axis follows the master axis.

③ When the Read Tappet Point Status instruction detects that the master axis phase is greater than or equal to 180, it will change to TRUE, triggering the MC SetCamPoint instruction to be executed, and after the cam point data change is completed, triggering the MC ChangeCamCurve instruction to be executed, in order to prepare for the effect of the newly changed cam point data.

④ At the end of the cam periodicity, the tappet point status automatically changes to FALSE.

5.7 MC_ChangeCamCurve（Change of cam data）

This instruction prepares the data for the cam point set by MC_SetCamPoint to take effect. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_ChangeCamCurve	Change of cam data	FB		MC_ChangeCamCurve _Instance (Execute:=parameter , CamTable:=parameter, Done=> parameter, Busy=> parameter, Error=> parameter, ErrorID=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Execute this instruction when the rising edge of this parameter is detected.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Setting the electronic cam relationship table number

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

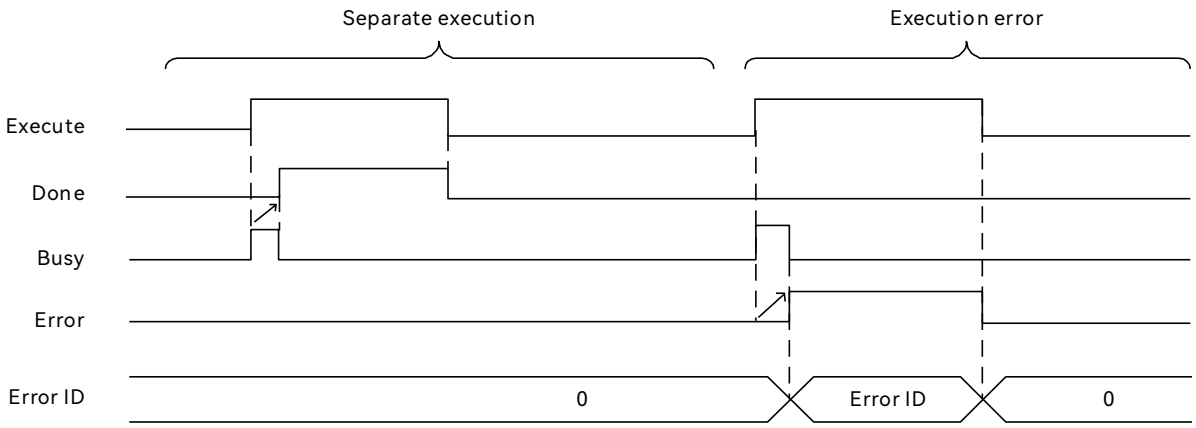
Name	Whether or not to become TRUE	Whether or not to become False
Done	When execution is complete	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one cycle later.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

- This instruction prepares the data for the cam point set by MC_SetCamPoint to take effect. This instruction is to be used with the MC_SetCamPoint and MC_CamIn instruction commands. When this instruction is executed, it prepares for the validation of the data of all cam points set by MC_SetCamPoint.
- After the MC_CamIn instruction is executed, the MC_SetCamPoint instruction is executed first, and then this instruction is executed, and the cam point information changed by the MC_SetCamPoint instruction takes effect after the current cam cycle has been executed (the axis phase passes through the end phase of the cam master and takes effect). The MC_SetCamPoint instruction is executed first, then this

instruction is executed, then the MC_CamIn instruction is executed, and the cam point information changed by the MC_SetCamPoint instruction takes effect immediately when the MC_CamIn instruction is triggered for execution.

■ Timing description of output variable status



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

- **Execution error**

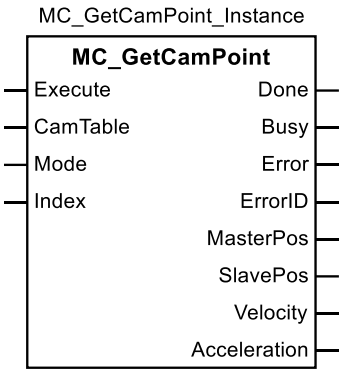
  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

■ Example program

For the example program, refer to the example program in the MC_SetCamPoint instruction.

5.8 MC_GetCamPoint（Read data from specified cam point）

This instruction is used to read the data of the specified cam point in the specified cam table. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_GetCamPoint	Read data from specified cam point	FB		MC_GetCamPoint _Instance (Execute:=parameter, CamTable:=parameter, Mode:=parameter , Index:=parameter , Done=> parameter , Busy => parameter, Error => parameter, ErrorID=> parameter , MasterPos=> parameter , SlavePos=> parameter , Velocity=> parameter , Acceleration=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes
Mode	Mode selection	BOOL	TRUE or FALSE	FALSE	FALSE: Read the cam point data of the currently used cam. TRUE: Read cam point data of spare cams.
Index	Cam point number	UINT	Depends on model	Required field	Cam point numbering in the cam table

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.
MasterPos	Master phase	LREAL	Positive number,0	Cam point in cam table for master phase
SlavePos	slave phase	LREAL	Positive or negative number,0	Cam point in cam table for slave phase
Velocity	Connection velocity	LREAL	Positive or negative number,0	Proportional value of slave axis velocity and master velocity at cam point
Acceleration	Connection acceleration	LREAL	Positive or negative number,0	Proportional value of follower acceleration and master acceleration at cam point

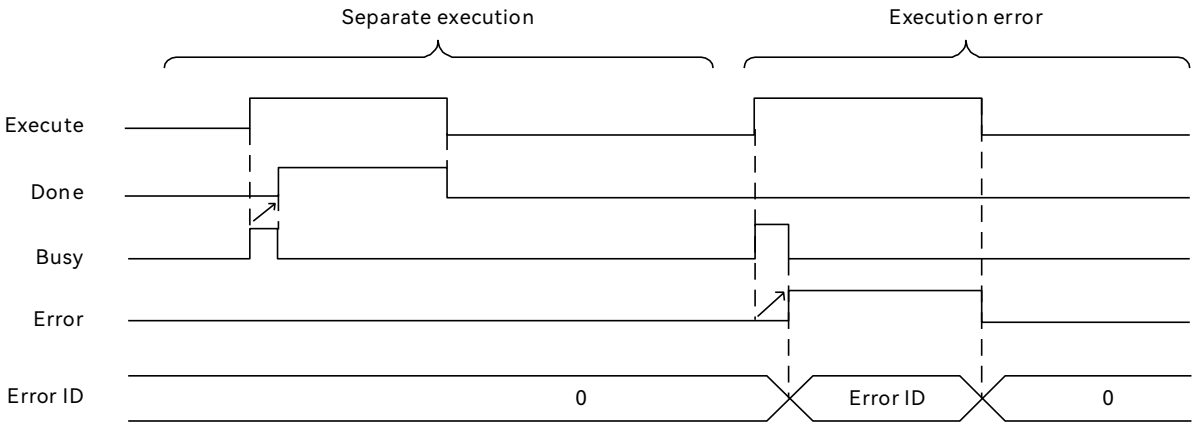
■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Done	When execution is complete	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one cycle later.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

- This instruction is used to read the data of the specified cam point in the specified cam table. The data of the cam point includes the master axis phase, slave axis phase, connection speed, and connection acceleration of the cam point.
- When Mode is FALSE, the cam point data of the currently used cam is read; when Mode is TRUE, the cam point data of the standby cam is read.
- If the cam period is 360, the master axis phase of the currently executing cam point is 100, the slave axis phase is 100, the connection speed is 0, and the connection acceleration is 0. The master axis phase of the cam point is changed to 200, the slave axis phase is changed to 200, the connection speed is changed to 0, and the connection acceleration is changed to 0 by using the MC_SetCamPoint and MC_ChangeCamCurve instructions. When modified cam point data has not yet taken effect, when the instruction Mode is FALSE, the master axis phase of the cam point is read as 100, the slave axis phase is read as 100, the connection speed is read as 0, and the connection acceleration is read as 0. When the instruction Mode is TRUE, the master axis phase of the cam point is read as 200, the slave axis phase is read as 200, the connection speed is read as 0, and the connection acceleration is read as 0. After one cam cycle, the newly changed cam point data takes effect. When the instruction Mode is FALSE, the master axis phase of the cam point is read as 200, the slave axis phase is read as 200, the connection speed is read as 0, and the connection acceleration is read as 0.

■ Timing description of output variable status



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

- **Execution error**

When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

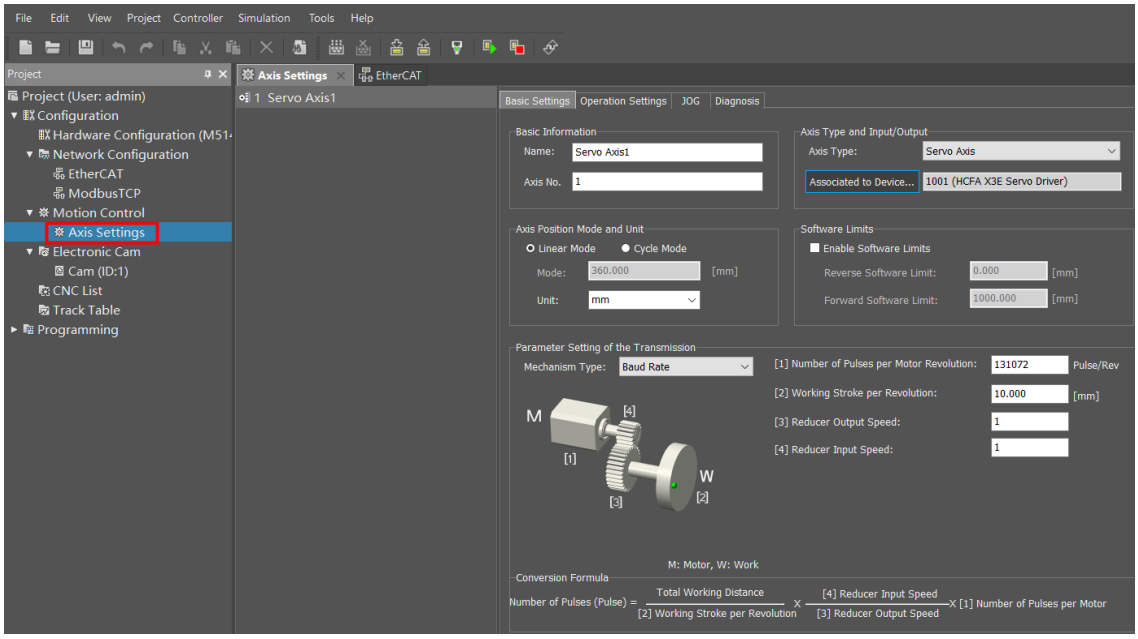
■ The example program is shown below

● Functionality

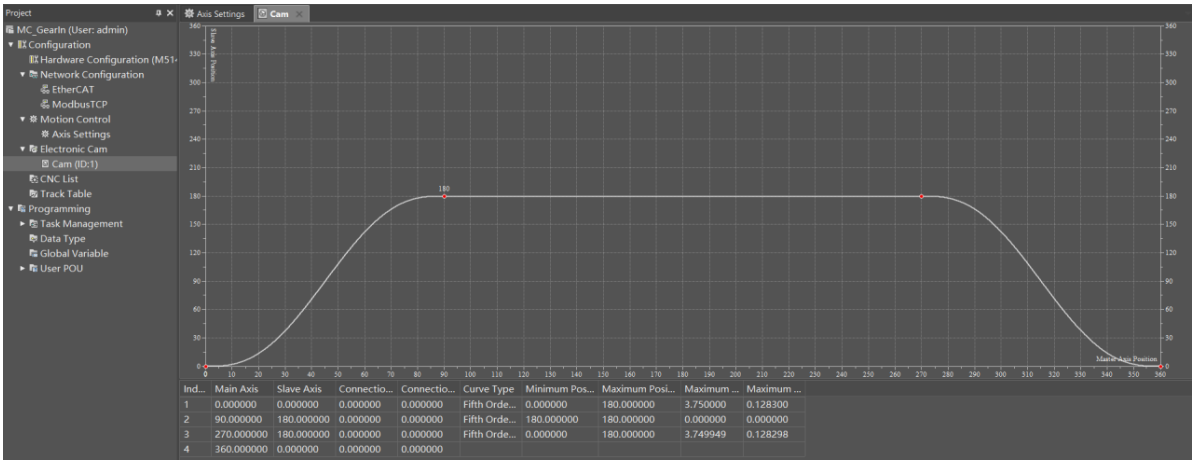
Reads the data of the specified cam point in the cam table.

● Axis parameters and cam curve setting

The axis parameters for mater axis 1 and slave axis 2 are the same, and the axis parameter settings for axis 1 are shown below.



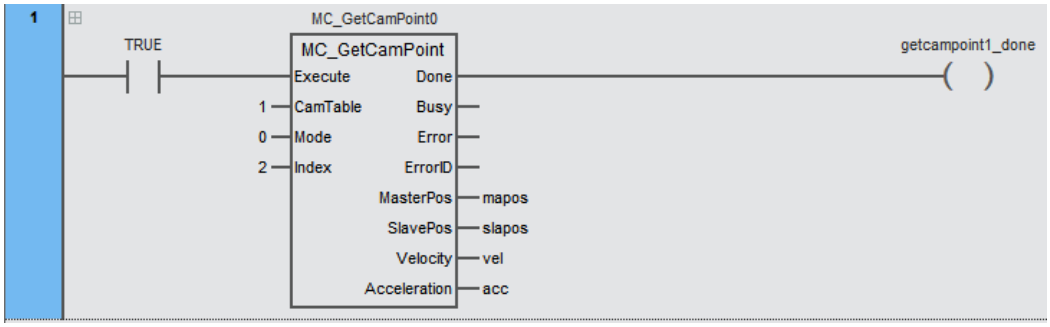
The cam curve created in the software is shown below



- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	getcampoint1		MC_GetCamPoint		
VAR	getcampoint1_execute		BOOL		
VAR	getcampoint1_done		BOOL		
VAR	masterpos		LREAL		
VAR	slavepos		LREAL		
VAR	vel		LREAL		
VAR	acc		LREAL		

- **LD**

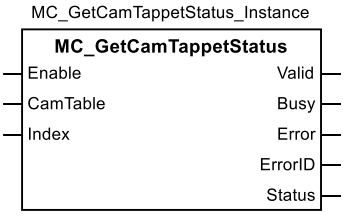


- **ST**

```
getcampoint1
    (Execute:= getcampoint1_execute ,
    CamTable:=1 ,
    Mode:=0 ,
    Index:=2 ,
    Done=> getcampoint1_done ,
    MasterPos=> masterpos,
    SlavePos=>slavepos,
    Velocity=>vel ,
    Acceleration=>acc
     );
```

5.9 MC_GetCamTappetStatus（Read cam tappet point status）

This instruction is used to read whether the current position of the axis passes through the position of the tappet point set in the cam table. Library: MotionControl_Part2

Input variable	Meaning	Data type	Valid range	Default
MC_GetCamTappetStatus	Read cam tappet point status	FB		MC_GetCamTappetStatus _Instance (Enable:=parameter , CamTable:=parameter, Index:=parameter , Valid=> parameter , Busy=> parameter , Error => parameter, ErrorID => parameter, Status=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Enable	Execute	BOOL	TRUE or FALSE	FALSE	TRUE: Read the status of the specified cam tappet point. FALSE: Stop read the status of the specified cam tappet point
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes
Index	Tappe point number	UINT	1~8	Required field	Tappe point number

■ Output variable

Output variable	Meaning	Data type	Default	Description
Valid	Read tappet point status valid	BOOL	TRUE or FALSE	When this command reads the status of the specified tappet point, the status of this parameter value is TRUE
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.
Status	tappet point status	BOOL	TRUE or FALSE	Specifies the status of the tappet point number

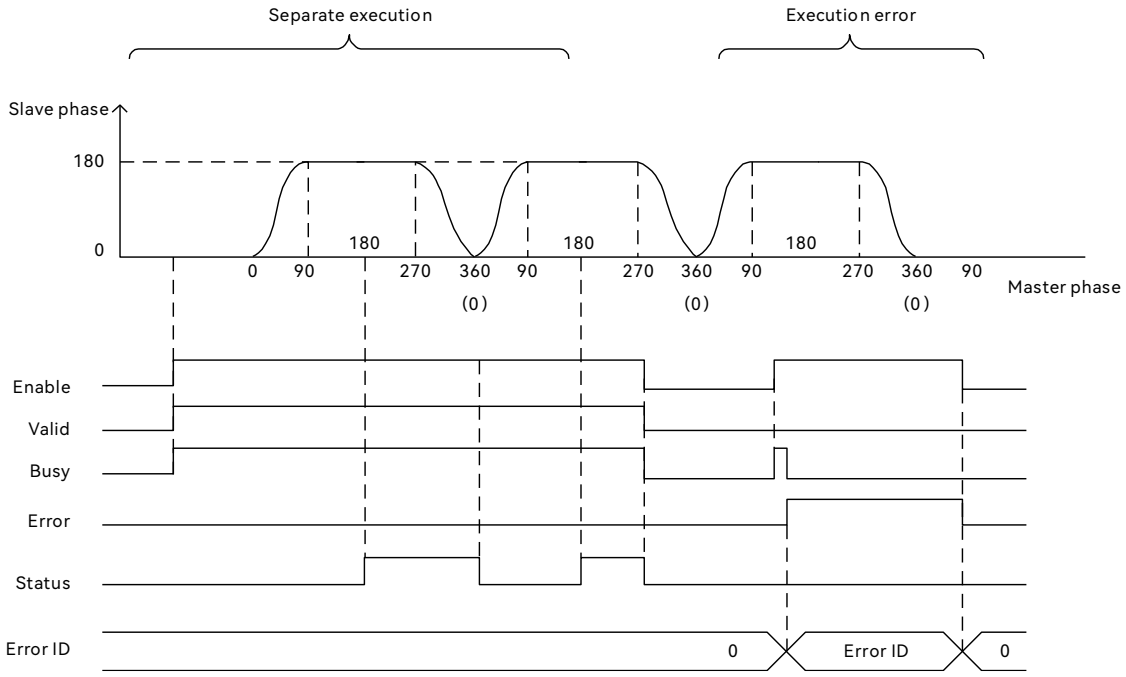
■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Valid	When Enable changes to TRUE	◆ Execute changes from TRUE to FALSE.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE.
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE.

Function description

Basic function descriptions

This instruction is used to read the status of the specified tappet point, which is used to determine whether the current position of the axis passes through the position of the tappet point set in the cam table. The state of the specified tappet point is the value when the master axis phase passes through the tappet point in the forward or reverse direction. The state of the tappet point is set by the tappet point in the "Electronic Cam" in the software or by the MC_SetCamTappet instruction. The status of each tappet point changes to FALSE when the master axis passes through the end phase of the cam in the forward direction or the start phase of the cam in the reverse direction.



Timing description of output variable status

Separate execution

When Enable changes from FALSE to TRUE, Busy and Valid change to TRUE at the same time, and Status can output the corresponding status according to the tappet point setting. For example, in this schematic diagram, the value of tappet point 1 is set to 180, and it is TRUE when it passes in the positive direction, otherwise when the cam master axis phase is greater than or equal to 180, the Status is changes to TRUE. At the end of the cam cycle, the Status is automatically changed to FALSE. When Enable is changed to FALSE, Busy, Valid and Status will also change to FALSE at the same time when Enable changes to FALSE.

Execution error

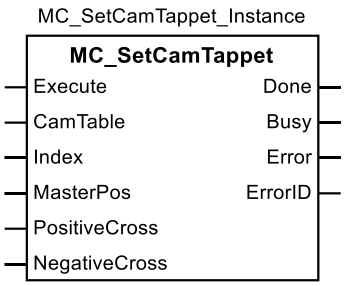
When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

Example program

For the example program, refer to the example program in the MC_SetCamPoint instruction.

5.10 MC_SetCamTappet（Set cam tappet point status）

The instruction is used to set cam tappet point status. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetCamTappet	Set cam tappet point status	FB		MC_SetCamTappet _Instance (Execute:=parameter , CamTable:=parameter, Index:=parameter , MasterPos:=parameter , PositiveCross :=parameter, NegativeCross:=parameter, Done=> parameter , Busy=> parameter , Error => parameter , ErrorID => parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes
Index	Tappe point number	UINT	Depends on model	Required field	Tappe point number
MasterPos	Master phase	LREAL	Positive number,0	0	Cam point in cam table for master phase
PositiveCross	Positive crossing mode	TappetCrossMode	0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert	0：TappetDisable	Mode when the axis passes positively through the tappet point: 0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert
NegativeCross	Negative crossing Mode	TappetCrossMode	0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert	0：TappetDisable	Mode when the axis passes negative through the tappet point: 0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False

Done	When execution is complete	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one cycle later.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

● Basic function descriptions

➢ This instruction sets the specified cam tappet point setting data. The tappet point setting data includes the master phase, positive pass mode and negative pass mode for that tappet point. The master phase positive pass mode and negative pass mode for the specified cam point can be set separately.

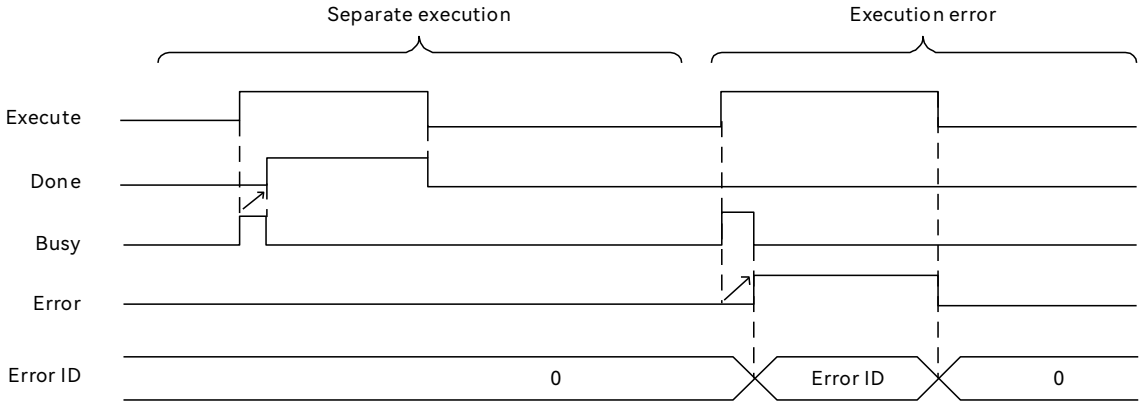
➢ The following table describes the modes and functions of the master phase positive passing through the specified cam point:

Mode	Function	Meaning
TappetDisable	None	No function, when the master passes the specified cam tappet point in the positive direction, the status of the tappet point does not change.
TappetOn	Open	The status of the tappet point is TRUE when the master passes the specified cam tappet point in the positive direction.
TappetOff	Close	The status of the tappet point is FALSE when the master passes the specified cam tappet point in the positive direction.
TappetInvert	Switch	The original tappet point status is reversed (FALSE: before the master passes the specified cam tappet point in the positive direction; TRUE: after the master passes the specified cam tappet point in the positive direction; TRUE: before the master passes the specified cam tappet point in the positive direction; FALSE : after the master passes the specified cam tappet point in the positive direction).

■ Timing description of output variable status

● Separate execution

When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.



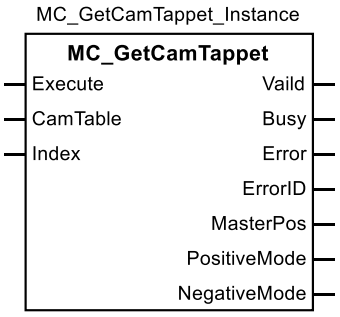
● Execution error

When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in

the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

5.11 MC_GetCamTappet（Get cam tappet point setting data）

This instruction is used to read cam tappet point setting data in the specified cam table. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_GetCamTappet	Get cam tappet point setting data	FB		MC_GetCamTappet_Instance (Execute:=parameter, CamTable :=parameter, Index:=parameter, Vaild=> parameter , Busy=> parameter , Error=> parameter , ErrorID=> parameter , MasterPos => parameter, PositiveMode=> parameter , NegativeMode=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes
Index	Tappe point number	UINT	Depends on model	Required field	Tappe point number

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.
MasterPos	Master phase	LREAL	Positive or negative number,0	Cam point in cam table for master phase
PositiveCross	Positive passing mode	TappetCrossMode	0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert	Mode when the axis passes positively through the tappet point: 0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert
NegativeCross	Negative Passing Mode	TappetCrossMode	0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert	Mode when the axis passes negative through the tappet point: 0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Valid	Enable changes to TRUE	◆ Enable changes from TRUE to FALSE
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

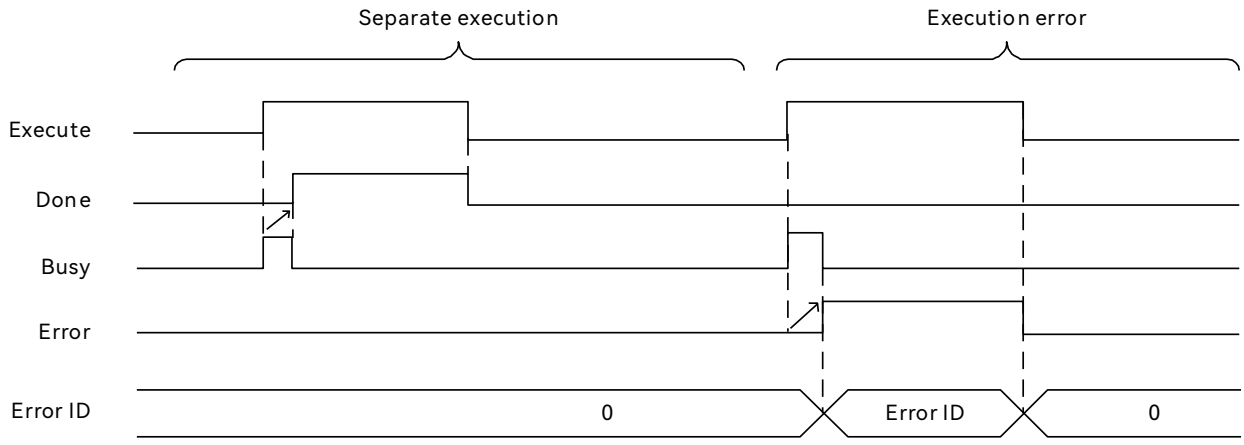
● Basic function descriptions

➢ This instruction read the specified cam tappet point setting data. The tappet point setting data includes the master phase, positive pass mode and negative pass mode for that tappet point. The master phase positive pass mode and negative pass mode for the specified cam point can be set separately.

➢ The following table describes the modes and functions of the master phase positive passing through the specified cam point:

Mode	Function	Meaning
TappetDisable	None	No function, when the master passes the specified cam tappet point in the positive direction, the status of the tappet point does not change.
TappetOn	Open	The status of the tappet point is TRUE when the master passes the specified cam tappet point in the positive direction.
TappetOff	Close	The status of the tappet point is FALSE when the master passes the specified cam tappet point in the positive direction.
TappetInvert	Switch	The original tappet point status is reversed (FALSE: before the master passes the specified cam tappet point in the positive direction; TRUE: after the master passes the specified cam tappet point in the positive direction; TRUE: before the master passes the specified cam tappet point in the positive direction; FALSE : after the master passes the specified cam tappet point in the positive direction).

■ Timing description of output variable status



● Separate execution

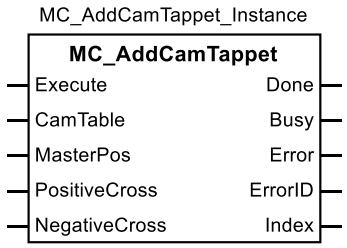
When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

- **Execution error**

  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

5.12 MC_AddCamTappet（Add cam tappet point）

This instruction is used to add a cam tappet point to the end of a table of available cam tappet points. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AddCamTappet	Add cam tappet point	FB		MC_AddCamTappet _Instance (Execute:=parameter, CamTable :=parameter, MasterPos :=parameter, PositiveMode:=parameter , NegativeMode :=parameter, Done => parameter, Busy=> parameter , Error => parameter, ErrorID => parameter, Index => parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes
MasterPos	Master phase	LREAL	Positive number,0	Required field	Cam point in cam table for master phase
PositiveCross	Positive passing mode	TappetCrossMode	0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert	0	Mode when the axis passes positively through the tappet point: 0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert
NegativeCross	Negative Passing Mode	TappetCrossMode	0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert	0	Mode when the axis passes negative through the tappet point: 0：TappetDisable 1：TappetOn 2：TappetOff 3：TappetInvert

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.
Index	Cam point number	UINT	1~8	Add the cam point number

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False

Done	When execution is complete	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one cycle later.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ When Error is TRUE and Execute changes from TRUE to FALSE; ◆ Execute changes from TRUE to FALSE

■ Function description

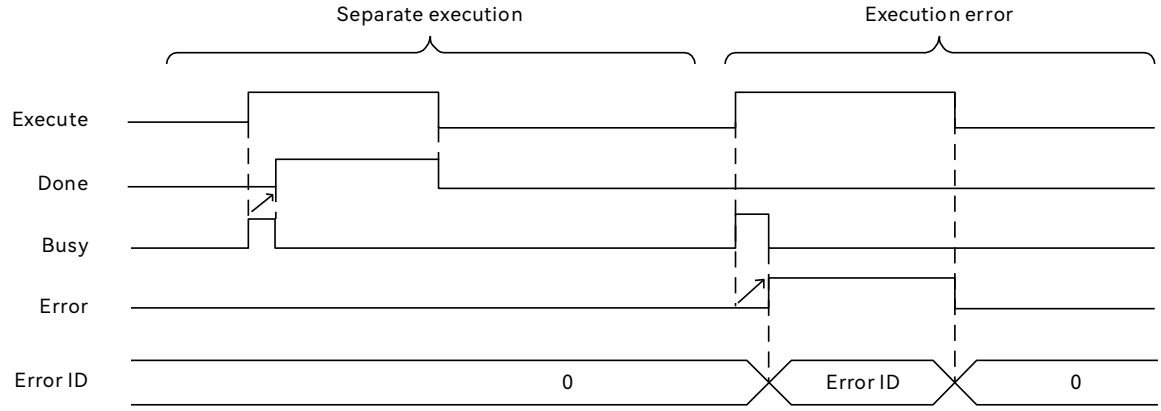
● Basic function descriptions

➢ This instruction is used to add a cam tappet point to the end of an existing cam tappet point table. The cam tappet points added include master phase, forward cross mode and reverse cross mode. This instruction is executed once, and the number of the newly added cam tappet point is increased sequentially from the maximum value of the existing number. For example, if there are 2 tappet points in the specified cam table, numbered 1 and 2 respectively, after the instruction is executed three times, the tappet points numbered 3, 4 and 5 are added sequentially, and there are 5 tappet points numbered 1, 2, 3, 4 and 5 in the cam table.

➢ The following table describes the modes and functions of the master phase positive passing through the specified cam point:

Mode	Function	Meaning
TappetDisable	None	No function, when the master passes the specified cam tappet point in the positive direction, the status of the tappet point does not change.
TappetOn	Open	The status of the tappet point is TRUE when the master passes the specified cam tappet point in the positive direction.
TappetOff	Close	The status of the tappet point is FALSE when the master passes the specified cam tappet point in the positive direction.
TappetInvert	Switch	The original tappet point status is reversed (FALSE: before the master passes the specified cam tappet point in the positive direction; TRUE: after the master passes the specified cam tappet point in the positive direction; TRUE: before the master passes the specified cam tappet point in the positive direction; FALSE : after the master passes the specified cam tappet point in the positive direction).

■ Timing description of output variable status



● Separate execution

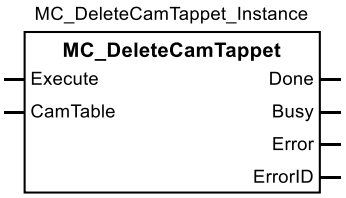
When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

- **Execution error**

  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

5.13 MC_DeleteCamTappet（Delete cam tappet point）

This instruction is used to delete a cam tappet point to the end of a table of available cam tappet points. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_DeleteCamTappet	Delete cam tappet point	FB		MC_DeleteCamTappet _Instance (Execute:=parameter , CamTable:=parameter , Done => parameter, Busy => parameter, Error => parameter, ErrorID => parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
CamTable	Electronic cam relationship table number	USINT	Depends on model	Required field	Cam table defining the relationship between the master and slave axes

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when execution of the instruction is complete
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

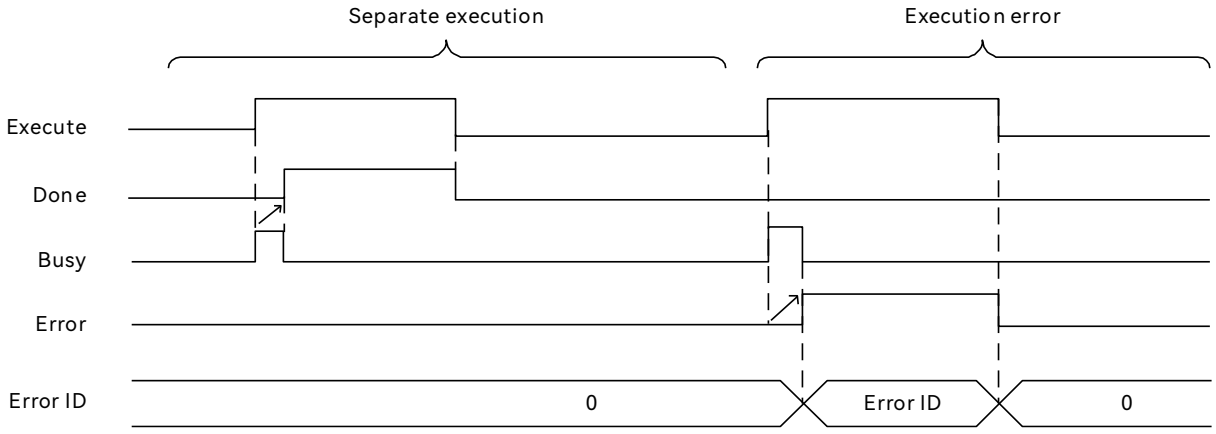
Name	Whether or not to become TRUE	Whether or not to become False
Done	When execution is complete	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one cycle later.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

- **Basic function descriptions**

  This instruction is used to delete the last cam tappet point in the specified cam table. Deleted the largest tappet point. This instruction is executed once to delete the tappet point with the largest number in a cam table. For example, if there are five tappet points in the specified cam table, numbered 1, 2, 3, 4, and 5 respectively, after the instruction is executed three times, the tappet points numbered 5, 4, and 3 will be deleted in turn, and only the tappet points numbered 1 and 2 will be deleted from the cam table.

■ Timing description of output variable status



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

- **Execution error**

  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

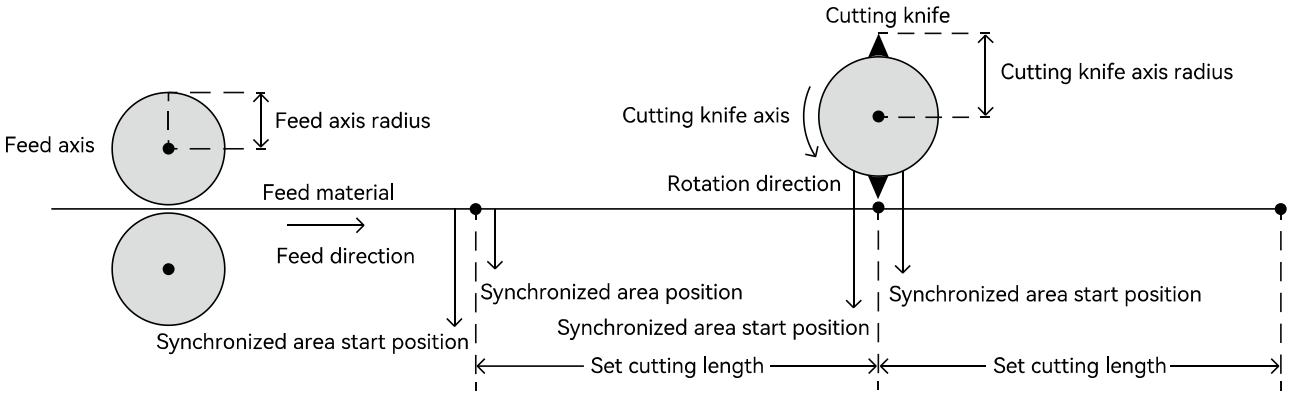
Chapter6 Rotary

Cutting Process and Related Instructions

6.1 Introduction of rotary cutting process and applicable occasion

The rotary cutting process means that the knife runs in one direction and cuts the continuously fed material to the length set by the user. The cutting length specified by the user is the cycle. When cutting, i.e. in the synchronization zone, the speed of the knife is synchronized with the speed of the feed material, and when there is no cutting, the knife axis can be accelerated or decelerated to prepare for the next synchronization in the synchronization zone and synchronization with the speed of the feed material.

The diagram of the rotary cutting process and common parameters are shown below.

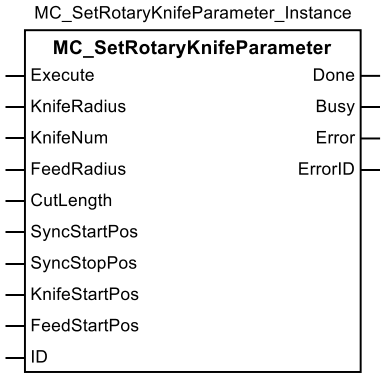


Applicable occasion

Applicable to thin material cutting occasions, such as can be widely used in packaging, printing, film processing, tissue paper production equipment.

6.2 MC_SetRotaryKnifeParameter(Parameter setting of rotary cutting mechanism)

This instruction sets the parameters setting of rotary cutting mechanism, and prepares for the establishment of the rotary cutting relationship between the knife axis and the feed axis. Library: MotionControl.

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetRotaryKnifeParameter	parameters setting of rotary cutting mechanism	FB		MC_SetRotaryKnifeParameter_Instance (Execute:=parameter , KnifeRadius:=parameter , KnifeNum:=parameter , FeedRadius :=parameter, CutLength:=parameter, SyncStartPos:=parameter , SyncStopPos:=parameter , KnifeStartPos :=parameter, FeedStartPos:=parameter , ID:=parameter , Done=> parameter, Busy => parameter, Error=> parameter , ErrorID => parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
KnifeRadius	Knife axis radius	LREAL	Positive number	Required field	Distance from the center of the knife axis to the tip of the knife blade. Refer to the diagram in the function description for details
KnifeNum	Number of knifes	USINT	1~8	Required field	Number of knifes on the knife axis
FeedRadius	Feed axis radius	LREAL	Positive number	Required field	Feed axis radius Refer to the diagram in the function description for details
CutLength	cut the length of (sth.)	LREAL	Positive number	Required field	Set the cut the length of (sth.) Refer to the diagram in the function description for details
SyncStartPos	Synchronization zone start position	LREAL	Positive number	Required field	Position of the knife axis and feed axis at the beginning of velocity synchronization Refer to the diagram in the function description for details
SyncStopPos	Synchronization zone end position	LREAL	Positive number	Required field	Position of the knife axis and feed axis at the ending of velocity Refer to the diagram in the function description for details
KnifeStartPos	Reserved	LREAL	Reserved	Reserved	Reserved
FeedStartPos	Reserved	LREAL	Reserved	Reserved	Reserved
ID	rotary cutting number	USINT	Depends on model	Required field	Rotary cutting number

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	TRUE when the Rotary Cutting Parameter Settings Completed.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

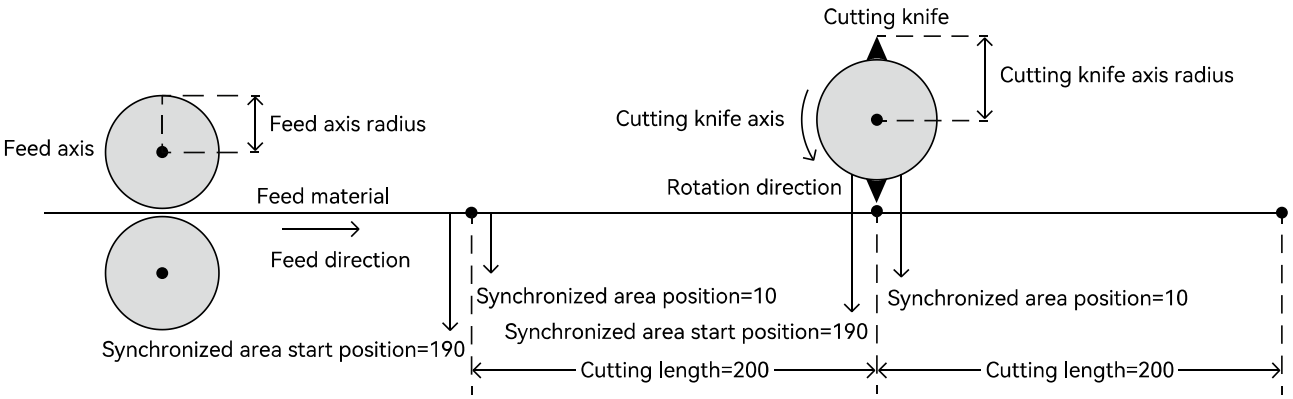
■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Done	When execution is complete	◆ When Done is TRUE and Execute changes to TRUE from FALSE ◆ After one period when Done changes to TRUE and Execute is FALSE or the instruction is aborted by other instruction.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

● Basic function descriptions

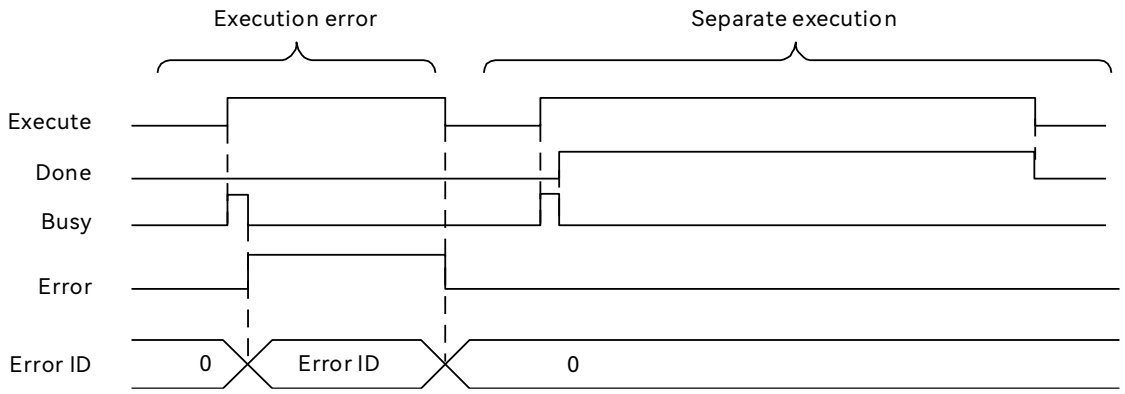
- This instruction is used to set the parameters of the rotary cutting system, the meaning of the input variables can refer to the following diagram of the rotary cutting relationship. The parameters such as knife axis radius, feed axis radius, number of knifes, etc. are determined by the actual organization, and the cutting length is determined by the user's actual application requirements.
- The set parameters will take effect immediately, if the feed axis and the knife axis have not established rotary cutting relationship, execute this instruction and then execute MC_RotaryKnife_In instruction. After the rotary relationship between feed axis and knife axis is established, execute this instruction to change the value of input variables, such as cut length, synchronization zone start position, synchronization zone end position and other parameters, the changed value will take effect at the end of the current rotary cycle (it will take effect at the beginning of the next rotary cycle).



Synchronization zone start position(SyncStartPos): The velocity of the knife is synchronized with the feed material velocity from the start position of the synchronization zone until the end position of the synchronization zone (SyncStopPos).

Synchronization zone end position(SyncStopPos): From the end position of the synchronization zone, the velocity of the knife may not be synchronized with the feed material velocity, but may be accelerated or decelerated as needed to prepare for the next time it is in the synchronization zone and synchronized with the feed material velocity.

■ Timing description of output variable status



- **Execution error**

  When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

6.3 MC_RotaryKnife_In（Rotary coupling）

According to the set parameters of the rotary cutting mechanism, the knife axis and the feed axis establish the rotary cutting relationship calculated by the controller. Library：MotionControl.

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_RotaryKnife_In	Rotary coupling	FB	MC_RotaryKnife_In_Instance MC_RotaryKnife_In Execute / Done KnifeAxis / Busy FeedAxis / Error ID / ErrorID	MC_RotaryKnife_In _Instance (Execute:=parameter, KnifeAxis:=parameter, FeedAxis:=parameter, ID:=parameter, Done=> parameter, Busy=> parameter, Error=> parameter, ErrorID => parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
KnifeAxis	Knife axis number	USINT	Depends on model	Required field	Specify the axis number of the knife axis.
FeedAxis	Feed axis number	USINT	Depends on model	Required field	Specify the axis number of the feed axis.
ID	rotary cutting number	USINT	Depends on model	Required field	rotary cutting number

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Busy	Executing	BOOL	TRUE after the rotary cut relationship building.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

■ Output variable refreshing timing

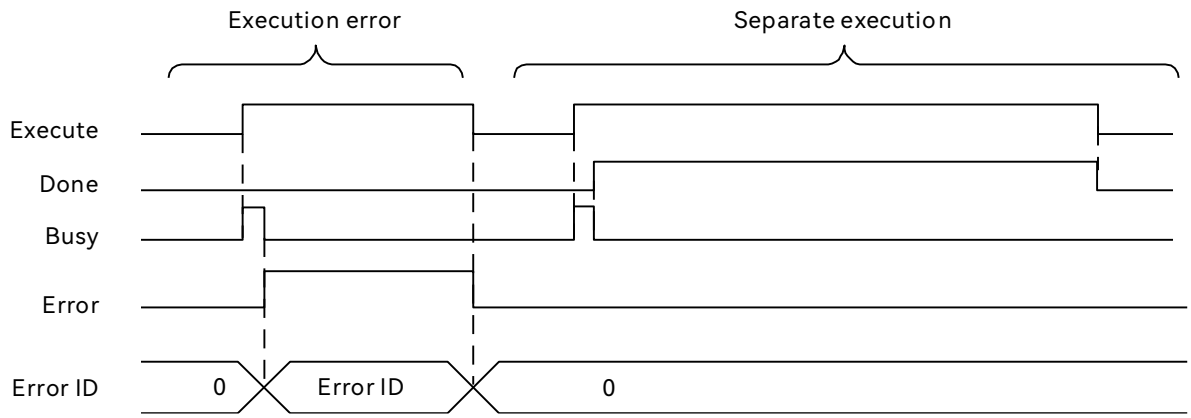
Name	Whether or not to become TRUE	Whether or not to become False
Done	When execution is complete	◆ After one period when Done changes to TRUE and Execute is FALSE or the instruction is aborted by other instruction.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not within the allowed range	◆ Execute changes from TRUE to FALSE

■ Function description

● Basic function descriptions

According to the set parameters of the rotary cutting mechanism, the knife axis and the feed axis establish the rotary cutting relationship calculated by the controller. After the rotary relationship is established, the knife axis follows the feed axis in periodic operation.

■ Timing description of output variable status



Execution error

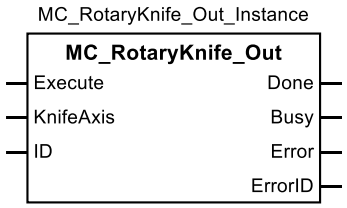
When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

6.4 MC_RotaryKnife_Out（Decoupling of the rotational relationship）

According to the set parameters of the rotary cutting mechanism, the knife axis and the feed axis are out of the rotary cutting relationship calculated by the controller. Library: MotionControl.

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_RotaryKnife_Out	Decoupling of the rotational relationship	FB		MC_RotaryKnife_Out _Instance (Execute:=parameter , KnifeAxis:=parameter , ID:=parameter , Done=> parameter , Busy => parameter, Error=> parameter , ErrorID=> parameter);

■ Input variable

Input variable	Meaning	Data type	Valid range	Default	Description
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Specify master axis number in the electronic cam.
KnifeAxis	Knife axis number	USINT	Depends on model	Required field	Specify the axis number of the knife axis.
ID	rotary cutting number	USINT	Depends on model	Required field	rotary cutting number

■ Output variable

Output variable	Meaning	Data type	Default	Description
Done	Done	BOOL	TRUE or FALSE	The knife axis and the feed axis are released from the rotary cutting relationship and become TRUE after stopping.
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE while there is an error.
ErrorID	Error code	WORD	0~65535	Refer to "Instruction Error Code Description" for the meaning of the output error code value when an instruction execution exception occurs.

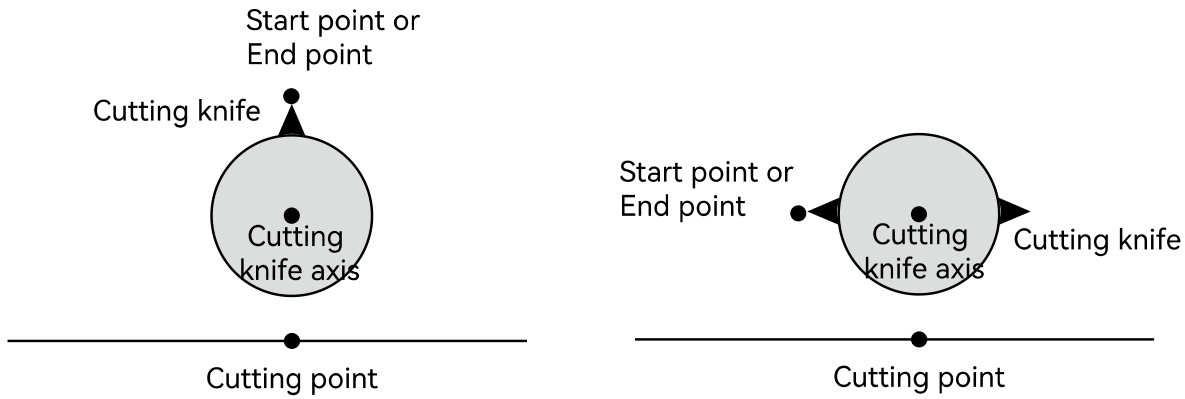
■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become False
Done	When execution is complete	◆ When Done is TRUE, Execute changes to TRUE from FALSE ◆ After one period when Done changes to TRUE and Execute is FALSE or the instruction is aborted by other instruction.
Busy	Rising edge of Execute	◆ When Done changes to TRUE; ◆ When Error changes to TRUE;
Error	Command input variable value is not in the allowed range	◆ When Error is TRUE and Execute changes from TRUE to FALSE; ◆ Execute changes from TRUE to FALSE

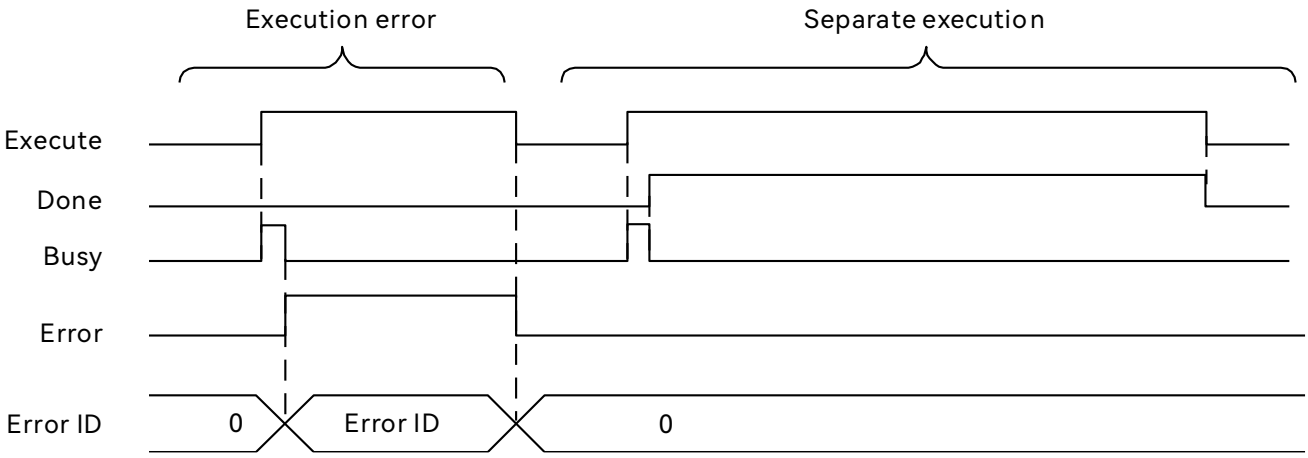
■ Function description

- **Basic function descriptions**
  - The knife axis and feed axis are decoupled according to the set rotary cutting parameters. The position where the knife axis and the feed axis stop after they are uncoupled becomes the exit point and the starting point for the knife axis and the feed axis establish the next rotation. The figure below shows the exit point when the knife axis has 1 tool and 2 tools.

- After the rotary cutting relationship is established between the knife axis and the feed axis, the instruction is executed when the feed axis is in motion. After the instruction is executed, the knife axis stops at the position it was in before the rotary cutting relationship was established, preventing the knife axis from coming into contact with the object on the feed axis mechanism, and the Done of the instruction changes to TRUE at the same time.
- After the rotary relationship between the knife axis and the feed axis is established and the command is executed when the master is stopped, the knife axis will not run and will not immediately be out of the rotary relationship with the feed axis, but will wait for the knife axis to run to the position before the rotary relationship was established to stop and be out of the rotary relationship.
- The instruction can be executed only after the rotary cutting relationship is established between the knife axis and the feed axis, otherwise an error will be reported.



■ **Timing description of output variable status**



Execution error

When the value of the input variable of this instruction is not within the permissible range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that user can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

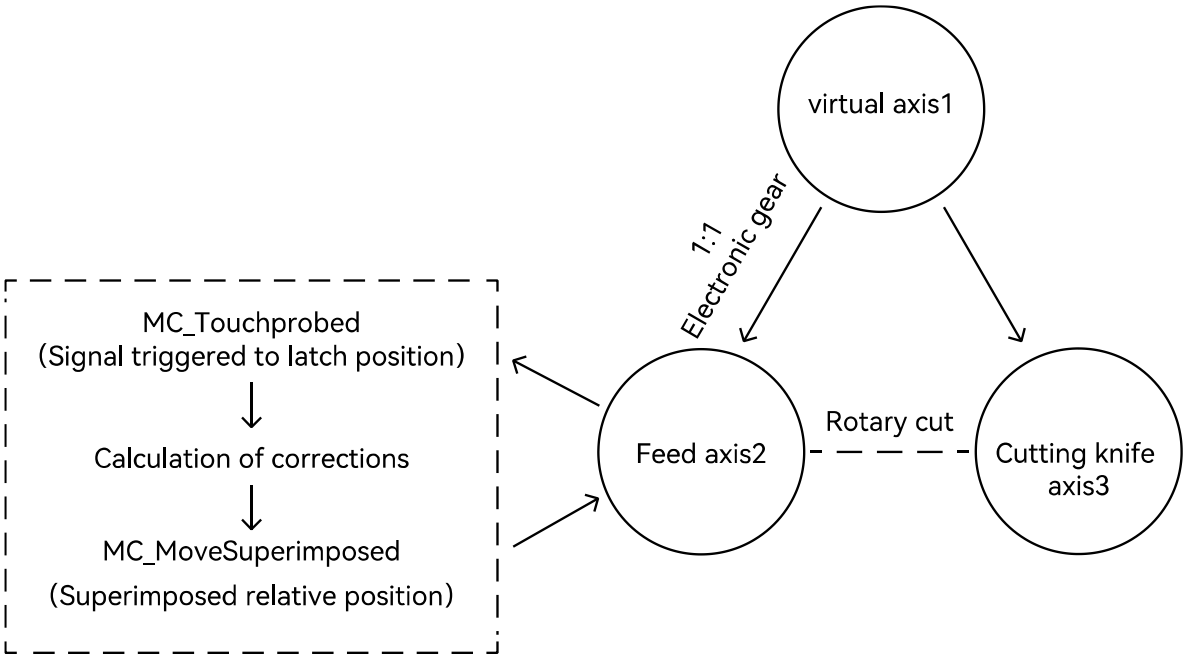
- **Separate execution**

When Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the next cycle

Done changes to TRUE while Busy changes to FALSE. when Execute changes to FALSE, Done changes to FALSE at the same time.

- **The example program of rotary cut is shown below**
  - **Functionality**

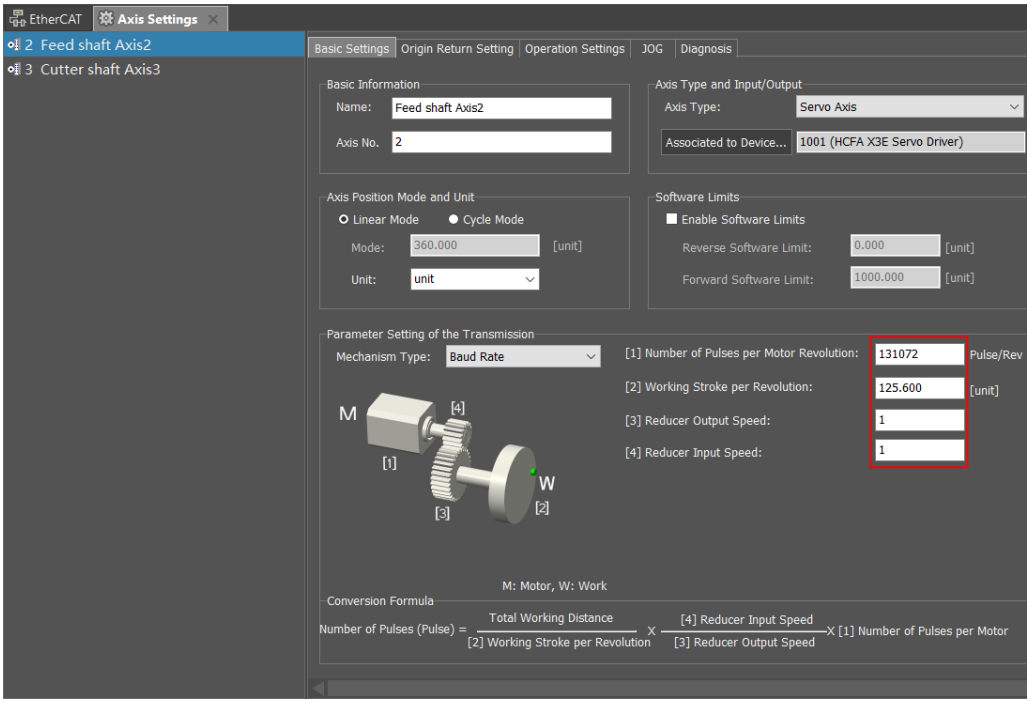
    As shown in the figure below, the program is designed to establish a 1:1 electronic gear relationship between the virtual servo axis 1 and the feed axis 2, and a rotary relationship between the virtual servo axis 1 and the knife axis 3, and then an indirect rotary relationship between the feed axis 2 and the knife axis 3. The advantage of this is that the feed axis 2 can be given to the need to correct the position in real time. But feed axis 2 and knife axis 3 to establish a direct rotary relationship between the program is not involved in the position of the correction, so the need for position correction, you can refer to the following figure at the dotted line box operation method. The MC_RotaryKnife_Out instruction is executed when the knife axis needs to be decoupled from the feed axis and stopped.



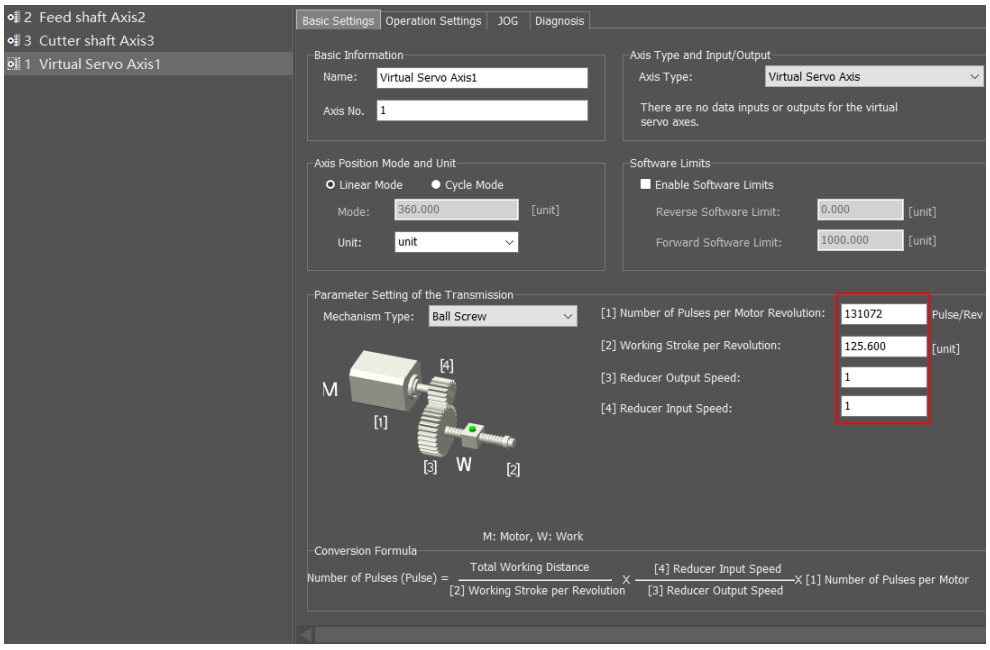
  - **Axis parameters setting**

    The feed axis(axis 2) parameters are set as shown below.

"Work stroke per revolution " = 2*3.14*20 (feed axis radius) = 125.6.

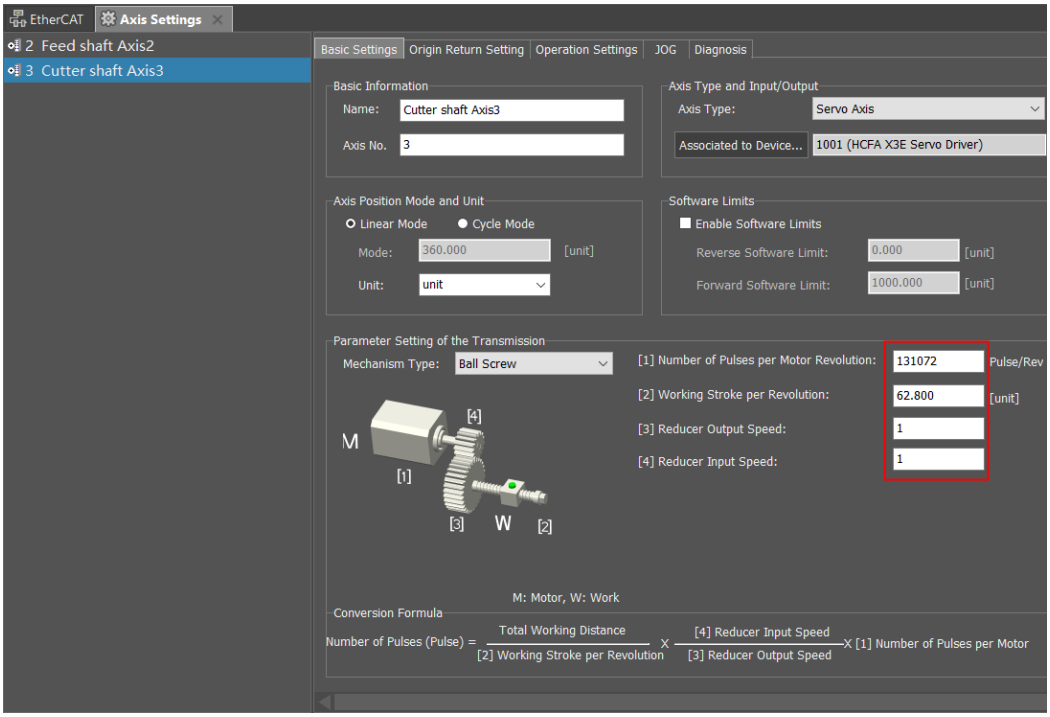


The virtual servo axis and the feed axis are in a 1:1 electronic gear relationship, and the virtual servo axis (axis 1) parameters are the same as the feed axis parameters, as shown below.



The knife axis (axis 3) parameters are set as shown below.

"Work stroke per revolution " =2*3.14*10（Knife axis radius）=62.8.

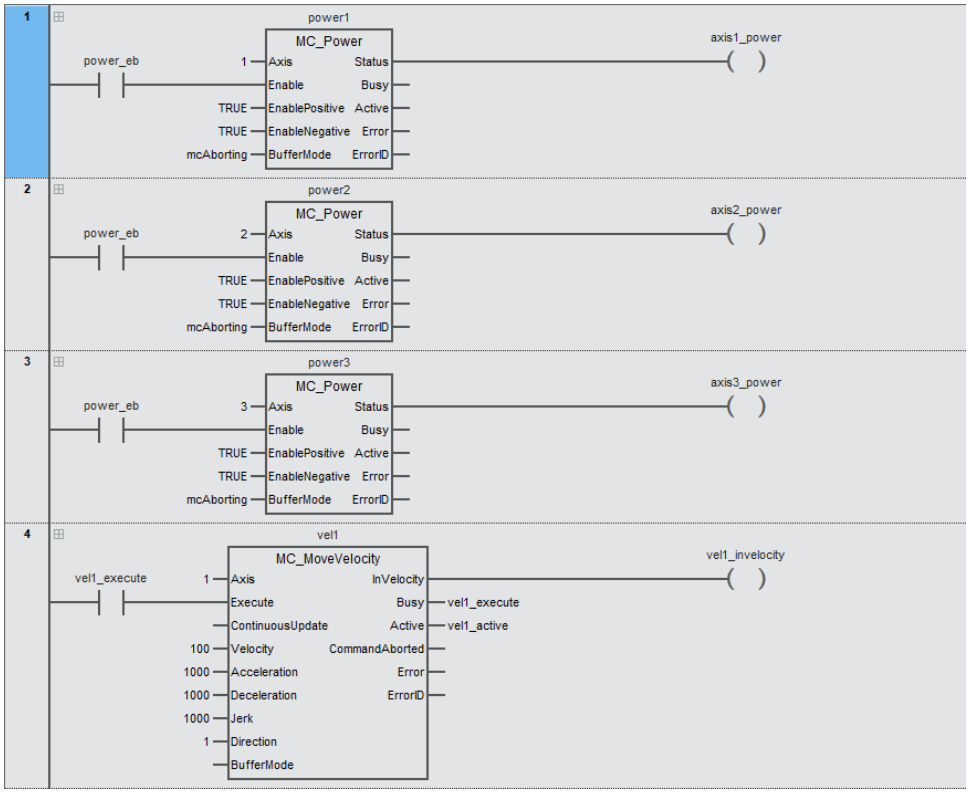


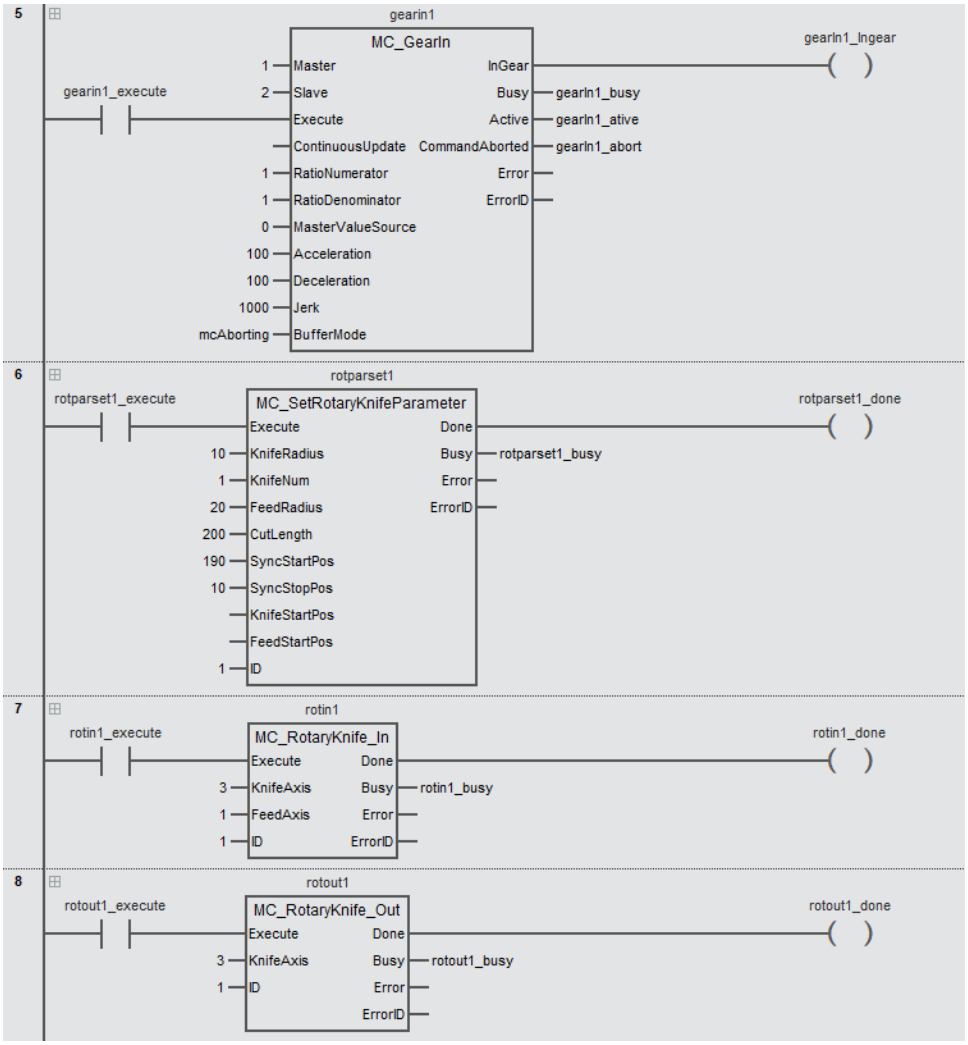
- **Variable table**

Category	Name	Assigned to	Data type	Initial value	Comment
VAR	power_eb		BOOL		
VAR	power1		MC_Power		
VAR	axis1_power		BOOL		
VAR	power2		MC_Power		
VAR	axis2_power		BOOL		
VAR	power3		MC_Power		
VAR	axis3_power		BOOL		
VAR	vel1		MC_MoveVelocity		
VAR	vel1_execute		vel1_execute		
VAR	vel1_invelocity		BOOL		
VAR	vel1_busy		BOOL		
VAR	vel1_active		BOOL		
VAR	gearin1		MC_GearIn		
VAR	gearin1_execute		BOOL		
VAR	gearIn1_Ingear		BOOL		
VAR	gearIn1_busy		BOOL		
VAR	gearIn1_ative		BOOL		
VAR	gearIn1_abort		BOOL		
VAR	rotparset1		MC_SetRotaryKnifePar ameter		
VAR	rotparset1_execute		BOOL		
VAR	rotparset1_done		BOOL		
VAR	rotparset1_busy		BOOL		
VAR	rotin1		MC_RotaryKnife_In		
VAR	rotin1_execute		BOOL		
VAR	rotin1_done		BOOL		

VAR	rotin1_busy		BOOL		
VAR	rotout1		MC_RotaryKnife_Out		
VAR	rotout1_execute		BOOL		
VAR	rotout1_done		BOOL		
VAR	rotout1_busy		BOOL		

LD





ST

```
power1(
    Axis:=1 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis1_power
    );

power2(
    Axis:=2 ,
    Enable:=power_eb ,
    EnablePositive:=TRUE ,
    EnableNegative:=TRUE ,
    BufferMode:=mcAborting ,
    Status=>axis2_power
    );

power3(
    Axis:=3 ,
    Enable:=power_eb ,
```

```
        EnablePositive:=TRUE ,
        EnableNegative:=TRUE ,
        BufferMode:=mcAborting ,
        Status=>axis3_power
        );

    vel1(
        Axis:=1 ,
        Execute:= vel1_execute ,
        Velocity:=30 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        Direction:=1 ,
        BufferMode:=mcAborting ,
        InVelocity=> vel1_invelocity ,
        Busy=> vel1_busy ,
        Active=> vel1_active ,
        CommandAborted=> vel1_abort
        );

    gearin1(
        Master:=1 ,
        Slave:=2 ,
        Execute:=gearin1_execute ,
        RatioNumerator:= 1 ,
        RatioDenominator:=1 ,
        MasterValueSource:=0 ,
        Acceleration:=100 ,
        Deceleration:=100 ,
        Jerk:=1000 ,
        BufferMode:=mcAborting ,
        InGear=>gearIn1_Ingear ,
        Busy=>gearIn1_busy ,
        Active=>gearIn1_busy ,
        CommandAborted=>gearIn1_abort
        );

    rotparset1(
        Execute:=rotparset1_execute ,
        KnifeRadius:=10 ,
        KnifeNum:=1 ,
        FeedRadius:=20 ,
        CutLength:=200 ,
        SyncStartPos:=190 ,
        SyncStopPos:=10 ,
        ID:=1 ,
        Done=>rotparset1_done ,
        Busy=>rotparset1_busy
        );

    rotin1(
        Execute:=rotin1_execute ,
```

```
    KnifeAxis:=3 ,
    FeedAxis:=1 ,
    ID:=1 ,
    Done=>rotin1_done ,
    Busy=>rotin1_busy
    );

rotout1(
    Execute:=rotout1_execute ,
    KnifeAxis:=3 ,
    ID:=1 ,
    Done=> rotout1_done ,
    Busy=> rotout1_busy
    );
```

- **Program description**

When gearin1_execute becomes TRUE after axis is enabled, virtual servo axis 1 and feed axis 2 establish a 1:1 electronic gear relationship.

When rotparset1_execute becomes TRUE, the parameters related to rotary cutting are set, and when the parameter setting is completed and rotin1_execute becomes TRUE, the virtual servo axis 1 and the knife axis 3 are established in a rotary cutting relations

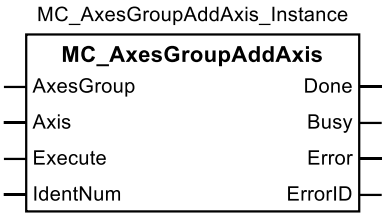
When vel1_execute becomes TRUE, virtual servo axis 1 executes the velocity command and feed axis 2 and knife axis 3 follow virtual servo axis 1. A rotary cut relationship is indirectly established for feed axis 2 and knife axis 3. So that user can cut the length accurately and to facilitate positional corrections to the feed axes.

When rotout1_execute becomes TRUE, the knife axis runs to the position before the rotary relationship is established and stops, rotout1_done becomes TRUE, the rotary relationship is released, and the knife axis no longer follows the feed axis.

Chapter7 Axes Group

7.1 MC_AxesGroupAddAxis (designated axis number in the axis group)

MotionControl_Designated axis number in the axis group, preparation for executing the axis group instruction.
Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupAddAxis	Designated axis number in the axis group	FB		MC_AxesGroupAddAxis_Instance (AxesGroup, Axis:=parameter , Execute:=parameter, IdentNum :=parameter, Done=> parameter, Busy=> parameter, Error=> parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Axis	Axis number	USINT	Depend on model	Required field	Specify axis number in the axis group
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge
IdentNum	Logical axis number in axis group	USINT	1~8	Required field	Logic axis number in axis group

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one cycle, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

■ Function description

- **Basic function description**

➢ This instruction is used to assign a designated axis to a designated axis group and establish a correspondence with the logical axis number in the axis group, in preparation for controlling the designated axis with axis group motion instructions. The designated axis is determined by the input variable 'Axis 'the designated axis group is determined by the input variable 'AxesGroup' and the logical axis number in the axis group is determined by the input variable 'IdentNum'. After the execution of this instruction, a correspondence is established between the designated axis and the logical axis in the axis group.

➢ Within the axis group, there are fixed logical axis numbers ranging from 1 to 8. When executing axis group motion instructions, each logical axis is assigned a corresponding position. Through this instruction, logical axis 1 can be designated to be the axis configured as axis 1 in the software, or it can be designated as axis 2. When designated as axis 1, logical axis 1 controls the motion of axis 1; when designated as axis 2, logical axis 1 controls the motion of axis 2.

➢ The following table shows the establishment of the correspondence between logical axis numbers and designated axes through the execution of 8 instances of this instruction. When controlling axis group motion commands, it is the logical axis that is controlled, and then, based on the correspondence between the logical axis number and the designated axis, the designated axis is controlled.

Case 1:

AxesGroup	1							
IdentNum	1	2	3	4	5	6	7	8
Axis	1	2	3	4	5	6	7	8
AxesGroup	1							
IdentNum	1	2	3	4	5	6	7	8
Axis	2	1	4	3	7	6	8	5

- **Instruction completion timing**

After establishing a corresponding relationship between the designated axis and the logical axis number in the axis group, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

This instruction can be re-executed when the axis group is not running, and when Execute changes from FALSE to TRUE again, it can be re-executed. After the axis group runs, executing this instruction will result in an error. After the axis group runs, if users want to restart this command, they need to first use MC_ The AxesGroupEnable command unloads the axis group and executes the instruction again.

- **Execute this instruction while other instructions are in processing**

When the axis group is not running, this instruction can be executed by executing MC_AxesGroupEnable instruction to solve the running status of axes groups. When other motion instructions are executed, executing this instruction has no effect on the execution of other instructions.

- **Execute other instructions while this instruction is in processing**

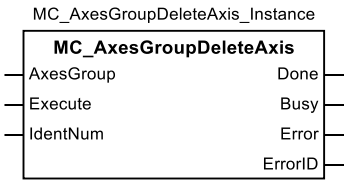
When the instruction MC_AxesGroupAddAxis is executed, execute MC_ AxesGroupDeleteAxis or MC_ The AxesGroupDeleteAllAxis instruction to unlock the correspondence between the designated axis and the logical axis number in the axis group. Executing other instructions has no impact on the execution of that instruction.

- **Abnormality elimination**

When the instruction is executing, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.2 MC_AxesGroupDeleteAxis (delete the number of the logical axis in the axis group)

Delete the number of the designated logical axis in the designated axis group. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupDeleteAxis	Delete the number of the logical axis in the axis group	FB		MC_AxesGroupDeleteAxis _Instance (AxesGroup :=parameter, Execute:=parameter , IdentNum:=parameter, Done=> parameter , Busy => parameter, Error => parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge
IdentNum	Logical axis number in axis group	USINT	1~8	Required field	Logical axis number in axis group

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE and after one cycle, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

■ Function description

- **Basic function description**
  - ➢ This instruction is used to delete a designated logical axis from a designated axis group. The designated axis group is determined by the input variable AxesGroup, and the logical axis number within the group is determined by the input variable IdentNum. After this instruction is executed, the designated logical axis cannot be controlled through axis group motion instructions.

- If a relationship between the designated axis and the logical axis number in the axis group was established using the MC_AxesGroupAddAxis instruction, that relationship will persist even if the designated axis executes other single-axis instructions. However, this instruction can be used to cancel the established relationship.

- **Instruction completion timing**

  After the specified designated axis is removed from the designated axis group, the instruction is completed, and 'Done' changes from FALSE to TRUE.

- **Re-execute the instruction**

  This instruction can be re-executed when the axis group is not running. When "Execute" changes from FALSE to TRUE again, it can be re-executed. If the axis group is running, an error will occur when this instruction is executed. If users want to re-execute this instruction after the axis group is running, they need to first disable the axis group operation using the MC_AxesGroupEnable instruction, and then execute this instruction.

- **Execute this instruction while other instructions are in processing**

  When the axis group is not running, this instruction can be executed by executing MC_ The AxesGroupEnable instruction to solve the running status of axis groups. When other motion instructions are executed, executing this instruction has no effect on the execution of other instructions.

- **Execute other instructions while this instruction is in processing**

  When this instruction is in processing, executing MC_The AxesGroupAddAxis instruction to re-establishes the logical axis number in the axis group, and executing other instructions has no effect on the execution of this instruction.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.3 MC_AxesGroupDeleteAllAxis (delete all logical axis numbers in the axis group)

Delete all logical axis numbers in the axis group. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupDeleteAllAxis	Delete all logical axis numbers in the axis group	FB		MC_AxesGroupDeleteAllAxis _Instance (AxesGroup:=parameter , Execute:=parameter , Done => parameter, Busy => parameter, Error=> parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

➢ This instruction is used to remove all logical axis numbers from the specified axis group. The specified axis group is specified by the input variable AxesGroup. After the execution of this command, all logical axes cannot be controlled by axis group motion instructions anymore.

➢ Specify the logical axis number in the axis and axis group through MC_ After the AxesGroupAddAxis instruction establishes a corresponding relationship, the relationship will persist even after the specified axis executes other single-axis instructions. The established relationship can be canceled through this instruction.

- **Instruction completion timing**

  After all axis numbers are removed from the specified axis group, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

  This instruction can be re-executed when the axis group is not running, and when Execute changes from FALSE to TRUE again, it can be re executed. After the axis group runs, executing this instruction will result in an error. After the axis group runs, if users want to restart this command, they need to first use MC_ The AxesGroupEnable instruction unloads the axis group and executes the instruction again.

- **Execute this instruction while other instructions are in processing**

  This instruction can be executed when the axis group is not running, and the axis group running state is released by executing the MC_AxesGroupEnable instruction. When other motion instructions are executed, the execution of this instruction has no effect on the execution of other instructions.

- **Execute other instructions while this instruction is in processing**

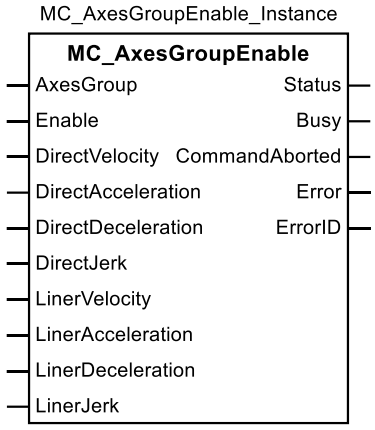
  When this instruction is in processing, executing MC_The AxesGroupAddAxis instruction to re-establishes the logical axis number in the axis group, and executing other instructions has no effect on the execution of this instruction.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.4 MC_AxesGroupEnable (axis group enable and disable)

Control the axis group into the running state or release the running state. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupEnable	Axis group enable and disable	FB		MC_AxesGroupAddAxis_Instance (AxesGroup:=parameter , Enable:=parameter , DirectVelocity:=parameter , DirectAcceleration:=parameter , DirectDeceleration :=parameter , DirectJerk :=parameter, LinerVelocity:=parameter , LinerAcceleration:=parameter , LinerDeceleration :=parameter, LinerJerk:=parameter , Status => parameter, Busy => parameter, CommandAborted=> parameter , Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Enable	Enable	BOOL	TRUE or FALSE	FALSE	Set to TRUE to control the axis group to enter the running state Set to FALSE to release the axis from the running state.
MoveDirectVelocity	Fast positioning speed	ARRAY [1..8] OF LREAL	Positive number	Required field	Fast positioning speed
MoveDirectAcceleration	Fast positioning acceleration	ARRAY [1..8] OF LREAL	Positive number	Required field	Fast positioning acceleration
MoveDirectDeceleration	Fast positioning deceleration	ARRAY [1..8] OF LREAL	Positive number	Required field	Fast positioning deceleration
MoveDirectJerk	Fast positioning jerk	ARRAY [1..8] OF LREAL	Positive number	Required field	Fast positioning jerk
LinerVelocity	Linear interpolation speed	LREAL	Positive number	Required field	Reserved
LinerAcceleration	Linear interpolation acceleration	LREAL	Positive number	Required field	Reserved
LinerDeceleration	Linear interpolation deceleration	LREAL	Positive number	Required field	Reserved
LinerJerk	Linear interpolation jerk	LREAL	Positive number	Required field	Linear interpolation jerk

■ Output variable

Name	Meaning	Data type	Valid range	Description
Status	Axis group running state	BOOL	TRUE or FALSE	If this output variable is TRUE, it means that the state of the axis group is in the running state If the output variable is FALSE, it means that the state of the axis group is in the released state
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed
CommandAborted	Abort	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
CommandAborted	When the instruction is aborted by other instructions	◆ CommandAborted is TRUE , and Enable changes from TRUE to FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

- This instruction is used to control the specified axis group to enter the running state or to release the axis group from the running state. The input variable Enable is TRUE to put the specified axes into operation, and FALSE to deactivate the axes. The output variable Status is TRUE to indicate that the axis group is in operation and the axes are in DiscreteMotion; FALSE to indicate that the axis group is released from operation.
- If the status of a single axis in an axis group is in the standstill state, the instruction can be executed only if the input variable Enable is changed from FALSE to TRUE.
- When the Enable bit changes from TRUE to FALSE, the operation state of the axis group is released regardless of whether the axes in the axis group are running or not. When an axis group motion instruction controls an axis, the axis speed is reduced to 0 in one cycle, and the state of each axis in the group is changed to standstill. When the axis group motion instruction controls the axis operation, it is recommended to stop the axis group motion first and then release the axis running state.
- The axis group motion instruction can be executed only after this instruction is executed.
- After this instruction is executed, when an axis in an axis group executes a motion instruction that is not related to the axis group, the instruction will be interrupted, and the state of the axis that executes the motion instruction will be determined by the corresponding motion instruction, and the state of the other axes will be changed to standstill.

- **Instruction completion timing**

  The output variable Status of this instruction is TRUE to indicate that the status of the axis group enters the running state; FALSE indicates that the running state of the axis group is released.

- **Re-execute the instruction**

  When the input variable Enable is set to TRUE, control the specified axis group to enter the running state; When it is FALSE, release the running state of the axis group.

- **Execute this instruction while other instructions are in processing**

  Refer to the basic function description.

- **Execute other instructions while this instruction is in processing**

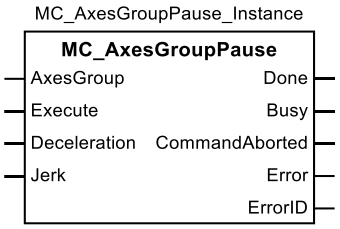
  Refer to the basic function description.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.5 MC_AxesGroupPause (pause the motion of the axis group)

Control the axis group into the running state or release the running state. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupPause	Pause the motion of the axis group	FB		MC_AxesGroupPause_Instance (AxesGroup :=parameter, Execute:=parameter , Deceleration:=parameter , Jerk:=parameter , Done=> parameter , Busy=> parameter , CommandAborted => parameter , Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge
Deceleration	Deceleration	LREAL	Positive number	Required field	Reserved
Jerk	Jerk	LREAL	Positive number	Required field	Reserved

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
CommandAborted	Abort	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
CommandAborted	When the instruction is aborted by other instructions	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

Function description

Basic Function description

- This instruction is used to pause the executing axis group movement. After the instruction is executed, use the value of the input variable that is executing the axis group motion instruction to slow down and stop. After the instruction is executed, the axis status is Discrete Motion (in positioning action). The instruction inputs variables Deceleration and Jerk as Reserved parameters, and the user input values are invalid.
- Axis group motion instruction MC_ MoveDirectAbsolute, MC_ MoveDirectRelative, MC_ MovelinearAbsolute, MC_ MovelinearRelative, MC_ MoveCircularAbsolute or MC_MoveCircularRelative is executed, MC_ The AxesGroupPause instruction can only be executed, otherwise executing this instruction will result in an error.
- After executing this instruction, if users want to continue executing the axis group motion instruction, they need to execute the instruction MC_ AxesGroupResume is released from pause.
- If users want to terminate the execution of the axis group instruction or if the axis in the axis group needs to execute other motion instructions, they need to first execute MC_ AxesExit instruction, then MC_ The AxesGroupEnable directive sets Enable to FALSE. Suggested execution sequence: Execute MC first_ AxesGroupPause instruction, then execute MC_ AxesGroupExit instruction, then MC_ The AxesGroupEnable directive sets Enable to FALSE.

Instruction completion timing

After the instruction is executed and the pause instruction is given, the instruction completes and Done changes from FALSE to TRUE. When Done is TRUE for this instruction does not mean that the axis has stopped. Whether the axis is stopped or not can be judged by the axis instruction speed or axis feedback speed in the axis structure.

Re-execute the instruction

Refer to the basic function description

Execute this instruction while other instructions are in processing

Refer to the basic function description

Execute other instructions while this instruction is in processing

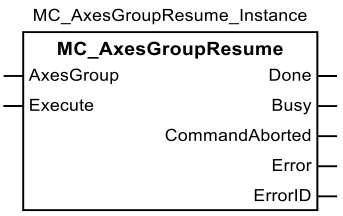
Refer to the basic function description

Abnormality elimination

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.6 MC_AxesGroupResume (resumption of Axis Group Motion)

Resume the remaining part of the axis group motion from the pause. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupResume	Resumption of Axis Group Motion	FB		MC_AxesGroupResume_Instance (AxesGroup:=parameter , Execute:=parameter , Done => parameter, Busy => parameter, CommandAborted=> parameter , Error => parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executing
CommandAborted	Abort	BOOL	TRUE or FALSE	Changes to TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
CommandAborted	When the instruction is aborted by other instructions	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

➢ This instruction is used to resume the axis group motion from a pause and continue the remaining part of the axis group motion. After the execution of this instruction, the axis group motion uses the values of the input variables of the executing axis group motion instruction for motion.

➢ The MC_AxesGroupPause instruction must be executed before this instruction can be executed, otherwise an error will occur when executing this instruction.

● Instruction completion timing

After the instruction is executed and the pause instruction is given, the instruction completes and Done changes from FALSE to TRUE.

● Re-execute the instruction

Refer to the basic function description

● Execute this instruction while other instructions are in processing

Refer to the basic function description

● Execute other instructions while this instruction is in processing

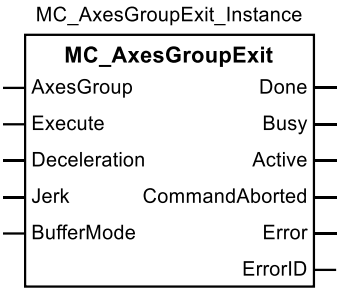
When this instruction is executed, non-axis group-related motion instructions can interrupt this instruction.

● Abnormality elimination

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.7 MC_AxesGroupExit (axis group motion emergency stop)

Resume the remaining part of the axis group motion from the pause. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupExit	Axis group motion emergency stop	FB		MC_AxesGroupExit_Instance (AxesGroup :=parameter, Execute:=parameter , Deceleration:=parameter , Jerk:=parameter , BufferMode:=parameter , Done=> parameter , Busy=> parameter , Active=> parameter , CommandAborted=> parameter, Error => parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge
Deceleration	Deceleration	LREAL	Positive number	Required field	Reserved
Jerk	Jerk	LREAL	Positive number	Required field	Reserved
BufferMode	Buffer mode	MC_Buffer_Mode	Reserved	—	Reserved

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when execution of the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Active	under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the instruction execution is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes from FALSE to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE
Active	When the instruction is started	◆ When Error becomes TRUE ◆ When CommandAborted becomes TRUE
CommandAborted	When the instruction is aborted by other instructions	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE

		◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

- ➢ This instruction is used to perform an emergency stop on an axis group motion that is being executed. When this instruction is executed, the axis will stop in one period during the process the axis state is in DiscreteMotion. It is not recommended to use this instruction when the axes in an axis group are running. It is recommended that the MC_AxesGroupPause instruction be executed first, followed by the MC_AxesGroupExit instruction, and then the MC_AxesGroupEnable instruction Enable is set to FALSE.
- ➢ This instruction can be executed only when the state of the axis group is in the running state (the state of the axis group is controlled to be in the running state by MC_AxesGroupEnable), otherwise, the execution of this instruction will report an error.
- ➢ After this instruction is executed, if users want to unlock the running state of the axis group, they can unlock the running state of the axis group by setting the MC_AxesGroupEnable instruction Enable to FALSE.

● Instruction completion timing

After this instruction executes, axis group will stop in one cycle, then the instruction is complete, and Done changes from FALSE to TRUE.

● Re-execute the instruction

Refer to the basic function description.

● Execute this instruction while other instructions are in processing

Refer to the basic function description.

● Execute other instructions while this instruction is in processing

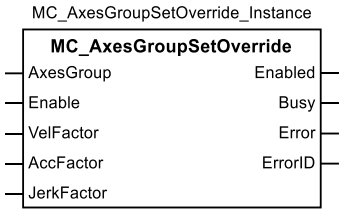
Refer to the basic function description.

● Abnormality elimination

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.8 MC_AxesGroupSetOverride (axis group speed ratio setting)

This instruction is used to change the target speed ratio for axis group motion instructions. Library : MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupSetOverride	Axis group speed ratio setting	FB		MC_AxesGroupSetOverride _Instance (AxesGroup:=parameter , Enable :=parameter , VelFactor:=parameter, AccFactor :=parameter , JerkFactor :=parameter , Enabled=> parameter , Busy => parameter , Error => parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Enable	Enable	BOOL	TRUE or FALSE	FALSE	When set to TRUE, the target speed ratio value takes effect; When set to FALSE, the target speed ratio value becomes “100%”.
VelFactor	Factor for axis velocity	LREAL	0~500	100	Target velocity ratio (Unit: %)
AccFactor	Factor for axis acceleration	LREAL	Positive number	Required field	Reserved
JerkFactor	Factor for axis Jerk	LREAL	Positive number	Required field	Reserved

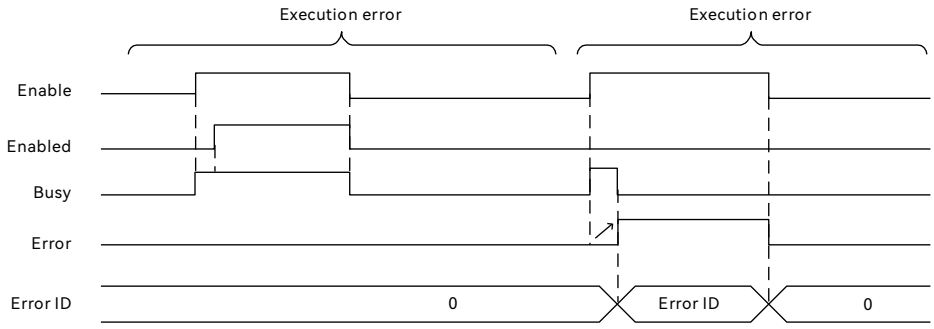
■ Output variable

Name	Meaning	Data type	Valid range	Description
Enabled	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to “instruction error code description” for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Enabled	When the instruction starts to execute .	◆ When Enable becomes FALSE ◆ When Error becomes TRUE
Busy	When Enable becomes TRUE	◆ Enable becomes FALSE ◆ Error becomes TRUE
Error	When an instruction input variable is illegal or an error occurs during instruction execution	◆ Enable changes from TRUE to FALSE

■ Output variable timing diagram



- **Separate execution**

  When Enable changes from FALSE to TRUE, Busy changes to TRUE at the same time, and Enabled changes to TRUE after one cycle, and the value of the VelFactor parameter for this instruction takes effect. When Enable becomes FALSE, Busy and Enabled become FALSE at the same time.

- **Execution error**

  When the value of the input variable of this instruction is not within the valid range, the Enable changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, and the value of ErrorID can be used to find the cause of the problem. When the instruction Enable changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

■ Function description

- **Basic Function description**

➢ This instruction is used to change the target speed factor of an axis group motion instruction. When the Enable of this instruction is TRUE, the target speed of the axis group motion instruction can be changed in real-time by changing the value of VelFactor of this instruction. The unit of VelFactor is a percentage, such as “50” means “50%”. The value of the VelFactor parameter ranges from 0 to 500, if the value is out of the range, the instruction will report an error.

➢ The instructions that can be used to change the target speed ratio when the axis group instruction is executed are shown in the table below.

MC_MovelinearRelative (Relative linear interpolation)	MC_MoveDirectRelative (Relative fast positioning)
MC_MovelinearAbsolute (Absolute linear interpolation)	MC_MoveDirectAbsolute (Absolute fast positioning)
MC_MoveCircularRelative (Relative circular interpolation)	
MC_MoveCircularAbsolute (Absolute circular interpolation)	

➢ After this instruction is executed, the new target Velocity is as follows:

New target velocity = The current axis group instruction’ s target velocity × Velocity ratio of the instruction

➢ If the Enable of this instruction is TRUE, the value of the input variable VelFactor of this instruction is changed, and acceleration or deceleration to the target speed is performed according to the input variables of the axis group motion control instruction. If the value of VelFactor is changed when the MC_MovelinearRelative instruction is executed, the target speed is accelerated or decelerated according to the set values of Acceleration and Deceleration of the input variables of the MC_MovelinearRelative instruction.

➢ When VelFactor is set to “0”, the target speed becomes “0”. To maintain the execution of the motion instruction while pausing the motion to control the axis, VelFactor can be set to “0” for this instruction. In this case, the axis state does not change.

- When the Enable of this instruction changes from TRUE to FALSE, the speed of the axis group changes from the current speed to the target velocity of the axis group motion instruction, which is equivalent to the case where the value of VelFactor is 100 at the time of execution of this instruction. The speed change of the axis group is accelerated or decelerated to the target speed according to the input variable of the axis group instruction of the current control axis. If the value of VelFactor is 200 and Enable is TRUE when the MC_MovelinearRelative instruction is executed and Enable changes from TRUE to FALSE, the speed of the axis group is accelerated or decelerated to the target velocity in accordance with the input variable Acceleration of the MC_MovelinearRelative instruction, Deceleration will be accelerated or decelerated to the target speed set by the MC_MovelinearRelative instruction according to the set values of the input variables Acceleration and Deceleration of the MC_MovelinearRelative instruction.

  - **Re-execute the instruction**

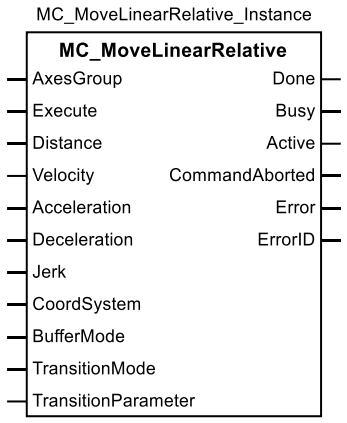
    This instruction is executed continuously when Enable is TRUE.

  - **Abnormality elimination**

    When the instruction is executed, if the input variable value of the instruction is not within the valid range or does not meet the execution requirements, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

7.9 MC_MoveLinearRelative (relative linear interpolation)

Control multiple axes to start and stop simultaneously, perform linear interpolation, and specify the distance as a relative.Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveLinearRelative	Relative linear interpolation	FB		MC_MoveLinearRelative_Instance (AxesGroup:=parameter, Execute:=parameter , Distance:=parameter , Velocity:=parameter , Acceleration:=parameter , Deceleration:=parameter , Jerk:=parameter , CoordSysytem :=parameter, BufferMode :=parameter, TransitionMode:=parameter , TransitionParameter :=parameter, Done => parameter, Busy=> parameter , Active=> parameter , CommandAborted => parameter, Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	When set to TRUE, the target speed ratio value takes effect; When set to FALSE, the target speed ratio value becomes “100%”.
Distance	Moving distane	LREAL	Positive number, Negative value, 0	0	Specify the distance to move with the current position as the reference. *1 (Unit: Travel unit) *2
Velocity	Target velocity	LREAL	Positive number	Required field	Target velocity*1 (Unit: Travel unit/s) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Target acceleration*1 (Unit: Travel unit/s^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Target deceleration*1 (Unit: Travel unit/s^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Target Jerk*1 (Unit: Travel unit/s^3) *2
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved
BufferMode	Buffer mode	MC_Buffer_Mode	1: mcBuffered 3: mcBlendingPrevious	Required field	Setting the buffer mode between two instructions *3 1: Buffered 3: Buffered at current instruction target velocity.

TransitionMode	Transition mode	MC_Transition_Mode	0: mcTMNone 2: TMCons-tantVelocity 3: mcTMCorner-Distance	Required field	Setting the connection between the interpolation instruction track of the current control axis and the buffered interpolation instruction track 0: Velocity decreases to 0 when the position is reached. 2: Transit at the specified speed 3: Transit at the set additional angle
TransitionParameter	Additional angle	LREAL	Positive number, 0	0	Transition parameters are set when TransitionMode is selected to transit at an additional angle.

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions"*.

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

*3: For details of BufferMode, please refer to "*Buffer mode during multi-starting of the same axis*".

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the positioning is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE ◆ CommandAborted becomes TRUE from FALSE
Active	When the instruction is started	◆ Done becomes TRUE ◆ Error becomes TRUE ◆ CommandAborted becomes TRUE from FALSE
CommandAborted	When the instruction is aborted by other instructions	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE ◆ Instruction has been executed and Execute becomes FALSE, when the instruction is aborted by other instructions during the execution, Error becomes TRUE for one period and then becomes FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE ◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.

■ Function description

● Basic function description

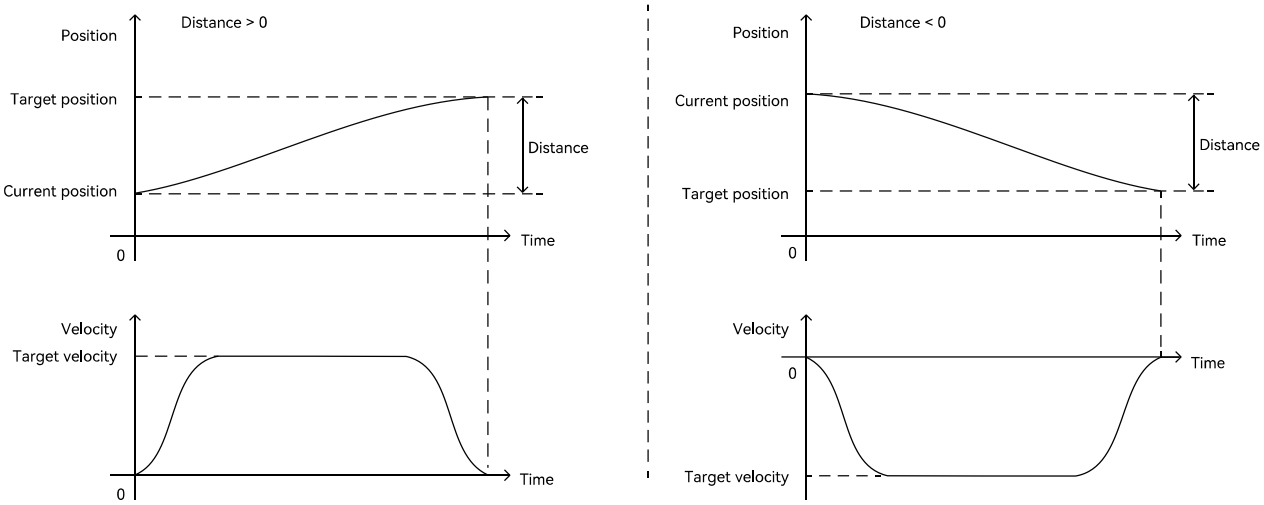
This instruction controls multiple axes to start and stop at the same time so that they can perform linear interpolation movements. During the process, up to 8 axes can move at the same time. The distance moved by each axis is set individually by the input variable Distance, the input variable Velocity is the synthesized velocity of all axes, and the input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. This instruction moves according to the values set by the input variables when Execute changes from FALSE to TRUE.

● Target velocity

The input variable Velocity is the synthesized velocity of all axes. The relationship between the synthesized velocity and the velocity of each axis is: the sum of the squares of the synthesized velocity equals the sum of the squares of the velocity of each axis. The input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. The relationship between the synthesized acceleration and deceleration and the acceleration and deceleration of each axis is: the sum of the squares of the synthesized acceleration and deceleration equals the sum of the squares of the acceleration and deceleration of each axis.

● Moving distance

- The target position of an axis is determined by the current position of each axis in the axis group and the value of Distance, i.e., Target Position = Current Position + Distance. The value of Distance can be positive, 0, or negative, and the values of Velocity, Acceleration, and Deceleration can only be positive.
- When axis in standstill state while this instruction is executed. If the value of Distance is greater than 0, the axis runs in the forward direction; if the value of Distance is less than 0, the axis runs in the reverse direction.
- When the value of Distance is equal to 0, this instruction does not control the axis operation, and the Done of the instruction becomes TRUE after one cycle of this instruction.



● Instruction completion timing

When the command position of each axis in the axis group reaches the set distance, the instruction is completed and Done changes from FALSE to TRUE.

● Re-execute the instruction

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

- **Execute this instruction while other instructions are in processing**

This instruction is activated when other axis group motion instructions are executed. How this instruction and other motion instructions are buffered is jointly determined by the values of BufferMode and TransitionMode. The detailed description is shown in the following table.

BufferMode	TransitionMode	Description
mcBuffered (Await)	mcTMNone (Velocity becomes 0 when position is reached)	Wait for the execution of the current axis group instruction to be completed (when the target position is reached, the velocity becomes 0) and switch to the buffered axis group instruction control axis.
mcBlendingPrevious (Buffered at the target speed of the current instruction)	mcTMConstantVelocity (Transition at a specified velocity) This mode is generally used for curve fitting or continuous interpolation of multi-line segments. It is recommended that more than one instruction be triggered together using the same execution bit, and the transition speed is the target speed of the first executed axis group instruction.	When the target position of the current axis group instruction is reached, the axis group speed is the target speed of the current axis group instruction, and then the target speed of the current axis group instruction is used as the speed for the transition, and then it is executed with the input parameter set by the buffered axis group instruction.
	mcTMCornerDistance (Transit at an additional angle)	The moment of completion of the target position of the current axis group instruction is the target position of the current axis group instruction minus the attached additive angle position set by mcTMCornerDistance. After the execution of the current axis group instruction is completed, the buffered axis group instruction is executed immediately. The buffered axis group instruction first moves with the additional angle set by mcTMCornerDistance, and then moves with the parameter set by the buffered axis group instruction.

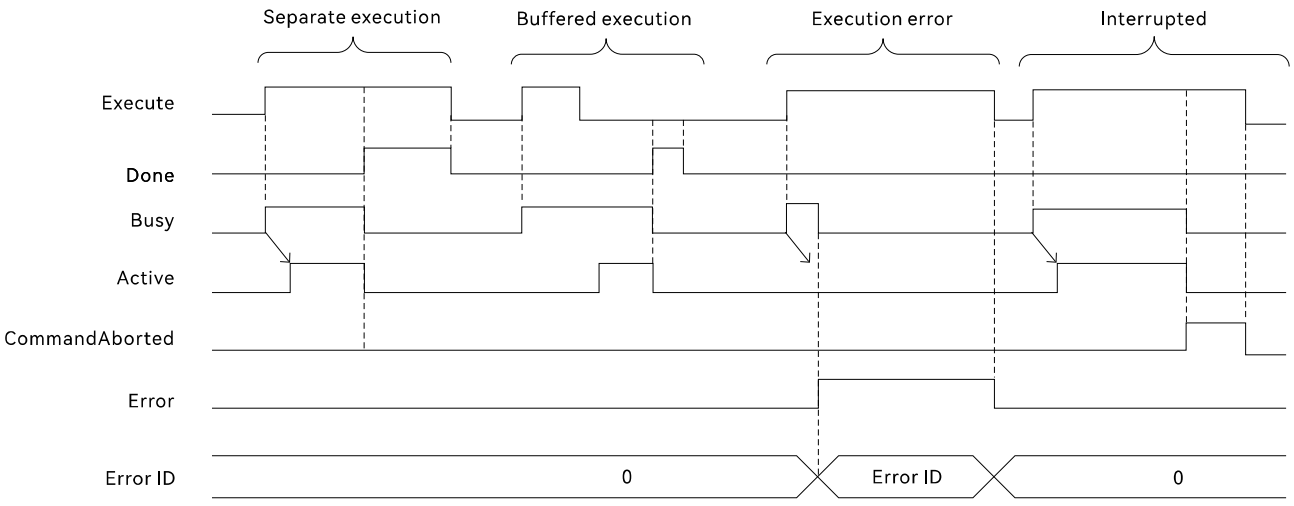
- **Execute other instructions while this instruction is in processing**

When this instruction is executed, other axis group motion instructions are started, and how other axis group motion instructions and this instruction are buffered is jointly determined by the BufferMode and TransitionMode values of other motion instructions. When other motion instructions are buffered with this instruction, the other instructions are executed when the Done instruction becomes TRUE. If there is no BufferMode pin for other motion instructions, the instruction is normally aborted. When this instruction is executed, the execution of other non-axis group-related motion instructions will abort this instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input variable of the executed instruction; the other axes in the axis group will enter into the state of standstill, and the axes speed will be reduced to 0 for one period.

- **Abnormality elimination**

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. If an abnormality (such as an axis alarm, etc.) is encountered when this instruction is executed, the instruction will also report an error and the axis will stop immediately.

■ Output variable timing diagram



- **Separate execution**

When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next cycle; when Distance is reached, Done becomes TRUE, and Busy and Active become FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Buffered execution**

The instruction is executed when the other instruction controls the axis (BufferMode is not in mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction is completed.

- **Execution error**

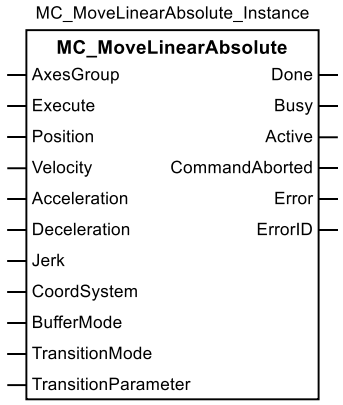
When the value of the input variable of this instruction is not within the valid range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Interrupted**

When the execution of this instruction is aborted by other instructions, the instruction CommandAborted becomes TRUE and Busy and Active become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

7.10 MC_MoveLinearAbsolute (absolute linear interpolation)

Controls multiple axes to start and stop at the same time to perform linear interpolation motions, with the specified position as an absolute value. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveLinearAbsolute	Absolute linear interpolation	FB		MC_MoveLinearAbsolute _Instance (AxesGroup :=parameter , Execute :=parameter, Position :=parameter, Velocity:=parameter , Acceleration:=parameter , Deceleration:=parameter , Jerk:=parameter , CoordSysytem :=parameter, BufferMode :=parameter, TransitionMode:=parameter , TransitionParameter , Done=> parameter , Busy => parameter, Active => parameter, CommandAborted => parameter, Error => parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	When set to TRUE, the target speed ratio value takes effect; When set to FALSE, the target speed ratio value becomes “100%”.
Distance	Moving distane	LREAL	Positive number, Negative value, 0	0	Specify the distance to move with the current position as the reference. *1 (Unit: Travel unit) *2
Velocity	Target velocity	LREAL	Positive number	Required field	Target velocity*1 (Unit: Travel unit/s) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Target acceleration*1 (Unit: Travel unit/s^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Target deceleration*1 (Unit: Travel unit/s^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Target Jerk*1 (Unit: Travel unit/s^3) *2
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved
BufferMode	Buffermode	MC_Buffer_Mode	1: mcBuffered 3: mcBlendingPrevious	Required field	Setting the buffer mode between two instructions *3 1: Buffered 3: Buffered at current instruction target velocity
TransitionMode	Transition mode	MC_Transition_Mode	0: mcTMNone 2: TMCons-tantVelocity 3: mcTMCorner-Distance	Required field	Setting the connection between the interpolation instruction track of the current control axis and the buffered interpolation instruction track 0: Velocity decreases to 0 when the

					position is reached. 2: Transit at the specified speed 3: Transit at the set additional angle
TransitionParameter	Additional angle	LREAL	Positive number, 0	0	Transition parameters are set when TransitionMode is selected to transit at an additional angle

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions"*.

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

*3: For details of BufferMode, please refer to "*Buffer mode during multi-starting of the same axis*".

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Aborted	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE and then to FALSE one period later
Busy	When Execute changes to TRUE	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
Active	When the instruction is started	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
CommandAborted	When the instruction is aborted by another instruction	◆ When CommandAborted is TRUE, Enable becomes TRUE from FALSE ◆ Instruction has been executed and Execute becomes FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period and then becomes FALSE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE ◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.

■ Function description

● Basic function description

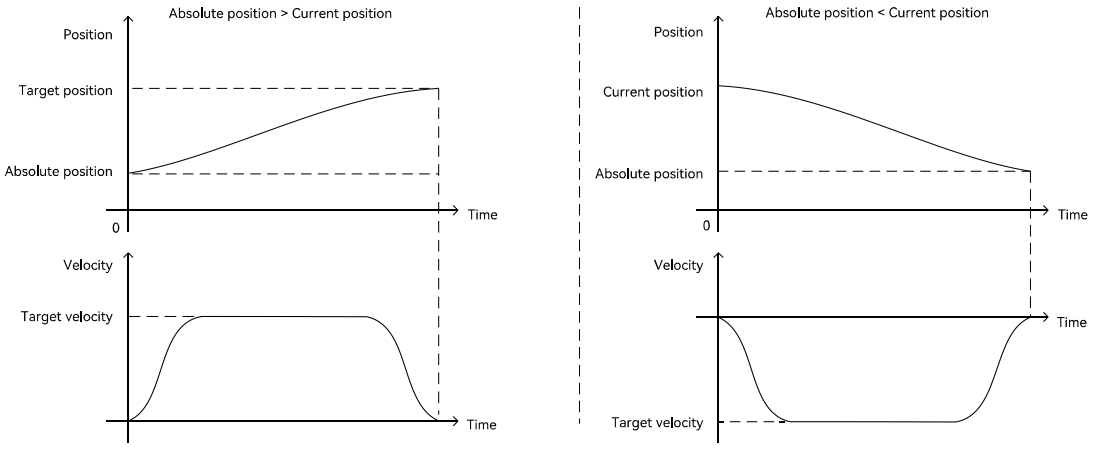
This instruction controls multiple axes to start and stop at the same time so that they can perform linear interpolation movements. During the process, up to 8 axes can move at the same time. The distance moved by each axis is set individually by the input variable Distance, the input variable Velocity is the synthesized velocity of all axes, and the input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. This instruction moves according to the values set by the input variables when Execute changes from FALSE to TRUE.

- **Target Velocity**

  The input variable Velocity is the synthesized velocity of all axes. The relationship between the synthesized velocity and the velocity of each axis is: the sum of the squares of the synthesized velocity equals the sum of the squares of the velocity of each axis. The input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. The relationship between the synthesized acceleration and deceleration and the acceleration and deceleration of each axis is: the sum of the squares of the synthesized acceleration and deceleration equals the sum of the squares of the acceleration and deceleration of each axis.

- **Absolute position**

  - The value of the input variable Position represents the absolute position using the 0 point position as a reference point. The value can be positive, 0, or negative. Values for Velocity, Acceleration, Deceleration, and Jerk can only be positive.
  - This instruction is executed when the axis is stationary, and the position curve and speed curve when the value of Position is greater than the current position and less than the current position are shown in the figure below. When the value of Position is greater than the current position, the axis will run in the forward direction; when the value of Position is less than the current position, the axis will run in the reverse direction.
  - When the value of Position is equal to the current position, the instruction does not control the axis operation, and the Done of the instruction becomes TRUE one scan cycle after executing the instruction.



- **Instruction completion timing**

  When the command position of each axis in the axis group reaches the set position, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed. When the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

- **Execute this instruction while other instructions are in processing**

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

BufferMode	TransitionMode	Description
mcBuffered (Await)	McTMNone (Velocity becomes 0 when position is reached)	Wait for the execution of the current axis group instruction to be completed (when the target position is reached, the velocity becomes 0) and switch to the buffered axis group instruction control axis
mcBlendingPrevious (Buffered at the target speed of the current instruction)	mcTMConstantVelocity (Transition at a specified velocity) This mode is generally used for curve fitting or continuous interpolation of multi-line segments, it is recommended that more than one instruction be triggered together using the same execution bit, and the transition speed is the target speed of the first executed axis group instruction.	When the target position of the current axis group instruction is reached, the axis group speed is the target speed of the current axis group instruction, and then the target speed of the current axis group instruction is used as the transition speed for the transition, and then it is executed with the input parameter set by the buffered axis group instruction
	mcTMCornerDistance (Transit at an additional angle)	The moment of completion of the target position of the current axis group instruction is the target position of the current axis group instruction minus the attached additive angle position set by mcTMCornerDistance. After the execution of the current axis group instruction is completed, the buffered axis group instruction is executed immediately. The buffered axis group instruction first moves with the additional angle set by mcTMCornerDistance, and then moves with the parameter set by the buffered axis group instruction

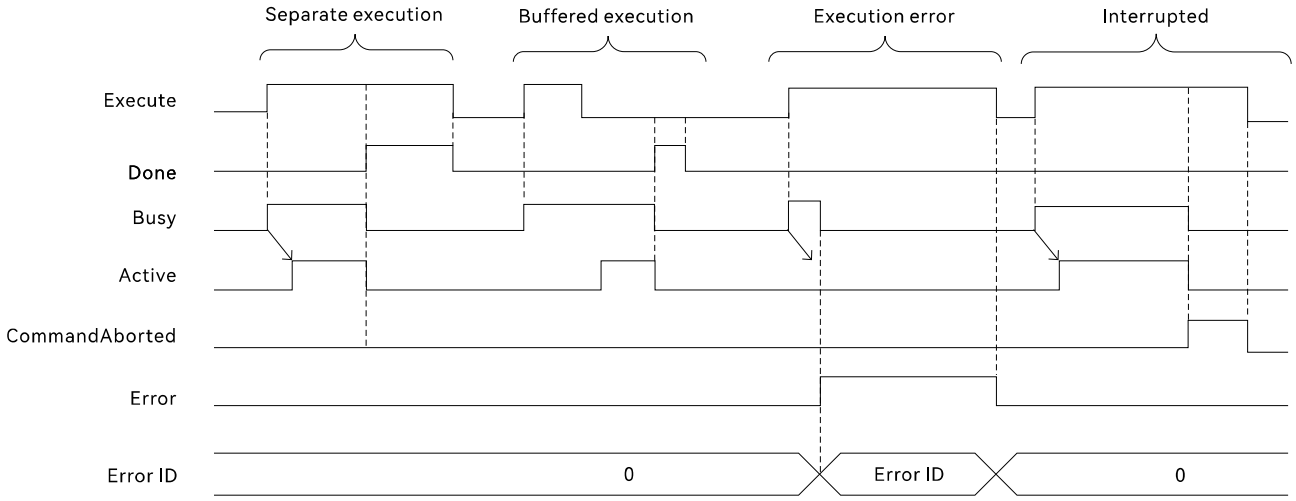
- **Execute other instructions while this instruction is in processing**

When this instruction is executed, other axis group motion instructions are started, and how other axis group motion instructions and this instruction are buffered is jointly determined by the BufferMode and TransitionMode values of other motion instructions. When other motion instructions are buffered with this instruction, the other instructions are executed when the Done instruction becomes TRUE. If there is no BufferMode pin for other motion instructions, the instruction is normally interrupted. When this instruction is executed, the execution of other non-axis group-related motion instructions will interrupt this instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input variable of the executed instruction; the other axes in the axis group will enter into the state of standstill, and the axes speed will be reduced to 0 for one period.

- **Abnormality elimination**

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. If an abnormality (such as an axis alarm, etc.) is encountered when this instruction is executed, the instruction will also report an error and the axis will stop immediately.

- **Output variable timing diagram**



- **Separate execution**

When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When Distance is reached, Done becomes TRUE, and Busy and Active become FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Buffered execution**

The instruction is executed when the other instruction controls the axis (BufferMode is not in mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction is completed.

- **Execution error**

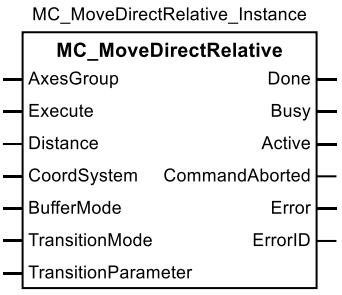
When the value of the input variable of this instruction is not within the valid range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Interrupted**

When the execution of this instruction is interrupted by other instructions, the instruction CommandAborted becomes TRUE and Busy and Active become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

7.11 MC_MoveDirectRelative (relative fast positioning)

Controls multiple axes to start simultaneously, and stopping is determined by the travel distance specified for each axis, which is a relative. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveDirectRelative	Individual relative positioning	FB		MC_MoveDirectRelative_Instance (AxesGroup:=parameter , Execute:=parameter , Distance :=parameter, CoordSysytem:=parameter , BufferMode:=parameter , TransitionMode:=parameter , TransitionParameter:=parameter , Done => parameter , Busy => parameter , Active=> parameter , CommandAborted => parameter, Error=> parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	When set to TRUE, the target speed ratio value takes effect; When set to FALSE, the target speed ratio value becomes “100%”.
Distance	Moving distane	LREAL	Positive number, Negative value, 0	0	Specify the distance to move with the current position as the reference. *1 (Unit: Travel unit) *2
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved
BufferMode	Buffermode	MC_Buffer_Mode	1: mcBuffered 3: mcBlendingPrevious	Required field	Setting the buffer mode between two instructions *3 1: Buffered 3: Buffered at current instruction target velocity.
TransitionMode	TransitionMode	MC_Transition_Mode	0: mcTMNone 2: TMCons-tantVelocity 3: mcTMCorner-Distance	Required field	Setting the connection between the interpolation instruction track of the current control axis and the buffered interpolation instruction track 0: Velocity decreases to 0 when the position is reached. 2: Transit at the specified speed 3: Transit at the set additional angle
TransitionParameter	Additional angle	LREAL	Positive number, 0	0	Transition parameters are set when TransitionMode is selected to transit at an additional angle.

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *“Parameter description of motion control instructions”*.

*2: For details of the instruction units, please refer to *“Parameter unit of motion control instructions”*.

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction execution is complete
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Abort	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When the positioning is completed	◆ When Done is TRUE, and Execute changes from TRUE to FALSE ◆ When the instruction is executed and the Execute is FALSE, Done becomes TRUE for one period and then becomes FALSE
Busy	When Execute changes to TRUE	◆ Done becomes TRUE ◆ Error becomes TRUE ◆ CommandAborted becomes TRUE from FALSE
Active	When the instruction is started	◆ Done becomes TRUE ◆ Error becomes TRUE ◆ CommandAborted becomes TRUE from FALSE
CommandAborted	When the instruction is aborted by other instructions	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE ◆ Instruction has been executed and Execute becomes FALSE, when the instruction is aborted by other instructions during the execution, Error becomes TRUE for one period and then becomes FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE ◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.

■ Function description

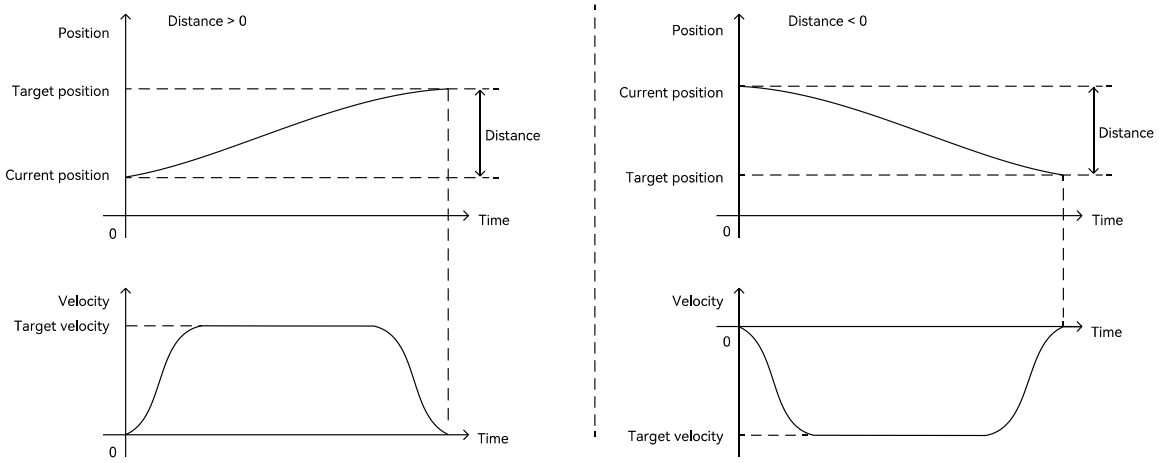
● Basic function description

This instruction controls multiple axes to start and stop at the same time so that they can perform linear interpolation movements. During the process, up to 8 axes can move at the same time. The distance moved by each axis is set individually by the input variable Distance, the input variable Velocity is the synthesized velocity of all axes, and the input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. This instruction moves according to the values set by the input variables when Execute changes from FALSE to TRUE.

● Moving distance

➢ The target position of an axis is determined by the current position of each axis in the axis group and the value of Distance, i.e., Target Position = Current Position + Distance; the value of Distance can be positive, 0, or negative; the values of Velocity, Acceleration, Deceleration, and Jerk can only be positive.

- This instruction is executed when the axis is stationary, and the position curve and speed curve when the value of Distance is greater than 0 and less than 0 are shown in the figure below. If the value of Distance is greater than 0, the axis operates in the forward direction; if the value of Distance is less than 0, the axis operates in the reverse direction.
- When the value of Distance is 0, the instruction does not control the axis operation, and the Done of the instruction becomes TRUE after one scan period of executing the instruction.
- When the value of Distance is equal to 0, the instruction does not control the axis operation, and the Done of the instruction becomes TRUE after one scan period of executing the instruction.



- **Instruction completion timing**

When the instruction position of each axis in the axis group reaches the set distance, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed. When the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

- **Execute this instruction while other instructions are in processing**

This instruction is activated when other axis group motion instructions are executed, and only mcBuffered can be selected for this instruction BufferMode, and only mcTMNone (velocity decreases to 0 when position is reached) can be selected for TransitionMode.

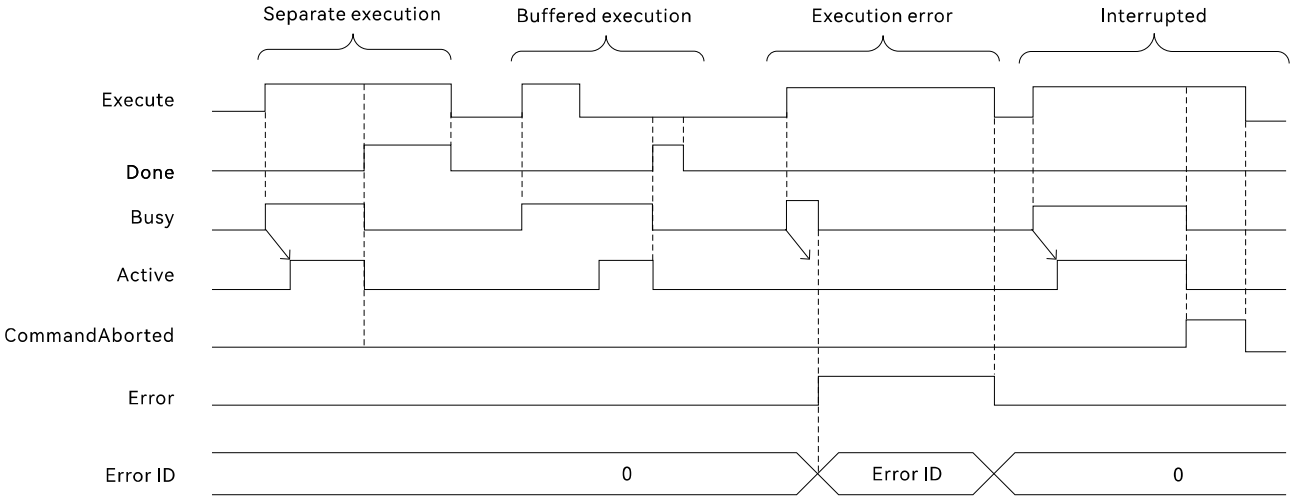
- **Execute other instructions while this instruction is in processing**

When this instruction is executed, other axis group motion instructions are started, and how other axis group motion instructions and this instruction are buffered is determined by the BufferMode and TransitionMode values of the other motion instructions. When other motion instructions are buffered with this instruction, the other instructions are executed when the Done instruction becomes TRUE. If there is no BufferMode pin for other motion instructions, the instruction is normally interrupted. When this instruction is executed, the execution of other non-axis group-related motion instructions will interrupt this instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input variable of the executed instruction; the other axes in the axis group will enter into the state of standstill, and the axes' speed will be reduced to 0 for one period.

- **Abnormality elimination**

  When this instruction is executed, if the value of the input variable of the instruction is out of the valid range or does not satisfy the execution conditions of the instruction, the instruction Error becomes TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. If the instruction encounters an abnormality (e.g., an axis alarm), the instruction will also report an error and the axis will stop immediately.

Output variable timing diagram



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next cycle; when Distance is reached and positioning is completed, Done becomes TRUE, and Busy and Active become FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Buffered execution**

  The instruction is executed when the other instruction controls the axis (BufferMode is not in mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction is completed.

- **Execution error**

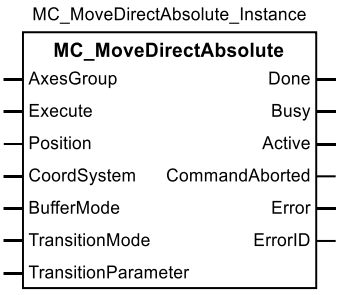
  When the value of the input variable of this instruction is not within the valid range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Interrupted**

  When the execution of this instruction is aborted by other instructions, the instruction CommandAborted becomes TRUE and Busy and Active become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

7.12 MC_MoveDirectAbsolute (absolute fast positioning)

Controls multiple axes to start simultaneously. The stopping of multiple axes is determined by the travel distance specified for each axis, which is an absolute value. Library：MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveDirectAbsolute	Individual absolute positioning	FB		MC_MoveDirectAbsolute _Instance (AxesGroup:=parameter , Execute :=parameter, Position:=parameter , CoordSysytem :=parameter, BufferMode :=parameter, TransitionMode:=parameter , TransitionParameter :=parameter, Done => parameter, Busy=> parameter , Active => parameter, CommandAborted => parameter , Error => parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	When set to TRUE, the target speed ratio value takes effect; When set to FALSE, the target speed ratio value becomes “100%”
Distance	Moving distane	LREAL	Positive number, Negative value, 0	0	Specify the distance to move with the current position as the reference *1 (Unit: Travel unit) *2
Velocity	Target velocity	LREAL	Positive number	Required field	Target velocity*1 (Unit: Travel unit/s) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Target acceleration*1 (Unit: Travel unit/s^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Target deceleration*1 (Unit: Travel unit/s^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Target Jerk*1 (Unit: Travel unit/s^3) *2
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved
BufferMode	Buffermode	MC_Buffer_Mode	1: mcBuffered 3: mcBlendingPrevious	Required field	Setting the buffer mode between two instructions *3 1: Buffered 3: Buffered at current instruction target velocity
TransitionMode	Transition mode	MC_Transition_Mode	0: mcTMNone 2: TMConstantVelocity 3: mcTMCornerDistance	Required field	Setting the connection between the interpolation instruction track of the current control axis and the buffered interpolation instruction track 0: Velocity decreases to 0 when the position is reached. 2: Transiti at the specified speed 3: Transit at the set additional angle

TransitionParameter	Additional angle	LREAL	Positive number, 0	0	Transition parameters are set when TransitionMode is selected to transit at an additional angle

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions"*.

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

*3: For details of BufferMode, please refer to "*Buffer mode during multi-starting of the same axis*".

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control
CommandAborted	Abort	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

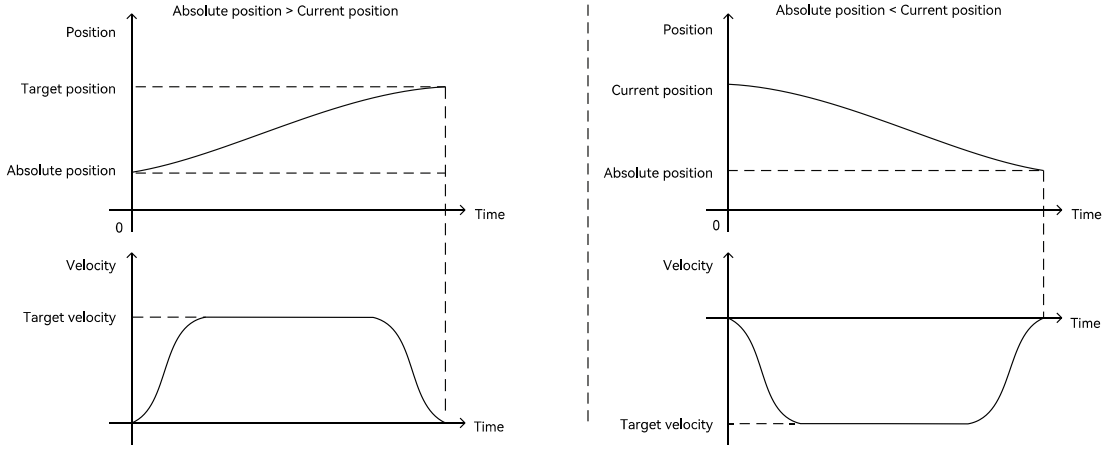
Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE. ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE for one period and then becomes FALSE
Busy	When Execute changes to TRUE	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
Active	When the instruction is started	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
CommandAborted	When the instruction is aborted by other instruction	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE ◆ Instruction has been executed and Execute becomes FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period and then becomes FALSE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE ◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.

■ Function description

● Basic function description

This instruction controls multiple axes to start and stop at the same time so that they can perform linear interpolation movements. During the process, up to 8 axes can move at the same time. The distance moved by each axis is set individually by the input variable Distance, the input variable Velocity is the synthesized velocity of all axes, and the input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. This instruction moves according to the values set by the input variables when Execute changes from FALSE to TRUE.

- **Absolute position**
  - The value of the input variable Position represents the absolute position using the 0 point position as a reference point. The value can be positive, 0, or negative. Values for Velocity, Acceleration, Deceleration, and Jerk can only be positive.
  - This instruction is executed when the axis is stationary, and the position curve and speed curve when the value of Position is greater than the current position and less than the current position are shown in the figure below. When the value of Position is greater than the current position, the axis will run in the forward direction; when the value of Position is less than the current position, the axis will run in the reverse direction.
  - When the value of Position equals the current position, this instruction does not control axis operation, and the Done bit of this instruction becomes TRUE after one scan period of executing this instruction.



- **Instruction completion timing**

  When the command position of each axis in the axis group reaches the set position, the instruction is completed and Done changes from FALSE to TRUE.

- **Re-execute the instruction**

  When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

- **Execute this instruction while other instructions are in processing**

  This instruction is executing when other axis group motion instructions are executed, and only mcBuffered can be selected for this instruction BufferMode, and only mcTMNone (velocity decreases to 0 when position is reached) can be selected for TransitionMode.

- **Execute other instructions while this instruction is in processing**

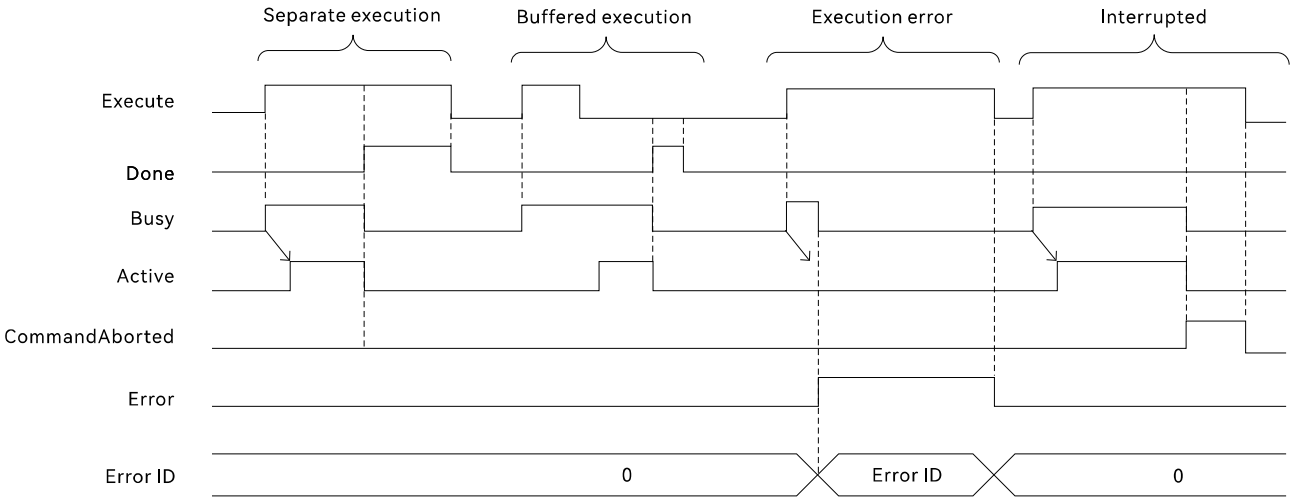
  When this instruction is executed, other axis group motion instructions are started, and how other axis group motion instructions and this instruction are buffered is jointly determined by the BufferMode and TransitionMode values of other motion instructions. When other motion instructions are buffered with this instruction, the other instructions are executed when the Done instruction becomes TRUE. If there is no BufferMode pin for other motion instructions, the instruction is normally interrupted. When this instruction is executed, the execution of other non-axis group-related motion instructions will interrupt this instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input

variable of the executed instruction; the other axes in the axis group will enter into the state of standstill, and the axes speed will be reduced to 0 for one cycle.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. If an abnormality (such as an axis alarm, etc.) is encountered when this instruction is executed, the instruction will also report an error and the axis will stop immediately.

■ Output variable timing diagram



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next period. When Distance is reached and positioning is completed, Done becomes TRUE, and Busy and Active become FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Buffered execution**

  The instruction is executed when the other instruction controls the axis (BufferMode is not in mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction is completed.

- **Execution error**

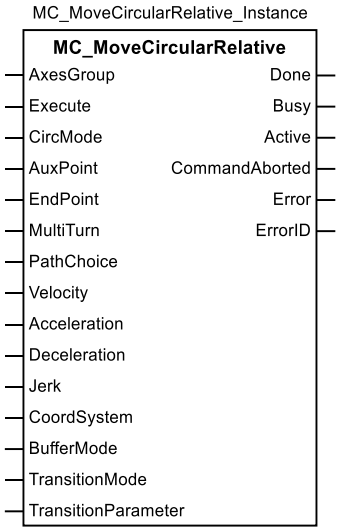
  When the value of the input variable of this instruction is not within the valid range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Interrupted**

  When the execution of this instruction is interrupted by other instructions, the instruction CommandAborted becomes TRUE and Busy and Active become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

7.13 MC_MoveCircularRelative (relative circular interpolation)

Control multiple axes to start and stop at the same time. Two axes constitutes a circular interpolation with other axes follow the circular motion and start or stop at the same time. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveCircularRelative	Relative circular interpolation	FB		MC_MoveCircularRelative _Instance (AxesGroup:=parameter , Execute :=parameter, CircMode :=parameter, AuxPoint :=parameter, EndPoint :=parameter, MultiTurn:=parameter , PathChoice:=parameter , Velocity :=parameter, Acceleration :=parameter , Deceleration:=parameter , Jerk:=parameter , CoordSysytem:=parameter , BufferMode:=parameter , TransitionMode:=parameter , TransitionParameter:=parameter , Done=> parameter , Busy => parameter, Active=> parameter , CommandAborted=> parameter , Error => parameter, ErrorID=> parameter);

■ **Input variable**

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge
CircMode	circular interpolation mode	INT	0~5	0	Setting the mode of circular interpolation 0: Specify the center of circle coordinate in XY plane. 1: Specify the center of circle in XZ plane. 2：YZ plane specified center coordinates 3：XY plane specified radius 4：XZ plane specified radius 5：YZ plane specified radius
AuxPoint	Coordinates or radius of the center of the circle	ARRAY [1..2] OF LREAL	Positive number, Negative value, 0	0	When specifying the center coordinate to draw an arc, this parameter indicates the center coordinate; When specifying the radius to draw a circle, Auxpoint[1] indicates the radius and AuxPoint[2] has no effect.
EndPoint	End coordinate	ARRAY [1..8] OF LREAL	Positive number, Negative value, 0	0	The relative position of each axis is the current position as the reference point

MultiTurn	Turns	UINT	Positive number, 0	0	Set the number of turns to draw a circle
PathChoice	Path selection	INT	0、1	0	The direction of the arc path 0 : Clockwise direction 1 : Counterclockwise direction
Velocity	Target velocity	LREAL	Positive number	Required field	Target velocity*1 (Unit: Travel unit/s) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Target acceleration*1 (Unit: Travel unit/s^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Target deceleration*1 (Unit: Travel unit/s^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Target Jerk*1 (Unit: Travel unit/s^3) *2
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved
BufferMode	Buffer mode	MC_Buffer_Mode	1: mcBuffered 3: mcBlendingPrevious	Required field	Setting the buffer mode between two instructions *3 1: Buffered 3: Buffered at current instruction target velocity.
TransitionMode	TransitionMode	MC_Transition_Mode	0: mcTMNone 2: TMConstantVelocity 3: mcTMCornerDistance	Required field	Setting the connection between the interpolation instruction track of the current control axis and the buffered interpolation instruction track 0: Velocity decreases to 0 when the position is reached. 2: Transit at the specified speed 3: Transit at the set additional angle
TransitionParameter	Additional angle	LREAL	Positive number, 0	0	Transition parameters are set when TransitionMode is selected to transit at an additional angle.

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to *"Parameter description of motion control instructions"*.

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

*3: For details of BufferMode, please refer to "*Buffer mode during multi-starting of the same axis*".

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Active	Under control	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
CommandAborted	Abort	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE. ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE for one period and then changes to FALSE
Busy	When Execute changes to TRUE	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
Active	When the instruction is started	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
CommandAborted	When the instruction is aborted by other instruction	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE ◆ Instruction has been executed and Execute becomes FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period and then becomes FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE ◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.

■ Function description

● Basic function description

The instruction is used to control multiple axes to start and stop at the same time. Two axes are circularly interpolated, and the other axes start and stop at the same time following the circular motion. When the Execute changes from FALSE to TRUE, the instruction moves according to the value set by the input variable.

● Target velocity

The input variable Velocity is the synthesized velocity of all axes. The relationship between the synthesized velocity and the velocity of each axis is: the sum of the squares of the synthesized velocity equals the sum of the squares of the velocity of each axis. The input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. The relationship between the synthesized acceleration and deceleration and the acceleration and deceleration of each axis is: the sum of the squares of the synthesized acceleration and deceleration equals the sum of the squares of the acceleration and deceleration of each axis.

● Instruction completion timing

When the commanded position of each axis in the axis group reaches the set distance, the instruction is completed and Done changes from FALSE to TRUE.

● Re-execute the instruction

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

● Execute this instruction while other instructions are in processing

This instruction is activated when other axis group motion instructions are executed. How this instruction and other motion instructions are buffered is jointly determined by the values of BufferMode and TransitionMode. The detailed description is shown in the following table.

BufferMode	TransitionMode	Description
mcBuffered (Await)	mcTMNone (Velocity becomes 0 when position is reached)	Wait for the execution of the current axis group instruction to be completed (when the target position is reached, the velocity becomes 0) and switch to the buffered axis group instruction control axis.
mcBlendingPrevious (Buffered at the target speed of the current instruction)	mcTMCornerDistance (Transit at an additional angle.)	The moment of completion of the target position of the current axis group instruction is the target position of the current axis group instruction minus the attached additive angle position set by mcTMCornerDistance. After the execution of the current axis group instruction is completed, the buffered axis group instruction is executed immediately. The buffered axis group instruction first moves with the additional angle set by mcTMCornerDistance, and then moves with the parameter set by the buffered axis group instruction

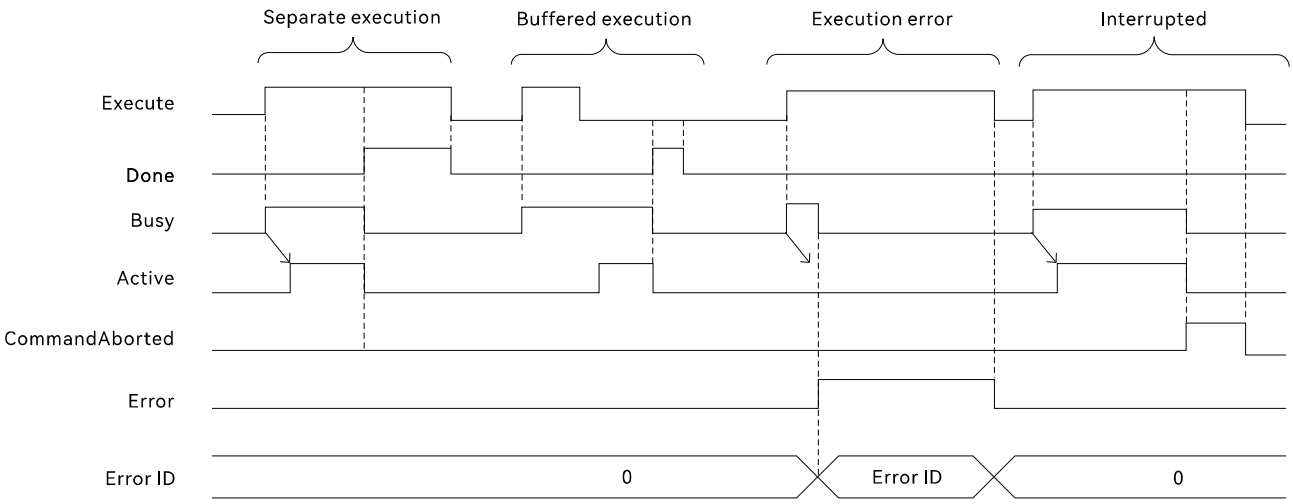
- **Execute other instructions while this instruction is in processing**

  When this instruction is executed, other axis group motion instructions are started, and how other axis group motion instructions and this instruction are buffered is jointly determined by the BufferMode and TransitionMode values of other motion instructions. When other motion instructions are buffered with this instruction, the other instructions are executed when the Done instruction becomes TRUE. If there is no BufferMode pin for other motion instructions, the instruction is normally interrupted. When this instruction is executed, the execution of other non-axis group-related motion instructions will interrupt this instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input variable of the executed instruction; the other axes in the axis group will enter into the state of standstill, and the axes speed will be reduced to 0 for one period.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. If an abnormality (such as an axis alarm, etc.) is encountered when this instruction is executed, the instruction will also report an error and the axis will stop immediately.

■ Output variable timing diagram



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next cycle; when Distance is reached and positioning is completed, Done becomes TRUE, and Busy and Active become FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Buffered execution**

  The instruction is executed when the other instruction controls the axis (BufferMode is not in mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction is completed.

- **Execution error**

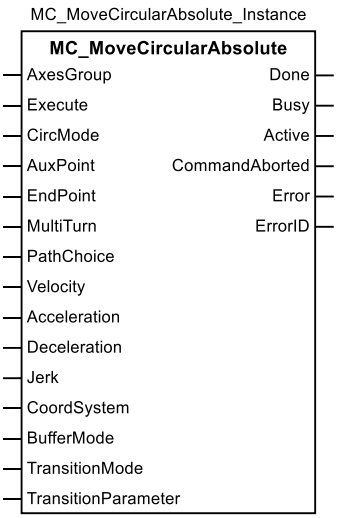
  When the value of the input variable of this instruction is not within the valid range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Interrupted**

  When the execution of this instruction is interrupted by other instructions, the instruction CommandAborted becomes TRUE and Busy and Active become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

7.14 MC_MoveCircularAbsolute (absolute circular interpolation)

Control multiple axes to start and stop at the same time Two axes constitutes a circular interpolation with other axes follow the circular motion and start or stop at the same time. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_MoveCircularAbsolute	Absolute circular interpolation	FB		MC_MoveCircularAbsolute_Instance (AxesGroup:=parameter , Execute:=parameter, CircMode:=parameter , AuxPoint :=parameter, EndPoint :=parameter, MultiTurn:=parameter , PathChoice :=parameter, Velocity:=parameter , Acceleration :=parameter, Deceleration:=parameter, Jerk :=parameter, CoordSysytem :=parameter, BufferMode:=parameter , TransitionMode , TransitionParameter , Done=> parameter , Busy=> parameter , Active => parameter, CommandAborted=> parameter , Error => parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Execute	Execute	BOOL	TRUE or FALSE	FALSE	Executing when detecting the rising edge
CircMode	circular interpolation mode	INT	0~5	0	Setting the mode of circular interpolation 0: Specify the center of circle coordinate in XY plane. 1: Specify the center of circle in XZ plane. 2：YZ plane specified center coordinates 3：XY plane specified radius 4：XZ plane specified radius 5：YZ plane specified radius
AuxPoint	Coordinates or radius of the center of the circle	ARRAY [1..2] OF LREAL	Positive number, Negative value, 0	0	When specifying the center coordinate to draw an arc, this parameter indicates the center coordinate; When specifying the radius to draw a circle, Auxpoint[1] indicates the radius and AuxPoint[2] has no effect.
EndPoint	End coordinate	ARRAY [1..8] OF LREAL	Positive number, Negative value, 0	0	The relative position of each axis is the current position as the reference point.

MultiTurn	Turns	UINT	Positive number, 0	0	Set the number of circles to draw a circle
PathChoice	Path selection	INT	0、1	0	The direction of the arc path 0 : Clockwise direction 1 : Counterclockwise direction
Velocity	Target velocity	LREAL	Positive number	Required field	Target velocity*1 (Unit: Travel unit/s) *2
Acceleration	Acceleration	LREAL	Positive number	Required field	Target acceleration*1 (Unit: Travel unit/s^2) *2
Deceleration	Deceleration	LREAL	Positive number	Required field	Target deceleration*1 (Unit: Travel unit/s^2) *2
Jerk	Jerk	LREAL	Positive number	Required field	Target Jerk*1 (Unit: Travel unit/s^3) *2
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved
BufferMode	Buffer mode	MC_Buffer_Mode	1: mcBuffered 3: mcBlendingPrevious	Required field	Setting the buffer mode between two instructions *3 1: Buffered 3: Buffered at current instruction target velocity.
TransitionMode	Transition mode	MC_Transition_Mode	0: mcTMNone 2: TMConstantVelocity 3: mcTMCornerDistance	Required field	Setting the connection between the interpolation instruction track of the current control axis and the buffered interpolation instruction track 0: Velocity decreases to 0 when the position is reached. 2: Transit at the specified speed 3: Transit at the set additional angle
TransitionParameter	Additional angle	LREAL	Positive number, 0	0	Transition parameters are set when TransitionMode is selected to transit at an additional angle.

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "*Parameter description of motion control instructions*".

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

*3: For details of BufferMode, please refer to "*Buffer mode during multi-starting of the same axis*".

■ Output variable

Name	Meaning	Data type	Valid range	Description
Done	Completed	BOOL	TRUE or FALSE	TRUE when the instruction execution is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when instruction is executed.
Active	Under control	BOOL	TRUE or FALSE	TRUE when the axis is under control the instruction is aborted
CommandAborted	Abort	BOOL	TRUE or FALSE	TRUE when the instruction is aborted
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	When positioning is completed	◆ When Done is TRUE and Execute changes from TRUE to FALSE. ◆ When the instruction is executed and Execute is FALSE, Done changes to TRUE for one period and then changes to FALSE. ◆
Busy	When Execute changes to TRUE	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
Active	When the instruction is started	◆ When Done becomes TRUE from FLASE ◆ When Error becomes TRUE from FLASE ◆ When CommandAborted becomes TRUE from FLASE
CommandAborted	When the instruction is aborted by other instruction	◆ When CommandAborted is TRUE, Enable will become TRUE from FALSE ◆ Instruction has been executed and Execute becomes FALSE, when the instruction is aborted by another instruction, CommandAborted becomes TRUE for one period and then becomes FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	◆ When Execute changes from FALSE to TRUE ◆ Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE for one period and then becomes FALSE.

■ Function description

● Basic function description

The instruction is used to control multiple axes to start and stop at the same time. The two axes perform circular interpolation, and the other axes follow the circular motion to start and stop at the same time. The end point position is the absolute target position with the 0 point position as the reference point. When the Execute changes from FALSE to TRUE, the instruction moves according to the value set by the input variable.

● Target velocity

The input variable Velocity is the synthesized velocity of all axes. The relationship between the synthesized velocity and the velocity of each axis is: the sum of the squares of the synthesized velocity = the sum of the squares of the velocity of each axis. The input variables Acceleration and Deceleration are the synthesized acceleration and deceleration of all axes. The relationship between the synthesized acceleration and deceleration and the acceleration and deceleration of each axis is: the sum of the squares of the synthesized acceleration and deceleration = the sum of the squares of the acceleration and deceleration of each axis.

● Instruction completion timing

When the commanded position of each axis in the axis group reaches the set distance, the instruction is completed and Done changes from FALSE to TRUE.

● Re-execute the instruction

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

● Execute this instruction while other instructions are in processing

This instruction is activated when other axis group motion instructions are executed. How this instruction and other motion instructions are buffered is jointly determined by the values of BufferMode and TransitionMode. The detailed description is shown in the following table.

BufferMode	TransitionMode	Description
mcBuffered (Await)	mcTMNone (Velocity becomes 0 when position is reached)	Wait for the execution of the current axis group instruction to be completed (when the target position is reached, the velocity becomes 0) and switch to the buffered axis group instruction control axis.
mcBlendingPrevious (Buffered at the target speed of the current instruction)	mcTMCornerDistance (Transit at an additional angle.)	The moment of completion of the target position of the current axis group instruction is the target position of the current axis group instruction minus the attached additive angle position set by mcTMCornerDistance. After the execution of the current axis group instruction is completed, the buffered axis group instruction is executed immediately. The buffered axis group instruction first moves with the additional angle set by mcTMCornerDistance, and then moves with the parameter set by the buffered axis group instruction.

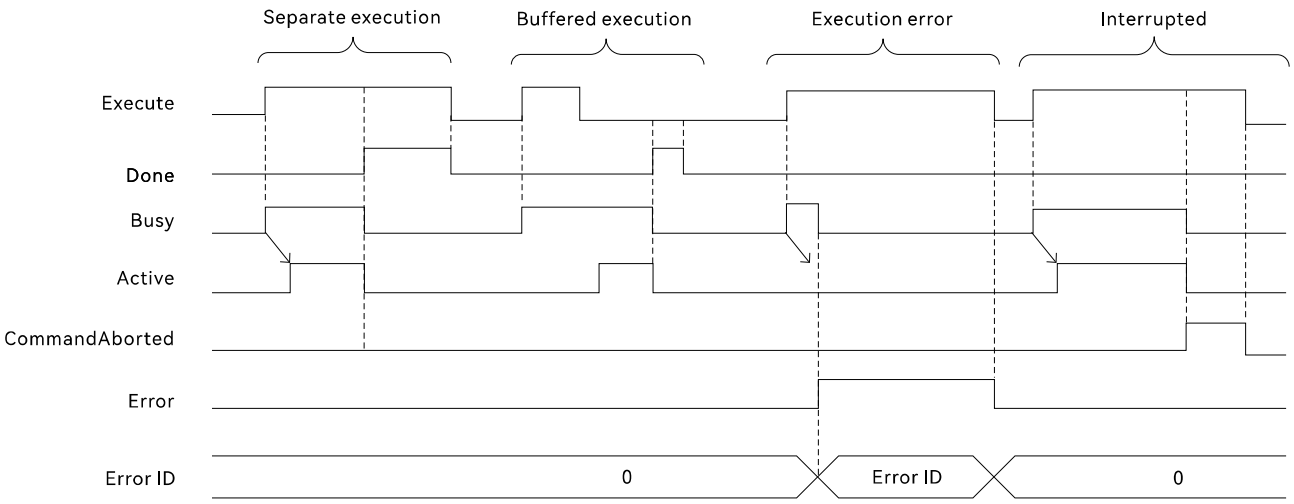
- **Execute other instructions while this instruction is in processing**

When this instruction is executed, other axis group motion instructions are started, and how other axis group motion instructions and this instruction are buffered is jointly determined by the BufferMode and TransitionMode values of other motion instructions. When other motion instructions are buffered with this instruction, the other instructions are executed when the Done instruction becomes TRUE. If there is no BufferMode pin for other motion instructions, the instruction is normally interrupted. When this instruction is executed, the execution of other non-axis group-related motion instructions will interrupt this instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input variable of the executed instruction; the other axes in the axis group will enter into the state of standstill, and the axes speed will be reduced to 0 for one cycle.

- **Abnormality elimination**

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become TRUE, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error. If an abnormality (such as an axis alarm, etc.) is encountered when this instruction is executed, the instruction will also report an error and the axis will stop immediately.

- **Output variable timing diagram**



- **Separate execution**

  When Execute changes from FALSE to TRUE, Busy becomes TRUE at the same time, and Active becomes TRUE in the next cycle; when Distance is reached and positioning is completed, Done becomes TRUE, and Busy and Active become FALSE at the same time. When Execute becomes FALSE, Done becomes FALSE at the same time.

- **Buffered execution**

  The instruction is executed when the other instruction controls the axis (the value of BufferMode is not mcAborting), and when Execute changes from FALSE to TRUE, Busy changes to TRUE at the same time, and the timing for Active to change to TRUE is when the previous instruction is completed.

- **Execution error**

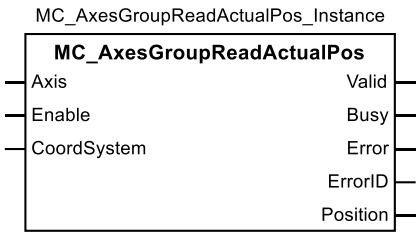
  When the value of the input variable of this instruction is not within the valid range, the instruction Execute changes from FALSE to TRUE while Busy changes to TRUE, and Error changes to TRUE in the next cycle while Busy changes to FALSE, and ErrorID outputs the corresponding error code, so that users can find out the cause of the problem by using the value of ErrorID. ErrorID outputs the corresponding error code, which can be used to find the cause of the problem by the value of ErrorID. When the instruction Execute changes from TRUE to FALSE, Error changes to FALSE, and the value of ErrorID changes to 0.

- **Interrupted**

  When the execution of this instruction is interrupted by other instructions, the instruction CommandAborted becomes TRUE and Busy and Active become FALSE at the same time; when Execute becomes FALSE, CommandAborted becomes FALSE at the same time.

7.15 MC_AxesGroupReadActualPosition（read the feedback position of each axis in the axis group）

The instruction is used to periodically read the feedback position of each axis in the axis group. Library: MotionControl_Part2

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_AxesGroupReadActualPos	Read the feedback position of each axis in the axis group	FB		MC_AxesGroupReadActualPos_Instance (Axis:=parameter, Enable:=parameter, CoordSystem:=parameter, Valid => parameter, Busy=> parameter , Error=> parameter , ErrorID => parameter, Position=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	Depend on model	Required field	Axis group number
Enable	Enable	BOOL	TRUE or FALSE	FALSE	Set to TRUE, read the specified axis feedback position Set to FALSE, stop reading the specified axis feedback position
CoordSystem	Coordinate System	INT	Reserved	Reserved	Reserved

■ Output variable

Name	Name	Data type	Valid range	Function
Valid	Valid	BOOL	TRUE or FALSE	TRUE when the instruction is completed
Busy	Executing	BOOL	TRUE or FALSE	TRUE when the instruction is executed
Error	Error	BOOL	TRUE or FALSE	TRUE when there is an error
ErrorID	Error Code	WORD	0~65535	Refer to "instruction error code description" for the meaning of the output error code value when an instruction execution error occurs.
Position	Actual position	ARRAY [1..8] OF LREAL	Positive number, Negative value, 0	Position unit converted from the axis "setup parameter" according to the feedback position of the specified axis: Travel unit

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Valid	When the feedback position could be read	◆ Enable becomes TRUE ◆ Error becomes TRUE
Busy	When Enable set to TRUE	◆ Enable becomes TRUE ◆ Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range,	◆ When Error is TRUE and Enable changes from TRUE to FALSE.

■ Function description

- **Basic function description**

  This instruction is used to read the feedback position of each axis in the specified axis group periodically, and is updated once per synchronized clock cycle; the output variable Position of this instruction is a one-dimensional array, and each member of the array corresponds to the position of an axis in the axis group, and the position of at most 8 axes in the axis group can be read. When the input variable of this instruction Enable is TRUE, the value of Position is updated once per synchronized clock cycle; when Enable is FALSE, the value of the axis actual position stops updating and keeps the value unchanged when the last Enable is TRUE.

- **Actual position calculation**

  The value of the actual position is the position converted to the mechanism through the axis position, and the unit is the travel unit. When the specified axis is a servo axis, the instruction reads the position from the number of pulses fed back from the motor to the driver; when the instruction reads the position from the virtual servo axis, the instruction reads the position from the virtual servo axis, and does not need to be converted. When the specified axis is servo axis or encoder axis, the value of the feedback position is obtained through the above sources after the conversion of the mechanism parameters, which can be viewed in the software [Motion Control] → [Axis Settings] → [Basic Settings], and the conversion relationship is calculated as shown in the following figure, which is calculated by the controller without the need for user operations.

$$\textbf{Actual position} = \frac{\textbf{The number of feedback pulses from motor}}{\textbf{The number of pulses received by the encoder}} \times \textbf{Working stroke per resolution} \times \frac{\textbf{Output speed of the reducer}}{\textbf{Number of pulses per revolution of the motor} \times \textbf{Input speed of the reducer}}$$

- **Actual position update cycle**

  This instruction reads the feedback position of the axis, which is updated once in one synchronization cycle, and the instruction reads the same feedback position at different moments in the same synchronization cycle.

- **Effect of MC_Home, MC_SetPosition and MC_HomeWithParm on actual position**

  After the MC_Home, MC_SetPosition and MC_HomeWithParm instructions are executed, the axis instruction position and feedback position will follow the set position of these instructions, and the position read by this instruction after these instructions are executed is the same as the feedback position in the axis structure body.

- **Abnormality elimination**

  When this instruction is executed, if the input variable is not legal, the instruction Error becomes TRUE, and there is a corresponding error code in ErrorID, which can be used to determine the cause of the error.

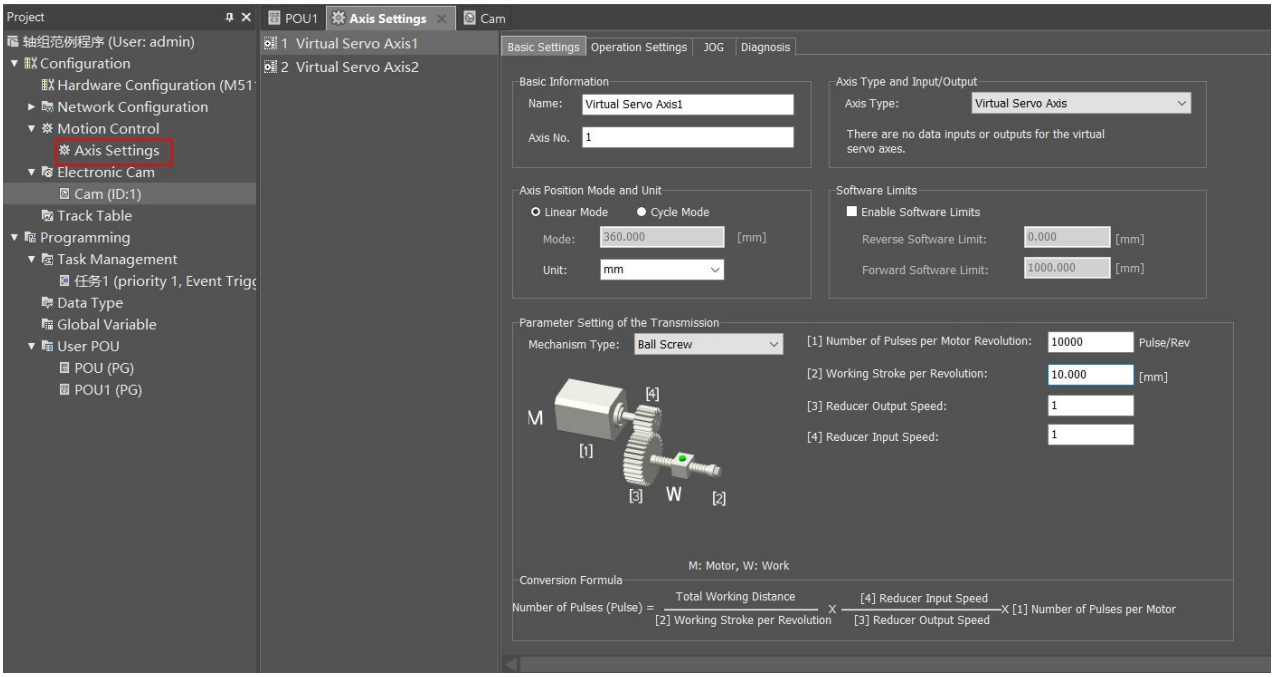
7.16 Example program for Axis Group Motion Instructions

■ Steps required for the execution of axis group motion commands

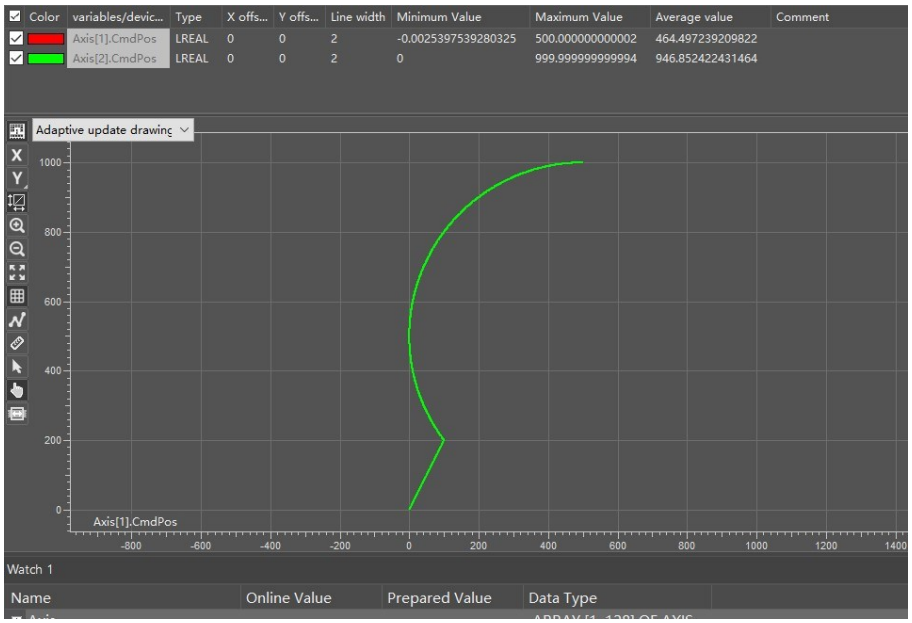
Step	Description	Note
Step1	Setting the axis parameters for each axis	Set in the software
Step 2	Enable the axis	The servo axis executes the MC_Power command, the virtual servo axis does not need to do it.
Step 3	Add axis to the axis group	Correspondence between the logical axes in the axis group and the configured axes in the software.
Step 4	Enable the axis group	Enable the axis group
Step 5	Execute the axis group motion instruction	Execute the axis group motion instruction

■ Axis parameter setting

Axis 1 and Axis 2 axis parameter settings are the same as shown in the following figure.



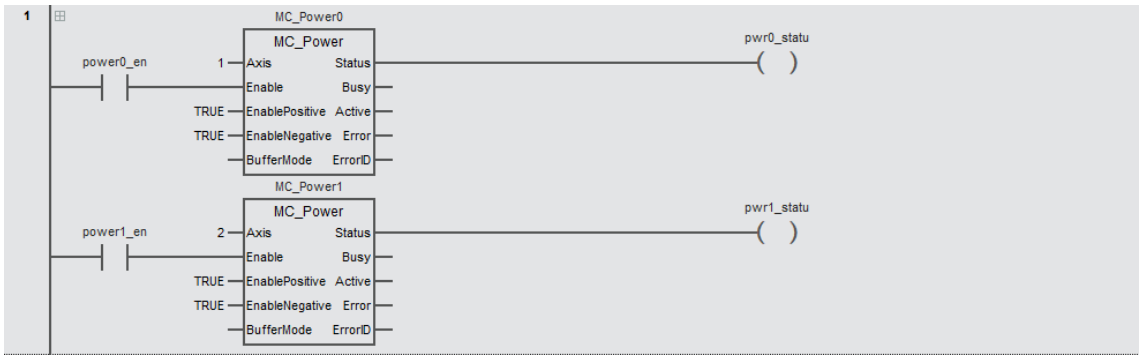
- Axis group motion trajectory figure

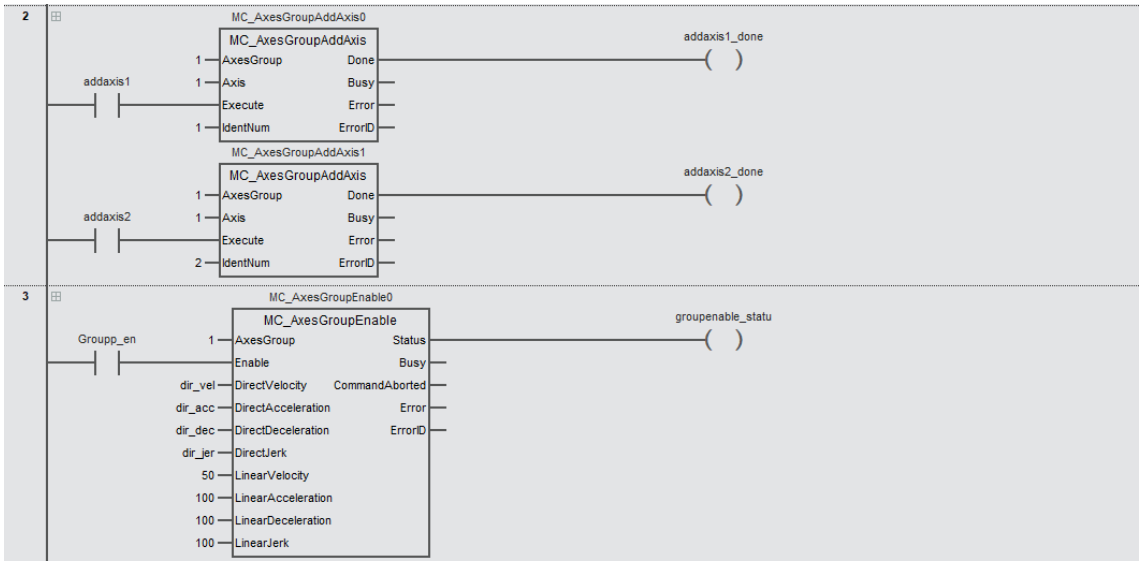


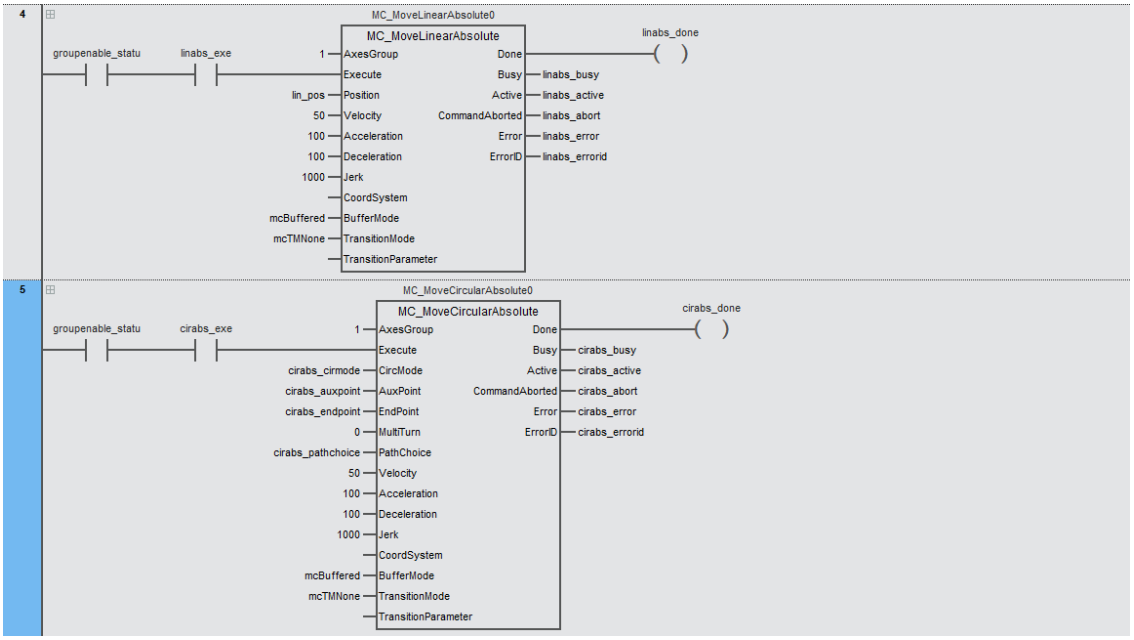
- **Variable table**

Type	Name	Allocated to	Data type	Default	Note
VAR	MC_Power0		MC_Power		
VAR	MC_Power1		MC_Power		
VAR	power0_en		BOOL		
VAR	power1_en		BOOL		
VAR	pwr0_statu		BOOL		
VAR	pwr1_statu		BOOL		
VAR	MC_AxesGroupAddAxis0		MC_AxesGroupAddAxis		
VAR	MC_AxesGroupAddAxis1		MC_AxesGroupAddAxis		
VAR	addaxis1		BOOL		
VAR	addaxis2		BOOL		
VAR	addaxis1_done		BOOL		
VAR	addaxis2_done		BOOL		
VAR	MC_AxesGroupEnable0		MC_AxesGroupEnable		
VAR	Groupp_en		BOOL		
VAR	dir_vel		ARRAY [1..8] OF LREAL	[8(50)]	
VAR	dir_acc		ARRAY [1..8] OF LREAL	[8(100)]	
VAR	dir_dec		ARRAY [1..8] OF LREAL	[8(100)]	
VAR	dir_jer		ARRAY [1..8] OF LREAL	[8(1000)]	
VAR	lin_vel		LREAL		
VAR	lin_acc		LREAL		
VAR	groupenable_statu		BOOL		
VAR	linabs_exe		BOOL		
VAR	MC_MoveLinearAbsolute0		MC_MoveLinearAbsolute		
VAR	lin_pos		ARRAY [1..8] OF LREAL	[100,200]	
VAR	linabs_done		BOOL		
VAR	linabs_busy		BOOL		
VAR	linabs_active		BOOL		
VAR	linabs_abort		BOOL		
VAR	linabs_error		BOOL		
VAR	linabs_errorid		WORD		
VAR	MC_MoveCircularAbsolute0		MC_MoveCircularAbsolute		
VAR	cirabs_exe		BOOL		
VAR	cirabs_auxpoint		ARRAY [1..2] OF LREAL	[400,300]	
VAR	cirabs_endpoint		ARRAY [1..8] OF LREAL	[500,1000]	
VAR	cirabs_cirmode		INT		
VAR	cirabs_pathchoice		INT		
VAR	cirabs_done		BOOL		
VAR	cirabs_busy		BOOL		
VAR	cirabs_active		BOOL		
VAR	cirabs_abort		BOOL		
VAR	cirabs_error		BOOL		
VAR	cirabs_errorid		WORD		

- **LD**


1
MC_Power0
MC_Power
power0_en
1
Axis
Status
pwr0_statu
Enable
Busy
TRUE
EnablePositive
Active
TRUE
EnableNegative
Error
BufferMode
ErrorID
MC_Power1
MC_Power
power1_en
2
Axis
Status
pwr1_statu
Enable
Busy
TRUE
EnablePositive
Active
TRUE
EnableNegative
Error
BufferMode
ErrorID


2
MC_AxesGroupAddAxis0
MC_AxesGroupAddAxis
1
AxesGroup
Done
addaxis1_done
addaxis1
1
Axis
Busy
Execute
Error
1
IdentNum
ErrorID
MC_AxesGroupAddAxis1
MC_AxesGroupAddAxis
1
AxesGroup
Done
addaxis2_done
addaxis2
1
Axis
Busy
Execute
Error
2
IdentNum
ErrorID
3
MC_AxesGroupEnable0
MC_AxesGroupEnable
Groupp_en
1
AxesGroup
Status
groupenable_statu
Enable
Busy
dir_vel
DirectVelocity
CommandAborted
dir_acc
DirectAcceleration
Error
dir_dec
DirectDeceleration
ErrorID
dir_jer
DirectJerk
50
LinearVelocity
100
LinearAcceleration
100
LinearDeceleration
100
LinearJerk


4
MC_MoveLinearAbsolute0
MC_MoveLinearAbsolute
groupenable_statu
linabs_exe
1
AxesGroup
Done
linabs_done
Execute
Busy
linabs_busy
lin_pos
Position
Active
linabs_active
50
Velocity
CommandAborted
linabs_abort
100
Acceleration
Error
linabs_error
100
Deceleration
ErrorID
linabs_errorid
1000
Jerk
CoordSystem
mcBuffered
BufferMode
mcTMNone
TransitionMode
TransitionParameter
5
MC_MoveCircularAbsolute0
MC_MoveCircularAbsolute
groupenable_statu
cirabs_exe
1
AxesGroup
Done
cirabs_done
Execute
Busy
cirabs_busy
cirabs_cirmode
CircMode
Active
cirabs_active
cirabs_auxpoint
AuxPoint
CommandAborted
cirabs_abort
cirabs_endpoint
EndPoint
Error
cirabs_error
0
MultiTurn
ErrorID
cirabs_errorid
cirabs_pathchoice
PathChoice
50
Velocity
100
Acceleration
100
Deceleration
1000
Jerk
CoordSystem
mcBuffered
BufferMode
mcTMNone
TransitionMode
TransitionParameter

- **Structured text (ST)**

```
MC_Power0(
        Axis:=1 ,
        Enable:= power0_en,
        EnablePositive:= TRUE,
        EnableNegative:= TRUE,
        BufferMode:=0 ,
        Status=> pwr0_statu
        );

MC_Power1(
        Axis:=2 ,
        Enable:= power1_en,
        EnablePositive:= TRUE,
        EnableNegative:= TRUE,
        BufferMode:=0 ,
        Status=>pwr0_statu
        );

 MC_AxesGroupAddAxis0(
        AxesGroup:=1 ,
        Axis:=1 ,
        Execute:= addaxis1 ,
        IdentNum:=1 ,
        Done=> addaxis1_done
        );

 MC_AxesGroupAddAxis1(
        AxesGroup:=1 ,
        Axis:=2 ,
        Execute:= addaxis2,
        IdentNum:=2 ,
        Done=> addaxis2_done
        );

 MC_AxesGroupEnable0(
        AxesGroup:=1 ,
        Enable:= Groupp_en ,
        DirectVelocity:= dir_vel,
        DirectAcceleration:= dir_acc ,
        DirectDeceleration:= dir_dec ,
        DirectJerk:= dir_jer ,
        LinearVelocity:= 50,
        LinearAcceleration:=100 ,
        LinearDeceleration:= 100,
        LinearJerk:= 1000,
        Status=> groupenable_statu
        );

MC_MoveLinearAbsolute0(
        AxesGroup:= 1,
        Execute:= groupenable_statu AND linabs_exe,
        Position:= lin_pos ,
```

```
        Velocity:= 50,
        Acceleration:=100 ,
        Deceleration:= 100,
        Jerk:= 1000,
        BufferMode:=mcBuffered ,
        TransitionMode:=mcTMNone ,
        Done=> linabs_done,
        Busy=> linabs_busy,
        Active=>linabs_active,
        CommandAborted=>linabs_abort,
        Error=>linabs_error,
        ErrorID=>linabs_errorid
        );

MC_MoveCircularAbsolute0(
        AxesGroup:=1 ,
        Execute:= groupenable_statu AND cirabs_exe ,
        CircMode:= cirabs_cirmode,
        AuxPoint:= cirabs_auxpoint ,
        EndPoint:= cirabs_endpoint,
        MultiTurn:= 0,
        PathChoice:= cirabs_pathchoice ,
        Velocity:= 50,
        Acceleration:=100 ,
        Deceleration:= 100,
        Jerk:= 1000,
        BufferMode:=mcBuffered ,
        TransitionMode:=mcTMNone ,
        Done=>cirabs_done,
        Busy=>cirabs_busy,
        Active=>cirabs_active,
        CommandAborted=>cirabs_abort,
        Error=>cirabs_error,
        ErrorID=>cirabs_errorid
        );
```

Chapter8 CNC Introduction

8.1 CNC Introduction

M500 series controllers (excluding M500S series), as high-level motion controllers, support basic CNC function, such as support for G0, G1, G2/G3, G4, G90/G91, G50/G51/G52, etc.; in addition, they support logical processing of M code and other program variables in the task, such as through the CNC code can be controlled by the cylinder action. When multiple CNC codes are executed, it is possible to control the cylinder action by CNC code.

When multiple CNC codes are executed, the controller will pre-read multiple CNC codes, and after one line of CNC code is executed, the speed can continue to execute the next line of CNC code without decreasing to 0, so as to improve the machining efficiency. This Function can be realized by G51 or G52 code.

The CNC code can be edited by the user in the software, or it can be converted to DXF by CAD related software and imported into the software. after editing the CNC code, the motion trajectory corresponding to the CNC code will be displayed in the software. The CNC code in the software must be downloaded to the control before it can be executed.

■ Controller that support CNC Function

Only M500 series controllers (excluding M500S series) support the CNC function.

■ CNC Code format

The CNC program includes G codes and M codes, and the meanings and Input formats of the corresponding codes when written separately are shown in the table below:

Code	Code name	Input format*	Max. number of axes
G0	Fast positioning	Format: N& G0 X& Y& Z& A& B& C& P& Q& Example:N0 G0 X10.1 Y10 Z10 A10 B10 C10 P10 Q10	8 axes
G1	Linear interpolation	Format: N& G1 X& Y& Z& A& B& C& P& Q& E& E& F& Example:N0 G1 X10.1 Y10 Z10 A10 B10 C10 P10 Q10 E100 E-100 F10	8 axes
G2	Clockwise circular interpolation, helical interpolation	Format1：N& G2 X& Y& Z& A& B& C& P& Q& I& J& T& E& E& F& Example:N0 G2 X500 Y1000 I400 J300 E100 E-100 F10 Format2：N& G2 X& Y& Z& A& B& C& P& Q& R& T& E& E& F& Example:N0 G2 X500 Y1000 R500 E100 E-100 F10	8 axes
G3	Counterclockwise circular interpolation, helical interpolation	Format1：N& G3 X& Y& Z& A& B& C& P& Q& I& J& T& E& E& F& Example:N0 G3 X100 Y200 I0 J-500 E100 E-100 F10 Format2：N& G3 X& Y& Z& A& B& C& P& Q& R& T& E& E& F& Example:N0 G3 X100 Y200 R500 E100 E-100 F10	8 axes
G4	Time delay	Format: N& G4 K& Example: N0 G4 K2	Not directly related to the number of axes
G17	Set to XY plane	Format: N& G17 Example: N0 G17	
G18	Set to ZX plane	Format: N& G18 Example: N0 G18	
G19	Set to YZ plane	Format: N& G19 Example: N0 G19	
G90	Absolute mode	Format: N& G90 Example: N0 G90	
G91	Relative mode	Format: N& G91 Example: N0 G91	
G50	Precision stop	Format: N& G50 Example: N0 G50	
G51	Arc transition	Format: N& G51 Example: N0 G51	

G52	Smooth transition	Format: N& G52 Example: N0 G52	
M0~M99	M code	Format1： N& M& Example: N0 M1 Format2： N& M& D& Example:N0 M1 D6.2	

*& denotes the value of the corresponding parameter in the G code, please refer to the description of the specific G code for the unit of the parameter value.

CNC Code combination format

G code	G code/M code	Input format*
G17/G18/G19	G2/G3	Example:N0 G17 G2
G0/G1/G2/G3/G4	M0~M99	Example:N0 G1 X10.1 Y10 M10 D12.5

*1: When G17/G18/G19 and G2/G3 codes are used in combination, G17/G18/G19 should be placed at the beginning of the same line.

*2: When G code and M code are used in combination, M code should be placed at the end of the same line.

Parameters that can be left out of the CNC code

- One or more of X& Y& Z& A& B& C& P& Q& in G0 code can be left out, and parameters not input are not executed.
- One or more of G1 X& Y& Z& A& B& C& P& Q& E& E& F& in G1 code can be left out and parameters not input are not executed (E, F as Default).
- One or more of X& Y& Z& A& B& C& P& Q& E& E& F& can be left out after G2 and G3 codes, and I&, J&, or R& must be entered. If XY plane is specified, the parameters corresponding to X and Y must be input; if XZ plane is specified, the parameters corresponding to X and Z must be input; if YZ plane is specified, the parameters corresponding to Y and Z must be input.
- The D parameter in the M code can be left out.

CNC Code Parameters' device representation methods

- If the relevant parameter in the G code is not a fixed value and needs to be specified externally, the corresponding parameter can be expressed by using the device, which is expressed by the $ symbol on both sides of the ML device number, and the % corresponding to the ML device number does not need to be inputted. example:N00 G1 X$ML10$ Y$ML11$ Z$ML12$, such as %ML10, %ML11, %ML12 are 100, 200, 300 respectively, the execution effect of the above G code using the device and N00 G1 X100 Y200 Z300 is the same. ML11$ and %ML12$ are 100, 200 and 300 respectively, the above G-code of the used device has the same effect as that of N00 G1 X100 Y200 Z300.
- X, Y, Z, A, B, C, P, Q, E, F, I, J, R, T, K after the parameter can be used %ML device, the use of the device, pay attention to the corresponding parameters of the Data type, each parameter corresponds to the Data type can be viewed in the [G code Function details] → [parameter description], you can define the corresponding Data type variables in the software. You can define the corresponding Data type variable in the software and assign it to the device used in the specified CNC code.

CNC code default value

- Default Absolute Mode: when there is no G90 and G91 in CNC code, the default is Absolute Mode. The position in CNC code is Absolute Mode or Relative Mode can be switched by executing G90 and G91.
- Default Arc Interpolation Plane: When there is no G17/G18/G19 in CNC code, the default plane of arc interpolation is XY plane. The plane can be switched by G17/G18/G19.
- Cache mode default between CNC codes: the default is waiting mode. The cache mode can be switched via G50/G51/G52.
- Related default values in G0 code: the default values of velocity, acceleration, and leap degree of each axis in G0 are 1000, and the unit is travel unit. The above parameter values can be set by MC_SetMoveDirectParm instruction.
- Default values of velocity, acceleration, deceleration, and leap degree in G1, G2, and G3 codes: the default values of velocity, acceleration, deceleration, and leap degree in the relevant codes are 1000, and the unit

is travel unit. The default values of the above parameters can be set by MC_SetMoveLinearParm instruction. The speed, acceleration, deceleration of the related code can be set by the F and E parameters in the CNC code.

Inheritance of CNC code parameters

- Absolute Mode and Relative Mode Inheritance: After executing G90, if the following G codes do not execute G91, the position modes in the following G codes are all absolute mode, i.e., inheriting the execution of G90; if the execution of G90 is followed by the continuation of the execution of G91, the position modes in the G codes following G91 are all relative mode, i.e., inheriting the execution of G91. In the G-code shown below, the position in the G-code between N0~N6 is relative mode, and the position in the G-code after line N7 is absolute mode.
- Circular interpolation plane inheritance: the G code shown below, line N1 has execution G17, then the circular interpolation in line N9 still inherits the G17 plane in line N1.
- Inheritance of cache mode between CNC codes: For the following G-code, if G50 is executed in line N2, the cache mode between the other two lines of G-code inherits the G50 cache mode in line N2.
- Inheritance of parameter values in G0, G1, G2, G3 codes: The velocity and acceleration parameters in the same G code are inherited. As shown in the following G code, G0 in line N3 has set E (acceleration) and F (speed), and G0 in line N4 has not written E and F parameter values, but it will still inherit the E and F parameter values set in line N3, that is, the corresponding E and F parameter values of G0 in line N4 are the same as those in line N3. Similarly, the E and F parameter values of G1 in line N6 are the same as those in line N5, and the E and F parameter values of G2 in line N11 are the same as those in G2 in line N9.

```
N0 G91
N1 G17
N2 G50
N3 G0 X100 Y100 E100 E-100 F100
N4 G0 X200 Y200
N5 G1 X1000 Y1000 E100 E-100 F100
N6 G1 X2000 Y2000
N7 G90
N8 G1 X100 Y200
N9 G2 X500 Y1000 I400 J300 E100 E-100 F10
N10 G1 X100 Y200
N11 G2 X500 Y1000 I400 J300
```

8.2 G code function introduction

8.2.1 G90（Absolute mode）

- **Format:**
  N& G90
- **Example:**
  N0 G90
- **Description：**
  The & after the N in format represents the line number in the CNC code.
- **Function description：**
  Setting the position in the CNC code to the absolute mode (except for the center-of-circle coordinate position), the position in the CNC code indicates the position with the 0 position as the reference point. After this code is executed, the position in subsequent CNC codes is in absolute mode (except for the position of the center of circle coordinates) and can be switched to relative mode by executing the G91 code.

8.2.2 G91（Relative mode）

- **Format:**
  N& G91
- **Example:**
  N& G91
- **Description：**
  The & after the N in Format represents the line number in the CNC code.
- **Function description：**
  The position in the CNC code is set to the relative mode, and the position in the CNC code indicates the relative position using the position before the execution of the CNC code as the reference point. After this code is executed, the position in the subsequent CNC code is in relative mode and can be switched to absolute mode by executing the G90 code.

8.2.3 G0（Fast positioning）

- **Format:**
  N& G0 X& Y& Z& A& B& C& P& Q&
- **Example:**
  N0 G0 X100.1 Y100.2 Z200.1 A100.5 B100.2 C100.2 P100.2 Q100.2
- **Description：**

The meanings of the values following the relevant parameters in the G0 code are shown in the table below

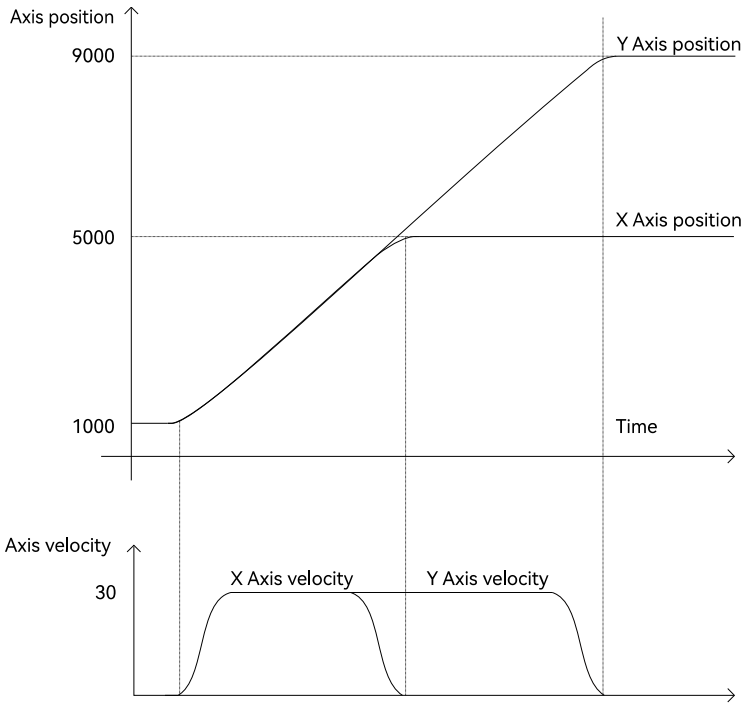
Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after X	LREAL	Positive, negative, 0	0	Target position of the logical axis X-axis in the axis group (Unit: Travel unit/s^2)*1
Value after Y	LREAL	Positive, negative, 0	0	Target position of the logical axis Y-axis in the axis group (Unit: Travel unit/s^2)*1
Value after Z	LREAL	Positive, negative, 0	0	Target position of the logical axisZ-axis in the axis group (Unit: Travel unit/s^2)*1
Value after A	LREAL	Positive, negative, 0	0	Target position of the logical axis A-axis in the axis group (Unit: Travel unit/s^2)*1
Value after B	LREAL	Positive, negative, 0	0	Target position of the logical axis B-axis in the axis group (Unit: Travel unit/s^2)*1
Value after C	LREAL	Positive, negative, 0	0	Target position of the logical axis C-axis in the axis group (Unit: Travel unit/s^2)*1
Value after P	LREAL	Positive, negative, 0	0	Target position of the logical axis P-axis in the axis group (Unit: Travel unit/s^2)*1
Value after Q	LREAL	Positive, negative, 0	0	Target position of the logical axis Q-axis in the axis group (Unit: Travel unit/s^2)*1

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "*Parameter description of motion control instructions*".

- **Code function description:**
  - This code is used to control multiple axes to move individually, up to 8 axes, all axes are started at the same time, and stopping is determined by the position specified for each axis in the CNC code. The position specified for each axis can be set in absolute or relative mode, and is set by G90 and G91. The target velocity, acceleration, deceleration, and jerk of each axis are set by the input variables Velocity, Acceleration, Deceleration, and Jerk of the MC_SetMoveDirectParm instruction.
  - X, Y, Z, A, B, C, P, Q in this code indicate the logical axes in the axes group, corresponding to the values 1~8 of the input variable IdentNum in the MC_AxesGroupAddAxis instruction. The correspondence between the logical axes X, Y, Z, A, B, C, P, Q and the configured axes in the software is determined by the value of IdentNum in the MC_AxesGroupAddAxis instruction and the value of Axis.
- **Absolute mode example:**
  - The initial position of both the X and Y axes is 1000, and the axis parameters are default values. The G-code that will be executed is as follows:

    N0 G90

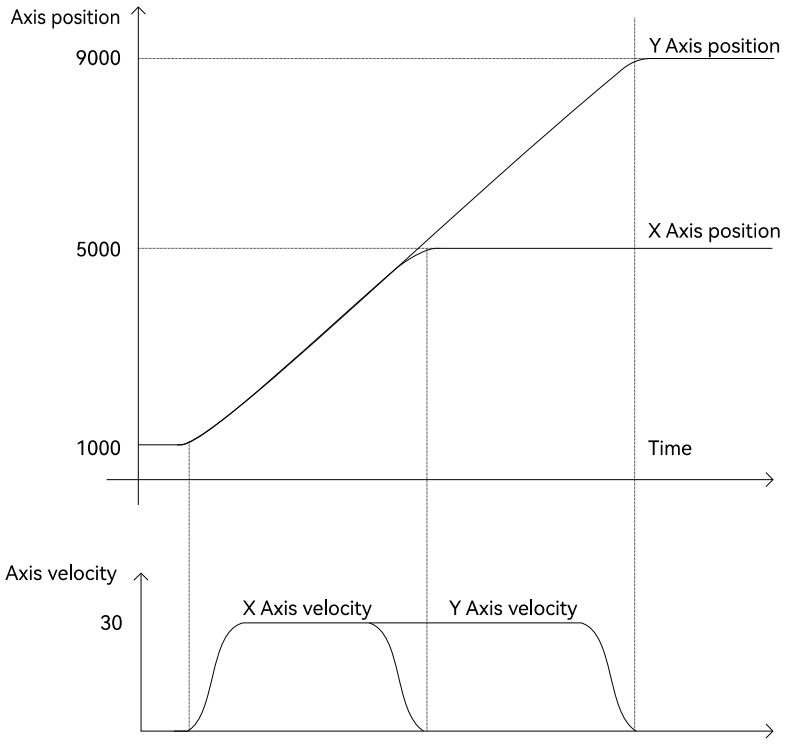
    N1 g0 x5000 y9000
  - After the G code is executed, the corresponding position and velocity curves of the X-axis and Y-axis are shown below:



- **Relative mode example:**
  - The initial position of both the X and Y axes is 1000, and the axis parameters are default values. The G-code that will be executed is as follows:

    N00 G91

    N01 g0 x4000 y8000
  - After the G code is executed, the corresponding position and velocity curves of X-axis and Y-axis are shown below:



8.2.4 G1 (Linear interpolation)

- **Format:**

  N& G1 X& Y& Z& A& B& C& P& Q& E& E& F&

- **Example:**

  N0 G1 X100.1 Y100.2 Z200.1 A100.5 B100.2 C100.2 P100.2 Q100.2 E10 E-20 F10

- **Description:**

Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after X	LREAL	Positive, negative, 0	0	Target position of the logical axis X-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Y	LREAL	Positive, negative, 0	0	Target position of the logical axis Y-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Z	LREAL	Positive, negative, 0	0	Target position of the logical axisZ-axis in the axis group (Unit: Travel unit/s^2)*2
Value after A	LREAL	Positive, negative, 0	0	Target position of the logical axis A-axis in the axis group (Unit: Travel unit/s^2)*2
Value after B	LREAL	Positive, negative, 0	0	Target position of the logical axis B-axis in the axis group (Unit: Travel unit/s^2)*2
Value after C	LREAL	Positive, negative, 0	0	Target position of the logical axis C-axis in the axis group (Unit: Travel unit/s^2)*2
Value after P	LREAL	Positive, negative, 0	0	Target position of the logical axis P-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Q	LREAL	Positive, negative, 0	0	Target position of the logical axis Q-axis in the axis group (Unit: Travel unit/s^2)*2
Value after E	LREAL	Positive	Required field	Specified acceleration *1pecified acceleration *1Specified deceleration *1 (Unit: Travel unit/s^2)*2
Value after E	LREAL	Negative	Required field	Specified deceleration *1 (Unit: Travel unit/s^2)*2
Value after F	LREAL	Positive	Required field	Specified acceleration *1 (Unit: Travel unit/s^2)*2

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "P*arameter description of motion control instructions*".

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

- **Function description:**

  - This code is used to control multiple axes to start and stop at the same time for linear interpolation action, up to 8 axes can be actuated at the same time. The target position of each axis is set by the input variables in the G-code. Input variable F is the synthesized speed of all axes, and input variable E is the synthesized acceleration and deceleration of all axes, and the value after E is a positive s-number to indicate the acceleration, and a negative number to indicate the deceleration.

  - X, Y, Z, A, B, C, P, Q in this code indicate the logical axes in the axis group, corresponding to the value 1~8 of the input variable IdentNum in the MC_AxesGroupAddAxis instruction. The correspondence

between logical axes X, Y, Z, A, B, C, P, Q and the configured axes in the software is determined by the value of IdentNum in the MC_AxesGroupAddAxis instruction and the value of Axis.

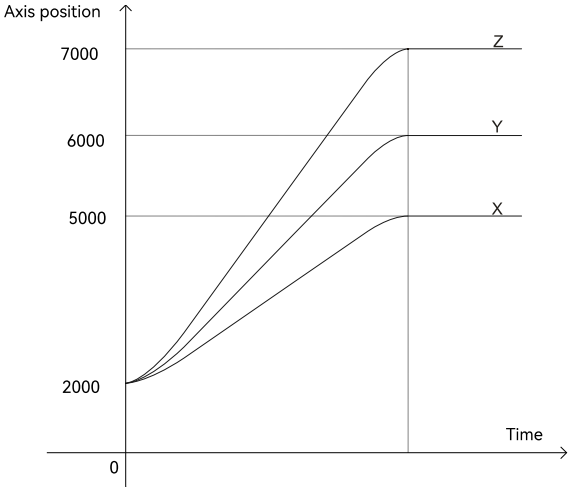
- The input variable F is the synthesized velocity for all axes, and the square of the synthesized velocity is the sum of the squares of the velocities of the axes in the code.
- When this code is used, which axes need to be controlled, enter the corresponding axes in X, Y, Z, A, B, C, P, and Q. Those that do not need to be controlled can be left out.

- **Absolute mode example:**
  - The initial positions of the X, Y, and Z axes are all 2000, and the axis parameters are all default values. The G-code that will be executed is as follows:

    N00 G90

    N01 g1 x5000 y6000 z7000

  - After the G code is executed, the correspondence between the X, Y, and Z axis position and time is shown below:

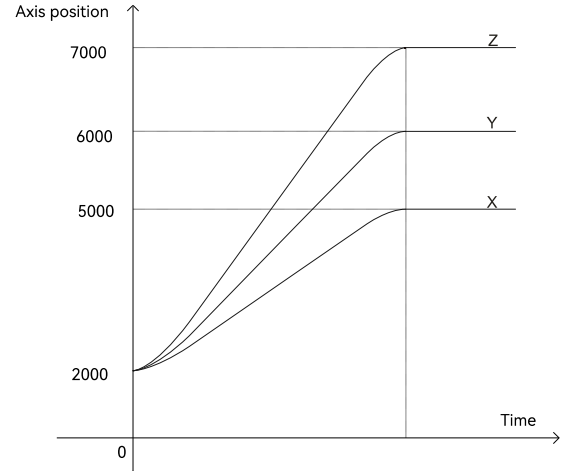


- **Relative mode Example:**
  - The initial positions of the X, Y, and Z axes are all 2000, and the axis parameters are all default values. The G-code that will be executed is as follows:

    N00 G90

    N01 g1 x3000 y4000 z5000

  - After the G code is executed, the correspondence between the X, Y, and Z axis position and time is shown below:



8.2.5 G2 (Clockwise circular/helical interpolation)

- **Format:**

  Format1：N& G2 X& Y& Z& A& B& C& P& Q& E& E& F& I& J& T&

  Format2：N& G2 X& Y& Z& A& B& C& P& Q& E& E& F& R& T&

- **Example:**

  Format1 Example:N0 G2 X500 Y1000 I400 J300 E100 E100 F10

  Explanation of Format1: Interpolation of arcs with specified center coordinates in XY plane.

  Format2 Example:N2 G2 X500 Y1000 R500 E100 E100 F10

  Explanation of Format2: Interpolation of arcs with specified radius in XY plane.

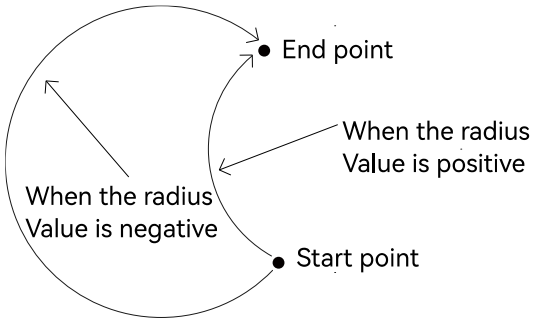
- **Description:**

Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after X	LREAL	Positive, negative, 0	0	Target position of the logical axis X-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Y	LREAL	Positive, negative, 0	0	Target position of the logical axis Y-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Z	LREAL	Positive, negative, 0	0	Target position of the logical axisZ-axis in the axis group (Unit: Travel unit/s^2)*2
Value after A	LREAL	Positive, negative, 0	0	Target position of the logical axis A-axis in the axis group (Unit: Travel unit/s^2)*2
Value after B	LREAL	Positive, negative, 0	0	Target position of the logical axis B-axis in the axis group (Unit: Travel unit/s^2)*2
Value after C	LREAL	Positive, negative, 0	0	Target position of the logical axis C-axis in the axis group (Unit: Travel unit/s^2)*2
Value after P	LREAL	Positive, negative, 0	0	Target position of the logical axis P-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Q	LREAL	Positive, negative, 0	0	Target position of the logical axis Q-axis in the axis group (Unit: Travel unit/s^2)*2
Value after I/I/J	LREAL	Positive, negative	0	Coordinates of the center of the circle Relative value of X direction in XY plane with respect to the starting point Relative value of X direction of XZ plane with respect to the starting point Relative value of YZ plane Y direction relative to the starting point. For more details, please refer to Code Function Description→Circle Center Coordinate Description section.
Value after/K/K	LREAL	Positive, negative	0	Coordinates of the center of the circle Relative value of Y direction of XY plane with respect to the starting point Relative value of the Z-direction of the XZ-plane with respect to the starting point. The relative value of the Z direction of the YZ

Input	Data type	Vaild range	Default	Description
				plane relative to the starting point. For more details, please refer to Code Function Description→Circle Center Coordinate Description section.
Value after R	LREAL	Positive, negative, 0	0	Radius of an arc
Value after T	ULINT	Positive	0	Number of circles
Value after E	LREAL	Positive	Required field	Specified acceleration *1 (Unit: Travel unit/s^2)*2
Value after E	LREAL	Negative	Required field	Specified deceleration *1 (Unit: Travel unit/s^2)*2
Value after F	LREAL	Positive	Required field	Specified acceleration *1 (Unit: Travel unit/s^2)*2

*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "P*arameter description of motion control instructions*".
*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

- **Code function description**
  - This instruction is used to control multiple axes to start and stop at the same time, two axes perform clockwise circular interpolation, and other axes follow the circular motion to start and stop at the same time. If X, Y and Z axes move at the same time, it is helical interpolation.
  - The two axes of clockwise circular interpolation are two axes of X, Y and Z. The specific two axes for circular interpolation are determined by the specified plane, and the default is XY plane. If G17 is specified as XY plane, X-axis and Y-axis will be interpolated; if G18 is specified as XZ plane, X-axis and Z-axis will be interpolated; if G19 is specified as YZ plane, Y-axis and Z-axis will be interpolated.
  - T is the number of circles, when T is 1, the running track of the code is a whole circle when the starting point and the ending point of the arc are the same; when the starting point and the ending point of the arc are different, the running track of the code is an arc; when the value of T is greater than 1, the running track of the code is an arc and the number of circles specified by T.
  - X, Y, Z, A, B, C, P, and Q in this code denote the logical axes in the axis group, and correspond to the value 1~8 of the input variable IdentNum in the MC_AxesGroupAddAxis instruction. The correspondence between logical axes X, Y, Z, A, B, C, P, Q and the configured axes in the software is determined by the value of IdentNum in the MC_AxesGroupAddAxis instruction and the value of Axis.
  - The input variable F is the synthesized velocity of all axes, and the square of the synthesized velocity is the sum of the squares of the velocities of the axes in the code.
  - When this code is used, E (acceleration) and F (target velocity) can be left out. When left out, it inherits the set values of E (acceleration) and F (target velocity) from the previous G code, and if E and F are not specified in all the G codes, you can set the E (acceleration) and F (target velocity) to be used with the MC_SetMovelinearParm instruction.
  - When this code is used, which axes need to be controlled, enter the corresponding axes in X, Y, Z, A, B, C, P, Q. Those that do not need to be controlled can be left out.
  - When the radius is specified for arc interpolation, the code runs on a shorter arc when the value of the R (radius) parameter is positive; when the value of the R (radius) parameter is negative, the code runs on a longer arc. The schematic diagram is shown below.

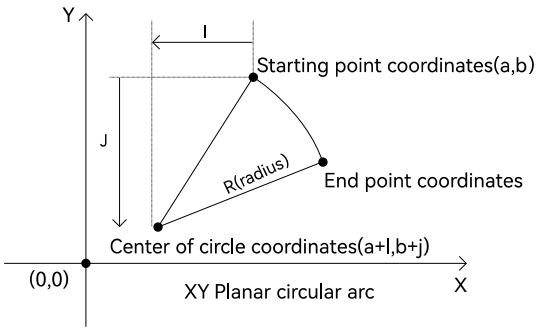
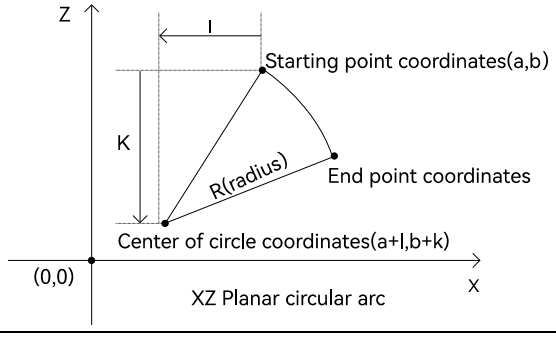
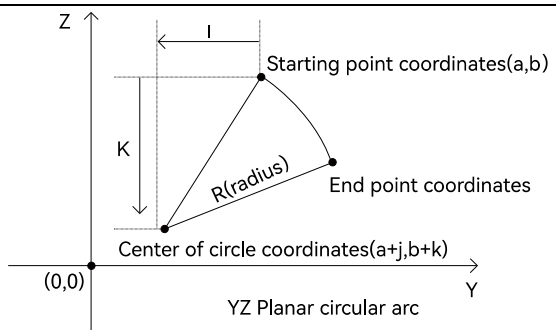


➢ Description of Circle Center Coordinate

When specifying the center of the circle for circular interpolation, the center of the circle coordinate represents the relative value of the coordinates of the two directions in the specified plane with respect to the starting point of the arc. Whether in relative value mode (set by G91) or absolute mode (set by G90), the center of circle coordinate is the relative value relative to the starting point of the arc.

When different planes are selected, the parameters of the centroid value are different, and the planes can be selected through G17, G18 and G19. When XY plane is selected, the centroid value is indicated by I and J; when XZ plane is selected, the centroid value is indicated by I and K; when YZ plane is selected, the centroid value is indicated by J and K. The following table shows the relative value of the centroid coordinates when different planes are selected.

The following table shows the relationship between the center of the circle coordinates and the starting point coordinates in different planes as shown in the figure below.

Plane	Relationship between the coordinates of the center of the circle and the coordinates of the starting point
When the center of circle value is set to I and J	
XZ plane When the center of circle value is set to I and K	
YZ plane When the center of circle value is set to J and K	

- **Example1: Arc interpolation when center or radius is specified in absolute mode.**

The current position of X-axis and Y-axis are 100 and 200, that is, the starting point of the arc in the following figure (100,200), and the end point of the arc is (500,1000), that is, the end point of the arc in the following figure, and the center of the circle is (500,500), but the center of the circle in the program is written as (400,300) because the center of the circle is a relative value relative to the starting point. In absolute mode, the start and end positions of the arc are absolute positions, and the center position is relative to the start point.

- **G-code writing method when specifying the center of a circle**

N0 G90

N1 G17

N2 G2 X500 Y1000 I400 J300 E100 E-100 F10

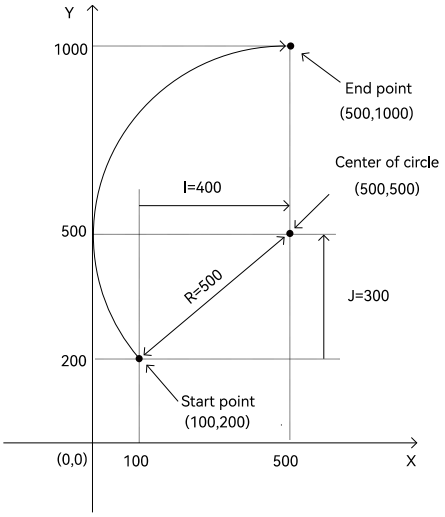
- **G-code writing method when specifying a radius**

N0 G90

N1 G17

N2 G2 X500 Y1000 R500 E100 E-100 F10

- **After the execution of the above G code, the arc trajectory corresponding to the XY plane is shown below.**



- **Example2: Arc interpolation in relative mode when center or radius is specified.**

➢ The current position of X-axis and Y-axis are 100 and 200, that is, the starting point of the arc (100,200) in the following figure, and the end point of the arc is (500,1000), that is, the end point of the arc in the following figure, but the end point of the arc written in the program is (400,800), because the current mode is relative mode, and the end point of the arc written in the program is a relative value with respect to the starting point. The center of the circle written in the program is (400,300), and the coordinates of the center of the circle are also relative to the starting point.

- **G-code writing method when specifying the center of the circle**

N0 G91

N1 G17

N2 G2 X400 Y800 I400 J300 E100 E-100 F10

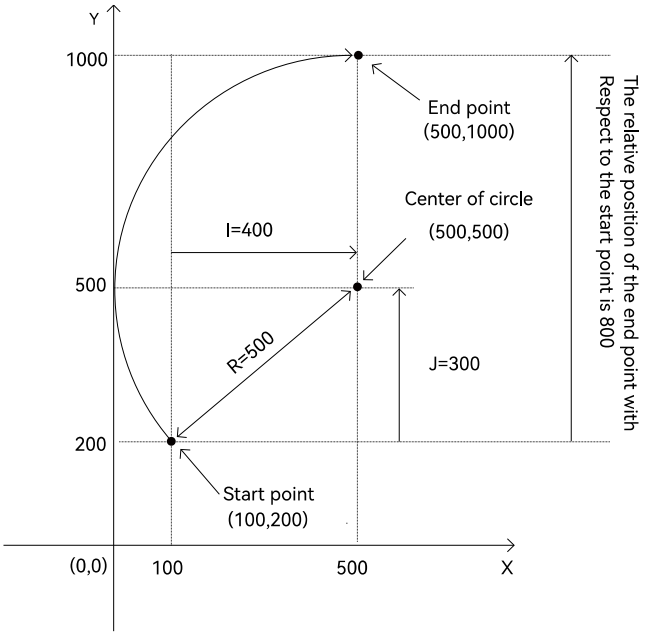
- **G-code writing method when specifying a radius**

N0 G91

N1 G17

N2 G2 X400 Y800 R500 E100 E-100 F10

- **After the execution of the above G code, the arc trajectory corresponding to the XY plane is shown below.**



■ **Example3: Whole circle interpolation when the center of the circle is specified in absolute mode:**

- The current positions of X-axis and Y-axis in the axis group are 100 and 200, that is, the starting point of the arc (100,200) in the following figure, and the end position and the starting point of the arc are the same position, that is, the end point of the arc in the following figure, and the center of the circle is (400,300). In this mode, the start position and the end position of the arc are absolute positions, and the center position is relative to the start point.

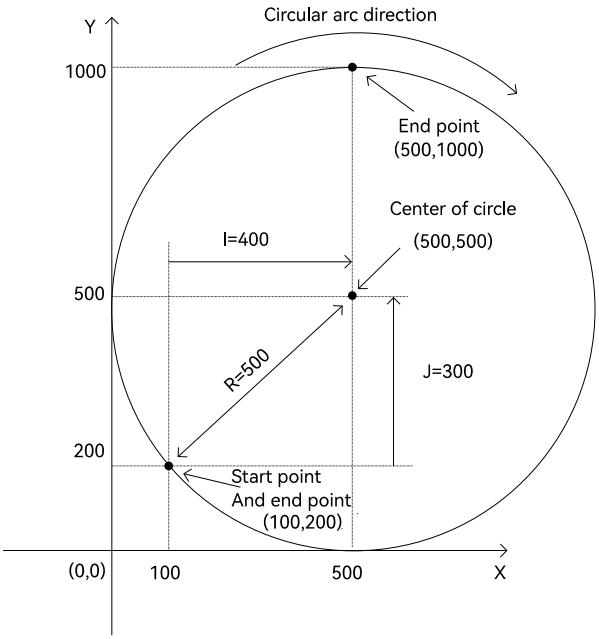
- **G-code writing method when specifying the center of the circle**

N0 G90

N1 G17

N2 G2 X100 Y200 I400 J300 E100 E100 F10

- **After the execution of the above G code, the arc trajectory corresponding to the XY plane is shown below.**



8.2.6 G3 (Counterclockwise circular/helical interpolation.)

- **Format:**

  Format1：N& G3 X& Y& Z& A& B& C& P& Q& E& E& F& I& J& T&

  Format2：N& G3 X& Y& Z& A& B& C& P& Q& E& E& F& R& T&

- **Example:**

  Format1 Example:N0 G3 X100 Y200 I0 J-500 E100 E100 F10

  Format1 description: Interpolation of arcs with specified center coordinates in the XY plane.

  Format2 Example:N2 G3 X100 Y200 R500 E100 E100 F10

  Format2 description: Interpolation of arcs with specified radius in the XY plane.

- **Description:**

The & in Format represents the value of the relevant parameter, and the meaning of the value after the relevant parameter in the code is shown in the following table:

Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after X	LREAL	Positive, negative, 0	0	Target position of the logical axis X-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Y	LREAL	Positive, negative, 0	0	Target position of the logical axis Y-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Z	LREAL	Positive, negative, 0	0	Target position of the logical axisZ-axis in the axis group (Unit: Travel unit/s^2)*2
Value after A	LREAL	Positive, negative, 0	0	Target position of the logical axis A-axis in the axis group (Unit: Travel unit/s^2)*2
Value after B	LREAL	Positive, negative, 0	0	Target position of the logical axis B-axis in the axis group (Unit: Travel unit/s^2)*2
Value after C	LREAL	Positive, negative, 0	0	Target position of the logical axis C-axis in the axis group (Unit: Travel unit/s^2)*2
Value after P	LREAL	Positive, negative, 0	0	Target position of the logical axis P-axis in the axis group (Unit: Travel unit/s^2)*2
Value after Q	LREAL	Positive, negative, 0	0	Target position of the logical axis Q-axis in the axis group (Unit: Travel unit/s^2)*2
Value after I/I/J	LREAL	Positive, negative	0	Coordinates of the center of the circle Relative value of X direction in XY plane with respect to the starting point Relative value of X direction of XZ plane with respect to the starting point Relative value of YZ plane Y direction relative to the starting point. For more details, please refer to Code Function Description→Circle Center Coordinate Description section.

Input	Data type	Vaild range	Default	Description
Value after J/K/K	LREAL	Positive, negative	0	Coordinates of the center of the circle Relative value of Y direction of XY plane with respect to the starting point Relative value of the Z-direction of the XZ-plane with respect to the starting point. The relative value of the Z direction of the YZ plane relative to the starting point. For more details, please refer to Code Function Description→Circle Center Coordinate Description section.
Value after R	LREAL	Positive, negative, 0	0	Radius of an arc
Value after T	ULINT	Positive	0	Number of circles
Value after E	LREAL	Positive	Required field	Specified acceleration *1 (Unit: Travel unit/s^2)*2
Value after E	LREAL	Negative	Required field	Specified deceleration *1 (Unit: Travel unit/s^2)*2
Value after F	LREAL	Positive	Required field	Specified acceleration *1 (Unit: Travel unit/s^2)*2

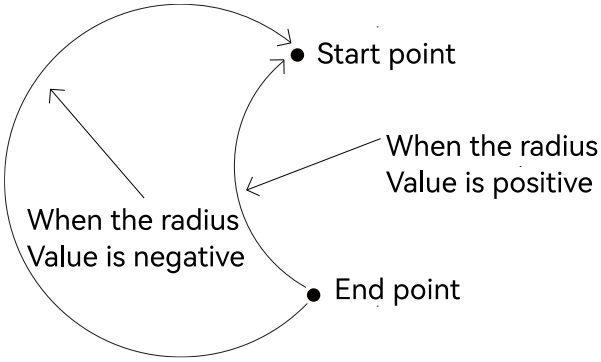
*1: For the relation among Velocity, Acceleration, Deceleration and Jerk, please refer to "P*arameter description of motion control instructions*".

*2: For details of the instruction units, please refer to "*Parameter unit of motion control instructions*".

- **Code function description**
  - This instruction is used to control multiple axes to start and stop at the same time, two axes perform counterclockwise circular interpolation, and other axes follow the circular motion to start and stop at the same time. If X, Y and Z axes move at the same time, it is helical interpolation.
  - The two axes of counterclockwise circular interpolation are the two axes of X, Y and Z. The specific two axes for circular interpolation are determined by the specified plane, and the default is XY plane. If G17 is specified as XY plane, X-axis and Y-axis will be interpolated; if G18 is specified as XZ plane, X-axis and Z-axis will be interpolated; if G19 is specified as YZ plane, Y-axis and Z-axis will be interpolated.
  - T is the number of circles, when T is 1, the running track of the code is a whole circle when the start point and the end point of the circle are the same; when the start point and the end point of the circle are different, the running track of the code is an arc; when T is greater than, the running track of the code is an arc and the number of circles specified by T.
  - When the center of the circle is specified for circular interpolation, the coordinates of the center of the circle indicate the relative values of the coordinates of the two directions in the specified plane with respect to the starting point of the arc. Whether in relative value mode (set by G91) or absolute mode (set by G90), the center of circle coordinate is the relative value with respect to the starting point of the arc.
  - X, Y, Z, A, B, C, P, and Q in this code denote the logical axes in the axis group, corresponding to values 1 to 8 of the input variable IdentNum in the MC_AxesGroupAddAxis instruction. The correspondence between logical axes X, Y, Z, A, B, C, P, Q and the configured axes in the software is determined by the value of IdentNum in the MC_AxesGroupAddAxis instruction and the value of Axis.
  - The input variable F is the synthesized velocity of all axes, and the sum of the squares of the synthesized velocities is the sum of the squares of the velocities of the axes in the code.
  - When this code is used, E (Acceleration) and F (Target Velocity) can be omitted and not written, when not written, it will inherit the set values of E (Acceleration) and F (Target Velocity) in the previous G

code, if E and F are not specified in all the G codes, E (Acceleration) and F (Target Velocity) to be used can be set by the MC_SetMovelinearParm instruction.

- When this code is used, which axes need to be controlled, enter the corresponding axes in X, Y, Z, A, B, C, P, Q. Axes that do not need to be controlled can be left out!
- Specify the radius for arc interpolation, R (radius) parameter after the value of a positive number, the running track of the code for the shorter arc; R (radius) parameter after the value of a negative number, the running track of the code for the longer arc. The schematic diagram is shown below.

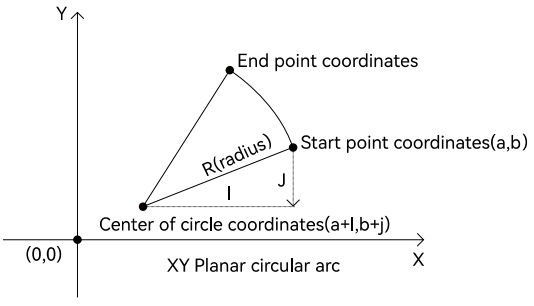
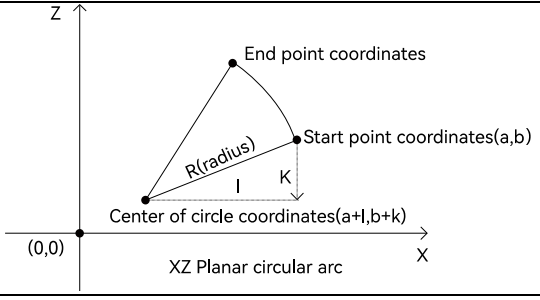


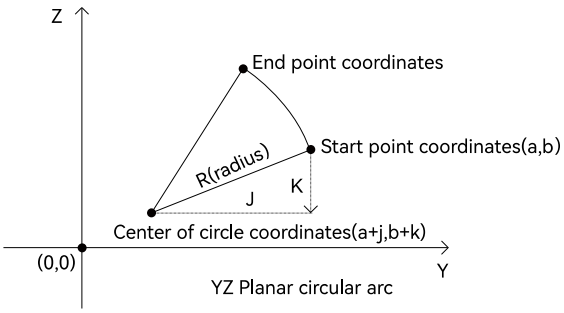
- Description of circle center coordinates.

When specifying the center of the circle for circular interpolation, the center of the circle coordinate represents the relative value of the coordinates of the two directions in the specified plane with respect to the coordinates of the starting point of the arc. Whether in relative value mode (set by G91) or absolute mode (set by G90), the center coordinate is the relative value relative to the starting point of the arc.

When different planes are selected, the parameters of the centroid value are different, and the planes can be selected through G17, G18 and G19. When XY plane is selected, the centroid value is indicated by I and J; when XZ plane is selected, the centroid value is indicated by I and K; when YZ plane is selected, the centroid value is indicated by J and K. The following table shows the relative value of the centroid coordinates when different planes are selected.

The following table shows the relationship between the center of the circle coordinates and the starting point coordinates in different planes as shown in the figure below.

Plane	Relationship between the coordinates of the center of the circle and the coordinates of the starting point
XY plane When the center of circle value is set to I and J	
XZ plane When the center of circle value is set to I and K	

YZ plane When the center of circle value is set to J and K	

■ **Example1: Arc interpolation when center or radius is specified in absolute mode.**

- The current positions of the X-axis and Y-axis are 500 and 1000, that is, the starting point of the arc in the figure below is (500,1000), the end point of the arc is (100,200), and the center of the circle is (500,500), but the center of the circle written in the program is (0,-500) because the center of the circle written in the program is a relative value with respect to the starting point. In absolute mode, the start and end positions of the arc are absolute positions, and the center position is a relative value with respect to the start point.
- **G-code writing method when specifying the center of a circle**

  N0 G91

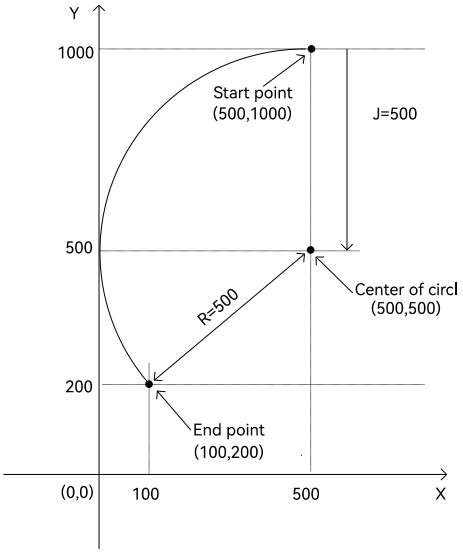
  N1 G17

  N2 G3 X100 Y200 I0 J-500 E100 E100 F10
- **G-code writing method when specifying a radius**

  N0 G91

  N1 G17

  N2 G3 X100 Y200 R500 E100 E100 F10
- **After the execution of the above G code, the arc trajectory corresponding to the XY plane is shown below.**



■ **Example2: Arc interpolation in relative mode when center or radius is specified.**

- The current positions of X-axis and Y-axis are 500 and 1000, i.e., the starting point of the arc in the following figure is (500,1000), the end position of the arc is (100,200), and the center of the circle is (500,500), but the end position written in the program is (-400,-800), and the center of the circle is (0,-500), under the Relative Mode, the end position of the arc and the center of the circle written in the program are the relative positions with respect to the starting point.

- **G-code writing method when specifying the center of a circle**

  N0 G90

  N1 G17

  N2 G3 X-400 Y-800 I0 J-500 E100 E100 F10

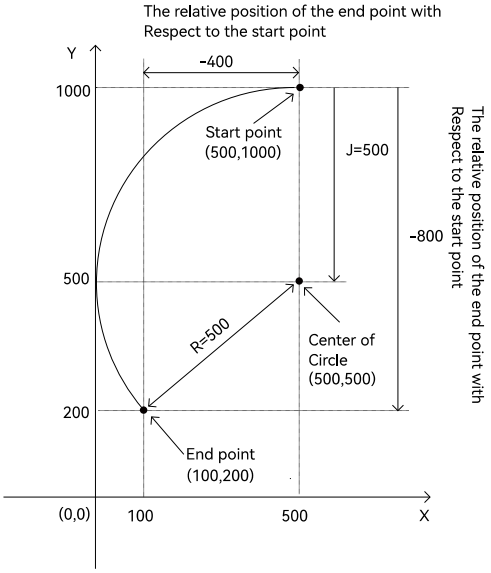
- **G-code writing method when specifying a radius**

  N0 G90

  N1 G17

  N2 G3 X-400 Y-800 R500 E100 E100 F10

- **After the execution of the above G code, the arc trajectory corresponding to the XY plane is shown below.**



■ **Example3: Whole circle interpolation when the center of the circle is specified in absolute mode:**

- The current positions of X-axis and Y-axis are 100 and 200, i.e., the starting point of the arc in the following figure is (100,200), and the end position of the arc is the same as the starting point, i.e., the starting point and the end position of the arc are both (100,200). The center of the circle is at (500,500), but the center of the circle written in the program is at (400,300), because the center of the circle written in the program is a relative value with respect to the starting point. In absolute mode, the arc start position and end position are absolute positions, and the circle center position is a relative value with respect to the start point.

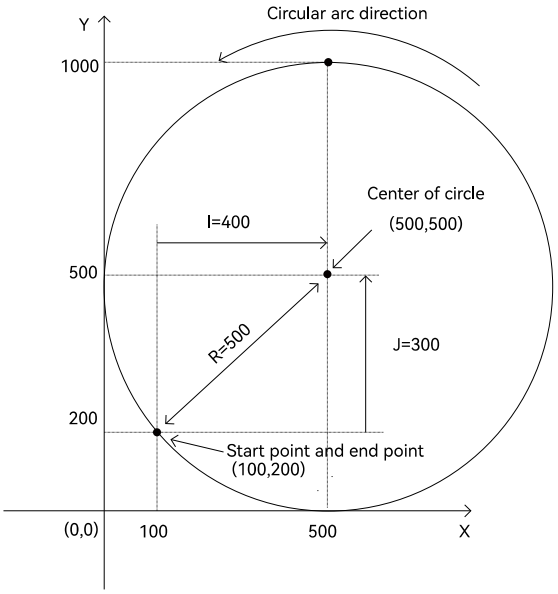
- **G-code writing method when specifying the center of a circle**

  N0 G90

  N1 G17

  N3 G3 X100 Y200 I400 J300 E100 E100 F10

- **After the execution of the above G code, the arc trajectory corresponding to the XY plane is shown below.**



8.2.7 G17/G18/G19 (Specify the arc interpolation plane)

- **Format:**

  N& G&

- **Example:**

  Format1 Example:N0 G17

  Format2 Example:N0 G18

  Format3 Example:N0 G19

- **Description:**

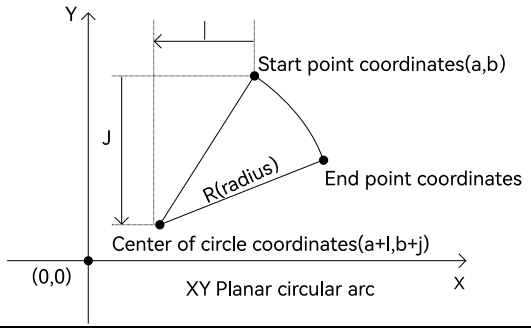
  The & after N in Format represents the line number in the CNC code.

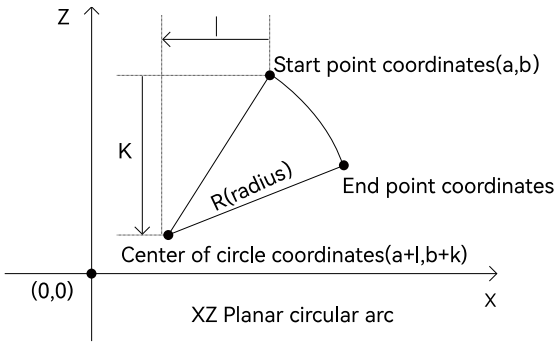
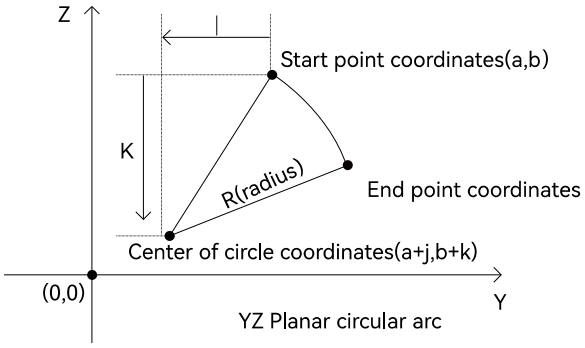
  The & after G in Format represents the corresponding G code, which can be 17, 18 or 19.

- **Code function description**

G17, G18 or G19 are used to set the plane for the arc operation. G17 specifies the arc plane as XY plane, G18 specifies the arc plane as XZ plane and G19 specifies the arc plane as YZ plane.

The schematic diagrams of the different planes are shown in the following table:

Plane	Relationship between the coordinates of the center of the circle and the coordinates of the starting point
XY plane (G17) When the center of circle value is set to I and J	

XZ plane (G18) When the center of circle value is set to I and K	
YZ plane (G19) When the center of circle value is set to J and K	

8.2.8 G4 (Delay instruction)

- **Format:**

  N& G4 K&

- **Example:**

  N0 G4 K2

- **Description:**

Format in the & on behalf of the value of the relevant parameters, the code of the relevant parameters after the value of the meaning of the following table:

Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after K	LREAL	0.001~100000	0	The set delay time. (Unit: second)

- **Code function description:**

  This code is used to set the specified delay time. The next G code is executed when the set delay time is up.

8.2.9 G50 (No transition curve mode)

- **Format:**

  N& G50

- **Example:**

  N0 G50

- **Description:**

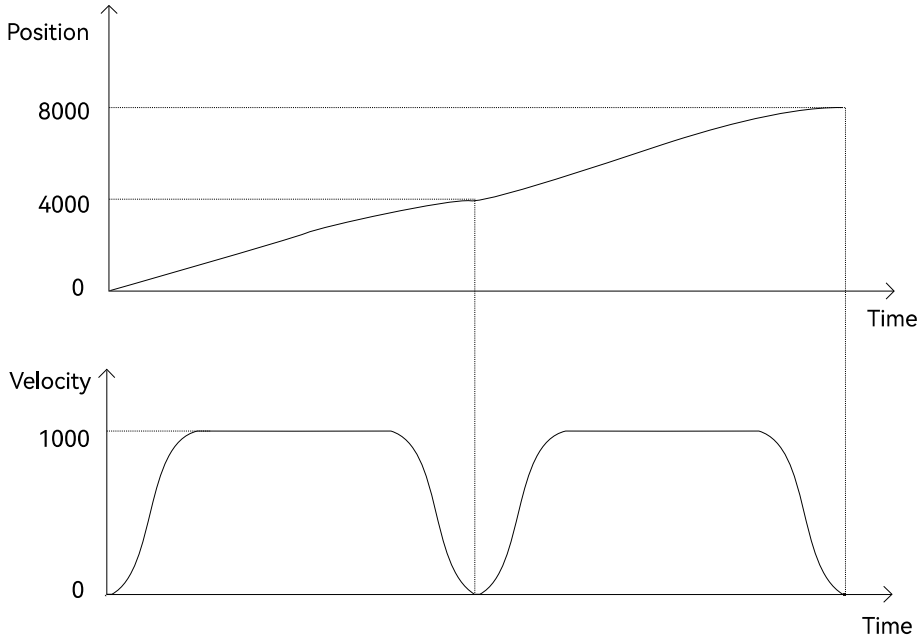
The & after the N in Format represents the line number in the CNC code.

- **Code function description:**
- This code is used to set the transition mode between two adjacent codes of CNC code, the Function of this code indicates that when the last CNC code position is reached, the speed is also reduced to 0 at the same time, and then the execution of the next line of CNC code is started.
- The transition mode between two codes of CNC code is G50 by default, which can be switched by G51 and G52 codes, and all transitions between G codes can use G50.
- **Function Example:**

  N00 G50

  N01 G1 X4000 Y4000 E5000 E-5000 F1000

  N02 G1 X8000 Y8000 E5000 E-5000 F1000



8.2.10 G51 (Specify additional corner transition modes)

- **Format:**

  N& G51 D&

- **Example:**

  N0 G51 D100

- **Description:**

Format in the & on behalf of the value of the relevant parameters, the code of the relevant parameters after the value of the meaning of the following table:

Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after ND	LREAL	Positive	0	Specified transition distance

- **Code function description:**

➢ This code is used to set the transition mode between two adjacent codes of CNC code. The function of this code indicates that the next line of CNC code is executed when the previous line of CNC code reaches the end position without the speed dropping to 0. There is an arc transition between the end position of the previous line and the start position of the next line of CNC code, and the interpolating speed does not drop to 0 during the transition.

➢ The transition mode can be specified as G51 only when the interpolation motion G code is executed, but not for other G codes. The transition between two CNC codes can be switched by the G50 and G51 codes.

➢ When the G51 transition mode is used between two neighboring codes, the transition start point is the end position in the previous line of G code minus the transition distance specified in that code, and the transition end point is the end position in the previous line of G code plus the transition distance specified in that code. For details, refer to the description in Example.

➢ When the G51 transition mode is used between two adjacent codes, the speed of the G code is based on the synthesis speed specified in the first G code after the G51 code. If the subsequent G code does not use G51 again, the subsequent G code still uses the synthesis speed specified in the first G code after the G51 code even if the synthesis speed is set by F. If you want to change the speed during G code execution, you can change the synthesis speed by changing the value of the MC_CoorMotion instruction input variable VelOverride.

➢ When multiple CNC codes are executed, the following table shows the cases in which G51 can be selected for the transition mode between two neighboring CNC codes:

Different G-code transitions
Transition between G1 and G1
Transition between G1 and G2 or Transition between G2 and G1
Transition between G1 and G3 or Transition between G3 and G1
Transition between G2and G3 or Transition between G3 and G2
Transition between G2 and G2 or Transition between G3 and G3

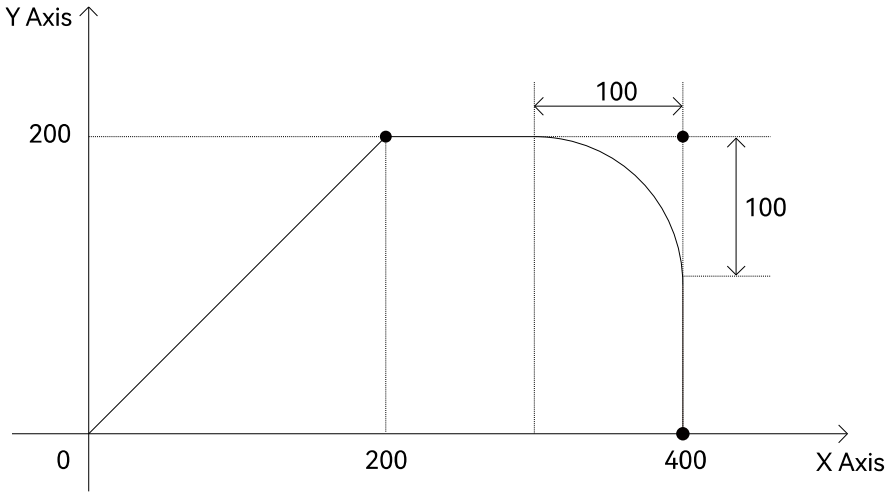
■ **Function Example1:**

N00 G1 X200 Y200

N01 G51 D100

N02 G1 X400 Y200

N03 G1 X400 Y0

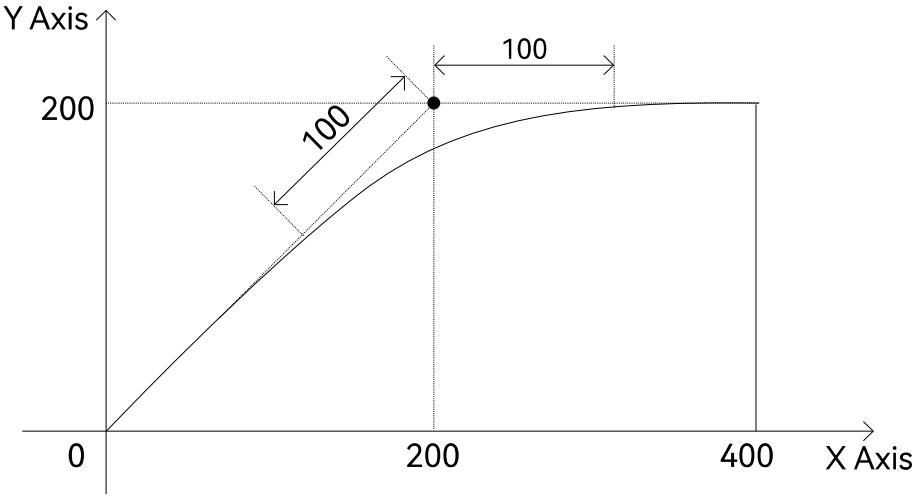


■ **Function Example2:**

N00 G51 D100

N01 G1 X200 Y200

N02 G1 X400 Y200



8.2.11 G52 (Transition at constant speed mode)

- **Format:**

  N& G52

- **Example:**

  N0 G52

- **Description:**

  The & after the N in Format represents the line number in the CNC code.

- **Code function description:**

  - This code is used to set the transition mode between two adjacent codes of CNC code. The Function of this code indicates that when the previous line of CNC code reaches the end position, the next line of CNC code will be executed without the speed dropping to 0, and the transition will be performed at the set speed (the synthesized speed of the previous line of CNC code).
  - The transition mode can be specified as G52 only when executing the G1, G2, and G3 codes, but not

the other G codes. The transition between two CNC codes can be switched by the G51 and G52 codes.

- This mode is generally used when the incremental position between multiple CNC codes is small, and the curve is fitted by multiple lines of CNC code positions.
- When the G52 transition mode is used between two adjacent codes, the speed of the G code is based on the synthesized speed specified in the first G code after the G52 code, and if the subsequent G code does not use G52 again, the subsequent G code uses the synthesized speed specified in the first G code after the G52 code even if it is set by F. If you want to change the speed during G code execution, you can change the synthesis speed by changing the value of the MC_CoorMotion instruction input variable VelOverride.
- The following table shows the cases in which G52 can be selected for the transition mode between two neighboring CNC codes when multiple CNC codes are executed:

Different G-code transitions	Condition
Transition between G1 and G1	Executable
Transition between G1 and G2 or Transition between G2 and G1	It can only be executed when a straight line is tangent to an arc.
Transition between G1 and G3 or Transition between G3 and G1	It can only be executed when a straight line is tangent to an arc.
Transition between G2and G3 or Transition between G3 and G2	It can only be executed when a straight line is tangent to an arc.
Transition between G2 and G2 or Transition between G3 and G3	It can only be executed when a straight line is tangent to an arc.

8.2.12 M codes

- **Format:**

  N& M& D&

- **Example:**

  Example1：N0 M0 D123.12

  Example2：N1 G1 X100.5 Y100.5 E100 E-100 F30.5 M0 D123.12

- **Description：**

  The & after the N in format represents the line number in the CNC code.

Input	Data type	Vaild range	Default	Description
Value after N	UINT	0~65535	Required field	Line number in CNC code
Value after M	USINT	0~99	0	M code number
Value after ND	LREAL	Positive, negative, 0	0	Specify the value corresponding to the execution of the M code

- **Code Function description:**
  - M code is generally used to interact with G code and program variables in other tasks. For example, M code can be used to trigger the controller body outputs to turn on or off (body outputs can be used to control the cylinder action, etc.), and M code can be used to determine where the G code has been executed.
  - M code and M code of the D parameter does not have a specific meaning, the user in the use, you can give it a specific meaning. When using M code, when executing to the line where M code is located, the

corresponding M code will be TRUE, and the corresponding D parameter will be assigned to the specified value of M code, the status of M code and the value of D parameter can be read by MC_GetMCodeStatus instruction, and the status of M code needs to be reset by MC_ResetMCode instruction, and the value of M code will not be automatically changed to FALSE after it becomes TRUE. automatically change to FALSE.

- M code can be on a separate line in CNC code or on the same line with other G codes.
- When M code and G code are on the same line, M code should be placed after G code. When execution reaches the line where the M code is located (when the G code corresponding to that line starts to execute), the status corresponding to the M code is TRUE and the value of the D parameter is the value specified by the M code. After the execution of the G code of the line is completed, the execution of the next line of CNC code continues. Refer to Example1 for details.
- When M code is executed on a separate line, when the execution reaches the line where the M code is located (when the execution of the corresponding M code on that line starts), the state corresponding to the M code is TRUE and the value of the D parameter is the value specified by the M code. The CNC code after the M code of the line stops execution, and it is necessary to reset the M code to continue execution, and the corresponding M code can be reset by the MC_ResetMCode instruction. Refer to Example2 for details.

■ **Example1: M code and G code on the same line**

- Related CNC code

```
N1 G1 X100.5 Y100.5 E100 E-100 F30.5
N2 G1 X300.5 Y300.5 E100 E-100 F30.5    M0 D123.12
N3 G1 X500.5 Y500.5 E100 E-100 F30.5
```

- When the CNC code is executed to line N2 (line N2 has just begun execution), the value of M0 is TRUE and the value of the D parameter is 123.2. After the execution of the G1 code in line N2 is completed, the execution of the G code in line N3 continues.

■ **Example2: M code and G code on the different line**

- Related CNC code

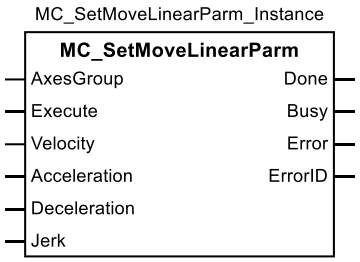
```
N1 G1 X100.5 Y100.5 E100 E-100 F30.5
N2 M1 D123.12
N3 G1 X300.5 Y300.5 E100 E-100 F30.5
N4 G1 X500.5 Y500.5 E100 E-100 F30.5
```

- When the CNC code is executed to line N2, the value of M1 is TRUE and the value of the D parameter is 123.12. The CNC code in lines N3 and N4 is not executed, and M1 continues to execute the G code in lines N3 and N4 after it is reset by the MC_ResetMCode instruction.

8.3 MC_SetMoveLinearParm（Default Parameter Settings for CNC Interpolation ）

This instruction is used to set the default motion parameters needed for CNC interpolation motion. Library：MotionControl_Part2

Applicable models: M500 series（M500s series not included）

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetMovelinearParm	Default Parameter Settings for CNC Interpolated Motion	FB		MC_SetMoveLinearParm _Instance (AxesGroup:=parameter , Execute:=parameter , Velocity:=parameter , Acceleration :=parameter, Deceleration:=parameter , Jerk :=parameter, Done=> parameter , Busy=> parameter , Error=> parameter , ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	M500 series：1~8 Other devices：1	Required field	Axis group number
Execute	Execute	BOOL	TRUE OR FALSE	FALSE	Executing when detecting the rising edge
Velocity	Interpolation velocity	LREAL	Positive	Required field	Interpolation velocity
Acceleration	Interpolation acceleration	LREAL	Positive	Required field	Interpolation acceleration
Deceleration	Interpolation deceleration	LREAL	Positive	Required field	Interpolation deceleration
Jerk	Interpolation jerk	LREAL	Positive	Required field	Interpolation jerk

■ Output variable

Name	Meaning	Data type	Valid range	Function
Done	Done	BOOL	TRUE OR FALSE	Change to TRUE after execution of the instruction is complete
Busy	Executing	BOOL	TRUE OR FALSE	Change to True when instruction is executing.
Error	Error	BOOL	TRUE OR FALSE	Change to True when error occurs.
ErrorID	Error code	WORD	0~65535	Output error code when instruction execution error occurs Please refer to the ‘Instruction error code description’

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After instruction execution is completed	When Done is TRUE, and Execute changes from TRUE to FALSE When the instruction is executed and the Execute is FALSE, Done becomes True and after one cycle, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	Done becomes TRUE Error becomes TRUE

Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

This instruction is used to set the default speed, acceleration, deceleration, and jump parameters used in G1, G2, and G3 of CNC. The variables Velocity, Acceleration, Deceleration, and Jerk are inputted through this instruction for setting. When interpolation speed, acceleration, and deceleration are not specified through E and F in G1, G2, and G3, use the interpolation speed, acceleration, and deceleration set in this command. The value of Jerk in G1, G2, and G3 can only be set through this command.

● Instruction completion timing

After this instruction executes, axis group will stop in one cycle, then the instruction is complete, and Done changes from FALSE to TRUE.

● Abnormality elimination

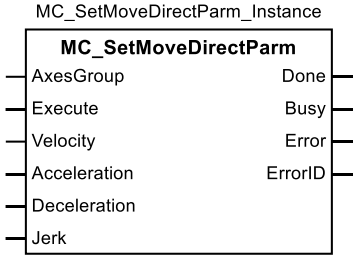
When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become True, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

8.4 MC_SetMoveDirectParm (CNC individual positioning motion parameters setting)

This instruction is used to set the motion parameters required for CNC individual positioning.

Library：MotionControl_Part2

Applicable models: M500 series（M500s series not included）

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetMoveDirectParm	CNC individual positioning motion parameter setting	FB		MC_SetMoveDirectParm_Instance (AxesGroup:=parameter , Execute:=parameter, Velocity:=parameter , Acceleration:=parameter , Deceleration:=parameter , Jerk:=parameter , Done=> parameter , Busy=> parameter , Error=> parameter , ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	1~8	Required field	Axis group number
Execute	Execute	BOOL	TRUE OR FALSE	FALSE	Executing when detecting the rising edge
Velocity	Fast positioning velocity	ARRAY [1..8] OF LREAL	Positive	Required field	Fast positioning velocity
Acceleration	Fast positioning acceleration	ARRAY [1..8] OF LREAL	Positive	Required field	Fast positioning acceleration
Deceleration	Fast positioning deceleration	ARRAY [1..8] OF LREAL	Positive	Required field	Fast positioning deceleration
Jerk	Fast positioning jerk	ARRAY [1..8] OF LREAL	Positive	Required field	Fast positioning jerk

■ Output variable

Name	Meaning	Data type	Valid range	Function
Done	Done	BOOL	TRUE OR FALSE	Change to TRUE after execution of the instruction is complete
Busy	Executing	BOOL	TRUE OR FALSE	Change to True when instruction is executing.
Error	Error	BOOL	TRUE OR FALSE	Change to True when error occurs.
ErrorID	Error code	WORD	0~65535	Output error code when instruction execution error occurs Please refer to the ‘Instruction error code description’

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After instruction execution is completed	When Done is TRUE, and Execute changes from TRUE to FALSE When the instruction is executed and the Execute is FALSE, Done becomes True and after one cycle, it becomes FALSE

Busy	When Execute changes from FALSE to TRUE	Done becomes TRUE Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	When Execute changes from FALSE to TRUE

Function description

Basic function description

This instruction is used to set the velocity, acceleration, deceleration, and jerk parameters to be used in G0 in the CNC, which are set by inputting the variables Velocity, Acceleration, Deceleration, and Jerk in this instruction. The position to be used in G0 is set in G0.When G0 is used in the CNC, this instruction is used to set the motion parameters that are required for each axis first.

Instruction completion timing

After this instruction executes, axis group will stop in one cycle, then the instruction is complete, and Done changes from FALSE to TRUE.

Abnormality elimination

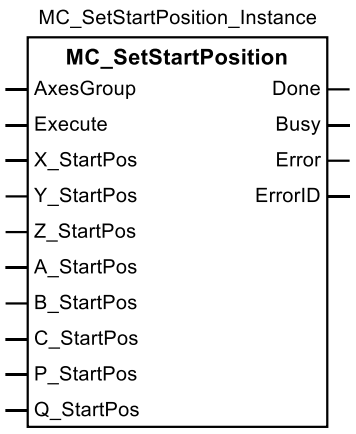
When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become True, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error

8.5 MC_SetStartPosition (Axis group initial position setting)

T This instruction is used to set the initial position of each axis in the specified axis group.

Library：MotionControl_Part2

Applicable models: M500 series（M500s series not included）

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_SetStartPosition_Instance	Axis group initial position setting	FB		MC_SetStartPosition_Instance (AxesGroup:=parameter, Execute:=parameter , X_StartPos :=parameter, Y_StartPos:=parameter , Z_StartPos:=parameter , A_StartPos :=parameter, B_StartPos:=parameter , C_StartPos:=parameter , P_StartPos:=parameter , Q_StartPos:=parameter , Done=> parameter , Busy=> parameter , Error=> parameter, ErrorID => parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	1~8	Required field	Axis group number
Execute	Execute	BOOL	TRUE OR FALSE	FALSE	Executing when detecting the rising edge
X_StartPos	Initial setting position of logical axis X	LREAL	Positive	Required field	Initial setting position of logical axis X
Y_StartPos	Initial setting position of logical axis Y	LREAL	Positive	Required field	Initial setting position of logical axis Y
Z_StartPos	Initial setting position of logical axis Z	LREAL	Positive	Required field	Initial setting position of logical axis Z
A_StartPos	Initial setting position of logical axis A	LREAL	Positive	Required field	Initial setting position of logical axis A
B_StartPos	Initial setting position of logical axis B	LREAL	Positive	Required field	Initial setting position of logical axis B
C_StartPos	Initial setting position of logical axis C	LREAL	Positive	Required field	Initial setting position of logical axis C
P_StartPos	Initial setting position of logical axis P	LREAL	Positive	Required field	Initial setting position of logical axis P
Q_StartPos	Initial setting position of logical axis Q	LREAL	Positive	Required field	Initial setting position of logical axis Q

■ Output variable

Name	Meaning	Data type	Valid range	Function
Done	Done	BOOL	TRUE OR FALSE	Change to TRUE after execution of the instruction is complete
Busy	Executing	BOOL	TRUE OR FALSE	Change to True when instruction is executing.
Error	Error	BOOL	TRUE OR FALSE	Change to True when error occurs.

ErrorID	Error code	WORD	0~65535	Output error code when instruction execution error occurs Please refer to the 'Instruction error code description'

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After instruction execution is completed	When Done is TRUE, and Execute changes from TRUE to FALSE When the instruction is executed and the Execute is FALSE, Done becomes True and after one cycle, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	Done becomes TRUE Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

- This instruction is used to set the commanded position of each axis in the axis group to the absolute position set by the instruction. After this instruction is executed, both the commanded position and the feedback position of the axis change to the set position. The execution of this instruction does not cause the axes to move, nor does it leave the physical position of the specified axes unchanged, and is generally used to redefine the position coordinates or coordinate offsets. This instruction is realized by the internal operation of the controller, and will not change the position of the driver when operating the servo axis.
- For example, if the current position of X-axis and Y-axis in the axis group is set to 2000 by this instruction, and the CNC code is G90 G1 X3000 Y5000, then after executing the G code, the X-axis will move from the 2000 position to the 3000 position with a distance of 1000, and the Y-axis will move from the 2000 position to the 5000 position with a distance of 3000.
- When the MC_CoorMotion instruction is used to execute the CNC code, the instruction will set the feedback position of each axis in the axis group to the starting position of each axis in the axis group, so it is generally not necessary to execute this instruction. This instruction is not normally required because the starting position of each axis in the axis group is set to the user-specified position.

● Instruction completion timing

The output variable Done of this instruction is true, indicating that the parameter setting is complete.

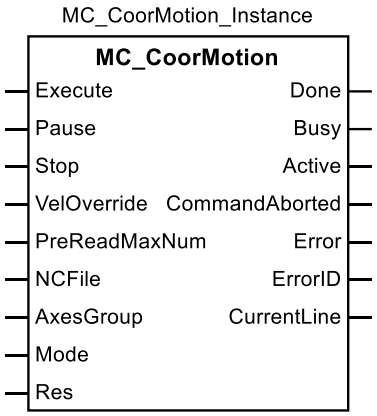
● Abnormality elimination

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become True, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

8.6 MC_CoorMotion(CNC execution)

This instruction is used to execute the CNC code in the controller. Library：MotionControl_Part2

Applicable models: M500 series（M500s series not included）

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_CoorMotion	CNC execution instruction	FB		MC_CoorMotion _Instance (Execute:=parameter, Pause:=parameter, Stop:=parameter, VelOverride:=parameter, PreReadMaxNum:=parameter, NCFile:=parameter , AxesGroup:=parameter, Mode:=parameter, Res:=parameter, Done=> parameter, Busy=> parameter, Active=> parameter, CommandAborted=> parameter , Error=> parameter, ErrorID=> parameter, CurrentLine=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	1~8	Required field	Axis group number
Execute	Execute	BOOL	TRUE OR FALSE	FALSE	Executing when detecting the rising edge
Pause	Pause	BOOL	TRUE OR FALSE	FALSE	Pause the motion of all axes in the axis group and decelerate and stop according to the deceleration of the command executed in the CNC.
Stop	Stop	BOOL	TRUE OR FALSE	FALSE	Stop all axes in axis group in one cycle
VelOverride	Velocity override	LREAL	0~500	0	Velocity override
PreReadMaxNum	Number of pre-read CNC commands	UINT	1~5	Required field	Number of pre-read CNC commands
NCFile	CNC number	UINT	1~64	Required field	CNC number
AxesGroup	Axis group number	USINT	1~8	Required field	Axis group number
Mode	Mode	INT	Reserve	Reserve	Reserve
Res	Reserve	LREAL	Reserve	Reserve	Reserve

■ Output variable

Name	Meaning	Data type	Valid range	Function
Done	Done	BOOL	TRUE OR FALSE	Change to TRUE after execution of the instruction is complete
Busy	Executing	BOOL	TRUE OR FALSE	Change to True when instruction is executing.
Active	Under control	BOOL	TRUE OR FALSE	Change to TRUE when the axis is under control
CommandAborted	Abortion	BOOL	TRUE OR FALSE	Change to TRUE when the instruction is aborted
Error	Error	BOOL	TRUE OR FALSE	Change to True when error occurs.

ErrorID	Error code	WORD	0~65535	Output error code when instruction execution error occurs Please refer to the 'Instruction error code description'
CurrentLine	Currently executing lines	UDINT	0~65535	Currently executing lines

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After instruction execution is completed	When Done is TRUE, and Execute changes from TRUE to FALSE When the instruction is executed and the Execute is FALSE, Done becomes True and after one cycle, it becomes FALSE
Busy	When Execute changes from FALSE to TRUE	Done becomes TRUE Error becomes TRUE CommandAborted becomes TRUE
Active	The axis is under control	Done becomes TRUE from FALSE Error becomes TRUE from FALSE CommandAborted becomes TRUE from FALSE
CommandAborted	When the instruction is aborted by other instructions	When CommandAborted is TRUE, Enable will become TRUE from FALSE Instruction has been executed and Execute becomes FALSE, when an exception is encountered during the execution of the instruction, Error becomes TRUE, and after one cycle, it becomes FALSE.
Error	The input variable value of the instruction is not within the valid range, does not meet the execution conditions of the instruction, or encounters an exception during the instruction execution process	When Execute changes from FALSE to TRUE

■ Function description

● Basic function description

- This instruction is used to execute the CNC code downloaded to the controller from the software. Before executing this instruction, the state of each axis in the axis group must be in StandStill state, otherwise the instruction will report an error.
- Before executing this instruction, please add each axis to the axis group by MC_ AddAxisToGroup instruction, and then execute this instruction. Which axes are a group is determined by the value of AxesGroup in the MC_ AddAxisToGroup instruction. Multiple axis groups can be executed at the same time, such as forming 1, 2, and 3 axes into one axis group, and 4, 5, and 6 axes into one axis group, which is accomplished by using two MC_CoorMotion instructions, with different values of AxesGroup specified in the two instructions.
- The speed and acceleration of the CNC code can be specified in the CNC code or through the MC_SetMoveDirectParm and MC_SetMovelinearParm instructions.
- The NCFile input variable is used to specify the CNC number to be executed, which can be viewed through the CNC number created in the software.
- The Pause input variable is used to pause the execution of the CNC code. When the input variable is set to TRUE, the CNC code will decelerate and stop according to the deceleration rate set in the CNC code, and the executing CNC code will be paused; when the input variable is set to FALSE, the CNC code will continue to execute according to the speed and acceleration rate set in the code. When the

CNC code is paused, the status of each axis in the axis group remains unchanged.

- The Stop input variable is used to stop the execution of CNC code. When the input variable is TRUE, all axes in the axis group will stop immediately (the command speed becomes 0 in one cycle), the axis state enters into the state of standstill, and the output variable of the command, Done, changes to TRUE. If you need to interrupt the execution of the current CNC code, it is recommended to set the Pause introduction to TRUE and then set the Stop input variable to TRUE.
- The MC_CoorMotion instruction controls the target speed together with the target speed of other motion instructions. The target speed can be changed in real time by changing the value of the VelFactor (target speed factor) of this instruction, and the unit of VelFactor is 1%. The unit of VelFactor is 1%, such as "50" means "50%". The value of VelFactor parameter is in the range of 0~500, if the value of VelFactor is out of the range, the instruction will report an error.
- You can set the value of VelFactor to 0 to realize the pause function.
- The output variable CurrentLine of this instruction is used to display the number of executed lines of CNC code specified by AxesGroup.

- **Instruction completion timing**

This instruction is executed after Execute changes from FALSE to TRUE, and executes the CNC code specified by NCFile, and when the CNC code is executed, Done changes from FALSE to TRUE. During the execution of the CNC code, the CNC code stops when the input variable Stop changes from FALSE to TRUE, and the output variable When the input variable Stop changes from FALSE to TRUE during CNC code execution, CNC code execution stops and the output variable of the instruction changes from FALSE to TRUE, and the axis state enters the standstill state.

- **Re-execute the instruction**

When the execution of the instruction is completed and Execute changes from FALSE to TRUE again, the instruction can be re-executed; when the instruction is being executed and Execute changes from FALSE to TRUE again, there will be no effect on the execution of the instruction, and the instruction will still execute the instruction in accordance with the input variables that have not been executed.

- **Execute this instruction while other instructions are in processing**

The status of each axis in the axis group must be in the StandStill state before this instruction can be executed. If this instruction is executed when other motion instructions are executed, the instruction will report an error.

- **Execute other instructions while this instructions is in processing**

When this instruction is executed, the execution of the motion instruction related to the axis group will report an error, and the execution of other motion instructions not related to the axis group will interrupt the instruction, and the acceleration or deceleration mode of the specified axis will be determined by the input variable of the executed instruction; the other axes in the axis group will enter the state of standstill, and the axes' speeds will be reduced to 0 for one cycle.

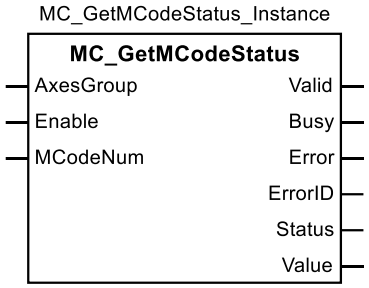
- **Abnormality elimination**

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become True, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

8.7 MC_GetMCodeStatus(Get M code status)

This instruction is used to get the M code status and the value. Library：MotionControl_Part2

Applicable models: M500 series（M500s series not included）

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_GetMCodeStatus	Get M code status	FB		MC_GetMCodeStatus_Instance (AxesGroup :=parameter, Enable:=parameter , MCodeNum:=parameter , Valid=> parameter , Busy => parameter, Error=> parameter, ErrorID=> parameter, Status=> parameter , Value=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	1~8	Required field	Axis group number
Enable	Enable	BOOL	TRUE OR FALSE	FALSE	Set to TRUE to read the status of M code Set to FALSE to stop reading M code status.
MCodeNum	Specified M code number	USINT	Positive	Required field	Specified M code number

■ Output variable

Name	Meaning	Data type	Valid range	Function
Valid	Valid	BOOL	TRUE OR FALSE	This instruction reads the status of the M code with this parameter value status TRUE
Busy	Executing	BOOL	TRUE OR FALSE	Change to True when instruction is executing.
Error	Error	BOOL	TRUE OR FALSE	Change to True when error occurs.
ErrorID	Error code	WORD	0~65535	Output error code when instruction execution error occurs Please refer to the ‘Instruction error code description’
Status	Status of M code	BOOL	TRUE OR FALSE	Status of M code
Value	Values corresponding to M code	LREAL	Positive	Values corresponding to M code

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Valid	When M code status can be read	Enable becomes FALSE. Error becomes TRUE.
Busy	Enable changes from FALSE to TRUE.	Done becomes FALSE. Error becomes TRUE.
Error	The input variable value of the instruction is not within the valid range	Enable changes from TRUE to FALSE.

Function description

Basic function description

- This instruction is used to read the status of the specified M code and the value corresponding to the M code. The number of the M code is specified by the input variable MCodeNum. When the input variable Enable of this instruction is TRUE, the values of the output variables Status and Value are updated once per cycle; when Enable is FALSE, the values of the output variables Status and Value stop being updated and the value at the last Enable of TRUE is kept unchanged.
- The status of the M code and the value corresponding to the M code change according to the execution of the M code in the CNC code when the CNC code is executed.
- When G code and M code are not on the same line:.

  N1 G1 X100.5 Y100.5

  N2 M10 D66.6

  N3 G1 X200.8 Y300.8

  After the CNC code executes the M code in line N2, the status of the M code numbered 10 is TRUE, the value corresponding to the M code numbered 10 is 66.6, the status of the output variable Status of this instruction is TRUE, and the value of Value is 66.6. When the G code and the M code are not in the same line, the CNC code suspends the execution when the execution of the line where the M code is located is finished, and the CNC code needs to reset the status of the M code through the MC_ResetMcode instruction resets the status of the M code before execution continues.
- When G code and M code on the same line:

  N1 G1 X100.5 Y100.5 M10 D66.6

  N2 G1 X200.8 Y300.8

  After the CNC code executes the G code and M code in line N1, the status of the M code numbered 10 is TRUE, and the value corresponding to the M code numbered 10 is 66.6, and the status of the output variable Status of the instruction is TRUE, and the value of Value is 66.6. When the G code and the M code are on the same line, the CNC code will continue to be executed in the following lines after the execution of the line in which the M code is located is completed. G-code and M-code are on the same line. The status of output variable Status can be reset by MC_ResetMcode instruction.

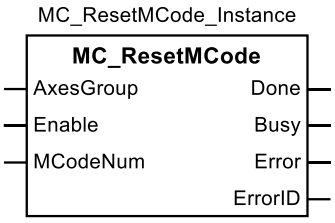
Abnormality elimination

When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become True, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

8.8 MC_ResetMCode (M code reset)

This instruction is used to reset the M code of the specified number. Library：MotionControl_Part2

Applicable models: M500 series（M500s series not included）

Instruction	Name	FB/FUN	Graphic expression	ST expression
MC_ResetMCode	M code reset	FB		MC_ResetMCode_Instance (AxesGroup:=parameter, Enable:=parameter, MCodeNum:=parameter, Done=> parameter, Busy=> parameter, Error=> parameter, ErrorID=> parameter);

■ Input variable

Name	Meaning	Data type	Valid range	Default	Description
AxesGroup	Axis group number	USINT	1~8	Required field	Axis group number
Execute	Execute	BOOL	TRUE OR FALSE	FALSE	Executing when detecting the rising edge
MCodeNum	The specified M code number.	USINT	Positive number	Required field	The specified M code number.

■ Output variable

Name	Meaning	Data type	Valid range	Function
Done	Done	BOOL	TRUE OR FALSE	Change to TRUE after execution of the instruction is complete
Busy	Executing	BOOL	TRUE OR FALSE	Change to True when instruction is executing.
Error	Error	BOOL	TRUE OR FALSE	Change to True when error occurs.
ErrorID	Error code	WORD	0~65535	Output error code when instruction execution error occurs Please refer to the ‘Instruction error code description’

■ Output variable refreshing timing

Name	Whether or not to become TRUE	Whether or not to become FALSE
Done	After execution of the instruction is complete	When Done is TRUE, and Execute changes from TRUE to FALSE When the instruction is executed and the Execute is FALSE, Done becomes True and after one cycle, it becomes FALSE
Busy	Enable changes from FALSE to TRUE.	Done becomes TRUE Error becomes TRUE
Error	The input variable value of the instruction is not within the valid range	Enable changes from TRUE to FALSE.

■ Function description

● Basic Function description

➢ This instruction is used to reset the M code of the specified number. the status of the M code becomes TRUE after the execution of the M code, and you can reset the status of the M code to FALSE by this instruction.

- When the G code and the M code are not on the same line, it is written as follows

  N1 G1 X100.5 Y100.5

  N2 M10 D66.6

  N3 G1 X200.8 Y300.8

  After the CNC code finishes executing the M code in line N2, the status of the M code numbered 10 is TRUE. When the G code and the M code are not in the same line, the CNC code suspends execution after the execution of the line where the M code is located is completed, and it is necessary to reset the status of the M code by this instruction before continuing execution.

- When the G code and the M code are on the same line, it is written as follows

  N1 G1 X100.5 Y100.5 M10 D66.6

  N2 G1 X200.8 Y300.8

  After the CNC code executes the G code and M code in line N1, the state of the M code numbered 10 is TRUE, and the state of the M code can be reset by this instruction. when the G code and the M code are in the same line, the CNC code will continue to be executed in the following lines after the execution of the line where the M code is located is completed.

- **Abnormality elimination**

  When the instruction is executed, if the input variable value of the instruction is not within the valid range, the instruction Error will become True, and there is a corresponding error code in the ErrorID, which can be used to determine the cause of the error.

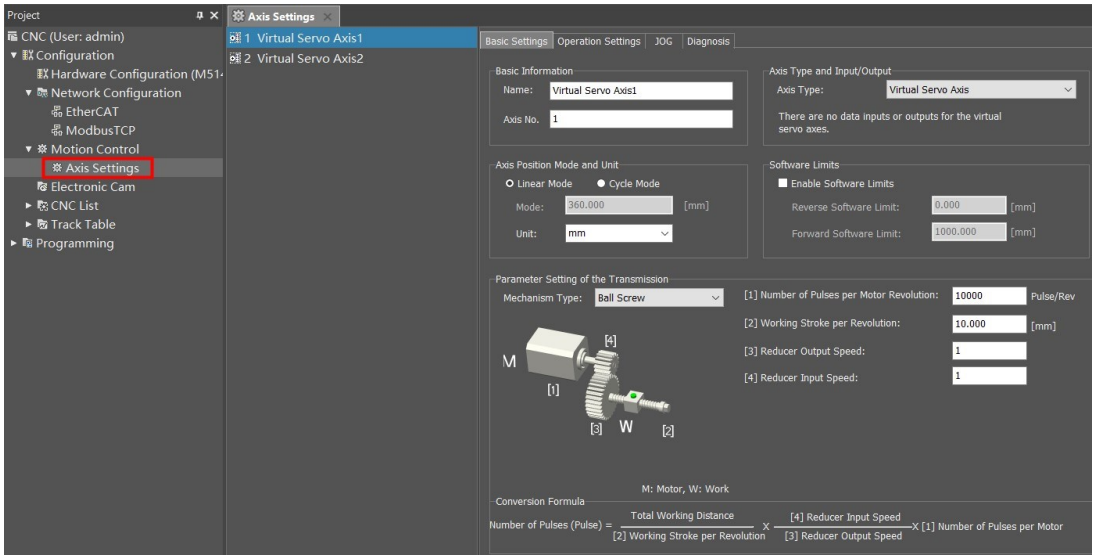
8.9 CNC code calling and executing sample program

■ Steps required to call CNC code

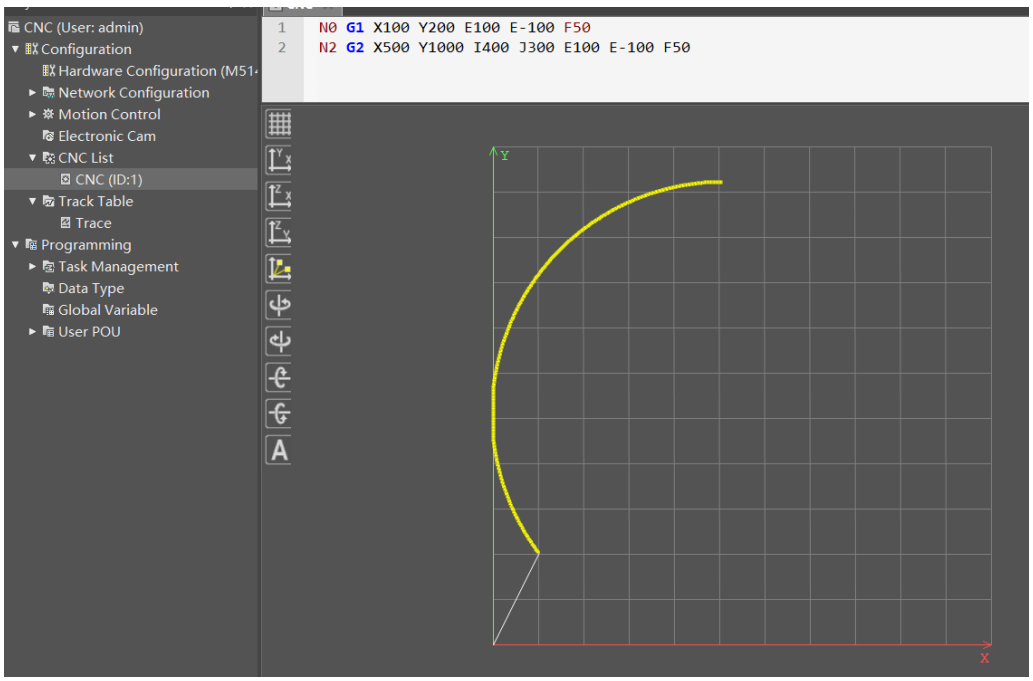
Step	Step description	Note
Step 1	Setting the axis parameters for each axis	Set in the software
Step 2	Axis enable	The servo axis executes the MC_Power command, the virtual servo axis does not need to do it.
Step 3	Add axis to the axis group	Correspondence between the logical axes in the axis group and the configured axes in the software.
Step 4	Execute CNC Execution Instruction Call CNC Code Execution	Execute the CNC code

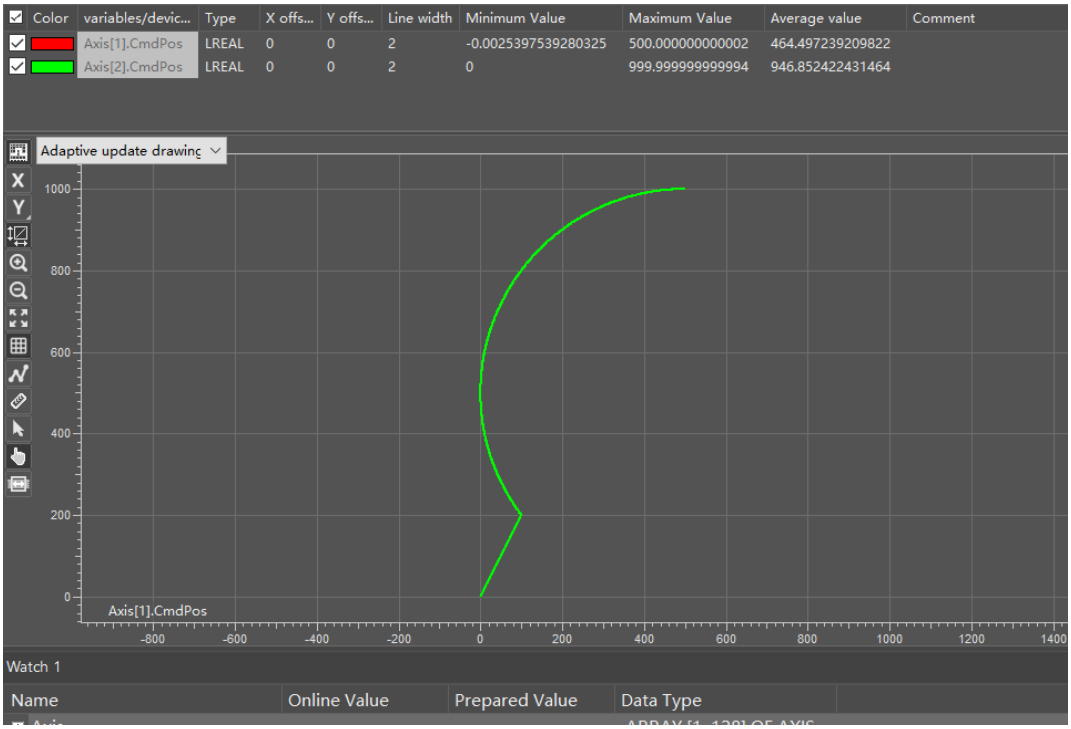
■ Axis parameters setting

Axis 1 and Axis 2 axis parameter settings are the same as shown in the following figure.



■ NC Code Trajectory Graph

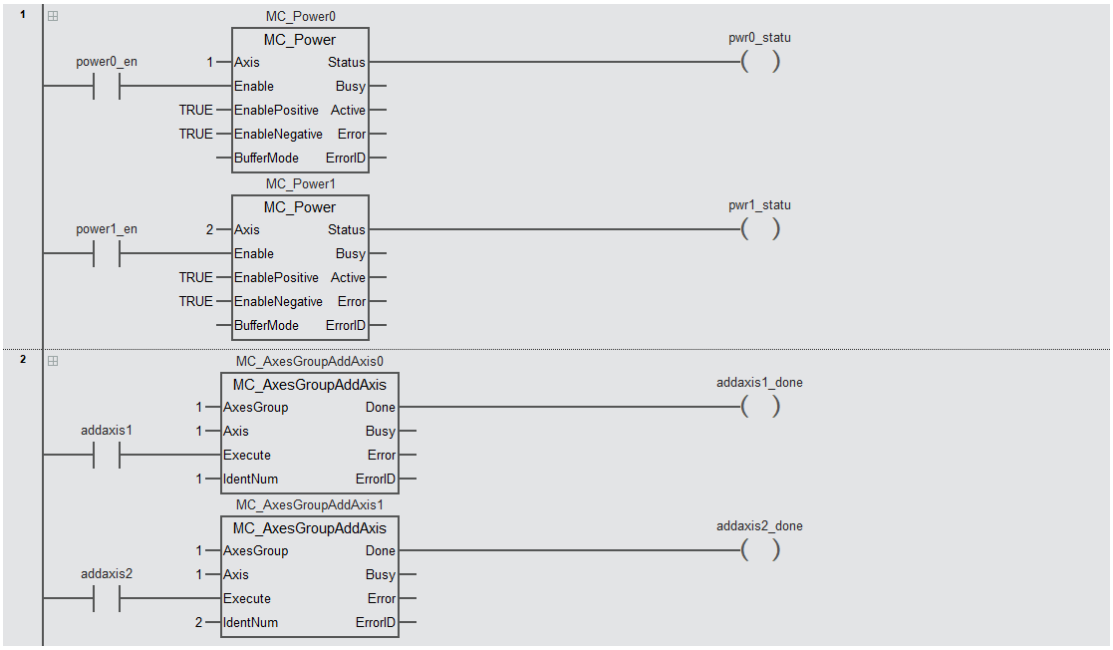


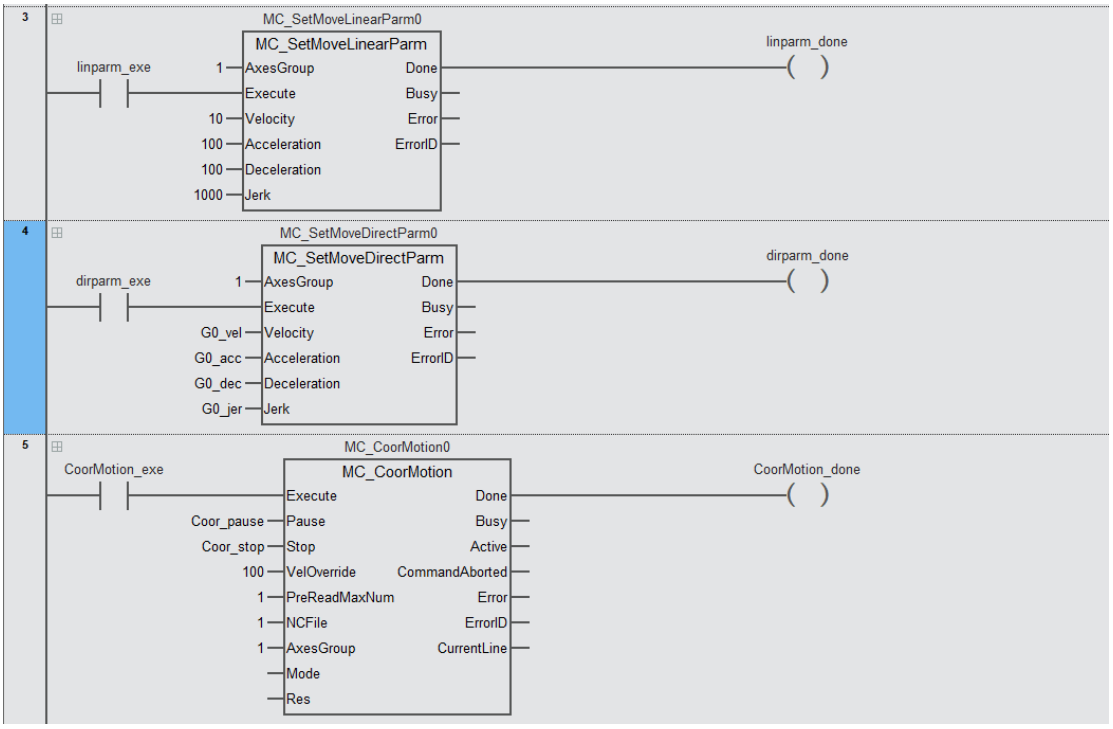


■ Variable list

Type	Name	Allocated to	Data type	Default	Note
VAR	MC_Power0		MC_Power		
VAR	MC_Power1		MC_Power		
VAR	power0_en		BOOL		
VAR	power1_en		BOOL		
VAR	pwr0_statu		BOOL		
VAR	pwr1_statu		BOOL		
VAR	MC_AxesGroupAddAxis0		MC_AxesGroupAddAxis		
VAR	MC_AxesGroupAddAxis1		MC_AxesGroupAddAxis		
VAR	addaxis1		BOOL		
VAR	addaxis2		BOOL		
VAR	addaxis1_done		BOOL		
VAR	addaxis2_done		BOOL		
VAR	MC_SetMoveLinearParm0		MC_SetMoveLinearParm		
VAR	linparm_exe		BOOL		
VAR	linparm_done		BOOL		
VAR	MC_SetMoveDirectParm0		MC_SetMoveDirectParm		
VAR	dirparm_exe		BOOL		
VAR	G0_vel		ARRAY [1..8] OF LREAL	[8(50)]	
VAR	G0_acc		ARRAY [1..8] OF LREAL	[8(100)]	
VAR	G0_dec		ARRAY [1..8] OF LREAL	[8(100)]	
VAR	G0_jer		ARRAY [1..8] OF LREAL	[8(1000)]	
VAR	dirparm_done		BOOL		
VAR	MC_CoorMotion0		MC_CoorMotion		
VAR	CoorMotion_exe		BOOL		
VAR	Coor_pause		BOOL		
VAR	Coor_stop		BOOL		
VAR	CoorMotion_done		BOOL		

■ LD


1
MC_Power0
MC_Power
power0_en
1 Axis
Status
Enable
Busy
TRUE EnablePositive
Active
TRUE EnableNegative
Error
BufferMode
ErrorID
pwr0_statu
MC_Power1
MC_Power
power1_en
2 Axis
Status
Enable
Busy
TRUE EnablePositive
Active
TRUE EnableNegative
Error
BufferMode
ErrorID
pwr1_statu
2
MC_AxesGroupAddAxis0
MC_AxesGroupAddAxis
1 AxesGroup
Done
addaxis1
1 Axis
Busy
Execute
Error
1 IdentNum
ErrorID
addaxis1_done
MC_AxesGroupAddAxis1
MC_AxesGroupAddAxis
1 AxesGroup
Done
addaxis2
1 Axis
Busy
Execute
Error
2 IdentNum
ErrorID
addaxis2_done


3
MC_SetMoveLinearParm0
MC_SetMoveLinearParm
linparm_exe
1 AxesGroup
Done
Execute
Busy
10 Velocity
Error
100 Acceleration
ErrorID
100 Deceleration
1000 Jerk
linparm_done
4
MC_SetMoveDirectParm0
MC_SetMoveDirectParm
dirparm_exe
1 AxesGroup
Done
Execute
Busy
G0_vel Velocity
Error
G0_acc Acceleration
ErrorID
G0_dec Deceleration
G0_jer Jerk
dirparm_done
5
MC_CoorMotion0
MC_CoorMotion
CoorMotion_exe
Execute
Done
Coor_pause Pause
Busy
Coor_stop Stop
Active
100 VelOverride
CommandAborted
1 PreReadMaxNum
Error
1 NCFile
ErrorID
1 AxesGroup
CurrentLine
Mode
Res
CoorMotion_done

ST

```
MC_Power0(
        Axis:=1 ,
        Enable:= power0_en,
        EnablePositive:= TRUE,
        EnableNegative:= TRUE,
        BufferMode:=0 ,
        Status=> pwr0_statu
        );

MC_Power1(
        Axis:=2 ,
        Enable:= power1_en,
        EnablePositive:= TRUE,
        EnableNegative:= TRUE,
        BufferMode:=0 ,
        Status=>pwr0_statu
        );

 MC_AxesGroupAddAxis0(
        AxesGroup:=1 ,
        Axis:=1 ,
        Execute:= addaxis1 ,
        IdentNum:=1 ,
        Done=> addaxis1_done
        );

 MC_AxesGroupAddAxis1(
        AxesGroup:=1 ,
        Axis:=2 ,
        Execute:= addaxis2,
        IdentNum:=2 ,
        Done=> addaxis2_done
        );

MC_SetMovelinearParm0(
      AxesGroup:=1 ,
      Execute:= linparm_exe ,
      Velocity:= 10,
      Acceleration:=100 ,
      Deceleration:=100 ,
      erk:=1000 ,
      Done=> linparm_done
      );

MC_SetMoveDirectParm0(
      AxesGroup:=1 ,
      Execute:= dirparm_exe ,
      Velocity:= G0_vel,
      Acceleration:=G0_acc ,
      Deceleration:=G0_dec ,
      Jerk:=G0_jer,
      Done=> dirparm_exe
```

```
);

MC_CoorMotion0(Execute:=CoorMotion_exe ,
  Pause:=Coor_pause ,
  Stop:=Coor_stop ,
  VelOverride:=100 ,
  PreReadMaxNum:=1 ,
  NCFile:=1 ,
  AxesGroup:=1 ,
  Done=>CoorMotion_done
);
```

Appendix

Appendix 1 Homing Mode

According to the home switch signal、limit switch signal and encoder Z signal, the CiA402 protocol defines 31 homing modes. Set 6060H to 6 to enable homing mode, which is applicable to EtherCAT Drive Drives.

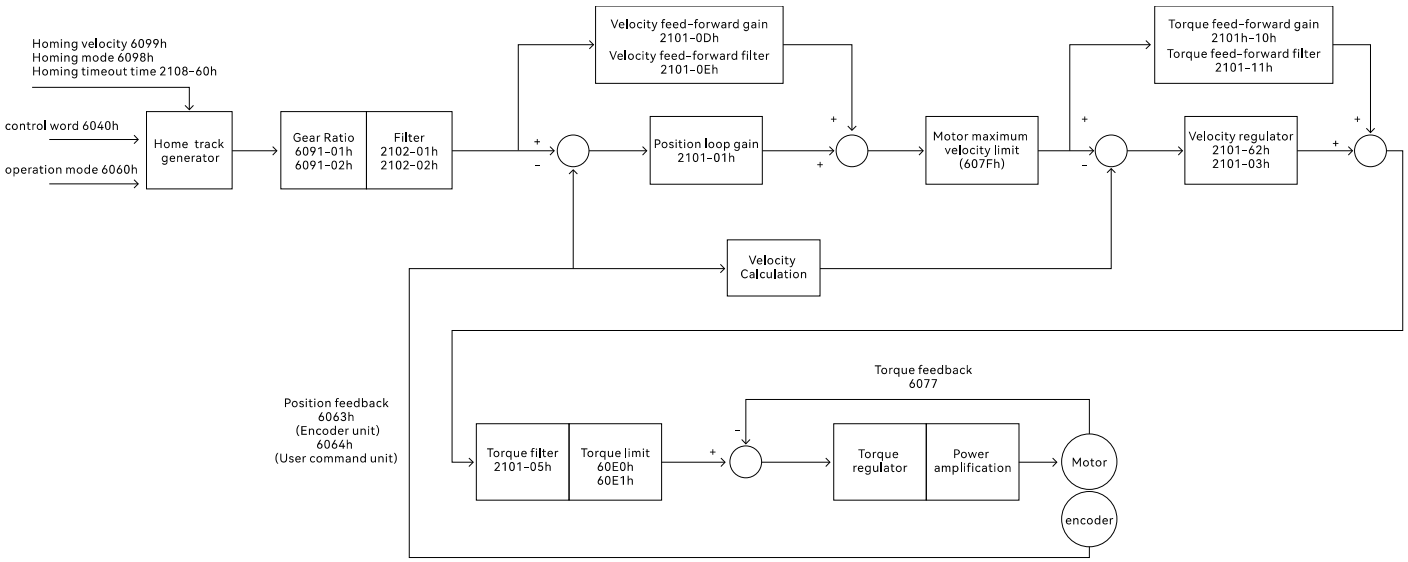


Figure Appendix 1-1 Homing Mode Control Block Diagram

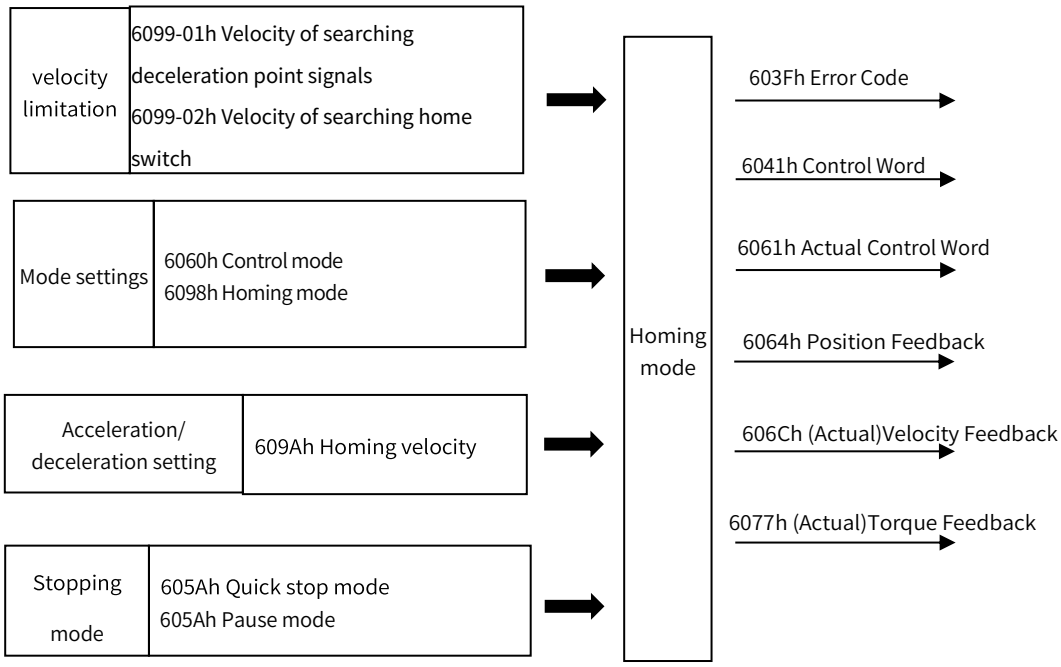


Figure Appendix 1-2 Input and Output in Homing Mode

Appendix 1.1 Control Word Settings in Homing Mode(60400010h)

When the homing mode is selected, the significance of each bit of the control word (6040h) is shown in Appendix 1-1 of the table, where the specific control commands for the homing mode are marked with a darker color.

Table Appendix 1-1 Control Word Description in Homing Mode

Bit	Name	Description
0	Switch on	When the Drive is enabled, it must be set to 1.
1	Enable voltage	When the Drive is enabled, it must be set to 1.
2	Quick stop	When the Drive is enabled, it must be set to 1, otherwise the Drive will rapidly shutdown.
3	Operation enable	When the Drive is enabled, it must be set to 1.
4	Enable Home	0: Invalid, 1: Valid. Valid to start homing process, during the homing process, it must be Mastertained as valid, switching to invalid will stop the homing process.
5、6	Homing mode reserved	reserved
7	Fault reset	Performs a fault reset on a 0→1 change, if multiple resets are required, then multiple 0→1 changes are required. When this bit is set to 1, other control commands are invalid
8	Pause	0: Invalid,1: valid. Slow down and stop the homing process when valid.
9	Home mode reserved	reserved
10	Reserve	reserved
11~15	Factory customized	reserved

Appendix1.2 Status Word Definition in Homing Mode（60410010h）

When the homing mode is selected, the significance of each bit of the control word (6041h) is shown in Appendix 1-2 of the table, where the specific control commands for the homing mode are marked with a darker color.

Table Appendix 1-2 Status Word Description in Homing Mode

Bit	Name	Description
0	Ready to switch on	0: Invalid,1: Valid. The Drive can be enabled when valid
1	Switched on	0: Invalid,1: Valid. The Drive can be enabled when valid
2	Operation enabled	0: Invalid, 1: Valid. The Drive is enabled when valid
3	Drive malfunction	0: No fault,1: Faulty
4	Voltage enabled	0: Invalid,1: Valid. The Drive can be enabled when valid
5	Quick stop	0: Quick stop is valid, 1: Quick stop is invalid
6	Switch on disabled	0: Invalid,1: Valid. The Drive can be enabled when valid
7	Warning	0: No warning,1: Warning
8	Factory customized	Reserved
9	Remote control	0: Invalid, 1: Valid. The control word has taken effect when valid

10	Position reached	60400010h bit 8 (pause)=0， 0: Position is not reached, 1: position is reached; 60400010h bit 8 (pause)=1， 0: Decelerating, 1: The velocity is 0.
11	Internal soft limit status	0: The soft limit is not reached, 1: The soft limit is reached
12	Homing completed output	0: Homing is not completed,1:Homing is completed
13	Home error	0：No error,1: An error occurred when homing
14	Factory customized	Reserved
15	Homing completed	0: Invalid,1: Homing is completed For absolute system: P06.47 is set to 2, and the 2nd bit Slave the right of the hexadecimal value of P09.14 is set to 1, the value of bit15 will be stored after successful restore (persistent). The value of bit15 will be stored after successful home (persistent), and the stored value can be cleared by setting P20.06 to 7.

Appendix 1.3 Parameters related to Homing Mode

Table Appendix 1-3, lists the object dictionary involved in the homing mode

Table Appendix 1-3 Dictionary of objects related to the home model

Index	Sub-Index	Name	Access type	Data type	Default value
603Fh		Error code	ro	unsigned16	0
6040h		Control word	rw	unsigned16	0
6041h		Status word	ro	unsigned16	0
6060h		Control mode	rw	integer8	0
6061h		Control mode display	ro	integer8	0
6062h		User command position	ro	integer32	0
6063h		Motor position feedback	ro	integer32	0
6064h		User position feedback	ro	integer32	0
6065h		User position difference too large threshold	rw	unsigned32	1000000000
6067h		Position reaching threshold	rw	unsigned32	100
6068h		Position reaching time	rw	unsigned16	1
606Bh		User velocity command value	ro	integer32	0
606Ch		User actual velocity feedback	ro	integer32	0
607Ch		Home bias	rw	integer32	0
607Dh	01h	Soft limit: minimum position limit	rw	integer32	-2147483648
	02h	Soft limit: maximum position limit	rw	integer32	2147483647
6098h		Homing mode	rw	integer8	0
6099h	01h	Velocity of searching for deceleration signal in homing mode	rw	unsigned32	218453
	02h	Velocity of searching for switch signal in homing mode	rw	unsigned32	21845
609Ah		Homing acceleration	rw	unsigned32	1310720

Appendix1.4 Simple Tutorial of Homing Mode (taking X3E Drive as an example)

1. Set the X3E drive parameters and configure the home related DI parameters (Group 4 parameters: digital input and output, P6.28=0)

Appendix 1.4 Parameter Configuration of Drive when operating Homing Mode.

Parameter address	Set value	Description
P00.01 (2100-02h)	7	CANopen/EtherCAT mode
P09.00 (2109-01h)	1	CANopen slave address (default 1)
P09.18 (2109-12h)	1	EtherCAT slave address (default 1)
P09.13 (2109-0Eh)	5	Baud rate (default 500K, EtherCAT does not need to be set, fixed 100M)

2. The controller connects to the drive and sets the CANopen communication parameters, configuration PDO parameters, etc. EtherCAT does not need to be set.

Table Appendix 1-5 homing mode startup and operation process

Address	Name	Value setting(decimal value)
60600008h	Control mode	6
60980008h	Homing mode	1~35
60400010h Control word	Alarm reset	Arbitrary number→ 128 (rising edge valid, if resettable)
	Homing	6 → 7 → 15 → 31 (homing triggered by bit4 rising edge)
60990120h	Velocity of searching for deceleration signal in homing mode	Default value:218453 (Instruction unit/s)
60990220h	Velocity of searching for switch signal in homing mode	Default value:21845 (Instruction unit/s)
609A0020h	Homing acceleration	Default value:1310720 (Instruction unit/s^2)

Appendix1.5 Introduction of Homing Mode

As described in Table Appendix 1-6, CiA402 defines internally 31 homing modes (applicable to CANopen/EtherCAT).

In the following description, HSW indicates the home position sensor signal, NL indicates the negative limit signal, and PL indicates the positive limit signal. ON indicates the valid state of the signal, and OFF indicates the invalid state of the signal. OFF→ON indicates the jumping edge of the signal Slave the invalid state to the valid state, and ON→OFF indicates the jumping edge of the signal Slave the valid state to the invalid state. The following are respectively introduced various homing mode operation track and signal state change, various home mode icon and the icon meaning is shown in Figure Appendix1-3.

Table Appendix 1-6 Homing mode startup and operation process

Homing mode	Description
0	None
1	Run in the negative direction when starting. when encountering the OFF→ON status of HL during negative operation, change to a low velocity, and then retreat to find the nearest Z-pulse position as the origin.
2	Run in the positive direction when starting. when encountering the OFF→ON status of HL during positive operation, change to a low velocity, and then retreat to find the nearest Z-pulse position as the origin.
3	If the HSW is invalid when starting, run in the positive direction, otherwise run it in the negative direction. When running in the negative direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run negatively to find the nearest Z-pulse position as the origin.
4	If the HSW is invalid when starting, run in the positive direction, otherwise run it in the negative direction. When running in the positive direction, when encountering the OFF→ON status of HSW, change to a low velocity operation, and then continue to run positively to find the nearest Z-pulse position as the origin

5	If the HSW is invalid when starting, run in the negative direction, otherwise run it in the positive direction. When running in the positive direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run positively to find the nearest Z-pulse position as the origin
6	If the HSW is invalid when starting, run in the negative direction, otherwise run it in the positive direction. When running in the negative direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run negatively to find the nearest Z-pulse position as the origin
7	If the HSW is invalid when starting, run in the positive direction, otherwise run it in the negative direction. When running in the negative direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run negatively to find the nearest Z-pulse position as the origin
8	If the HSW is invalid when starting, run in the positive direction, otherwise run it in the negative direction. When running in the positive direction, when encountering the OFF→ON status of HSW, change to a low velocity operation, and then continue to run positively to find the nearest Z-pulse position as the origin
9	Regardless of whether the HSW is valid or not, starts it in a positive direction. When running in the negative direction, when encountering the OFF→ON status of HSW, change to a low velocity operation, and then continue to run negatively to find the nearest Z-pulse position as the origin
10	Regardless of whether the HSW is valid or not, starts it in a positive direction. When running in the positive direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run positively to find the nearest Z-pulse position as the origin
11	If the HSW is invalid when starting, run in the negative direction, otherwise run it in the positive direction. When running in the positive direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run positively to find the nearest Z-pulse position as the origin
12	If the HSW is invalid when starting, run in the positive direction, otherwise run it in the negative direction. When running in the negative direction, when encountering the OFF→ON status of HSW, change to a low velocity operation, and then continue to run negatively to find the nearest Z-pulse position as the origin
13	Regardless of whether the HSW is valid or not, starts it in a negative direction. When running in the positive direction, when encountering the OFF→ON status of HSW, change to a low velocity operation, and then continue to run positively to find the nearest Z-pulse position as the origin
14	Regardless of whether the HSW is valid or not, starts it in a negative direction. When running in the negative direction, when encountering the ON→OFF status of HSW, change to a low velocity operation, and then continue to run negatively to find the nearest Z-pulse position as the origin
15	Reserved
16	Reserved
17	Similar to Mode 1, but without looking for the Z-pulse, the OFF→ON status position where the negative runtime encounters NL as the origin
18	Similar to Mode 2, but without looking for the Z-pulse, the OFF→ON status position where the positive runtime encounters PL as the origin
19	Similar to Mode 3, but without looking for the Z-pulse, the ON→OFF status position where the negative runtime encounters HSW as the origin
20	Similar to Mode 4, but without looking for the Z-pulse, the OFF→ON status position where the positive runtime encounters HSW as the origin
21	Similar to Mode 5, but without looking for the Z-pulse, the ON→OFF status position where the positive runtime encounters HSW as the origin
22	Similar to Mode 6, but without looking for the Z-pulse, the OFF→ON status position where the negative runtime encounters HSW as the origin
23	Similar to Mode 7, but without looking for the Z-pulse, the ON→OFF status position where the negative runtime encounters HSW as the origin
24	Similar to Mode 8, but without looking for the Z-pulse, the OFF→ON status position where the positive runtime encounters HSW as the origin
25	Similar to Mode 9, but without looking for the Z-pulse, the OFF→ON status position where the negative runtime encounters HSW as the origin
26	Similar to Mode 10, but without looking for the Z-pulse, the ON→OFF status position where the positive runtime encounters HSW as the origin
27	Similar to Mode 11, but without looking for the Z-pulse, the ON→OFF status position where the positive runtime encounters HSW as the origin
28	Similar to Mode 12, but without looking for the Z-pulse, the OFF→ON status position where the negative runtime encounters HSW as the origin

29	Similar to Mode 13, but without looking for the Z-pulse, the OFF→ON status position where the positive runtime encounters HSW as the origin
30	Similar to Mode 14, but without looking for the Z-pulse, the ON→OFF status position where the negative runtime encounters HSW as the origin
31	Reserved
32	Reserved
33	Find the nearest Z pulse position and set it as the origin in negative direction when starting
34	Find the nearest Z pulse position and set it as the origin in positive direction when starting
35	Set current position as the origin

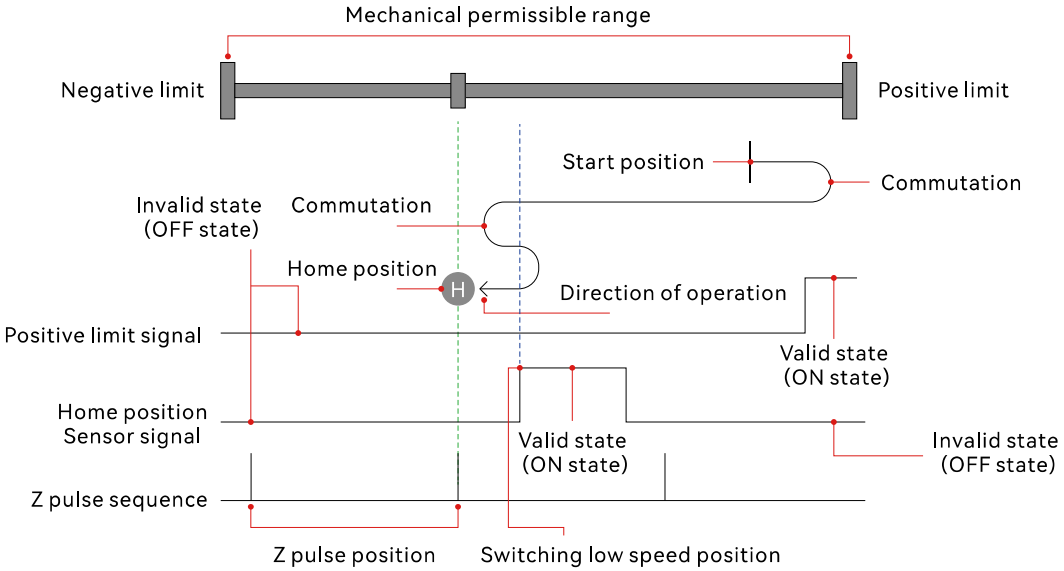


Figure Appendix 1-3 The significance of the various icons in the homing mode

In general, it is suggested to apply homing mode 3~6 and 19~22 to the OFF/ON status of HSW, which precisely divides the entire mechanical allowable travel range into two parts. Because in these 8 modes, whenever encounter NL or PL, it will stops and alarms and it does not automatically reverse the search for the origin

It is suggested to apply home 7~14 and 23~30 to the ON status of HSW, which precisely divides the entire mechanical allowable travel range into three parts. At this time, the ON status interval only occupies a small part of the entire mechanical allowable travel range (i.e., the ON status is a short-term transient).

The above are only suggestions and not mandatory requirements.

■ Mode 1, Find Negative Limit and Z Pulse

If the NL is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF → ON status of the NL, and then running in the positive direction at low velocity. After encountering the ON → OFF status of the NL, run in the positive direction at low velocity to find the nearest Z pulse position and set it as the origin.

If the NL is valid when staring, run in the positive direction at low velocity. After encountering the ON → OFF status of the NL, keep running to find the nearest Z pulse position and set it as the origin.

As shown in Appendix 1-4, see Table Appendix 1-6.

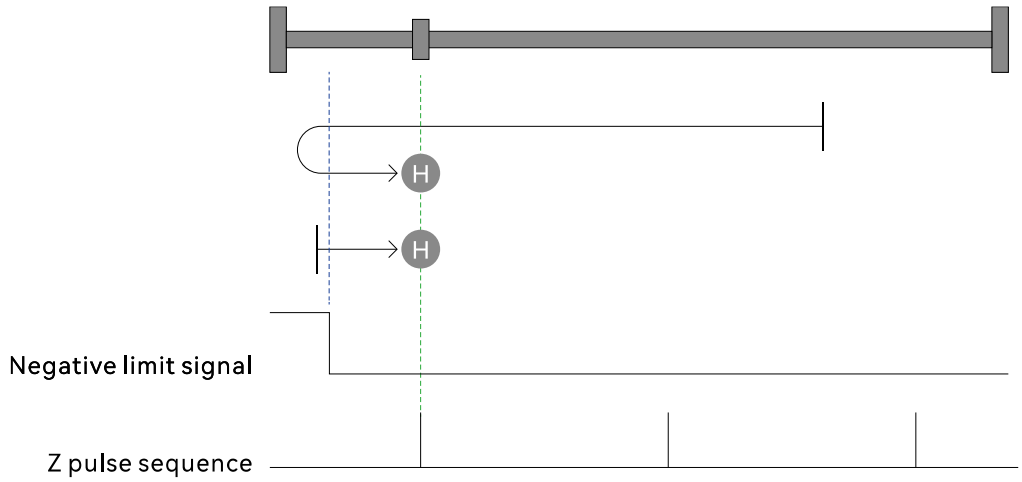


Figure Appendix 1-4 Homing Mode 1 Trajectory and Signal Status

■ Mode 2, Find the positive Limit and Z Pulse

If the PL is invalid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of the PL, and then running in negative direction at low velocity. After encountering the ON → OFF status of the PL, run in the negative direction at low velocity to find the nearest Z pulse position and set it as the origin.

If the PL is valid when staring, run in negative direction at low velocity. After encountering the ON → OFF status of the PL, keep running to find the nearest Z pulse position and set it as the origin.

As shown in Appendix 1-5, see Table Appendix 1-6.

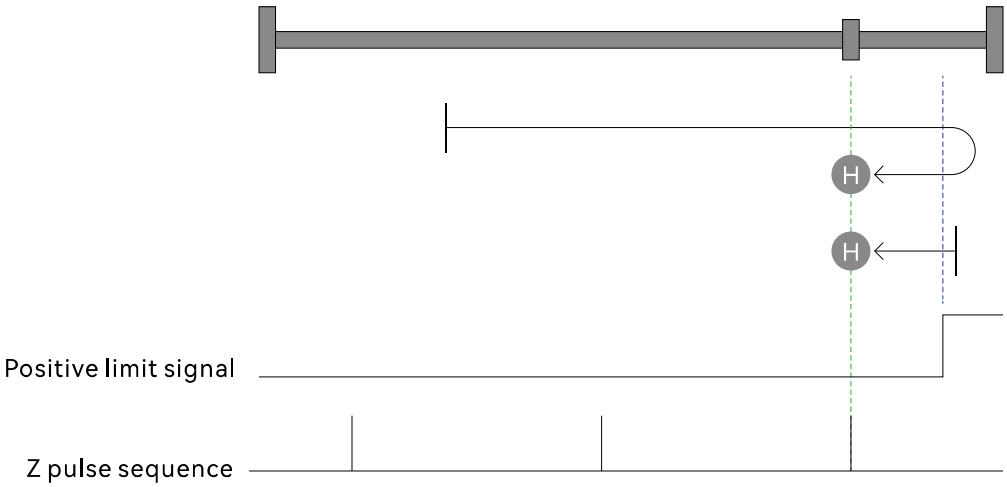


Figure Appendix 1-5 Homing Mode 2 trajectory and Signal Status

■ Mode 3, Find HSW ON→OFF position and Z pulse when running in negative direction

If the HSW is invalid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of the HSW, and then running in negative direction at low velocity. After encountering the ON → OFF status of the HSW, run in the negative direction at low velocity to find the nearest Z pulse position and set it as the origin.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the ON → OFF status of the HSW, then reverse back to the HSW valid position at high velocity and running in negative direction at low velocity after decelerate to stop. After encountering the ON → OFF status of the HSW, run in the negative direction at low velocity to find the nearest Z pulse position and set it as the origin.

In this homing mode, no matter encountering NL or PL at ON status, stop homing process and alarm.

As shown in Appendix 1-6, see Table Appendix 1-6.

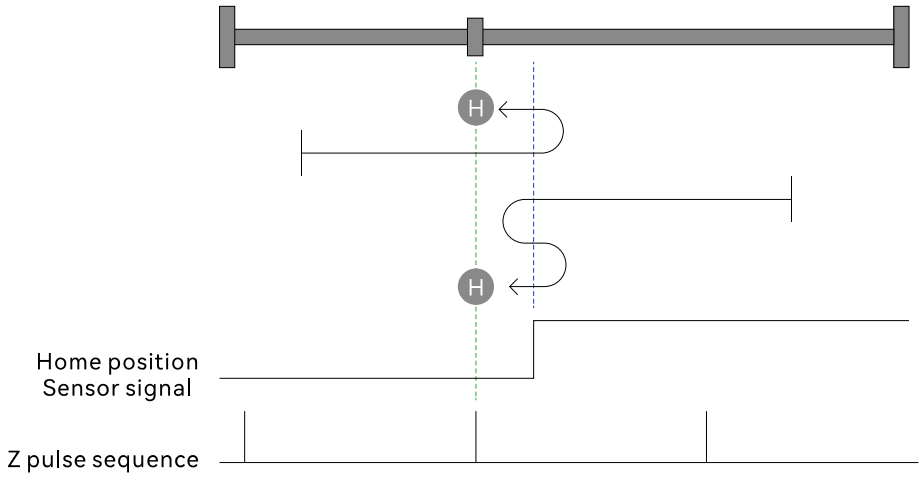


Figure Appendix 1-6 Homing Mode 3 Trajectory and Signal Status

■ Mode 4, Find HSW OFF→ON position and Z pulse when running in positive direction

If the HSW is invalid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of the HSW, then reverse back to the HSW invalid position at high velocity and running in the positive direction at low velocity after decelerate to stop. After encountering the OFF → ON status of the HSW, run in the positive direction at low velocity to find the nearest Z pulse position and set it as the origin.

If the HSW is valid when starting, run in the negative direction at high velocity. Decelerate to stop after encountering the ON → OFF status of the HSW, and then running in positive direction at low velocity. After encountering the OFF → ON status of the HSW, run in the positive direction at low velocity to find the nearest Z pulse position and set it as the origin.

In this homing mode, no matter encountering NL or PL at ON status, stop homing process and alarm.

As shown in Appendix 1-7, see Table Appendix 1-6.

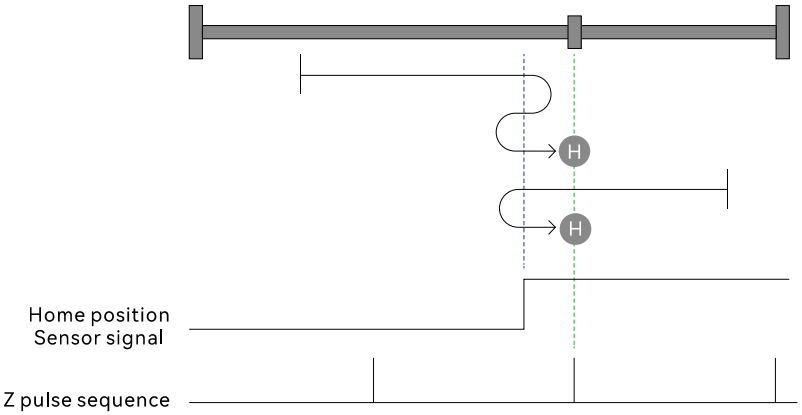


Figure Appendix 1-7 Homing Mode 4 Trajectory and Signal Status

■ Mode 5, Find HSW ON→OFF position and Z pulse when running in the positive direction.

If the HSW is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF → ON status of the HSW, and then running in the positive direction at low velocity. After encountering the ON → OFF status of the HSW, run in the positive direction at low velocity to find the nearest Z pulse position and set it as the origin.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the ON → OFF status of the HSW, then reverse back to the HSW valid position at high velocity and running in the positive direction at low velocity after decelerate to stop. After encountering the ON → OFF status of the HSW, run in the positive direction at low velocity to find the nearest Z pulse position and set it as the origin.

In this homing mode, no matter encountering NL or PL at ON status, stop homing process and alarm.

As shown in Appendix 1-8, see Table Appendix 1-6.

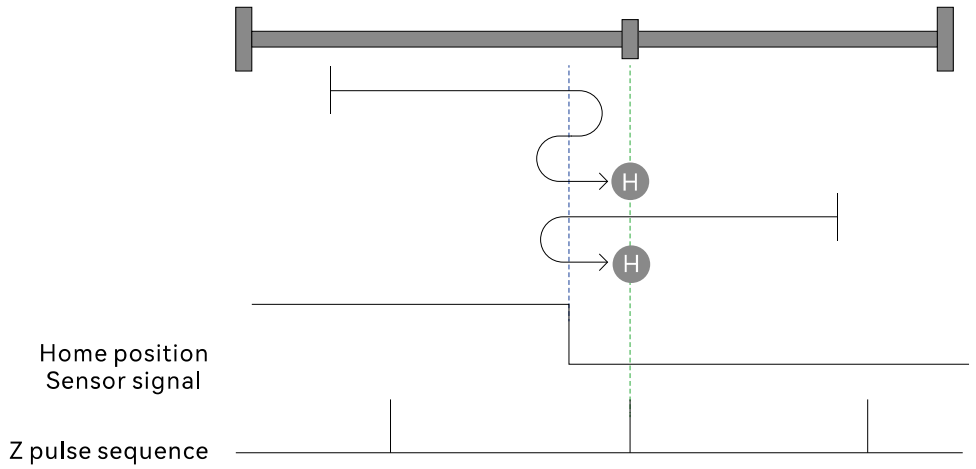


Figure Appendix 1-8 Homing Mode 5 Trajectory and Signal Status

■ Mode 6, Find HSW OFF → ON position and Z pulse when running in negative direction.

If the HSW is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF → ON status of the HSW, then reverse back to the HSW invalid position at high velocity and running in negative direction at low velocity after decelerate to stop. After encountering the OFF → ON status of the HSW, run in the negative direction at low velocity to find the nearest Z pulse position and set it as the origin.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the ON → OFF status of the HSW, and then running in negative direction at low velocity. After encountering the OFF → ON status of the HSW, run in the negative direction at low velocity to find the nearest Z pulse position and set it as the origin.

In this homing mode, no matter encountering NL or PL at ON status, stop homing process and alarm.

As shown in Appendix 1-9, see Table Appendix 1-6.

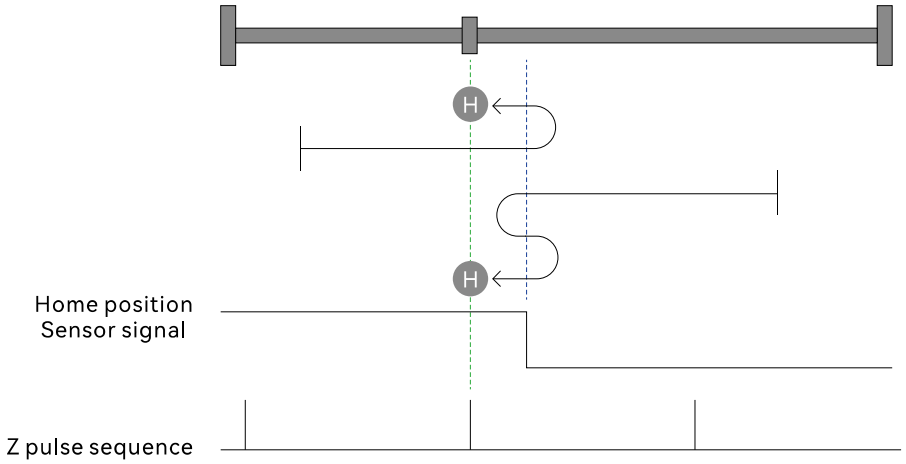


Figure Appendix 1-9 Homing Mode 6 Trajectory and Signal Status

■ Mode 7, Find HSW ON → OFF position and Z pulse when running in negative direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at a high velocity. Decelerate to stop after encountering the ON→OFF status of HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering OFF→ON status of HSW and running in negative direction at a low velocity, find the nearest Z pulse position and set it as the home after encountering the ON→OFF status of HSW.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to the HSW valid position at high velocity, and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for the second time, stop homing process and Alarm

As shown in Appendix 1-10, see Table Appendix 1-6.

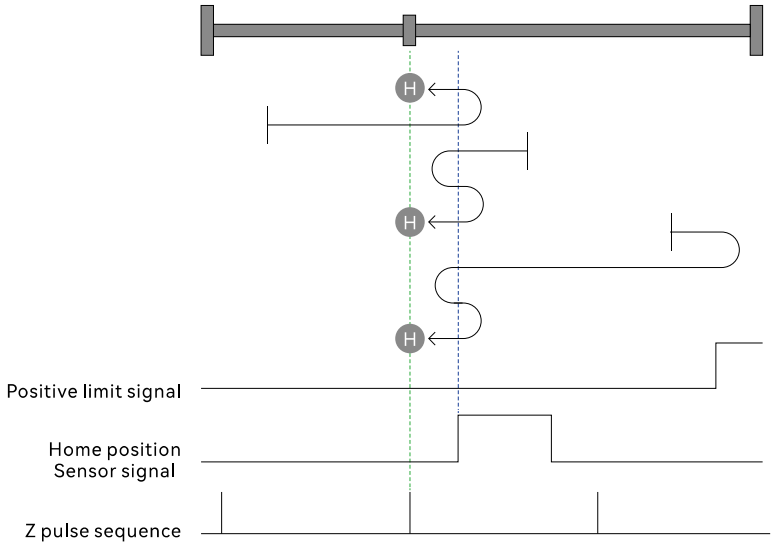


Figure Appendix 1-10 Homing Mode 7 Trajectory and Signal Status

■ Mode 8, Find HSW OFF → ON position and Z pulse when running in the positive direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at high velocity. Decelerate to stop after encountering the ON→OFF status of HSW and running the positive direction at low velocity, then find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering OFF→ON status of HSW and reverse back to the HSW invalid position at high velocity, and decelerate to stop, then running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in the positive direction at a low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for second time, stop homing process and alarm.

As shown in Appendix 1-11, see Table Appendix 1-6.

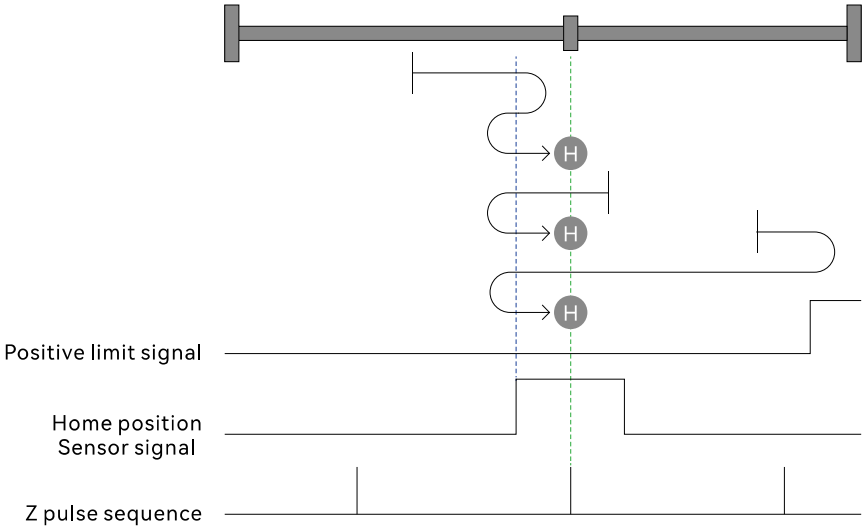


Figure Appendix 1-11 Homing Mode 8 Trajectory and Signal Status

■ Mode 9, Find HSW OFF → ON position and Z pulse when running in negative direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at a high velocity. Decelerate to stop after encountering the OFF→ON status of HSW and reverse back to the HSW invalid position at high velocity, and decelerate to stop, then running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in negative direction at a low velocity, find the nearest Z pulse position and set is as the home after encountering the OFF → ON status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for second time, stop homing process and Alarm

As shown in Appendix 1-12, see Table Appendix 1-6.

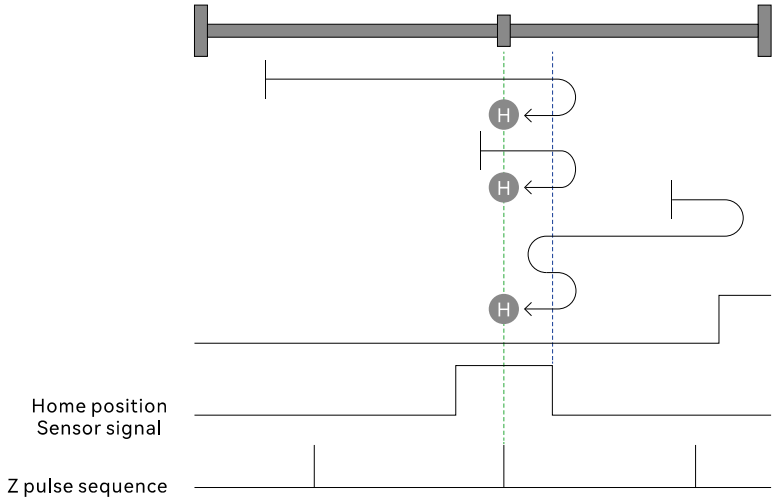


Figure Appendix 1-12 Homing Mode 9 Trajectory and Signal Status

■ Mode 10, Find HSW ON → OFF position and Z pulse when running in positive direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of HSW and running the positive direction at low velocity, then find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in positive direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop (If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area),then running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON→OFF status of HSW.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to the HSW valid position at high velocity, and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm

As shown in Appendix 1-13, see Table Appendix 1-6.

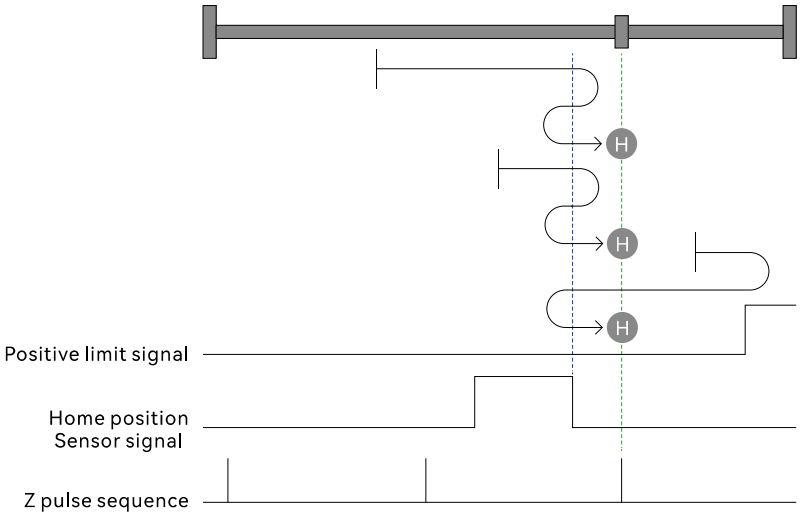


Figure Appendix 1-13 Homing Mode 10 Trajectory and Signal Status

■ Mode 11, Find HSW ON → OFF position and Z pulse when running in the positive direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of HSW and running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON status of NL and running in the positive direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop (If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area),then running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON→OFF status of HSW.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to the HSW valid position at high velocity, and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-14, see Table Appendix 1-6.

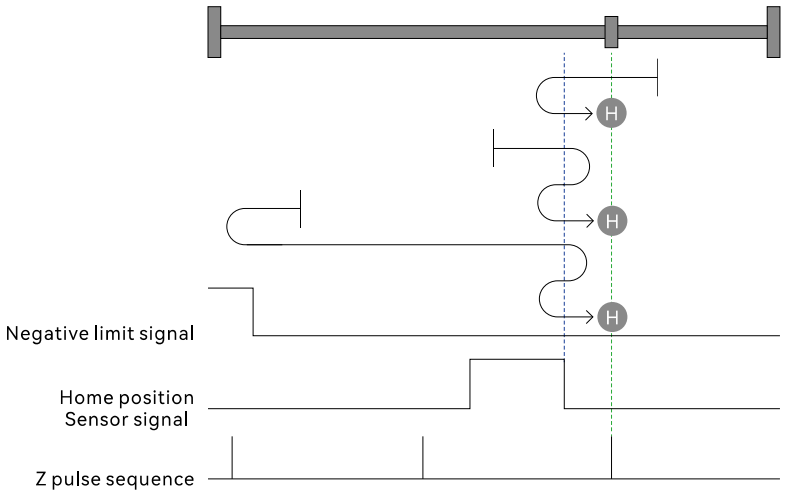


Figure Appendix 1-14 Homing Mode 11 Trajectory and Signal Status

■ Mode 12, Find HSW OFF → ON position and Z pulse when running in the positive direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop, then running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON status of NL and running in the positive direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF→ON status of HSW.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in the negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-15, see Table Appendix 1-6.

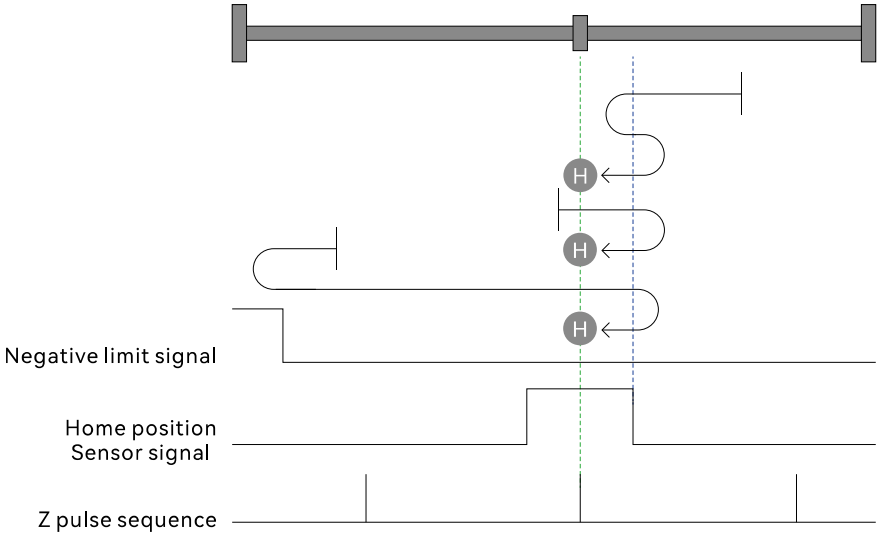


Figure Appendix 1-15 Homing Mode 12 Trajectory and Signal Status

■ Mode 13, Find HSW OFF → ON position and Z pulse when running in the positive direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering the ON→OFF status of HSW and running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON status of NL and running in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop, then running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF→ON status of HSW.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in the positive direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the OFF → ON status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-16, see Table Appendix 1-6.

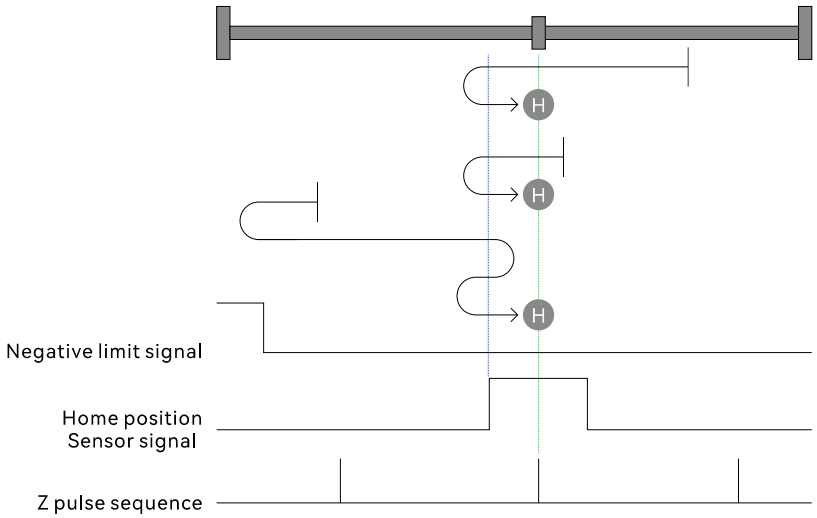


Figure Appendix 1-16 Homing Mode 13 Trajectory and Signal Status

- Mode 14, Find HSW NO → OFF position and Z pulse when running in the negative direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of the HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop (If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area),then running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON→OFF status of the HSW.

If the HSW is invalid and home at sensor's negative side when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON status of NL and running in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of HSW and running in negative direction at low velocity, find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to HSW valid position at high velocity and decelerate to stop, then running in negative position at low velocity ,find the nearest Z pulse position and set it as the home after encountering the ON → OFF status of HSW.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for second time, stop homing process and alarm.

As shown in Appendix 1-17, see Table Appendix 1-6.

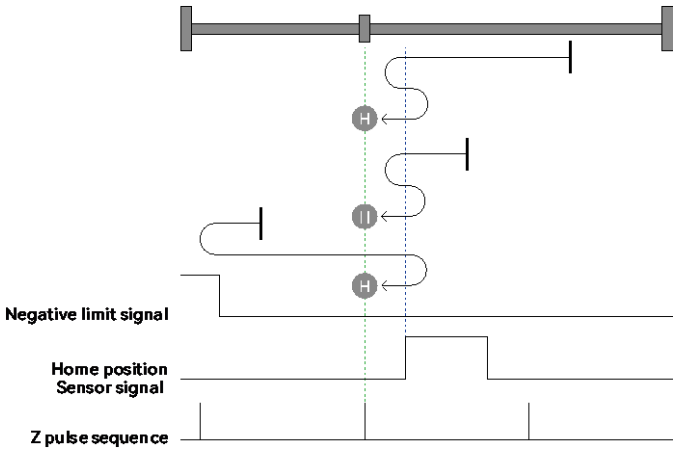


Figure Appendix 1-17 Homing Mode 14 Trajectory and Signal Status

- Mode 15, reserved please do not set.
- Mode 16, reserved please do not set.
- Mode 17, find NL

If the NL is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of NL and running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of NL, set stop position as home.

If the NL is valid when starting, run in the positive direction at low velocity. Decelerate to stop after encountering ON→OFF status of NL, set stop position as home.

As shown in Appendix 1-18, see Table Appendix 1-6.

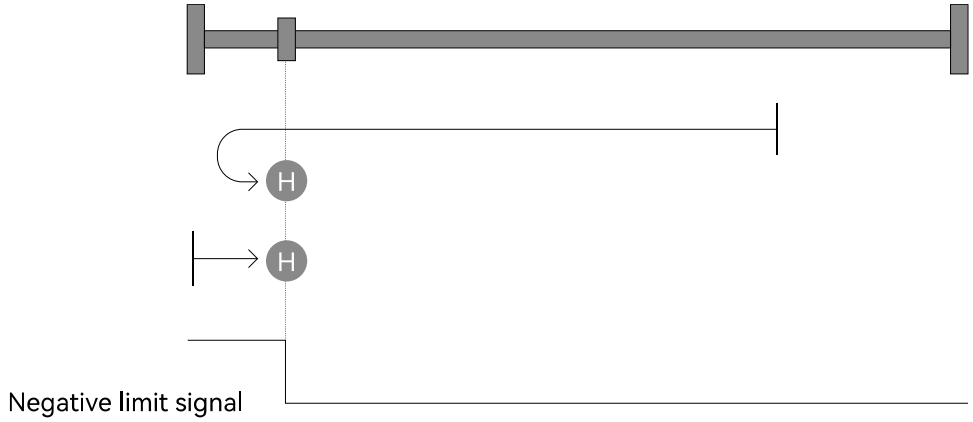


Figure Appendix 1-18 Homing Mode 17 Trajectory and Signal Status

■ Mode 18, Find PL

If the PL is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of PL and running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of PL, set stop position as home.

If the PL is valid when starting, run in the positive direction at low velocity. Decelerate to stop after encountering ON→OFF status of PL, set stop position as home.

As shown in Appendix 1-19, see Table Appendix 1-6.

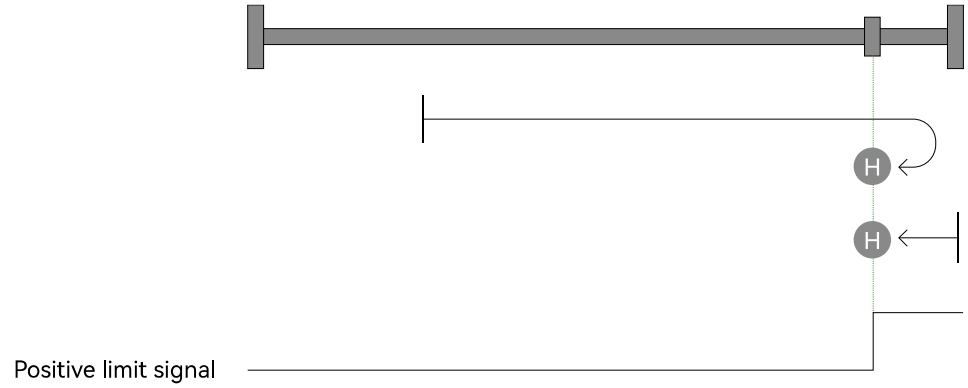


Figure Appendix 1-19 Homing Mode 18 Trajectory and Signal Status

■ Mode 19, Find HSW ON→OFF position when running in negative direction

If the HSW is invalid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the OFF→ON status of the HSW and running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set stop position as home.

If the HSW is valid when starting, run in the negative direction at high velocity. Decelerate to stop after encountering the ON→OFF status of the HSW and reverse back to the HSW valid position at high velocity and decelerate to stop, then running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

In this homing mode, no matter encountering the PL or NL in ON status, the homing process will stop and alarm

As shown in Appendix 1-20, see Table Appendix 1-6.

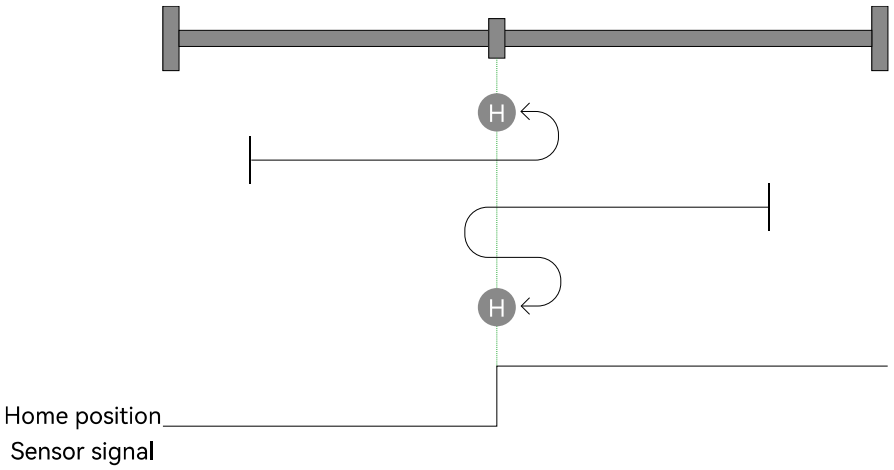


Figure Appendix 1-20 Homing Mode 19 Trajectory and Signal Status

■ Mode 20, Find HSW OFF→ON position when running in the positive direction

If the HSW is invalid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the OFF→ON status of the HSW and reverse back to the HSW valid position at high velocity and decelerate to stop, then running in the positive direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the ON→OFF status of the HSW and running in the positive direction at low velocity.

In this homing mode, no matter encountering the PL or NL in ON status, the homing process will stop and alarm

As shown in Appendix 1-21, see Table Appendix 1-6.

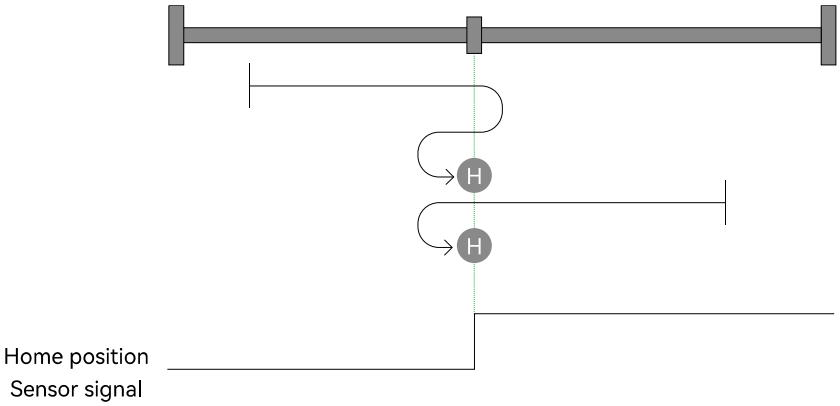


Figure Appendix 1-21 Homing Mode 20 Trajectory and Signal Status

■ Mode 21, Find HSW ON→OFF position when running in the positive direction

If the HSW is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of the HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set stop position as home.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the ON→OFF status of the HSW and reverse back to the HSW valid position at high velocity and decelerate to stop, then running in the positive direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

In this homing mode, no matter encountering the PL or NL in ON status, the homing process will stop and alarm.

As shown in Appendix 1-22, see Table Appendix 1-6.

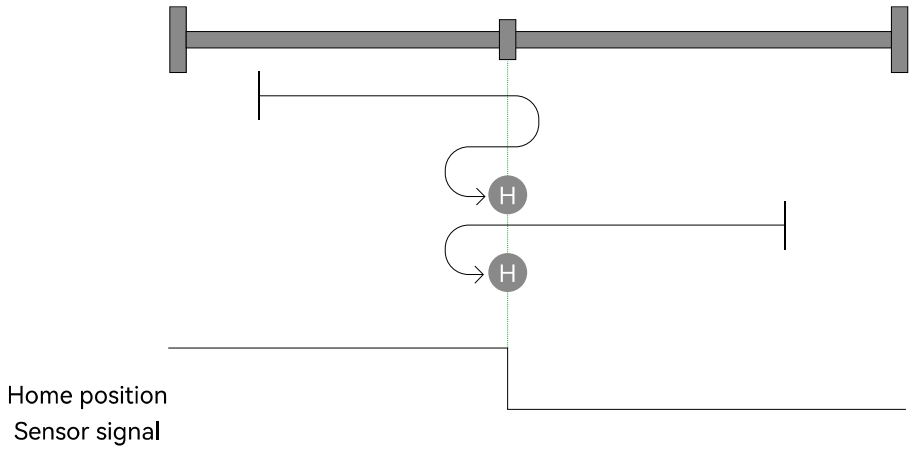


Figure Appendix 1-22 Homing Mode 21 Trajectory and Signal Status

■ Mode 22, Find HSW OFF→ON position when running in negative direction

If the HSW is invalid when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF→ON status of the HSW and reverse back to the HSW valid position at high velocity and decelerate to stop, then running in negative direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the ON→OFF status of the HSW and running in negative direction at low velocity.

In this homing mode, no matter encountering the PL or NL in ON status, the homing process will stop and alarm.

As shown in Appendix 1-23, see Table Appendix 1-6.

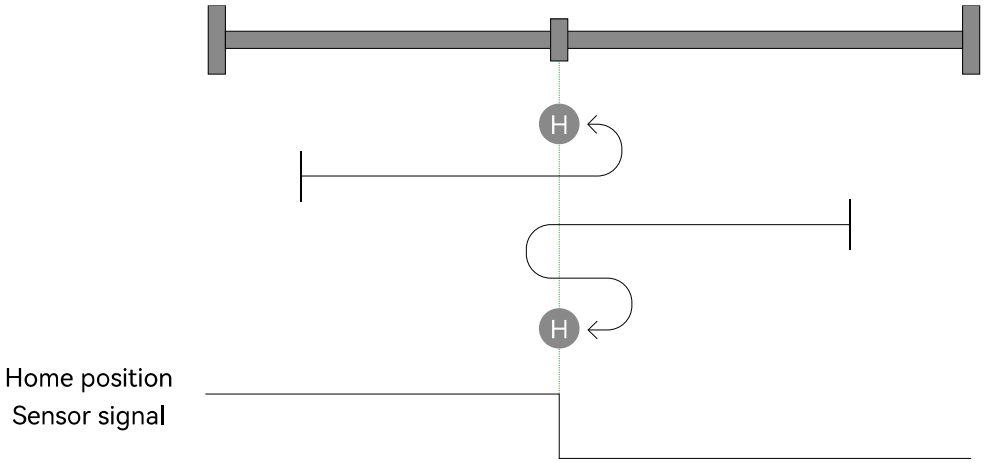


Figure Appendix 1-23 Homing Mode 22 Trajectory and Signal Status

■ Mode 23, Find HSW ON → OFF position when running in negative direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area),then running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering OFF → ON status of the HSW and running in negative direction at high velocity. Decelerate to stop after encountering the ON →OFF status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering

ON→OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area),then running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for the second time, stop homing process and alarm.

As shown in Appendix 1-24, see Table Appendix 1-6.

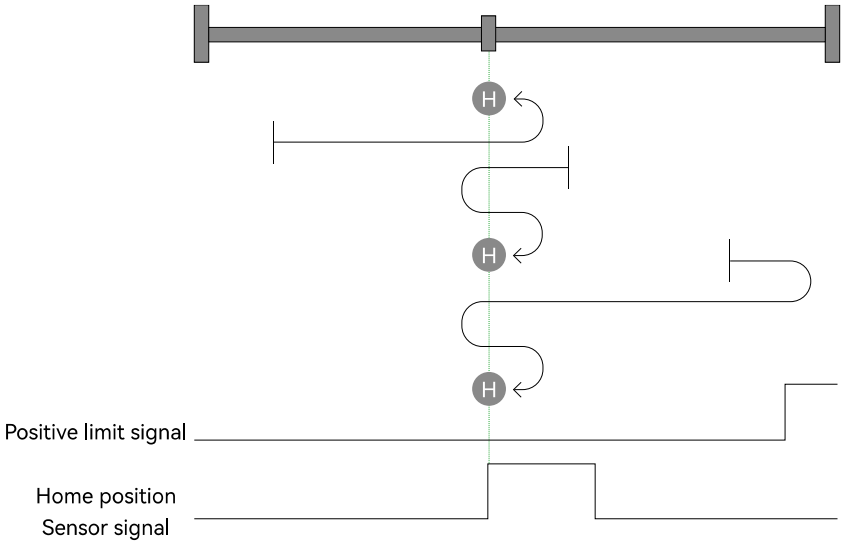


Figure Appendix 1-24 Homing Mode 23 Trajectory and Signal Status

■ Mode 24, Find HSW OFF → ON position when running in the positive direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the OFF → ON status of HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering OFF → ON status of the HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop, then running in the positive direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for the second time, stop homing process and alarm.

As shown in Appendix 1-25, see Table Appendix 1-6.

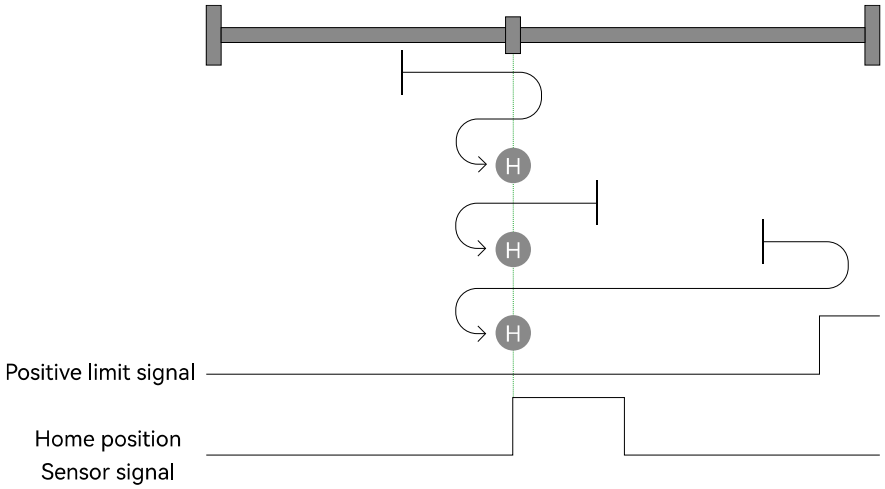


Figure Appendix 1-25 Homing Mode 24 Trajectory and Signal Status

■ Mode 25, Find HSW OFF → ON position when running in negative direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at high velocity. Decelerate to stop after encountering the OFF → ON status of HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop, then running in negative direction at low velocity. Decelerate to stop after encountering the OFF → ON status of HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON → OFF status of the HSW and running in negative direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in negative direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for the second time, stop homing process and Alarm

As shown in Appendix 1-26, see Table Appendix 1-6.

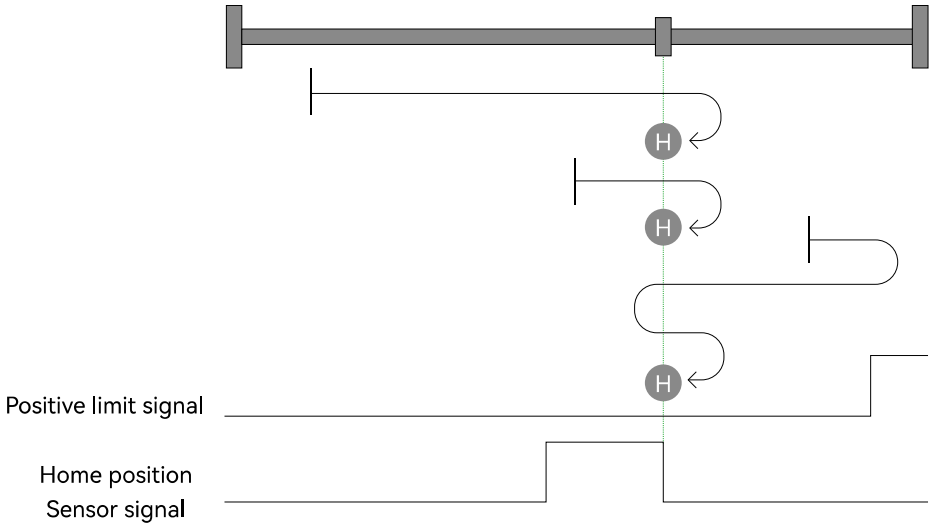


Figure Appendix 1-26 Homing Mode 25 Trajectory and Signal Status

■ Mode 26, Find HSW ON → OFF position when running in negative direction, while encountering PL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in negative direction at high velocity. Decelerate to stop after encountering OFF → ON status of the HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in the positive direction at low velocity. Decelerate to stop after encountering the ON → OFF status of HSW, set the stop position as home.

If the HSW is valid when starting, run in positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in the positive direction at low velocity.. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for the second time, stop homing process and alarm.

As shown in Appendix 1-27, see Table Appendix 1-6.

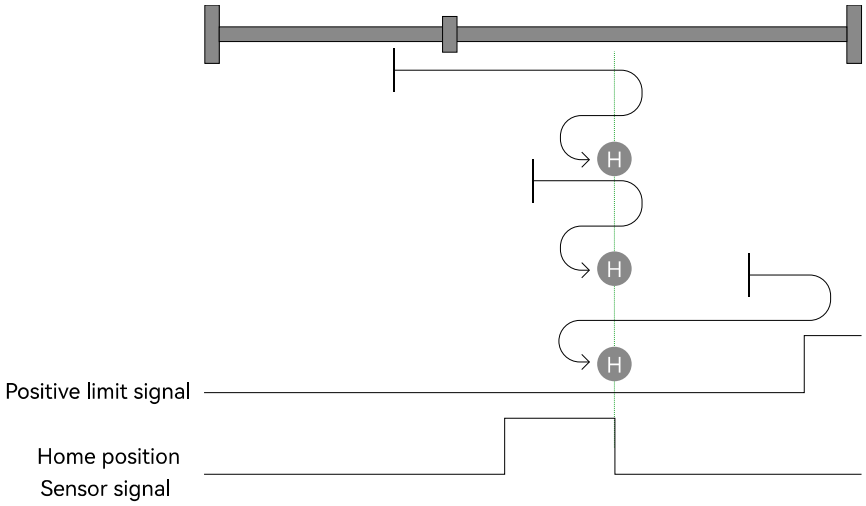


Figure Appendix 1-27 Homing Mode 26 Trajectory and Signal Status

■ Mode 27, Find HSW ON → OFF position when running in the positive direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of NL and running in positive direction at high velocity. Decelerate to stop after encountering ON → OFF status of the HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in the positive direction at low velocity. Decelerate to stop after encountering the ON →OFF status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area),then running in the positive direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status

of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-28, see Table Appendix 1-6.

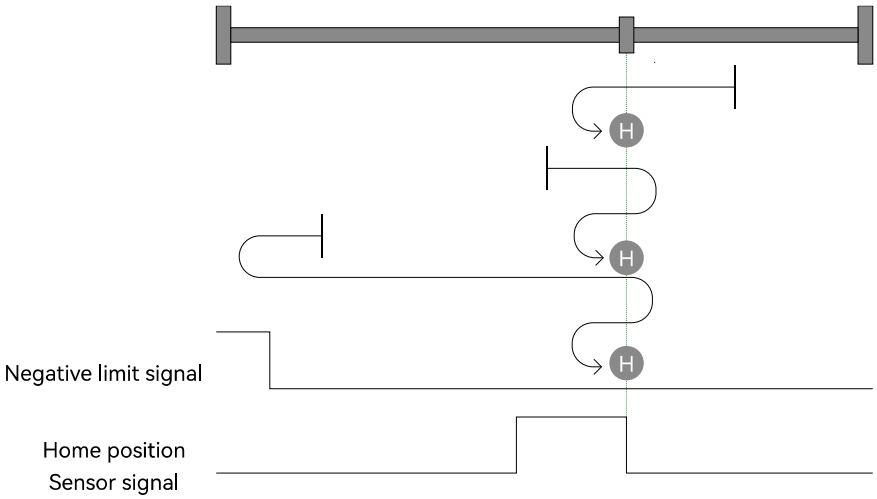


Figure Appendix 1-28 Homing Mode 27 Trajectory and Signal Status

■ Mode 28, Find HSW OFF → ON position when running in the positive direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering the OFF → ON status of HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop, then running in negative direction at low velocity. Decelerate to stop after encountering the OFF → ON status of HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON status of PL and running in the positive direction at high velocity. Decelerate to stop after encountering ON → OFF status of the HSW and running in negative direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in the positive direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in negative direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-29, see Table Appendix 1-6.

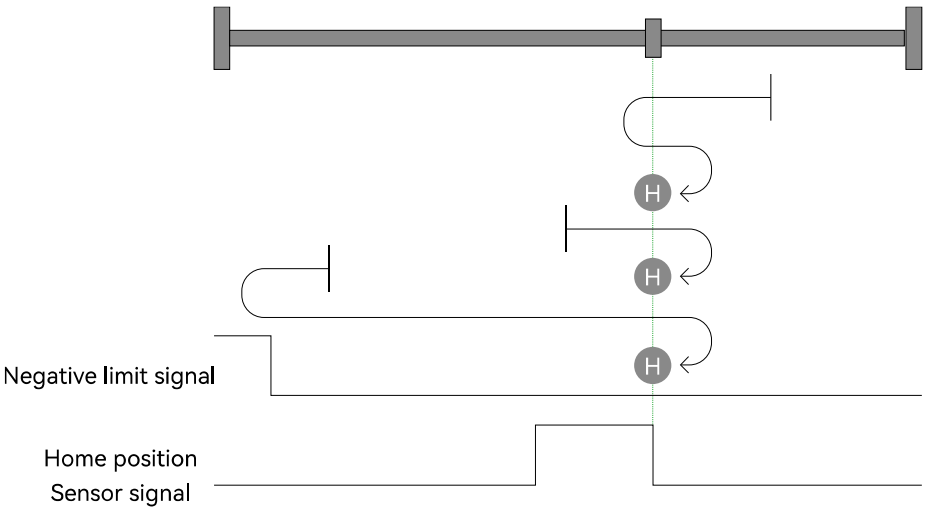


Figure Appendix 1-29 Homing Mode 28 Trajectory and Signal Status

■ Mode 29, Find HSW OFF → ON position when running in the positive direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering the ON → OFF status of HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the OFF → ON status of HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in negative direction at high velocity. Decelerate to stop after encountering OFF → ON status of the HSW and reverse back to the HSW invalid position at high velocity and decelerate to stop, then running in the positive direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and running in the positive direction at low velocity. Decelerate to stop after encountering the OFF→ON status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-30, see Table Appendix 1-6.

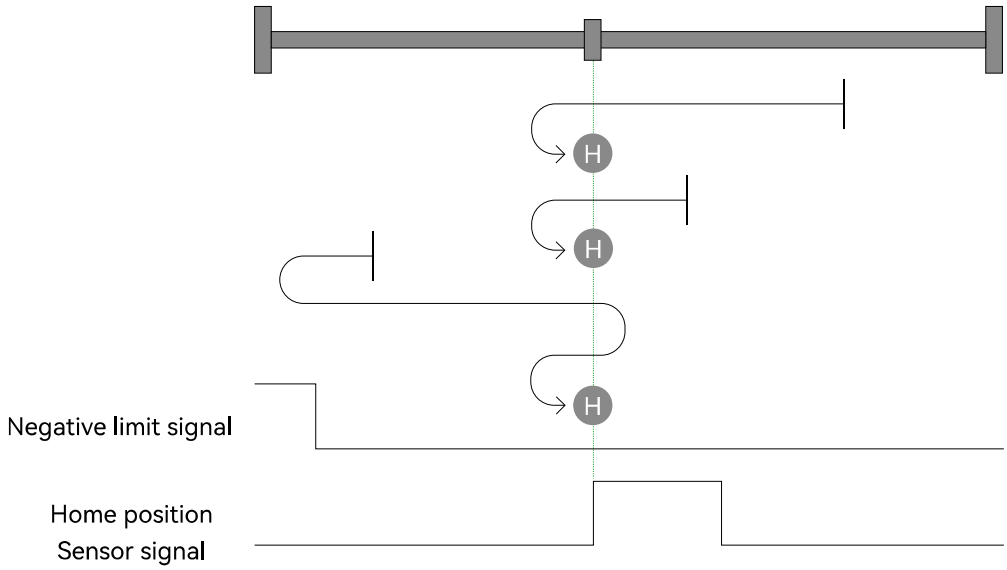


Figure Appendix 1-30 Homing Mode 29 Trajectory and Signal Status

■ Mode 30, Find HSW ON → OFF position when running in negative direction, while encountering NL automatically reverse

If the HSW is invalid and home at sensor's the positive side when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→ OFF status of the HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

If the HSW is invalid and home at sensor's negative side when starting, run in negative direction at high velocity. Decelerate to stop after encountering the ON status of NL and running in the positive direction at high velocity. Decelerate to stop after encountering the OFF → ON status of HSW and running in negative direction at low velocity. Decelerate to stop after encountering the ON → OFF status of HSW, set the stop position as home.

If the HSW is valid when starting, run in negative direction at high velocity. Decelerate to stop after encountering ON→OFF status of HSW and reverse back to the HSW valid position at high velocity and decelerate to stop(If the HSW valid area is narrow, it might enter the other side of the HSW invalid position area), then running in negative direction at low velocity. Decelerate to stop after encountering the ON→OFF status of the HSW, set the stop position as home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop

homing process and alarm.

As shown in Appendix 1-31, see Table Appendix 1-6.

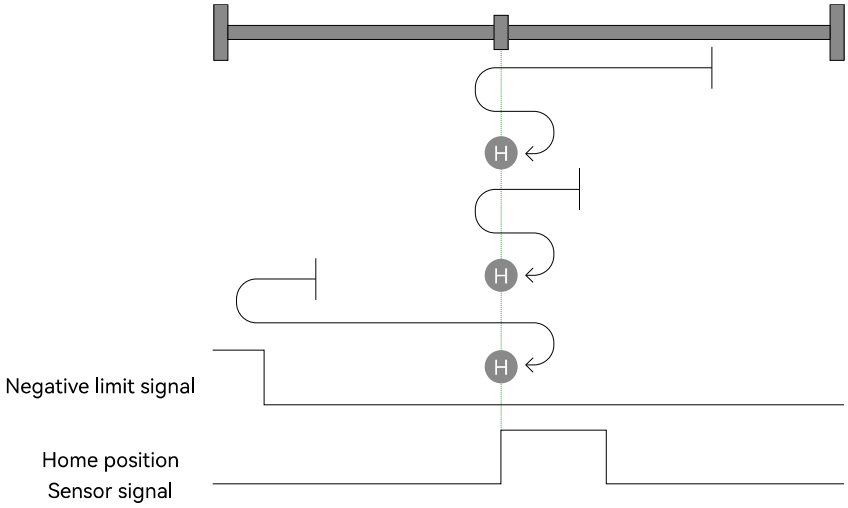


Figure Appendix 1-31 Homing Mode 30 Trajectory and Signal tatus

- Mode 31, reserved please do not set.
- Mode 32, reserved please do not set.
- Mode 33, Find the nearest Z pulse when running in negative direction

Find the nearest Z pulse position and set it as the home in negative direction at low velocity when starting. If running in negative direction encountering the ON status of NL before Z pulse position, decelerate to stop and find the nearest Z pulse position in the positive direction then set as the home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the NL; Encountering the ON status of the PL or encountering the ON status of the NL for the second time, stop homing process and alarm.

As shown in Appendix 1-32, see Table Appendix 1-6.

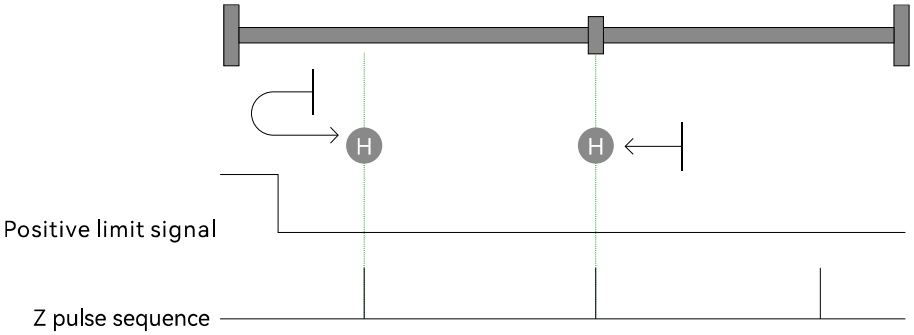


Figure Appendix 1-32 Homing Mode 33 Trajectory and Signal Status

- Mode 34, Find the nearest Z pulse when running in the positive direction

Find the nearest Z pulse position and set it as the home in negative direction at low velocity when starting. If running in negative direction encountering the ON status of PL before Z pulse position, decelerate to stop and find the nearest Z pulse position in negative direction then set as the home.

In this homing mode, automatically reverse after running in the positive direction and encountering the ON status of the PL; Encountering the ON status of the NL or encountering the ON status of the PL for the second time, stop homing process and alarm.

As shown in Appendix 1-33, see Table Appendix 1-6.

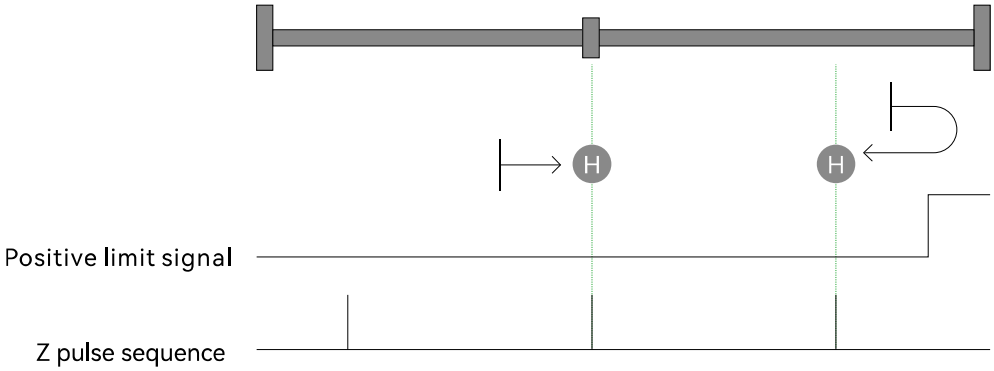


Figure Appendix 1-33 Homing Mode 34 Trajectory and Signal Status

Appendix2 Instruction error code description

When the instruction is execute incorrectly, the ErrorID has a corresponding value. Search the following table for the meaning of the ErrorID value and solution.

Error code		Meaning of ErrorID	Solution
Hexadecimal	Decimalism		
1001	4097	The axis of instruction operation is not configured in the software	Check whether the axis of instruction operation is configured in the software
1002	4098	The encoder axis can’t execute the operation	The encoder axis can’t execute the operation
1003	4099	Incorrect axis type or capture mode	Check the axis type or whether it is the encoder axis
1004	4100	Axis out of range	Please set according to the supported axis range of the controller
1005	4101	Acceleration out of range	Check whether the parameter is within the legal range
1006	4102	Deceleration out of range	Check whether the parameter is within the legal range
1007	4103	Jerk out of range	Check whether the parameter is within the legal range
1008	4104	Velocity out of range	Check whether the parameter is within the legal range
1009	4105	Position out of range	The value of the MC_MoveAbsolute input Position cannot exceed the value of the modulus in the axis configuration
100A	4106	Direction out of range	Check whether the parameter is within the legal range
100B	4107	BufferMode out of range	Check whether the parameter is within the legal range
100C	4108	Velocity overshoot value out of range	Check whether the parameter is within the legal range
100D	4109	Reference type input error	Check whether the parameter is within the legal range
100E	4110	Instructions cannot be executed	Instructions can’t be executed when MC_Home/ MC_TorqueControlWithVelocity instructions are executed.
100F	4111	Instructions cannot be executed	This instruction can be executed when the axis is in the standstill
1010	4112	MC_ Reset execution failed	Check the axis type and Drive status
1011	4113	Instruction execution will result in exceeding the software limit	Modify the instruction execution position
1012	4114	Failed to write Position in MC_Home Instruction.	Check whether the parameter is within the legal range
1013	4115	Axis type error	Axis can’t execute MC_Home instruction
1014	4116	The input trigger bit of the MC_TouchProbe instruction error	Check whether the value is within the legal range
1015	4117	The input trigger bit of the MC_TouchProbe instruction is being used	Set the value of the TriggerInput to False
1016	4118	MC_TouchProbe window function error.	Modify the upper and lower limit range of window settings.
1017	4119	Only one MC_TouchProbe instruction can be executed on an axis at the same time.	Check whether the value is within the legal range
1018	4120	MC_TouchProbe instruction mode setting error	Check whether the value is within the legal range
1019	4121	MC_TouchProbe instruction can’t be executed	MC_TouchProbe instruction cannot be executed due to the axis disconnection
1100	4352	Electronic cam ID out of range	Check whether the parameter is within the legal range

Error code		Meaning of ErrorID	Solution
Hexadecimal	Decimalism		
1101	4353	Master axis out of range	Check whether the parameter is within the legal range
1102	4354	Coupling mode out of range	Check whether the parameter is within the legal range
1103	4355	Master axis position scaling out of range	Check whether the parameter is within the legal range
1104	4356	Slave axis position scaling out of range	Check whether the parameter is within the legal range
1105	4357	Selection of Master axis position source out of range	Check whether the parameter is within the legal range
1106	4358	Conflict between Master and Slave axis ID	Master and slave axis numbers can’ t be the same
1107	4359	Electronic gear numerator out of range	Check whether the parameter is within the legal range
1108	4360	Electronic gear denominator out of range	Check whether the parameter is within the legal range
1109	4361	Cam coupling Function activation position out of range	Check whether the parameter is within the legal range
110A	4362	Master axis position offset out of Master axis cam periodicity range	The legal range is between the negative and positive value of the Master axis periodicity range (Master axis cam periodicity=Master axis maximum value-Master axis minimum value)
110B	4363	Slave axis position offset out of Slave axis cam periodicity range	The legal range is between the negative and positive value of the Slave axis periodicity range (Slave axis cam periodicity=Slave axis maximum value-Slave axis minimum value)
110C	4364	Cam curve was not established in the software	Check whether the corresponding cam curve of the CamTable is established in the software
110D	4365	Rotary cutting axis radius out of range	Check whether the parameter is within the legal range
110E	4366	Feed axis radius out of range	Check whether the parameter is within the legal range
110F	4367	Rotary cutting length out of range	Check whether the parameter is within the legal range
1110	4368	Start position of synchronization area out of range	Check whether the parameter is within the legal range
1111	4369	End position of synchronization area out of range	Check whether the parameter is within the legal range
1112	4370	Synchronization position of synchronization area out of range	Check whether the parameter is within the legal range
1113	4371	Rotary cutting ID out of range	Check whether the parameter is within the legal range
1114	4372	Rotary cutting blade number out of range	Check whether the value is within the legal range
1115	4373	Internal status of rotary cutting is illegal	Modify the rotary cutting initialize parameters
1116	4374	Rotary cutting function is not initialized	Please initialize of the rotary cutting function first
1200	4608	Prereading depth out of range	Check whether the value is within the legal range
1201	4609	Synthesis velocity overshoot out of range	Check whether the parameter value is within the legal range
1202	4610	GCode file code out of range	Check whether the parameter value is within the legal range
1203	4611	GCode file is empty	Check whether the file is legal
1204	4612	GCode file parsing error	Check whether the file is legal
1205	4613	Axis group ID out of range	Check whether the value is within the legal range
1206	4614	Mode out of range	Check whether the value is within the legal range

Error code		Meaning of ErrorID	Solution
Hexadecimal	Decimalism		
120A	4618	GCode instruction format error	Check and ensure that the format of the GCode instruction meets the requirements
1301	4865	SDO timeout	Check whether the communication port connection is appropriate and the baud rate is consistent
1302	4866	SDO input error	Check whether the value is within the legal range
1303	4867	SDO other errors	Check whether the slave station is in the working status
1401	5121	Movement direction limit	Check the setting of MC_Power instruction，EnablePositive and EnableNegative settings
1403	5123	MC_HaltSuperimposed cannot be executed	MC_HaltSuperimposed instruction can be executed when MC_MoveSuperimposed instruction is executing.
1501	5377	State machine restrictions, function cannot be executed.	Please refer to execution rules of the instruction status machine
1502	5378	BufferMode cache exceeded	The motion command BufferMode only supports one level of motion buffering, please use the previous instruction Busy to trigger the next instruction, so as to realize multi-level motion buffering.
1503	5379	Instruction not supports BufferMode function	The instruction cannot execute BufferMode operation
1504	5380	Axis abnormality disable	Please check the drive status
1601	5633	Axis typesetting out of range	Modify the axis type in the axis configuration
1602	5634	Drive alarm	Clear the Drive alarm before control
1603	5635	Drive communication timeout	Check whether the connection between controller and Drive is correct
1604	5636	Software limit exceeded	Check whether the axis status or software limit setting is appropriate
1606	5638	Position following difference exceeded	Check whether the axis status or the Position following difference is appropriate
4000	16384	Expansion module ID out of range	Set the expansion module ID(parameter ModularID) within the allowed range
4001	16385	Communication timeout between controller and expansion module	Check and ensure that the controller and expansion module are connected properly
4002	16386	Communication error between controller and expansion module	1. Check and ensure that the controller and expansion module are connected properly 2. Avoid installing and using the product in an environment with strong interference
4100	16640	Pulse output channel number setting error	Set the pulse output channel number properly (ChannelNum)
4101	16641	Pulse output channel has been used	Check pulse channel usage to avoid duplication
4120	16672	Pulse input edge detection setting error	Set the pulse input edge detection properly (EdgeSelect)
4121	16673	Pulse input mode setting error	Ste the pulse input mode properly (SignalMode)
4122	16674	Pulse input counting mode setting error	Set the pulse input counting mode properly (CounterDirection)
4123	16675	Pulse input frequency multiplication mode setting error	Set the Pulse input frequency multiplication mode counting mode properly (Multi_Frequency)
4200	16896	Trigger Stop again during the Stop process	Modify Stop execution timing
4201	16897	Trigger Stop when MC_StopAtPhase does not control axis in motion..	Modify Stop execution timing
4202	16898	RoundPhase setting error	Modify and ensure that the value of RoundPhase is greater than 0
4203	16899	StopPhase setting error	Modify and ensure that the value of StoppPhase is greater than 0

Error code		Meaning of ErrorID	Solution
Hexadecimal	Decimalism		
			and less than RoundPhase
4220	16928	TorqueRamp setting error	Modify and ensure that the value of TorqueRamp is greater than 0
4240	16960	ScaleDen or ScaleNum setting error	Modify and ensure that the s ScaleDen and ScaleNum are greater than 0.
4241	16961	UnitsPerTurn setting error	Modify and ensure that UnitsPerTurn is greater than 0
4242	16962	AxisType setting error	Modify and ensure that AxisType is within the allowed range
4243	16963	Modulo setting error	Modify and ensure that Modulo is greater than 0
4260	16992	Lag setting error	Modify and ensure that Lag is greater than or equal to 0
4261	16993	HoldTime setting error	Modify and ensure that HoldTime is greater than or equal to 0
4280	17024	Index (cam key points ID) setting error	Modify and ensure that Index is greater than 0 and not greater than the total number of keypoints in the cam table.
4281	17025	Index (cam tappets ID) setting error	Modify and ensure that Index is greater than 0 and not greater than the total number of tappets in the cam table.
4282	17026	Index (Master axis position) setting error	Modify and ensure that MasterPos is greater than 0 and not greater than the maximum spindle position of the cam table.
4283	17027	Tappet number exceeds the maximum	Executed when the number of tappets has not reached the maximum value
4290	17040	MaxNum out of range	Modify and ensure MaxNum is greater than 0 or less than 20
4291	17041	Precision out of range	Modify and ensure the Precision is greater than or equal to 0
4500	17664	Slave NodeID out of range	Ensure the slave NodeID is within the allowed range
4501	17665	Mode setting error	Modify and ensure the Mode is within the allowed range
4502	17666	Diagnostic slave Node does not exist (not configured)	Execute diagnostic function on configured slave Node
4503	17667	Local CANopen port is not master mode	Execute diagnostic function in master mode
4510	17680	Slave address NodeID out of range	Ensure the slave address NodeID is within allowed range
4511	17681	Diagnostic slave does not exist (not configured)	Execute diagnostic function on configured slave
5000	20480	Axis group status machine not allowed to execute	Execute in allowed status machine
5001	20481	IdentNum setting error	Modify and ensure the IdentNum is within the allowed range
5002	20482	Setting value of the IdentNum has been used in axis group	Modify and ensure the IdentNum is not duplicated with other IdentNum already used in axis group
5003	20483	The axis specified by pin Axis already belongs to other axis groups	Modify the parameters of the Axis pin
5004	20484	Axis group not established	Execute this function when the axis group has been established
5005	20485	The state of each axis belonging to an axis group does not allow the execution of the function.	Execute function when all axes Standstill
5006	20486	TransitionMode setting error	Modify and ensure TransitionMode is within the allowed range
5007	20487	TransitionParameter setting error	Modify and ensure TransitionParameter is within the allowed range
5008	20488	Parameter confliction between BufferMode with TransitionMode	Modify and ensure the parameter is within the allowed range
5009	20489	CircMode setting error	Modify and ensure the CircMode is within the allowed range
5010	20496	PathChoice setting error	Modify and ensure PathChoice is within the allowed range
5100	20736	MCode number error	Ensure the MCode number is within the allowed range

Error code		Meaning of ErrorID	Solution
Hexadecimal	Decimalism		
5200	20992	MoveMode setting error	Modify and ensure MoveMode is within the allowed range
5201	20993	Mode setting error	Modify and ensure Mode is within the allowed range
5300	21248	HomeMode setting error	Modify and ensure HomeMode is within the allowed range
5301	21249	VelocityFast setting error	Modify and ensure VelocityFast is within the allowed range
5302	21250	VelocitySlow setting error	Modify and ensure VelocitySlow is within the allowed range
5303	21251	Acceleration setting error	Modify and ensure Acceleration is within the allowed range
5304	21252	Deceleration setting error	Modify and ensure Deceleration is within the allowed range
5305	21253	Jerk setting error	Modify and ensure Jerk is within the allowed range

HCFA

创新 诚信 服务

HCFA

HCFA_ATC