SIEMENS

SIMOTION

PLCopen Blocks

Function Manual

Forward

Introduction 1

Description 2

Blocks 3

Troubleshooting PLCopen blocks 4

05/2009

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠DANGER
indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠WARNING
indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠CAUTION
with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

CAUTION
without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

NOTICE
indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Proper use of Siemens products

Note the following:

⚠WARNING
Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be adhered to. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

℗ 03/2009

Forward

SIMOTION Documentation

An overview of the SIMOTION documentation can be found in a separate list of references.

This documentation is included as electronic documentation with the supplied SIMOTION SCOUT.

The SIMOTION documentation consists of 9 documentation packages containing approximately 80 SIMOTION documents and documents on related systems (e.g. SINAMICS).

The following documentation packages are available for SIMOTION V4.1 SP4:

- SIMOTION Engineering System
- SIMOTION System and Function Descriptions
- SIMOTION Service and Diagnostics
- SIMOTION Programming
- SIMOTION Programming - References
- SIMOTION C
- SIMOTION P350
- SIMOTION D4xx
- SIMOTION Supplementary Documentation

Hotline and Internet addresses

Siemens Internet address

The latest information about SIMOTION products, product support, and FAQs can be found on the Internet at:

- General information:
  - **http://www.siemens.de/simotion** (German)
  - **http://www.siemens.com/simotion** (international)
- Downloading documentation
  Further links for downloading files from Service & Support.
  http://support.automation.siemens.com/WW/view/en/10805436
- Individually compiling documentation on the basis of Siemens contents with the My Documentation Manager (MDM), refer to **http://www.siemens.com/mdm**

  My Documentation Manager provides you with a range of features for creating your own documentation.
- FAQs
  You can find information on FAQs (frequently asked questions) by clicking **http://support.automation.siemens.com/WW/view/en/10805436/133000**.

Additional support

We also offer introductory courses to help you familiarize yourself with SIMOTION.

For more information, please contact your regional Training Center or the main Training Center in 90027 Nuremberg, Germany.

Information about training courses on offer can be found at:

http://www.sitrain.com

Technical support

If you have any technical questions, please contact our hotline:

	Europe / Africa
Phone	+49 180 5050 222 (subject to charge)
Fax	+49 180 5050 223
€0.14/min from German wire-line network, mobile phone prices may differ.	
Internet	http://www.siemens.com/automation/support-request

	Americas
Phone	+1 423 262 2522
Fax	+1 423 262 2200
E-mail	mailto:techsupport.sea@siemens.com

	Asia / Pacific
Phone	+86 1064 757575
Fax	+86 1064 747474
E-mail	mailto:support.asia.automation@siemens.com

Note

Country-specific telephone numbers for technical support are provided under the following Internet address:

http://www.automation.siemens.com/partner

Questions about this documentation

If you have any questions (suggestions, corrections) regarding this documentation, please fax or e-mail us at:

Fax	+49 9131- 98 2176
E-mail	mailto:docu.motioncontrol@siemens.com

Table of contents

Forward **3**

1 Introduction **13**

2 Description **15**

2.1 Description of PLCopen blocks 15

3 Blocks **17**

3.1 Handling PLCopen blocks 17
3.1.1 Block instances 17
3.1.2 Upgrading a PLCopen library from V2.3 to V4.1 18

3.2 SingleAxis 19
3.2.1 _MC_Power - Enabling/disabling axis 19
3.2.1.1 Overview 19
3.2.1.2 Schematic diagram 19
3.2.1.3 Purpose 19
3.2.1.4 Applicable for 20
3.2.1.5 Requirements 20
3.2.1.6 Input parameters 20
3.2.1.7 Output parameter 21
3.2.1.8 ErrorIDs 22
3.2.1.9 Example 23
3.2.2 _MC_Stop - Stopping axis 24
3.2.2.1 Overview 24
3.2.2.2 Schematic diagram 24
3.2.2.3 Purpose 24
3.2.2.4 Applicable for 25
3.2.2.5 Requirements 25
3.2.2.6 Input parameters 25
3.2.2.7 Output parameter 26
3.2.2.8 ErrorIDs 27
3.2.2.9 Example 28
3.2.3 _MC_Home - Homing axis/clearing absolute value encoder offset 29
3.2.3.1 Overview 29
3.2.3.2 Schematic diagram 29
3.2.3.3 Purpose 29
3.2.3.4 Applicable for 30
3.2.3.5 Requirements 30
3.2.3.6 Input parameters 31
3.2.3.7 Output parameter 32
3.2.3.8 ErrorIDs 32
3.2.3.9 Examples 33
3.2.4 _MC_MoveAbsolute - Absolute positioning of axis 36
3.2.4.1 Overview 36
3.2.4.2 Schematic diagram 36
3.2.4.3 Purpose 36
3.2.4.4 Applicable for 37
3.2.4.5 Requirements 37
3.2.4.6 Input parameters 37

3.2.4.7 Output parameter ... 39
3.2.4.8 ErrorIDs ... 39
3.2.4.9 Example ... 39
3.2.5 _MC_MoveRelative - Relative positioning of axis ... 41
3.2.5.1 Overview ... 41
3.2.5.2 Schematic diagram ... 41
3.2.5.3 Purpose ... 41
3.2.5.4 Applicable for ... 42
3.2.5.5 Requirements ... 42
3.2.5.6 Input parameters ... 42
3.2.5.7 Output parameter ... 43
3.2.5.8 ErrorIDs ... 44
3.2.5.9 Example ... 44
3.2.6 _MC_MoveVelocity - Traversing axis at a specified velocity ... 46
3.2.6.1 Overview ... 46
3.2.6.2 Schematic diagram ... 46
3.2.6.3 Purpose ... 46
3.2.6.4 Applicable for ... 47
3.2.6.5 Requirements ... 47
3.2.6.6 Input parameters ... 47
3.2.6.7 Output parameter ... 48
3.2.6.8 ErrorIDs ... 49
3.2.6.9 Example ... 50
3.2.7 _MC_MoveAdditive - Relative positioning to current target position ... 51
3.2.7.1 Overview ... 51
3.2.7.2 Schematic diagram ... 51
3.2.7.3 Purpose ... 51
3.2.7.4 Applicable for ... 52
3.2.7.5 Requirements ... 52
3.2.7.6 Input parameters ... 53
3.2.7.7 Output parameter ... 54
3.2.7.8 ErrorIDs ... 55
3.2.7.9 Example ... 55
3.2.8 _MC_MoveSuperimposed - Superimposed positioning ... 57
3.2.8.1 Overview ... 57
3.2.8.2 Schematic diagram ... 57
3.2.8.3 Purpose ... 57
3.2.8.4 Applicable for ... 58
3.2.8.5 Requirements ... 58
3.2.8.6 Input parameters ... 58
3.2.8.7 Output parameter ... 59
3.2.8.8 ErrorIDs ... 60
3.2.8.9 Example ... 60
3.2.9 _MC_PositionProfile - Traveling through position/time profile ... 62
3.2.9.1 Overview ... 62
3.2.9.2 Schematic diagram ... 62
3.2.9.3 Purpose ... 62
3.2.9.4 Applicable for ... 62
3.2.9.5 Requirements ... 63
3.2.9.6 Input parameters ... 63
3.2.9.7 Output parameter ... 64
3.2.9.8 ErrorIDs ... 65
3.2.9.9 Example ... 65
3.2.10 _MC_VelocityProfile - Traveling through velocity/time profile ... 67
3.2.10.1 Overview ... 67

3.2.10.2 Schematic diagram 67
3.2.10.3 Purpose 67
3.2.10.4 Applicable for 67
3.2.10.5 Requirements 68
3.2.10.6 Input parameters 68
3.2.10.7 Output parameter 69
3.2.10.8 ErrorIDs 70
3.2.10.9 Example 70
3.2.11 _MC_Reset - Resetting errors/alarms on the axis or triggering a restart 72
3.2.11.1 Overview 72
3.2.11.2 Schematic diagram 72
3.2.11.3 Purpose 72
3.2.11.4 Applicable for 73
3.2.11.5 Requirements 73
3.2.11.6 Input parameters 74
3.2.11.7 Output parameter 74
3.2.11.8 ErrorIDs 75
3.2.11.9 Example 75
3.2.12 _MC_ReadActualPosition - Reading actual position of axis 77
3.2.12.1 Overview 77
3.2.12.2 Schematic diagram 77
3.2.12.3 Purpose 77
3.2.12.4 Applicable for 78
3.2.12.5 Input parameters 78
3.2.12.6 Output parameter 78
3.2.12.7 ErrorIDs 79
3.2.12.8 Example 80
3.2.13 _MC_ReadStatus - Reading an axis status 81
3.2.13.1 Overview 81
3.2.13.2 Schematic diagram 81
3.2.13.3 Purpose 82
3.2.13.4 Applicable for 82
3.2.13.5 Input parameters 82
3.2.13.6 Output parameter 83
3.2.13.7 ErrorIDs 84
3.2.13.8 Example 85
3.2.14 _MC_ReadAxisError - Reading an axis error 86
3.2.14.1 Overview 86
3.2.14.2 Schematic diagram 86
3.2.14.3 Purpose 86
3.2.14.4 Applicable for 86
3.2.14.5 Input parameters 87
3.2.14.6 Output parameter 87
3.2.14.7 ErrorIDs 88
3.2.14.8 Example 88
3.2.15 _MC_ReadParameter - Reading axis parameter and outputting in data type LREAL 90
3.2.15.1 Overview 90
3.2.15.2 Schematic diagram 90
3.2.15.3 Purpose 90
3.2.15.4 Applicable for 90
3.2.15.5 Input parameters 91
3.2.15.6 Output parameter 92
3.2.15.7 ErrorIDs 92
3.2.15.8 Example 93
3.2.16 _MC_ReadBoolParameter - Reading axis parameter and outputting in data type BOOL 94

3.2.16.1 Overview ... 94
3.2.16.2 Schematic diagram ... 94
3.2.16.3 Purpose ... 94
3.2.16.4 Applicable for ... 95
3.2.16.5 Input parameters ... 95
3.2.16.6 Output parameter ... 96
3.2.16.7 ErrorIDs ... 96
3.2.16.8 Example ... 96
3.2.17 _MC_WriteParameter - Writing axis parameter of data type LREAL ... 98
3.2.17.1 Overview ... 98
3.2.17.2 Schematic diagram ... 98
3.2.17.3 Purpose ... 98
3.2.17.4 Applicable for ... 98
3.2.17.5 Input parameters ... 99
3.2.17.6 Output parameter ... 100
3.2.17.7 ErrorIDs ... 100
3.2.17.8 Example ... 101
3.2.18 _MC_WriteBoolParameter - Writing axis parameter of data type BOOL ... 102
3.2.18.1 Overview ... 102
3.2.18.2 Schematic diagram ... 102
3.2.18.3 Purpose ... 102
3.2.18.4 Applicable for ... 102
3.2.18.5 Input parameters ... 103
3.2.18.6 Output parameter ... 104
3.2.18.7 ErrorIDs ... 104
3.2.18.8 Example ... 105

3.3 MultiAxis ... 106
3.3.1 _MC_GearIn - Starting gearing ... 106
3.3.1.1 Overview ... 106
3.3.1.2 Schematic diagram ... 106
3.3.1.3 Purpose ... 107
3.3.1.4 Applicable for ... 107
3.3.1.5 Requirements ... 107
3.3.1.6 Input parameters ... 108
3.3.1.7 Output parameter ... 111
3.3.1.8 ErrorIDs ... 111
3.3.1.9 Examples ... 111
3.3.2 _MC_GearOut - Terminating gearing ... 115
3.3.2.1 Overview ... 115
3.3.2.2 Schematic diagram ... 115
3.3.2.3 Purpose ... 115
3.3.2.4 Applicable for ... 116
3.3.2.5 Requirements ... 116
3.3.2.6 Input parameters ... 116
3.3.2.7 Output parameter ... 116
3.3.2.8 ErrorIDs ... 117
3.3.2.9 Example ... 118
3.3.3 _MC_CamIn - Starting camming ... 119
3.3.3.1 Overview ... 119
3.3.3.2 Schematic diagram ... 119
3.3.3.3 Purpose ... 120
3.3.3.4 Applicable for ... 120
3.3.3.5 Requirements ... 120
3.3.3.6 Input parameters ... 121
3.3.3.7 Output parameter ... 123

3.3.3.8 ErrorIDs 124
3.3.3.9 Examples 124
3.3.4 _MC_CamOut - Terminating camming 134
3.3.4.1 Overview 134
3.3.4.2 Schematic diagram 134
3.3.4.3 Purpose 134
3.3.4.4 Applicable for 135
3.3.4.5 Requirements 135
3.3.4.6 Input parameters 135
3.3.4.7 Output parameter 135
3.3.4.8 ErrorIDs 136
3.3.4.9 Example 136
3.3.5 _MC_Phasing - Changing phase shift between the leading axis and following axis 138
3.3.5.1 Overview 138
3.3.5.2 Schematic diagram 138
3.3.5.3 Purpose 138
3.3.5.4 Applicable for 139
3.3.5.5 Requirements 139
3.3.5.6 Input parameters 139
3.3.5.7 Output parameter 140
3.3.5.8 ErrorIDs 141
3.3.5.9 Example 142

3.4 Advanced functions 143
3.4.1 _MC_Jog - Continuous or incremental jogging 143
3.4.1.1 Overview 143
3.4.1.2 Schematic diagram 143
3.4.1.3 Purpose 144
3.4.1.4 Applicable for 144
3.4.1.5 Requirements 144
3.4.1.6 Input parameters 144
3.4.1.7 Output parameter 146
3.4.1.8 Function 146
3.4.1.9 ErrorIDs 149

4 Troubleshooting PLCopen blocks 151

4.1 Troubleshooting - PLCopen Blocks 151

4.2 Error codes of the errorID (LOW word) 152

4.3 Command abort reason of the errorID (HIGH word) 153

4.4 Query of general errors with the _MC_ReadAxisError function block 155

Index 159

1 Introduction

The PLCopen blocks are designed for use in cyclic programs/tasks and enable motion control programming in a PLC environment. If preferred, they can be used in the LAD/FBD programming language.

PLCopen blocks are available as standard functions (directly from the command library).

The motion control functions meet the PLCopen specifications in terms of interfaces, functions and sequence, and are certified according to "PLCopen Compliance Procedure for Motion Control Library V1.1".

References

More detailed information on technology objects and the effect of PLCopen block input parameters on these TOs can be found in the following Function Manuals:

- TO Axis Electric / Hydraulic, External Encoder
- Technology Objects Synchronous Operation, Cam

Description

2

2.1 Description of PLCopen blocks

In SIMOTION, the following list of blocks, which are certified according to "PLCopen Compliance Procedure for Motion Control Library V1.1", can be used in cyclic programs/tasks.

If preferred, they can be used in the LAD/FBD programming language. PLCopen blocks are available as standard functions (directly from the command library).

Table 2- 1 SingleAxis functions for the axis

Function	Description
_MC_Power()	Enabling/disabling axis
_MC_Stop()	Stopping the axis
_MC_Home()	Homing axis/clearing absolute value encoder offset
_MC_MoveAbsolute()	Absolutely positioning axis
_MC_MoveRelative()	Relatively positioning axis
_MC_MoveVelocity()	Traversing axis at defined velocity
_MC_MoveAdditive()	Positioning relative to current target position (traversing axis using an additional, defined path, relative to current position setpoint)
_MC_MoveSuperimposed()	Superimposed positioning (traversing axis relative to current motion)
_MC_PositionProfile()	Traveling through position/time profile (traversing axis along a predefined, fixed position/time profile)
_MC_VelocityProfile()	Traveling through velocity/time profile (traversing axis along a predefined, fixed velocity/time profile)
Basic functions	
_MC_Reset()	Resetting errors/alarms on the axis or triggering a restart
_MC_ReadActualPosition()	Reading the actual position of axis
_MC_ReadStatus()	Reading the status of an axis
_MC_ReadAxisError()	Reading the error of an axis
_MC_ReadParameter()	Reading axis parameter and outputting in data type LREAL
_MC_ReadBoolParameter()	Reading axis parameter and outputting in data type BOOL
_MC_WriteParameter()	Writing axis parameter of data type LREAL
_MC_WriteBoolParameter()	Writing axis parameter of data type BOOL
Apart from the standard PLCopen functions, the following additional standard axis function is included:	
_MC_Jog()	Continuous or incremental jogging

Table 2- 2 MultiAxis functions for the axis

Function	Description
_MC_GearIn()	Starting gearing (synchronizing master and slave axis while taking into account a positional relationship described by a fixed gear ratio)
_MC_GearOut()	Terminating gearing (desynchronizing master and slave axis)
_MC_CamIn()	Starting camming (synchronizing master and slave axis while taking into account a positional relationship described by a cam)
_MC_CamOut()	Terminating camming (desynchronizing master and slave axis)
_MC_Phasing()	Changing phase shift between the leading axis and following axis

Table 2- 3 Functions for external encoder

Function	Description
_MC_Power()	Enabling external encoder
_MC_Reset()	Resetting external encoder
_MC_Home()	Homing external encoder
_MC_ReadActualPosition()	Reading actual position of external encoder
_MC_ReadStatus()	Reading external encoder status
_MC_ReadAxisError()	Reading external encoder error
_MC_ReadParameter()	Reading external encoder parameter and outputting in data type LREAL
_MC_ReadBoolParameter()	Reading external encoder parameter and outputting in data type BOOL

Other blocks

On the "Utilities & Applications" CD (included in SIMOTION SCOUT scope of delivery) you will also find a versatile block that can be used for axis control purposes (under Applications > Converting > Versatile Block for Axis Control).

The **FBLineAxis** function block serves a wide range of application cases and is, therefore, classified as versatile. It enables the user to not only control individual axis motions, but also define axis groups using master/slave relationships, e.g. as a setpoint cascade with closed-loop position-controlled axes.

Note

No liability assumed when using FBLineAxis

Siemens AG assumes no liability for any problems or errors which may arise from using FBLineAxis.

Blocks

3

3.1 Handling PLCopen blocks

3.1.1 Block instances

In order to use PLCopen blocks, you must create one block instance for each being used.

The data type of the instance corresponds to the block name.

Example

Below you can see how a block instance is structured, using the _MC_MoveAbsolute block in an ST program by way of example. Optional parameters are converted to comments.

Table 3- 1 Block instance for _MC_MoveAbsolute

```
VAR
myinst_moveAbsolute : _mc_moveabsolute;
END_VAR

myinst_moveAbsolute(
axis := Hour
// ,execute := 0
// ,position := 0.0
// ,velocity := -1.0
// ,acceleration := -1.0
// ,deceleration := -1.0
// ,jerk := -1.0
// ,direction := USER_DEFAULT
// ,done =>
// ,busy =>
// ,active =>
// ,commandaborted =>
// ,error =>
// ,errorid =>
);
```

3.1.2 Upgrading a PLCopen library from V2.3 to V4.1

Up until SIMOTION SCOUT V3.2 SP1, the PLCopen blocks formed part of the SIMOTION Function Library (provided on a separate CD).

With SIMOTION SCOUT V4.0 and higher, the PLCopen blocks are included in the SIMOTION SCOUT command library as standard functions ("PLCopen" command group). As part of this, they have been modified and considerably extended (to include PLCopen multiaxis functions, for example).

NOTICE

Incompatibility errors during compilation

After upgrading a PLCopen library from V3.2 to V4.1, a LAD/FBD incompatibility error occurs if you have integrated the new library using a namespace. The operations with the parameters can no longer be compiled without errors.

As an example, the namespace in this case should be called "L_SAxis".
During compilation, the relevant PLCopen blocks are downloaded to the namespace using the USELIB L_SAxis AS ns_L_SAxis operation. Therefore, all of the corresponding data types are also in this namespace. Versions of SIMOTION SCOUT up to V4.0 were fault-tolerant in the event of a faulty operation occurring without a namespace having been specified (e.g. Direction := POSITIVE_DIRECTION). However, SIMOTION SCOUT V4.1 and higher is no longer fault-tolerant and produces a compilation error.

Manual intervention is required when this occurs: **all** incomplete operations must be completed using the namespace extension "ns_L_SAxis".
For V4.1, the correct operation for the example above must be as follows:
Direction := ns_L_SAxis.POSITIVE_DIRECTION or
Direction := ns_L_SAxis.mc_direction#POSITIVE_DIRECTION

3.2 SingleAxis

3.2.1 _MC_Power - Enabling/disabling axis

3.2.1.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Example

3.2.1.2 Schematic diagram

Schematic diagram

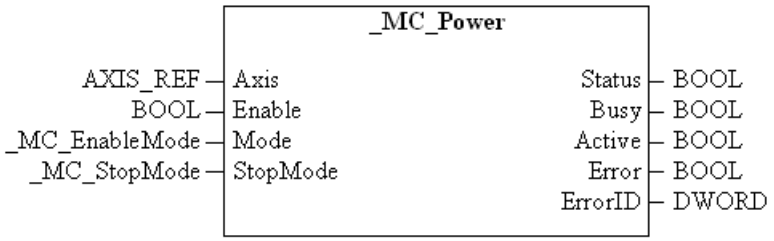


Figure 3-1 _MC_Power Schematic diagram

3.2.1.3 Purpose

Purpose

The **_MC_Power** function block enables or disables an Axis technology object or an External Encoder technology object.

If an active braking is possible before an axis is disabled, it is stopped with the set stop mode. The parameter **StopMode** specifies a stop mode for the axis. The stop mode is taken over with a falling edge at the **Enable** input.

3.2.1.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

External Encoders

3.2.1.5 Requirements

Requirements

No alarms preventing the enable may be present on the TO.

3.2.1.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Enable	BOOL	FALSE	Function block enable The axis is enabled with a rising edge at this input. If this is not possible, the attempt to set the enables is repeated as long as **Enable** is set. The axis is stopped with a falling edge at this input. The axis is disabled after reaching standstill.

Parameter	Data type	Initial value	Description
Mode	_MC_EnableMode	ALL	Specification of the axis enables to be set Not effective for external encoders. ALL: Set all enables and deactivate follow-up mode DRIVE: Only set drive enable
StopMode	_MC_StopMode	WITH_COMMAND_VALUE_ZERO	Specification of the stop mode Not effective for external encoders. WITH_COMMAND_VALUE_ZERO: The axis is stopped with emergency stop in the STOP_WITH_COMMAND_VALUE_ZERO mode. It is stopped via the emergency stop ramp in the controller. The ramp is set during axis configuration. WITH_MAXIMAL_DECELERATION: The axis is stopped with emergency stop in the STOP_WITH_MAXIMAL_DECELERATION mode. It is stopped according to interpolation with the maximum dynamic values of the axis. IN_DEFINED_TIME: The axis is stopped with emergency stop in the STOP_IN_DEFINED_TIME mode. The default setting for the stop time defined in the system variable userDefaultDynamics.stopTime is used. The specified time is adhered to irrespective of the starting velocity. DISABLE_DRIVE_IMMEDIATELY: The POWER enable is removed directly from the axis. The drive coasts to a standstill.

3.2.1.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Status	BOOL	FALSE	Display of the enable status of the axis With TRUE, the enables of the axis are set according to the **Mode** parameter. With TRUE, the enables of the external encoder are set. With FALSE, an individual enable or all enables are reset.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, a command is being processed by the command processing, i.e. the function block has active control of the axis.

Parameter	Data type	Initial value	Description
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred with the function block. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 22)).

3.2.1.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.1.9 Example

The axis is traversed with velocity 50 after the enable. After an error has occurred resulting in the removal of the axis enable, the error is corrected with **_MC_Reset**. The axis is enabled again. The axis is then traversed again with velocity 50 and the enable is removed. The axis brakes with the ramp set in the **StopMode** parameter.

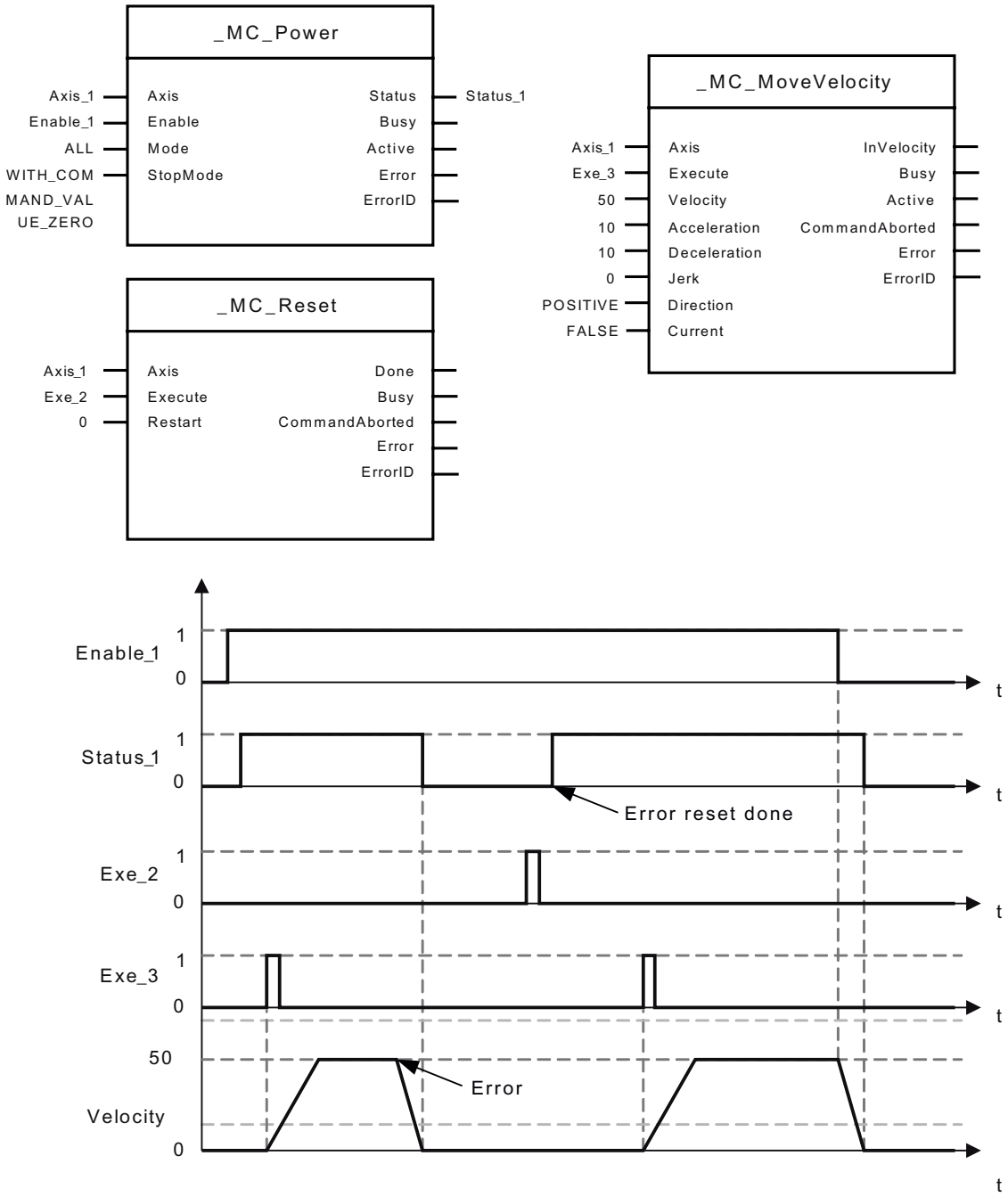


Figure 3-2 _MC_Power Example

3.2.2 _MC_Stop - Stopping axis

3.2.2.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Example

3.2.2.2 Schematic diagram

Schematic diagram

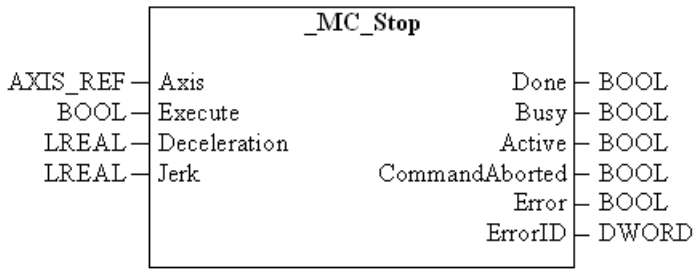


Figure 3-3 _MC_Stop Schematic diagram

3.2.2.3 Purpose

Purpose

The function block **_MC_Stop** terminates all active motion commands on an axis and decelerates it down to standstill. The function block can be overrided or aborted by another motion command, i.e. it is not possible to start a motion command.

The function block is terminated (**Done** equals **TRUE**) when the axis is stationary and the input **Execute** is reset to **FALSE**. It is then possible again to start a motion command on the axis.

The input parameters **Deceleration** and **Jerk** define the dynamic response of the stop procedure.

3.2.2.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

3.2.2.5 Requirements

Requirements

Axis enabled and not in follow-up mode

3.2.2.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The axis stops with a rising edge at this input.

Parameter	Data type	Initial value	Description
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

⚠WARNING

If the function block **_MC_Stop** is started with a jerk specification not equal to zero during the acceleration phase of an axis, its velocity can increase to the configured maximum velocity of the drive in an extreme situation. The axis is only decelerated once acceleration has been reduced by the jerk.

3.2.2.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block The axis is at standstill and the disable for the motion commands has been removed (**Execute** equals **FALSE**). This output is only set for one cycle.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.

Parameter	Data type	Initial value	Description
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by a stop response with the same or higher priority. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, **an error** has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 27)).

3.2.2.8 ErrorIDs

ErrorIDs

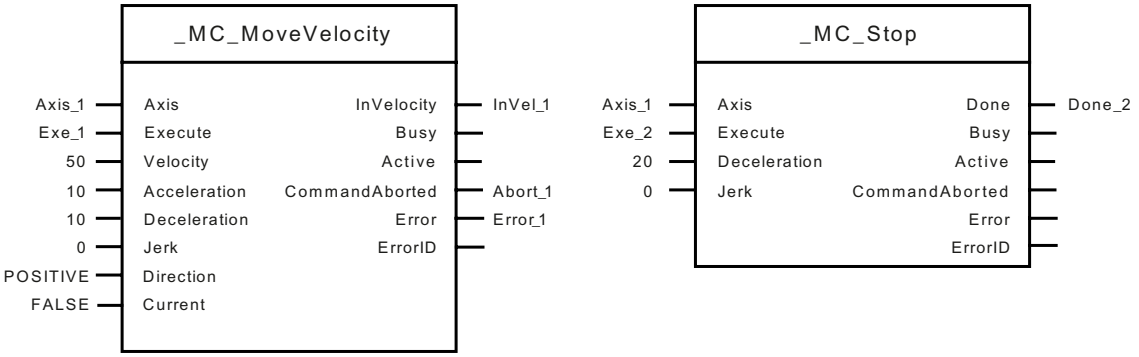
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.2.9 Example

The axis is started with velocity 50 and then decelerated to velocity 0 with **_MC_Stop**. A further call of **_MC_MoveVelocity** results in an error as the **Execute** input is set on the **_MC_Stop**. After the **Execute** input on the **_MC_Stop** is reset, the **_MC_MoveVelocity** can be executed again.



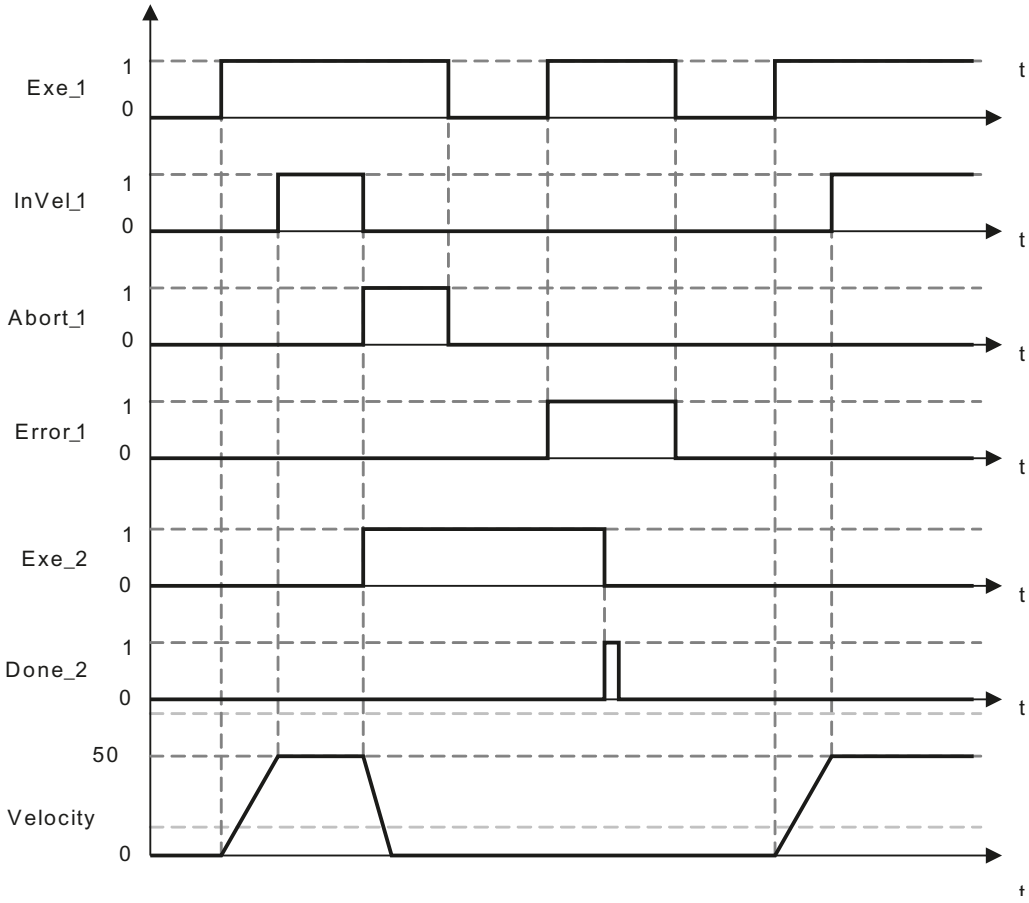


Figure 3-4 _MC_Stop Example

3.2.3 _MC_Home - Homing axis/clearing absolute value encoder offset

3.2.3.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Examples

3.2.3.2 Schematic diagram

Schematic diagram

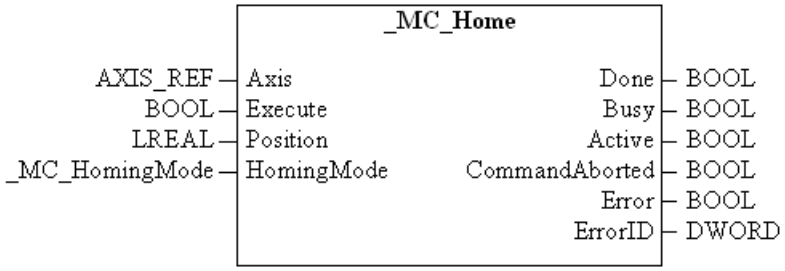


Figure 3-5 _MC_Home Schematic diagram

3.2.3.3 Purpose

Purpose

The function block **_MC_Home** establishes a positional relationship between the control and the mechanical system via a measuring system.

The following are supported:

- Active and passive homing of an axis

  The homing mode is defined at the function block. Parameters for the other data are set in the "Homing" axis dialog.
- Setting a position value
- Relative correction/shift of the actual value
- Absolute encoder adjustment

3.2.3.4 Applicable for

Applications

Positioning axes

Following axes

Path axes

External encoders

Restraints

The parameter **HomingMode** of the function block **_MC_Home** only defines the homing mode. The homing procedure itself is performed in accordance with the configuration of the encoder on the axis.

Homing mode	Virtual axis	Real axis with incremental encoder	Real axis with absolute encoder	External Encoders
ACTIVE_HOMING		X [1)]		
PASSIVE_HOMING		X		X
DIRECT_HOMING	X	X	X	X
DIRECT_HOMING_ RELATIVE	X	X	X	X
ENABLE_OFFSET_OF_ ABSOLUTE_ENCODER			X [2)]	X [2)]

1) If homing mode **MODE_NO_REFERENCE**is set in the configuration of the **TypeOfAxis.NumberOfEncoders.Encoder_<n>.IncHomingEncoder.HomingMode** encoder, then the homing mode **ACTIVE_HOMING** is not possible at the function block.

2) In homing mode **ENABLE_OFFSET_OF_ABSOLUTE_ENCODER**, the value transferred with input parameter **Position** is not effective.
For this mode you must enter the required offset in the encoder configuration before calling the **_MC_Home** function block. Enter the offset value in the configuration data under **TypeOfAxis.NumberOfEncoders.Encoder_<n>.absHomingEncoder.absShift** and select **absolute** or **relative**.
If **absolute** is selected, the axis is set to the specified offset value. If **relative** is selected, the specified offset value is added to the current axis position and the axis is set to this "total" value.

3.2.3.5 Requirements

Requirements

Enabling axes or external encoders

3.2.3.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Execute	BOOL	FALSE	Function block enable The homing procedure starts with a rising edge at this input.
Position	LREAL	0.0	Specification of the position at the reference point or the position setting value or the position offset value
HomingMode	_MC_HomingMode	ACTIVE_HOMING	Specification of the homing mode: ACTIVE_HOMING: Active homing PASSIVE_HOMING: On-the-fly homing DIRECT_HOMING: Setting current position as reference point DIRECT_HOMING_RELATIVE: Direct homing (the actual position of the axis is added to the value specified in the parameter **Position** as position difference). ENABLE_OFFSET_OF_ABSOLUTE_ENCODER: Absolute encoder adjustment

3.2.3.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the programmed target position has been reached.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 32)).

3.2.3.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.3.9 Examples

Example: Active homing

Sequence for active homing:

- Phase 1:

  Approach of the home position switch (BERO).

  The axis traverses with the homing approach velocity V_{app} (approach velocity).
- Phase 2:

  Synchronization with the zero mark.

  The axis traverses with the homing reduced velocity V_{red} (reduced velocity).
- Phase 3:

  Travel to the home position coordinate.

  The axis traverses with the homing entry velocity V_{ent} (entry velocity).

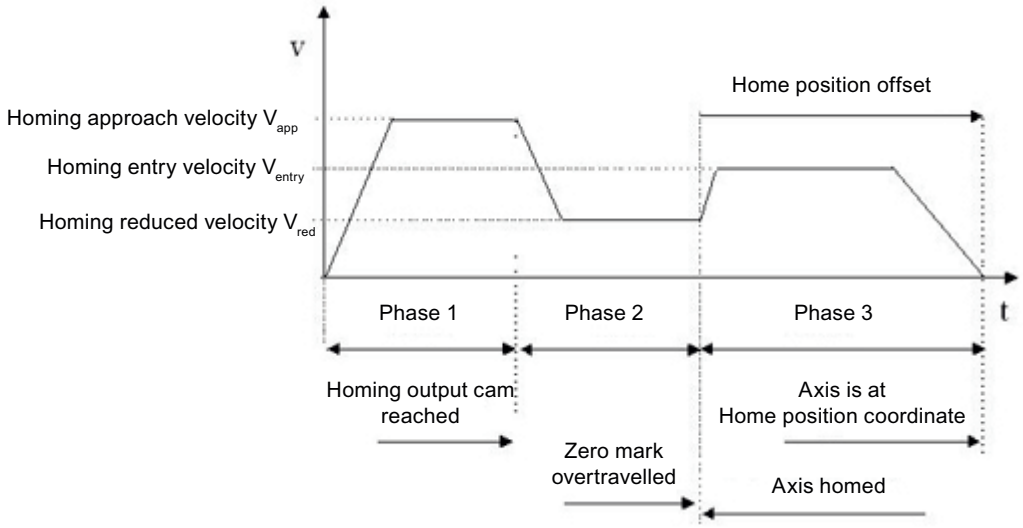


Figure 3-6 _MC_Home Example: Active homing

Example: Direct homing

The new absolute position is set in the next interpolator cycle after the call of the function block with the set Execute input. When calling the function block from a task which is not synchronous with the interpolator, the setting procedure is only recommended for axes at standstill.

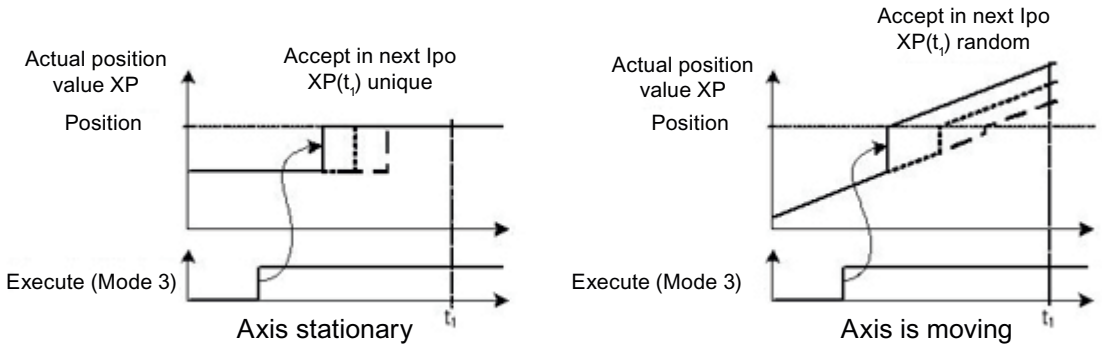


Figure 3-7 _MC_Home Example: Direct homing

Example: Passive homing

The axis is traversed at the velocity 50. Then the passive homing with zero mark is triggered. At the next zero pulse, the actual position of the axis is set to 90.

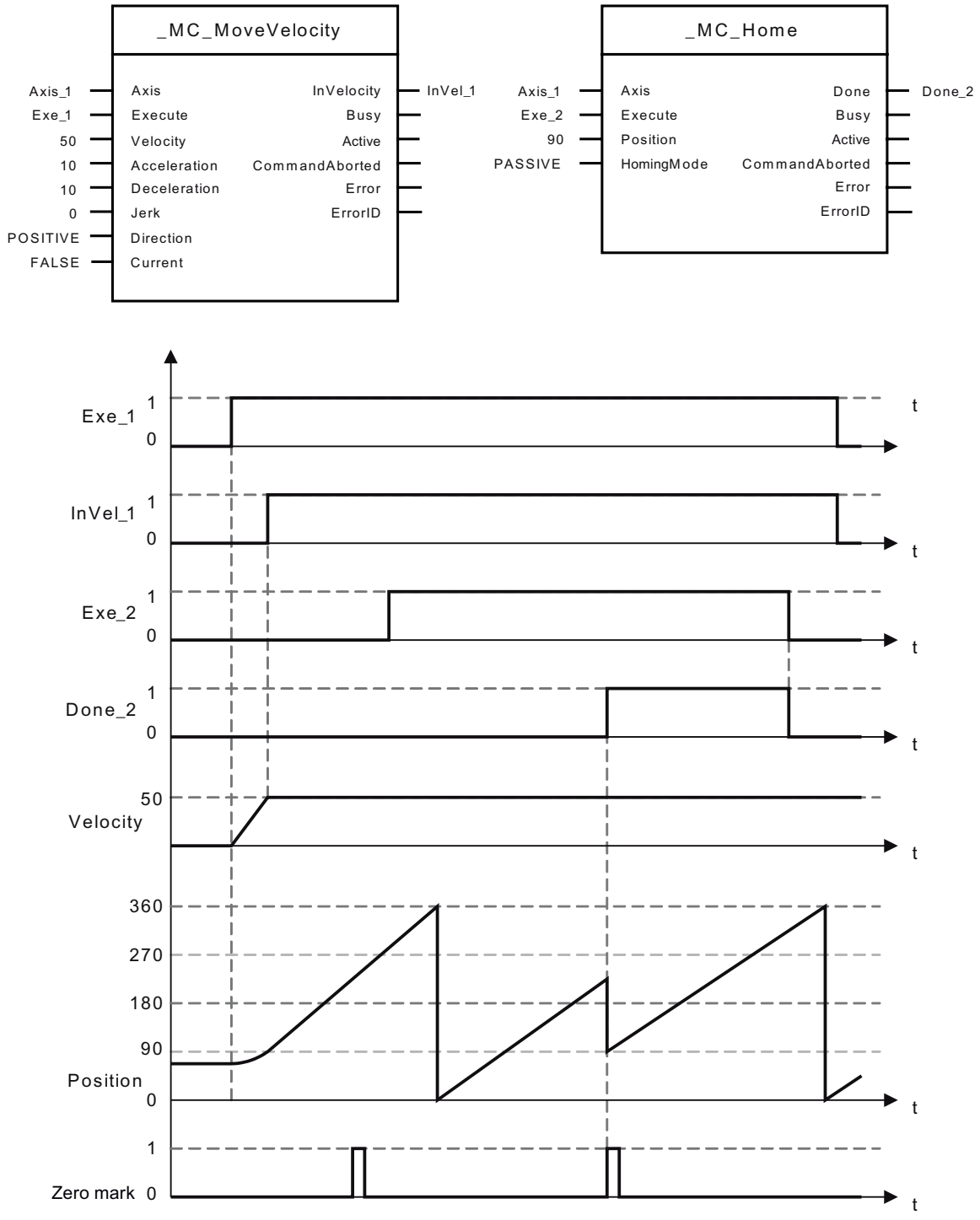


Figure 3-8 _MC_Home Example: Passive homing

3.2.4 _MC_MoveAbsolute - Absolute positioning of axis

3.2.4.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Example

3.2.4.2 Schematic diagram

Schematic diagram

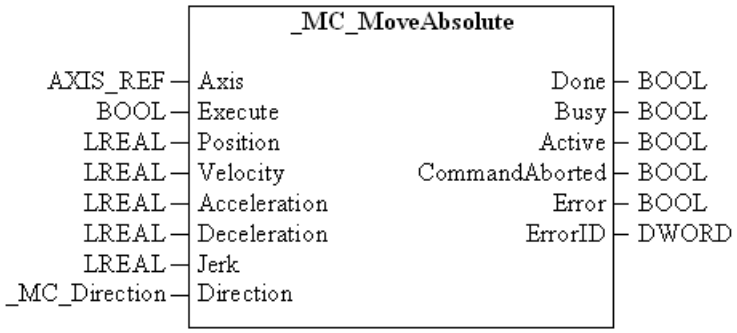


Figure 3-9 _MC_MoveAbsolute Schematic diagram

3.2.4.3 Purpose

Purpose

The function block **_MC_MoveAbsolute** starts a positioning motion of an axis to an absolute position.

The dynamic response parameters **Velocity**, **Acceleration**, **Deceleration** and **Jerk** define the dynamic response of the motion procedure.

The axis stops after completion of the positioning motion.

An active motion command is overridden by the function block.

With modulo axes, **SHORTEST_WAY** can be specified as an additional direction of motion.

3.2.4.4 Applicable for

Applications

Positioning axes

Following axes

Path axes

3.2.4.5 Requirements

Requirements

Axis enabled

Axis homed if the axis configuration data item **TypeOfAxis.Homing.referencingNecessary** was set to **YES** (homing required).

No **_MC_Stop** active

3.2.4.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The positioning operation starts with a rising edge at this input.
Position	LREAL	0.0	Specification of the absolute target position of the motion
Velocity	LREAL	-1.0	Specification of the maximum velocity The velocity is reached depending on the set values for traversing distance, acceleration and jerk. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameter	Data type	Initial value	Description
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.
Direction	_MC_Direction	USER_DEFAULT	Specification of direction of motion for modulo axes: USER_DEFAULT: Default value from axis configuration POSITIVE: Direction of rotation/motion positive SHORTEST_WAY: In direction of shortest distance [1)] NEGATIVE: Direction of rotation/motion negative EFFECTIVE: Last programmed direction of rotation/motion

[1)] For modulo axes only

3.2.4.7 Output parameter

Output parameters

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the programmed target position has been reached.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 39)).

3.2.4.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.4.9 Example

Case A:

Two _MC_MoveAbsolute blocks are started in succession.

Case B:

The second _MC_MoveAbsolute cancels the first _MC_MoveAbsolute block. The target position results relatively from the position at the start of the second block.

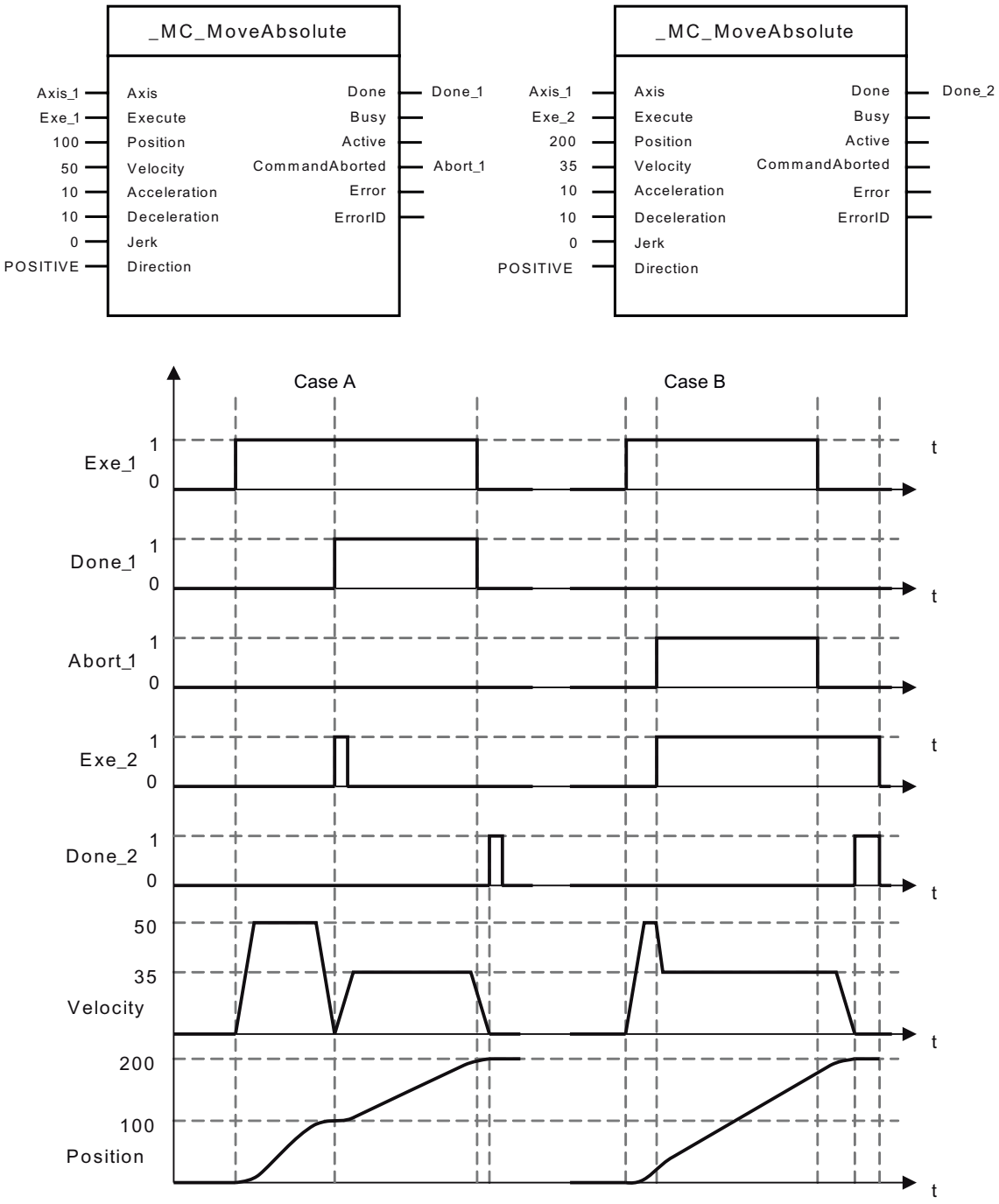


Figure 3-10 _MC_MoveAbsolute Example

3.2.5 _MC_MoveRelative - Relative positioning of axis

3.2.5.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.2.5.2 Schematic diagram

Schematic diagram

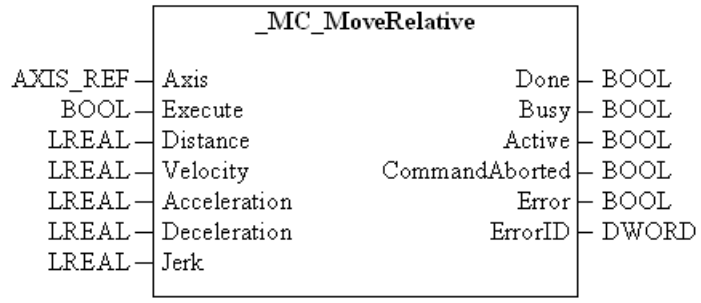


Figure 3-11 _MC_MoveRelative Schematic diagram

3.2.5.3 Purpose

Purpose

The function block **_MC_MoveRelative** positions an axis relative to the actual position of the axis. If the axis is already in motion when the job is started, the position that is present in the system at the start of the job processing is used as the start position. It must be taken into account that there is a response time between the processing of the function block and the execution of the motion, which depends on the user task in which the function block was programmed and on the set interpolation cycle clock.

The dynamic response parameters **Velocity**, **Acceleration**, **Deceleration** and **Jerk** define the dynamic response of the motion procedure.

The axis stops after completion of the positioning motion.

An active motion command is overridden by the function block.

3.2.5.4 Applicable for

Application

Positioning axes

Following axes

Path axes

3.2.5.5 Requirements

Requirements

Axis enabled

No **_MC_Stop** active

3.2.5.6 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The positioning operation starts with a rising edge at this input.
Distance	LREAL	0.0	Specification of the distance difference to be traversed
Velocity	LREAL	-1.0	Specification of the maximum velocity The velocity is reached depending on the set values for traversing distance, acceleration and jerk. Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameters	Data type	Initial value	Description
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

3.2.5.7 Output parameter

Output parameters

Parameters	Data type	Initial value	Description
Done	BOOL	FALSE	Indicates completion of the function block With TRUE, the programmed target position has been reached.
Busy	BOOL	FALSE	Indicates function block activity With TRUE, the function block has been started.
Active	BOOL	FALSE	Indicates command activity in the function block With TRUE, the command is being processed by the command processing function, i.e., the function block has active control over the axis.

Parameters	Data type	Initial value	Description
CommandAborted	BOOL	FALSE	Indicates that the function block has been aborted With TRUE, the function block has been aborted because of an error during command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Indicates an error in the function block With TRUE, an error has occurred during initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Indicates a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 44)).

3.2.5.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.5.9 Example

Case A:

Two _MC_MoveRelative blocks are started in succession.

Case B:

The second _MC_MoveRelative cancels the first _MC_MoveRelative block. The target position results relatively from the position at the start of the second block.

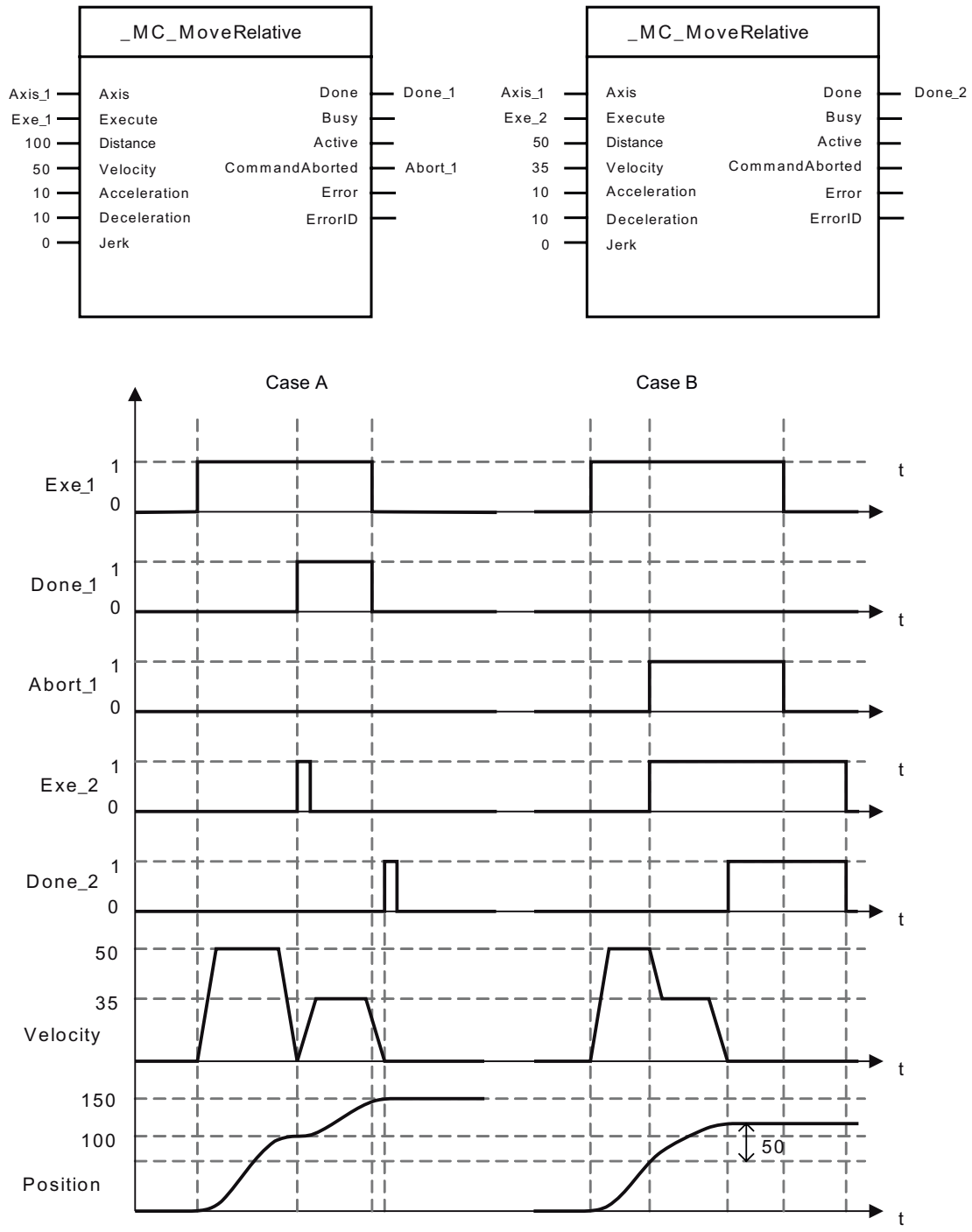


Figure 3-12 _MC_MoveRelative Example

3.2.6 _MC_MoveVelocity - Traversing axis at a specified velocity

3.2.6.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.2.6.2 Schematic diagram

Schematic diagram

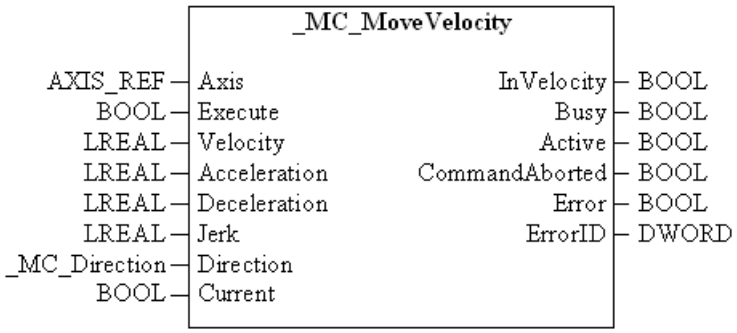


Figure 3-13 _MC_MoveAdditive Schematic diagram

3.2.6.3 Purpose

Purpose

The technology function **_MC_MoveVelocity** accelerates or decelerates an axis to a set velocity.

The dynamic response parameters **Acceleration**, **Deceleration** and **Jerk** define the dynamic response of the motion procedure.

If a velocity override is in effect, then the end velocity is calculated under consideration of the override. You must take this behavior into account in the user program.

3.2.6.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

3.2.6.5 Requirements

Requirements

Axis enabled

No **_MC_Stop** active

3.2.6.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The axis accelerates or decelerates to the programmed set velocity with a rising edge at this input.
Velocity	LREAL	-1.0	Specification of the set velocity Value ≥ 0: The specified value is used. Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameter	Data type	Initial value	Description
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.
Direction	_MC_Direction	USER_DEFAULT	Specification of the direction of motion: USER_DEFAULT: Default value from axis configuration POSITIVE: Direction of rotation/motion positive NEGATIVE: Direction of rotation/motion negative EFFECTIVE: Last programmed direction of rotation/motion
Current	BOOL	FALSE	Type of velocity specification With TRUE, the actual velocity of the axis is taken over as programmed set velocity. With FALSE, the set velocity programmed on the **Velocity** input is used.

3.2.6.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
InVelocity	BOOL	FALSE	Indicates termination of the function block. With TRUE, the axis has reached the programmed setpoint velocity. Until the abort of the function block, the output remains unchanged irrespective of the subsequent characteristic of the axis velocity.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.

Parameter	Data type	Initial value	Description
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 49)).

Note

When speed setpoint = zero (Velocity = 0.0), the following applies:

InVelocity is set when the axis has reached a standstill, and remains set as long as **Execute = 1**.

As soon as **InVelocity** is set, the command is terminated, i.e. **Busy = FALSE**. This means that neither **CommandAborted** nor **Error** can be signaled by the function block.

3.2.6.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

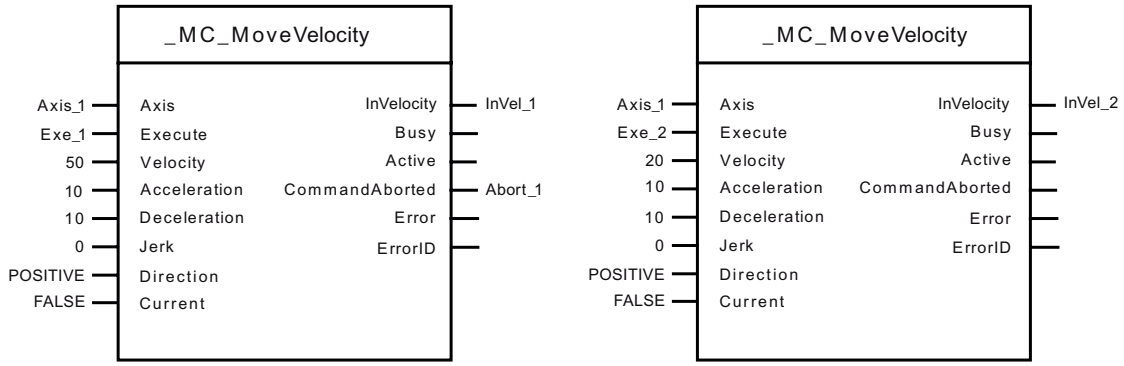
3.2.6.9 Example

Case A:

An _MC_MoveVelocity block is overridden by another _MC_MoveVelocity block after the end velocity has been reached (**InVelocity = TRUE**).

Case B:

An _MC_MoveVelocity block is overridden by another _MC_MoveVelocity block before the end velocity has been reached (**InVelocity = FALSE**).



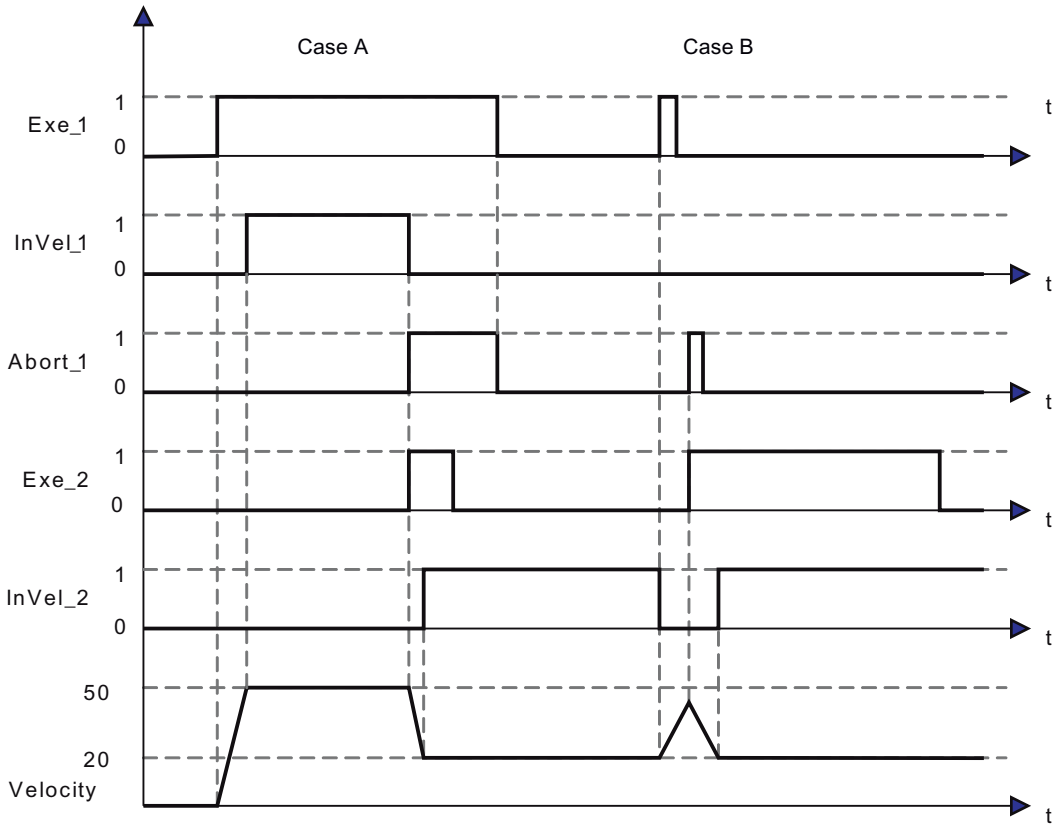


Figure 3-14 _MC_MoveVelocity Example

3.2.7 _MC_MoveAdditive - Relative positioning to current target position

3.2.7.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.2.7.2 Schematic diagram

Schematic diagram

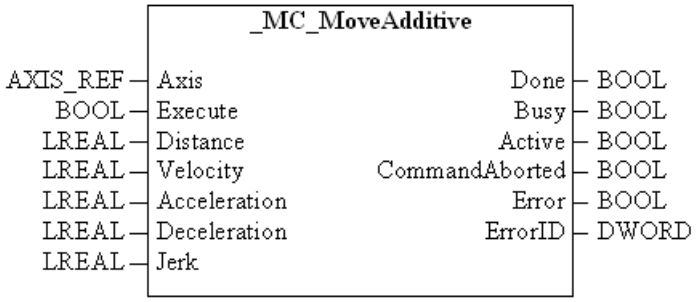


Figure 3-15 _MC_MoveAdditive Schematic diagram

3.2.7.3 Purpose

Purpose

The function block **_MC_MoveAdditive** positions an axis relative to the target position of the active positioning command. The function block enables a correction of the target position of the previous positioning command by a distance specified at the Distance input.

The dynamic response parameters **Velocity**, **Acceleration**, **Deceleration** and **Jerk** define the dynamic response of the motion procedure.

The axis stops after completion of the positioning motion.

An active motion command is overridden by the function block.

3.2.7.4 Applicable for

Applications

Positioning axes

Following axes

Path axes

3.2.7.5 Requirements

Requirements

Axis enabled

No **_MC_Stop** active

The axis must be homed if the TypeOfAxis.Homing.referencingNecessary axis configuration data item has been set to YES (homing required), and

- the axis is in motion, or
- **_MC_MoveAdditive** overrides an active motion command (exception: **_MC_MoveVelocity**).

Note

The function block **_MC_MoveAdditive** behaves like a **_MC_MoveRelative** function block, if

- The axis is stationary at the start of the job or
- An active motion command without defined target position is overridden by the function block. The target position then depends on the position of the axis at the time of the override and the additional distance to be traversed.

3.2.7.6 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The positioning operation starts with a rising edge at this input.
Distance	LREAL	0.0	Specification of the additional distance difference to be traversed
Velocity	LREAL	-1.0	Specification of the maximum velocity The velocity is reached depending on the set values for traversing distance, acceleration and jerk. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameters	Data type	Initial value	Description
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

3.2.7.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the axis is at the resulting position setpoint.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.

Parameter	Data type	Initial value	Description
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 55)).

3.2.7.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.7.9 Example

Case A:

Two _MC_MoveAdditive blocks are started in succession.

Case B:

The second _MC_MoveAdditive cancels the first _MC_MoveAdditive block. The target position results from the target position of the first block.

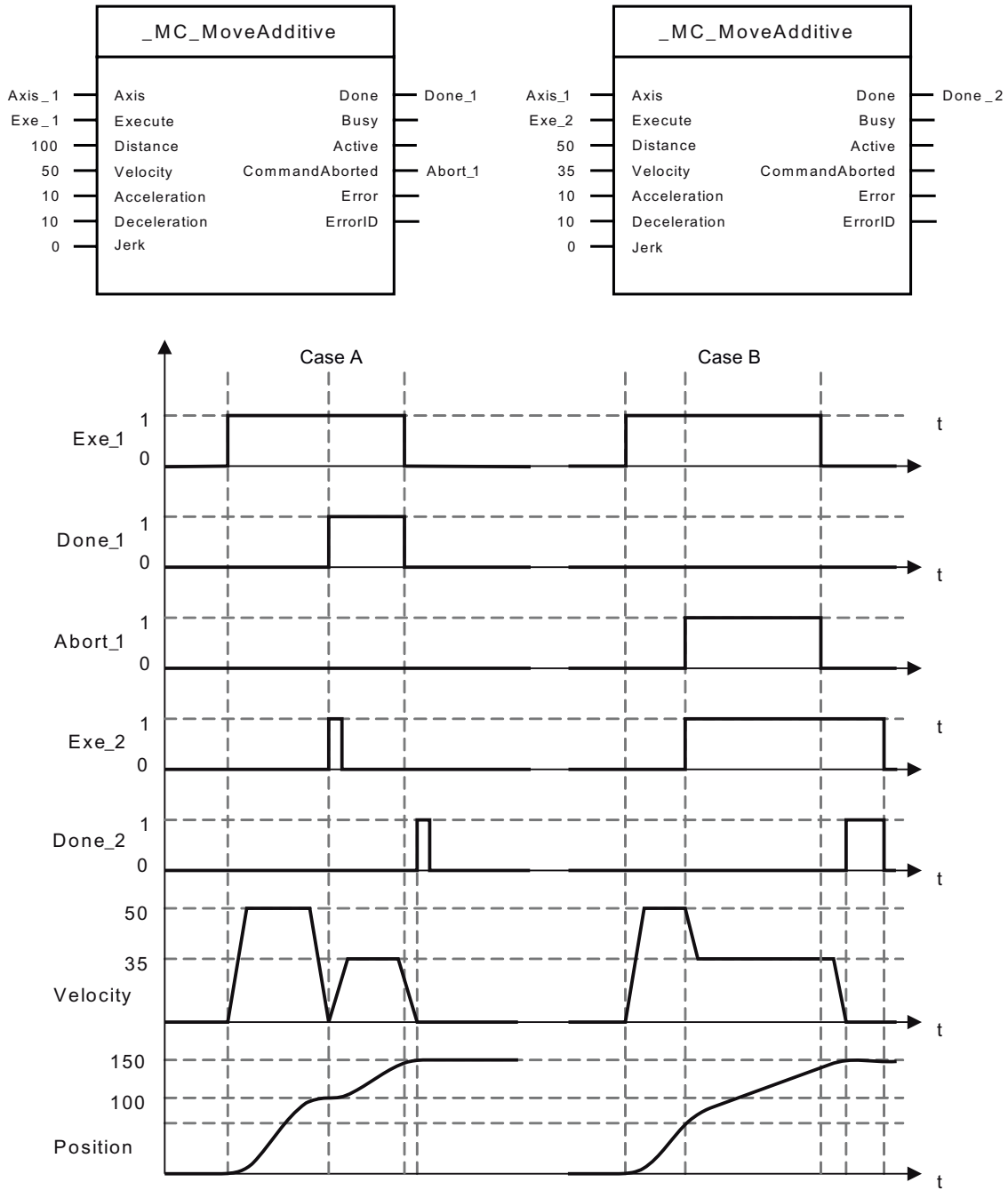


Figure 3-16 _MC_MoveAdditive Example

3.2.8 _MC_MoveSuperimposed - Superimposed positioning

3.2.8.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.2.8.2 Schematic diagram

Schematic diagram

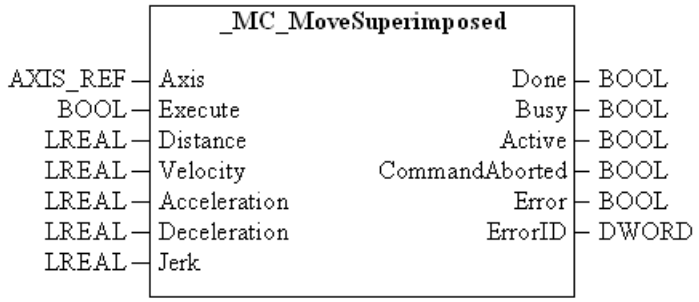


Figure 3-17 _MC_MoveSuperimposed Schematic diagram

3.2.8.3 Purpose

Purpose

The function block **_MC_MoveSuperImposed** starts a positioning motion relative to the active positioning motion of an axis. This enables a superimposed positioning of an axis, e.g., for the print-mark correction.

The dynamic response parameters **VelocityDiff**, **Acceleration**, **Deceleration** and **Jerk** define the dynamic response of the motion procedure.

An active motion command (main motion) is not overridden by the function block.

An active superimposed positioning motion is overridden by a restart of the function block **_MC_MoveSuperImposed**. The remaining distance-to-go of the overridden superimposed positioning motion is lost.

3.2.8.4 Applicable for

Applications

Positioning axes

Following axes

Path axes

3.2.8.5 Requirements

Requirements

Axis enabled

No **_MC_Stop** active

The axis velocity is increased for the superimposed positioning operation. Therefore, the basic motion of the axis should not be performed with the maximum permissible velocity.

3.2.8.6 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The positioning operation starts with a rising edge at this input.
Distance	LREAL	0.0	Specification of the distance difference to be traversed
Velocity	LREAL	-1.0	Specification of the maximum velocity The velocity is reached in relation to the set values for the traversing distance, acceleration, and jerk. Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameters	Data type	Initial value	Description
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

3.2.8.7 Output parameter

Output parameters

Parameters	Data type	Initial value	Description
Done	BOOL	FALSE	Indicates completion of the function block With TRUE, the programmed target position has been reached.
Busy	BOOL	FALSE	Indicates function block activity With TRUE, the function block has been started.
Active	BOOL	FALSE	Indicates command activity in the function block With TRUE, the command is being processed by the command processing function, i.e., the function block has active control over the axis.

Parameters	Data type	Initial value	Description
CommandAborted	BOOL	FALSE	Indicates that the function block has been aborted With TRUE, the function block has been aborted because of an error during command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Indicates an error in the function block With TRUE, an error has occurred during initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Indicates a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 60)).

3.2.8.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.8.9 Example

Case A:

An _MC_MoveSuperImposed is started during a relative positioning.

Case B:

MC_MoveSuperImposed is started again before _MC_MoveSuperImposed is completed.

Case C:

Start _MC_MoveSuperImposed with a stationary axis.

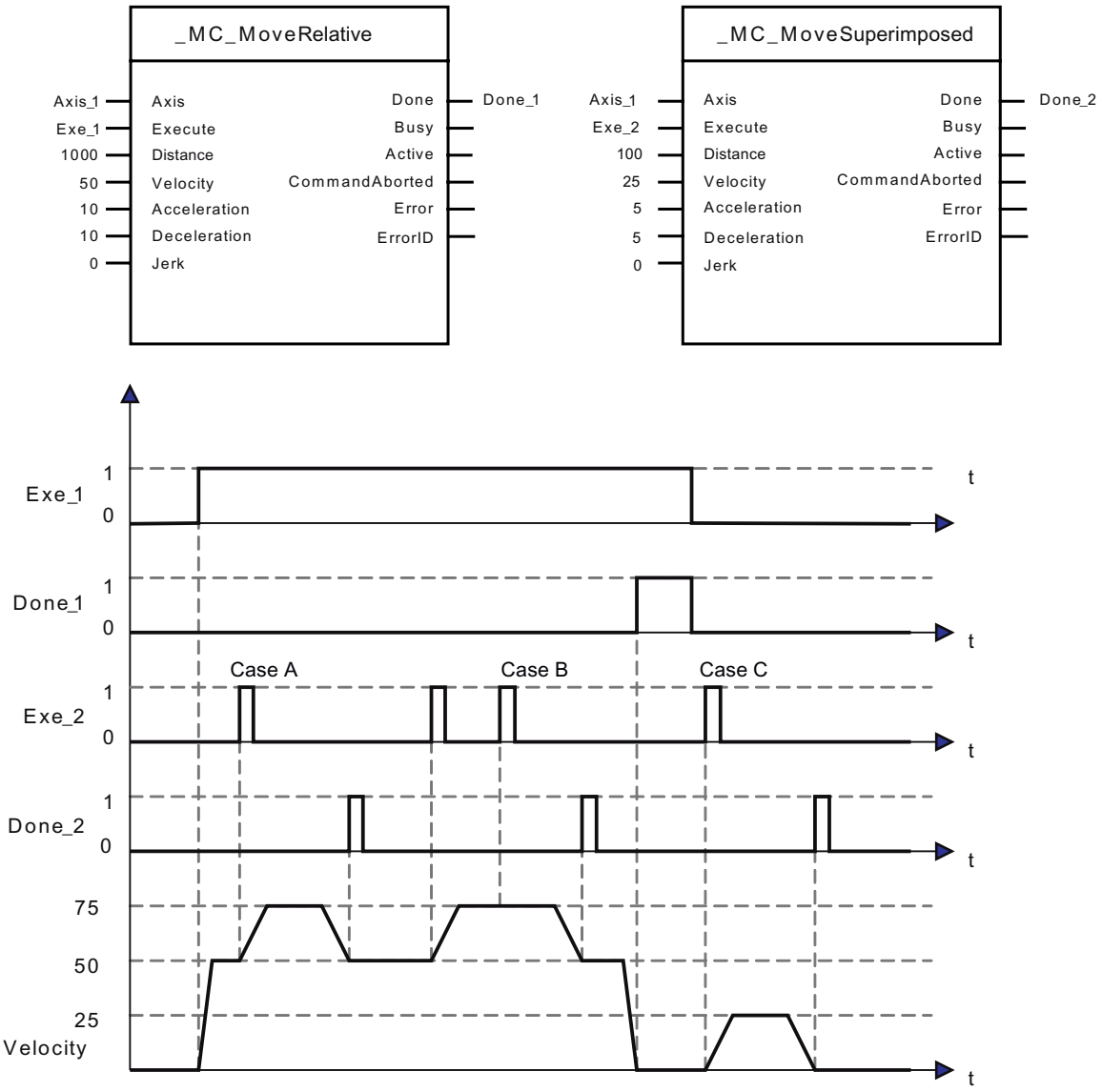


Figure 3-18 _MC_MoveSuperimposed Example

3.2.9 _MC_PositionProfile - Traveling through position/time profile

3.2.9.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Example

3.2.9.2 Schematic diagram

Schematic diagram

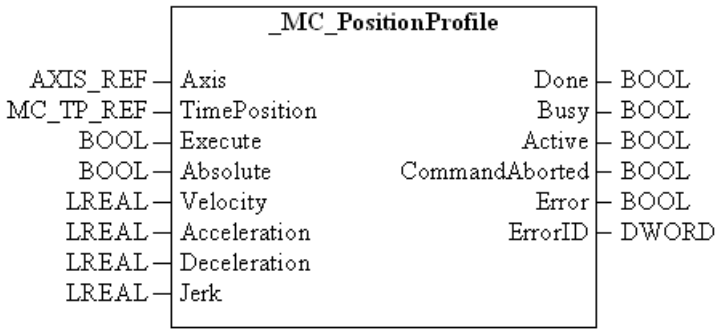


Figure 3-19 _MC_PositionProfile Schematic diagram

3.2.9.3 Purpose

Purpose

The function block **_MC_PositionProfile** traverses an axis along a position profile which is specified as an s(t) function.

3.2.9.4 Applicable for

Applications

Positioning axes
Following axes
Path axes

3.2.9.5 Requirements

Requirements

Axis enabled

Axis homed if the axis configuration data item **TypeOfAxis.Homing.referencingNecessary** was set to **YES** (homing required).

Position profiles are specified using cams and must be assigned to the axis (in the "Profiles" axis dialog).

No **_MC_Stop** active

3.2.9.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
TimePosition	MC_TP_REF	0	Specification of cam profile (name) that describes the position as a function of the time: • Cam (CamType data type)
Execute	BOOL	FALSE	Function block enable The positioning motion starts with a rising edge at this input.
Absolute	BOOL	TRUE	Specification of the traversing method With TRUE, the cam positions are approached according to the absolute values. With FALSE, the position profile of the cam is set at the actual position of the axis.
Velocity	LREAL	-1.0	Specification of the maximum velocity The velocity is reached depending on the set values for traversing distance, acceleration and jerk. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameter	Data type	Initial value	Description
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used. Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

3.2.9.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the axis has completed the specified position profile.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.

Parameter	Data type	Initial value	Description
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 65)).

3.2.9.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.9.9 Example

Case A:

Two _MC_PositionProfile blocks are started in succession.

Case B:

The second _MC_PositionProfile block aborts the motion of the first block.

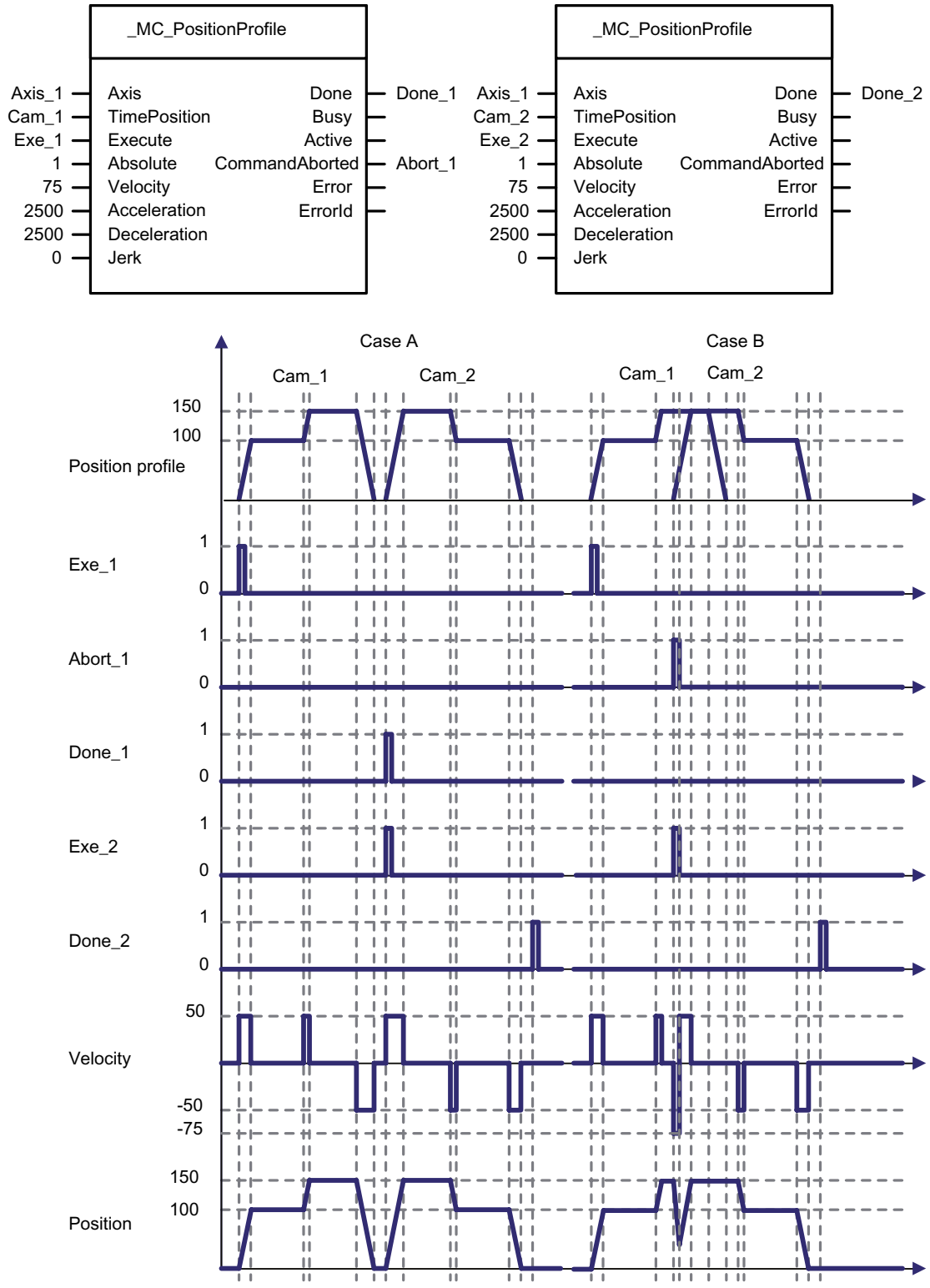


Figure 3-20 _MC_PositionProfile example

3.2.10 _MC_VelocityProfile - Traveling through velocity/time profile

3.2.10.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.2.10.2 Schematic diagram

Schematic diagram

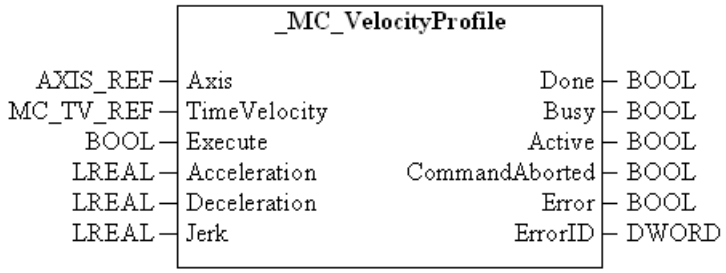


Figure 3-21 _MC_VelocityProfile Schematic diagram

3.2.10.3 Purpose

Purpose

The function block **_MC_VelocityProfile** traverses an axis along a velocity profile, which is specified as a v(t) function.

3.2.10.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

3.2.10.5 Requirements

Requirements

Axis enabled

Velocity profiles are specified using cams and must be assigned to the axis (in the "Profiles" axis dialog).

No **_MC_Stop** active

3.2.10.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
TimeVelocity	MC_TV_REF	0	Specification of cam profile (name) that describes the velocity as a function of the time: • Cam (CamType data type)
Execute	BOOL	FALSE	Function block enable The motion starts with a rising edge at this input.
Acceleration	LREAL	-1.0	Specification of the maximum acceleration (increasing energy in the motor) Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).

Parameter	Data type	Initial value	Description
Deceleration	LREAL	-1.0	Specification of the maximum deceleration (decreasing energy in the motor) Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

3.2.10.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the axis has traversed the specified velocity profile.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.

Parameter	Data type	Initial value	Description
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 70)).

3.2.10.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.10.9 Example

Case A:

Two _MC_VelocityProfile blocks are started in succession.

Case B:

The second _MC_VelocityProfile block aborts the motion of the first block.

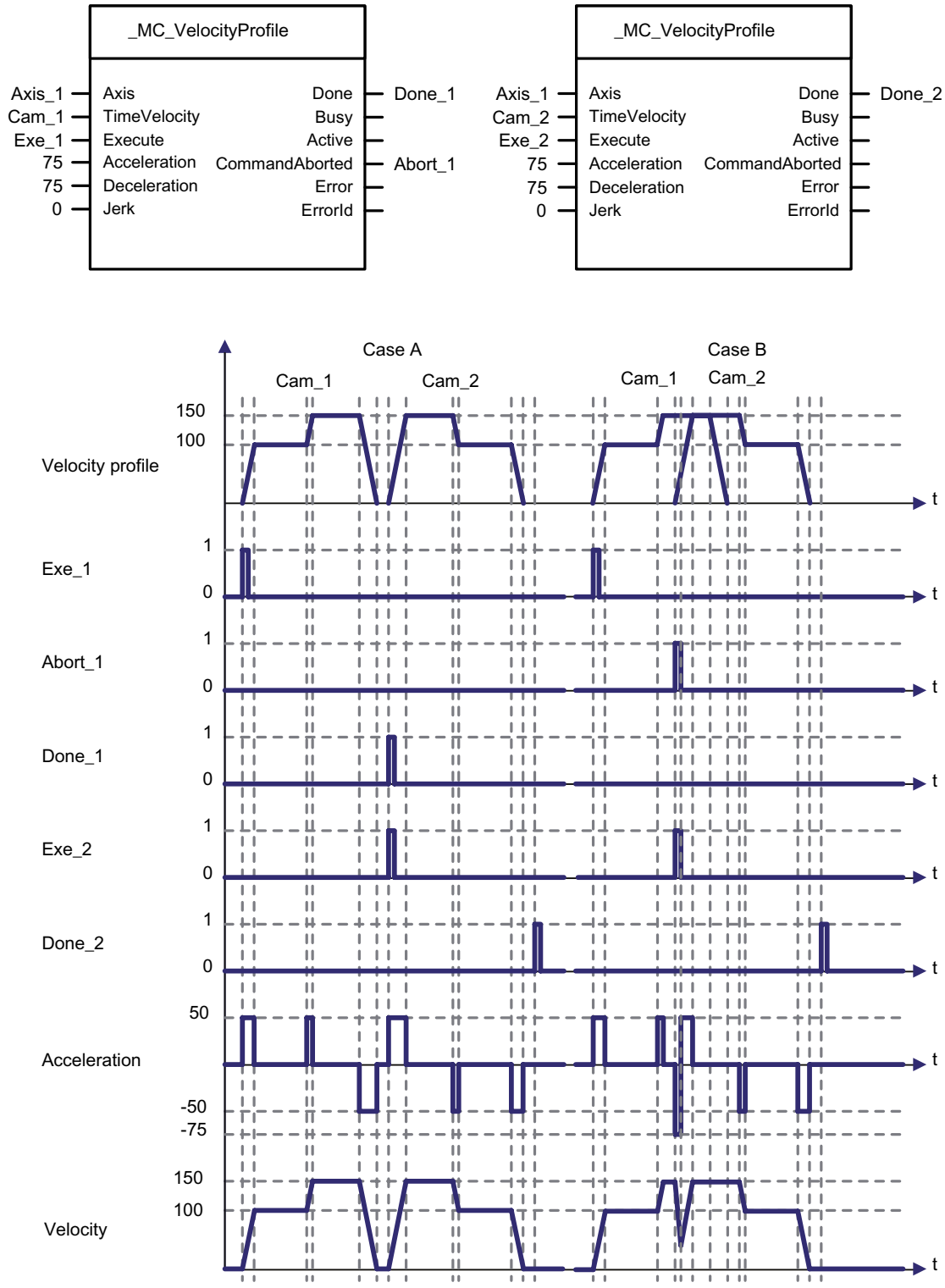


Figure 3-22 _MC_VelocityProfile example

3.2.11 _MC_Reset - Resetting errors/alarms on the axis or triggering a restart

3.2.11.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Example

3.2.11.2 Schematic diagram

Schematic diagram

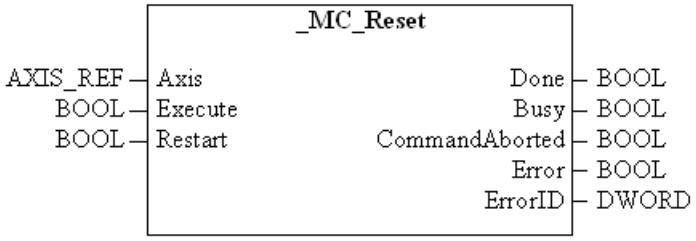


Figure 3-23 _MC_Reset Schematic diagram

3.2.11.3 Purpose

Purpose

The function block **_MC_Reset** resets all errors on an axis or external encoder that can be acknowledged by means of the software. Fatal errors can be acknowledged via Power off/on or reloading of the project to the module.

If the **Restart** input is set, the transferred technology object is re-initialized via **_MC_Reset**. Axes that are operated with incremental encoders return to the **Not homed** mode.

3.2.11.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

External Encoders

3.2.11.5 Requirements

Requirements

The restart of an axis depends on the condition set with configuration data item **restartAxisCondition**.

NOTICE
• Set the input **Restart** to **FALSE**, only when the errors present on the transferred technology object are to be acknowledged. • No variables of the technology object are updated during the restart operation!

3.2.11.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Execute	BOOL	FALSE	Function block enable With a rising edge at this input, the function block acknowledges the errors present on the transferred technology object or optionally triggers a restart.
Restart	BOOL	FALSE	Specification of the reset type With TRUE, the transferred technology object is restarted and modified configuration data are accepted. With FALSE, the alarms pending on the transferred technology object are acknowledged.

3.2.11.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block The alarms have been acknowledged or a restart performed.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error during the command processing. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 75)).

3.2.11.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.11.9 Example

Case A:

The _MC_Reset block resets active errors/alarms at the axis transferred on input parameter **Axis**.

Case B:

The _MC_Reset block triggers a restart at the axis transferred on input parameter **Axis**.

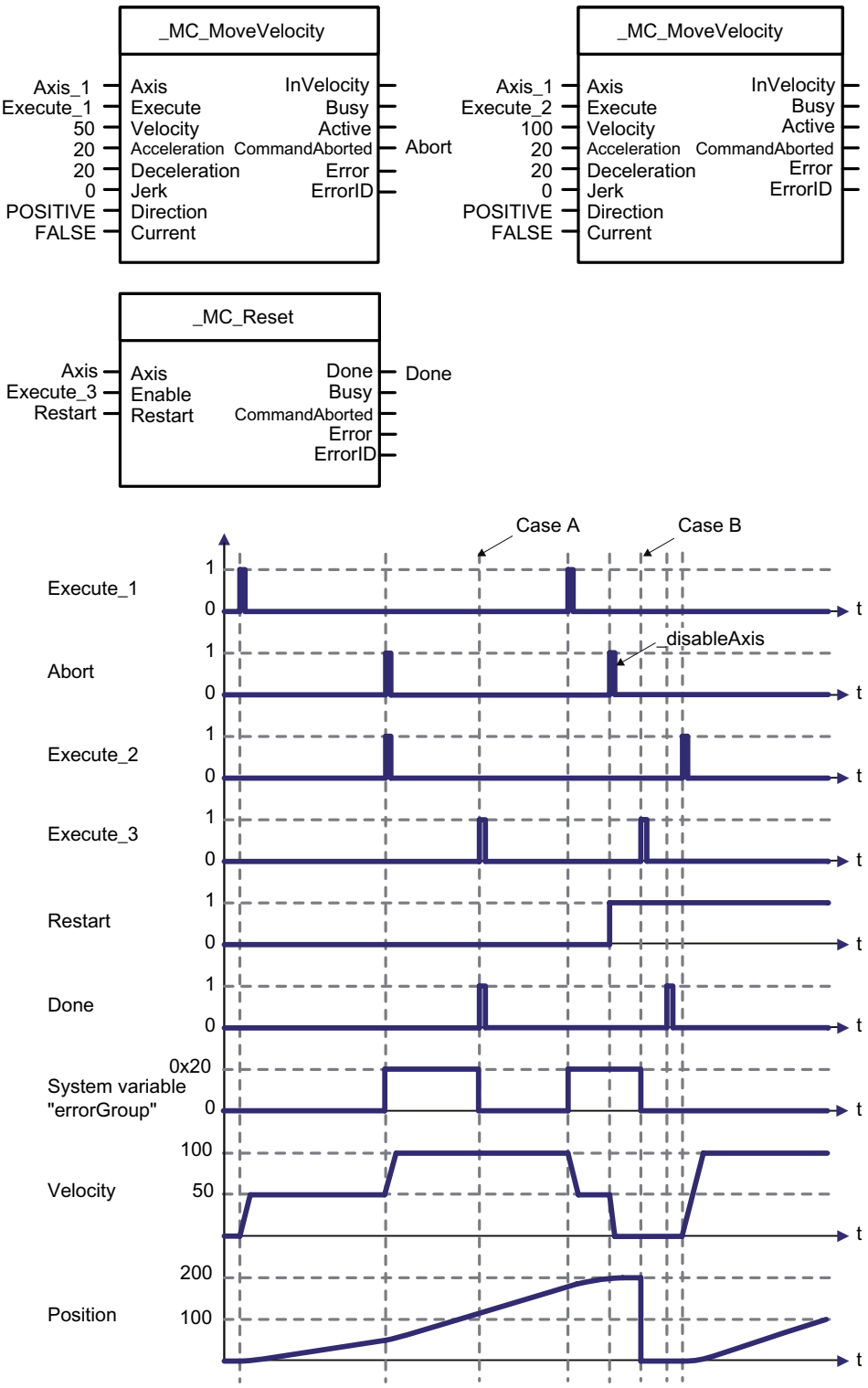


Figure 3-24 _MC_Reset example

3.2.12 _MC_ReadActualPosition - Reading actual position of axis

3.2.12.1 Overview

Schematic diagram
Purpose
Applicable for
Input parameters
Output parameters
ErrorIDs
Example

3.2.12.2 Schematic diagram

Schematic diagram

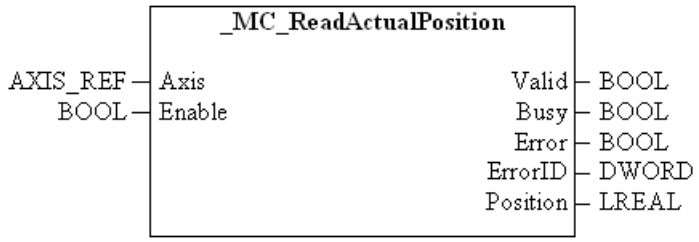


Figure 3-25 _MC_ReadActualPosition Schematic diagram

3.2.12.3 Purpose

Purpose

The function block **_MC_ReadActualPosition** reads the actual position of an axis or an external encoder.

As an output value, the block supplies:

- The value for the **positioningState.actualPosition** system variable, in the case of axes
- The value for the **motionState.position** system variable, in the case of external encoders

3.2.12.4 Applicable for

Applications

Positioning axes

Following axes

Path axes

External encoders

3.2.12.5 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Enable	BOOL	FALSE	Function block enable The value at output **Position** is updated as long as **Enable** equals **TRUE**.

3.2.12.6 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Valid	BOOL	FALSE	Display of the validity of the value which can be read at output **Position**.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the error status could not be read. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 79)).
Position	LREAL	0.0	Displays the actual position of the axis or external encoder

3.2.12.7 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.12.8 Example

The _MC_ReadActualPosition block reads the actual position of the axis transferred on the **Axis** input parameter.

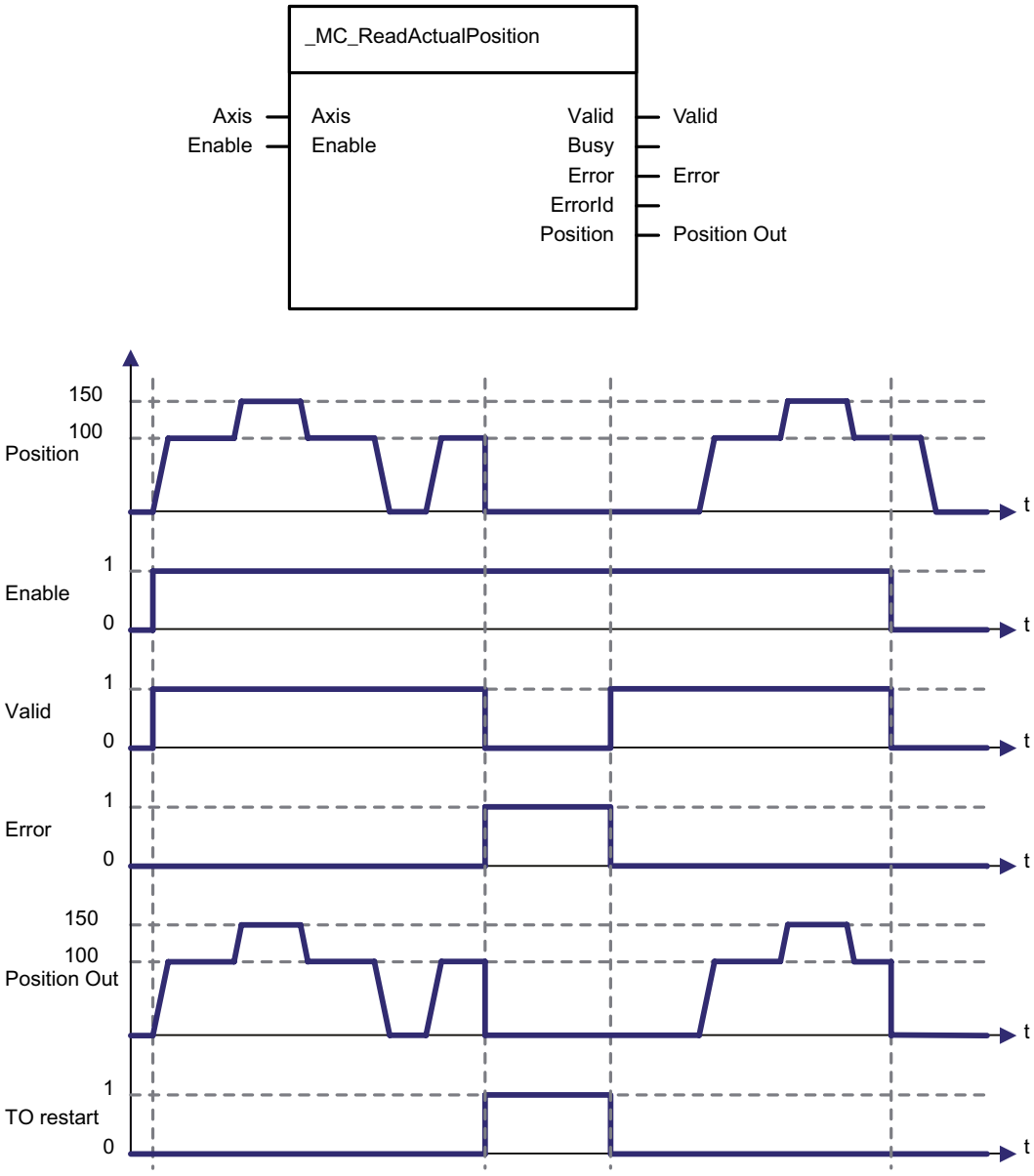


Figure 3-26 _MC_ReadActualPosition example

3.2.13 _MC_ReadStatus - Reading an axis status

3.2.13.1 Overview

Schematic diagram

Purpose

Applicable for

Input parameters

Output parameters

ErrorIDs

Example

3.2.13.2 Schematic diagram

Schematic diagram

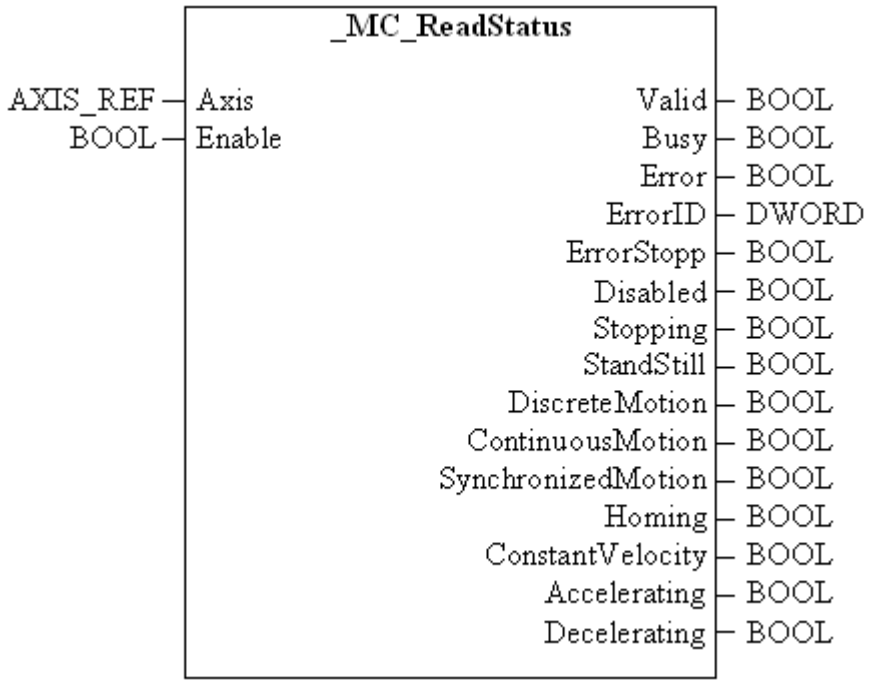


Figure 3-27 _MC_ReadStatus Schematic diagram

3.2.13.3 Purpose

Purpose

The function block **_MC_ReadStatus** reads various states of an axis or an external encoder.

3.2.13.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

External encoders

3.2.13.5 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Enable	BOOL	FALSE	Function block enable The values at the status outputs are updated as long as **Enable** equals **TRUE**.

3.2.13.6 Output parameter

Output parameter

Table 3- 2 Generally applicable output parameter

Parameter	Data type	Initial value	Description
Valid	BOOL	FALSE	Display of the validity of the values that can be read at the status outputs
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the states could not be read. The error description can be read at the **ErrorID** output
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 84)).

Table 3- 3 Output parameters for axes

Parameter	Data type	Initial value	Description
ErrorStop	BOOL	FALSE	Display of an active axis stop triggered by an error
Disabled	BOOL	FALSE	Display of no enable of the axis
Stopping	BOOL	FALSE	Display of an active regular axis stop
StandStill	BOOL	FALSE	Display of the standstill signal of the axis
DiscreteMotion	BOOL	FALSE	Display of an active single axis motion with discrete target position
ContinuousMotion	BOOL	FALSE	Display of an active single axis motion without a discrete target position
SynchronizedMotion	BOOL	FALSE	Display of an active synchronous operation on the axis
Homing	BOOL	FALSE	Display of an active homing procedure on the axis
ConstantVelocity	BOOL	FALSE	Display of traversing the axis with a constant velocity
Accelerating	BOOL	FALSE	Display of traversing the axis with an accelerating velocity
Decelerating	BOOL	FALSE	Display of traversing the axis with a decelerating velocity

Table 3- 4 Output parameter for external encoders

Parameter	Data type	Initial value	Description
ErrorStop	BOOL	FALSE	Display of an error on the external encoder
Disabled	BOOL	FALSE	Display of no enable of the external encoder
StandStill	BOOL	FALSE	Display of the standstill signal of the external encoder

Parameter	Data type	Initial value	Description
Homing	BOOL	FALSE	Display of an active homing procedure on the external encoder
ConstantVelocity	BOOL	FALSE	Display of a constant velocity that is measured/determined by the external encoder
Accelerating	BOOL	FALSE	Display of an accelerating velocity that is measured/determined by the external encoder
Decelerating	BOOL	FALSE	Display of a decelerating velocity that is measured/determined by the external encoder

3.2.13.7 ErrorIDs

ErrorIDs

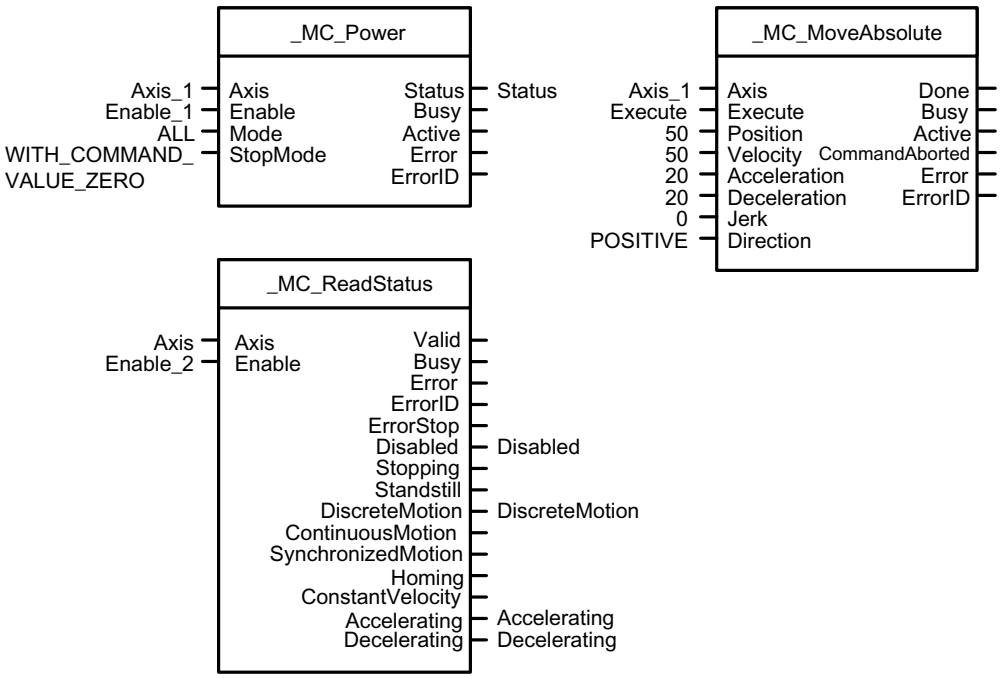
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.13.8 Example

The _MC_ReadStatus block reads the motion statuses of the axis transferred on the **Axis** input parameter during an active positioning command.



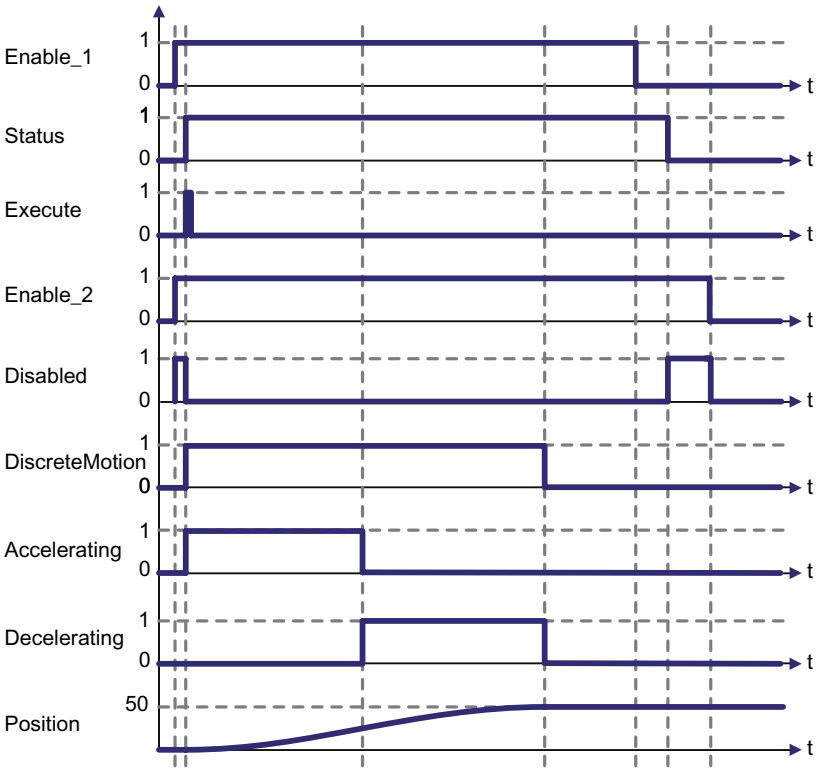


Figure 3-28 _MC_ReadStatus example

3.2.14 _MC_ReadAxisError - Reading an axis error

3.2.14.1 Overview

Schematic diagram
Purpose
Applicable for
Input parameters
Output parameters
ErrorIDs
Example

3.2.14.2 Schematic diagram

Schematic diagram

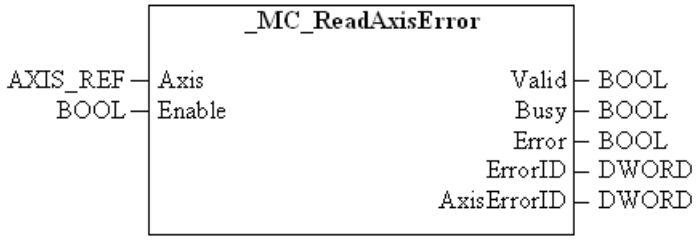


Figure 3-29 _MC_ReadAxisError Schematic diagram

3.2.14.3 Purpose

Purpose

The function block **_MC_ReadAxisError** reads the error status of an axis or an external encoder. The error status is a 32-bit representation of all alarms present on the technology object.

3.2.14.4 Applicable for

Applications

Drive axes
Positioning axes
Following axes
Path axes
External encoders

3.2.14.5 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Enable	BOOL	FALSE	Function block enable The value at output **AxisErrorID** is updated as long as **Enable** equals **TRUE**.

3.2.14.6 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Valid	BOOL	FALSE	Display of the validity of the value which can be read at output **AxisErrorID**.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the error status could not be read. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 88)).
AxisErrorID	DWORD	0	Displays the error status of the axis or external encoder For more information, refer to the section Query of general errors with the _MC_ReadAxisError function block (Page 155).

3.2.14.7 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

Generally, the following is valid:

- The **ErrorID** on the **_MC_...** function blocks refer to special block-related errors and to errors on the axis
- The **AxisErrorID** on the **_MC_ReadAxisError** function block refers to errors on the axis

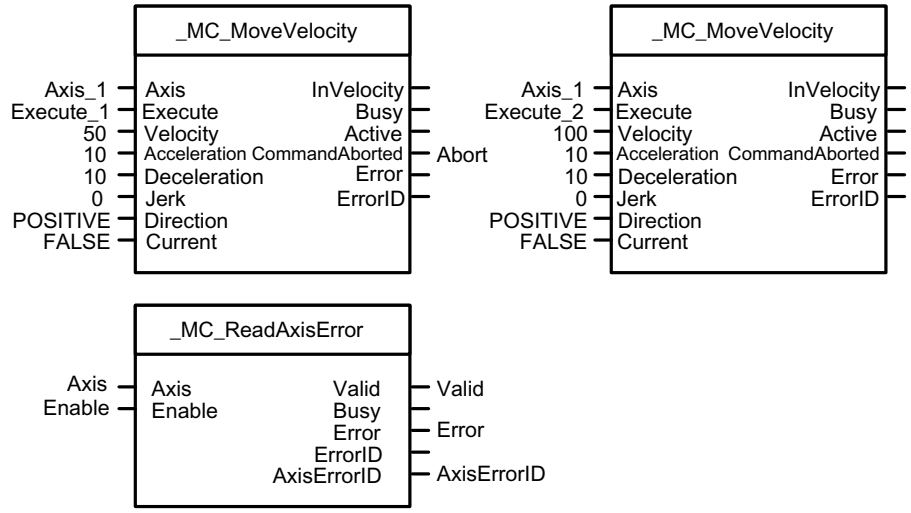
See also

Troubleshooting - PLCopen Blocks (Page 151)

Query of general errors with the _MC_ReadAxisError function block (Page 155)

3.2.14.8 Example

The _MC_ReadAxisError block reads the group error bit string ("errorGroup" system variable) value of the axis transferred on the **Axis** input parameter.



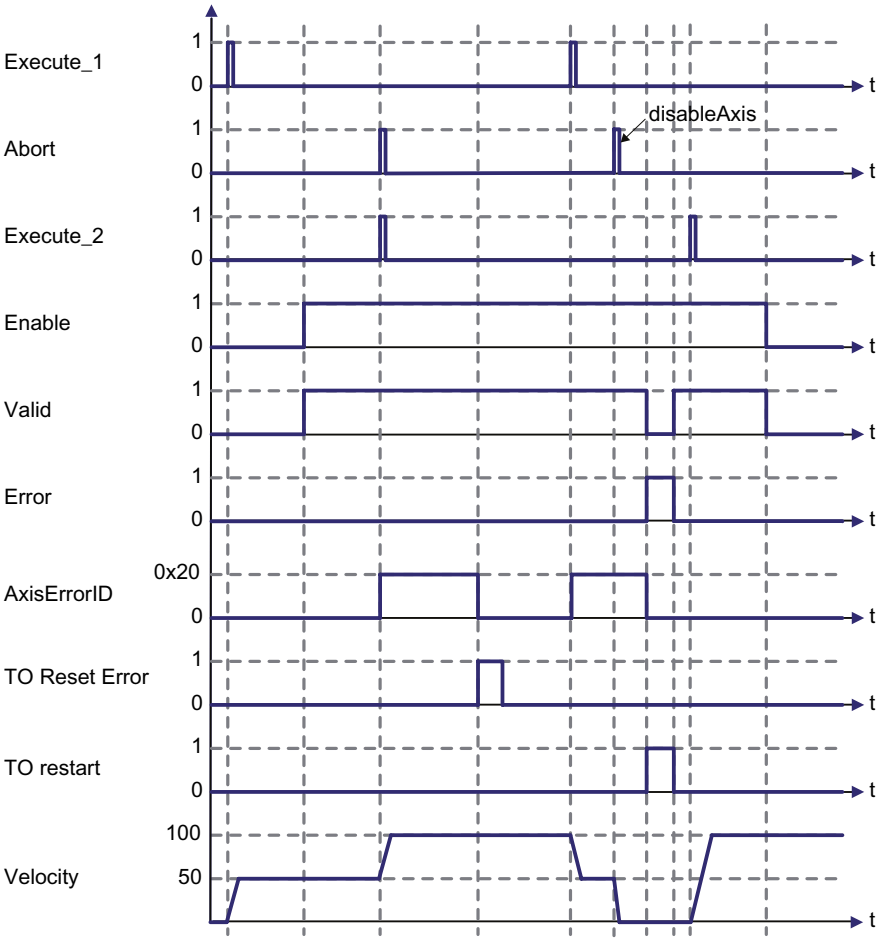


Figure 3-30 _MC_ReadAxisError example

3.2.15 _MC_ReadParameter - Reading axis parameter and outputting in data type LREAL

3.2.15.1 Overview

Schematic diagram
Purpose
Applicable for
Input parameters
Output parameters
ErrorIDs
Example

3.2.15.2 Schematic diagram

Schematic diagram

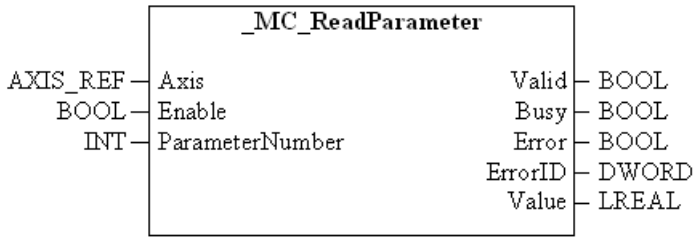


Figure 3-31 _MC_ReadParameter Schematic diagram

3.2.15.3 Purpose

Purpose

The function block **_MC_ReadParameter** reads values of various parameters of an axis or external encoder. Each parameter is specified by a number.

3.2.15.4 Applicable for

Applications

Drive axes
Positioning axes
Following axes
Path axes
External encoders

3.2.15.5 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Enable	BOOL	FALSE	Function block enable The value at output **Value** is updated as long as **Enable** equals **TRUE**.
ParameterNumber	INT	0	Specification of the number of the parameter to be read (see table below).

Parameter description for input ParameterNumber

Table 3- 5 Input ParameterNumber for axes

Parameter	Data type	Access	Description
1	LREAL	Read	**positioningstate.commandposition**: Axis position setpoint
2	LREAL	Read/Write	**swlimit.plusposition**: Position of the positive software limit switch (10^12 is output with inactive limit switch.)
3	LREAL	Read/Write	**swlimit.minusposition**: Position of the negative software limit switch (-10^12 is output with inactive limit switch.)
7 [1)]	LREAL	Read/Write	**typeofaxis.numberofdatasets.dataset_x.dynamicfollowing.minpositiontolerance**: Minimum permissible following error
8	LREAL	Read/Write	**typeofaxis.maxvelocity.maximum**: Maximum permissible velocity
10	LREAL	Read	**motionstatedata.actualvelocity:** Actual axis velocity (A value is only displayed if an encoder was configured on the axis. For drive axes, the encoder must be configured for the setpoint acceptance if applicable.)
11	LREAL	Read	**motionstatedata.commandvelocity**: Axis velocity setpoint
12	LREAL	Read/Write	**typeofaxis.maxacceleration.maximum**: Maximum permissible acceleration
16	LREAL	Read/Write	**typeofaxis.maxjerk.maximum**: Maximum permissible jerk

[1)] The "x" in the name "dataset_x" stands for the active data set.

Table 3- 6 Input ParameterNumber for external encoders

Parameter	Data type	Access	Description
10	LREAL	Read	**motionState.velocity:** Actual velocity measured by the external encoder

3.2.15.6 Output parameter

Output parameters

Parameters	Data type	Initial value	Description
Valid	BOOL	FALSE	Display of the validity of the value which can be read at output **Value**.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the error status could not be read. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 92)).
Value	LREAL	0.0	Display of the parameter value

3.2.15.7 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.15.8 Example

The _MC_ReadParameter block reads the actual velocity of the axis transferred on the **Axis** input parameter.

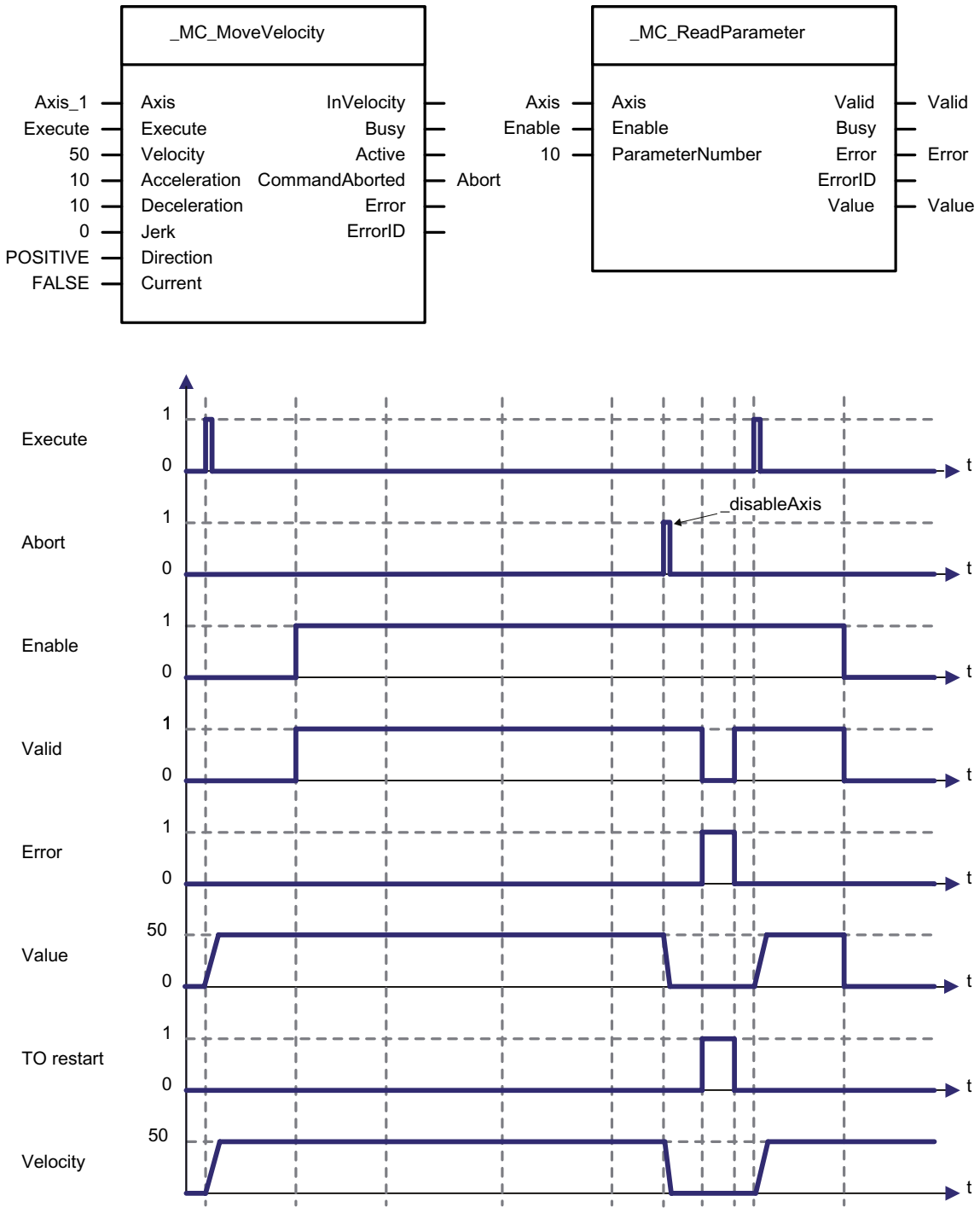


Figure 3-32 _MC_ReadParameter example

3.2.16 _MC_ReadBoolParameter - Reading axis parameter and outputting in data type BOOL

3.2.16.1 Overview

Schematic diagram
Purpose
Applicable for
Input parameters
Output parameters
ErrorIDs
Example

3.2.16.2 Schematic diagram

Schematic diagram

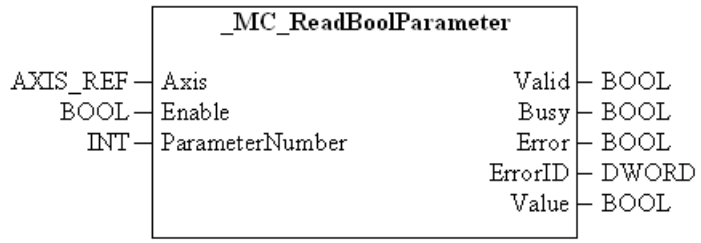


Figure 3-33 _MC_ReadBoolParameter Schematic diagram

3.2.16.3 Purpose

Purpose

The function block **_MC_ReadBoolParameter** reads values of various parameters of an axis or external encoder. Each parameter is specified by a number. The return value of the parameter is converted to the data type of the function block (BOOL), i.e. values that are not of the data type BOOL are converted to FALSE when they are exactly 0 and to TRUE in all other cases.

3.2.16.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

3.2.16.5 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Enable	BOOL	FALSE	Function block enable The value at output **Value** is updated as long as **Enable** equals **TRUE**.
ParameterNumber	INT	0	Specification of the number of the parameter to be read (see table below).

Parameter description for input ParameterNumber

Table 3- 7 Input ParameterNumber for axes

Parameter	Data type	Description
4	BOOL	**swlimit.state**: Activation state of the two software limit switches (The limit switches are always activated or deactivated together.)
6 [1)]	BOOL	**typeofaxis.numberofdatasets.dataset_x.dynamicfollowing.enable**: Activation state of the following error monitoring

[1)] The "x" in the name "dataset_x" stands for the active data set.

3.2.16.6 Output parameter

Output parameters

Parameters	Data type	Initial value	Description
Valid	BOOL	FALSE	Indicates the validity of the value that can be read at output **Value**
Busy	BOOL	FALSE	Indicates function block activity With TRUE, the function block has been started.
Error	BOOL	FALSE	Indicates an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the error status could not be read. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Indicates a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 96)).
Value	BOOL	FALSE	Display of the parameter value

3.2.16.7 ErrorIDs

ErrorIDs

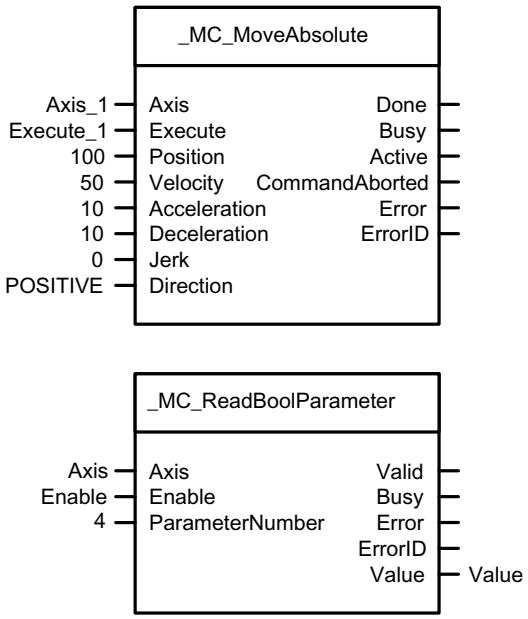
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

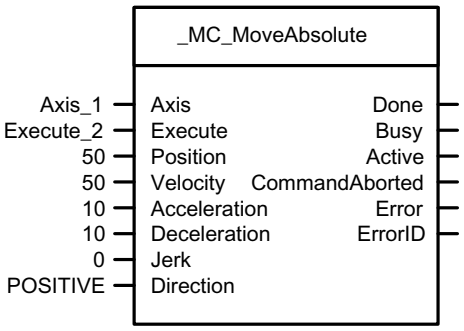
See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.16.8 Example

The _MC_ReadBoolParameter block reads the monitoring status of the software limit switches and converts the value read into the Boolean output value displayed.





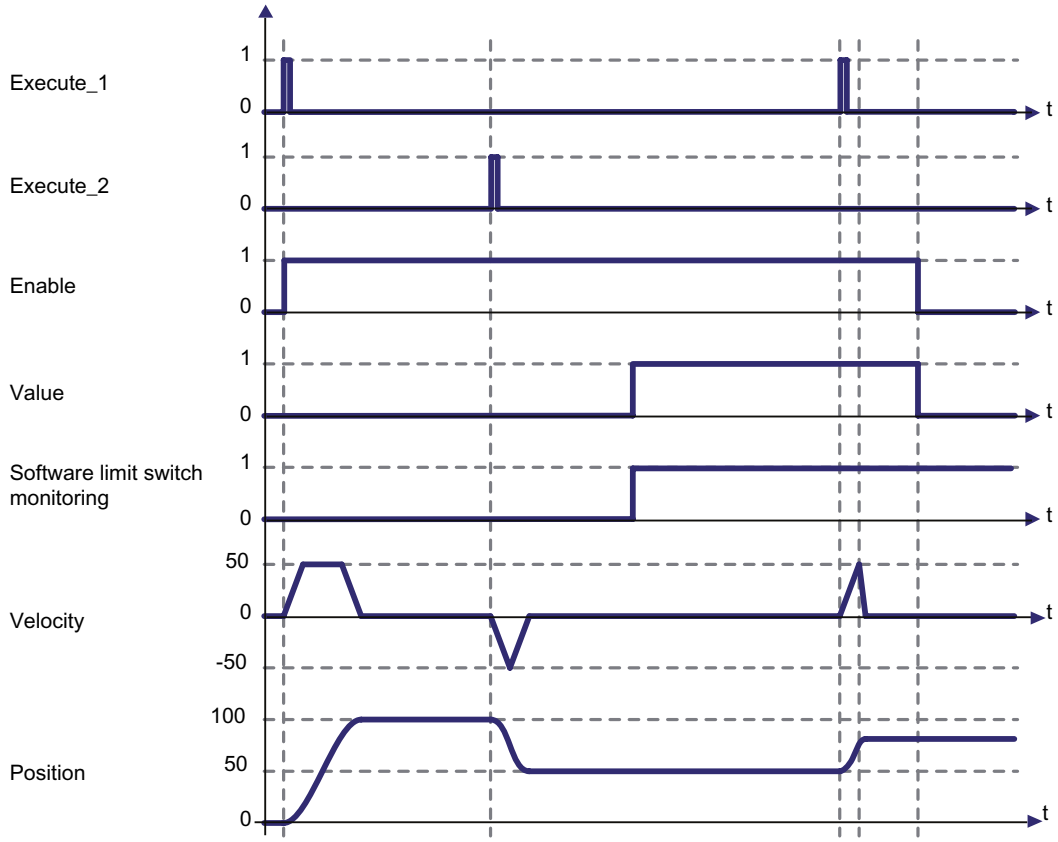


Figure 3-34 _MC_ReadBoolParameter example

3.2.17 _MC_WriteParameter - Writing axis parameter of data type LREAL

3.2.17.1 Overview

Schematic diagram

Purpose

Applicable for

Input parameters

Output parameters

ErrorIDs

Example

3.2.17.2 Schematic diagram

Schematic diagram

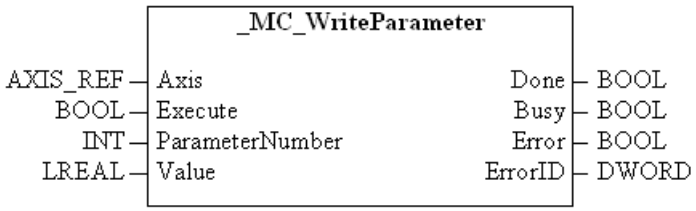


Figure 3-35 _MC_WriteParameter Schematic diagram

3.2.17.3 Purpose

Purpose

The function block **_MC_WriteParameter** writes values of various parameters of an axis. Each axis parameter is specified by a number.

3.2.17.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

3.2.17.5 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis) • Positioning axis (posAxis) • Following axis (followingAxis) • Path axis (pathAxis)
Execute	BOOL	FALSE	Function block enable The parameter at input **Value** is written with a rising edge at this input.
ParameterNumber	INT	0	Specification of the number of the parameter to be written (see table below)
Value	LREAL	0.0	Specification of the parameter value to be written

Parameter description for input ParameterNumber

Parameter number	Data type	Description
2	LREAL	**swlimit.plusposition**: Position of the positive software limit switch
3	LREAL	**swlimit.minusposition**: Position of the negative software limit switch
7 [1) 2)]	LREAL	**typeofaxis.numberofdatasets.dataset_x.dynamicfollowing.minpositiontolerance; typeofaxis.numberofdatasets.dataset_x.dynamicfollowing.maxpositiontolerance**: Minimum or maximum permissible following error
8	LREAL	**typeofaxis.maxvelocity.maximum**: Maximum permissible velocity (the change only takes effect after a restart)
12	LREAL	**typeofaxis.maxacceleration.maximum**: Maximum permissible acceleration
16	LREAL	**typeofaxis.maxjerk.maximum**: Maximum permissible jerk

1) The "x" in the name "dataset_x" stands for the active data set, i.e. the data item that is combined with parameter 7 is only changed at this data set.

2) Not available at virtual axes

3.2.17.6 Output parameter

Output parameters

Parameters	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the parameter could be written successfully.
Busy	BOOL	FALSE	Indicates function block activity With TRUE, the function block has been started.
Error	BOOL	FALSE	Indicates an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the parameter could not be written. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Indicates a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 100)).

3.2.17.7 ErrorIDs

ErrorIDs

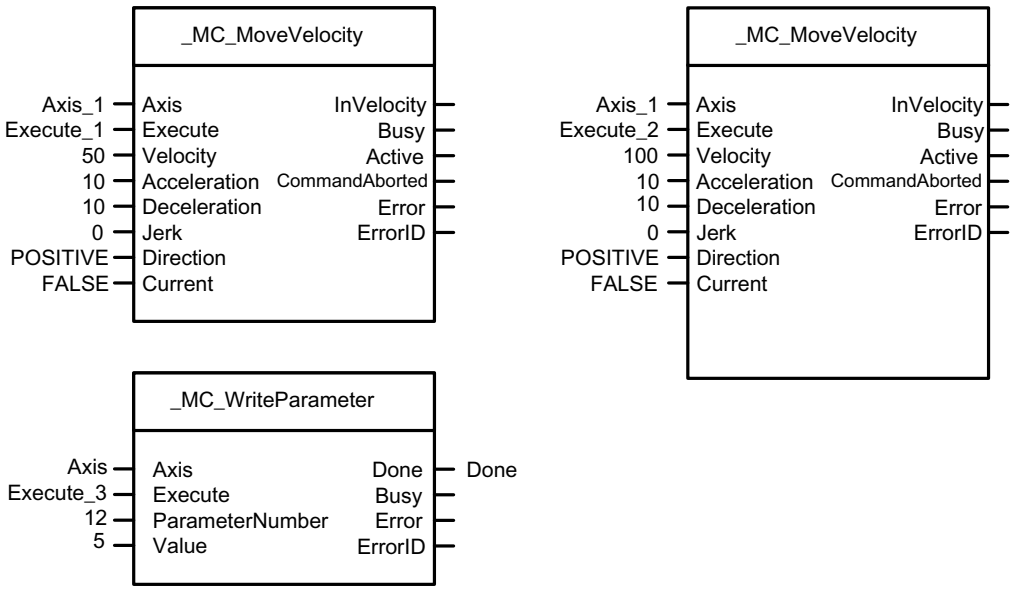
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.17.8 Example

The _MC_WriteParameter block changes the maximum permissible acceleration for the axis transferred on the **Axis** input parameter.



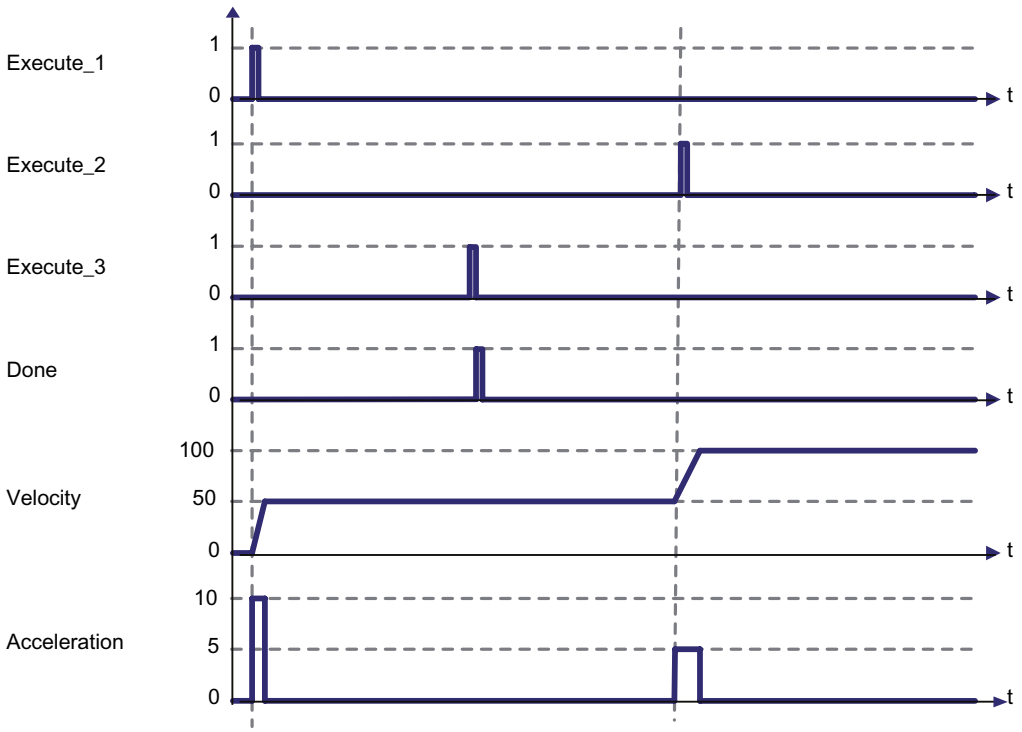


Figure 3-36 _MC_WriteParameter example

3.2.18 _MC_WriteBoolParameter - Writing axis parameter of data type BOOL

3.2.18.1 Overview

Schematic diagram

Purpose

Applicable for

Input parameters

Output parameters

ErrorIDs

Example

3.2.18.2 Schematic diagram

Schematic diagram

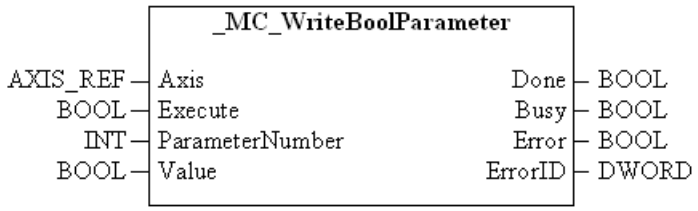


Figure 3-37 _MC_WriteBoolParameter Schematic diagram

3.2.18.3 Purpose

Purpose

The function block **_MC_WriteBoolParameter** writes values of various parameters of an axis. Each axis parameter is specified by a number.

3.2.18.4 Applicable for

Applications

Drive axes

Positioning axes

Following axes

Path axes

3.2.18.5 Input parameters

Input parameters

Parameters	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Drive axis (driveAxis data type) • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The parameter at input **Value** is written with a rising edge at this input.
ParameterNumber	INT	0	Specification of the number of the parameter to be written (see table below)
Value	BOOL	FALSE	Specification of the parameter value to be written

Parameter description for input ParameterNumber

Parameter number	Data type	Description
4	BOOL	**swlimit.state**: Activation of the positive and negative software limit switches (the limit switches are always activated or deactivated together)
6 [1) 2)]	BOOL	**typeofaxis.numberofdatasets.dataset_x.dynamicfollowing.enable**: Activation of following error monitoring (the change only takes effect after a restart) The data sets are configured at the corresponding technology object. Several data sets can be used, in order to (for example): • Change over controller data during operation • Change over the encoder used during operation (motor encoder, machine encoder, etc.) For more details, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual.

1) The "x" in the name "dataset_x" stands for all existing data sets, i.e. the data item that is combined with parameter 6 is changed in all data sets at the same time.

2) Not available at virtual axes

3.2.18.6 Output parameter

Output parameters

Parameters	Data type	Initial value	Description
Done	BOOL	FALSE	Indicates completion of the function block With TRUE, the parameter was able to be written successfully.
Busy	BOOL	FALSE	Indicates function block activity With TRUE, the function block has been started.
Error	BOOL	FALSE	Indicates an error in the function block With TRUE, either an error has occurred during the initialization of the function block or the parameter could not be written. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Indicates a function block error code The error code is always output in conjunction with the **Error** output (see Chapter ErrorIDs (Page 104)).

3.2.18.7 ErrorIDs

ErrorIDs

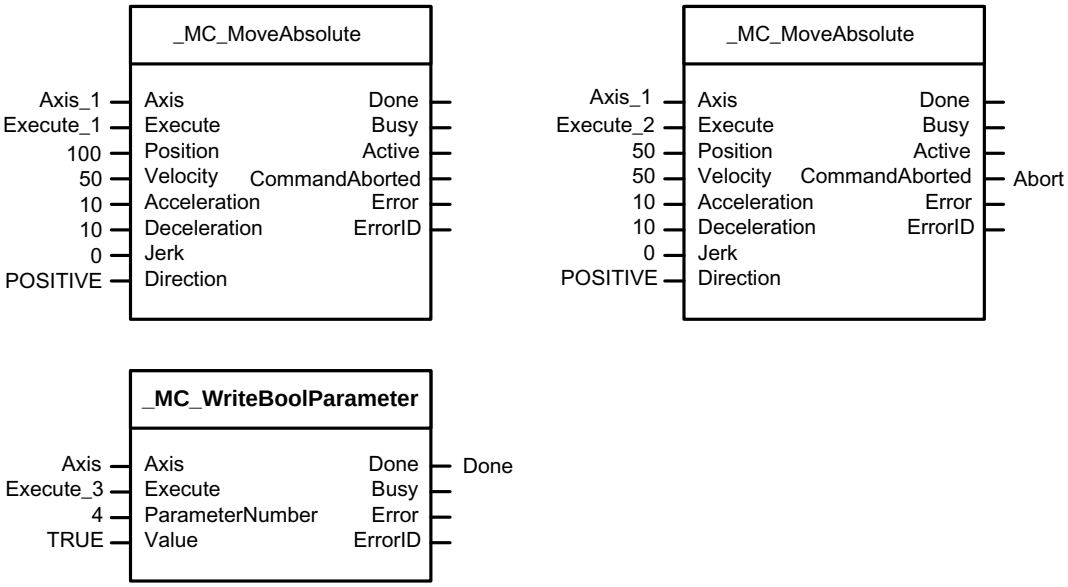
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.2.18.8 Example

The _MC_ WriteBoolParameter block activates software limit switch monitoring for the axis transferred on the **Axis** input parameter.



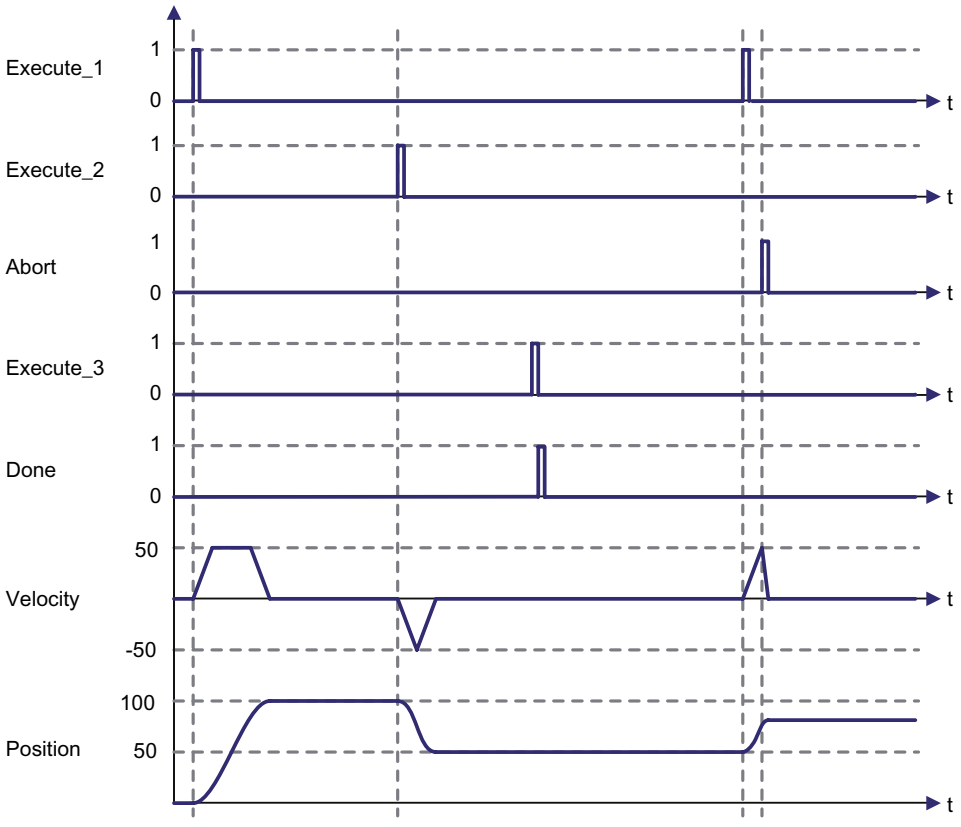


Figure 3-38 _MC_WriteBoolParameter example

3.3 MultiAxis

3.3.1 _MC_GearIn - Starting gearing

3.3.1.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Examples

3.3.1.2 Schematic diagram

Schematic diagram

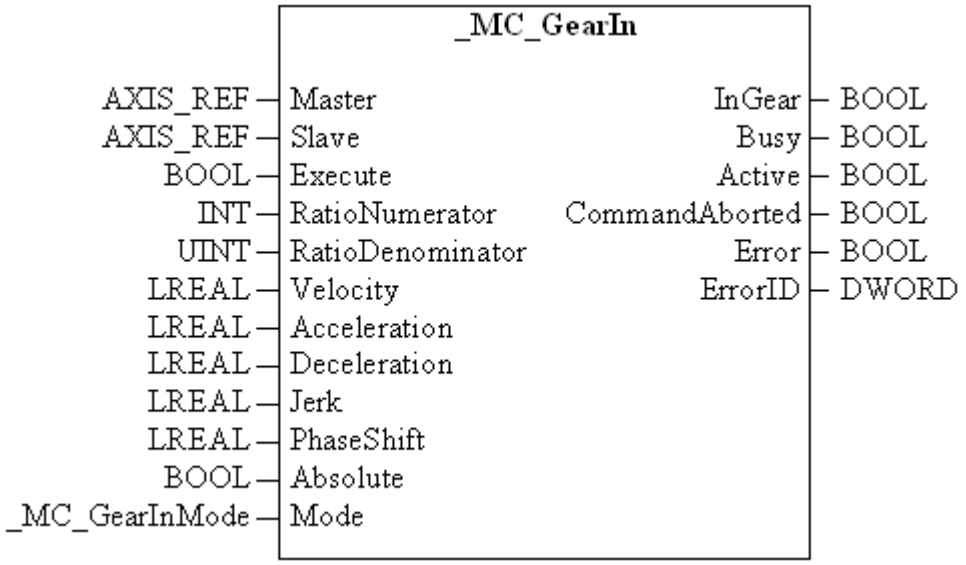


Figure 3-39 _MC_GearIn Schematic diagram

3.3.1.3 Purpose

Purpose

The function block **_MC_GearIn** starts a gearing between a master and a slave axis.

The dynamic response parameters **Velocity**, **Acceleration**, **Deceleration**, and **Jerk** define the dynamic response of the slave axis during synchronization in accordance with time.

The gear ratio is specified as a fraction.

Synchronous operation may be:

- relative to the start position or synchronization position
- absolute, in relation to the axis zero

By transmitting a new _MC_GearIn command, the gear ratio can be changed during operation. This operation does not require the master or slave axis to be stopped. Transitions are executed according to specified acceleration or deceleration values.

The function block can be started when the leading axis is at a standstill, or when it is in motion.

3.3.1.4 Applicable for

Applications

Master:

- Positioning axes
- Following axes
- Path axes
- External Encoders
- ...

Slave:

- Following axes
- Path axes with synchronous operation activated

3.3.1.5 Requirements

Requirements

The master is set for use as a potential master value at the synchronous object of the slave axis.

Master and slave axes are enabled.

No **_MC_Stop** active on the slave axis

3.3.1.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Master	AXIS_REF	0	Specification of reference to the master (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Slave	AXIS_REF	0	Specification of reference to the slave axis (name of TO) The following technology objects can be homed: • Following axis (followingAxis data type) • Path axis with synchronous operation activated (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The slave axis is synchronized with the interconnected master with a rising edge on this input.
RatioNumerator	INT	1	Specification of the numerator of the gear ratio
RatioDenominator	UINT	1	Specification of the denominator of the gear ratio
Velocity	LREAL	-1.0	Specification of the maximum synchronization velocity The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.velocity system variable of the interconnected synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Acceleration	LREAL	-1.0	Specification of the maximum synchronization acceleration The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.positiveaccel system variable of the interconnected synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).

Parameter	Data type	Initial value	Description
Deceleration	LREAL	-1.0	Specification of the maximum synchronization deceleration The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.neagtiveaccel system variable of the interconnected synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Jerk	LREAL	-1.0	Specification of the maximum synchronization jerk The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. To activate the jerk limitation, the configuration data **SyncingMotion.smoothAbsoluteSynchronization** on the interconnected synchronous object must be set to **YES**. Otherwise the parameter specification for **Jerk** is ignored and a trapezoidal velocity profile is always used. Value > 0: The specified value is used Value = 0: Use trapezoidal velocity profile Value < 0: The preset values in the system variables userdefault.syncdynamics.positiveaccelstartjerk, userdefault.syncdynamics.positiveaccelendjerk, userdefault.syncdynamics.negativeaccelstartjerk, and userdefault.syncdynamics.negativeaccelendjerk of the interconnected synchronous object are used (see "Preassignment > Dynamic Response" dialog at synchronous object).
PhaseShift	LREAL	0.0	Specification of the master value-related phase shift during synchronous operation The phase shift is absolute when synchronous operation is reached if **Absolute** = TRUE. The specified phase shift is added to the phase offset determined by the relative relationship if **Absolute** = FALSE. Value ≠ 0: The specified value is used Value = 0: No phase shift Slave value = gear ratio x (master value – PhaseShift)

Parameter	Data type	Initial value	Description
Absolute	BOOL	TRUE	Specification of the gearing type With TRUE, gearing is absolute in relation to the axis zero for the relevant axes. A phase shift can be set via the parameter **PhaseShift**. With FALSE, gearing is relative to the start position or synchronization position. A phase shift at the **PhaseShift** input during synchronization does not result in any additional shifting of the slave position in relation to the master. However, the phase shift is accepted as an absolute shift between the master and slave (system variable gearingadjustments.master.offset).
Mode	_MC_GearInMode	USER_DEFAULT	Specification of the synchronization mode / engage mode USER_DEFAULT: With the exception of the values set at the block inputs • Absolute (userDefault.gearingSettings.typeOfGearing), • RatioNumerator (userDefault.gearingSettings.defineMode, userDefault.gearingSettings.numerator), and • RatioDenominator (userDefault.gearingSettings.defineMode, userDefault.gearingSettings.denominator) the synchronization parameters preset at the synchronous object in the "Preassignment > Gearing" (→ system variables under userdefault.gearingsettings…) and "Preassignment > Dynamic Response" (→ system variables under userdefault.syncdynamics…) dialogs are accepted. IMMEDIATELY_BY_TIME_PROFILE: Synchronization is performed immediately according to time taking into account the dynamic response values set on the function block. Synchronous operation is operated using the synchronous object settings • "Time-related synchronization" profile specification from the "Preassignment > Dynamic Response" dialog (userDefault.syncprofile.syncprofilereference parameter equals RELATE_SYNC_PROFILE_TO_TIME), • Synchronization "with immediate effect" from the "Preassignment > Gear Synchronization" dialog (parameter userDefault.gearingsettings.synchronizingmode equals IMMEDIATELY), and • "Compatibility mode" synchronization direction (userDefault.gearingsettings.synchronizingdirection parameter equals SYSTEM_DEFINED [1]).

[1] SYSTEM_DEFINED corresponds to the **shortest distance**setting, although the direction of motion is maintained while the axis is in motion.

3.3.1.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
InGear	BOOL	FALSE	Display of the synchronism of the master and slave axis With TRUE, the slave axis is in synchronous operation with the master.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 111)).

3.3.1.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.3.1.9 Examples

Example: Gearing

At the master axis, a positioning command is active at 100 mm. Two mutually overriding _MC_GearIn blocks with different gear ratios establish the gearing between the axes transferred on the **master** and **slave** input parameters.

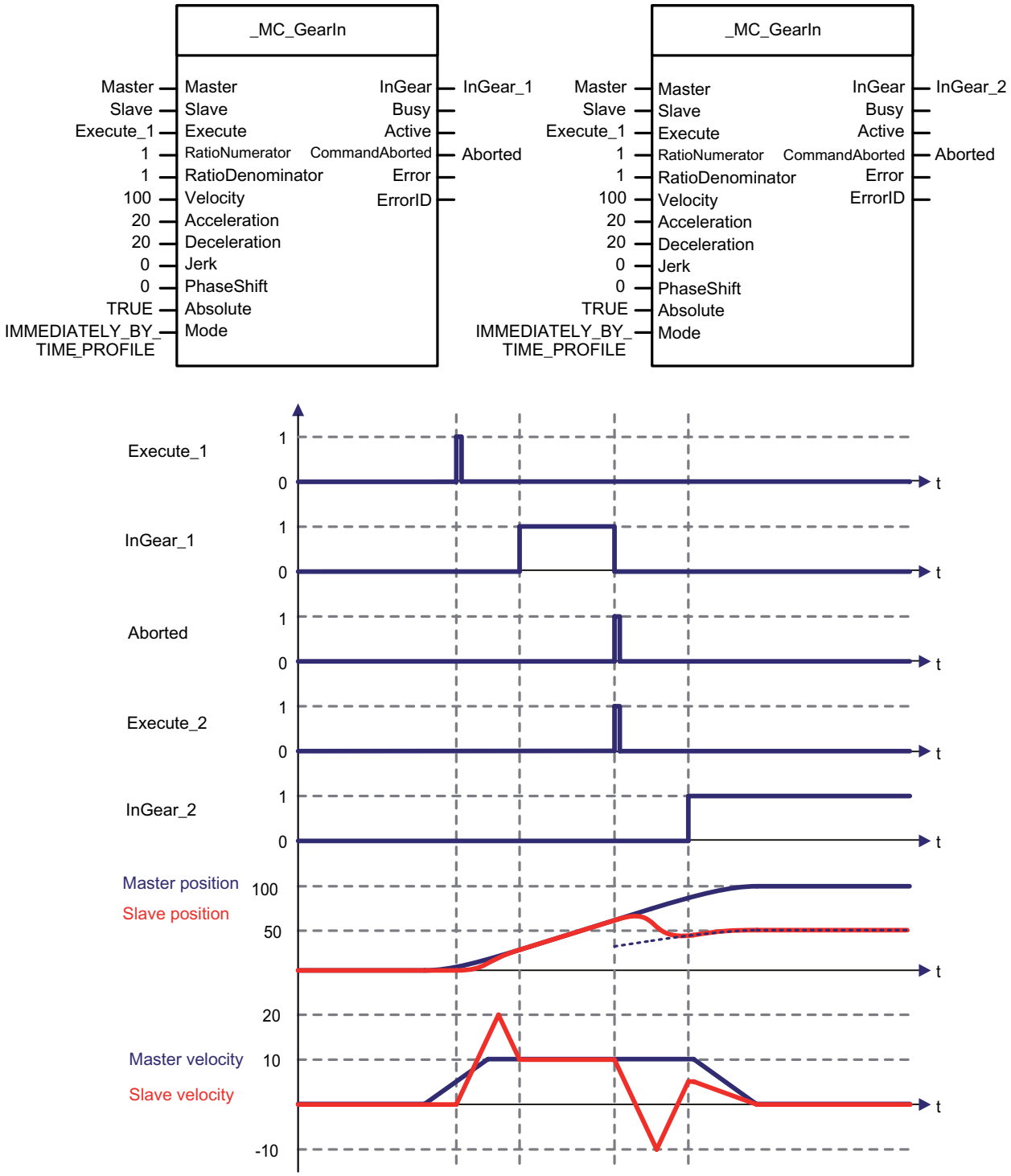
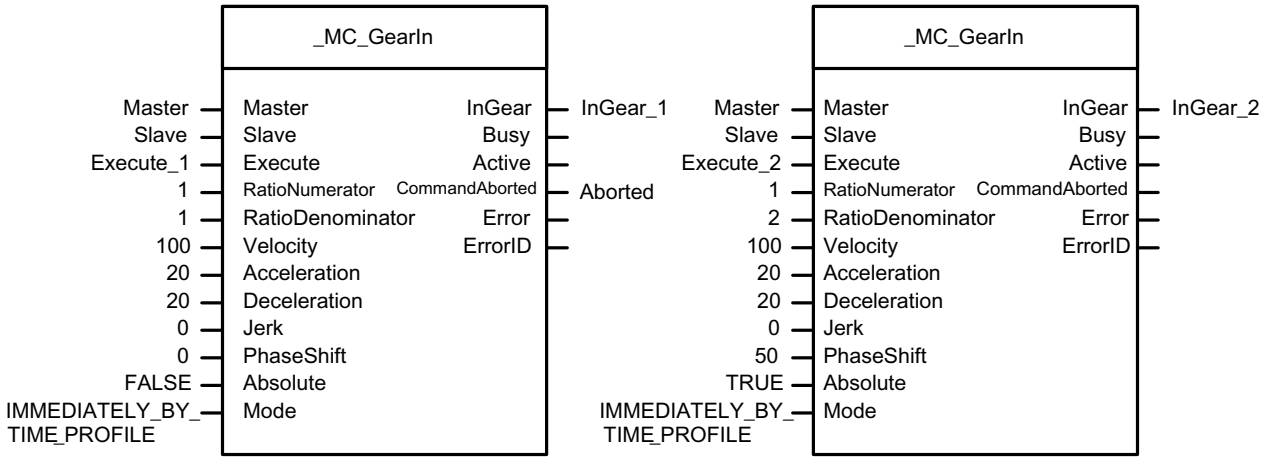


Figure 3-40 _MC_GearIn example: Gearing

Example: Gearing with gear ratio

At the master axis, a positioning command is active at 100 mm. Two mutually overriding _MC_GearIn blocks with different gear ratios establish the gearing between the axes transferred on the **master** and **slave** input parameters. While the first block is being executed, the slave position changes relative to the master position. While the second block is being executed, the slave position changes absolutely in relation to the master position, i.e. the slave position is the same as the master position evaluated using the gear ratio (RatioNumerator = 1, RatioDenominator = 2).



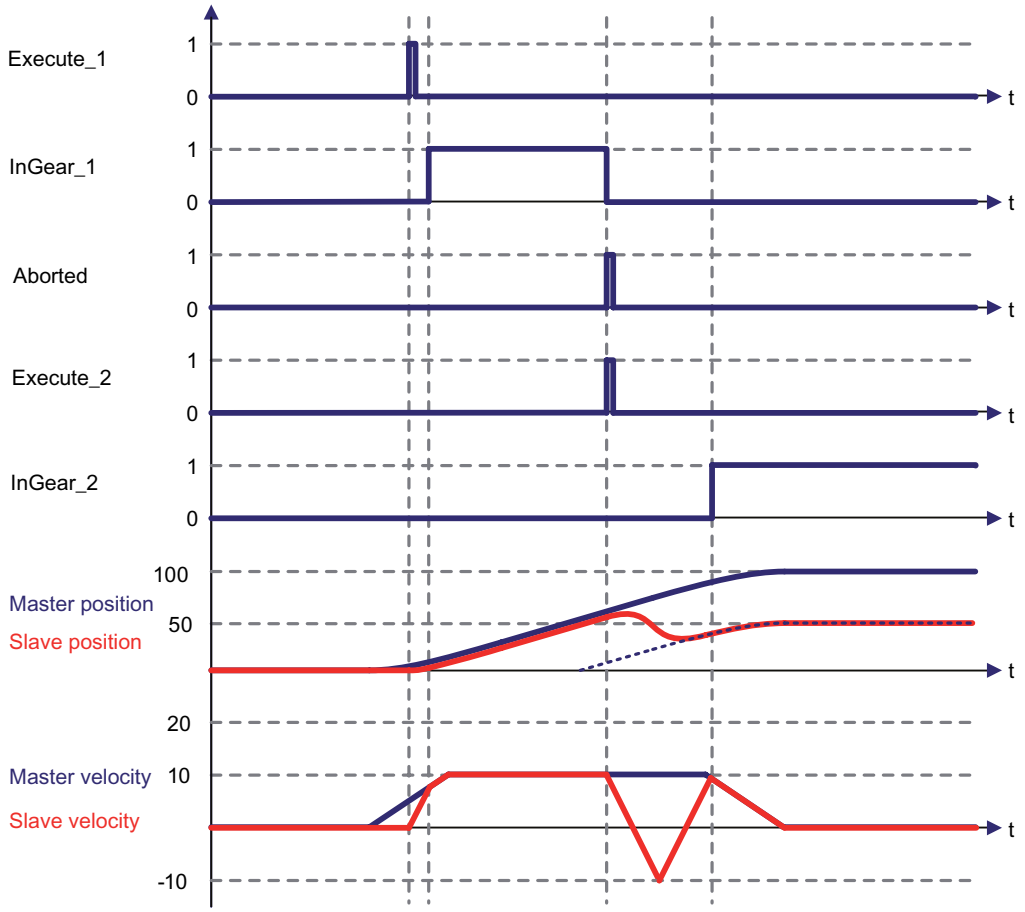


Figure 3-41 _MC_GearIn example: Gearing with gear ratio

3.3.2 _MC_GearOut - Terminating gearing

3.3.2.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.3.2.2 Schematic diagram

Schematic diagram

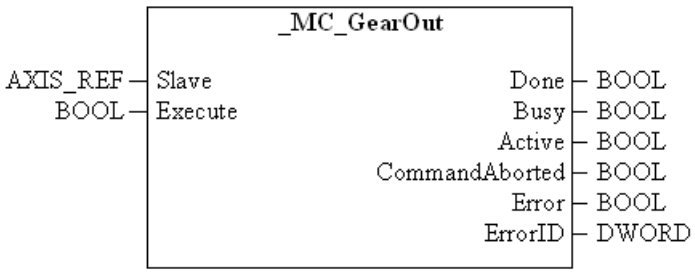


Figure 3-42 _MC_GearOut Schematic diagram

3.3.2.3 Purpose

Purpose

The function block **_MC_GearOut** terminates a gearing and stops the slave axis. The slave axis is desynchronized in accordance with the presettings at the synchronous object in the "Preassignment > Gearing" (→ system variables under **userdefault.gearingsettings…**) and "Preassignment > Dynamic Response" (→ system variables under **userdefault.syncdynamics…**) dialogs.

Recommendation

Use the function block when the shutdown procedure is to depend on the position of the master and/or the slave axis. You can also remove the slave axis from the synchronous operation with the technology functions **_MC_Stop**, **_MC_MoveRelative**, **_MC_MoveAdditive**, **_MC_MoveAbsolute** or **_MC_MoveVelocity**.

3.3.2.4 Applicable for

Applications

Following axes

Path axes with synchronous operation activated

3.3.2.5 Requirements

Requirements

A gearing must be active on the slave axis. If no synchronous operation is active, the function block is aborted.

No **_MC_Stop** active on the slave axis.

3.3.2.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Slave	AXIS_REF	0	Specification of reference to the slave axis (name of TO) The following technology objects can be homed: • Following axis (followingAxis data type) • Path axis with synchronous operation activated (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The synchronous operation of the slave axis with the interconnected master is terminated with a rising edge on this input.

3.3.2.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the slave axis has been desynchronized from the interconnected master.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.

Parameter	Data type	Initial value	Description
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 117)).

3.3.2.8 ErrorIDs

ErrorIDs

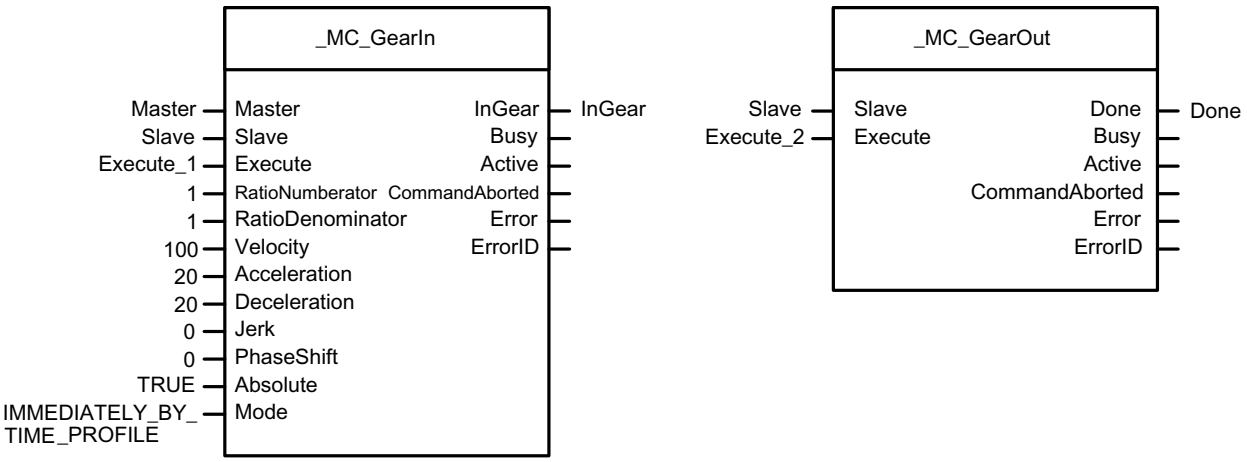
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.3.2.9 Example

At the master axis, a positioning command is active at 100 mm. The _MC_GearOut block terminates gearing of the axis transferred on the **slave** input parameter.



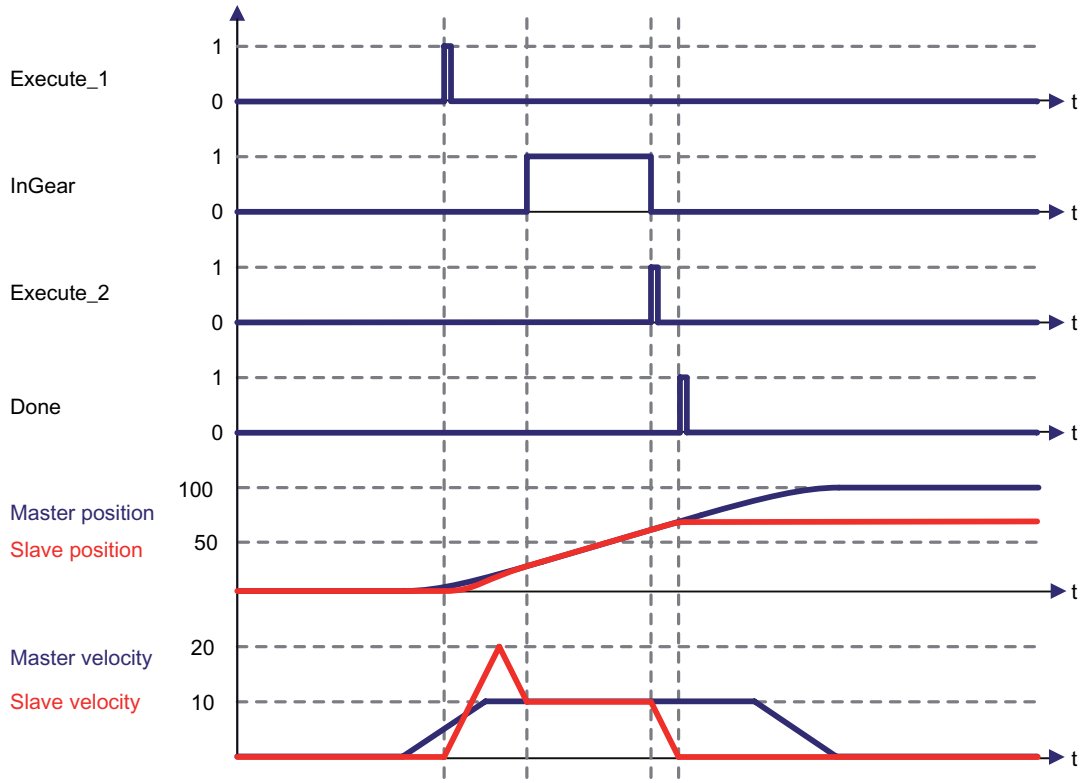


Figure 3-43 _MC_GearOut example

3.3.3 _MC_CamIn - Starting camming

3.3.3.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Examples

3.3.3.2 Schematic diagram

Schematic diagram

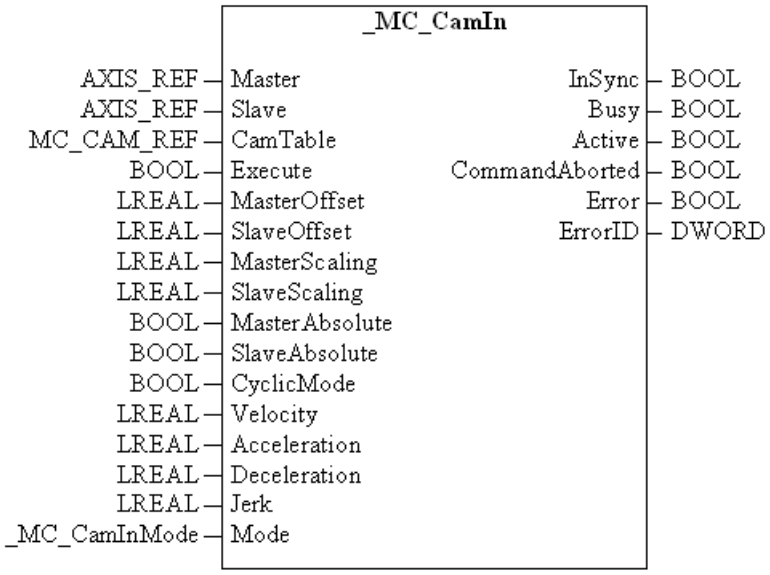


Figure 3-44 _MC_CamIn Schematic diagram

3.3.3.3 Purpose

Purpose

The function block **_MC_CamIn** starts a camming between a master and a slave axis.

The dynamic response parameters **Velocity**, **Acceleration**, **Deceleration**, and **Jerk** define the dynamic response of the slave axis during synchronization in accordance with time.

The cam profile can be scaled and/or the position offset.

The specified cam can optionally be run through once or periodically.

Camming can be relative or absolute.

3.3.3.4 Applicable for

Applications

Master:

- Positioning axes
- Following axes
- Path axes
- External Encoders
- ...

Slave:

- Following axes
- Path axes with synchronous operation activated

3.3.3.5 Requirements

Requirements

When configuring the synchronous object of the slave axis, you must have selected the required cam and master as potential interconnection objects.

Master and slave axes are enabled.

No **_MC_Stop** active on the slave axis

3.3.3.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Master	AXIS_REF	0	Specification of reference to the master (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type) • External encoder (externalEncoderType data type)
Slave	AXIS_REF	0	Specification of reference to the slave axis (name of TO) The following technology objects can be homed: • Following axis (followingAxis data type) • Path axis with synchronous operation activated (_pathAxis data type)
CamTable	MC_CAM_REF	0	Specification of the cam (name)
Execute	BOOL	FALSE	Function block enable The slave axis is synchronized with the interconnected master with a rising edge on this input.
MasterOffset	LREAL	0.0	Specification of the offset of the master values in the master coordinates.
SlaveOffset	LREAL	0.0	Specification of the offset of the slave values in the slave coordinates.
MasterScaling	LREAL	1.0	Specification of the scaling for the master values in the master coordinates.
SlaveScaling	LREAL	1.0	Specification of the scaling for the slave values in the slave coordinates.
MasterAbsolute	BOOL	TRUE	Specification of the evaluation method of the master values With TRUE, the master values are applied as absolute values in the domain of the cam. With FALSE, the master values are evaluated relative to the start value of the cam.
SlaveAbsolute	BOOL	TRUE	Specification of the evaluation method of the slave values With TRUE, the slave values are applied as absolute values in the range of the cam. With FALSE, the slave values are applied relative to the start value of the cam. During synchronization, the slave axis also travels the path difference between the start of the cam and the cam start value.
CyclicMode	BOOL	TRUE	Specification of the cam mode With TRUE, the cam repeats after reaching its end point. With FALSE, the function block is terminated after one cycle of the cam.

Parameter	Data type	Initial value	Description
Velocity	LREAL	-1.0	Specification of the maximum synchronization velocity The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.velocity system variable of the interconnected synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Acceleration	LREAL	-1.0	Specification of the maximum synchronization acceleration The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.positiveaccel system variable of the interconnected synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Deceleration	LREAL	-1.0	Specification of the maximum synchronization deceleration The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.negativeaccel system variable of the interconnected synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Jerk	LREAL	-1.0	Specification of the maximum synchronization jerk The parameter is only taken into account with **Mode** equals **IMMEDIATELY_BY_TIME_PROFILE**. To activate the jerk limitation, the configuration data **SyncingMotion.smoothAbsoluteSynchronization** on the interconnected synchronous object must be set to **YES**. Otherwise the parameter specification for **Jerk** is ignored and a trapezoidal velocity profile is always used. Value > 0: The specified value is used Value = 0: Use trapezoidal velocity profile Value < 0: The preset values in the system variables userdefault.syncdynamics.positiveaccelstartjerk, userdefault.syncdynamics.positiveaccelendjerk, userdefault.syncdynamics.negativeaccelstartjerk, and userdefault.syncdynamics.negativeaccelendjerk of the interconnected synchronous object are used (see "Preassignment > Dynamic Response" dialog at synchronous object).

Parameter	Data type	Initial value	Description
Mode	_MC_CamInMode	USER_DEFAULT	Specification of the synchronization mode / engage mode USER_DEFAULT: With the exception of the values set at the block inputs • MasterAbsolute (userDefault.cammingSettings.masterMode), • SlaveAbsolute (userDefault.cammingSettings.slaveMode) and • CyclicMode (userDefault.cammingSettings.cammingMode) the synchronization parameters preset at the synchronous object in the "Preassignment > Camming" (→ system variables under userdefault.cammingsettings…) and "Preassignment > Dynamic Response" (→ system variables under userdefault.syncdynamics…) dialogs are accepted. IMMEDIATELY_BY_TIME_PROFILE: Synchronization is performed immediately according to time taking into account the dynamic response values set on the function block. Synchronous operation is performed using the synchronous object settings • "Time-related synchronization" profile specification from the "Preassignment > Dynamic Response" dialog (userDefault.syncprofile.syncprofilereference parameter equals RELATE_SYNC_PROFILE_TO_TIME), • Synchronization "with immediate effect" from the "Preassignment > Cam Synchronization" dialog (parameter userDefault.cammingsettings.synchronizingmode equals IMMEDIATELY), and • "Compatibility mode" synchronization direction (userDefault.cammingsettings.synchronizingdirection parameter equals SYSTEM_DEFINED [1]).

[1] SYSTEM_DEFINED corresponds to the **shortest distance**setting, although the direction of motion is maintained while the axis is in motion.

3.3.3.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
InSync	BOOL	FALSE	Display of the synchronism of the master and slave axis With TRUE, the slave axis is in synchronous operation with the master.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the slave axis.

Parameter	Data type	Initial value	Description
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 124)).

3.3.3.8 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.3.3.9 Examples

Example: Camming

At the master axis, a positioning command is active at 100 mm. Two mutually overriding _MC_CamIn blocks with different cam profiles establish the camming between the axes transferred on the **master** and **slave** input parameters.

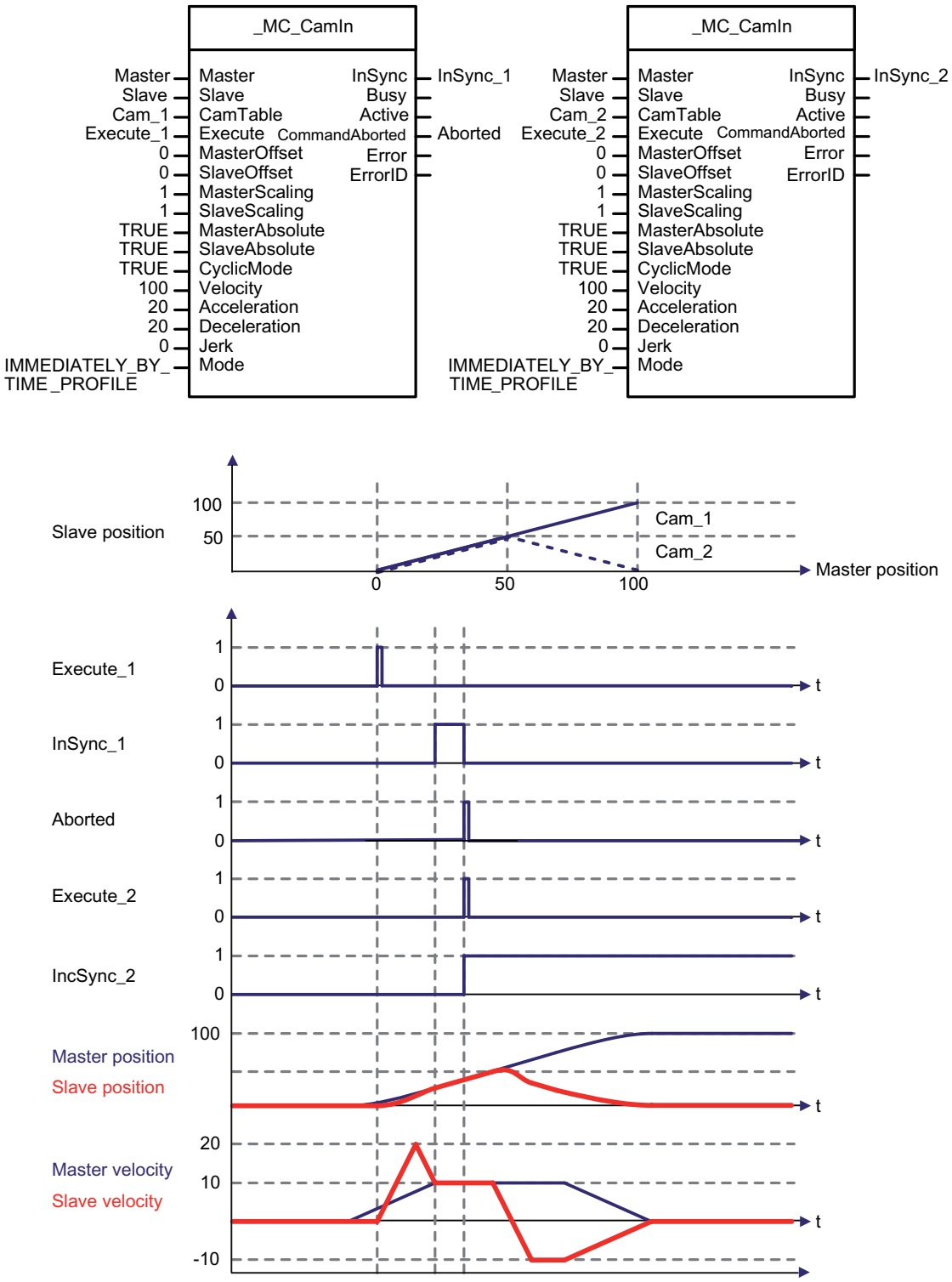


Figure 3-45 _MC_CamIn example: Camming

Example: Camming with offset

At the master axis, a positioning command is active at 100 mm. Two mutually overriding _MC_CamIn blocks with different cam profiles establish the camming between the axes transferred on the **master** and **slave** input parameters. While the first block is being executed, the master position shifts by 10 mm in relation to the slave position (MasterOffset). While the second block is being executed, the slave position shifts by 10 mm in relation to the master position (SlaveOffset).

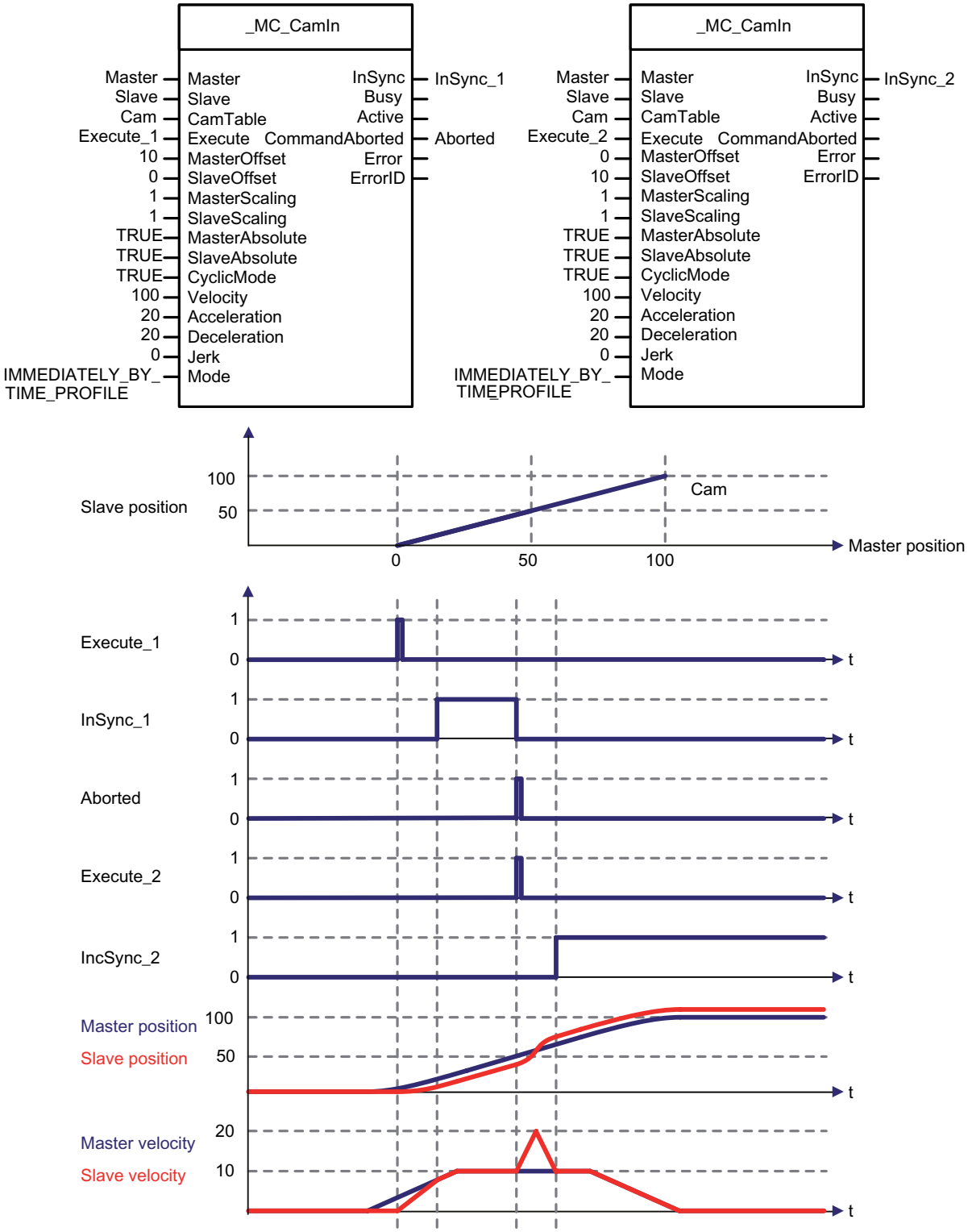


Figure 3-46 _MC_CamIn example: Camming with offset

Example: Camming with scaling

At the master axis, a positioning command is active at 100 mm. Two mutually overriding _MC_CamIn blocks with different cam profiles establish the camming between the axes transferred on the **master** and **slave** input parameters. While the first block is being executed, the cam is scaled by a factor of 2 in relation to the master position (MasterScaling). While the second block is being executed, the cam is scaled by a factor of 2 in relation to the slave position (SlaveScaling).

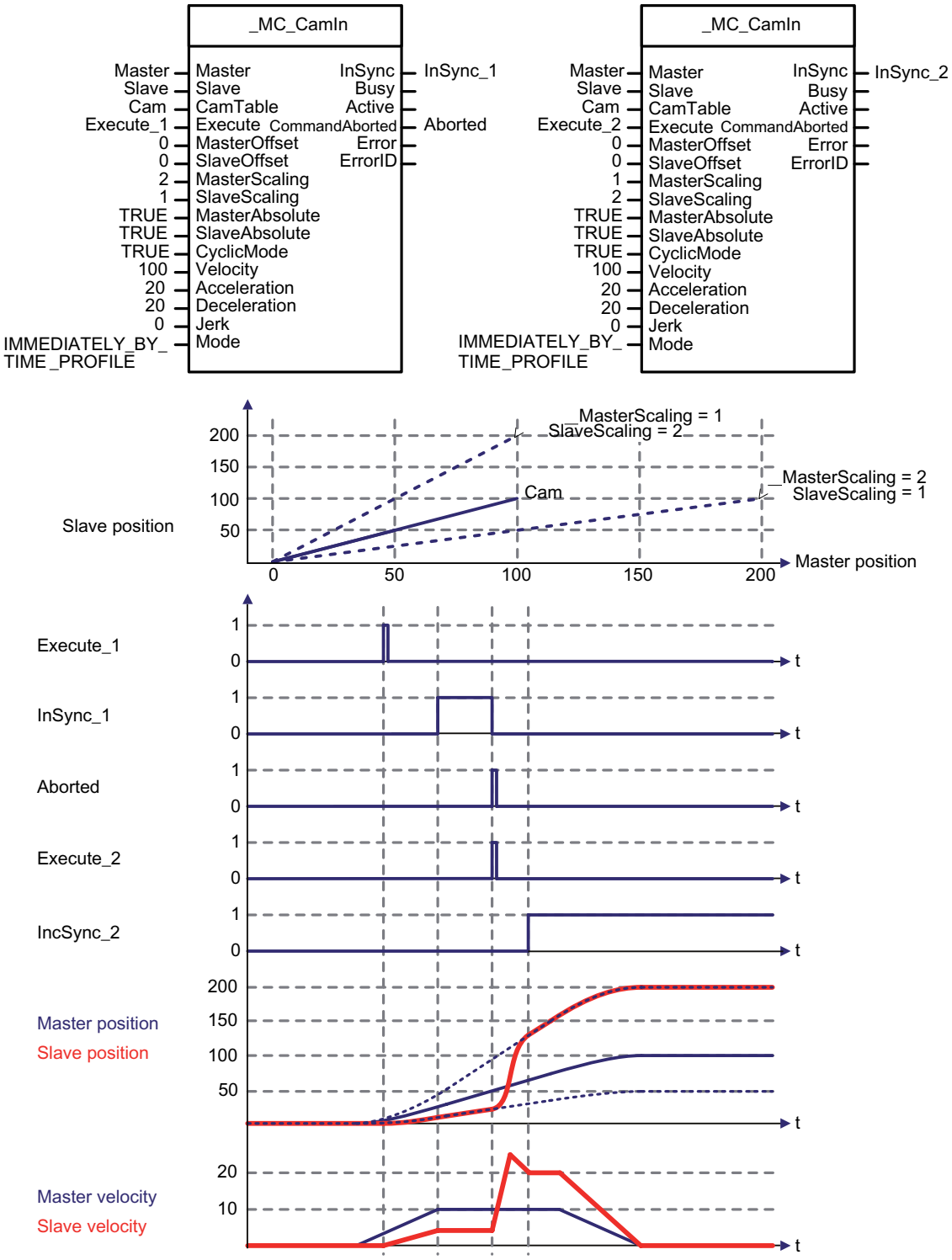


Figure 3-47 _MC_CamIn example: Camming with scaling

Example: Camming with starting point shift to actual position of slave axis

At the master axis, a positioning command is active at 100 mm. When the first block starts (Execute_1 = TRUE), the slave axis synchronizes to the master axis in accordance with the positional relationship $Pos_{Slave} = Pos_{Master}$ and then follows this axis.

When the second block starts (Execute_2 = TRUE), the cam starting point is shifted to the actual position of the slave axis (Pos_{Slave} = 48 mm) (→ interpretation of slave position relative to cam starting point). The cam does not change in relation to the master position (→ interpretation of master position is absolute in the cam's range of values). Therefore, for the second block the equation $Pos_{Slave} = Pos_{Master}$ + 48 mm applies as the positional relationship.

In the figure below, in accordance with this positional relationship the position setpoint of the slave axis is shifted by 48 mm in relation to the master position. Once the master position setpoint of 100 mm has been reached, the result is a position setpoint of 148 mm (100 mm + 48 mm = 148 mm) for the slave axis.

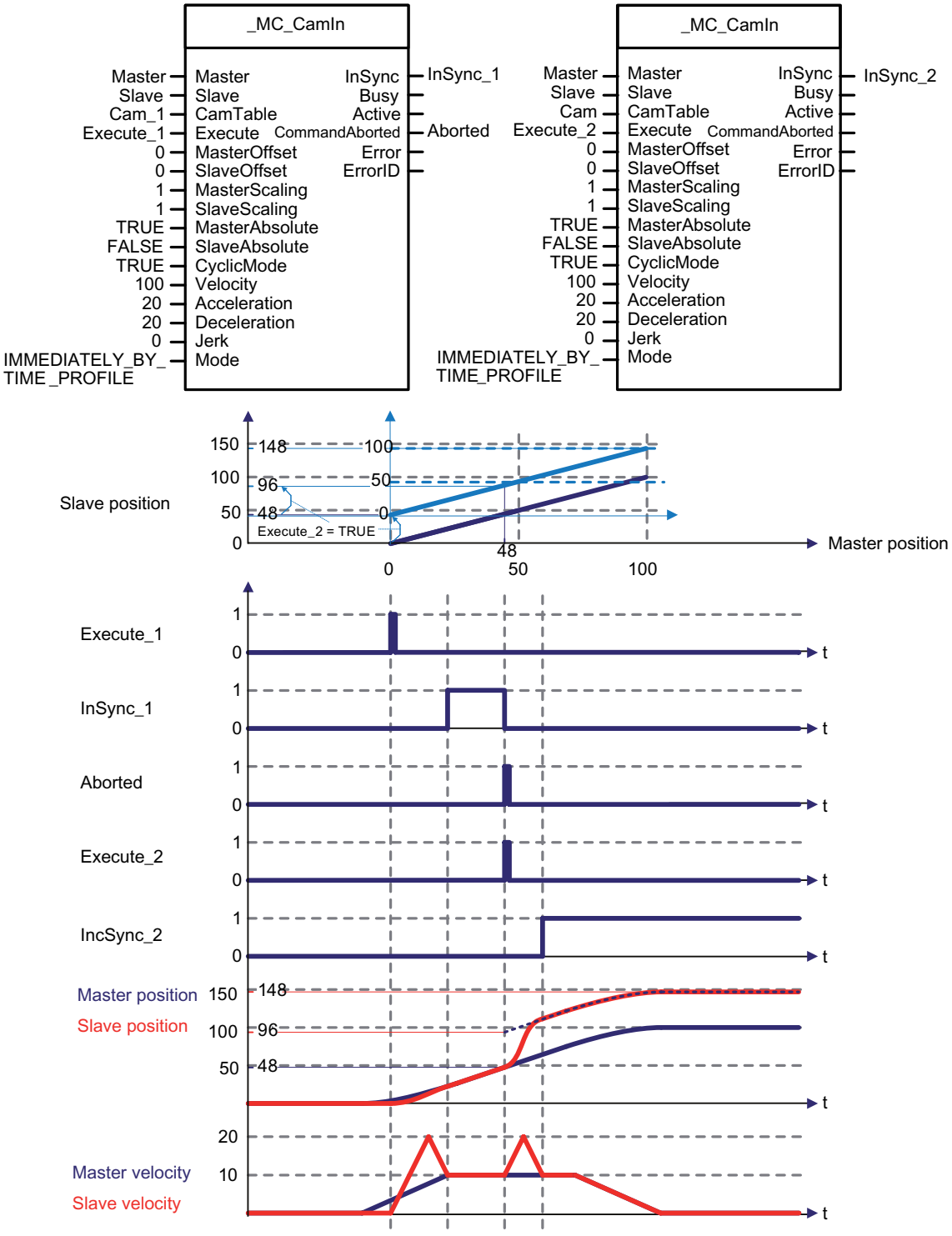


Figure 3-48 _MC_CamIn example: Camming with starting point shift to actual position of slave axis

Example: Camming with starting point shift to actual position of master axis

At the master axis, a positioning command is active at 100 mm. When the first block starts (Execute_1 = TRUE), the cam starting point is shifted to the actual position of the master axis (Pos_{Master} = 2 mm) (→ interpretation of master position relative to cam starting point). For this reason, the positional relationship between the slave and master positions must be corrected as follows:
(PosSlave = PosMaster) ⇒ Pos_{Slave} = Pos_{Master} – 2 mm.

This results in a 2 mm shift of the slave position in relation to the master position that is permanent pending further notice.

When the second block is started (Execute_2 = TRUE), the cam starting point is shifted repeatedly to the actual position of the master axis (Pos_{Master} = 48 mm). Therefore, the following equation applies for the second block:
(Pos_{Slave} = Pos_{Master} – 2 mm) ⇒ Pos_{Slave} = Pos_{Master} – 48 mm

In accordance with the positional relationship the position setpoint of the slave axis reset to zero, and in the figure below, is shifted by 48 mm in relation to the master position. Once the master position setpoint of 100 mm has been reached, the result is a position setpoint of 52 mm (100 mm – 48 mm = 52 mm) for the slave axis.

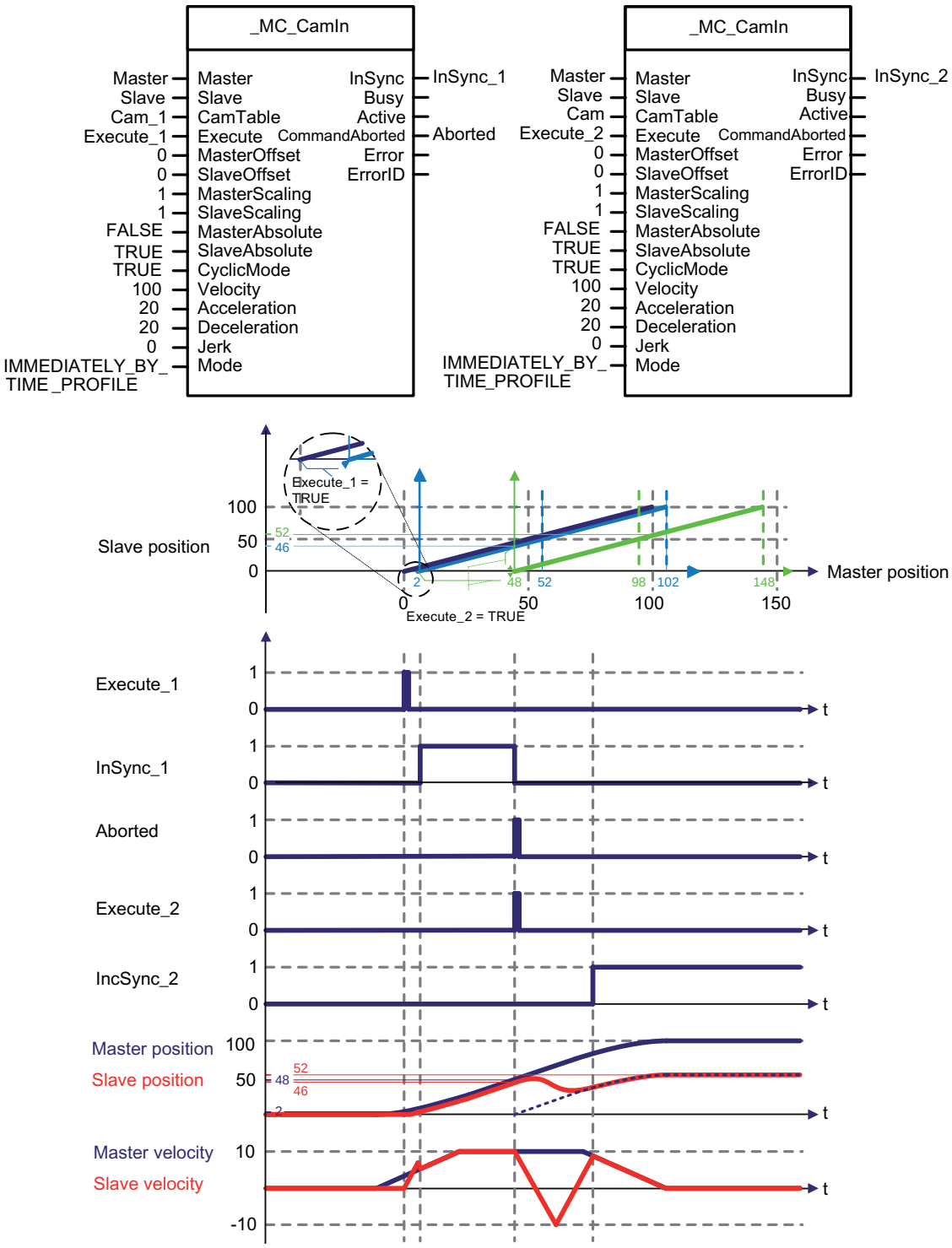


Figure 3-49 _MC_CamIn example: Camming with starting point shift to actual position of master axis

3.3.4 _MC_CamOut - Terminating camming

3.3.4.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

ErrorIDs

Example

3.3.4.2 Schematic diagram

Schematic diagram

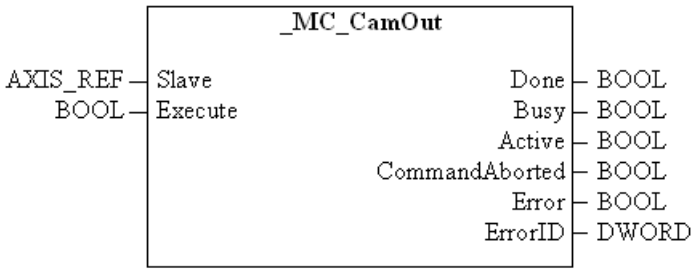


Figure 3-50 _MC_CamOut Schematic diagram

3.3.4.3 Purpose

Purpose

The function block **_MC_CamOut** terminates a camming and stops the slave axis. The slave axis is desynchronized in accordance with the presettings at the synchronous object in the "Preassignment > Camming" (→ system variables under **userdefault.cammingsettings…**) and "Preassignment > Dynamic Response" (→ system variables under **userdefault.syncdynamics…**) dialogs.

Recommendation

Use the function block when the shutdown procedure is to depend on the position of the master and/or the slave axis. You can also remove the slave axis from the synchronous operation with the technology functions **_MC_Stop**, **_MC_MoveRelative**, **_MC_MoveAdditive**, **_MC_MoveAbsolute** or **_MC_MoveVelocity**.

3.3.4.4 Applicable for

Applications

Following axes

Path axes with synchronous operation activated

3.3.4.5 Requirements

Requirements

A camming must be active on the slave axis. If no synchronous operation is active, the function block is aborted.

No **_MC_Stop** active on the slave axis

3.3.4.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Slave	AXIS_REF	0	Specification of reference to the slave axis (name of TO) The following technology objects can be homed: • Following axis (followingAxis data type) • Path axis with synchronous operation activated (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable The synchronous operation of the slave axis with the interconnected master is terminated with a rising edge on this input.

3.3.4.7 Output parameter

Output parameters

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Display of the completion of the function block With TRUE, the slave axis has been desynchronized from the interconnected master.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the slave axis.

Parameter	Data type	Initial value	Description
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 136)).

3.3.4.8 ErrorIDs

ErrorIDs

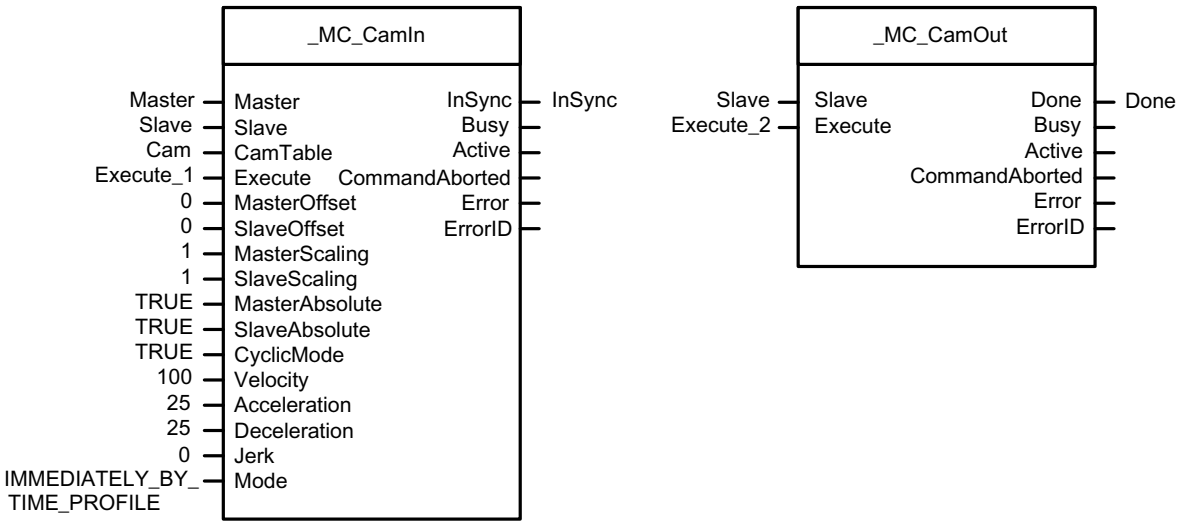
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.3.4.9 Example

At the master axis, a positioning command is active at 100 mm. The _MC_CamOut block terminates camming of the axis transferred on the **slave** input parameter.



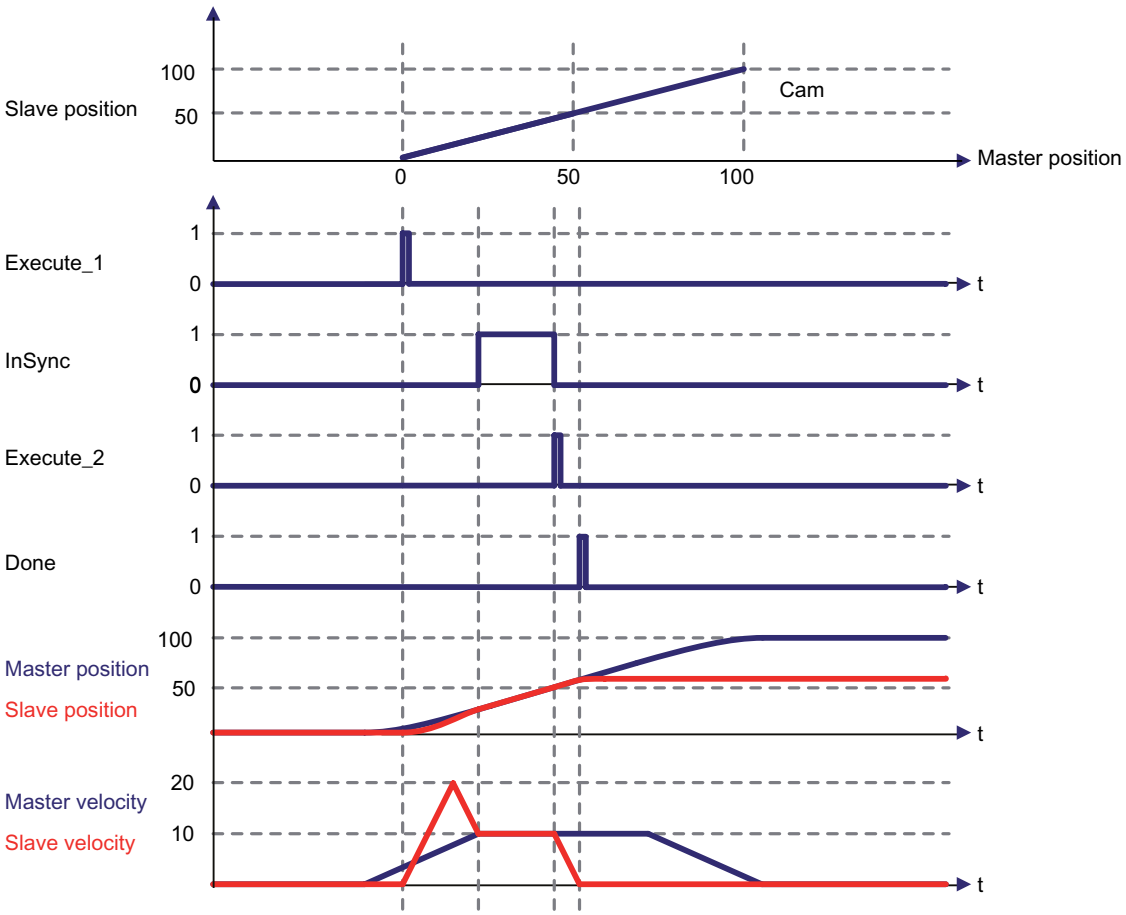


Figure 3-51 _MC_CamOut example:

3.3.5 _MC_Phasing - Changing phase shift between the leading axis and following axis

3.3.5.1 Overview

Schematic diagram
Purpose
Applicable for
Requirements
Input parameters
Output parameters
ErrorIDs
Example

3.3.5.2 Schematic diagram

Schematic diagram

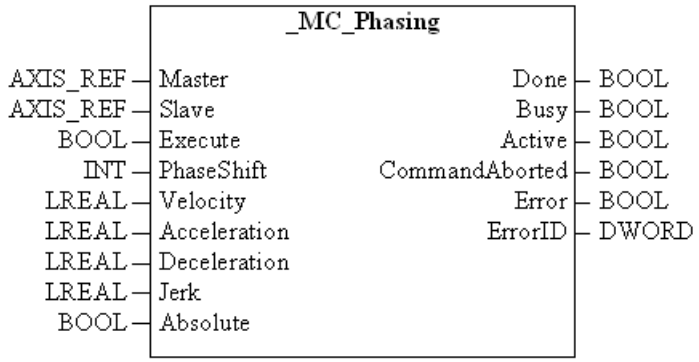


Figure 3-52 _MC_Phasing Schematic diagram

3.3.5.3 Purpose

Purpose

The function block **_MC_Phasing** shifts the position of the slave axis with respect to the master.

The effect on a following axis during camming equates to a horizontal cam shift.

The shift can be absolute or relative to the existing offsets.

The dynamic response parameters **Velocity**, **Acceleration**, **Deceleration** and **Jerk** define the dynamic response of the motion procedure.

3.3.5.4 Applicable for

Applications

Following axes operating during camming or gearing

Path axes with synchronous operation activated during camming or gearing

3.3.5.5 Requirements

Requirements

No **_MC_Stop** active on the slave axis

A camming or gearing must be active on the slave axis.

3.3.5.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Master	AXIS_REF	0	Specification of reference to the master (name of TO) This parameter presently has no function. The slave axis is always offset with respect to the current master of the active synchronous operation.
Slave	AXIS_REF	0	Specification of reference to the slave axis (name of TO) The following technology objects can be homed: • Following axis (followingAxis data type) • Path axis with synchronous operation activated (_pathAxis data type)
Execute	BOOL	FALSE	Function block enable With a rising edge on this input, the position of the slave axis is offset with respect to the position of the master.
PhaseShift	LREAL	1.0	Specification of the phase shift
Velocity	LREAL	-1.0	Specification of the maximum velocity Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.velocity system variable of the synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Acceleration	LREAL	-1.0	Specification of the maximum acceleration Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.positiveaccel system variable of the synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).

Parameter	Data type	Initial value	Description
Deceleration	LREAL	-1.0	Specification of the maximum deceleration Value > 0: The specified value is used Value = 0: Not permissible Value < 0: The preset value in the userdefault.syncdynamics.negativeaccel system variable of the synchronous object is used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Jerk	LREAL	-1.0	Specification of the maximum jerk Value > 0: The specified value is used Value = 0: Use trapezoidal velocity profile Value < 0: The preset values in the system variables userdefault.syncdynamics.positiveaccelstartjerk, userdefault.syncdynamics.positiveaccelendjerk, userdefault.syncdynamics.negativeaccelstartjerk, and userdefault.syncdynamics.negativeaccelendjerk of the synchronous object are used (see "Preassignment > Dynamic Response" dialog at synchronous object).
Absolute	BOOL	TRUE	Indicates the type of phase shift With TRUE, the phase shift is taken over as an absolute value. With FALSE, the phase shift is added to an existing offset.

3.3.5.7 Output parameter

Output parameters

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Indicates termination of the function block. With TRUE, the master and slave axis have been displaced with respect to one another by the absolute value of the phase shift.
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started.
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Display of the abort of the function block With TRUE, the function block has been aborted because of an error in the command processing or by an overriding command. The error description can be read at the **ErrorID** output.

Parameter	Data type	Initial value	Description
Error	BOOL	FALSE	Display of an error in the function block With TRUE, an error has occurred during the initialization of the function block. The function block is terminated. The error description can be read at the **ErrorID** output.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 141)).

3.3.5.8 ErrorIDs

ErrorIDs

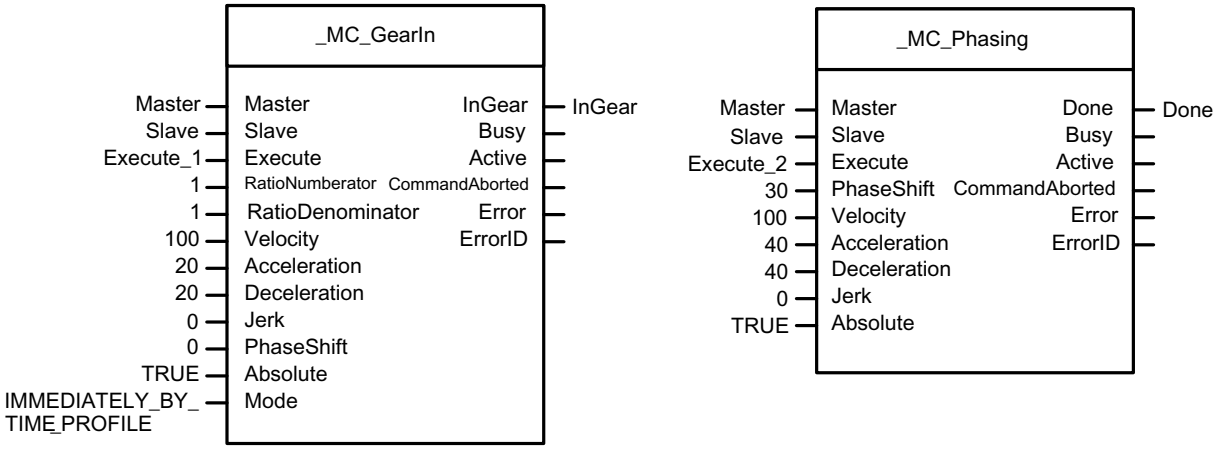
The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

See also

Troubleshooting - PLCopen Blocks (Page 151)

3.3.5.9 Example

At the master axis, a positioning command is active at 100 mm. The _MC_Phasing block shifts the position of the slave following axis transferred on the **slave** input parameter.



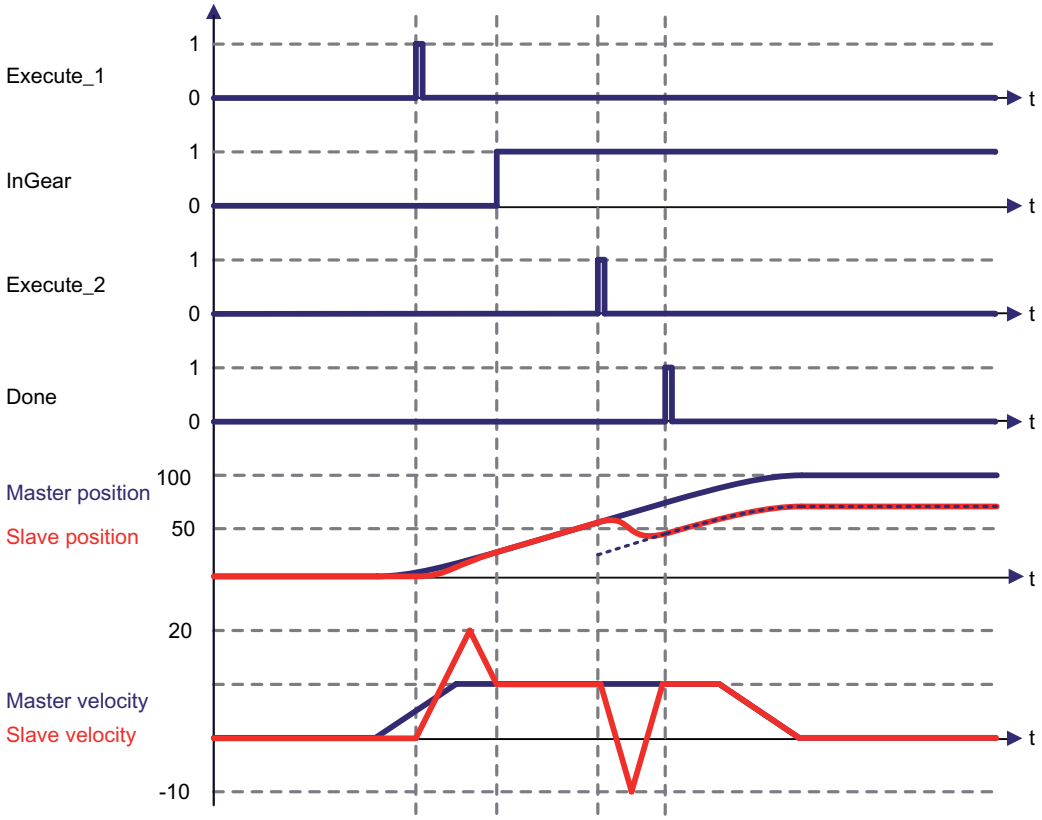


Figure 3-53 _MC_Phasing example

3.4 Advanced functions

3.4.1 _MC_Jog - Continuous or incremental jogging

3.4.1.1 Overview

Schematic diagram

Purpose

Applicable for

Requirements

Input parameters

Output parameters

Function

ErrorIDs

3.4.1.2 Schematic diagram

Schematic diagram

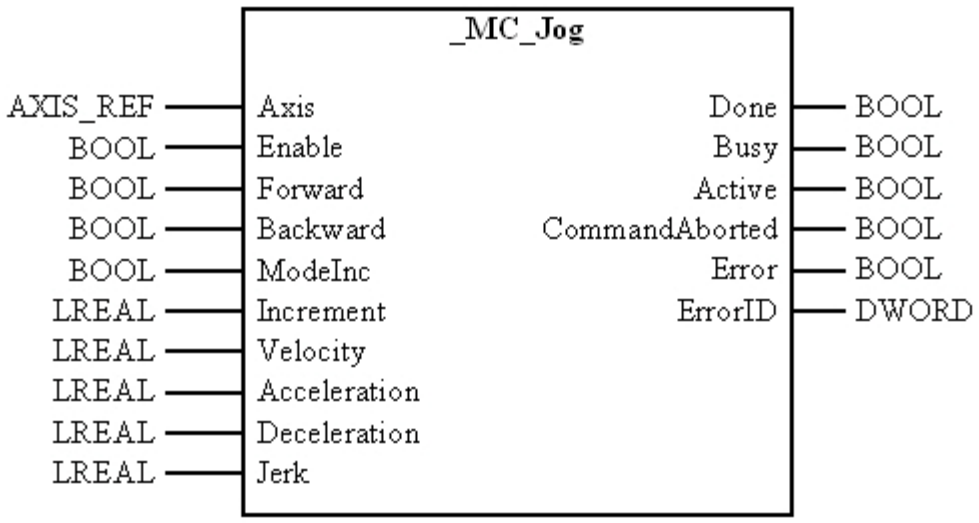


Figure 3-54 _MC_Jog Schematic diagram

3.4.1.3 Purpose

Purpose

The function block **_MC_Jog** implements a continuous or incremental jogging of an axis.

3.4.1.4 Applicable for

Applications

Positioning axes

Following axes

Path axes

3.4.1.5 Requirements

Requirements

Axis enabled

No **_MC_Stop** active

3.4.1.6 Input parameters

Input parameters

Parameter	Data type	Initial value	Description
Axis	AXIS_REF	0	Specification of axis reference (name of TO) The following technology objects can be homed: • Positioning axis (posAxis data type) • Following axis (followingAxis data type) • Path axis (_pathAxis data type)
Enable	BOOL	FALSE	Function block enable
Forward	BOOL	FALSE	Start of the traversing motion in positive direction with positive edge Stop of the traversing motion with negative edge TRUE = motion in positive direction as long as **Forward** is set
Backward	BOOL	FALSE	Start of the traversing motion in negative direction with positive edge Stop of the traversing motion with negative edge TRUE = motion in negative direction as long as **Backward** is set

Parameter	Data type	Initial value	Description
ModeInc	BOOL	FALSE	FALSE = continuous jogging TRUE = incremental jogging Each status change on the **ModeInc** results in an axis stop and deletion of the distance-to-go
Increment	LREAL	0.0	Value for incremental jogging
Velocity	LREAL	-1.0	Maximum velocity that is reached depending on the set traversing distance, acceleration and jerk (is not always reached) Value > 0: Use the specified value Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.velocity system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Acceleration	LREAL	-1.0	Acceleration (increasing energy in the motor): Value > 0: Use the specified value Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.positiveaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Deceleration	LREAL	-1.0	Deceleration (decreasing energy in the motor): Value > 0: Use the specified value Value = 0: Not permissible Value < 0: The preset value in the userdefaultdynamics.negativeaccel system variable of the axis is used (see "Preassignment > Dynamic Response" axis dialog).
Jerk	LREAL	-1.0	Specification of maximum jerk in conjunction with velocity profile definition Value > 0: Acceleration-constant velocity profile (SMOOTH); specified jerk is used Value = 0: Trapezoidal velocity profile (TRAPEZOIDAL) Value < 0: Type of velocity profile results from setting of axis system variable userdefaultdynamics.profile In the case of an effective, acceleration-constant velocity profile (SMOOTH), the preset values in the userdefaultdynamics.positiveaccelstartjerk, userdefaultdynamics.positiveaccelendjerk, userdefaultdynamics.negativeaccelstartjerk, and userdefaultdynamics.negativeaccelendjerk system variables are used. For information on the effects these have, refer to the "TO Axis Electric / Hydraulic, External Encoder" Function Manual; Command variable calculation > Velocity profiles.

3.4.1.7 Output parameter

Output parameter

Parameter	Data type	Initial value	Description
Done	BOOL	FALSE	Incremental or continuous jogging finished, the axis has been stopped
Busy	BOOL	FALSE	Display of the activity of the function block With TRUE, the function block has been started, distance-to-go pending
Active	BOOL	FALSE	Display of the command activity in the function block With TRUE, the command is being processed by the command processing, i.e., the function block has active control of the axis.
CommandAborted	BOOL	FALSE	Command has been aborted by another command or by an error during the command processing.
Error	BOOL	FALSE	Indicates an error **An error** has occurred during the command execution. The cause can be found in the **ErrorID**.
ErrorID	DWORD	0	Display of a function block error code The error code is always output in conjunction with the **CommandAborted** or **Error** output (see Chapter ErrorIDs (Page 149)).

3.4.1.8 Function

Function

The function block traverses the axis with the parameterized dynamic response values either continuously or incrementally in a positive or negative direction.

A positive edge at the input parameters **Forward** or **Backward** starts the traversing motion; it is stopped with the negative edge.

When incremental jogging is selected, the axis is stopped by the function block after traversing the parameterized distance. If the axle is stopped before reaching the parameterized distance, a restart of the motion completes the residual distance.

When incremental jogging is selected, the direction can only be changed when there is no remaining distance to be traversed.

- **Enable**
  - **Enable = TRUE**: Functionality of the block is processed and traversing motions are possible; with the transition FALSE/TRUE at Enable, all output parameters are reset and the distance-to-go is deleted.

    Note: Busy is only set with an action.
  - **Enable = FALSE**: Stop of the traversing motion - traversing motions no longer possible; with the transition from Enable = TRUE to Enable = FALSE, the **_MC_Stop** is always called internally, irrespective of whether the axis is in motion or not.
- **Busy/Active**
  - Incremental jogging:

    Busy = TRUE / Active = TRUE: Axis moves, distance-to-go != 0

    Busy = TRUE / Active = FALSE: Intermediate stop - the axis is stationary, distance-to-go != 0
  - Continuous jogging:

    After a positive edge at the input parameter **Forward** or **Backward**, **Busy = TRUE** is output in the same cycle. **Active** is only output in the interpolator after activation of the motion command. This may occur several cycles later.
- **Continuous jogging**
  - The jogging is started with a positive edge at **Forward** or **Backward**. The axis traverses with continuous jogging as long as the level on one of the two parameters is TRUE. The motion is stopped with the falling edge.
  - If the two inputs **Forward** and **Backward** are set to TRUE at the same time, no axis motion is started (**Active** = FALSE, **Busy** = FALSE, **Done** = FALSE), the distance-to-go is deleted and no error message is generated.
  - If the input parameter for the opposite direction is also set during the traversing motion, the traversing motion is stopped, the distance-to-go is deleted and also no error message is generated.
  - A positive edge at **Forward** and a negative edge at **Backward** in the same cycle results in a change of direction. The same applies for the reverse situation.

- **Incremental jogging**
  - With incremental jogging, the traversing motion is stopped when the distance parameterized in the input parameter **Increment** is reached. If a falling edge is detected at the input parameter **Forward** (or **Backward**) before this, the traversing motion is also stopped.
  - Behavior with distance-to-go present:

    If a distance-to-go is present, i.e., the traversing motion was stopped before reaching the distance parameterized at the input parameter **Increment** by a falling edge at **Forward** or **Backward**, this state is signaled with **Busy** = TRUE, **Done** = FALSE, **Active** = FALSE.

    With another positive edge in the same direction of motion, the present distance-to-go is traversed.

    Incremental jogging in the opposite direction with a stopped axis and distance-to-go present is not possible. To do this, the distance-to-go present must be deleted with selection of continuous jogging or **Enable** = 0/1 and incremental jogging selected again.
  - Behavior when distance-to-go is zero:

    The transition **Busy** = TRUE to FALSE, **Active** = TRUE to FALSE and **Done** = FALSE to TRUE signals the complete traversing of the selected increment. The distance-to-go is 0.
- **Switchover from incremental / continuous jogging**

  A change of **ModeInc** during traversing with an active traversing motion results in an axis stop without error message. If the input parameter **ModeInc** is changed to continuous jogging (**ModeInc**=FALSE), the distance-to-go is deleted.
- **Override**

  If the **Override** = 0 with active, started traversing motion, then there is no traversing motion and **Active** is TRUE.
- **Parameter acceptance**

  The parameter acceptance of the distances/**Increment** and the dynamic response parameters (**Velocity**, **Acceleration**, ...) is performed with a rising edge of **Forward**/**Backward**, i.e., a change of **Increment**, **Velocity**, **Acceleration**, **Deceleration** or **Jerk** has no effect during traversing.
- **Axis not in position control**

  If jogging is issued and the axis is not in position control, the corresponding error message of the used PLCopen FB is output.

3.4.1.9 ErrorIDs

ErrorIDs

The error code contains the number and, when available, the associated reason for the error that has occurred in the function block. The error number occupies the lower 16 bits of the error code (see Error codes of the errorID (LOW word) (Page 152)). The error reason, when available, is also coded as a number and occupies the upper 16 bits of the error code.

Note

ErrorIDs from other blocks possible

During execution of the function block **_MC_Jog**, the function blocks **_MC_MoveVelocity**, **_MC_MoveRelative** and **_MC_Stop** are called internally.
If the **_MC_Jog** is aborted, errorIDs can therefore also be reported from these three function blocks.

See also

Troubleshooting - PLCopen Blocks (Page 151)

Troubleshooting PLCopen blocks

4

4.1 Troubleshooting - PLCopen Blocks

The function blocks indicate that errors have occurred at output parameters **Error**, **ErrorID** , or **CommandAborted**.

Errors at output parameters Error or ErrorID

If output parameter **Error** = TRUE, this means that the function block was unable to initiate the command. The cause of the error is indicated by the value at output parameter **ErrorID**.

If the function block indicates an error, you need to call it either using the correct parameters or at a different time (provided that this function is permitted).

It is neither required nor possible to acknowledge the errors or warnings. The error remains active until input parameters **Enable** and **Execute** have been reset.

The error codes of the function blocks (errorID) have been taken over from the return values of the ST commands in SIMOTION and from the command abort reason. The return value of the ST commands is written to the LOW word of the errorID. The command abort reason is in the HIGH word of the errorID.

Indications at output parameter CommandAborted

Output parameter **CommandAborted** indicates that an active command, which was triggered during command execution, has been aborted by either a new command or an error.

- If no error is displayed in the **error** system variable of the technology object, this means that the command has been aborted by a subsequent command.
- If an error is indicated in output parameter **Error**, this means that an error affecting the technology object during command execution has caused the command to be aborted. You can find information on reading out the error status in the chapter titled "Query of general errors with the _MC_ReadAxisError function block".

Acknowledging errors at the technology object

1. Acknowledge all errors.

   To do this, first remove the causes of the errors and then acknowledge them using **_MC_Reset** (**Restart** = FALSE).

2. You can then re-enable the technology object using the **_MC_Power** function.

4.2 Error codes of the errorID (LOW word)

Error codes of the errorID (LOW word)

Error code		Description
Hex	dec	
0	0	No error
1	1	Illegal command parameter
2	2	Illegal range specification in command parameters
3	3	Command aborted
4	4	Unknown command
5	5	Command cannot be executed due to current object state
6	6	Command aborted due to termination of user task
7	7	Command rejected due to suspension of command interpretation of the addressed technology object
8	8	Command aborted due to full command buffer
9	9	Insufficient memory
A	10	A connection to a technology object required for this operation does not exist
B	11	No object configuration
C	12	The error to be reset cannot be reset due to its configuration
D	13	Axis is not homed
E	14	Measuring job on virtual axis not possible
F	15	Ambiguous **commandId**
10	16	Command not implemented
11	17	Read access denied
12	18	Write access denied
13	19	Command argument not supported
14	20	The cam has already been interpolated and cannot be manipulated
15	21	The interpolation condition was violated
16	22	The programmed jerk is 0
17	23	The alarm to be deleted is not pending
18	24	Command not possible on a virtual axis
19	25	A synchronized start of this command is not possible
1a	26	Superimposed command has been aborted as it is not permitted by the active command
1b	27	Time-out during communication with the drive
1c	28	Actual values are not valid
1d	29	The command cannot be executed when velocity control is active
1e	30	The command cannot be executed when position control is active
1f	31	The command cannot be executed in torque-reduced operation or during travel to fixed end stop

Error code		Description
20	32	The command can only be executed when force or pressure control is active
21	33	The command cannot be executed when force/pressure control is active
22	34	The command can only be executed when pressure limiting is active
23	35	Master values are not valid
24	36	Slave values are not valid
25	37	No slave value can be defined for a master value
26	38	No master value can be defined for a slave value
27	39	The command cannot be executed when synchronous operation is inactive
28	40	The command cannot be executed with non-synchronous operation
29	41	The command cannot be executed when gearing or camming is active
2a	42	The command cannot be executed when camming is inactive
2b	43	This command can only be used for an interpolated cam
2c	44	The command can only not be executed when pressure limiting is active
2d	45	Insufficient interpolation points are available to interpolate the cam
2e	46	The specified path point cannot be reached due to restrictions of the kinematics
2f	47	Path axis values are not valid
30	48	Reserved memory is full
2710	10000	(Greater than or equal to) internal error

See also

Troubleshooting - PLCopen Blocks (Page 151)

4.3 Command abort reason of the errorID (HIGH word)

Command abort reason of the errorID (HIGH word)

Command abort reason		Description
Hex	dec	
0	0	No abort reason
1	1	Reset of the command buffer
2	2	Abort by another command
3	3	Abort by a stop
4	4	Abort by a higher-order stop
5	5	Abort by a pending error response
6	6	Abort due to ambiguous **commandId**
7	7	Acknowledgement delay
8	8	No actual value for axis (e.g., encoder or data bus not ready)

Command abort reason		Description
9	9	Abort due to abort of a dependent command
a	10	Abort due to active synchronous operation
b	11	Abort due to active superimposed motion
c	12	Abort due to active speed-controlled controller mode
d	13	Abort due to active position-controlled controller mode
e	14	Abort due to active travel to fixed end stop
f	15	Axis is not in pressure-limiting operation
10	16	Abort due to active pressure-controlled operation
11	17	Abort due to inactive pressure-controlled operation
12	18	Superimposed command is not permitted
13	19	Abort due to error during cam access
14	20	Slave is not ready for operation
15	21	Error in the slave synchronization
16	22	Abort by command on the slave
17	23	Abort by stop on the slave
18	24	Abort by a pending error response on the slave
19	25	No actual values for slave (e.g., encoder or data bus not ready)
1a	26	Abort by reset on the slave
1b	27	Master values are not valid
1c	28	Active command in recursive TO interconnection
1d	29	Abort due to error during synchronization
1e	30	Axis is in pressure-limiting operation
1f	31	Maximum number of active commands exceeded
20	32	Abort due to active correction command
21	33	Action only permissible in standstill
22	34	Path axis is not ready for operation
23	35	Error during synchronization with a path axis
24	36	Abort due to command on a path axis
25	37	Abort due to stop on a path axis
26	38	Abort due to a pending error response on a path axis
27	39	No actual values for path axis (e.g., encoder or data bus not ready)
28	40	Abort due to reset on a path axis
29	41	Command parameters became invalid during processing
2a	42	A required connection to a technology object does not exist
2b	43	Abort through user program

See also

Troubleshooting - PLCopen Blocks (Page 151)

4.4 Query of general errors with the _MC_ReadAxisError function block

General errors on the **axis**, **external encoder**, **synchronous object**, and **cam** can be queried using the **_MC_ReadAxisError** function block. The function block converts the alarms pending on the technology object into a 32-bit representation, which is output at the AxisErrorID parameter.
The alarm numbers of each technology object are assigned to specific alarm groups. In this respect, some alarm groups are defined for all technology objects, while others are TO-specific. A bit in the "errorGroup" system variable is assigned to each alarm group. If a bit is set here, an alarm from this group is pending at the technology object.

Note

Which alarm is assigned to which alarm group for a technology object is fixed and cannot be changed (reconfigured).

It is possible to use SIMOTION SCOUT in the execution system of a device in order to read which alarm group an individual alarm is assigned to. Pressing the "Alarm configuration" button produces the relevant information for each technology object that has been created, under "ExceptionFaultTask" (last column in window that appears).

The tables below indicate which bit numbers are assigned to which alarm groups for the individual technology objects.

Meaning of the bit numbers for the axis

Bit number	Meaning (alarm group)
0	System fault
1	Config fault
2	User fault
3	Peripheral fault
4	Function fault
5	Function aborted
6	Reset/restart fault
7	Distributed motion fault
8	Following error
9	Standstill positioning error
10	Dynamic limit
11	Clamping error
12	Software limit
13	Limit switch
14	Sensor fault
15	Reference not found
16	Output limit
17	Force dynamic limit
18	Additional sensor fault
19	Synchronous motion fault

Bit number	Meaning (alarm group)
20	Following object
21	Path synchronous motion fault
22	Path motion fault
23	Path object

Meaning of the bit numbers for the external encoder

Bit number	Meaning (alarm group)
0	System fault
1	Config fault
2	User fault
3	Peripheral fault
4	Function fault
5	Function aborted
6	Reset/restart fault
7	Distributed motion fault
9	Standstill positioning error
14	Sensor fault
15	Reference not found
19	Synchronous motion fault

Meaning of the bit numbers for the following object

Bit number	Meaning (alarm group)
0	System fault
1	Config fault
2	User fault
3	Peripheral fault
4	Function fault
5	Function aborted
6	Reset/restart fault
7	Distributed motion fault
8	Direct sync
9	Stability
10	Dynamic limit
11	Value ambiguous

Meaning of the bit numbers for the cam

Bit number	Meaning (alarm group)
0	System fault
1	Config fault
2	User fault
3	Peripheral fault
4	Function fault
5	Function aborted
6	Reset/restart fault
8	Value not valid
9	Interpolation fault

Alarms and, therefore, error states, which switch the controller to STOP mode (e.g. 20001 **Internal error**), can be evaluated by the **_MC_ReadAxisError** block in the shutdown task. In addition to the 32-bit error representation, the TO alarms are also output.

See also

Troubleshooting - PLCopen Blocks (Page 151)

Index

_

_MC_CamIn, 119
  Purpose, 120
  Schematic diagram, 119
_MC_CamOut, 134
  Purpose, 134
  Schematic diagram, 134
_MC_GearIn, 106
  Purpose, 107
  Schematic diagram, 106
_MC_GearOut, 115
  Purpose, 115
  Schematic diagram, 115
_MC_Home, 29
  Purpose, 29
  Schematic diagram, 29
_MC_Jog, 143
  Purpose, 144
  Schematic diagram, 143
_MC_MoveAbsolute, 36
  Purpose, 36
  Schematic diagram, 36
_MC_MoveAdditive, 51
  Purpose, 51
  Schematic diagram, 51
_MC_MoveRelative, 41
  Purpose, 41
  Schematic diagram, 41
_MC_MoveSuperimposed, 57
  Purpose, 57
  Schematic diagram, 57
_MC_MoveVelocity, 46
  Purpose, 46
  Schematic diagram, 46
_MC_Phasing, 138
  Purpose, 138
  Schematic diagram, 138
_MC_PositionProfile, 62
  Purpose, 62
  Schematic diagram, 62
_MC_Power, 19
  Purpose, 19
  Schematic diagram, 19
_MC_ReadActualPosition, 77
  Purpose, 77
  Schematic diagram, 77
_MC_ReadAxisError, 86
  Purpose, 86
  Schematic diagram, 86
_MC_ReadBoolParameter, 94
  Purpose, 94
  Schematic diagram, 94
_MC_ReadParameter, 90
  Purpose, 90
  Schematic diagram, 90
_MC_ReadStatus, 81
  Purpose, 82
  Schematic diagram, 81
_MC_Reset, 72
  Purpose, 72
  Schematic diagram, 72
_MC_Stop, 24
  Purpose, 24
  Schematic diagram, 24
_MC_VelocityProfile, 67
  Purpose, 67
  Schematic diagram, 67
_MC_WriteBoolParameter, 102
  Purpose, 102
  Schematic diagram, 102
_MC_WriteParameter, 98
  Purpose, 98
  Schematic diagram, 98

P

PLCopen, 15
Purpose
  _MC_CamIn, 120
  _MC_CamOut, 134
  _MC_GearIn, 107
  _MC_GearOut, 115
  _MC_Home, 29
  _MC_Jog, 144
  _MC_MoveAbsolute, 36
  _MC_MoveAdditive, 51
  _MC_MoveRelative, 41
  _MC_MoveSuperimposed, 57
  _MC_MoveVelocity, 46
  _MC_Phasing, 138
  _MC_PositionProfile, 62
  _MC_Power, 19

_MC_ReadActualPosition, 77
_MC_ReadAxisError, 86
_MC_ReadBoolParameter, 94
_MC_ReadParameter, 90
_MC_ReadStatus, 82
_MC_Reset, 72
_MC_Stop, 24
_MC_VelocityProfile, 67
_MC_WriteBoolParameter, 102
_MC_WriteParameter, 98

R

References, 3

S

Schematic diagram
_MC_CamIn, 119
_MC_CamOut, 134
_MC_GearIn, 106
_MC_GearOut, 115
_MC_Home, 29
_MC_Jog, 143
_MC_MoveAbsolute, 36
_MC_MoveAdditive, 51
_MC_MoveRelative, 41
_MC_MoveSuperimposed, 57
_MC_MoveVelocity, 46
_MC_Phasing, 138
_MC_PositionProfile, 62
_MC_Power, 19
_MC_ReadActualPosition, 77
_MC_ReadAxisError, 86
_MC_ReadBoolParameter, 94
_MC_ReadParameter, 90
_MC_ReadStatus, 81
_MC_Reset, 72
_MC_Stop, 24
_MC_VelocityProfile, 67
_MC_WriteBoolParameter, 102
_MC_WriteParameter, 98