

IndraDrive

IndraMotion MLD (2G)
Libraries as of MPx-18

Reference Book
R911338916

Edition 06



Title	IndraDrive IndraMotion MLD (2G) Libraries as of MPx-18
Type of Documentation	Reference Book
Document Typecode	DOK-INDRV*-MLD-SYSLIB3-RE06-EN-P
Internal File Reference	RS-86e472c00f785de60a6846a5017a43c8-6-en-US-4
Purpose of Documentation	This documentation describes the PLC functions/function blocks and data types provided by IndraMotion MLD as well as a list of possible error reactions. The documentation applies to the following firmwares: • MPx-18 and above • PSB-19 and above
Edition	Edition 06, 2018-10 Record of revisions see ""Editions of this documentation" on page 7"
Copyright	© Bosch Rexroth AG 2018 All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
Liability	The specified data is intended for product description purposes only and shall not be deemed to be a guaranteed characteristic unless expressly stipulated in the contract. All rights are reserved with respect to the content of this documentation and the availability of the product.
Editorial Department	DC-AE/EPI5 (SA), DC-AE/EME1 (ML), DC-AE/EFC2 (KA); DC-AE/EPI4 (BB)

Table of Contents

	Page
1 About this documentation.....	7
2 General information.....	11
3 Libraries.....	13
3.1 Supported libraries.....	13
3.2 Dependencies between axis parameterization, functional package and selected function block.....	14
3.3 Function blocks for PSB firmware derivative.....	15
4 Signal Time Diagrams of Most Frequently Used Inputs and Outputs.....	17
5 Error handling.....	23
5.1 General information.....	23
5.2 Error reference lists.....	24
6 Data types.....	31
7 Parameter structure.....	33
7.1 Structure.....	33
7.2 Sercos attribute.....	33
8 IecSfc.....	35
9 MX_Base.....	37
9.1 Introduction.....	37
9.2 Data types.....	37
9.3 Global Variable List.....	38
10 MX_CanL2.....	41
10.1 Introduction.....	41
10.2 MX_CANL2_FifoReceive.....	41
10.3 MX_CANL2_FifoSend.....	42
10.4 MX_CANL2_Init.....	42
10.5 MX_CANL2_SetLED.....	44
10.6 MX_fCANL2_MBoxReceive.....	45
10.7 MX_fCANL2_MBoxSend.....	46
10.8 MX_fCANL2_Status.....	46
10.9 MX_fCANL2_MBoxRecStatus.....	47
10.10 MX_fCANL2_MBoxSndStatus.....	47
10.11 MX_fCANL2_MBoxUpdate.....	48
10.12 MX_fCANtoHI.....	49

Table of Contents

	Page
10.13 MX_fCANtoHs.....	49
10.14 MX_fHtoCANI.....	50
10.15 MX_fHtoCANS.....	50
 11 MX_CheckRtv.....	 53
 12 MX_Debug.....	 55
 13 MX_PLCOpen.....	 57
13.1 Introduction.....	57
13.2 "Drive control" - Functions for Diagnostics.....	57
13.2.1 MX_fCheckCmdRequest.....	57
13.2.2 MX_fGetDriveWarning.....	57
13.2.3 MX_fSetDriveError.....	58
13.2.4 MX_fSetDriveErrorText.....	58
13.2.5 MX_fSetDriveWarning.....	59
13.2.6 MX_fSetDriveWarningText.....	59
13.2.7 MX_SetCmdState.....	60
13.3 "Motion" - Function blocks/functions for drive control.....	61
13.3.1 General properties.....	61
13.3.2 MB_ChangeCamData.....	61
13.3.3 MB_ChangeProfileSet	64
13.3.4 MB_ChangeProfileStep.....	66
13.3.5 MB_ClearAllError.....	69
13.3.6 MB_GearInPos.....	70
13.3.7 MB_Home.....	75
13.3.8 MB_MotionProfile.....	76
13.3.9 MB_Phasing.....	79
13.3.10 MB_PhasingSlave.....	81
13.3.11 MB_PreSetMode.....	83
13.3.12 MB_SetPositionControlMode.....	84
13.3.13 MB_Stop.....	86
13.3.14 MC_CamIn.....	88
13.3.15 MC_CamOut.....	91
13.3.16 MC_GearIn.....	92
13.3.17 MC_GearOut.....	94
13.3.18 MC_Jog.....	95
13.3.19 MC_MoveAbsolute.....	98
13.3.20 MC_MoveAdditive.....	100
13.3.21 MC_MoveRelative.....	103
13.3.22 MC_MoveVelocity.....	105
13.3.23 MC_Power.....	108
13.3.24 MC_ReadStatus.....	109
13.3.25 MC_Reset.....	111
13.3.26 MC_Stop.....	112

Table of Contents

	Page
13.3.27 MC_TorqueControl.....	114
13.3.28 MX_MoveAbsolute.....	116
13.3.29 MX_MoveAdditive.....	118
13.3.30 MX_MoveRelative.....	121
13.3.31 MX_SetControl.....	124
13.3.32 MX_SetDeviceMode.....	126
13.3.33 MX_SetOpMode.....	127
13.4 "Parameters" - Function blocks/functions for parameters.....	130
13.4.1 MB_Command.....	130
13.4.2 MB_ReadListParameter.....	135
13.4.3 MB_ReadMaxRealValue.....	137
13.4.4 MB_ReadMaxValue.....	139
13.4.5 MB_ReadMinRealValue.....	140
13.4.6 MB_ReadMinValue.....	142
13.4.7 MB_ReadName.....	143
13.4.8 MB_ReadParameter.....	145
13.4.9 MB_ReadRealListParameter.....	146
13.4.10 MB_ReadRealParameter.....	148
13.4.11 MB_ReadSercosAttribute	150
13.4.12 MB_ReadSercosDataStatus.....	151
13.4.13 MB_ReadStringParameter.....	153
13.4.14 MB_ReadUnit.....	155
13.4.15 MB_WriteListParameter	156
13.4.16 MB_WriteParameter.....	159
13.4.17 MB_WriteRealListParameter	160
13.4.18 MB_WriteRealParameter.....	163
13.4.19 MB_WriteStringParameter.....	165
13.4.20 MX_Command.....	167
13.4.21 MX_fReadParamDINT.....	170
13.4.22 MX_fReadStringParam.....	170
13.4.23 MX_fSetParamLimits.....	171
13.4.24 MX_fSetParamName.....	172
13.4.25 MX_fSetParamUnit.....	172
13.4.26 MX_fWriteParamDINT.....	173
13.4.27 MX_fWriteStringParam.....	174
13.4.28 MX_ReadParamDINT.....	175
13.4.29 MX_WriteParamDINT.....	176
13.5 "Scaling" - Functions for Scaling.....	178
13.5.1 MX_fDINT_AccTo_REAL.....	178
13.5.2 MX_fDINT_DistTo_REAL	178
13.5.3 MX_fDINT_VelTo_REAL	179
13.5.4 MX_fREAL_AccTo_DINT	179
13.5.5 MX_fREAL_DistTo_DINT	180
13.5.6 MX_fREAL_TorqueRampTo_DINT	180
13.5.7 MX_fREAL_TorqueTo_DINT	181
13.5.8 MX_fREAL_VelTo_DINT	181

Table of Contents

	Page
13.6 "Tools" - General functions.....	182
13.6.1 MX_fGetFreeTicks.....	182
13.6.2 MX_fGetHighResTime.....	183
13.6.3 MX_fHighResTimerTicks_to_us.....	183
13.6.4 MX_IECTaskGetLoad.....	183
13.6.5 MX_PositionLoopEventInfo.....	184
14 Basic libraries.....	185
14.1 MX_CommonTypes.....	185
14.1.1 Introduction.....	185
14.1.2 Global variables.....	186
14.1.3 Sercos attributes.....	186
14.1.4 Data types of logical addresses.....	186
14.1.5 Data types of PLCOpen.....	187
14.1.6 "ERROR_STRUCT" structure.....	193
14.1.7 Data types to return the date and the absolute time specifications in IndraMotion function blocks.....	193
14.1.8 Data types to support probe functions.....	195
14.1.9 Data types to support cams, MotionProfiles and FlexProfiles.....	200
14.2 RIL_HMI_Uilities.....	223
14.3 RIL_LoopControl.....	224
14.4 RIL_ModbusTCP.....	224
14.5 RIL_SercosIII.....	224
14.5.1 General information.....	224
14.6 RIL_SocketComm.....	224
14.7 RIL_Uilities.....	224
14.8 RMB_SercosIII_Util.....	225
15 System libraries.....	227
15.1 General information.....	227
15.2 Standard.....	227
15.3 Util.....	227
15.4 SysDir.....	227
15.4.1 General information.....	227
15.4.2 Instructions for use.....	227
15.5 SysFile.....	228
15.5.1 General information.....	228
15.5.2 Instructions for use.....	228
15.6 SyslecTasks23.....	228
15.7 SysMem.....	229
15.8 SysCallback23.....	229
15.9 SysSem.....	229
15.10 SysSockets.....	229
15.11 SysStr23	230
15.11.1 Overview.....	230

Table of Contents

	Page
15.11.2 SysStrCmp	230
15.11.3 SysStrCmpl	230
15.11.4 SysStrCmpN	230
15.11.5 SysStrCmpNI	231
15.11.6 SysStrCpy	231
15.11.7 SysStrLen	231
15.12 SysTimeRTC.library.....	232
15.12.1 Introduction.....	232
15.12.2 SysTimeRtcGet.....	232
15.12.3 SysTimeRtcSet.....	232
15.12.4 SysTimeRtcHighResGet.....	233
15.12.5 SysTimeRtcHighResSet.....	233
15.13 IoDrvEtherNetIP.library.....	233
15.13.1 Licensing and operation.....	234
15.13.2 MLD performance and cycle times.....	234
15.13.3 Resource requirement in the boot project.....	234
15.13.4 Supported interfaces and coexistence with Sercos.....	235
15.13.5 Requirements on the IP network.....	235
15.13.6 Restrictions of the functional scope of the library.....	236
15.13.7 EtherNet/IP scanner configuration.....	237
15.13.8 Editors for the Ethernet/IP configuration.....	239
15.13.9 Acyclic communication (Explicit Messaging Service).....	245
16 Technology libraries.....	249
16.1 Introduction.....	249
16.2 MX_TechInterface.....	249
16.2.1 Overview.....	249
16.2.2 Using the axis interface in the IndraMotion MLD system.....	249
16.3 MX_TechBase.....	252
16.4 MX_TechMotionBase.....	252
16.5 MX_TechMotion.....	252
16.6 MX_TechWinder.....	252
16.7 MX_TechCrosscutCrossseal	253
16.8 RMB_TechCam.....	253
17 Free PLC library from "OSCAT".....	255
Index.....	257

1 About this documentation

Editions of this documentation

Edition	Release date	Notes
01	2013-01	Contains descriptions of libraries for the "IndraDrive MP18" target
02	2013-11	Corrections and additions
03	2014-07	- Added "MX_TechInterface" library - Corrections and additions
04	2015-10-09	Contains descriptions of libraries for the "IndraDrive MP20" target - Added "SysTimeRTC" library - Added "MX_TechMotionBase", "MX_TechBase", "MX_TechMotion", "MX_TechWinder" technology libraries - Corrections and additions - Extended description of "MX_Base" - Extended description of "MX_CommonTypes"
05	2016-08-11	- MPx-20 and above: New libraries "IoDrvEtherNetIP" "RMB_SercosIII_Util" "MX_TechCrosscutCrossseal" technology library - MPx-20 and above: In case of synchronization function blocks, VmAxis1, RmAxis1, LinkAxis1 can also be used as master axis in MLD-S
06	2018-10-31	- Added error table for "MX_CANL2_Init" - Adjusted "Sercos" spelling - Data type of pResult in "SysTimeRtcGet" corrected in RTS_IEC_RESULT - Included notes under "MX_PLCOpen.library", "Parameters": - Long runtime for write/read access to some parameters - If parameters are write-protected by a password, it is also impossible to write parameter values using functions or function blocks - Added "RMB_TechCam" technology library

Tab. 1-1: Record of revisions

Means of representation in this documentation

To facilitate reading of this documentation, the table below contains the means of representation and notations of recurring terms.

What?	How?	For example...
Paths and menu sequences...	...appear step by step in boldface	Click Window ► Show view ► Properties
Buttons and keys...	...appear in angle brackets	Click <Add> or Press <Ctrl>+<Alt>

About this documentation

What?	How?	For example...
Important facts which are to be highlighted in the body text	Boldface	For remote axes , "Active"=TRUE signals that ...
Parameter names, diagnostic message names, function designations	Quotation marks	"Axis2" to "Axis10" correspond to the remote axes. The remote axes contained in the list "P-0-1801.x.10, CCD: Addresses of projected drives" ...

Tab. 1-2: Conventions of notation

All important notes are highlighted. A symbol tells you what kind of information is used in the text. The symbols have the following significances:

⚠ DANGER

...

DANGER indicates a hazardous situation which, if not avoided, **will** result in death or serious injury.

⚠ WARNING

...

WARNING indicates a hazardous situation which, if not avoided, **could** result in death or serious injury.

⚠ CAUTION

...

In case of non-compliance with this safety instruction, minor or moderate injury **could** occur.

NOTICE

...

In case of non-compliance with this safety instruction, property damage **could** occur.



This box contains important information that should be taken into consideration.



This symbol highlights useful tips and tricks.

Your Feedback

Your experience is important for our improvement processes of products and documentations.

If you discover mistakes in this documentation or suggest changes, you can send your feedback to the following e-mail address:

Dokusupport@boschrexroth.de

We need the following information to handle your feedback:

- The number indicated under "Internal File Reference".
- The page number.

Documentation as online help

This documentation has been included in the IndraDrive help system.

About this documentation

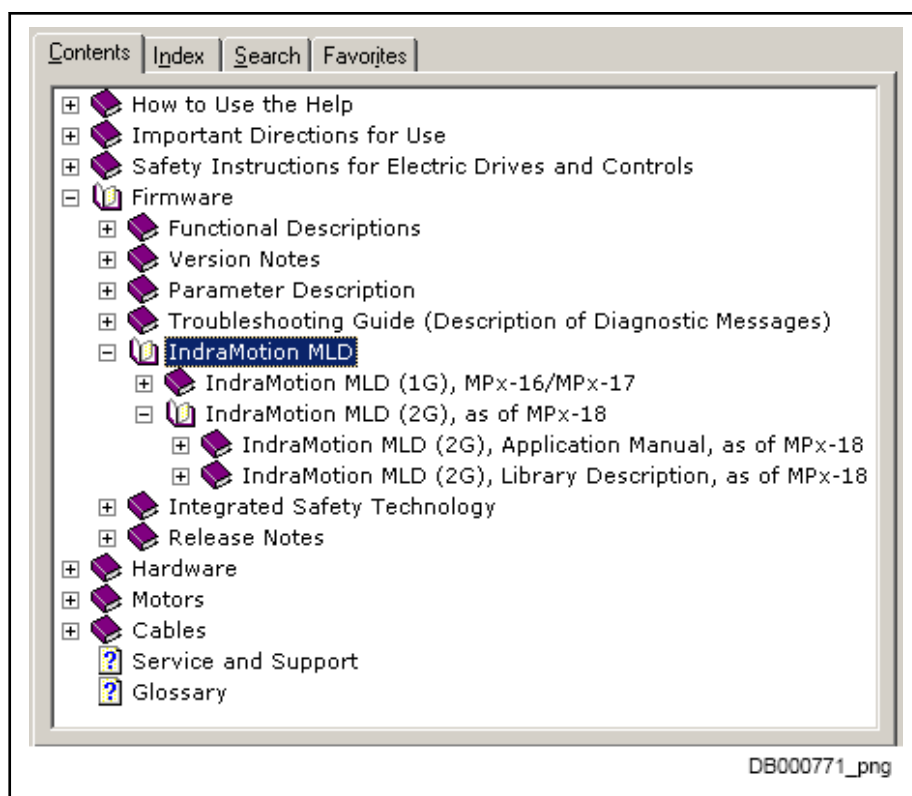


Fig. 1-1: Description of "Rexroth IndraDrive Firmware Libraries" in the IndraDrive help system

Reference documentations

Title	Type of documentation	Document typecode ¹⁾	Material number
IndraDrive IndraMotion MLD as of MPx18	Application Manual	DOK-INDRV*-MLD3-**VRS*-APRS-EN-P	R911338914
IndraWorks 14VRS IndraLogic 2G PLC Programming System	Application Manual	DOK-IWORKS-IL2GPRO*V14-APxx-EN-P	R911343571
IndraWorks 14VRS Basic Libraries IndraLogic 2G	Library	DOK-IL*2G*-BASLIB**V14-Llxx-EN-P	R911343920
IndraWorks 11VRS PLC System Libraries IndraLogic 2G	Library	DOK-IL*2G*-SYSLIB**V11-Llxx-EN-P	R911336157

About this documentation

Title	Type of documentation	Document typecode ¹⁾	Material number
IndraMotion MLC 14VRS Technology Libraries	Library	DOK-MLC***-TF*LIB**V14-LIxx-EN-P	R911341511
IndraWorks 14VRS Field Bus Libraries	Library	DOK-IWORKS-FB*LIB**V14-LIxx-EN-P	R911343575

1) In the document typecodes, "xx" is a placeholder for the current edition of the documentation (e.g.: AP01 is the first edition of an Application Manual)

Tab. 1-3: Documentations – overview

2 General information

The IndraLogic 2G PLC programming system allows PLC components to be programmed and commissioned for an IndraLogic XLC / IndraMotion MLC, IndraMotion MLD or IndraMotion MTX control, or supported by internally used visualization. IndraLogic supports the management of different libraries.

See the appropriate Application Manual for the hardware requirements for using drive-integrated "IndraMotion MLD" controls:

Description	Title	Document typecode ¹⁾	Material number
Firmwares FWA-INDRV-MP*02VRS to FWA-INDRV-MP*08VRS	Rexroth IndraDrive Rexroth IndraMotion MLD	DOK-INDRV*-MLD-**VRS*- AWxx-EN-P	R911306084
Firmware FWA-INDRV-MP*17VRS	Rexroth IndraDrive Rexroth IndraMotion MLD as of MPx-17	DOK-INDRV*-MLD2-**VRS*- APxx-EN-P	R911334351
As of firmware FWA-INDRV-MP*18VRS	IndraDrive IndraMotion MLD as of MPx-18	DOK-INDRV*-MLD3-**VRS*- APxx-EN-P	R911338914

1) In the document typecodes, "xx" is a placeholder for the current edition of the documentation (e.g.: AP01 is the first edition of an Application Manual)

Tab. 2-1: Documentations – overview

In addition, one of the functional expansion packages "IndraMotion MLD" has to be enabled (see also Functional Description of firmware "Enabling functional packages").

When a project is created with an IndraDrive target, some Rexroth libraries are automatically included in the project. The library manager allows subsequently adding libraries to a project to use their functions.

Libraries can contain both function blocks and declarations and/or variables. The drive-integrated PLC (Rexroth IndraMotion MLD) comes with basic libraries containing essential function blocks and definitions/variables.

Depending on the type of library, IEC functions/function blocks can be created or firmware function blocks can be defined.

The IEC code is contained in so-called "internal libraries".



The documentation often only refers to the drive/drive controller or the axis, since the functions have originally been developed for drive controllers (MPx firmware). In many cases, the functions can also be used in the PSB firmware (IndraDrive ML as supply units).

3 Libraries

3.1 Supported libraries

NOTICE

Property damage caused when non-supported libraries are included.

Only the libraries listed below are allowed to be included.

Library	Description
IecSfc	Provides IEC steps conforming to standard in "Sequential Function Chart" (SFC)
IoDrvEtherNetIP	An EtherNet/IP scanner is provided. Using this scanner, other EtherNet/IP devices can be integrated as slaves (e.g., barcode reader, RFID, flow rate meter, laser distance meter, I/O, HMI,...).
MX_Base (IndraDrive) / MY_Base (HydraulicDrive)	• Data types or structures • Cyclic parameters as direct variables (system-wide variables) • Axis structures (for a multi-axis system [MLD system mode])
MX_CanL2 (IndraDrive only)	Contains data types, functions and function blocks for CAN communication on level 2
MX_CommonTypes	Contains data types and structures most of which are only used internally
MX_CheckRtv	For program-internal use only! Functions for checking or signaling runtime errors
MX_Debug	For internal test purposes only (laboratory)!
MX_PLCOpen	• Functions for diagnostics • Function blocks/functions for drive control • Function blocks/functions for parameters • Functions for scaling • General functions
MX_TechInterface (Firmware version 20 and above)	The axis interface bundles and enhances PLCOpen motion function blocks and provides an easy-to-use interface for drive functionality. Less code and more efficient commands speed up the programming of applications. The axis interface contains control signals and parameters for the various operation modes of the master axis and slave axis, as well as options for setting selected process values.
MX_TechBase (Firmware version 20 and above)	Contains basic functionalities such as control functions, functions to query system information and trigonometric functions.
MX_TechMotionBase (Firmware version 20 and above)	Contains functionalities such as probe function, position monitoring, position switch and functions to manage the cyclic data channel.
MX_TechMotion (Firmware version 20 and above)	Contains a function block to control the tension with regard to a command value. For this purpose, the measured value of a force transducer is evaluated.
MX_TechWinder (Firmware version 20 and above)	Contains function blocks providing functions for winding and unwinding fabric webs (winders).
MX_TechCrosscutCrossseal (Firmware version 20 and above)	Contains function blocks for cross sealer and cross cutter applications.
RIL_Fieldbus	Provides common data types (field bus types) for the Rexroth field bus libraries
RIL_HMI_Uilities	Contains function blocks supporting the HMI devices
RIL_LoopControl	Provides basic elements and controllers of control technology
RIL_ModbusTCP	Provides function blocks which enable communication between Ethernet devices supporting the Modbus protocol
RIL_SercosIII (IndraDrive only)	Provides functions which enable communication between the PLC programming environment and the Sercos III nodes
RIL_SocketComm	Functions and function blocks for using TCP/UDP communication
RIL_Uilities	Functions and function blocks for converting and influencing different data types

Libraries

Library	Description
RMB_SercosIII_Util (Firmware version 20 and above)	Function blocks for axis control via Sercos bus
RMB_TechCam (IndraDrive only; firmware version 20 and above)	Function blocks for creating cam tables
CmplecTask	System library containing, amongst others, interface functions used to switch off a watchdog (and switch it back on afterwards) (see MLD Application Manual: "Task monitoring (watchdog)")
SysDir / SysDir23*1	Functions for synchronously accessing a file directory system on the target
SysFile / SysLibFile23*1	Functions for synchronously accessing files
SysFileAsync / SysFileAsync23*1	Functions for asynchronously accessing files
SyslecTasks23	Supports the "SysIECTaskGetConfig" function that is used to read the task configuration
SysMem / SysMem23*1	Functions for memory management
NetVarUdp	For program-internal use only! Functions for processing network variables
Standard	Functions and function blocks that are required by IEC 61131-3 as standard blocks for an IEC programming system
SysCallback23*1	Functions for activating defined callback functions for runtime events
SysSem / SysSem23*1	Functions for creating and using semaphores for task synchronization
SysSockets / SysSockets23*1	Functions supporting the access to sockets for communication via TCP/IP and UDP
SysStr23*1	Functions for handling strings
SysTime	Provides functions for reading and setting the real-time clock of the local system, as well as enabling various conversions of the time data. NOTE: To begin with, please verify whether the "RIL_Uutilities" library can fulfill your requirements.
SysTimeRTC	Provides functions to read or write the system time of the drive in the IEC 61588 format ("S-0-1305.0.1, System time").
Util	Function blocks that can be used for BCD conversion, bit/byte functions, mathematical auxiliary functions, as controllers, signal generators, function manipulators and for analog value processing
OSCAT	Free "OSCAT" ("Open Source Community for Automation Technology") PLC library. This open-source library contains many useful functions in the areas of "automation technology" and "building automation".

***1** Libraries with the extension "23" (e.g., "SysSockets23") are used to port projects; for new projects, use the libraries without the extension "23"



"*.library" is the standard format for libraries.

The codes of the libraries supplied via IndraWorks are protected, they have the "*.compiled-library-ge33" format.

3.2 Dependencies between axis parameterization, functional package and selected function block

Depending on the functional package and axis parameterization, not all function blocks are executable. The table below shows the dependencies

between the parameterization of the axis, the functional package and the selected operation mode.

P-0-2003 (1)	P-0-0045, bit 14/15 (2)	P-0-0075 (3)	Positioning block mode (MX_MoveRelative / MC_MoveRelative, MX_MoveAbsolute / MC_MoveAbsolute, MX_MoveAdditive / MC_MoveAdditive, MC_Jog)	Torque/Force Control (MC_TorqueControl)	Velocity synchronization (MC_GearIn)	Position synchronization (Phase synchronization, electronic cam, MotionProfile: MC_CamIn, MB_GearInPos, MB_MotionProfile)
0x 0000 0001	-	-	-	-	-	-
0x 0000 0021	-	-	-	-	+	-
0x 0000 0002	0b00	-	+	+	-	-
	0b01	-	-	-	-	-
	0b10		+	-	-	-
	0b11	0	-	-	-	-
		>0	+	-	-	-
0x 0000 0022	0b00	-	+	+	+	+
	0b01	-	-	-	+	-
	0b10	-	+	-	+	+
	0b11	0	-	-	+	-
		>0	+	-	+	+

- Function block is **not** operable
- + Function block is operable
- 1 P-0-2003, Selection of functional packages
- 2 P-0-0045, Control word of current controller
- 3 "P-0-0075, Encoder type 2 (optional encoder)"; is only evaluated as of the following firmwares: MPx06V14, MPx07V10, MPx17

Tab. 3-1: Dependence Between Parameterization of the Axis, Functional Package and Selected Function Block

3.3 Function blocks for PSB firmware derivative

All libraries are available for the PSB firmware for supply units. However, the function blocks of the "MX_PLCOpen" library, which are intended specifically for motion axes, cannot be used.

In the attempt to command a function block that is not supported, the Error output is set at this function block.

4 Signal Time Diagrams of Most Frequently Used Inputs and Outputs

Function blocks according to PLCopen use input and output signals with fixed functions.

Input Signals "Execute" / "Enable"

To trigger the processing of a function block or to activate it, a function block has an "Execute" input, an "ExecuteLock" input or an "Enable" input. All three inputs are of the BOOL type.

The "Execute" activation input is used at function blocks which work in edge-controlled form. With a positive edge of "Execute", the input variables are collected and at the same time the command triggering of a one-time execution of the function block takes place. The outputs "Done" or "InVelocity", "InTorque", "In_xxxx" depend on the input. New input values only take effect via the repeated positive edge of "Execute". When command triggering has not yet been completed and is interrupted, this is signaled by the fact that the "CommandAborted" output is set.



Retriggering is possible for edge-controlled function blocks with "Execute" input; this means that when the activation input, with active function block, gets a repeated edge, the old task is rejected, the inputs are applied again and the new task is continued with the new values.

The "ExecuteLock" activation input behaves like the "Execute" input. But as long as this input is TRUE, the function block cannot be interrupted. Exception: The function block "MC_Stop" can interrupt the function block "MB_Stop".

The "Enable" activation input is used at function blocks which work in status-controlled form. The input variables are applied with the positive edge of "Enable". New input values only take effect via the repeated positive edge of "Enable". If the input variables are cyclically transmitted (e.g. with process controller function blocks), this has to be explicitly documented and marked at the function block input.

Output Signal "Active"

For remote axes, "Active"=TRUE signals that the function block has accepted the processing command (positive edge at "Execute", "ExecuteLock" or "Enable").



The "Active" signal can be delayed at the start of the function block, as possibly required preprocessing has not been completed.

"Active" becomes FALSE, when "Done", "CommandAborted" or "Error" become TRUE. For status-controlled function blocks, "Active" becomes FALSE when "Enable" is reset.



In the case of **parameter function blocks**, "Active" is not set for a **single axis**!

Output Signal "Done"

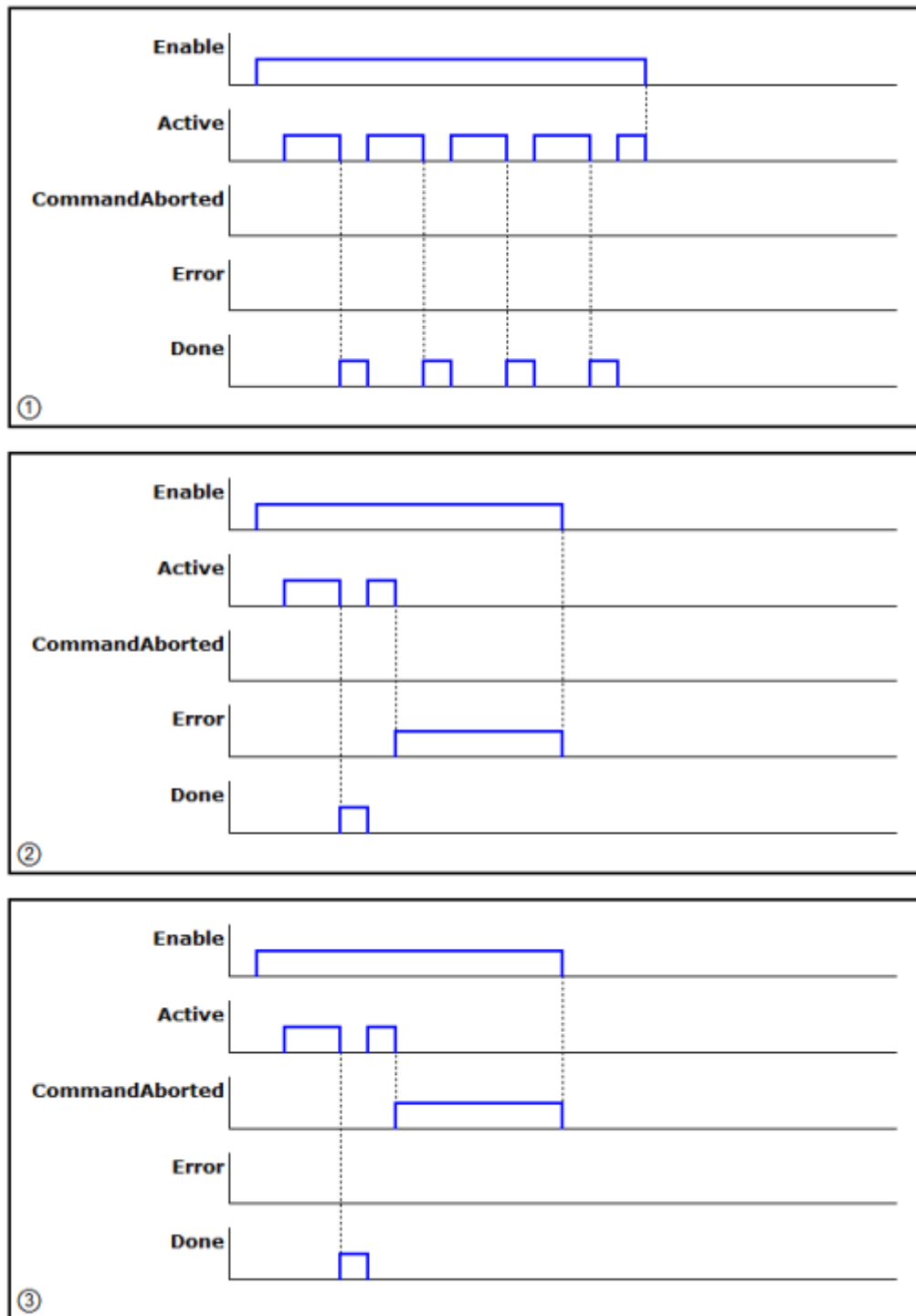
Function blocks which have an administrative task ("MC_Power", ...) or motion function blocks (such as "MC_MoveAbsolute", "MC_MoveRelative", ...) which end in the "standstill" state, signal via the "Done" output that their task has been completed and are no longer active after they have been executed; in other words: The "Done" output signals that a function block has successfully completed its task and possibly provided data are valid.

Signal Time Diagrams of Most Frequently Used Inputs and Outputs

For edge-controlled function blocks, "Done" remains TRUE for one cycle when the "Execute" input had been FALSE at the point of time at which "Done" was set to TRUE. If the input "Execute"=TRUE when "Done" is set, "Done" remains TRUE until "Execute" is set to FALSE.

For status-controlled function blocks, when "Enable"=TRUE, "Done" after successful processing remains TRUE for exactly one cycle before processing starts again.

Signal Time Diagrams of Most Frequently Used Inputs and Outputs

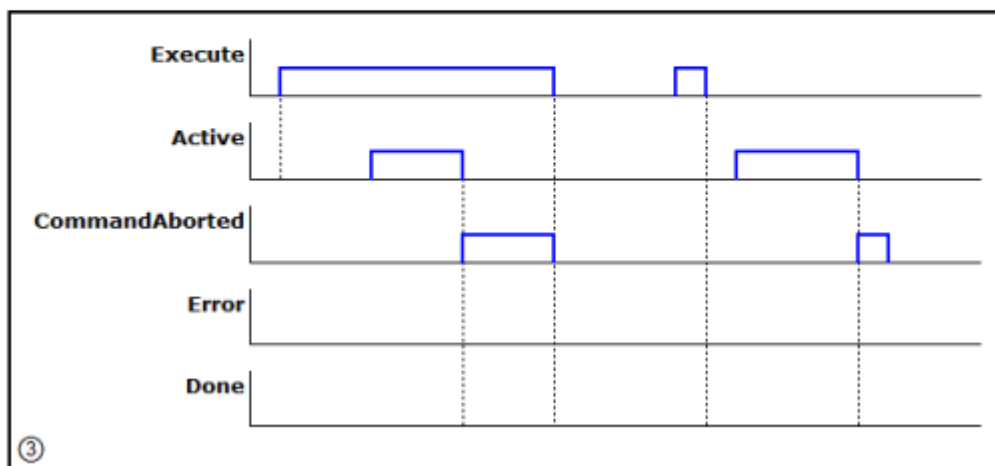
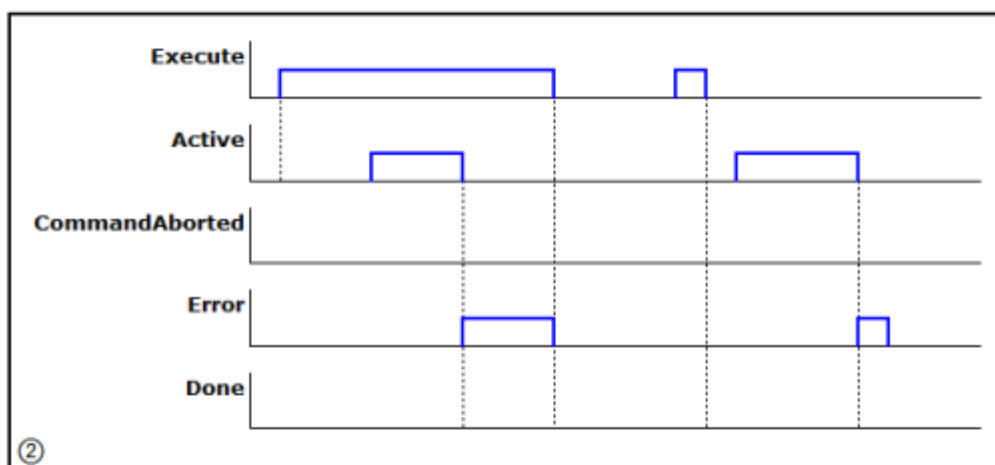
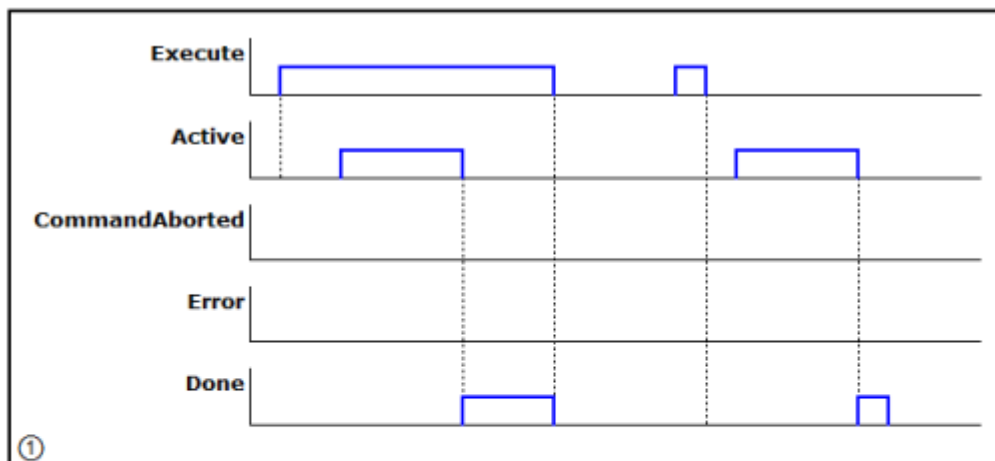


- ① Status-controlled function block processing successfully completed
- ② Status-controlled function block processing aborted with error
- ③ Status-controlled function block processing interrupted

Fig. 4-1:

Signal-Time Behavior of Status-Controlled Function Blocks (with "Enable" Input); in the Case of Parameter Function Blocks, "Active" is not Set for a Single Axis!

Signal Time Diagrams of Most Frequently Used Inputs and Outputs



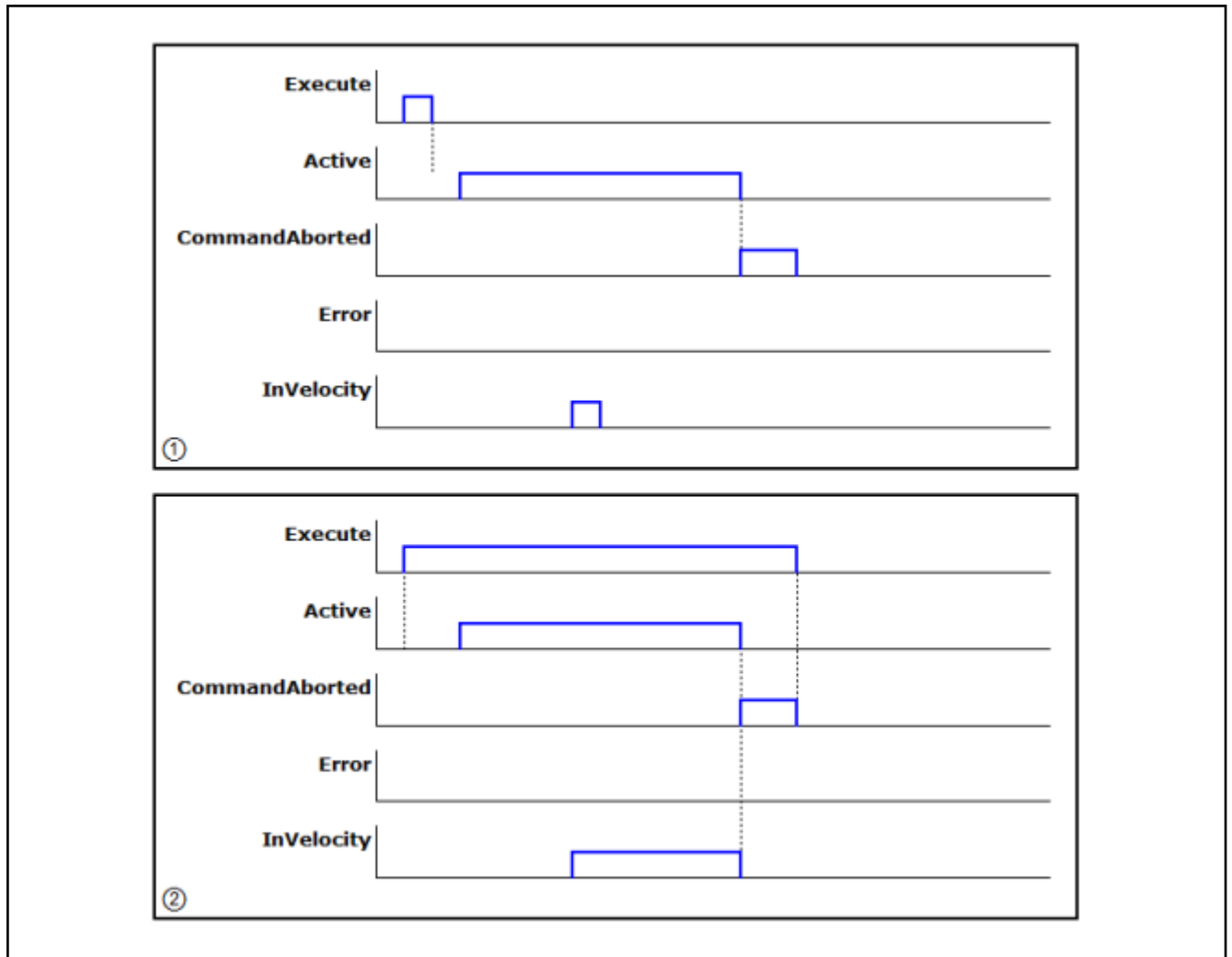
- ① Edge-controlled function block processing successfully completed
- ② Edge-controlled function block processing aborted with error
- ③ Edge-controlled function block processing interrupted

Fig. 4-2:

Signal-Time Behavior of Edge-Controlled Function Blocks (with "Execute" Input); in the Case of Parameter Function Blocks, "Active" is not Set for a Single Axis!

Signal Time Diagrams of Most Frequently Used Inputs and Outputs

There are function blocks (such as "MC_MoveVelocity", "MC_TorqueControl", "MB_MotionProfile") which continue their motion or force activity after their task has been fulfilled and which are aborted only by "MC_Stop" or another function block being called. They signal this state by remaining "Active"; i.e. "Active" and, for example, "InVelocity" etc. are at the same time TRUE.



①+②

Edge-controlled function block processing interrupted

Fig. 4-3: Signal-Time Behavior of Edge-Controlled Function Blocks (with "Execute" Input); "Active" in Combination with "InVelocity"

Output Signal "CommandAborted"

The "CommandAborted" output signals that the function block was unable to complete its task, because it had been interrupted by another motion function block or the axis had been switched off due to an error.

For edge-controlled function blocks, the "CommandAborted" output remains TRUE for one cycle when the "Execute" input had been FALSE at the point of time at which "CommandAborted" was set to TRUE. If the input "Execute"=TRUE when "CommandAborted" is set, "CommandAborted" remains TRUE until "Execute" is set to FALSE.

For status-controlled function blocks, the "CommandAborted" output is reset when the "Enable" input is reset.

5 Error handling

5.1 General information

According to their target, the function blocks have two or three outputs which contain information in case an error occurs:

- "Error",
- "ErrorID" and
- "ErrorIdent"

Error If the processing of a task could not be successfully completed, this is signaled by the "Error" output. The "Error" output is reset with the inputs "Execute"=FALSE or "Enable"=FALSE; an additional reset input therefore is not required.

For edge-controlled function blocks, the "Error" output remains TRUE for one cycle if the "Execute" input had been FALSE at the time at which the "Error" output was set to TRUE. If the input "Execute"=TRUE when "Error" is set, the "Error" output remains TRUE until the "Execute" input is set to FALSE.

For state-controlled function blocks, the "Error" output remains TRUE until the "Enable" input is set to FALSE.

ErrorID The "ErrorID" output is of the ERROR_CODE type. When an error has occurred at the function block, the error classification can be taken from the "ErrorID" output.



The output values of "ErrorID" must be interpreted differently for each target!

The ERROR_CODE type has been defined in the "CommonTypes" library of the respective target.

ErrorIdent The "ErrorIdent" output signals the exact cause of the error.

"ErrorIdent" is of the ERROR_STRUCT type. ERROR_STRUCT contains the following elements:

- "Table",
- "Additional1" und
- "Additional2".



The [ERROR_STRUCT](#) structure and its elements have been defined in the "MX_CommonTypes" library.

Error handling

5.2 Error reference lists



When debugging a PLC program, you can select via **Debug ► Display Mode** in IndraWorks whether binary data types (BYTE, WORD, DWORD) are to be displayed in a decimal, hexadecimal or binary form. The activated selection is identified by a check mark.

MLD_TABLE At the "ErrorIdent" output, the "Table" element refers to the error table "MLD_TABLE".



If no error is present, the value of the "Table" element is "NO_TABLE_USED".

"ErrorID"	"ErrorIdent", Element "Additional1" (hexadecimal)	"ErrorIdent", Element "Additional2"	Description
-	16#00000000	Transmitted axis number	If no error is present, the value of the "Table" element is "NO_TABLE_USED"
INPUT_INVALID_ERROR	16#00000001	Transmitted axis number	Selected function is not available for this axis
RESOURCE_ERROR	16#00000001	Transmitted axis number	Invalid axis reference / undefined axis. Specified axis (also slave axis) is not available
ACCESS_ERROR	16#00000002	Transmitted axis number	The function block used may only have one instance in the system (e.g.: "MX_SynchronControl")
STATE_MACHINE_ERROR	16#00000003	Transmitted axis number	Drive is not in operating status "Ab" / communication phase 4 - or - If the virtual master axis is to be commanded via a "motion" function block: The master axis has not been activated. Activate the master axis by setting bit 0 in P-0-0917 (control word of master axis generator) or / and entering P-0-0758 (virtual master axis, actual position value) in P-0-0916 (master axis format converter signal selection)
STATE_MACHINE_ERROR	16#00000004	Transmitted axis number	Drive is not in operating status "Ab"
ACCESS_ERROR	16#00000005	Transmitted axis number	MLD has neither permanent (P-0-1367) nor temporary control of the specified (local) axis
ACCESS_ERROR	16#00000006	Transmitted axis number	Motion command or "MC_Stop": Drive is torque-free / not under torque
STATE_MACHINE_ERROR	16#00000006	Transmitted axis number	"MC_Power": Drive is not ready for operation. Drive was switched off externally
RESOURCE_ERROR	16#00000007	Transmitted axis number	Function block not possible with current function package / control type (see table " Dependence Between Parameterization of the Axis, Functional Package and Selected Function Block "). - or - A parameter required by a motion function block has already been cyclically configured and therefore is write-protected.
INPUT_RANGE_ERROR	16#00000009	Transmitted axis number	Parameterized acceleration (positioning acceleration) is "0" - or - The acceleration transmitted at "Acceleration" input is "0"
INPUT_RANGE_ERROR	16#00000010	Transmitted axis number	The deceleration transmitted at "Deceleration" input is "0"

Error handling

"ErrorID"	"ErrorIdent", Element "Additional1" (hexadecimal)	"ErrorIdent", Element "Additional2"	Description
STATE_MACHINE_ERROR	16#00000011	Transmitted axis number	Once activated with "MB_PreSetMode", there was an attempt to switch on using "MC_Power" without motion command triggering
STATE_MACHINE_ERROR	16#00000012	Transmitted axis number	An instance of "MB_Stop" is attempting to command an axis while an instance of "MC_Stop" is active - or - The axis is in "Stopping" and must not be commanded ("MB_Stop"/"MC_Stop" with active "Execute" input)
INPUT_RANGE_ERROR	16#00000013	Transmitted axis number	- Cam mode: The "CamTableID" input is not within the valid range (1 .. 8) - MotionProfile mode: - Selected set is invalid ("SetSelection" or "SetNumber" input) Valid range: 0 .. 1 - or - - Selected step is invalid ("StepNumber" input) Valid range: 1 .. 8
STATE_MACHINE_ERROR	16#00000014	Transmitted axis number	MLD is in "ErrorStop" state. Remedy: Execute "MC_Reset"
STATE_MACHINE_ERROR	16#00000015	Transmitted axis number	Activating "MC_Power". "DriveEnable" is missing from the axis control word (P-0-0116, bit 14)
ACCESS_ERROR	16#00000016	Transmitted axis number	MLD in slave axis has permanent control. Therefore, the MLD-M master cannot control the slave
INPUT_RANGE_ERROR	16#00000017	Transmitted axis number	Parameterized acceleration is ≤ 0
INPUT_RANGE_ERROR	16#00000018	Transmitted axis number	Parameterized deceleration is ≤ 0
INPUT_RANGE_ERROR	16#00000019	Transmitted axis number	Parameterized velocity is < 0
ACCESS_ERROR	16#00000020	Index of the second task	Several tasks are using the same instance of a function block, which is not allowed
STATE_MACHINE_ERROR	16#00000021	Transmitted axis number	An additive motion command was started that is not allowed in this axis status
ACCESS_ERROR	16#00000023	Transmitted axis number	Motion command could not be processed due to buffer overflow
INPUT_INVALID_ERROR	16#00000024	Transmitted axis number	Identical axis reference for master and slave axis. The slave axis has to follow itself; recursive master axis combinations are not supported.
OTHER_ERROR	16#00000199	Transmitted axis number	Internal processing error (undefined state)
ACCESS_ERROR	16#00000200	Parameter IDN	Internal error: Undefined parameter data format, parameter cannot be converted to FLOAT
ACCESS_ERROR	16#00000201	Parameter IDN	With this function block it is impossible to read/write an ASCII parameter
ACCESS_ERROR	16#00000202	Parameter IDN	With this function block it is impossible to read/write a list parameter
INPUT_INVALID_ERROR	16#00000203	Parameter IDN	Parameter IDN does not belong to any command
OTHER_ERROR	16#00000204	Transmitted axis number	Internal processing error (string too short)
ACCESS_ERROR	16#00000205	Parameter IDN	The required number of bytes is smaller than the actual length of the list

Error handling

"ErrorID"	"ErrorIdent", Element "Additional1" (hexadecimal)	"ErrorIdent", Element "Additional2"	Description
INPUT_INVALID_ERROR	16#00000206	Transmitted axis number / transmitted Sercos address	Invalid pointer at the function block input [address of the applied array is outside of the PLC range or is in the memory range of a function (use global array)]. For "RIL_SercosIII.lib" library only: "Additional2" contains the transmitted Sercos address
ACCESS_ERROR	16#00000207	Parameter IDN	This function block can only be used to process ASCII parameters
CCD_ERROR	16#00000208	16#00000200	Too many simultaneous read/write tasks on remote axes
SYSTEM_ERROR	16#00000208	CCD error number	Internal CCD system error, see "Additional2" for error number
SYSTEM_ERROR	16#00000209	CoSeMa error number	Internal CoSeMa system error, see "Additional2" for error number
ACCESS_ERROR	16#00000210	99	Reset command S-0-0099 incorrectly executed
ACCESS_ERROR	16#00000211	Transmitted axis number	"MC_Reset" cannot reset the error (timeout)
RESOURCE_ERROR	16#00000220	Transmitted axis number	No external encoder available
RESOURCE_ERROR	16#00000250	Transmitted axis number	Master communication synchronizes the drive
RESOURCE_ERROR	16#00000251	Transmitted axis number	NC cycle time (S-0-0001) does not match MLD task cycle time
INPUT_RANGE_ERROR	16#00000252	Transmitted axis number	The "CmdDelay" input is greater than the maximum value
RESOURCE_ERROR	16#00000253	Transmitted axis number	The real-time channel may only use one MLD task
RESOURCE_ERROR	16#00000254	Transmitted axis number	Too many real-time variables used. Four RTcW_ variables and four RTcR_ variables are currently allowed
CALCULATION_ERROR	16#00000255	Transmitted axis number	Internal error (variable offset too great)
CALCULATION_ERROR	16#00000256	Transmitted axis number	Internal error (variable access error)
RESOURCE_ERROR	16#00000257	Transmitted axis number	Real-time channel actual values but no command values defined
ACCESS_ERROR	16#00000258	Transmitted axis number	Synchronous motion task (CCD / master communication) of the "externally event-controlled" type does not support local real-time channel
ACCESS_ERROR	16#00000259	Transmitted axis number	The local real-time channel has not been configured for the task
ACCESS_ERROR	16#00000300	Transmitted axis number	MLD already has permanent control. There is no reason to additionally switch to temporary control
ACCESS_ERROR	16#00000301	Transmitted axis number	Temporary control is not possible with virtual slave
ACCESS_ERROR	16#00000302	Transmitted axis number	Manual mode active, temporary control impossible
STATE_MACHINE_ERROR	16#00000320	Transmitted axis number	Master communication internally synchronizes the device / does not allow phase switch (P-0-4086, bit 0=0)

Error handling

"ErrorID"	"ErrorIdent", Element "Additional1" (hexadecimal)	"ErrorIdent", Element "Additional2"	Description
STATE_MACHINE_ERROR	16#00000321	Transmitted axis number	Drive initialization not yet completed (incl. reaching of target phase) For master communication with an analog or parallel interface, it is necessary to wait until the automatic phase switch to "P4" is over. If a transition command error occurs during this first phase switch, you have to wait for 10 s before it is allowed to specify the phase with the "MX_SetDeviceMode" function block
ACCESS_ERROR	16#00000322	Transmitted axis number	A transition command was started or cleared from a different device
DEVICE_ERROR	16#00000350	Transmitted axis number	Unknown secondary operation mode ("NewOpMode">7 was set)
RESOURCE_ERROR	16#00000355	Transmitted axis number	Axis applied at the "Master" input is not a valid master axis. Only those axes are supported as master axes that are either contained in the master axis configuration list (P-0-1820.0.1) or that are locally available in the slave axis (VmAxisInt, RmAxisInt and LinkAxisInt).
INPUT_RANGE_ERROR	16#00000370	Incorrect "CmdState"	Invalid "CmdState" input (wrong enum value)
INPUT_RANGE_ERROR	16#00000371	Incorrect "CmdState"	"CmdState" not allowed, e.g., "CmdState"="MX_CMD_NO_ACTIVE" while "P-0-1449, C4900 PLC command"="3"
INPUT_RANGE_ERROR	16#00000380	Incorrect diagnostic message number	Invalid input of diagnostic message number (N<1 or N>4)
ACCESS_ERROR	16#00000400	Parameter IDN	The specified parameter cannot be read / written with this function block. For example, parameter of HEX or BIN data type with "MB_WriteRealParameter"
ACCESS_ERROR	16#00000401	Parameter IDN	This function block cannot be used to read / write a FLOAT parameter. PLC registers (e.g., P-0-1370) can be configured as FLOAT.
INPUT_RANGE_ERROR	16#00000402	Transmitted axis number / transmitted Sercos address	The applied parameter value does not conform to the data format of the parameter For "RIL_SercosIII.lib" library only: "Additional2" contains the transmitted Sercos address
INPUT_INVALID_ERROR	16#00000403	Transmitted axis number	The address applied at the "ValueAdr" input of the "MB_ReadListParameter" or "MB_WriteListParameter" function block is not WORD-compatible (2Byte); i.e. the address is not divisible by 2.
INPUT_INVALID_ERROR	16#00000404	Transmitted axis number	The address applied at the "ValueAdr" input of the "MB_ReadListParameter" or "MB_WriteListParameter" function block is not DWORD-compatible (4Byte); i.e. the address is not divisible by 4.
ACCESS_ERROR	16#00000420	Transmitted axis number	A deactivated axis was addressed.
ACCESS_ERROR	16#00000422	Transmitted axis number	The service channel is no longer available (phase < 2).
ACCESS_ERROR	16#00000423	Transmitted axis number	Double address assignment. At least 2 axes have the same Sercos address.
SYSTEM_ERROR	16#00000424	Transmitted axis number	Queue error. Too many tasks were transmitted to an axis.
INPUT_RANGE_ERROR	16#00000500	Transmitted port	A value for "Port"<>"COM1" was transmitted
ACCESS_ERROR	16#00000501	Transmitted port	COM1 has already been opened

Tab. 5-1: MLD_TABLE error table

SERCOS_TABLE

Error handling

Error code		
Hexadecimal	Decimal	Explanation
0x1001	4097	No IDN
0x1009	4105	Invalid access to element 1
0x2001	8193	No name
0x2002	8194	Name transmission too short
0x2003	8195	Name transmission too long
0x2004	8196	Name cannot be changed (read only)
0x2005	8197	Name is write-protected at this time
0x3002	12290	Attribute transmission too short
0x3003	12291	Attribute transmission too long
0x3004	12292	Attribute cannot be changed (read only)
0x3005	12293	Attribute is write-protected at this time
0x4001	16385	No units
0x4002	16386	Unit transmission too short
0x4003	16387	Unit transmission too long
0x4004	16388	Unit cannot be changed (read only)
0x4005	16389	Unit is write-protected at this time
0x5001	20481	No minimum input value
0x5002	20482	Minimum input value transmission too short
0x5003	20483	Minimum input value transmission too long
0x5004	20484	Minimum input value cannot be changed (read only)
0x5005	20485	Minimum input value is write-protected at this time
0x6001	24577	No maximum input value
0x6002	24578	Maximum input value transmission too short
0x6003	24579	Maximum input value transmission too long
0x6004	24580	Maximum input value cannot be changed (read only)
0x6005	24581	Maximum input value is write-protected at this time
0x7002	28674	Operation data transmission too short
0x7003	28675	Operation data transmission too long
0x7004	28676	Operation data cannot be changed (read only)
0x7005	28677	Operation data is write-protected at this time (reason: communication phase or mode)
0x7006	28678	Operation data is smaller than the minimum input value
0x7007	28679	Operation data is greater than the maximum input value
0x7008	28680	Invalid operation data (e.g. IDN not supported, invalid bit number, invalid bit combination, invalid list length)

Error handling

Error code		
Hexadecimal	Decimal	Explanation
0x7009	28681	Operation data write protected by a password
0x700A	28682	Operation data is write protected, it is configured cyclically (IDN is configured in the MDT or AT. Therefore writing via the service channel is not allowed).
0x700B	28683	Invalid indirect addressing (e.g., data container, list handling)
0x700C	28684	Operation data is write protected, due to other settings (e.g., parameter, operation mode, drive enable, drive on etc.)
0x7010	28688	Procedure command already active
0x7011	28689	Procedure command not interruptible
0x7012	28690	Procedure command at this time not executable (e.g., in this phase the procedure command cannot be activated)
0x7013	28691	Procedure command not executable (invalid or false parameters)

Tab. 5-2: Sercos error codes

INDRV_TABLE



"INDRV_TABLE" actually is not a table, but refers to the documentation "Rexroth IndraDrive, Diagnostics". In this documentation, the error number (diagnostic message number) provides information on cause and remedy.

6 Data types

Data type	Bit size	Description
BOOL	With indication of address (e.g. "AT %MX1.2"): 1	"BOOL" can have the values "TRUE" or "FALSE"; without indication of address, 8 bits are used
	Without indication of address: 8	
BYTE	8	Byte value 0 to 255
WORD	16	Word value 0 to 65535
DWORD	32	Double word values 0 to (2 ³²)-1
LWORD	64	Long word values 0 to (2 ⁶⁴)-1
INT	16	Integer (16 bits) from -32768 to +32767
DINT	32	Integer (32 bits) from -2 ³¹ to (2 ³¹)-1
SINT	8	Integer (8 bits) from -128 to +128
USINT	8	Integer (8 bits) from 0 to +255
UINT	16	Integer (16 bits) from 0 to +65535
UDINT	32	Integer (32 bits) from 0 to +4294967295
LINT	64	Integer (64 bits) from -2 ⁶³ to +2 ⁶³ -1
ULINT	64	Integer (64 bits) from 0 to +2 ⁶⁴ -1
REAL	32	IEEE floating point number
LREAL	64	IEEE floating point number
TIME	32	IEC time in intervals of 1 ms
LTIME	64	Like "TIME". In addition, "µs" (microseconds) and "ns" (nanoseconds) can be used
DATE	32	IEC date in intervals of 1 day
DATE_AND_TIME, DT	32	Time in steps of 1s: year-month-day-hour:minute:second
TIME_OF_DAY, TOD	32	Time in steps of 1ms: Hour (0-23), min. (0-59), second (0-59), ms (0-999)
SYSTIME	64	Time in milliseconds since 01/01/1970 00:00:00

Tab. 6-1: General data types

7 Parameter structure

7.1 Structure

Parameters consist of the following elements:

Element number	Description	Request
1	IDN	necessary
2	Name	optional
3	Attribute (see "Sercos attribute")	necessary
4	Unit	optional
5	Minimum value	optional
6	Maximum value	optional
7	Data	necessary

Tab. 7-1: Parameter structure

7.2 Sercos attribute

The figure / table below shows the structure of the Sercos attribute.

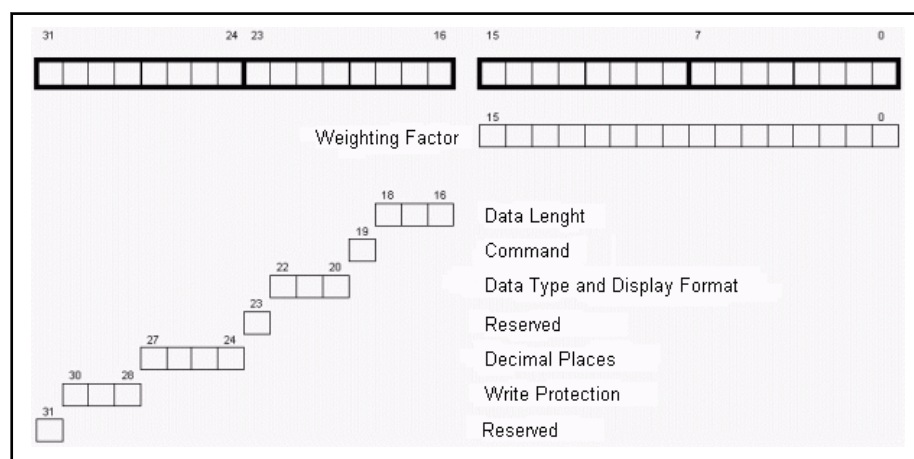


Fig. 7-1: Structure of the Sercos attribute

Binary digit	Designation/function
30	Write protection of parameter operating data in operating mode (Sercos phase 4) 0: Operating data can be written 1: Operating data is write-protected
29	Write protection of parameter operating data (Sercos phase 3) 0: Operating data can be written 1: Operating data is write-protected
28	Write protection of parameter operating data in parameter mode (Sercos phase 2) 0: Operating data can be written 1: Operating data is write-protected

Parameter structure

Binary digit	Designation/function	
27...24	Number of decimal places of operating data 0000 (no decimal places) to 1111 (15 decimal places)	
23	Reserved	
22...20	Data type 000: Binary number 001: Unsigned integer 010: Integer 011: Unsigned integer 100: Expanded character set 101: Unsigned integer 110: Floating-point number 111: Unsigned integer	Display format Binary Unsigned decimal number Signed decimal number Hexadecimal Text Parameter IDN format (Sercos standard, 2 bytes) Signed decimal number Parameter IDN format (Sercos standard, 4 bytes)
19	Operating data function 0: Operating data or parameter 1: Command	
18...16	Operating data length (The length of the operating data is required by the control unit to properly complete the data transmission via the service channel.) 000: Reserved 001: Operating data is 2 bytes long 010: Operating data is 4 bytes long 011: Reserved 100: Operating data has a variable length with 1-byte data 101: Operating data has a variable length with 2-byte data 110: Operating data has a variable length with 4-byte data 111: Reserved	
15...0	Weighting factor The weighting factor is an unsigned integer used to convert numerical data to a display format. The weighting factor is set to the value of "1" if it is not used to display data.	

Tab. 7-2: Elements of the Sercos attribute

8 IecSfc

"IecSfc.library" is automatically included in PLC projects. The library provides the "SFCActionControl" function.

Various qualifiers can be used to configure how an action assigned to an IEC step is to be executed. The qualifiers are processed by the "SFCActionControl" function.

See also documentation "IndraLogic 2G, PLC Programming System", chapter "Qualifiers for actions in SFC".

9 MX_Base

9.1 Introduction

The "MX_Base.library" library provides the following to IndraMotion MLD:

- Data types or structures
- Cyclic parameters as direct variables (system-wide variables)
- Axis structures (for a multi-axis system [MLD system mode])

The structure of the "MX_Base.lib" library is as follows:

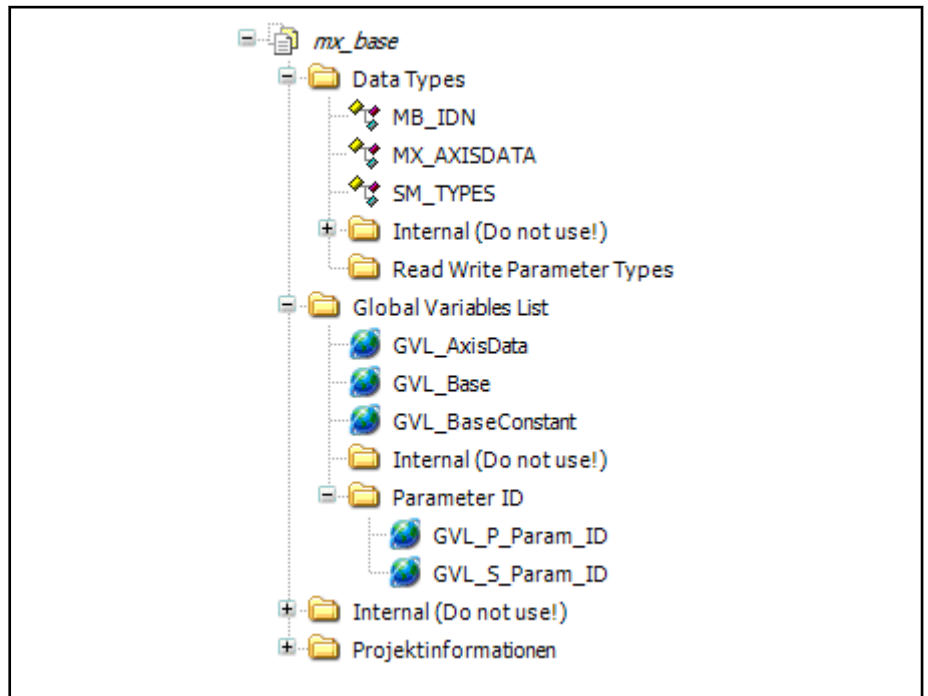


Fig. 9-1: "MX_Base.lib" library structure

9.2 Data types

"Data types" structure

- MB_IDN
Data type to address parameters.
- MX_AXISDATA
See IndraMotion MLD, Application Manual: "Structure of AxisData"
- SM_TYPES
See IndraMotion MLD, Application Manual: "SM_TYPES"

MX_Base

9.3 Global Variable List

GVL_AxisData

Name	Data type	Comment
AxisData	ARRAY [MB_MIN_AXIS_NUMBER..MB_MAX_AXIS_NUMBER] OF MX_AXISDATA	AxisData of all real axes

GVL_Base The Global Variable List "GVL_Base" contains all axis types that can be declared in IndraMotion MLD.

Name	Data type	Initial value	Comment
NO_AXIS	AXIS_REF	STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := NO_OBJECT)	Definition of "no axis"
AXIS1		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := AXIS_1)	Local axis 1, used as axis, slave or master (MLD-M)
AXIS2		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := AXIS_2)	Removed axis 2, used as axis, slave or master (MLD-M)
...		...	...
AXIS10		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := AXIS_10)	Removed axis 10, used as axis, slave or master (MLD-M)
VmAxis1		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := VMA_1)	Virtual axis 1, used as axis, slave (internal) or master (MLD-M)
VmAxisInt		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := VMA_INT)	Virtual master axis, P-0-0761, used as master
RmAxis1		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := RMA_1)	Real master axis 1, used as master (MLD-M)
RmAxisInt		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := RMA_INT)	Real master axis, P-0-0052, used as master
LinkAxis1		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := MA_LINK_1)	External master axis 1, used as master (MLD-M)
LinkAxisInt		STRUCT(CntrlNo := LOCAL_CNTRL, AxisNo := MA_LINK_INT)	External master axis, P-0-0053 or P-0-0787, used as master
DV_Axis	ARRAY [AXIS_1..AXIS_1] OF MX_DirectVarAxis		Direct variables structure for "all" local axes

GVL_BaseConstant The Global Variable List "GVL_BaseConstant" contains system-dependent constants, depending on the respective hardware and firmware:

Name	Data type	Initial value	Comment
IL_TARGET_TYPE	UINT	4	MLD
IL_TARGET_VERSION		20 ¹⁾	MLD*VRS
IL_SYSTEM_MLD	INT	1	System MLD
IL_TARGET_NAME	STRING	IndraMotion_MLD_E	IndraDrive
		IndraMotion_MLD_H	HydraulicDrive
MB_MIN_AXIS_NUMBER	INT	AXIS_1	First MLD axis number
MB_MAX_AXIS_NUMBER		VMA_1	Last MLC axis number (virtual master axis)
MB_NUMBER_OF_AXIS		((MB_MAX_AXIS_NUMBER - MB_MIN_AXIS_NUMBER) + 1)	Maximum number of axes
MB_MAX_RECOMMENDED_AXIS			Recommended maximum number of axes
INT_MB_IDN_32BIT		1	Internal information for "RIL_SercosIII.library"
MB_MIN_SERCOS_ADDRESS	INT	0	First Sercos address (0==master)
MB_MAX_SERCOS_ADDRESS		253	Last Sercos address
MB_NUMBER_OF_SERCOS_DEVICES		((MB_MAX_SERCOS_ADDRESS - MB_MIN_SERCOS_ADDRESS) + 1)	Maximum number of Sercos devices

1) The value depends on the firmware version; "20", as shown here, stands for the initial value of firmware version 20

- Parameter ID**
- **GVL_P_Param_ID**
 The Global Variable List "GVL_P_Param_ID" contains constants to be able to address S-parameters via the IDN.
 Example: FP_P_0_0001, to be able to address parameter P-0-0001 (Switching frequency of the power output stage).
 - **GVL_S_Param_ID**
 The Global Variable List "GVL_S_Param_ID" contains constants to be able to address S-parameters via the IDN.

10 MX_CanL2

10.1 Introduction

The "MX_CanL2" library contains functions, function blocks and data types for level 2 CAN communication.

Functions and function blocks

- MX_CANL2_FifoReceive
- MX_CANL2_FifoSend
- MX_CANL2_Init
- MX_CANL2_SetLED
- MX_fCANL2_MBoxReceive
- MX_fCANL2_MBoxSend
- MX_fCANL2_Status
- MX_fCANL2_MBoxRecStatus
- MX_fCANL2_MBoxSndStatus
- MX_fCANL2_MBoxUpdate
- MX_fCANtoHI
- MX_fCANtoHs
- MX_fHtoCANI
- MX_fHtoCANs

Data types

- MX_CANL2_LED_MODE
- MX_CANL2_MSG_INIT

10.2 MX_CANL2_FifoReceive

The "MX_CANL2_FifoReceive" function is used to read a CAN telegram from the CAN Fifo.



Fig. 10-1: "MX_CANL2_FifoReceive" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"
CobId	POINTER TO UDINT	Pointer to UDINT variable for COB-ID of the mailbox
BufLength	INT	Length of read buffer in bytes
ReadData	POINTER TO BYTE	Address of read buffer (typ. ARRAY[1..8] of BYTE)

Tab. 10-1: Input variables of the "MX_CANL2_FifoReceive" firmware function

MX_CanL2

Return value	Data type	Description
MX_CANL2_FifoReceive	DINT	Number of transmitted bytes (0..8) - or - Error number: -1: Invalid handle -103: Length "BufLength" too small -602: No new telegram available

Tab. 10-2: Return value of the "MX_CANL2_FifoReceive" firmware function

10.3 MX_CANL2_FifoSend

The "MX_CANL2_FifoSend" function is used to transmit a CAN telegram via the CAN Fifo.



Fig. 10-2: "MX_CANL2_FifoSend" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"
CobId	UDINT	COB-ID of mailbox
Length	UDINT	Number of bytes to be transmitted
WriteData	POINTER TO BYTE	Address of transmit buffer (typ. ARRAY[1..8] of BYTE)

Tab. 10-3: Input variables of the "MX_CANL2_FifoSend" firmware function

Return value	Data type	Description
MX_CANL2_FifoSend	DINT	Number of transmitted bytes (0..8) - or - Error number: -1: Invalid handle -104: Invalid length -601: Transmit Fifo is full

Tab. 10-4: Return value of the "MX_CANL2_FifoSend" firmware function

10.4 MX_CANL2_Init

The "MX_CANL2_Init" firmware function block is used to initialize a CAN interface and configure its telegrams.

The CAN interface thereby is initialized once. It cannot be changed during bus communication, i.e. the Init function has to be called again to make a change, deactivating then re-enabling the CAN controller.

There are 16 transmit and 16 receive mailboxes available, each with a COB-ID configured. In addition, a transmit FIFO and a receive FIFO are made available for non-configured telegrams. It is currently possible to have 2

MX_CanL2

ports. Port 1 addresses the first CAN card found that is not used by master communication, port 2 addresses the next CAN card.

Example: For the pitch converter, port 1 is at slot X50.27 and port 2 is at slot X50.32.

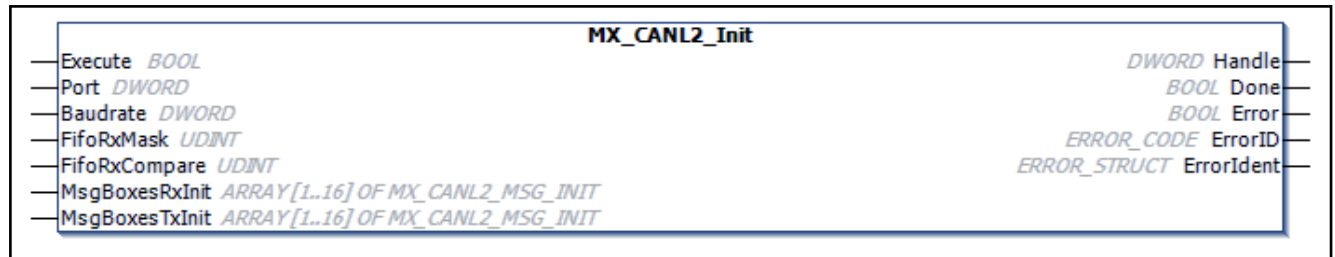


Fig. 10-3: "MX_CANL2_Init" firmware function block

Input variables	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
Port	DWORD	CAN port of the hardware 1: First CAN module not used by master communication 2: Second CAN module not used by master communication 3: Third CAN module not used by master communication ...
Baud rate	DWORD	Desired CAN baud rate
FifoRxMask	UDINT	Mask for COB-ID comparison*1
FifoRxCompare	UDINT	Value for COB-ID comparison*1

*1 Example: "Compare"=0000 1000 0011 0000,
 "Mask"=1111 1111 1111 0000 ⇒ range between 2096-2111
 (830h- 83Fh)

Tab. 10-5: Input variables of the "MX_CANL2_Init" firmware function block

Output variables	Data type	Description
Handle	DWORD	Handle of the port for additional access
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

Tab. 10-6: Output variables of the "MX_CANL2_Init" firmware function block

"ErrorID"	"ErrorIdent", Element "Additional1"	"ErrorIdent", Element "Additional2"	Description
INPUT_INVALID_ERROR	-205	0	The access index has not been defined
INPUT_INVALID_ERROR	-10	0	The Controller has a fatal error
INPUT_INVALID_ERROR	-101	0	The Controller is not in the configuration mode

MX_CanL2

"ErrorID"	"ErrorIdent", Element "Additional1"	"ErrorIdent", Element "Additional2"	Description
INPUT_INVALID_ERROR	-201	0	A COB-ID is used twice in the same direction
INPUT_INVALID_ERROR	-202	0	The value of a COB-ID is invalid

Tab. 10-7: MLD_TABLE error table of the "MX_CANL2_Init" firmware function block

Input/output variable (VAR_IN_OUT)	Data type	Description
MsgBoxesRxInit	ARRAY[1..16] OF MX_CANL2_MSG_INIT	Set of initialized structures for receiving
MsgBoxesTxInit	ARRAY[1..16] OF MX_CANL2_MSG_INIT	Set of initialized structures for transmitting

Tab. 10-8: Input/output variables of the "MX_CANL2_Init" firmware function block

"MX_CANL2_MSG_INIT" data type The "MX_CANL2_MSG_INIT" data type is used to initialize a mailbox of the CAN interface.

Variables	Data type	Description
udiCobId	UDINT	COB-ID of mailbox
bInitialized	BOOL	TRUE: Mailbox parameters have been initialized
bNonOverwrite	BOOL	RX: TRUE: No more receiving until data have been read TX: TRUE: No more transmitting until data have been transmitted
bRtr	BOOL	Only valid for RX mailboxes FALSE: Transmit immediately TRUE: Only transmit upon request
bEnable	BOOL	TRUE: Activate mailbox

Tab. 10-9: "MX_CANL2_MSG_INIT" data type variables

10.5 MX_CANL2_SetLED

The "MX_CANL2_SetLED" function is used to control the LED on the CAN module.

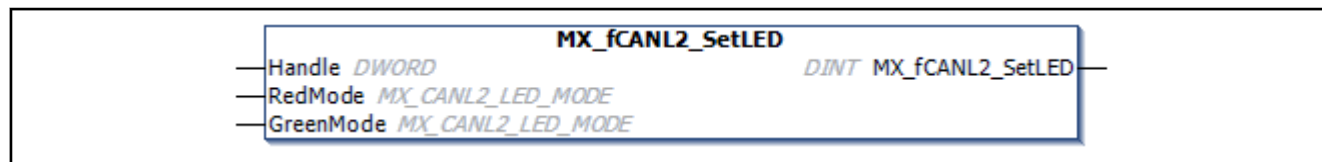


Fig. 10-4: "MX_CANL2_SetLED" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"
RedMode	MX_CANL2_LED_MODE	Display mode of the red LED
GreenMode	MX_CANL2_LED_MODE	Display mode of the green LED

Tab. 10-10: Input variables of the "MX_CANL2_SetLED" firmware function block

MX_CanL2

Return value	Data type	Description
MX_CANL2_SetLED	DINT	0: No error -1: Invalid handle -301: Invalid LED mode

Tab. 10-11: Return value of the "MX_CANL2_SetLED" firmware function block

"MX_CANL2_LED_MODE" data type

The "MX_CANL2_LED_MODE" data type is used to control the LED on the CAN module.

Values	Description
LED_OFF	Off
LED_FLICKERING	Flickers with 10 Hz
LED_BLINKING	Flashes with 2.5 Hz
LED_ON	On
LED_SINGLE_FLASH	One 0.2-second pulse and a one-second pause
LED_DOUBLE_FLASH	Two 0.2-second pulses and a one-second pause
LED_TRIPLE_FLASH	Three 0.2-second pulses and a one-second pause
LED_QUAD_FLASH	Four 0.2-second pulses and a one-second pause

Tab. 10-12: "MX_CANL2_LED_MODE" data type variables

10.6 MX_fCANL2_MBoxReceive

The "MX_fCANL2_MBoxReceive" function is used to read a mailbox from the CAN driver.



Fig. 10-5: "MX_fCANL2_MBoxReceive" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"
MsgBoxNum	INT	Number of mailbox
Length	INT	Length of read buffer in bytes
ReadData	POINTER TO BYTE	Address of read buffer (typ. ARRAY[1..8] of BYTE)

Tab. 10-13: Input variables of the "MX_fCANL2_MBoxReceive" firmware function

MX_CANL2

Return value	Data type	Description
MX_fCANL2_MBoxReceive	INT	Number of transmitted bytes (0..8) - or - Error number: -2 : Invalid range of target memory -103: Target buffer is too small -501: Invalid mailbox number -502: Mailbox not initialized -503: Mailbox not switched on -505: No new telegram available

Tab. 10-14: Return value of the "MX_fCANL2_MBoxReceive" firmware function

10.7 MX_fCANL2_MBoxSend

The "MX_fCANL2_MBoxSend" function is used to move a CAN telegram to a mailbox to transmit it.



Fig. 10-6: "MX_fCANL2_MBoxSend" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"
MsgBoxNum	INT	Number of mailbox
Length	INT	Number of bytes to be transmitted
WriteData	POINTER TO BYTE	Address of transmit buffer (typ. ARRAY[1..8] of BYTE)

Tab. 10-15: Input variables of the "MX_fCANL2_MBoxSend" firmware function

Return value	Data type	Description
MX_fCANL2_MBoxSend	INT	Number of transmitted bytes (0..8) - or - Error number: -1: Invalid handle -501: Invalid mailbox number -502: Mailbox not initialized -503: Mailbox not switched on -504: Mailbox not available (NonOverwrite flag switched on)

Tab. 10-16: Return value of the "MX_fCANL2_MBoxSend" firmware function

10.8 MX_fCANL2_Status

The "MX_fCANL2_Status" function is used to read the status of the CAN driver.



Fig. 10-7: "MX_fCANL2_Status" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"

Tab. 10-17: Input variables of the "MX_fCANL2_Status" firmware function

Return value	Data type	Description
MX_fCANL2_Status	INT	0001: "Init" status 0100: "Operational" status 8001: "ErrorWarning" status 8002: "Error" status 8003: "BusOff" status -1: Invalid handle

Tab. 10-18: Return value of the "MX_fCANL2_Status" firmware function

10.9 MX_fCANL2_MBoxRecStatus

The "MX_fCANL2_MBoxRecStatus" function is used to return the states of all receive mailboxes in bit-coded form. Each of the 16 mailboxes controls one bit of the return value. All 32 bits are set in the case of an error.

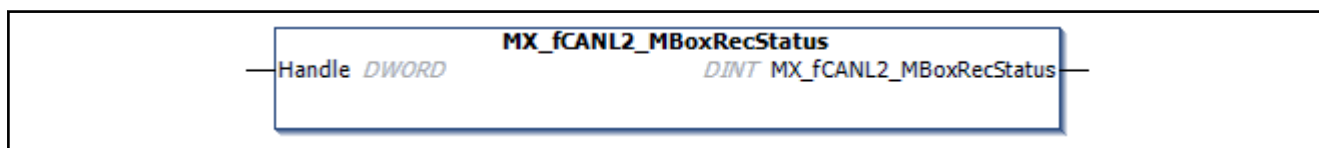


Fig. 10-8: "MX_fCANL2_MBoxRecStatus" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"

Tab. 10-19: Input variables of the "MX_fCANL2_MBoxRecStatus" firmware function

Return value	Data type	Description
MX_fCANL2_MBoxRecStatus	DINT	Bits #0..#15 correspond to the states of mailboxes 1..16. Status bit 0: Mailbox is empty Status bit 1: Mailbox contains data -1: Invalid handle

Tab. 10-20: Return value of the "MX_fCANL2_MBoxRecStatus" firmware function

10.10 MX_fCANL2_MBoxSndStatus

The "MX_fCANL2_MBoxSndStatus" function is used to return the states of all transmit mailboxes in bit-coded form. Each of the 16 mailboxes controls one bit of the return value. All 32 bits are set in the case of an error.

MX_CanL2



Fig. 10-9: "MX_fCANL2_MBoxSndStatus" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"

Tab. 10-21: Input variables of the "MX_fCANL2_MBoxSndStatus" firmware function

Return value	Data type	Description
MX_fCANL2_MBoxSndStatus	DINT	Bits #0..#15 correspond to the states of mailboxes 1..16. Status bit 0: Mailbox is empty / was transmitted Status bit 1: Mailbox still contains data / was not yet transmitted -1: Invalid handle

Tab. 10-22: Return value of the "MX_fCANL2_MBoxSndStatus" firmware function

10.11 MX_fCANL2_MBoxUpdate

The function block is used

- to write data to the mailbox (mailbox update)
- or -
- to transmit a remote request (see "Length" input variable).



Fig. 10-10: "MX_fCANL2_MBoxUpdate" firmware function

Input variables	Data type	Description
Handle	DWORD	Handle of "MX_CANL2_Init"
MsgBoxNum	DINT	Number of mailbox
Length	DINT	- Length > "0": Update of the transmit mailbox in the controller; the content is not actively transmitted - Length = "0": Remote request; after having received the request, the requested mailbox transmits its content
WriteData	POINTER TO BYTE	Pointer to source memory

Tab. 10-23: Input variables of the "MX_fCANL2_MBoxUpdate" firmware function

Return value	Data type	Description
MX_fCANL2_MBoxUpdate	DINT	Length of the written data (0..8) - or - Error number: -1: Interface error (invalid handle) -501: Invalid mailbox number -502: Mailbox not initialized -503: Mailbox not switched on -504: Mailbox not available (NonOverwrite flag switched on)

Tab. 10-24: Return value of the "MX_fCANL2_MBoxUpdate" firmware function

10.12 MX_fCANtoHI

Swap functions to simplify the data exchange between host (MLD) and CAN. They encapsulate the endians and are used for programming the data exchange via CAN in a way that is compatible across systems.

The "MX_fCANtoHI" function receives a 32-bit value in CAN byte order and returns a 32-bit value in the byte order of the target (MLD).

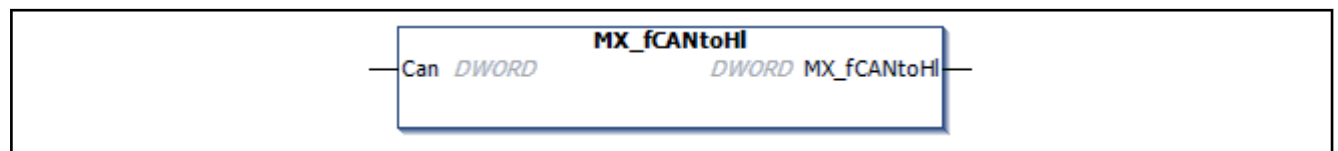


Fig. 10-11: "MX_fCANtoHI" firmware function

Input variables	Data type	Description
Can	DWORD	Data from CAN

Tab. 10-25: Input variables of the "MX_fCANtoHI" firmware function

Return value	Data type	Description
MX_fCANtoHI	DWORD	Data for MLD

Tab. 10-26: Return value of the "MX_fCANtoHI" firmware function

10.13 MX_fCANtoHs

Swap functions to simplify the data exchange between host (MLD) and CAN. They encapsulate the endians and are used for programming the data exchange via CAN in a way that is compatible across systems.

The "MX_fCANtoHs" function receives a 16-bit value in CAN byte order and returns a 16-bit value in the byte order of the target (MLD).

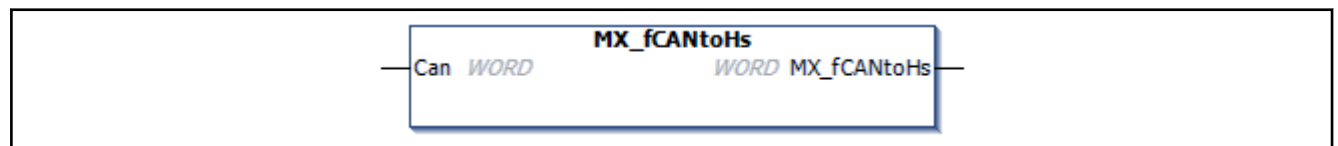


Fig. 10-12: "MX_fCANtoHs" firmware function

MX_CanL2

Input variables	Data type	Description
Can	WORD	Data from CAN

Tab. 10-27: Input variables of the "MX_fCANtoHs" firmware function

Return value	Data type	Description
MX_fCANtoHs	WORD	Data for MLD

Tab. 10-28: Return value of the "MX_fCANtoHs" firmware function

10.14 MX_fHtoCANI

Swap functions to simplify the data exchange between host (MLD) and CAN. They encapsulate the endians and are used for programming the data exchange via CAN in a way that is compatible across systems.

The "MX_fHtoCANI" function receives a 32-bit value in host byte order (MLD) and returns a 32-bit value in the byte order of the CAN network.

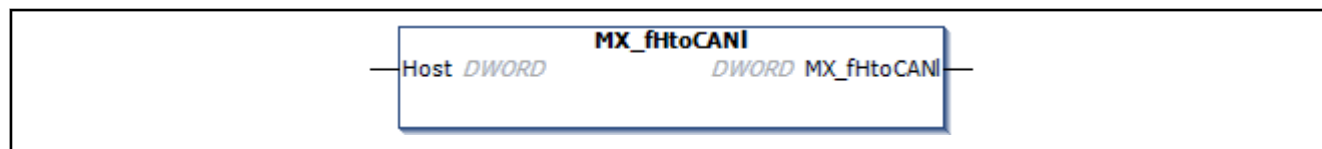


Fig. 10-13: "MX_fHtoCANI" firmware function

Input variables	Data type	Description
Host	DWORD	Data from MLD

Tab. 10-29: Input variables of the "MX_fHtoCANI" firmware function

Return value	Data type	Description
MX_fHtoCANI	DWORD	Data for CAN

Tab. 10-30: Return value of the "MX_fHtoCANI" firmware function

10.15 MX_fHtoCANs

Swap functions to simplify the data exchange between host (MLD) and CAN. They encapsulate the endians and are used for programming the data exchange via CAN in a way that is compatible across systems.

The "MX_fHtoCANs" function receives a 16-bit value in host byte order (MLD) and returns a 16-bit value in the byte order of the CAN network.

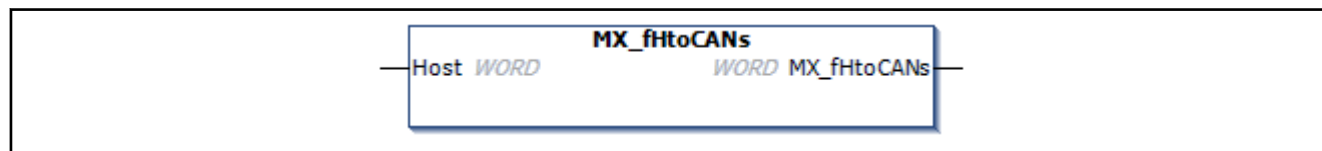


Fig. 10-14: "MX_fHtoCANs" firmware function

Input variables	Data type	Description
Host	WORD	Data from MLD

Tab. 10-31: Input variables of the "MX_fHtoCANs" firmware function

MX_CanL2

Return value	Data type	Description
MX_fHtoCANs	WORD	Data for CAN

Tab. 10-32: Return value of the "MX_fHtoCANs" firmware function

11 MX_CheckRtv



The functions of the "MX_CheckRtv" library are implicitly called by the PLC program and must not be included in a user program!

The "MX_CheckRtv" library contains functions for checking or signaling runtime errors, such as division by zero.

When a new project with the "IndraDrive" or "HydraulicDrive" target is created, the corresponding library is automatically included and visible.

Operating Principle

The functions of the "CheckRtv" library are called at each corresponding action (e.g. division) and check the access. If access is allowed, the program runs without restriction. If access is not allowed, a drive error (F6010) is generated and the PLC is stopped.

See also "Notes on Application" in the Application Manual on Rexroth IndraMotion MLD.

Functions

- **CheckBounds**
- **CheckDivReal**
- **CheckDivLReal**
- **CheckPointer**
- **CheckRangeSigned**
- **CheckRangeUnsigned**

12 MX_Debug

The functions contained in the "MX_Debug" library are required for internal testing purposes (laboratory).

⚠ WARNING

Property damage, death or serious injury due to incorrect control of motors when using "MX_Debug" library functions!

The functions from the "MX_Debug" library are required for internal testing purposes and should not be accessed by the user!

13 MX_PLCoPen

13.1 Introduction

The "MX_PLCoPen.library" provides the following function blocks/functions:

- Functions for diagnostics
- Function blocks/functions for drive control
- Function blocks/functions for parameters
- Functions for scaling
- General functions

13.2 "Drive control" - Functions for Diagnostics

13.2.1 MX_fCheckCmdRequest

The firmware function "MX_fCheckCmdRequest" is used to return the command input which was preset via the drive parameter "P-0-1449, C4900 PLC command".

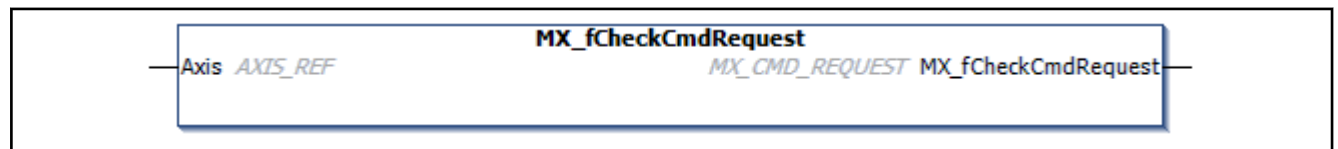


Fig. 13-1: Firmware Function "MX_fCheckCmdRequest"

Input variable	Data type	Description
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-1: Input Variables of the Firmware Function "MX_fCheckCmdRequest"

Return value	Data type	Description
MX_fCheckCmdRequest	MX_CMD_REQUEST	The function "MX_fCheckCmdRequest" can return the following values: • MX_CMD_REQ_ERROR: An invalid axis number was transmitted • MX_CMD_REQ_STOP: Command is to be terminated • MX_CMD_REQ_BREAK: Command is to be interrupted • MX_CMD_REQ_START: Command is to be started

Tab. 13-2: Return Value of the Firmware Function "MX_fCheckCmdRequest"

13.2.2 MX_fGetDriveWarning

The firmware function "MX_fGetDriveWarning" allows polling whether a certain warning diagnosis is active in the drive.

Depending on the value at the "WarningNo" input (1 to 7), the warning messages E2011 ... E2017 are checked.

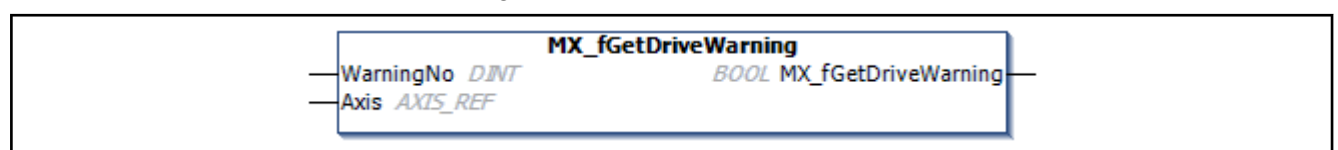


Fig. 13-2: Firmware Function "MX_fGetDriveWarning"

MX_PLCOpen

Input variable	Data type	Description
WarningNo	DINT	Order number of warning to be read
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-3: Input Variables of the Firmware Function "MX_fGetDriveWarning"

Return value	Data type	Description
MX_fGetDriveWarning	BOOL	If the warning to be checked is active in the drive, the function provides TRUE as return value, otherwise FALSE

Tab. 13-4: Return Value of the Firmware Function "MX_fGetDriveWarning"

13.2.3 MX_fSetDriveError

By means of the firmware function "MX_fSetDriveError", an error diagnosis can be triggered in the drive.

Depending on the value at the "ErrorNo" input (1 to 7), the errors F2011 ... F2017 are triggered in the drive.

The error diagnosis can be cleared by executing the function block "MC_Reset".

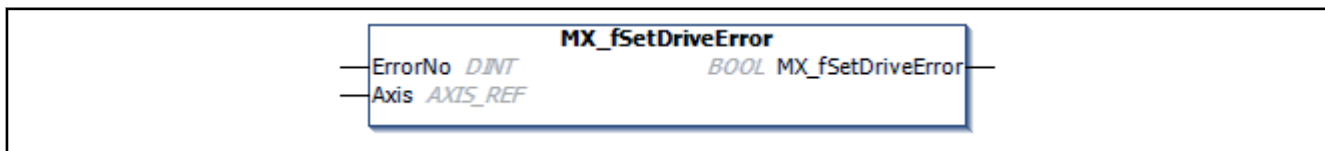


Fig. 13-3: Firmware Function "MX_fSetDriveError"

Input variable	Data type	Description
ErrorNo	DINT	Order number of error to be triggered
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-5: Input Variables of the Firmware Function "MX_fSetDriveError"

Return value	Data type	Description
MX_fSetDriveError	BOOL	If the value of "ErrorNo" is within the allowed range (1 to 7), the function provides TRUE as return value, otherwise FALSE

Tab. 13-6: Return Value of the Firmware Function "MX_fSetDriveError"

13.2.4 MX_fSetDriveErrorText

With the firmware function "MX_fSetDriveErrorText", the diagnostic text of the PLC errors F2011 to F2017 can be changed.

By switching the drive off and on again, the PLC errors are initialized again with their standard diagnostic texts (e.g. "PLC - Error no. 1").

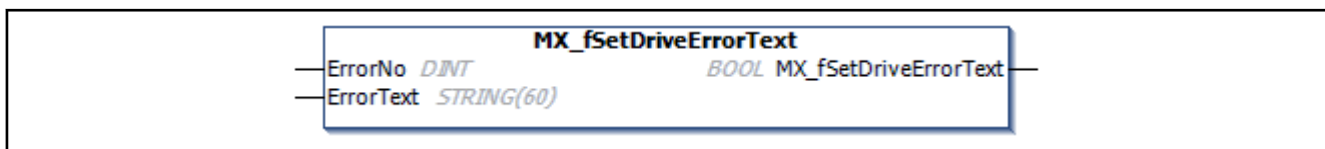


Fig. 13-4: Firmware Function "MX_fSetDriveErrorText"

Input variable	Data type	Description
ErrorNo	DINT	Order number of the PLC error the diagnostic text of which is to be changed: • "F2011 PLC - Error no. 1": "ErrorNo": "1" • "F2012 PLC - Error no. 2": "ErrorNo": "2" • ... • "F2017 PLC - Error no. 7": "ErrorNo": "7"
ErrorText	STRING (60)	Diagnostic text to be changed; maximum length: 60 characters (including string end "\0")

Tab. 13-7: Input Variable of the Firmware Function "MX_fSetDriveErrorText"

Return value	Data type	Description
MX_fSetDriveErrorText	BOOL	When the diagnostic text was successfully changed, the function provides the return value TRUE, otherwise FALSE

Tab. 13-8: Return Value of the Firmware Function "MX_fSetDriveErrorText"

13.2.5 MX_fSetDriveWarning

The firmware function "MX_fSetDriveWarning" is used to set or clear a warning diagnosis in the drive.

Depending on the value at the "WarningNo" input (1 to 7), the warning messages E2011 ... E2017 can be triggered in the drive.

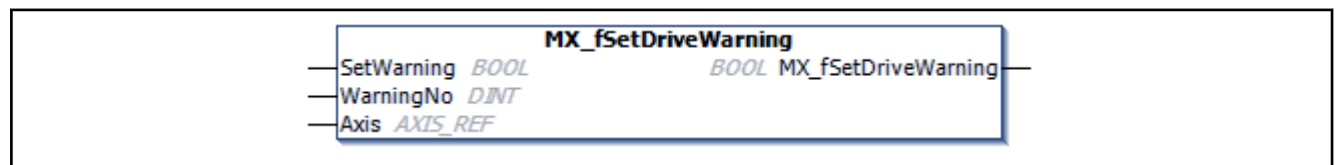


Fig. 13-5: Firmware Function "MX_fSetDriveWarning"

Input variable	Data type	Description
SetWarning	BOOL	TRUE: set warning, FALSE: clear warning
WarningNo	DINT	Order number of warning to be triggered
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-9: Input Variables of the Firmware Function "MX_fSetDriveWarning"

Return value	Data type	Description
MX_fSetDriveWarning	BOOL	If the value of "WarningNo" is within the allowed range (1 to 5), the function provides TRUE as return value, otherwise FALSE

Tab. 13-10: Return Value of the Firmware Function "MX_fSetDriveWarning"

13.2.6 MX_fSetDriveWarningText

With the firmware function "MX_fSetDriveWarningText", the diagnostic text of the PLC warnings E2011 to E2017 can be changed.

By switching the drive off and on again, the PLC warnings are initialized again with their standard diagnostic texts (e.g. "PLC - Warning no. 1").

MX_PLCOpen

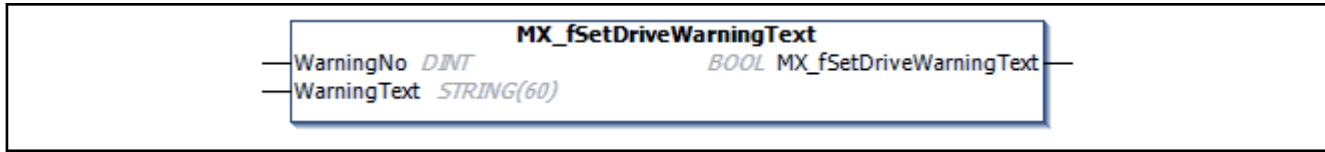


Fig. 13-6: Firmware Function "MX_fSetDriveWarningText"

Input variable	Data type	Description
WarningNo	DINT	Order number of the PLC warning the diagnostic text of which is to be changed: "E2011 PLC - Warning no. 1": "WarningNo": "1" "E2012 PLC - Warning no. 2": "WarningNo": "2" ... "E2017 PLC - Warning no. 7": "WarningNo": "7"
ErrorText	STRING (60)	Diagnostic text to be changed; maximum length: 60 characters (including string end "\0")

Tab. 13-11: Input Variable of the Firmware Function "MX_fSetDriveWarningText"

Return value	Data type	Description
MX_fSetDriveWarningText	BOOL	When the diagnostic text was successfully changed, the function provides the return value TRUE, otherwise FALSE

Tab. 13-12: Return Value of the Firmware Function "MX_fSetDriveWarningText"

13.2.7 MX_SetCmdState

With the firmware function block "MX_SetCmdState" it is possible to set the command status of "P-0-1449, C4900 PLC command".

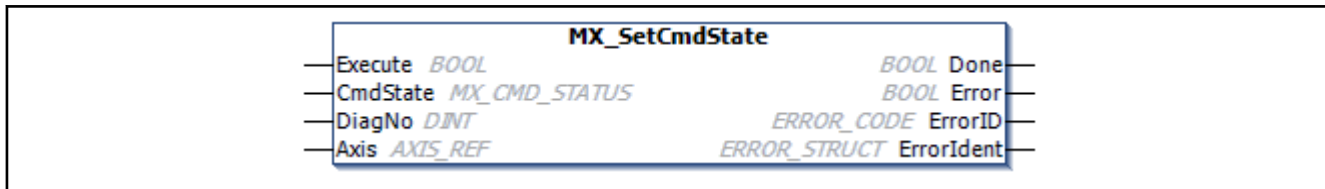


Fig. 13-7: Firmware Function "MX_SetCmdState"

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
CmdStatus	MX_CMD_STATUS	At the "CmdStatus" input, the following values can be transmitted to the function block: "MX_CMD_NO_ACTIVE": Command has not been set "MX_CMD_SETTING": Command has been set "MX_CMD_END_OK": Command completed without error "MX_CMD_IS_BREAK": Command interrupted "MX_CMD_IS_WORK": Command has been set and enabled "MX_CMD_BREAK_ERROR": Command aborted with error

Input variable	Data type	Description
DiagNo	DINT	For the case "CmdStatus"==MX_CMD_BREAK_ERROR, a command error number can be transmitted. 1: C4901 PLC command error no. 1 2: C4902 PLC command error no. 2 3: C4903 PLC command error no. 3 4: C4904 PLC command error no. 4
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-13: Input Variables of the Firmware Function "MX_SetCmdState"

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the function block signals that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description
ErrorIdent *	ERROR_STRUCT	Via the elements: Table, Additional1 and Additional2, "ErrorIdent" shows the exact cause of the error

* See note below

Tab. 13-14: Output Variables of the Firmware Function Block "MX_SetCmdState"



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

13.3 "Motion" - Function blocks/functions for drive control

13.3.1 General properties

The function blocks/functions for drive control have the following general properties:

- Positioning function blocks have no input for "jerk"
- Function blocks have "AXIS_REF" as "VAR_IN_OUT"
- Positions and distances are transmitted as REAL (floating point)

13.3.2 MB_ChangeCamData

Brief description

The "MB_ChangeCamData" function block supports loading of cam profiles to the designated drive parameters. At present, drives of the "Rexroth IndraDrive" product range support 8 cams. 4 cams with a maximum of 1024 cam elements and 4 cams with a maximum of 128 cam elements.

Starting with the location in the PLC memory indicated by the "CamTableAdr" pointer, this function block transfers the volume of 4-byte data specified by the "NumberOfElements" input to the cam parameter selected with "CamTableID".

The table below shows the assignment between the "CamTableID" input, the corresponding drive parameter and the possible number of cam elements for the "NumberOfElements" input.

MX_PLCOpen

CamTableID	Drive parameter	NumberOfElements
CAM_TABLE_1	P-0-0072, Cam table 1	0 or 3 to 1024
CAM_TABLE_2	P-0-0092, Cam table 2	0 or 3 to 1024
CAM_TABLE_3	P-0-0780, cam table 3	0 or 3 to 1024
CAM_TABLE_4	P-0-0781, cam table 4	0 or 3 to 1024
CAM_TABLE_5	P-0-0782, cam table 5	0 or 3 to 128
CAM_TABLE_6	P-0-0783, cam table 6	0 or 3 to 128
CAM_TABLE_7	P-0-0784, cam table 7	0 or 3 to 128
CAM_TABLE_8	P-0-0785, cam table 8	0 or 3 to 128

Tab. 13-15: Assignment between "CamTableID", drive parameter and "NumberOfElements"

The individual data to be transmitted may be in a range from -799,999,999 to +799,999,999. In the drive, the individual data to be transmitted are converted to a percentage with 6 decimal places, i.e., the value to be transmitted is divided by 1,000,000.



The function block does not check the content of the data to be transmitted, so the user has to make sure that the data volume and data type (DINT) of the information to be transmitted are correct.

The time required for transmitting cam profiles depends on various factors. These factors include

- Location of the target axis (local axis or remote CCD slave axis)
- Control section being used
- Number of elements to be transmitted
- Cycle time of the PLC task
- Active CCD cycle time

Typical times required to transmit **1024** cam elements:

Control section	Local axis	Remote CCD slave axis
CSH1.1	6 - 15 ms	1024 * CCD cycle time + 250ms
CSB01.1 / CDB01.1	30 - 45 ms	1024 * CCD cycle time + 250ms

Tab. 13-16: Typical Times Required to Transmit 1024 Cam Elements



The specified times are **not** absolute.



When the cam profile is transmitted to a local axis, the PLC program only continues running when the "MB_ChangeCamData" function block signals with "Done"=TRUE that it has successfully written the data, or when it signals with "Error"=TRUE that an error has occurred.

Notes on application:

When the function block "MB_ChangeCamData" is used to write data to a local axis, a freewheeling PLC task or a cyclic PLC task with a big cycle time

should be used so that the watchdog (F6010) is not generated when the watchdog monitor has been activated.



In the drive, a cam profile is traveled by means of linear or cubic spline interpolation.

For cubic spline interpolation, coefficients are calculated when a cam profile is written; therefore, it takes about 5 ms longer to transmit a cam profile with 1024 elements compared to linear cam interpolation.

Linear cam interpolation is activated via "P-0-0086, Configuration word synchronous operation modes", bit 7="1".



Drives of the "Rexroth IndraDrive" product range can process two types of cam formats (see "P-0-0072, Cam table 1").

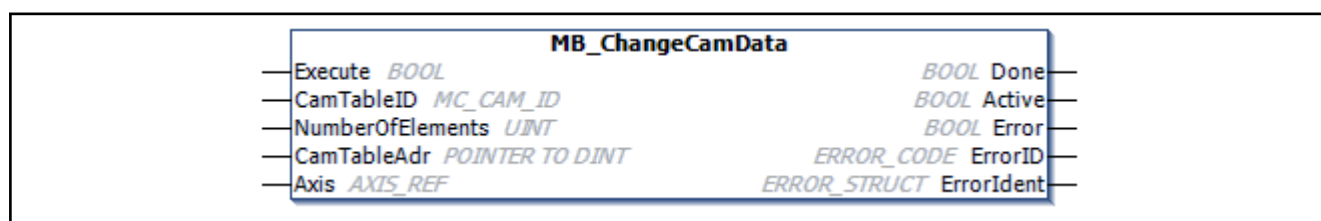


Fig. 13-8: "MB_ChangeCamData" function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
CamTableID	MC_CAM_ID	Number of the cam table to be written (1..8 according to parameters P-0-0072, P-0-0092, P-0-0780..P-0-0785)
NumberOfElements	UINT	Number of data points that are to be transmitted to the specified cam table
CamTableAdr	POINTER TO DINT	Pointer to PLC memory range containing the data to be transferred. Example: CamTableAdr := ADR(arFeld1)

Tab. 13-17: Input variables of the "MB_ChangeCamData" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the cam data were transmitted
Active	BOOL	The function block signals with "Active"=TRUE that it has accepted the processing command.
Error	BOOL	The function block signals with "Error"=TRUE that an error has occurred
ErrorID *	ERROR_CODE	"ErrorID" provides a general description of the error
ErrorIdent *	ERROR_STRUCT	Via the elements, "ErrorIdent" returns: Table, Additional1 and Additional2 return the exact cause of the error

*

See note below

Tab. 13-18: *Output variables of the "MB_ChangeCamData" firmware function block*



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Axis number ; the following global variables can be transmitted here: "Axis1" (axis for MLD-S) "Axis1" (local axis or only axis for MLD-M), "Axis2" to "Axis8" (first or last slave in the order of the list "P-0-4031, Overview of device addresses")

Tab. 13-19: *Input/output variable of the "MB_ChangeCamData" firmware function block*

13.3.3 MB_ChangeProfileSet

The "MB_ChangeProfileSet" firmware function block is used to switch the set of a MotionProfile.

In the case of a positive edge at "Execute", those data are applied which describe the set of a motion profile. The data include

- the number of motion steps
- the master axis velocity, as well as the lists for
 - master axis initial position,
 - distance,
 - slave axis velocity and
 - mode.

For the lists, you have to indicate the initial address of an array in which the data are contained.



The number of elements in each of the 4 lists must be at least as high as the indicated number of motion steps.

"SetNumber" defines which set is changed.

The units of the data contained in the lists for "Distance" and "SlaveAxisVelocity" are defined by the scaling parameters of the axis (S-0-0076, S-0-0044).

The "Done" output is set when the set was completely transmitted and the changed profile was checked. The "CheckOk" output is set depending on the result of the check. If "CheckOk"=TRUE, no discrepancy was detected.

The data transmitted with "MB_ChangeProfileSet" are stored in volatile form. The function block can work on the local axis "Axis1" and on remote axes "Axis2" etc.

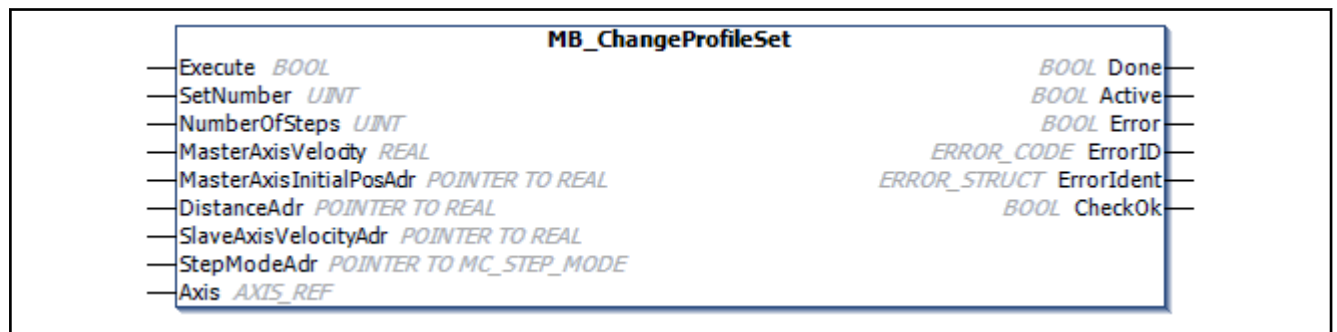


Fig. 13-9: "MB_ChangeProfileSet" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
SetNumber	UINT	Selection of motion profile set 0: Motion profile set 0 (parameters P-0-0703 to P-0-0708) 1: Motion profile set 1 (parameters P-0-0710 to P-0-0715)
NumberOfSteps	UINT	Number of motion steps Each motion profile set can consist of up to 8 motion steps - Parameter P-0-0703 for set 0 - Parameter P-0-0710 for set 1
MasterAxisVelocity	REAL	Master axis velocity Velocity of the master axis at the output of the electronic gear in rpm - Parameter P-0-0704 for set 0 - Parameter P-0-0711 for set 1
MasterAxisInitialPosAdr	POINTER TO REAL	Pointer to an array with the master axis initial positions of the steps Master axis initial positions indicated in degrees - Data for parameter P-0-0705 in set 0 - Data for parameter P-0-0712 in set 1
DistanceAdr	POINTER TO REAL	Pointer to an array with the distance values of the steps - Data for parameter P-0-0707 in set 0 - Data for parameter P-0-0714 in set 1

MX_PLCOpen

Input variable	Data type	Description
SlaveAxisVelocityAdr	POINTER TO REAL	Pointer to an array with the slave axis velocities of the steps - Data for parameter P-0-0708 in set 0 - Data for parameter P-0-0715 in set 1
StepModeAdr	POINTER TO MC_STEP_MODE	Pointer to an array with the modes of the steps - Data for parameter P-0-0706 in set 0 - Data for parameter P-0-0713 in set 1

Tab. 13-20: Input variables of the "MB_ChangeProfileSet" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE, the data for the motion set were applied "P-0-0089, Status word synchronization modes" (bit 12 for set 0, bit 14 for set 1)
Active	BOOL	With "Active"=TRUE, the data transfer is active
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
CheckOk	BOOL	With "CheckOk"=TRUE, the motion profile set is error-free "P-0-0089, Status word synchronization modes" (bit 13 for set 0, bit 15 for set 1)

* See note below

Tab. 13-21: Output variables of the "MB_ChangeProfileSet" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input / output variable	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-22: Input/output variable of the "MB_ChangeProfileSet" firmware function block

13.3.4 MB_ChangeProfileStep

The firmware function block "MB_ChangeProfileStep" is used to change the four parameters which define a segment of a motion profile set; these parameters are

- master axis initial position,

- distance,
- slave axis velocity and
- mode.

The step which is to be changed is defined by the values at the inputs "SetNumber" and "StepNumber".

The units for the function block inputs "Distance" and "SlaveAxisVelocity" are defined by the scaling parameters of the axis (S-0-0076, S-0-0044).

The "StepMode" input defines the kind of step.

The "Done" output is set when the changed profile was checked. The "CheckOk" output is set depending on the result of the check. If "CheckOk"=FALSE, a discrepancy has been detected.

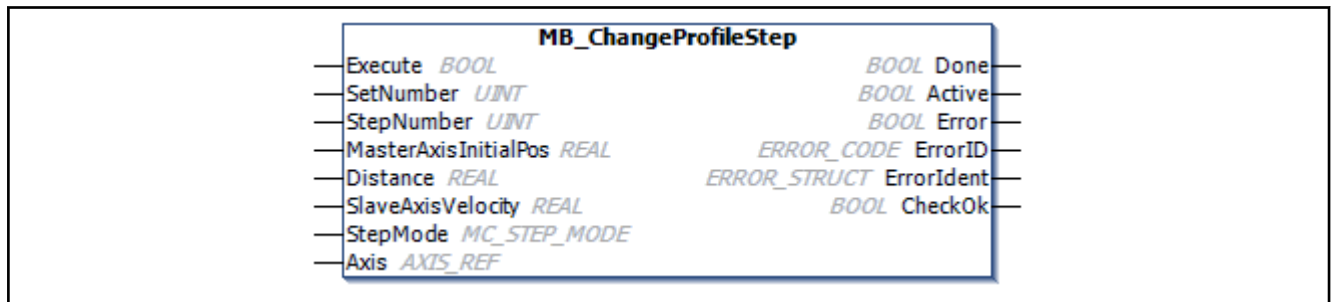


Fig. 13-10: "MB_ChangeProfileStep" firmware function block

MX_PLCopen

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
SetNumber	UINT	Selection of motion profile set 0: Motion profile set 0 (parameters P-0-0705 to P-0-0708) 1: Motion profile set 1 (parameters P-0-0712 to P-0-0715)
StepNumber	UINT	Selection of motion step to be changed Each motion profile set can consist of up to 8 motion steps 1: Motion step1 / list element 0 (from P-0-0705 to P-0-0708 or P-0-0712 to P-0-0715) 2: Motion step2 / list element 1 (from P-0-0705 to P-0-0708 or P-0-0712 to P-0-0715) ... 8: Motion step8 / list element 7 (from P-0-0705 to P-0-0708 or P-0-0712 to P-0-0715)
MasterAxisInitialPos	REAL	Setting of master axis initial position of selected motion step in degrees P-0-0705, List of master axis initial positions, set 0 P-0-0712, List of master axis initial positions, set 1
Distance	REAL	Setting of slave axis distance of motion step P-0-0707, List of distances, set 0 P-0-0714, List of distances, set 1
SlaveAxisVelocity	REAL	Setting of slave axis velocity within motion step P-0-0708, List of slave axis velocities, set 0 P-0-0715, List of slave axis velocities, set 1
StepMode	MC_STEP_MODE	Definition of processing mode of selected motion step P-0-0706, List of motion laws, set 0 P-0-0713, List of motion laws, set 1 • REST_IN_REST_INCLINEDSINE (0) • REST_IN_REST_POLY5 (16#100) • REST_IN_REST_POLY5_VLIM (16#1200) • REST_IN_VELOCITY_POLY5 (16#200) • VELOCITY_IN_REST_POLY5 (16#400) • VELOCITY_IN_VELOCITY_POLY5 (16#700) • CONSTANT_VELOCITY (16#600) • CAM_TABLE_1 (1) • CAM_TABLE_2 (2) • CAM_TABLE_3 (3) • CAM_TABLE_4 (4) • CAM_TABLE_5 (5) • CAM_TABLE_6 (6) • CAM_TABLE_7 (7) • CAM_TABLE_8 (8)

Tab. 13-23: Input variables of the "MB_ChangeProfileStep" firmware function

block

Output variable	Data type	Description
Done	BOOL	Data for motion step were applied "P-0-0089, Status word synchronization modes" (bit 12 for set 0, bit 14 for set 1)
Active	BOOL	Data transfer active
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
CheckOk	BOOL	Motion profile set error-free "P-0-0089, Status word synchronization modes" (bit 13 for set 0, bit 15 for set 1)

* See note below
 Tab. 13-24: *Output variables of the "MB_ChangeProfileStep" firmware function block*



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input / output variable	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-25: *Input/output variable of the "MB_ChangeProfileStep" firmware function block*

13.3.5 MB_ClearAllError

The firmware function block "MB_ClearAllError" is used to reset errors of all real activated devices and of all activated I/O modules with the command "S-0-0099, C0500 Reset class 1 diagnostics".

In order that the function block can correctly complete the reset command, it has to be cyclically called until it signals "Done"=TRUE or "Error" in the case of error.

If the command "S-0-0099, C0500 Reset class 1 diagnostics" is already active for a device or I/O module, it should first be completed before "MB_ClearAllError" is called.

If an error of a device or I/O module cannot be reset via the command "S-0-0099, C0500 Reset class 1 diagnostics", there is **no** error output at the function block.

In the case of deactivated axes or I/O modules, the function block does **not** output an error.

MX_PLCoPen

In the case of remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done" at a later call. This means the call times for this function block are always brief. The outputs "Done" and "Error" have to be cyclically polled. The data transmission can take several seconds, depending on the load of the service channel. Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase 2. Any attempt to do so results in an error.

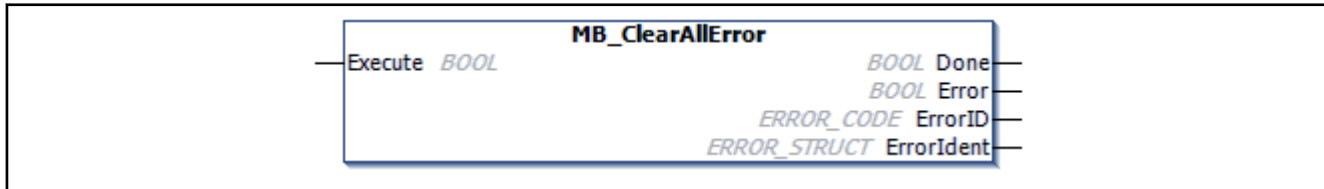


Fig. 13-11: "MB_ClearAllError" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"

Tab. 13-26: Input variable of the "MB_ClearAllError" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed.
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-27: Output variables of the "MB_ClearAllError" firmware function block



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

13.3.6 MB_GearInPos

The "MB_GearInPos" function block is used to move the slave axis in a phase-synchronous way to the master axis

The slave axis can be a real axis or the internal virtual master axis (VmAxis1).

When the slave axis is a real axis, a synchronous position command value for the slave axis is determined from the position of the master axis.

The master axis has to be selected at the "Master" input.



When "VmAxis1" has been selected as the slave axis, this axis cannot be the master axis at the same time.

The synchronous position command value " X_{Sync} " is essentially determined according to the equation below (see also Functional Description of firmware "Phase synchronization with real/virtual master axis"):

$$X_{\text{Sync}} = X_{\text{Master}} * \frac{\text{RatioNumerator}}{\text{RatioDenominator}} * (1 + \text{MasterFineadjust})$$

Fig. 13-12: Calculating the Synchronous Position Command Value "Xsync"

The mode in which the synchronization process is to be carried out has to be set at the "StartMode" input. The velocity is adjusted (relative synchronization) and so is the position in the case of absolute synchronization. "Adjusted" means that the actual values (velocity and position) are adjusted to the command values calculated from the master position. If "StartMode" "0" or "1" has been set, the synchronization process only occurs with the first positive edge at "Execute". Another positive edge only causes the data at the input of the function block to be applied.

For real axes: If the synchronization process is to occur with every positive edge at "Execute", the StartMode "2" or "3" can be set; the repeated synchronization is not supported for the virtual axis.

For a real axis, the parameters "P-0-0143, Synchronization velocity" and "P-0-0142, Synchronization acceleration" take effect during the synchronization process, for the virtual axis, it is the parameters "P-0-0920, Synchronization acceleration, secondary master" and "P-0-0921, Synchronization velocity, secondary master".

Synchronization can occur by motion in positive direction, in negative direction or over the shortest distance. This has to be set at the "SyncMode" input.

The position of the master axis is directly converted into a slave axis position.



When the function block "MB_GearInPos" is active, the position controller of the drive is also active for real axes. Make the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

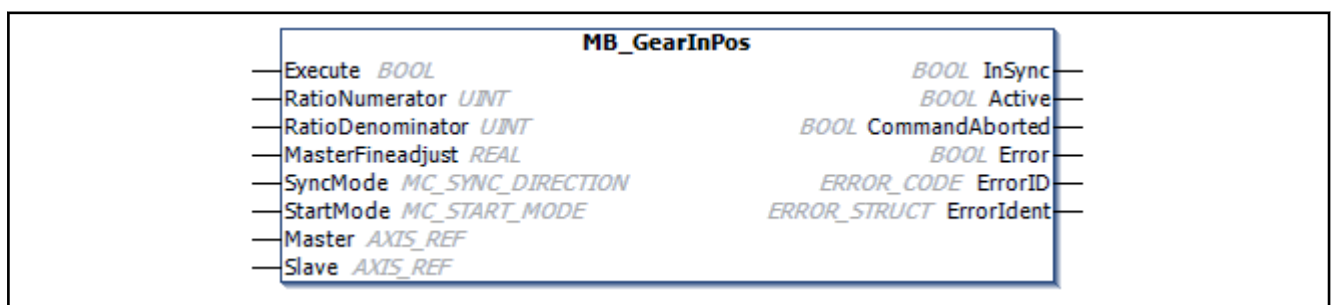


Fig. 13-13: "MB_GearInPos" firmware function block

MX_PLCOpen

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
RatioNumerator	UINT	Gear ratio - numerator • "P-0-0157, Master drive gear output revolutions" or • "P-0-0926, Master drive gear output revolutions, secondary master"
RatioDenominator	UINT	Gear ratio - denominator • "P-0-0156, Master drive gear input revolutions" or • "P-0-0925, Master drive gear input revolutions, secondary master"
MasterFineadjust	REAL	"P-0-0083, Gear ratio fine adjustment" or "P-0-0927, Master drive gear fine adjustment, secondary master" in %
SyncMode	MC_SYNC_DIRECTION	"P-0-0154, Synchronization direction" or "P-0-0922, Preferred synchronization direction, secondary master": 0:= Shortest distance 1:= Positive direction 2:= Negative direction
StartMode	MC_START_MODE	"P-0-0155, Synchronization mode", bit 1 or "P-0-0919, Synchronization mode, secondary master" 0:= Absolute synchronization only at first execution 1:= Relative synchronization only at first execution 2:= Absolute synchronization at every execution 3:= Relative synchronization at every execution

Tab. 13-28: Input variables of the "MB_GearInPos" firmware function block

Output variable	Data type	Description
InSync	BOOL	The function block uses the TRUE output value to signal that the slave axis is synchronous. "InSync" corresponds to P-0-0115, bit 12 of the parameterized axis. For this application, bit 12 of P-0-0115 corresponds to bit 8 (synchronization mode in synchronization) of "P-0-0089, Status word synchronization modes". For the virtual axis, "InSync" corresponds to bit 8 of "P-0-0768, Virtual master axis, positioning status"
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power" or "MX_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

*

See note below

Tab. 13-29: Output variables of the "MB_GearInPos" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description	
		MLD-S	MLD-M
Master	AXIS_REF	Master axis selection options	
		• VmAxis1^{*1} / VmAxisInt: P-0-0761 • RmAxis1^{*1} / RmAxisInt: P-0-0052 • LinkAxis1^{*1} / LinkAxisInt: P-0-0053	• VmAxisInt: P-0-0761^{*3} • RmAxisInt: P-0-0052 • LinkAxisInt: P-0-0053 for Axis1, P-0-0787 for Axis2..10 • VmAxis1^{*2}: P-0-0761^{*3} • RmAxis1^{*2}: P-0-1820.0.3 • LinkAxis1^{*2}: P-0-0053 • Axisn^{*2}: P-0-1808.x.10 or P-0-1821.x.1 (selected in P-0-1820.0.4)
		*1: can be used as master axis with MPx-20 and above	*2: has to be configured as master axis in P-0-1820.0.1, is adjusted *3: in order for P-0-0761 to return the correct value, a selection parameter has to be entered in P-0-0916
Slave	AXIS_REF	Slave axis selection options	
		• Axis1 • VmAxis1^{*4}	• VmAxis1^{*4} • Axis1 • Axis2..Axis10 The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)
		*4: In case VmAxis1 is used as slave, the secondary master mode is activated. The virtual master axis can follow two different master axes: • RmAxisInt: P-0-0052 • LinkAxisInt: P-0-0053	

P-0-0052: Actual position value of measuring encoder

P-0-0053: Master axis position

P-0-0761: Master axis position for slave axis

P-0-0787: Group axis 1 position

P-0-0916: Master axis format converter signal selection

P-0-1808.x.10: CCD: Active actual position value

P-0-1820.0.1: CCD: Master axes configuration list

P-0-1820.0.3: CCD: Actual position value of measuring encoder

P-0-1820.0.4: CCD: Axis position selection

P-0-1821.x.1: CCD: Position command value for master axis n

Tab. 13-30: Input/output variable of the "MB_GearInPos" firmware function block

13.3.7 MB_Home

With the function block "MB_Home" it is possible to carry out the "drive-controlled homing procedure" in the selected axis.

It is thereby possible to establish the position data reference for non-absolute encoders.

In the homing parameter, details for carrying out the homing procedure have to be set. After the function block was carried out, the axis is in standstill.

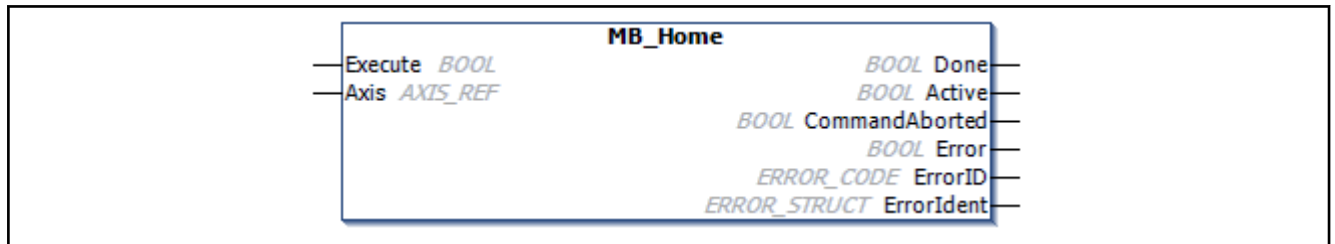


Fig. 13-14: "MB_Home" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"

Tab. 13-31: Input variable of the "MB_Home" firmware function block

Output variable	Data type	Description
Done	BOOL	Homing completed, velocity 0
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred.
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-32: Output variables of the "MB_Home" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-33: Input/output variable of the "MB_Home" firmware function block

13.3.8 MB_MotionProfile

The "MB_MotionProfile" function block implements an electronic cam function. The cams are analytically defined by the parameters belonging to the "electronic motion profile" mode. A synchronous position command value for the slave axis is determined from the position of the master axis.

The master axis has to be selected at the "Master" input.

To calculate the synchronous position command value " X_{Synch} ", a table access angle (P-0-0227) is first determined according to the equation below:

$$\varphi_{\text{Access}} = \varphi_{\text{Master}} * \frac{\text{RatioNumerator}}{\text{RatioDenominator}} * (1 + \text{MasterFineadjust})$$

Fig. 13-15: Table access angle

With the table access angle as an argument, a synchronous position command value is calculated using a 5th order polynomial.

The mode in which the synchronization process is to be carried out has to be set at the "StartMode" input. The velocity is adjusted (relative synchronization) and so is the position in the case of absolute synchronization. "Adjusted" means that the actual values (velocity and position) are adjusted to the command values calculated from the master position. If StartMode "0" or "1" has been set, the synchronization process only occurs with the first positive edge at "Execute". Another positive edge only causes the data at the input of the function block to be applied.

If the synchronization process is to occur with every positive edge at "Execute", StartMode "2" or "3" can be set.

The parameters "P-0-0143, Synchronization velocity" and "P-0-0142, Synchronization acceleration" take effect for the synchronization process.



If the function block "MB_MotionProfile" is active, the position controller of the drive is active, too. The "MB_SetPositionControlMode" function block must be used to set the position controller (with/without lag error, actual position value 1 / 2).

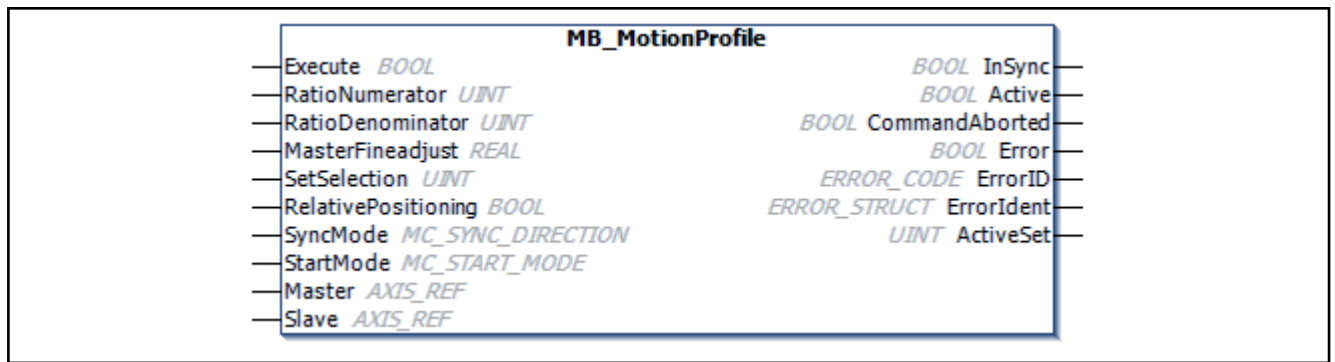


Fig. 13-16: "MB_MotionProfile" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
RatioNumerator	UINT	Gear ratio - numerator P-0-0157, Master drive gear output revolutions
RatioDenominator	UINT	Gear ratio - denominator P-0-0156, Master drive gear input revolutions
MasterFineadjust	REAL	"P-0-0083, Gear ratio fine adjustment" in %
SetSelection	UINT	P-0-0088, Bit9: Selection of motion profile set (0 or 1)
RelativePositioning	BOOL	P-0-0088, Bit10: FALSE = fixed (absolute) position reference, TRUE = relative positioning.
SyncMode	MC_SYNC_DIRECTION	"P-0-0154, Synchronization direction": 0:= Shortest distance 1:= Positive direction 2:= Negative direction
StartMode	MC_START_MODE	"P-0-0155, Synchronization mode", bit 1 0:= Absolute synchronization only at first execution 1:= Relative synchronization only at first execution 2:= Absolute synchronization at every execution 3:= Relative synchronization at every execution

Tab. 13-34: Input variables of the "MB_MotionProfile" firmware function block

MX_PLCOpen

Output variable	Data type	Description
InSync	BOOL	The function block uses the TRUE output value to signal that the slave axis is synchronous. "InSync" corresponds to P-0-0115, bit 12 of the parameterized axis and SetSelection(P-0-0088.9) = ActiveSet(P-0-0089.3). For this application, bit 12 of P-0-0115 corresponds to bit 8 "synchronization mode in synchronization" of parameter "P-0-0089, Status word synchronization modes".
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power" or "MX_Power".
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
ActiveSet	UINT	P-0-0089, Bit3: The output displays the number of the active motion profile set.

*

See note below

Tab. 13-35: Output variables of the "MB_MotionProfile" firmware function block

NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
		MLD-SMLD-M
Master	AXIS_REF	Master axis selection options
		- VmAxis1^{*1} / VmAxisInt: P-0-0761 - RmAxis1^{*1} / RmAxisInt: P-0-0052 - LinkAxis1^{*1} / LinkAxisInt: P-0-0053
		- VmAxisInt: P-0-0761^{*3} - RmAxisInt: P-0-0052 - LinkAxisInt: P-0-0053 for Axis1, P-0-0787 for Axis2..10 - VmAxis1^{*2}: P-0-0761^{*3} - RmAxis1^{*2}: P-0-1820.0.3 - LinkAxis1^{*2}: P-0-0053 - Axisn^{*2}: P-0-1808.x.10 or P-0-1821.x.1 (selection in P-0-1820.0.4)
		*1: from MPx-20, can be used as master axis *2: has to be configured as master axis in P-0-1820.0.1, is adjusted *3: in order for P-0-0761 to return the correct value, a selection parameter has to be entered in P-0-0916
Slave	AXIS_REF	Slave axis selection options
		Axis1 - Axis1 - Axis2..Axis10 The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

P-0-0052: Actual position value of measuring encoder

P-0-0053: Master axis position

P-0-0761: Master axis position for slave axis

P-0-0787: Group axis 1 position

P-0-0916: Master axis format converter signal selection

P-0-1808.x.10: CCD: Active actual position value

P-0-1820.0.1: CCD: Master axes configuration list

P-0-1820.0.3: CCD: Actual position value of measuring encoder

P-0-1820.0.4: CCD: Axis position selection

P-0-1821.x.1: CCD: Position command value for master axis n

Tab. 13-36: Input/output variables of the firmware function block

13.3.9 MB_Phasing

The function block "MB_Phasing" allows realizing an offset to the master axis which is connected as input to a synchronization function block.

The parameter "P-0-0054, Additive master axis position" is changed. The "Velocity" and "Acceleration" inputs determine the motion profile with which the distance is traveled.

The units of the inputs which define the drive motion are fixed.

MX_PLCOpen

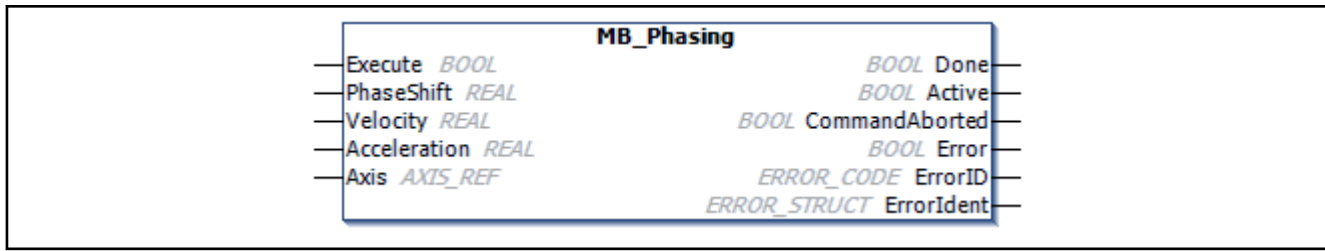


Fig. 13-17: "MB_Phasing" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
PhaseShift	REAL	Additive master axis position (P-0-0054) in degrees
Velocity	REAL	Additive master axis position, positioning velocity (P-0-0688) in rpm
Acceleration	REAL	Additive master axis position, positioning acceleration (P-0-0689) in rad/s ²

Tab. 13-37: Input variables of the "MB_Phasing" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that PhaseShift has been reached (acknowledged by bit 6 of "P-0-0089, Status word synchronization modes")
Active	BOOL	With TRUE the "Active" input signals that the preset value has not yet been reached
CommandAborted	BOOL	The "CommandAborted" output is set to TRUE, when the firmware function block has not yet signaled with "Done"=TRUE that it has been completed and the firmware function block was aborted with another motion function block
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-38: Output variables of the "MB_Phasing" firmware function block



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-39: Input/output variable of the "MB_Phasing" firmware function block

13.3.10 MB_PhasingSlave

With the function block "MB_PhasingSlave" it is possible to adjust the slave axis by a position offset. The adjustment only takes effect with active synchronization function block ("MC_CamIn", "MB_GearInPos" or "MB_MotionProfile").

Moving to the new value for the position offset ("PhaseShift" input) takes place in consideration of the values for velocity ("Velocity" input) and acceleration ("Acceleration" input) provided at the function block.

When using "MB_PhasingSlave" with a real axis (Axis1...Axis10), the units for the "PhaseShift", "Velocity" and "Acceleration" inputs are specified by the scaling parameters of the axis as follows:

- "PhaseShift": "S-0-0076, Position data scaling type"
- "Velocity": "S-0-0044, Velocity data scaling type"
- "Acceleration": "S-0-0160, Acceleration data scaling type"

When using the "MB_PhasingSlave" with the virtual master axis $VmAxis1$ (secondary axis mode), the units for the "PhaseShift", "Velocity" and "Acceleration" inputs are specified as follows:

- "PhaseShift": "P-0-0928, Additive master axis position, secondary master"
- "Velocity": "P-0-0929, Change accel. of add. master axis posit., secondary master"
- "Acceleration": "P-0-0930, Change accel. of add. master axis posit., secondary master")

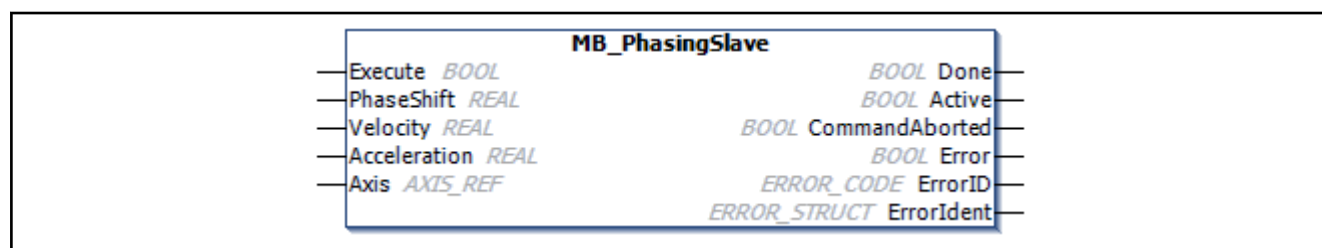


Fig. 13-18: "MB_PhasingSlave" firmware function block

MX_PLCOpen

Input variable	Data type	Description	
Execute	BOOL	See "Signal Time Diagrams"	
		Real axis	Virtual master axis
PhaseShift	REAL	S-0-0048, Additive position command value	P-0-0928, Additive master axis position, secondary master
Velocity	REAL	P-0-0686, Additive position command value, positioning velocity	P-0-0929, Change accel. of add. master axis posit., secondary master
Acceleration	REAL	P-0-0687, Additive position command value, positioning acceleration	P-0-0930, Change accel. of add. master axis posit., secondary master

Tab. 13-40: Input variables of the "MB_PhasingSlave" firmware function block

Output variable	Data type	Description
Done	BOOL	PhaseShift reached (acknowledged by "P-0-0152, Synchronization completed", bit 0=1)
Active	BOOL	With "Active"=TRUE the firmware function block signals that it contributes to the axis motion
CommandAborted	BOOL	With "CommandAborted"=TRUE the firmware function block signals that the synchronization was aborted
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-41: Output variables of the "MB_PhasingSlave" firmware function block

NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Internal virtual axis: "VmAxis1"=P-0-0761

Tab. 13-42: Input/output variable of the "MB_PhasingSlave" firmware function block

13.3.11 MB_PreSetMode

With the function block "MB_PreSetMode" the axis can be prepared for command triggering before power is switched on (PreSetMode / preselecting operation modes).

"MB_PreSetMode" already commands the axis with the corresponding data, but the axis only starts when power is switched on with "MC_Power" and then directly follows the input values.



"MB_PreSetMode" can only be activated in the operating mode without power and without error. After activation with a rising "Execute" edge it is allowed to activate a motion function block (such as "MC_MoveVelocity") or "MX_SetOpMode".

The following points exemplify a typical sequence when using the function block "MB_PreSetMode":

1. Switch drive on
2. Switch or wait until drive in operating mode ("bb")
3. Activate "MB_PreSetMode" with rising "Execute" edge
4. Start a motion function block (in "Ab")
5. Switch power on with "MC_Power"

The axis immediately starts moving in the desired way

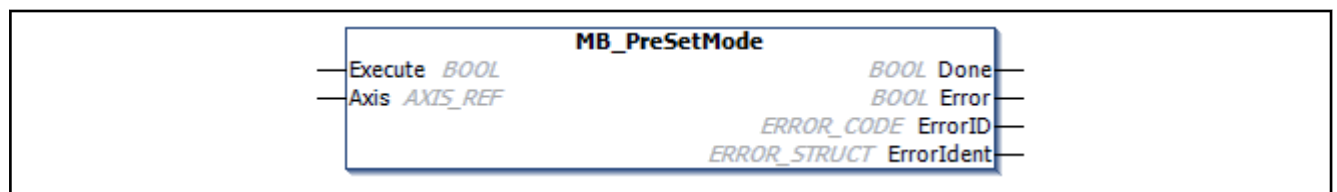


Fig. 13-19: "MB_PreSetMode" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"

Tab. 13-43: Input variable of the "MB_PreSetMode" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Value	DINT	If the output "Done"=TRUE, "Value" displays the value of the selected element

* See note below

Tab. 13-44: Output variables of the "MB_PreSetMode" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-45: Input/output variable of the "MB_PreSetMode" firmware function block

13.3.12 MB_SetPositionControlMode

The "MB_SetPositionControlMode" function block is used to set the position controller.



MoveAbsTransitionMode The input is available in MPx-20 / HDx-20 and above

Fig. 13-20: "MB_SetPositionControlMode" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
LaglessPositionControl	BOOL	The value at the "LaglessPositionControl" input affects bit 2 of "S-0-0520, Axis control word": FALSE: Position control with lag error TRUE: Position control without lag error
ExternalEncoder	BOOL	The value at the "ExternalEncoder" input affects bit 0 of "S-0-0520, Axis control word": FALSE: Control with motor encoder (actual position value-1) TRUE: Control with external encoder (actual position value-2)
MoveAbsUseCmdPos	BOOL	The value at the "MoveAbsUseCmdPos" input affects bit 4 of "S-0-0346, Positioning control word": FALSE: Current actual position value as dedicated point for relative positioning command values TRUE: Last effective target position as dedicated point for relative positioning command values
MoveAbsTransitionMode	MB_TRANSITION_MODE	NOTE: The input is available in MPx-20 / HDx-20 and above The value at the "MoveAbsTransitionMode" input affects bit 5, 6 and 7 of "S-0-0346, Positioning control word": MB_ABORTING: Applied directly (default) MB_BUFFERED: Start after intermediate stop MB_OLD_VELOCITY: Use sequential block with old velocity MB_NEW_VELOCITY: Use sequential block with new velocity

Tab. 13-46: Input variables of the "MB_SetPositionControlMode" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	The function block is processed very quickly; the change of states of the "Active" output cannot be evaluated with IndraMotion MLD
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID*	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-47: Output variables of the "MB_SetPositionControlMode" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-48: *Input/output variable of the "MB_SetPositionControlMode" firmware function block*

13.3.13 MB_Stop

The firmware function block "MB_Stop" is used to bring the drive to the status STOP.

With the rising edge at "ExecuteLock" the stopping process is started.

If the PLC does not have control over the drive, an error is returned.

As long as the "ExecuteLock" input is TRUE, no other motion function block can be started. For this purpose, the drive is brought to the status "AH" (Drive Halt). With "ExecuteLock"=FALSE the drive remains in "AH".



The "Drive Halt" function is described in the functional description of the firmware.

With the deceleration set at "Deceleration" the drive is brought to standstill.

When several instances are called by "MC_Stop" or "MB_Stop" in a project, it is always the last instance which takes effect; **exception:** An active "MC_Stop" cannot be interrupted by an "MB_Stop".

"MC_Stop" has higher priority than "MB_Stop"; this means that a running "MC_Stop" cannot be aborted by an "MB_Stop".



Via the function block "MB_Stop" it is also possible to bring "P-0-0759, Virtual master axis, actual velocity value" to standstill via a defined ramp. For this purpose, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter P-0-0758 must be written to parameter "P-0-0916, Master axis format converter signal selection".

See also Functional Description of firmware "Master Axis Generator"



The "ActivateCounter" instance variable can be used to test the function block. It counts the positive edges of the "ExecuteLock" signal and thereby signals whether the function block is run with an edge.

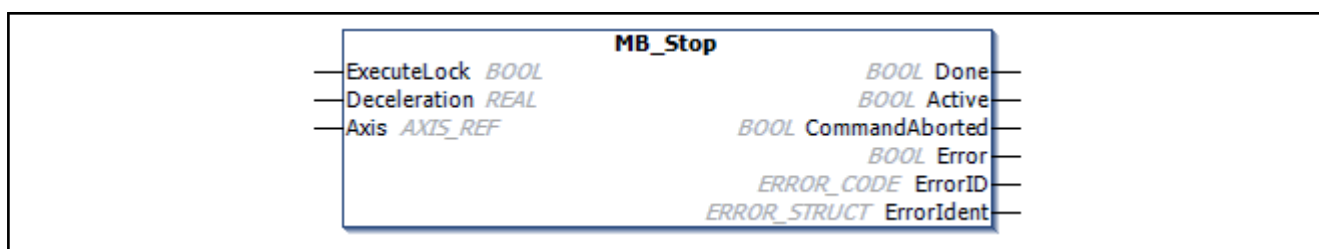


Fig. 13-21: "MB_Stop" firmware function block

Input variable	Data type	Description
ExecuteLock	BOOL	With a rising edge at "ExecuteLock" the drive is switched to "AH".
Deceleration	REAL	This input sets the deceleration. The deceleration which is set is written to parameter "S-0-0372, Drive Halt acceleration bipolar".

Tab. 13-49: Input variables of the "MB_Stop" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE (S-0-0331, bit0="1") the firmware function block signals that the drive has stopped. Only when the "ExecuteLock" input is TRUE is the "Done" output generated
Active	BOOL	With "Active"=TRUE the firmware function block signals that it determines the axis motion
CommandAborted	BOOL	With "CommandAborted"=TRUE the function block signals that another function block has commanded another status (other "MC_Stop" or "MB_Stop")
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-50: Output variables of the "MB_Stop" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input / output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-51: Input/output variable of the "MB_Stop" firmware function block

MX_PLCoPen

13.3.14 MC_CamIn

With the function block "MC_CamIn" it is possible to realize an electronic cam.

A synchronous position command value for the slave axis is determined from the position of the master axis. The master axis has to be selected at the "Master" input.

To calculate the synchronous position command value "XSynch", a table access angle (P-0-0227) is first determined according to the equation below:

$$\varphi_{Access} = \varphi_{Master} * \frac{\text{RatioNumerator}}{\text{RatioDenominator}} * (1 + \text{MasterFineadjust})$$

Fig. 13-22: Calculating P-0-0227, Cam Table, Access Angle (Table Access Angle)

With the calculated table access angle, the table selected at the "CamTableID" input is accessed and a percentage is taken from it. This table value is multiplied with the position value at the "CamShaftDistance" input and provides the synchronous position command value.

The mode in which the synchronization process is to be carried out has to be set at the "StartMode" input. The velocity is adjusted (relative synchronization) and so is the position in the case of absolute synchronization. "Adjustment" means that the actual values (velocity and position) are adjusted to the command values calculated from the master position with parameterized velocity (P-0-0143, Synchronization velocity) and acceleration (P-0-0142, Synchronization acceleration). If StartMode "0" or "1" has been set, the synchronization process only occurs with the first positive edge at "Execute". Another positive edge only causes the data at the input of the function block to be applied. The input values (gear, distance, cam) then take effect depending on the configuration of "P-0-0086, Configuration word synchronous operation modes".

If the synchronization process is to occur with every positive edge at "Execute", StartMode "2" or "3" can be set. The input values (gear, distance, cam) therewith take effect immediately, independent of P-0-0086.

With modulo position scaling, synchronization can take place by motion in positive direction, in negative direction or over the shortest distance. This has to be set at the "SyncMode" input.



When the function block "MC_CamIn" is active, the position controller of the drive is active, too. The "MB_SetPositionControlMode" function block must be used to set the position controller (with/without lag error, actual position value 1 / 2).

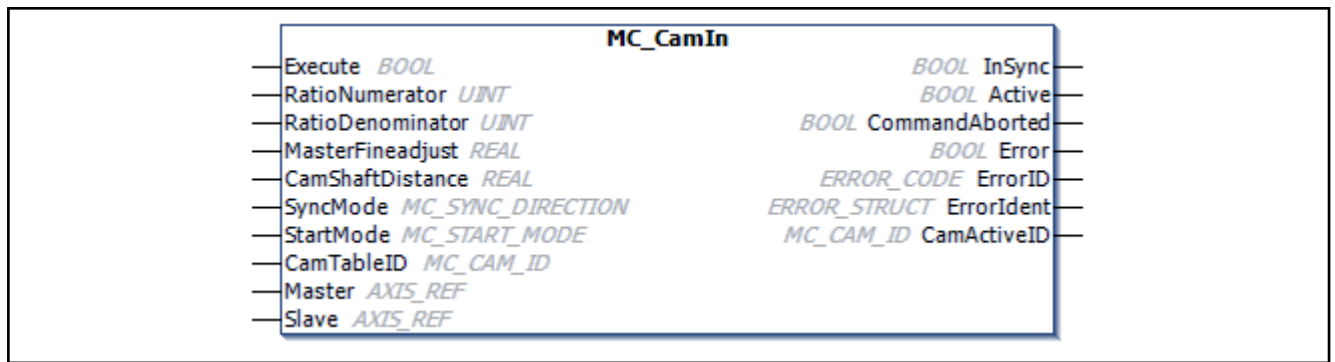


Fig. 13-23: "MC_CamIn" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
RatioNumerator	UINT	Gear ratio - numerator P-0-0157, Master drive gear output revolutions
RatioDenominator	UINT	Gear ratio - denominator P-0-0156, Master drive gear input revolutions
MasterFineadjust	REAL	"P-0-0083, Gear ratio fine adjustment" in %
CamShaftDistance	REAL	Cam distance. Scaling of percentage-based table values
SyncMode	MC_SYNC_DIRECTION	"P-0-0154, Synchronization direction": 0:= Shortest distance 1:= Positive direction 2:= Negative direction
StartMode	MC_START_MODE	"P-0-0155, Synchronization mode", bit 1 0:= Absolute synchronization only at first execution 1:= Relative synchronization only at first execution 2:= Absolute synchronization at every execution 3:= Relative synchronization at every execution
CamTableID	MC_CAM_ID	Number of cam table (1..8 according to parameters P-0-0072, P-0-0092, P-0-0780..P-0-0785)

Tab. 13-52: Input variables of the "MC_CamIn" firmware function block

MX_PLCOpen

Output variable	Data type	Description
InSync	BOOL	With "InSync"=TRUE the function block signals that the slave axis is synchronous. "InSync" corresponds to • "P-0-0115, Device control: Status word", bit12: "Command value reached" of the parameterized axis and • "CamTableID"="CamActiveID" and • "P-0-0089, Status word synchronization modes", bit5="1" (status distance switching). For this application, bit 12 of P-0-0115 corresponds to bit 8 "synchronization mode in synchronization" of parameter "P-0-0089, Status word synchronization modes".
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the "Error" outputs of "MC_Power".
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
CamActiveID		Number of the cam active in the function block

*

See note below

Tab. 13-53: *Output variables of the "MC_CamIn" firmware function block*

NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description	
		MLD-S	MLD-M
Master	AXIS_REF	Master axis selection options	
		• VmAxis1^{*1} / VmAxisInt: P-0-0761 • RmAxis1^{*1} / RmAxisInt: P-0-0052 • LinkAxis1^{*1} / LinkAxisInt: P-0-0053	• VmAxisInt: P-0-0761^{*3} • RmAxisInt: P-0-0052 • LinkAxisInt: P-0-0053 for Axis1, P-0-0787 for Axis2..10 • VmAxis1^{*2}: P-0-0761^{*3} • RmAxis1^{*2}: P-0-1820.0.3 • LinkAxis1^{*2}: P-0-0053 • Axisn^{*2}: P-0-1808.x.10 or P-0-1821.x.1 (selection in P-0-1820.0.4)
		*1: from MPx-20, can be used as master axis	*2: has to be configured as master axis in P-0-1820.0.1, is adjusted *3: in order for P-0-0761 to return the correct value, a selection parameter has to be entered in P-0-0916
Slave	AXIS_REF	Slave axis selection options	
		Axis1	• Axis1 • Axis2..Axis10 The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

P-0-0052: Actual position value of measuring encoder

P-0-0053: Master axis position

P-0-0761: Master axis position for slave axis

P-0-0787: Group axis 1 position

P-0-0916: Master axis format converter signal selection

P-0-1808.x.10: CCD: Active actual position value

P-0-1820.0.1: CCD: Master axes configuration list

P-0-1820.0.3: CCD: Actual position value of measuring encoder

P-0-1820.0.4: CCD: Axis position selection

P-0-1821.x.1: CCD: Position command value for master axis n

Tab. 13-54: Input/output variables of the firmware function block

13.3.15 MC_CamOut

The firmware function block "MC_CamOut" is used to terminate an active synchronization function block. Transition to velocity control takes place. The actual velocity value available with a rising Execute edge is taken as the velocity command value.

MX_PLCOpen

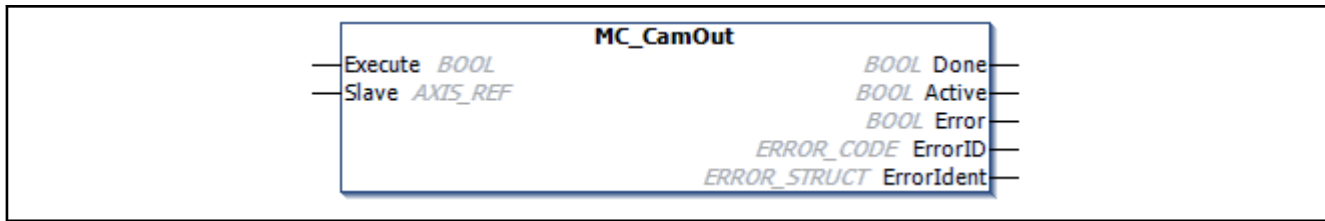


Fig. 13-24: "MC_CamOut" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"

Tab. 13-55: Input variables of the "MC_CamOut" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the preset velocity has been reached. (S-0-0330, Bit0=1)
Active	BOOL	With TRUE the "Active" output signals that it determines the axis motion.
Error	BOOL	The function block signals with "Error"=TRUE that an error has occurred
ErrorID *	ERROR_CODE	"ErrorID" provides a general description of the error
ErrorIdent *	ERROR_STRUCT	Via the elements, "ErrorIdent" returns: Table, Additional1 and Additional2 return the exact cause of the error

* See note below

Tab. 13-56: Output variables of the "MC_CamOut" firmware function block



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input / output variable (VAR_IN_OUT)	Data type	Description
Slave	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" (P-0-0761, Master axis position for slave axis)

Tab. 13-57: Input/output variables of the "MC_CamOut" firmware function block

13.3.16 MC_GearIn

With the function block "MC_GearIn" it is possible to realize a motion of the slave axis which is velocity synchronous to the master axis.

When the function block is activated, the slave axis accelerates to the synchronous velocity with the acceleration set in "P-0-0142, Synchronization acceleration". The synchronous velocity is generated according to the equation below, from the velocity of the master axis and the input data of the function block.

$$V_{\text{sync}} = V_{\text{Master}} * \frac{\text{RatioNumerator}}{\text{RatioDenominator}} * (1 + \text{MasterFineadjust})$$

Fig. 13-25: Calculating the synchronous velocity

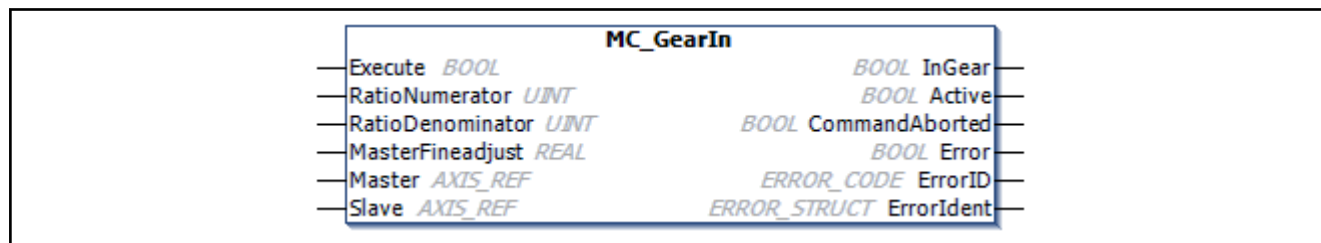


Fig. 13-26: "MC_GearIn" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
RatioNumerator	UINT	Gear ratio - numerator P-0-0157, Master drive gear output revolutions
RatioDenominator	UINT	Gear ratio - denominator P-0-0156, Master drive gear input revolutions
MasterFineadjust	REAL	"P-0-0083, Gear ratio fine adjustment" in %

Tab. 13-58: Input variables of the "MC_GearIn" firmware function block

Output variable	Data type	Description
InGear	BOOL	With "InGear"=TRUE the function block signals that the slave axis is velocity-synchronous. "InGear" corresponds to "P-0-0115, Device control: Status word", bit12 of the parameterized axis. For this application, this bit corresponds to bit8 "synchronization mode in synchronization" of parameter "P-0-0089, Status word synchronization modes".
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-59: Output variables of the "MC_GearIn" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description	
		MLD-S	MLD-M
Master	AXIS_REF	Master axis selection options	
		• VmAxis1^{*1} / VmAxisInt: P-0-0761 • RmAxis1^{*1} / RmAxisInt: P-0-0052 • LinkAxis1^{*1} / LinkAxisInt: P-0-0053	• VmAxisInt: P-0-0761^{*3} • RmAxisInt: P-0-0052 • LinkAxisInt: P-0-0053 for Axis1, P-0-0787 for Axis2..10 • VmAxis1^{*2}: P-0-0761^{*3} • RmAxis1^{*2}: P-0-1820.0.3 • LinkAxis1^{*2}: P-0-0053 • Axisn^{*2}: P-0-1808.x.10 or P-0-1821.x.1 (selection in P-0-1820.0.4)
		*1: from MPx-20, can be used as master axis	*2: has to be configured as master axis in P-0-1820.0.1, is adjusted *3: in order for P-0-0761 to return the correct value, a selection parameter has to be entered in P-0-0916
Slave	AXIS_REF	Slave axis selection options	
		Axis1	• Axis1 • Axis2..Axis10 The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

P-0-0052: Actual position value of measuring encoder

P-0-0053: Master axis position

P-0-0761: Master axis position for slave axis

P-0-0787: Group axis 1 position

P-0-0916: Master axis format converter signal selection

P-0-1808.x.10: CCD: Active actual position value

P-0-1820.0.1: CCD: Master axes configuration list

P-0-1820.0.3: CCD: Actual position value of measuring encoder

P-0-1820.0.4: CCD: Axis position selection

P-0-1821.x.1: CCD: Position command value for master axis n

Tab. 13-60: Input/output variables of the firmware function block

13.3.17 MC_GearOut

The firmware function block "MC_GearOut" is used to terminate an active synchronization function block. Transition to velocity control takes place for real axes. The actual velocity value available with a rising "Execute" edge is taken as the velocity command value.

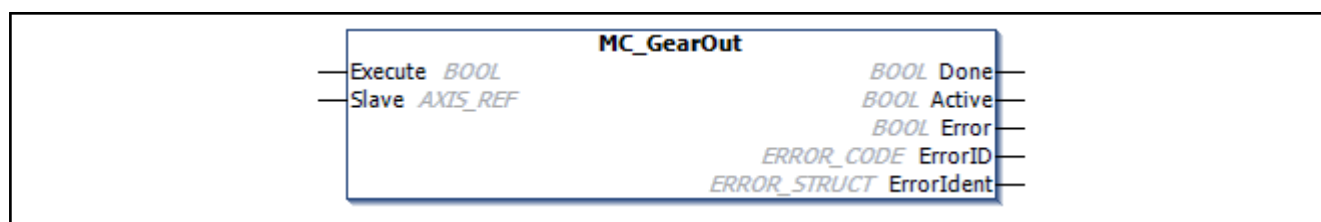


Fig. 13-27: "MC_GearOut" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"

Tab. 13-61: Input variables of the "MC_GearOut" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the preset velocity has been reached. (S-0-0330, bit0="1")
Active	BOOL	With TRUE the "Active" output signals that it determines the axis motion.
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-62: Output variables of the "MC_GearOut" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input / output variable (VAR_IN_OUT)	Data type	Description
Slave	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" (P-0-0761, Master axis position for slave axis) (as of IndraDrive MP05)

Tab. 13-63: Input/output variable of the "MC_GearOut" firmware function block

13.3.18 MC_Jog

The "MC_Jog" firmware function block is used to jog the drive ("infinite travel" positive or "infinite travel" negative). For this purpose, the jog mode is internally activated in the "drive-controlled positioning" mode.

The secondary operation mode 3 and its parameters are used for permanent control and remote axes. The secondary PLC operation mode 11 and its parameters are used for temporary control.

MX_PLCOpen

The jog velocity must be set at the "JogVelo" input. How the jog velocity is reached must be determined via the "JogAcc" and "JogDec" inputs.



Only one command triggering can be preset in a PLC cycle.



The "MC_Jog" firmware function block can also be used to specify a jog velocity for the virtual master axis generator. For this purpose, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1".

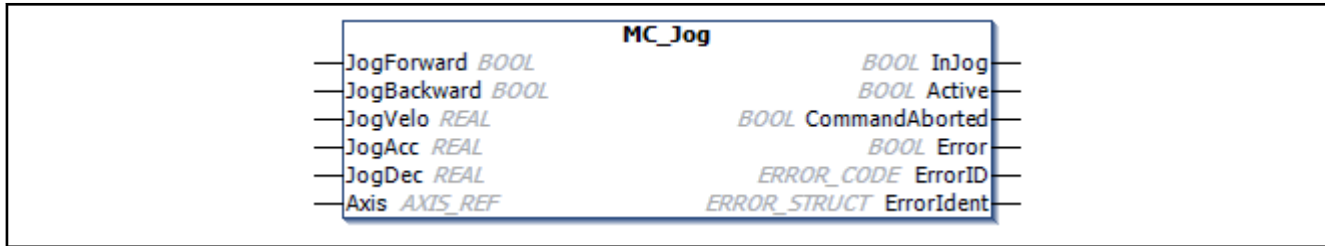


Fig. 13-28: "MC_Jog" firmware function block

Input variable	Data type	Description
JogForward	BOOL	The "JogVelo", "JogAcc", "JogDec" and "Axis" inputs are applied with a rising edge. The "drive-controlled positioning" mode is activated and "Jog positive" starts [positioning control word (real axis: S-0-0346 / virtual master axis: P-0-0760), bit 1/0=11]. When "JogForward" is reset (FALSE), positioning stop is commanded (real axis: S-0-0346 / virtual master axis: P-0-0760, bit2/1/0=111)
JogBackward	BOOL	The "JogVelo", "JogAcc", "JogDec" and "Axis" inputs are applied with a rising edge. The "drive-controlled positioning" mode is activated and "Jog negative" starts [positioning control word (real axis: S-0-0346 / virtual master axis: P-0-0760), bit2/1/0=101]. When "JogBackward" is reset (FALSE), positioning stop is commanded (real axis: S-0-0346 / virtual master axis: P-0-0760, bit2/1/0=111).
JogVelo	REAL	Positioning velocity (real axis: S-0-0259 / virtual master axis: P-0-0770) must be >="0". Note: When the function block is active, the "JogVelo" input is cyclically evaluated and applied when a change is made.
JogAcc	REAL	Positioning acceleration (real axis: S-0-0260 / virtual master axis: P-0-0771) must be >"0". Note: When the function block is active, the "JogAcc" input is cyclically evaluated and applied when a change is made.
JogDec	REAL	Positioning deceleration (real axis: S-0-0359 / virtual master axis: P-0-0772) must be >"0". Note: When the function block is active, the "JogDec" input is cyclically evaluated and applied when a change is made.

Tab. 13-64: Input variables of the "MC_Jog" firmware function block

Output variable	Data type	Description
InJog	BOOL	The function block signals with "InJog"=TRUE that the actual velocity is outside of the standstill window (real axis: S-0-0124 / virtual master axis: P-0-0912).
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred.
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-65: Output variables of the "MC_Jog" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

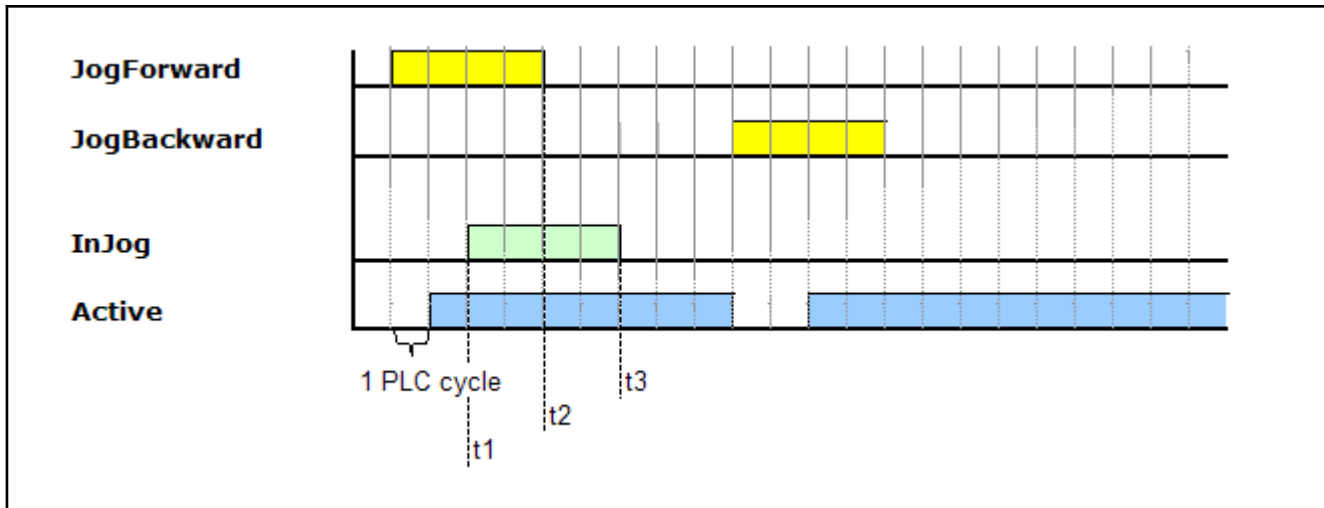
Tab. 13-66: Input/output variable of the "MC_Jog" firmware function block

Timing diagrams

The figure below shows the timing diagram for active jog mode ("jog positive" or "jog negative"). By a positive edge at the "JogForward" input, the input parameters ("JogVelo" etc.) are applied and infinite travel in positive direction (jog positive) is activated in the "drive-controlled positioning" mode. The "InJog" output is set when the actual velocity is greater than the standstill window (S-0-0124); "InJog" is removed when the actual velocity is smaller than the standstill window. "P-0-0912, Virtual master axis, standstill window" is used for the virtual master axis generator.

The "Active" output remains set until the function block is restarted (commanded) or interrupted by another function block instance (e.g., "MC_MoveAbsolute").

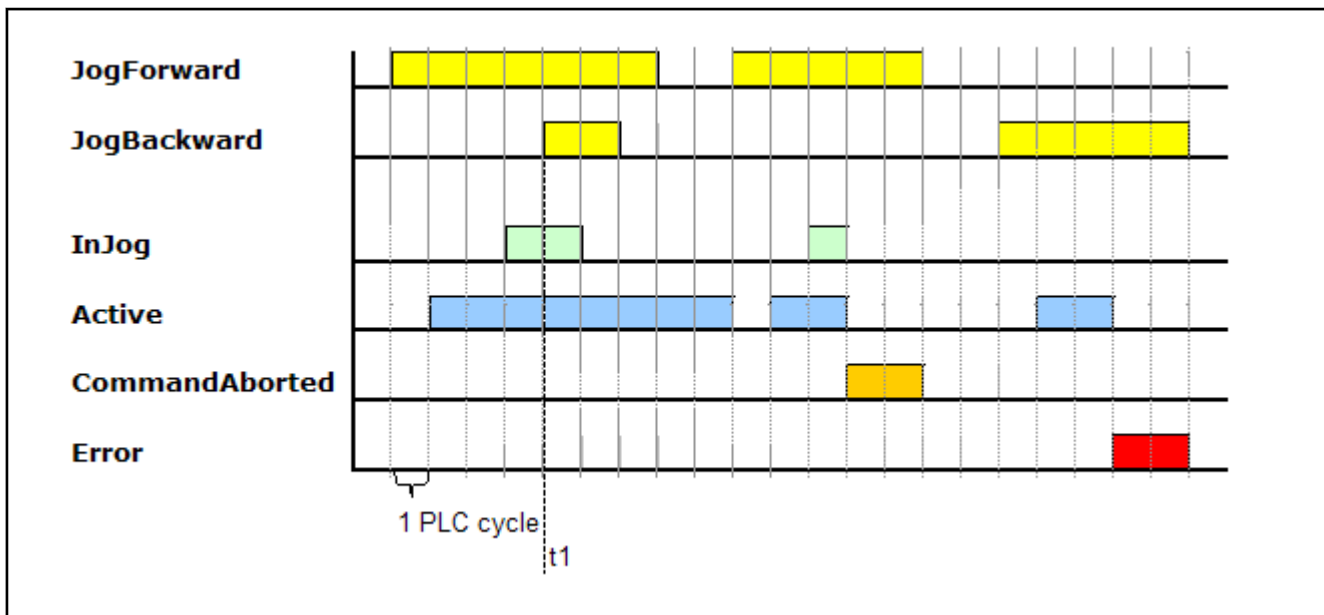
MX_PLCOpen



- t1** Actual velocity outside of standstill window (real axis: S-0-0124/ virtual master axis: P-0-0912)
t2 Axis is stopped
t3 Actual velocity within standstill window (real axis: S-0-0124 / virtual master axis: P-0-0912)

Fig. 13-29: Timing of the "Active" output

The figure below shows the timing behavior of the other standard outputs.



- t1** Axis is stopped

Fig. 13-30: Timing of "Active", "CommandAborted" and "Error" outputs

13.3.19 MC_MoveAbsolute

The firmware function block "MC_MoveAbsolute" is used to move the drive to a preset absolute position. Internally the operation mode "drive-controlled positioning" is activated for this purpose. Make the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

For permanent control and remote axes, the secondary operation mode 3 and its parameters are used. For temporary control, the secondary PLC operation mode 11 and its parameters are used.

Commanding of a virtual master axis

The target position must be set at the "Position" input. Via the inputs "Velocity", "Acceleration" and "Deceleration" you have to set how the drive moves to the target position (motion profile).

If a virtual master axis generator is to be positioned in absolute form via the "MC_MoveAbsolute" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").

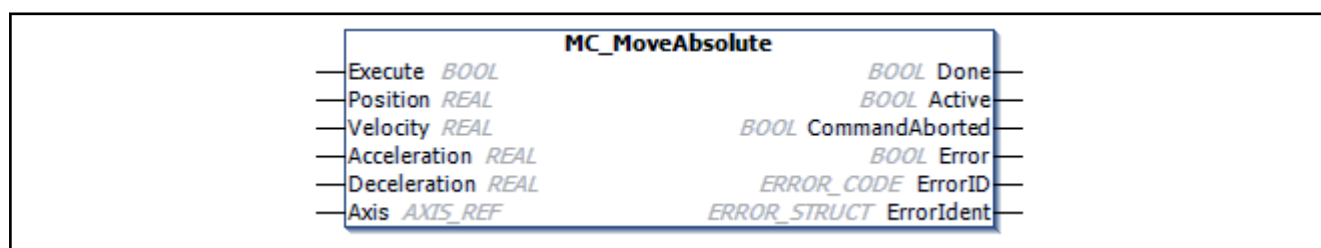


Fig. 13-31: "MC_MoveAbsolute" firmware function block



The following parameters must not be written cyclically as they are written by this function block or also other functions blocks:

- S-0-0346, Positioning control word
- P-0-1454, PLC/setting-up mode, positioning command value acceptance
- P-0-0760, Virtual master axis, positioning control word

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Position	REAL	The absolute position has to be preset at the "Position" input.	S-0-0259	P-0-1450	P-0-0766
Velocity	REAL	Positioning velocity in physical unit* ¹⁾	S-0-0259	P-0-1451	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit* ¹⁾	S-0-0260	P-0-1452	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit* ¹⁾	S-0-0359	P-0-1453	P-0-0772

① If PLC has permanent control over the axis or if a removed axis is commanded

② If PLC has temporary control over the axis

③ In case of a virtual master axis

*1) The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-67: Input variables of the "MC_MoveAbsolute" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the absolute target position has been reached. The processing is completed until the next 0-1- edge at the "Execute" input. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has been completed ("Done"=TRUE), the "Done" output is set to TRUE for one call when the motion target has been reached and the processing of the firmware function block has not been interrupted by an error or another firmware function block. - Real axes: The output is set to TRUE when bit 2 is set in parameter "S-0-0437, Positioning status". - Virtual master axis: The output is set to TRUE when bit 2 is set in parameter "P-0-0768, Virtual master axis, positioning status".
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

*

See note below

Tab. 13-68: Output variables of the "MC_MoveAbsolute" firmware function block

NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-69: Input/output variable of the "MC_MoveAbsolute" firmware function block

13.3.20 MC_MoveAdditive

The firmware function block "MC_MoveAdditive" is used to move the drive in relative form by a distance in addition to the target position. The new target position results from the old target position, plus a distance. The distance is

determined by the "Distance" input. The inputs "Velocity", "Acceleration" and "Deceleration" determine the motion profile.

Internally the operation mode "drive-controlled positioning" is activated. Make the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

For permanent control and in case of remote axes, the secondary operation mode 3 and its parameters are used. For temporary control, the secondary PLC operation mode 11 and its parameters are used.

Commanding of a virtual master axis

If a virtual master axis generator is to be traveled in relative form by a distance **in addition** to the current target position via the "MC_MoveAdditive" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").

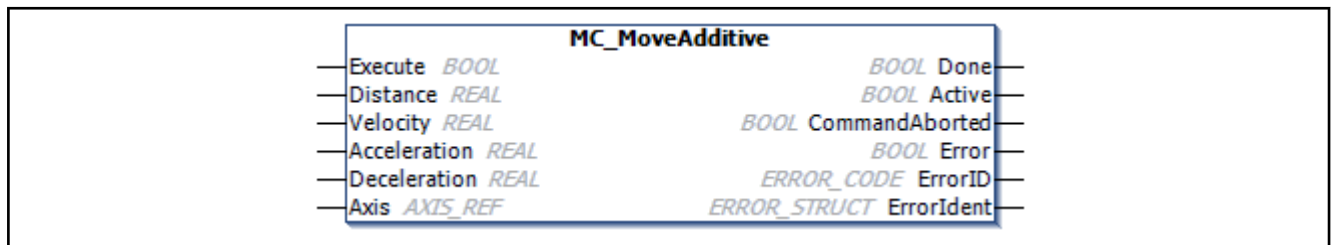


Fig. 13-32: "MC_MoveAdditive" firmware function block



The following parameters must not be written cyclically as they are written by this function block or also other functions blocks:

- S-0-0346, Positioning control word
- P-0-1454, PLC/setting-up mode, positioning command value acceptance
- P-0-0760, Virtual master axis, positioning control word

MX_PLCOpen

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Distance	REAL	At the "Distance" input the relative distance must be preset; the new target position results from the old target position and the value set at the "Distance" input.	S-0-0282	P-0-1450	P-0-0766
Velocity	REAL	Positioning velocity in physical unit ^{*1)}	S-0-0259	P-0-1451	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit ^{*1)}	S-0-0260	P-0-1452	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit ^{*1)}	S-0-0359	P-0-1453	P-0-0772

① If PLC has permanent control over the axis or if a removed axis is commanded

② If PLC has temporary control over the axis

③ In case of a virtual master axis

*1) The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-70: Input variables of the "MC_MoveAdditive" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the target position has been reached. The processing is completed until the next 0-1- edge at the "Execute" input. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has been completed ("Done"=TRUE), the "Done" output is set to TRUE for one call when the motion target has been reached and the processing of the firmware function block has not been interrupted by an error or another firmware function block. - Real axes: The output is set to TRUE when bit 2 is set in parameter "S-0-0437, Positioning status". - Virtual master axis: The output is set to TRUE when bit 2 is set in parameter P-0-0768, Virtual master axis, positioning status.
Active	BOOL	With "Active"=TRUE the firmware function block signals that it determines the axis motion.
CommandAborted	BOOL	With "CommandAborted"=TRUE the firmware function block signals that another motion function block was activated or a drive error was present. A drive error is displayed via the Error outputs of "MC_Power".
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-71: Output variables of the "MC_MoveAdditive" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-72: Input/output variable of the "MC_MoveAdditive" firmware function block

13.3.21 MC_MoveRelative

The firmware function block "MC_MoveRelative" is used to move the drive in relative form by a distance, starting from the current actual position.

Internally the operation mode "drive-controlled positioning" is activated for this purpose. Select the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

For permanent control and remote axes, the secondary operation mode 3 and its parameters are used. For temporary control, the secondary PLC operation mode 11 and its parameters are used.

The relative distance to be moved has to be preset at the "Distance" input. Via the inputs "Velocity", "Acceleration" and "Deceleration" you have to set the motion profile (determines how the relative distance is traveled).

Commanding of a virtual master axis

If a virtual master axis generator is to be traveled in relative form by a distance from the actual virtual position via the "MC_MoveRelative" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").



Note that the command value input is specified by "P-0-0769, Virtual master axis, command value mode" in case of modulo scaling of the virtual master axis generator (P-0-0756).

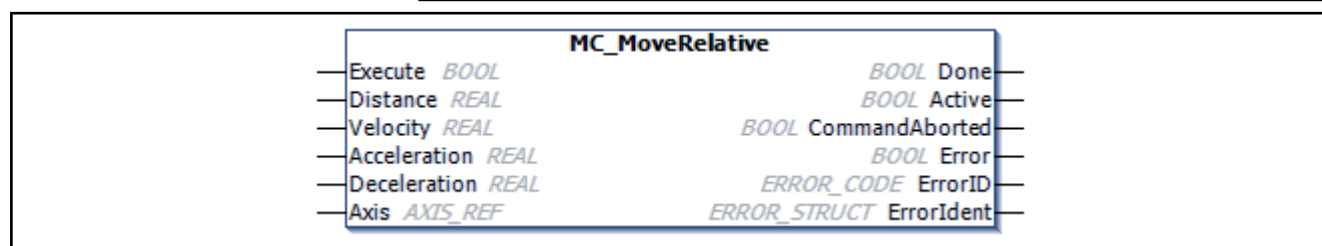


Fig. 13-33: "MC_MoveRelative" firmware function block

MX_PLCOpen



The following parameters must not be written cyclically as they are written by this function block or also other functions blocks:

- S-0-0346, Positioning control word
- P-0-1454, PLC/setting-up mode, positioning command value acceptance
- P-0-0760, Virtual master axis, positioning control word

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Distance	REAL	At the "Distance" input you have to preset the relative distance which is to be traveled starting from the current actual position*1)	S-0-0282	P-0-1450	P-0-0766
Velocity	REAL	Positioning velocity in physical unit*1)	S-0-0259	P-0-1451	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit*1)	S-0-0260	P-0-1452	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit*1)	S-0-0359	P-0-1453	P-0-0772

- ① If PLC has permanent control over the axis or if a removed axis is commanded
- ② If PLC has temporary control over the axis
- ③ In case of a virtual master axis
- *1) The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-73: Input variables of the "MC_MoveRelative" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the preset distance has been traveled. The processing is completed until the next 0-1-edge at the "Execute" input. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has been completed ("Done"=TRUE), the "Done" output is set to TRUE for one call when the motion target has been reached and the processing of the firmware function block has not been interrupted by an error or another firmware function block. - Real axes: The output is set to TRUE when bit 2 is set in parameter "S-0-0437, Positioning status". - Virtual master axis: The output is set to TRUE when bit 2 is set in parameter "P-0-0768, Virtual master axis, positioning status".
Active	BOOL	With TRUE the "Active" output signals that it determines the axis motion.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-74: Output variables of the "MC_MoveRelative" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-75: Input/output variable of the "MC_MoveRelative" firmware function block

13.3.22 MC_MoveVelocity

The firmware function block "MC_MoveVelocity" is used to preset a velocity for the drive. As the drive moves in velocity control, the firmware function block can be used for open-loop drives, too.

MX_PLCOpen

For permanent control and remote axes the primary operation mode and its parameters are used. For temporary control, the secondary PLC operation mode 8 and its parameters are used.



If they are scaling-dependent and unless otherwise described, the units of the inputs which define the drive motion are taken from the scaling parameters.

- S-0-0044, Velocity data scaling type
- S-0-0076, Position data scaling type
- S-0-0086, Torque/force data scaling type

Commanding of a virtual master axis

If a velocity is to be specified for the virtual master axis generator via the "MC_MoveVelocity" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").



Note that the command value input is specified by "P-0-0769, Virtual master axis, command value mode" in case of modulo scaling of the virtual master axis generator (P-0-0756).

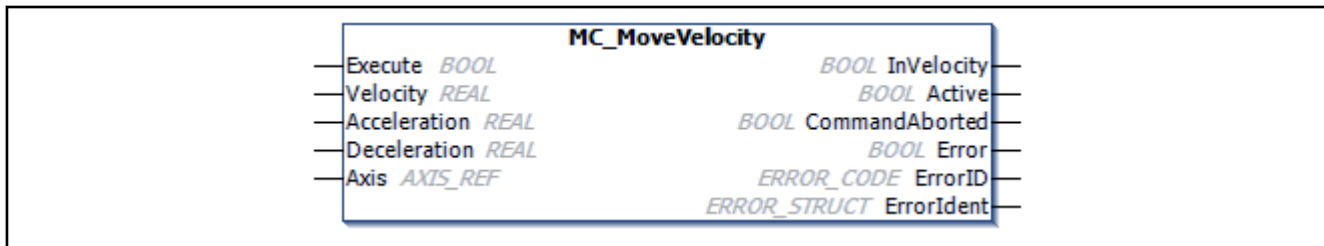


Fig. 13-34: "MC_MoveVelocity" firmware function block

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Velocity	REAL	Positioning velocity in physical unit ^{*1)}	S-0-0036	P-0-1460	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit ^{*1)}	P-0-1203	P-0-1461	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit ^{*1)}	P-0-1213	P-0-1463	P-0-0772

- ① If PLC has permanent control over the axis or if a removed axis is commanded
- ② If PLC has temporary control over the axis
- ③ In case of a virtual master axis
- *1) The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-76: Input variables of the "MC_MoveVelocity" firmware function block

Output variable	Data type	Description
InVelocity	BOOL	With "InVelocity"=TRUE the firmware function block signals that the velocity preset by "Velocity" has been reached. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has reached the command velocity ("InVelocity"=TRUE), the "InVelocity" output is set to TRUE for one call if the actual velocity corresponds to the command velocity. - Real axes: "Axis1" etc. InVelocity = "P-0-0115, Device control: Status word", bit 12 of the parameterized axis. In this application, bit 12 of P-0-0115 corresponds to "S-0-0330, Status "n_feedback = n_command". - Virtual master axis: "VmAxis1" InVelocity = bit4 AND bit13 of "P-0-0768, Virtual master axis, positioning status". - Bit4="1" → Generator supplies const. Velocity - Bit13="1" → P-0-0770 - P-0-0759 < 1Upm or <1mm/min
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-77: Output variables of the "MC_MoveVelocity" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-78: Input/output variable of the "MC_MoveVelocity" firmware function block

MX_PLCOpen

13.3.23 MC_Power

This function block is used to activate drive enable when the internal PLC has permanent or temporary control over the drive. Independent of the "Enable" input, the "Status" output returns the current drive status.

The "MC_Power" function block sets the PLC-internal "ON" signal to TRUE and thereby switches the drive on.

 The "ON" signal is the "ON" signal of the PLC that is linked to the external signal and becomes active in the drive, potentially with a delay (max. 1ms for ADVANCED, max. 2ms for BASIC or IndraDrive Mi). If several edges are generated (e.g., "TRUE - FALSE - TRUE"), this delay has to be taken into account.

If a WATCHDOG or an exception occurs in the PLC, the PLC-internal "ON" signal is automatically set to FALSE and remains FALSE until the PLC is reset.

Subsequent calls of "MC_Power" with TRUE signal errors.

If the drive is not yet ready for power on (e.g., not in "Ab"), the "Status" output provides FALSE and the "Error" output is TRUE.

 Querying `IF AxisData[Axis1.AxisNo].Axis_InAb THEN`, for example, checks whether or not the drive is ready for power on. For this query, the option for supporting the AxisData structure must have been activated in the PLC configuration (P-0-1367).

 "MC_Power" also provides an error if a motion function block has detected an error and shut down the axis.

After every error ("Error"=TRUE), a new 1-0- edge has to be applied to the "Enable" input to activate drive enable in the drive.

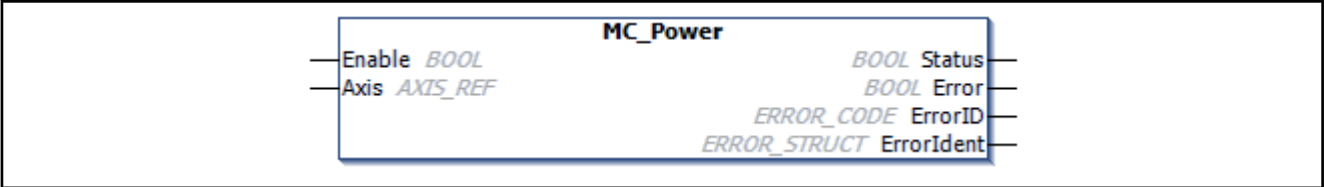


Fig. 13-35: "MC_Power" firmware function block

Input variable	Data type	Description
Enable	BOOL	Drive enable can be set for the drive via the "Enable" input. The input is edge-controlled for safety reasons. Drive enable is set with a rising edge and disabled with a falling edge.

Tab. 13-79: Input variables of the "MC_Power" firmware function block

Output variable	Data type	Description
Status	BOOL	Output "Status"=TRUE activates drive enable and the drive is under torque. Independent of the "Enable" input, the "Status" output returns the current drive status. For this purpose, the combination "drive in control and no error" is internally checked. Output "Status":= (P115.14 AND P115.15 AND NOT P115.13) This sets the "Status" output to FALSE in the case of drive errors
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-80: Output variables of the "MC_Power" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-81: Input/output variable of the "MC_Power" firmware function block

13.3.24 MC_ReadStatus

With the function block "MC_ReadStatus" it is possible to read the current status of the axis. The status of the axis is output in decoded form.



A modification of the axis at the input of the function block requires another 0/1 transition at "Enable".

MX_PLCOpen

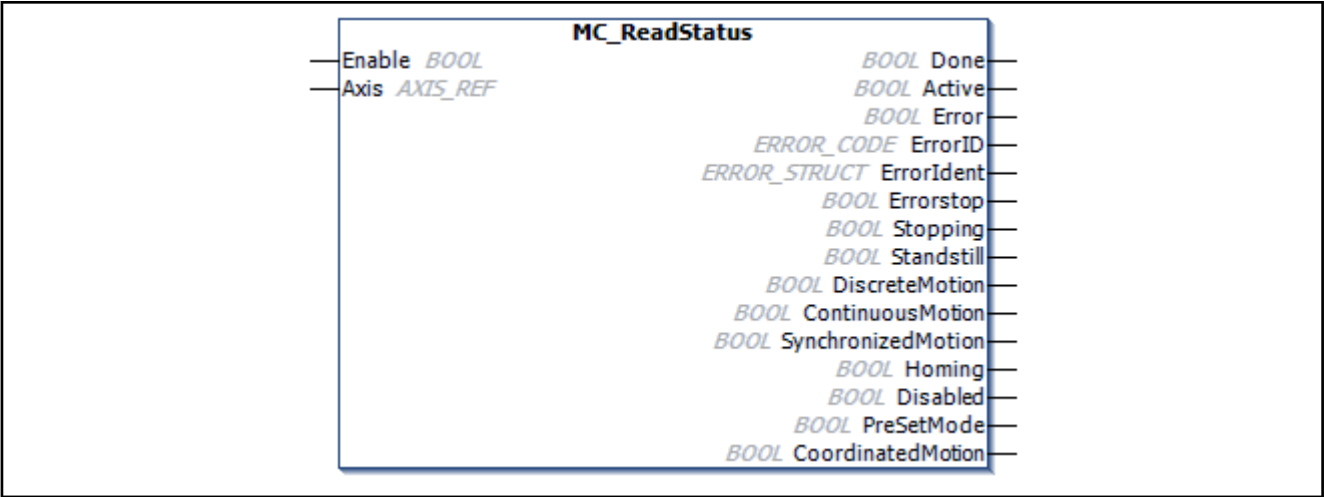


Fig. 13-36: "MC_ReadStatus" firmware function block

Input variable	Data type	Description
Enable	BOOL	See "Signal Time Diagrams"

Tab. 13-82: Input variable of the "MC_ReadStatus" firmware function block

Output variable	Data type	Description
Done	BOOL	A valid value is available
Active	BOOL	Processing of the data is running after completion of preprocessing
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Errorstop	BOOL	Drive/axis in status "Errorstop"
Stopping	BOOL	Drive/axis in status "Stopping"
StandStill	BOOL	Drive/axis in status "StandStill"
DiscreteMotion	BOOL	Drive/axis in status "DiscreteMotion"
ContinuousMotion	BOOL	Drive/axis in status "ContinuousMotion"
SynchronizedMotion	BOOL	Drive/axis in status "SynchronizedMotion"
Homing	BOOL	Drive in status "Homing"
Disabled	BOOL	Drive/axis in status "Disabled"
PreSetMode	BOOL	Drive in status "PreSetMode"
CoordinatedMotion	BOOL	Drive/axis in status "CoordinatedMotion"; not supported by the "IndraDrive" target!

* See note below

Tab. 13-83: Output variables of the "MC_ReadStatus" firmware function block

 NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-84: Input/output variable of the "MC_ReadStatus" firmware function block

13.3.25 MC_Reset

The function block "MC_Reset" is used to reset drive errors with the command "S-0-0099, C0500 Reset class 1 diagnostics"; drive enable remains unchanged.

In order that the function block can correctly complete the reset command, it has to be cyclically called until it signals "Done"=TRUE or "Error" in the case of error.



If the command "S-0-0099, C0500 Reset class 1 diagnostics" is already active, it should first be completed before "MC_Reset" is called. If "MC_Reset" had been activated and has not yet been completed, it is possible to trigger another reset with a rising edge at "Execute".



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done" at a later call. This means the call times for this function block are always brief. The outputs "Done" and "Error" have to be cyclically polled. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase 2. Any attempt to do so results in an error.

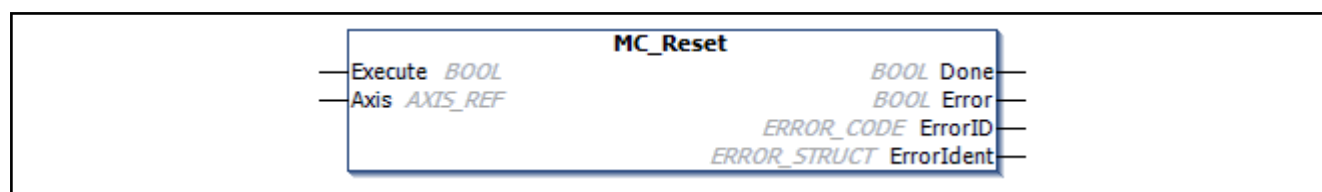


Fig. 13-37: "MC_Reset" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"

Tab. 13-85: Input variables of the "MC_Reset" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-86: Output variables of the "MC_Reset" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-87: Input/output variable of the "MC_Reset" firmware function block

13.3.26 MC_Stop

The firmware function block "MC_Stop" is used to bring the drive to the status STOP.

With the rising edge of "Execute" the stopping process is started.

If the PLC does not have control over the drive, an error is returned.

When the firmware function block "MC_Stop" is activated via a rising edge at the "Execute" input, the drive goes to the status "AH" (Drive Halt). Further command triggering of a motion function block will only be possible when the axis is in standstill and the "Execute" input has the value FALSE.



The "Drive Halt" function is described in the functional description of the firmware.

With the deceleration set at "Deceleration" the drive is brought to standstill.

When several instances are called by "MC_Stop" in a project, it is always the last instance which takes effect. An active "MC_Stop" cannot be interrupted by an "MB_Stop".

When an instance is called with an edge at "Execute", other instances when called (without edge) signal CommandAborted = TRUE.

"MC_Stop" has higher priority than "MB_Stop"; this means that a running "MC_Stop" cannot be aborted by an "MB_Stop".



To test the function block "MC_Stop", the instance variable "ActivateCounter" can be used. It counts the positive edges of the "Execute" signal and thereby signals whether or not the function block is run with an edge.



Via the firmware function block "MC_Stop" it is also possible to bring "P-0-0759, Virtual master axis, actual velocity value" to standstill via a defined ramp (defined via "Deceleration" input). For this purpose, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0=1. To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"),

"P-0-0758, Virtual master axis, actual position value" must be written to the parameter "P-0-0916, Master axis format converter signal selection".

See also Functional Description of firmware "Master axis generator" and "Master axis format converter"

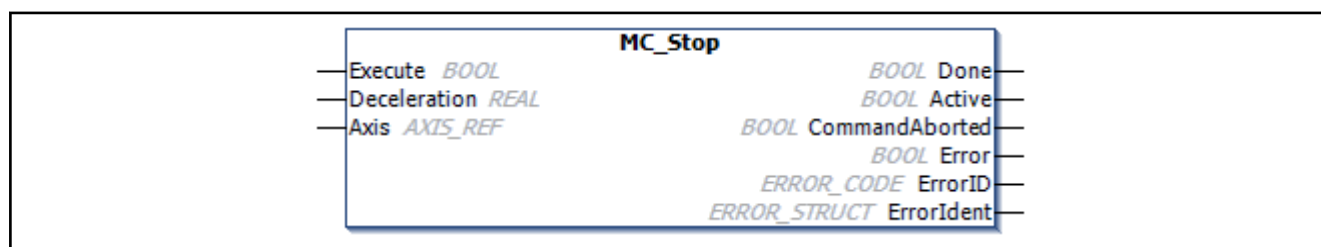


Fig. 13-38: "MC_Stop" firmware function block

Input variable	Data type	Description
Execute	BOOL	Slave axis ("Axis1" to "Axis8"): With a rising edge at "Execute" the drive is switched to "AH". Master axis generator ("VmAxis1"): With a rising edge at "Execute" the master axis generator is brought to the positioning mode "halt" ("P-0-0760, Virtual master axis, positioning control word", bit1 and bit2 to "1"). See "Signal time diagrams"
Deceleration	REAL	This input sets the deceleration. The deceleration which is set is written to parameter "S-0-0372, Drive Halt acceleration bipolar" of the slave axis (real axis) or parameter "P-0-0772, Virtual master axis, positioning deceleration" of the virtual master axis generator.

Tab. 13-88: Input variables of the "MC_Stop" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the axis has stopped. Slave axis (real axis): When S-0-0331, Bit0=1 Virtual master axis generator ("VmAxis1"): When the absolute value of P-0-0759 is smaller than P-0-0912
Active	BOOL	With "Active"=TRUE the firmware function block signals that it determines the axis motion
CommandAborted	BOOL	With "CommandAborted"=TRUE the function block signals that another "MC_Stop" was activated or a drive error was present. A drive error is displayed via the Error outputs of "MC_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

*

See note below

Tab. 13-89: Output variables of the "MC_Stop" firmware function block

 NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input / output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-90: Input/output variable of the "MC_Stop" firmware function block

13.3.27 MC_TorqueControl

The firmware function block "MC_TorqueControl" is used to activate the operation mode "torque/force control"; the drive follows the command value input.

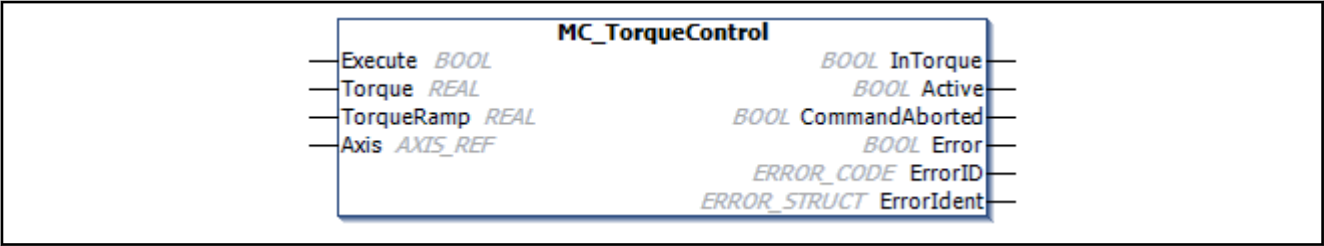


Fig. 13-39: "MC_TorqueControl" firmware function block

Input variable	Data type	Description	Parameters used	
Execute	BOOL	See "Signal Time Diagrams"	-	
			①	②
Torque	REAL	Torque/force command value* ¹⁾	S-0-0080	P-0-1465
TorqueRamp	REAL	Torque/force ramp in relation to the torque/force ramp time* ¹⁾ Note: If the "TorqueRamp" input is set to "0", the torque/force command value is input abruptly.	S-0-0822 / S-0-0823	P-0-1466 / P-0-1467

- ① If PLC has permanent control over the axis or if a removed axis is commanded
- ② If PLC has temporary control over the axis
- *¹⁾ The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-91: Input variables of the "MC_TorqueControl" firmware function block

Output variable	Data type	Description
InTorque	BOOL	Torque/force command value has been reached. Note: With the parameter "S-0-0826, Torque/force window", it is specified when the function block returns the message "InTorque".
Active	BOOL	Function block or operation mode is active
CommandAborted	BOOL	Function block was aborted
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-92: Output variables of the "MC_TorqueControl" firmware function block



NOTE: For the list of all possible errors, see: "Error Reference Lists"

Input / output variable (VAR_IN_OUT)	Data type	Description
Slave	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-93: Input/output variables of the "MC_TorqueControl" firmware function block

MX_PLCOpen

13.3.28 MX_MoveAbsolute

The firmware function block "MX_MoveAbsolute" is used to move the drive to a preset absolute position. For this purpose, the secondary PLC operation mode "drive-controlled positioning" is internally activated (see Functional Description of firmware "Drive-Controlled Positioning"). Make the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

For permanent control and remote axes, the secondary operation mode 3 and its parameters are used. For temporary control, the secondary PLC operation mode 11 and its parameters are used.

The target position must be set at the "Position" input. Via the inputs "Velocity", "Acceleration" and "Deceleration" you have to set how the drive moves to the target position (motion profile).

Commanding of a virtual master axis

If a virtual master axis generator is to be positioned in absolute form via the "MX_MoveAbsolute" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").



The following parameters must not be written cyclically as they are written by this function block or also other functions blocks:

- S-0-0346, Positioning control word
- P-0-1454, PLC/setting-up mode, positioning command value acceptance
- P-0-0760, Virtual master axis, positioning control word

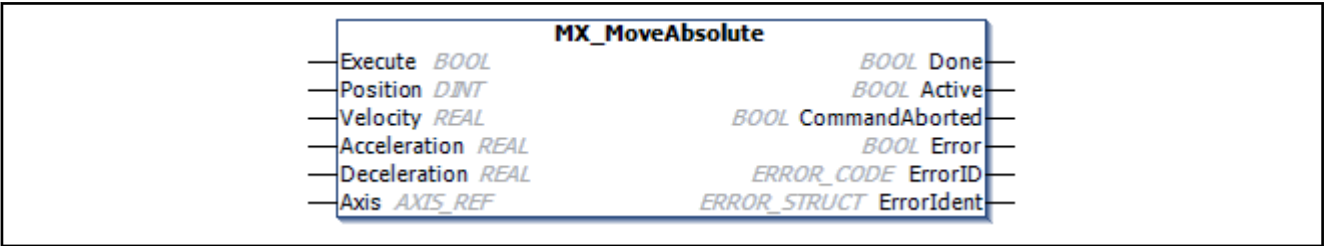


Fig. 13-40: "MX_MoveAbsolute" firmware function block

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Position	DINT	At "Position" you have to preset the target position in fixed point format. To preset the position correctly, the number of decimal places of the parameter has to be known. With preferred scaling (see Functional Description of firmware "Scaling Types"), the parameter has 4 decimal places. Example: If the drive is to be moved to a target position of 100.25°, the value 1002500 has to be written at "Position".	S-0-0282	P-0-1450	P-0-0766
Velocity	REAL	Positioning velocity in physical unit* ¹⁾	S-0-0259	P-0-1451	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit* ¹⁾	S-0-0260	P-0-1452	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit* ¹⁾	S-0-0359	P-0-1453	P-0-0772

- ① If PLC has permanent control over the axis or if a removed axis is commanded
- ② If PLC has temporary control over the axis
- ③ In case of a virtual master axis
- *1) The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-94: Input variables of the "MX_MoveAbsolute" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the absolute target position has been reached. The processing is completed until the next 0-1 edge at the "Execute" input. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has been completed ("Done"=TRUE), the "Done" output is set to TRUE for one call when the motion target has been reached and the processing of the firmware function block has not been interrupted by an error or another firmware function block. Real axis: The output is set to TRUE when bit 2 is set in parameter "S-0-0437, Positioning status". Virtual master axis generator (VmAxis1): The output is set to TRUE when bit 2 is set in parameter "P-0-0768, Virtual master axis, positioning status".
Active	BOOL	The function block signals with "Active"=TRUE that it determines the axis motion. "Active" remains TRUE until "CommandAborted" or "Error" is set.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power"
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-95: Output variables of the "MX_MoveAbsolute" firmware function block

NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-96: Input/output variable of the "MX_MoveAbsolute" firmware function block

13.3.29 MX_MoveAdditive

The firmware function block "MX_MoveAdditive" is used to move the drive in relative form by a distance in addition to the target position. The new target position results from the old target position, plus a distance. The distance is

MX_PLCOpen

determined by the "Distance" input. The inputs "Velocity", "Acceleration" and "Deceleration" determine the motion profile.

For this purpose, the secondary PLC operation mode "drive-controlled positioning" is internally activated (see Functional Description of firmware "Drive-Controlled Positioning"). Make the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

For permanent control and remote axes, the secondary operation mode 3 and its parameters are used. For temporary control, the secondary PLC operation mode 11 and its parameters are used.

Commanding of a virtual master axis

If a virtual master axis generator is to be traveled in relative form by a distance **in addition** to the current target position via the "MX_MoveAdditive" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").

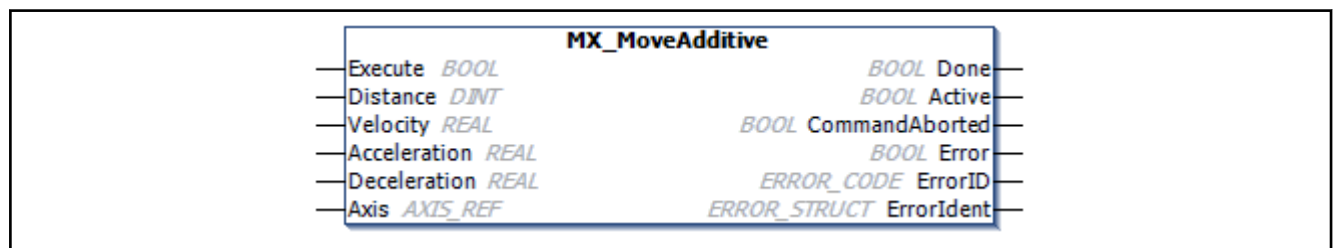


Fig. 13-41: "MX_MoveAdditive" firmware function block



The following parameters must not be written cyclically as they are written by this function block or also other functions blocks:

- S-0-0346, Positioning control word
- P-0-1454, PLC/setting-up mode, positioning command value acceptance
- P-0-0760, Virtual master axis, positioning control word

MX_PLCopen

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Distance	DINT	At "Distance" you have to preset the relative distance in fixed point format. To preset the distance correctly, the number of decimal places of the parameter has to be known. With preferred scaling (see Functional Description of firmware "Scaling Types"), the parameter has 4 decimal places. Example: If the drive is to be moved by a distance of 100.25°, the value 1002500 must be written at "Distance"	S-0-0282	P-0-1450	P-0-0766
Velocity	REAL	Positioning velocity in physical unit* ¹⁾	S-0-0259	P-0-1451	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit* ¹⁾	S-0-0260	P-0-1452	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit* ¹⁾	S-0-0359	P-0-1453	P-0-0772

- ① If PLC has permanent control over the axis or if a removed axis is commanded
- ② If PLC has temporary control over the axis
- ③ In case of a virtual master axis
- *¹⁾ The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-97: Input variables of the "MX_MoveAdditive" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the target position has been reached. The processing is completed until the next 0-1- edge at the "Execute" input. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has been completed ("Done"=TRUE), the "Done" output is set to TRUE for one call when the motion target has been reached and the processing of the firmware function block has not been interrupted by an error or another firmware function block. Real axes: The output is set to TRUE when bit 2 is set in parameter "S-0-0437, Positioning status". Virtual master axis: The output is set to TRUE when bit 2 is set in parameter "P-0-0768, Virtual master axis, positioning status".
Active	BOOL	With "Active"=TRUE the firmware function block signals that it determines the axis motion.
CommandAborted	BOOL	With "CommandAborted"=TRUE the firmware function block signals that another motion function block was activated or a drive error was present. A drive error is displayed via the Error outputs of "MC_Power".
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-98: Output variables of the "MX_MoveAdditive" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-99: Input/output variable of the "MX_MoveAdditive" firmware function block

13.3.30 MX_MoveRelative

The firmware function block "MX_MoveRelative" is used to move the drive in relative form by a distance, starting from the current actual position. For this purpose, the secondary PLC operation mode "drive-controlled positioning" is

MX_PLCOpen

internally activated (see Functional Description of firmware "Drive-Controlled Positioning").



Internally the operation mode "drive-controlled positioning" is activated. Select the settings of the position controller (with/without lag error, actual position value 1/2) with the function block "MB_SetPositionControlMode".

For permanent control and remote axes, the secondary operation mode 3 and its parameters are used. For temporary control, the secondary PLC operation mode 11 and its parameters are used.

The relative distance to be moved has to be preset at the "Distance" input. Via the inputs "Velocity", "Acceleration" and "Deceleration" you have to set the motion profile (determines how the relative distance is traveled).

Commanding of a virtual master axis

If a virtual master axis generator is to be traveled in relative form by a distance from the actual virtual position via the "MX_MoveRelative" firmware function block, the master axis generator has to be activated via "P-0-0917, Control word of master axis generator", bit0="1". To convert the virtual actual position value (P-0-0758) into the internal virtual master axis position (VmAxis1, "P-0-0761, Master axis position for slave axis"), the parameter "P-0-0758, Virtual master axis, actual position value" has to be written to parameter "P-0-0916, Master axis format converter signal selection" (also see firmware function description "Master axis generator" and "Master axis format converter").



The following parameters must not be written cyclically as they are written by this function block or also other functions blocks:

- S-0-0346, Positioning control word
- P-0-1454, PLC/setting-up mode, positioning command value acceptance
- P-0-0760, Virtual master axis, positioning control word

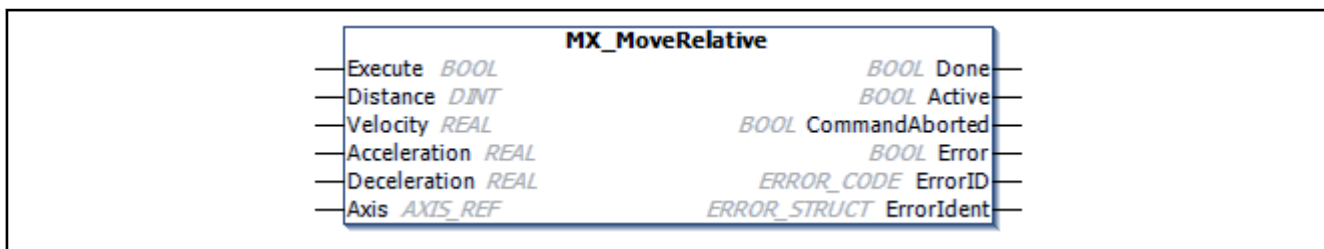


Fig. 13-42: "MX_MoveRelative" firmware function block

Input variable	Data type	Description	Parameters used		
Execute	BOOL	See "Signal Time Diagrams"	-		
			①	②	③
Distance	DINT	At "Distance" you have to preset the relative distance to be traveled in fixed point format. To preset the distance correctly, the number of decimal places of the parameter has to be known. With preferred scaling (see Functional Description of firmware "Scaling Types"), the parameter has 4 decimal places. Example: If the drive is to be moved by a distance of 100.25°, the value 1002500 must be written at "Distance"	S-0-0282	P-0-1450	P-0-0766
Velocity	REAL	Positioning velocity in physical unit* ¹⁾	S-0-0259	P-0-1451	P-0-0770
Acceleration	REAL	Positioning acceleration in physical unit* ¹⁾	S-0-0260	P-0-1452	P-0-0771
Deceleration	REAL	Positioning deceleration in physical unit* ¹⁾	S-0-0359	P-0-1453	P-0-0772

- ① If PLC has permanent control over the axis or if a removed axis is commanded
- ② If PLC has temporary control over the axis
- ③ In case of a virtual master axis
- *¹⁾ The number of possible decimal places depends on the parameterized scaling (see drive documentation). If the applied value has more decimal places, they are cut off

Tab. 13-100: Input variables of the "MX_MoveRelative" firmware function block

Output variable	Data type	Description
Done	BOOL	With "Done"=TRUE the firmware function block signals that the preset distance has been traveled. The processing is completed until the next 0-1-edge at the "Execute" input. When a 1-0- edge is applied to the "Execute" input before the firmware function block signals that it has been completed ("Done"=TRUE), the "Done" output is set to TRUE for one call when the motion target has been reached and the processing of the firmware function block has not been interrupted by an error or another firmware function block. - Real axes: The output is set to TRUE when bit 2 is set in parameter "S-0-0437, Positioning status". - Virtual master axis: The output is set to TRUE when bit 2 is set in parameter "P-0-0768, Virtual master axis, positioning status".
Active	BOOL	With TRUE the "Active" output signals that it determines the axis motion.
CommandAborted	BOOL	The function block signals with "CommandAborted"=TRUE that another motion function block was activated or a drive error had occurred. A drive error is displayed via the Error outputs of "MC_Power"
Error *	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-101: Output variables of the "MX_MoveRelative" firmware function block

MX_PLCOpen



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: - Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) "VmAxis1" corresponds to the virtual master axis generator

Tab. 13-102: Input/output variable of the "MX_MoveRelative" firmware function block

13.3.31 MX_SetControl

The function block "MX_SetControl" is used to switch control between master communication and PLC; this makes sense when the axis is first controlled via a master communication and the internal PLC is to continue controlling the axis in a certain situation (e.g. error).



When the axis generally is to be moved with the internal PLC, permanent control should be activated.

With "PLCControl"=TRUE the PLC gets control over the drive, with "PLCControl"=FALSE the external master communication gets control over the drive.

The signals "Drive Halt" (AH), "Drive on" (ON) and the operation mode selection are switched. With PLC control, the signals "AH" and "ON" are only operated internally. The "Drive Halt" signal is controlled by the function block "MB_Stop" or by stopping the PLC. The "ON" signal can be controlled with the function block "MX_Power".

If the corresponding function blocks are not used, the internal signals are set to TRUE.

When switching to temporary control, the internal "ON" signal is set to TRUE. This results in the following behavior when temporary control was gotten with "MX_SetControl":

- Switching off:** With the firmware function block "MC_Power".
In order that the firmware function block can detect a TRUE/FALSE transition at its Enable input, it must be called at least once with Enable=TRUE.
- Switching on:** With the firmware function block "MC_Power".
In order that the firmware function block can detect a FALSE/TRUE transition at its Enable input, it must be called at least once with "Enable"=FALSE. This measure is not necessary when switching from the parameter mode to the operating mode takes place between the call of "MX_SetControl" and "MC_Power".



The function block "MX_SetControl" supplies error if permanent control is already active (setting via "P-0-1367, PLC configuration").

While switching is running ("Done" and "Error" not yet TRUE), another switching process can be started with an "Execute" edge.

In the case of program download (no online change) or PLC reset, temporary control is automatically returned again.



To be observed with PLCOpen motion function blocks for the target "IndraDrive MP04" and above:

- With **temporary** control, the secondary PLC operation modes are used.
- With **permanent** control, the primary operation mode and the secondary operation modes 1-6 are parameterized and used.

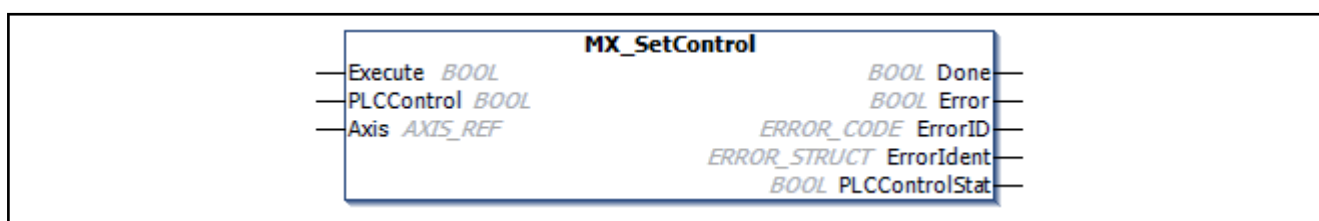


Fig. 13-43: "MX_SetControl" firmware function block

Input variable	Data type	Description
Execute	BOOL	This function block is activated with a rising edge at "Execute". All outputs, except for the "PLCControlStat" output, are reset with a falling edge at "Execute". Independent of "Execute", "PLCControlStat" provides the actual value.
PLCControl	BOOL	When "PLCControl"=TRUE, the internal PLC gets temporary control over the drive upon the next 0-1 edge at "Execute". When "PLCControl"=FALSE, control is returned to the master communication upon the next 0-1 edge at "Execute".

Tab. 13-103: Input variables of the "MX_SetControl" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
PLCControlStat	BOOL	With "PLCControlStat"=TRUE the drive-internal PLC has control over the drive. With "PLCControlStat"=FALSE the master communication has control over the drive. Independent of "Execute", "PLCControlStat" provides the actual value.

* See note below

Tab. 13-104: Output variables of the "MX_SetControl" firmware function block

MX_PLCOpen



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-105: Input/output variable of the "MX_SetControl" firmware function block

13.3.32 MX_SetDeviceMode

The "MX_SetDeviceMode" function block is used to switch the drive to the parameter or operating mode.

If switching is not allowed ("P-0-4086, Master communication status", bit 0="0"), the function block returns an error via "Error".

A mode is set by a rising edge of "Execute". Independent of "Execute", the current status is displayed in the "ActOperationMode" output.



During the switching process, "MX_SetDeviceMode" should be called until the "Done" or "Error" output becomes TRUE. Otherwise it may not be possible to directly switch again, since "MX_SetDeviceMode" may not be in its initial state.

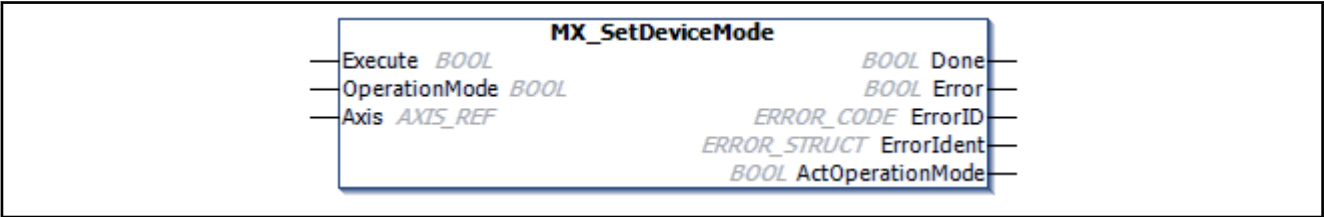


Fig. 13-44: "MX_SetDeviceMode" firmware function block

Input variable	Data type	Description
Execute	BOOL	This function block is activated with a rising edge at "Execute". All outputs, except for the "ActOperationMode" output, are reset with a falling edge at "Execute". Independent of "Execute", the current communication phase is displayed at "ActOperationMode".
OperationMode	BOOL	When "OperationMode"=TRUE, the drive is switched to operating mode (phase 4) upon the next 0-1 edge at "Execute". For this purpose, the function block starts the drive command C0200 (S-0-0422). When "OperationMode"=FALSE, the drive is switched to parameter mode (phase 2) upon the next 0-1 edge at "Execute". For this purpose, the function block starts the drive command C0400 (S-0-0420).

Tab. 13-106: Input variables of the "MX_SetDeviceMode" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
ActOperationMode	BOOL	When "ActOperationMode"=TRUE, the drive is in operating mode (phase 4). When "ActOperationMode"=FALSE, the drive is in parameter mode (2<=current phase 4). Independent of "Execute", the current communication phase is displayed at "ActOperationMode".

* See note below

Tab. 13-107: Output variables of the "MX_SetDeviceMode" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-108: Input/output variable of the "MX_SetDeviceMode" firmware function block

13.3.33 MX_SetOpMode

The function block "MX_SetOpMode" is used to directly switch the operation mode of the drive.

The current operation mode is provided independent of "Execute". The primary operation mode corresponds to the value "0" and the secondary operation modes correspond to the values "1" to "3" for firmware MPH02VRS and "1" to "7" for MPx03VRS firmwares.



"MX_SetOpMode" is provided for direct control of the drive via its operation modes and their parameters. The operation mode that was set is activated in the following position controller cycle.

If after the switching process the drive is moved with "MX_MoveRelative", for example, an operation mode >3 for MPH02VRS and >7 as of MPx03VRS is implicitly set.

During the switching process, "MX_SetOpMode" should be called until the "Done" output or the "Error" output becomes TRUE. Otherwise direct switching might not be possible again, as "MX_SetOpMode" then possibly is not in its initial status.

The switching process cannot be activated by several instances at the same time!

If the PLCopen profile is used for motion control, "MX_SetOpMode" cannot be used. The secondary PLC operation modes are not allowed as "NewOpMode" and internally provided for the PLCopen profile of the PLC.

MX_PLCOpen



To be observed for the target "IndraDrive MP04" and above:

- With this target and the targets above, the first 6 operation modes, in the case of permanent control, have a fixed parameter setting and cannot be changed.
- The 7th secondary operation mode can be freely parameterized and selected in operation with the "MX_SetOpMode" function block.
- When the function block "MB_PreSetMode" is used, the drive, after it has been switched on, immediately switches to this secondary operation mode without shortly remaining in "Drive Halt" before.
- Active motion function blocks signal "CommandAborted" when a new operation mode is set with "MX_SetOpMode".
- At the command "MX_SetOpMode", the axis changes from "AH" to "AF" or "AU".
- **Attention:** In the multi-axis system MLD-M, the command value parameters of the operation modes used by the PLCopen function blocks have already been configured in the multiplex channel (CCD process data) for remote axes. These parameters cannot be written any more!



The operation mode cannot be switched by multiple instances at the same time; this is prevented by the system.

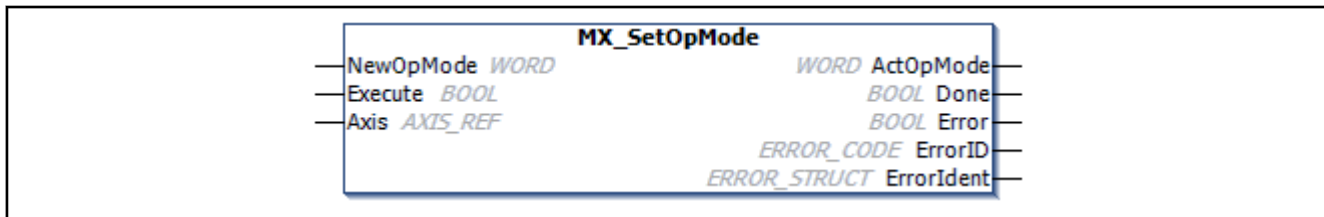


Fig. 13-45: "MX_SetOpMode" firmware function block

Input variable	Data type	Description
NewOpMode	WORD	The operation mode set at "NewOpMode" is activated by a 0-1 edge at "Execute". "NewOpMode" can presently have the following values: 0: S-0-0032, Primary operation mode 1: S-0-0033, Secondary operation mode 1 2: S-0-0034, Secondary operation mode 2 3: S-0-0035, Secondary operation mode 3 As of target "IndraDrive MP03": 4: S-0-0284, Secondary operation mode 4 5: S-0-0285, Secondary operation mode 5 6: S-0-0286, Secondary operation mode 6 7: S-0-0287, Secondary operation mode 7
Execute	BOOL	This function block is activated with a rising edge at "Execute". All outputs, except for the "ActOpMode" output, are reset with a falling edge at "Execute". Independent of "Execute", the current operation mode is displayed at "ActOpMode".

Tab. 13-109: Input variables of the "MX_SetOpMode" firmware function block

Output variable	Data type	Description
ActOpMode	WORD	"ActOpMode" signals which operation mode is active in the drive at drive enable. The output adopts the value set at "NewOpMode".
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-110: Output variables of the "MX_SetOpMode" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MLD-S: Axis1 MLD-M: - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves/remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-111: Input/output variable of the "MX_SetOpMode" firmware function block

MX_PLCopen

13.4 "Parameters" - Function blocks/functions for parameters

Runtime for write/read access In the case of write/read access with function blocks/functions, the values of some parameters are also transmitted to the processor for the integrated safety technology. Therefore, write/read access to the parameters mentioned below takes comparatively long (approx. 8 ms). To avoid problems with the task cycle time, it is recommended to run write/read access to the parameters mentioned below in a task with a cycle time of at least 8 ms (maybe without watchdog):

- Safety technology parameters (except for parameters with the "cyclic="AT"" parameter attribute): P-0-32xx, P-0-33xx, S-0-1800.x.x bis S-0-1830.x.x
- P-0-0117
- P-0-0119

Write protection by password If parameters are write-protected by a password, it is impossible to write parameter values using functions or function blocks (e.g., MX_WriteParamDINT).

13.4.1 MB_Command

Brief description The "MB_Command" function block is used to execute, monitor and abort commands.

Interface description

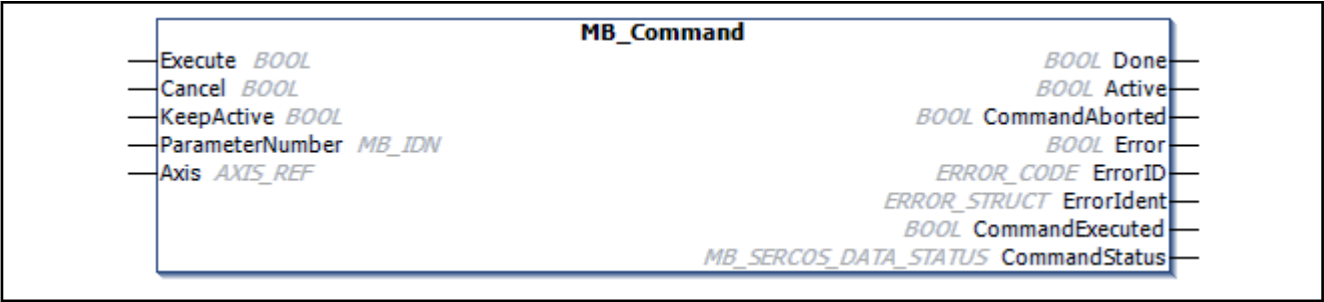


Fig. 13-46: "MB_Command" function block

Interface variables

	Name	Type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis
VAR_INPUT	Execute	BOOL	Enabling the function block (once, edge-controlled)
	Cancel	BOOL	Command execution aborted (one-time, edge-controlled)
	KeepActive	BOOL	Keep command active after execution (permanent, state-controlled)
	ParameterNumber	MB_IDN	Command

	Name	Type	Comment
VAR_OUTPUT	Done	BOOL	Command successfully completed and reset
	Active	BOOL	Command execution active
	CommandAborted	BOOL	Command successfully aborted and reset
	Error	BOOL	With "Error"=TRUE the function block signals that an error is present
	ErrorID	ERROR_CODE	"ErrorID" gives a rough error description.
	ErrorIdent	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
	CommandExecuted	BOOL	Command successfully completed
	CommandStatus	MB_SERCOS_DATA_STATUSES	Status of command execution

Tab. 13-112: Interface variables of MB_Command function block

Signal-time diagram

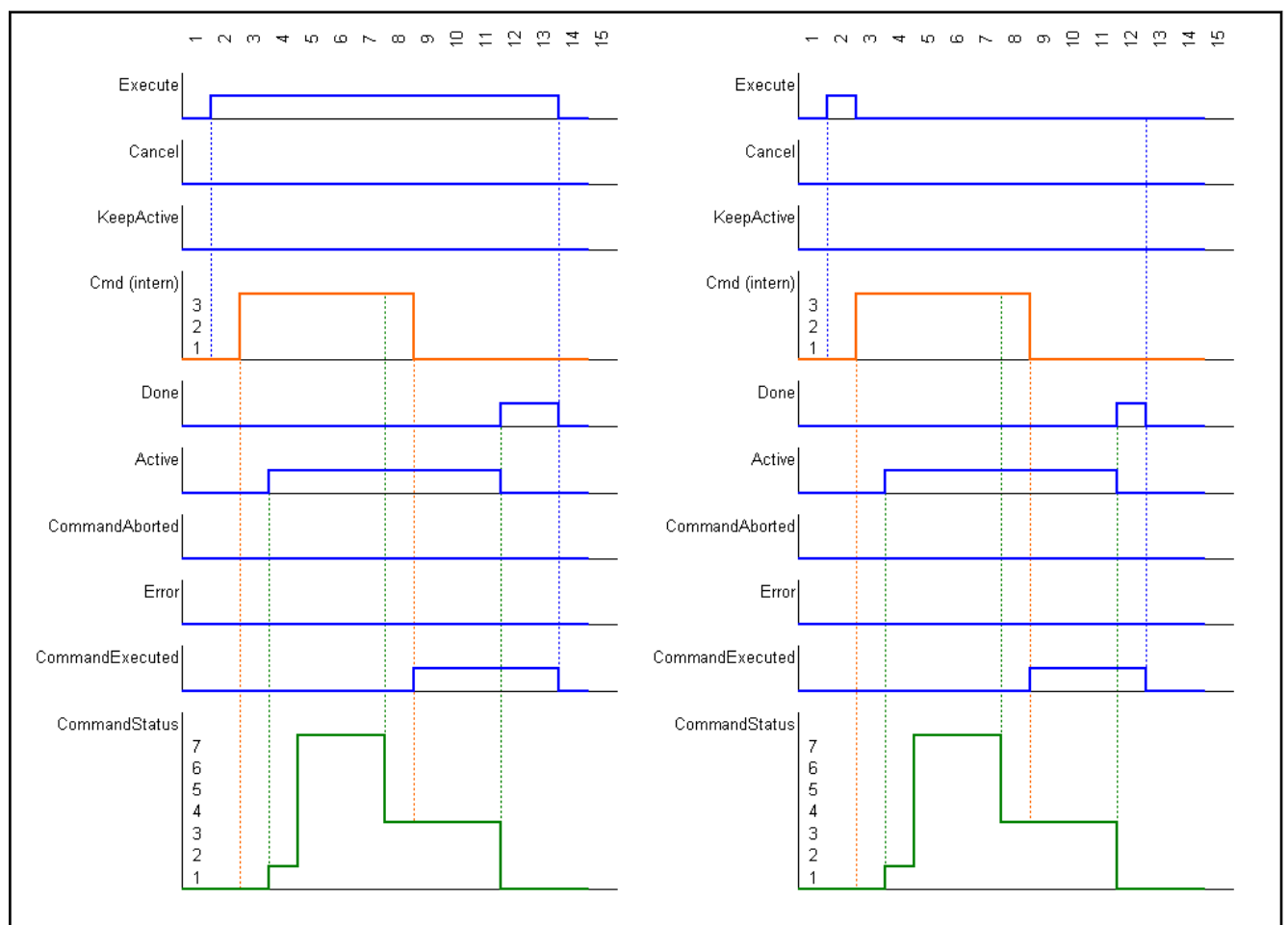


Fig. 13-47: Signal-time diagram for "Execute-Done" signal sequence

MX_PLCOpen

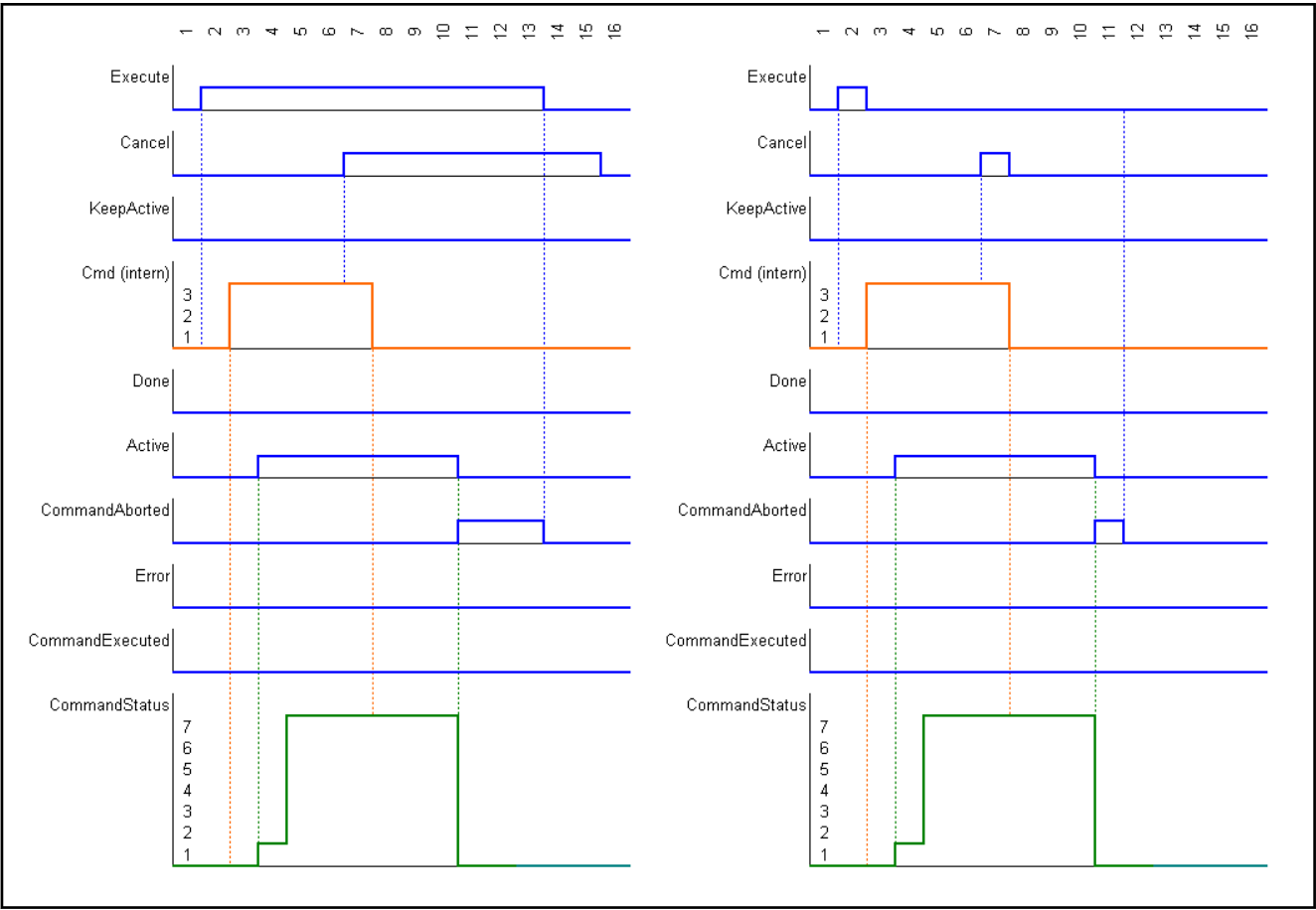


Fig. 13-48: Signal-time diagram for "Execute-Cancel-CommandAborted" signal sequence

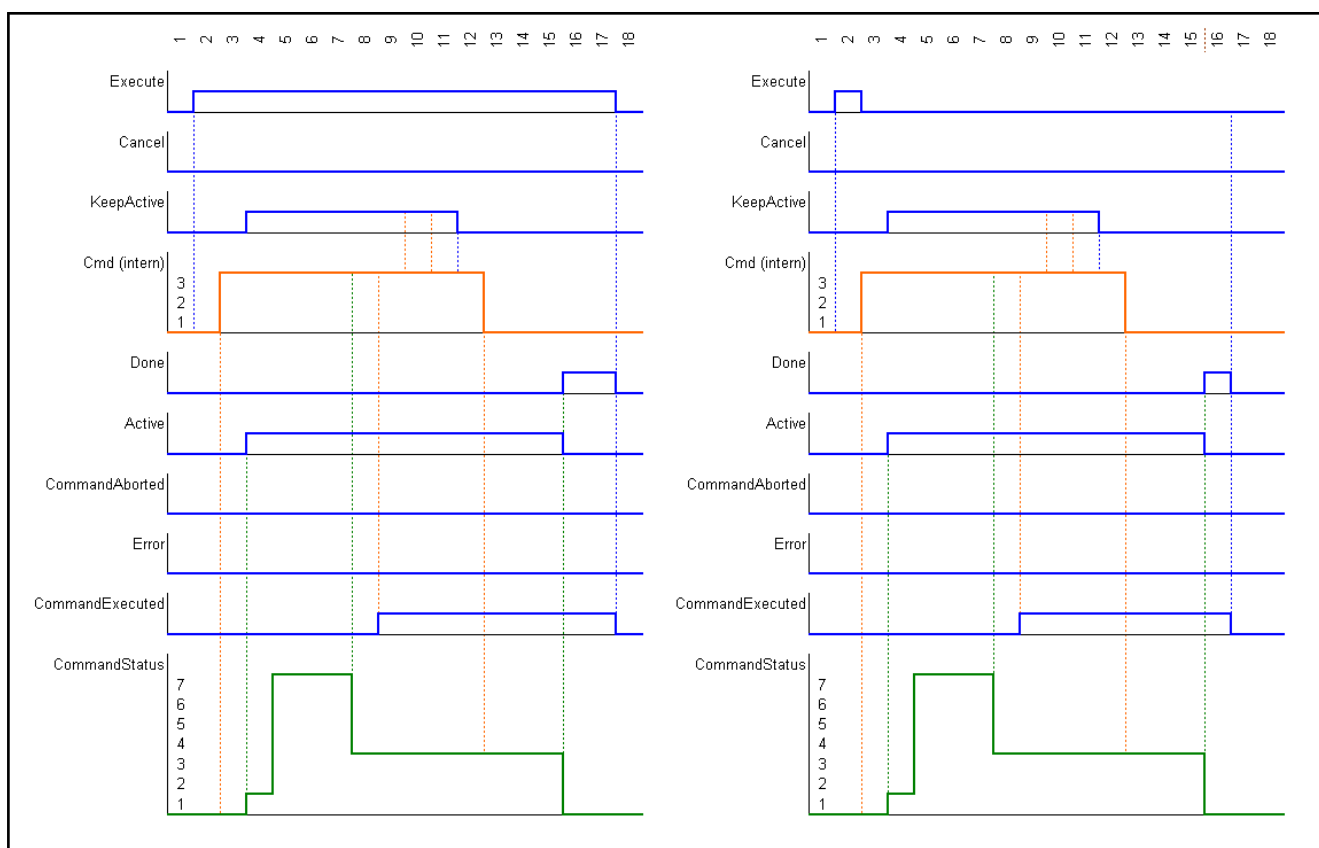


Fig. 13-49: Signal-time diagram for "Execute-KeepActive-Done" signal sequence

MX_PLCOpen

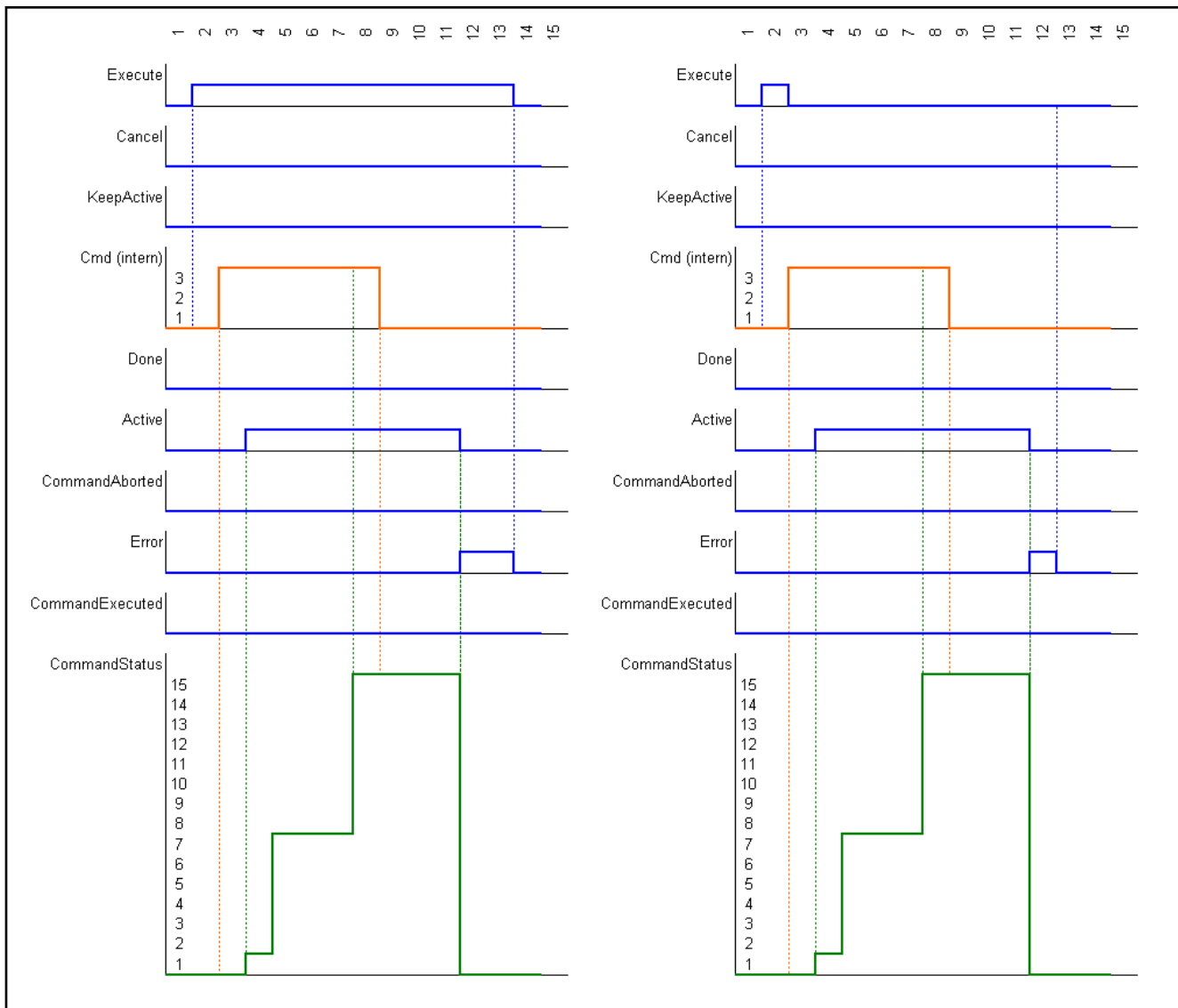


Fig. 13-50: Signal-time diagram for "Execute-Error" signal sequence

Functional description

Once the "MB_Command" function block is enabled, it starts by means of "Execute" to execute the command defined via the "ParameterNumber" input. Generally, the command is automatically reset once processing is complete. The command has to be specified in the parameter IDN format in accordance with Sercos standard. In addition to this direct specification, it is possible to use a suitably predefined constant from the global variables of the basic library (e.g., "ML_Base") or a conversion function from the "RIL_Utility" library. Command execution can be aborted at any time via the "Cancel" input, depending on the command itself. The "KeepActive" input allows the command to be kept active after its execution. If the "KeepActive" input is then deactivated, the command is also automatically reset. The "Done" or "CommandAborted" outputs signal that a command is successfully completed or aborted for the duration of at least one call. The "Active" output and the corresponding data at the "CommandStatus" output show that the command is being actively processed. The current status of command execution is permanently signaled at the "CommandStatus" output; the "CommandExecuted" output acknowledges that the command has been completed. Any error that occurs while the command is being processed is

signaled via the "Error" output and the error classification elements (ErrorID, ErrorIdent) for the duration of at least one call.

Troubleshooting

The "MB_Command" function block uses the error tables **F_RELATED_TABLE**, 16#0170 and **INDRV_TABLE**, 16#0070. It can generate the errors contained in the table below in Additional1 and Additional2. In addition to this error, the included function blocks "MB_WriteParameter", "MB_ReadSercosDataStatus", "MB_ReadParameter" and "MB_ReadSercosAttribute" can signal other errors.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR, 16#0006	16#0002	16#0001	Invalid input - Function block "MB_Command" does not support retriggering command execution.
INPUT_RANGE_ERROR, 16#0006	16#0002	16#0002	Invalid input - Control commands do not support inputs "Cancel" or "KeepActive".
INPUT_RANGE_ERROR, 16#0006	16#0002	16#0003	Invalid input - Input "ParameterNumber" does not contain a valid command.
DEVICE_ERROR, 16#0008	16#1701	16#0000	Error during command execution (control)
STATE_MACHINE_ERROR, 16#0005	16#0006	16#????	Invalid state of state machine 16#???? indicates the invalid state

Tab. 13-113: Error numbers in the error table "F_RELATED_TABLE" caused by "MB_Command"

ErrorID	Additional1	Additional2	Description
DEVICE_ERROR, 16#0008	16#?????	16#0000	Error during command execution (drive) - 16#???? contains the current value of S-0-0390.

Tab. 13-114: Error numbers in the error table "INDRV_TABLE" caused by "MB_Command"

13.4.2 MB_ReadListParameter

The "MB_ReadListParameter" function block is used to read the current value of a list parameter.

The IDN of the list parameter to be read has to be applied at the "ParameterNumber" input. The number of bytes to be read has to be indicated at the "NoOfBytes" input. The initial address of the field to which the bytes that have been read are to be written has to be indicated at the "ValueAdr" input. The processing of the function block is started with a rising edge at the "Execute" ("Execute"=TRUE) input.



The "ActivateCounter" instance variable can be used to test the "MB_ReadListParameter" function block. It counts the positive edges of the Execute signal and thereby signals whether or not the function block is run with an edge.

Local axis

When the function block is used for a local axis, all list elements of the selected parameter are immediately read when the function block is called. After the list elements have been read, control is returned to the program and TRUE is set at the "Done" output.



Observe that the read process can take some time, depending on the volume of data to be read. Use a separate task, if necessary.

Example: Reading all list elements of parameter P-0-0072 (1,024 long words) takes several 100 µs.

Remote axes



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

When the function block is used for remote axes, the read task is returned to the service channel (Sercos III). The function block can only signal "Done" at a later call.

The outputs "Done", "Active" and "Error" have to be polled cyclically.

The data transmission can take several seconds, depending on the volume.



Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

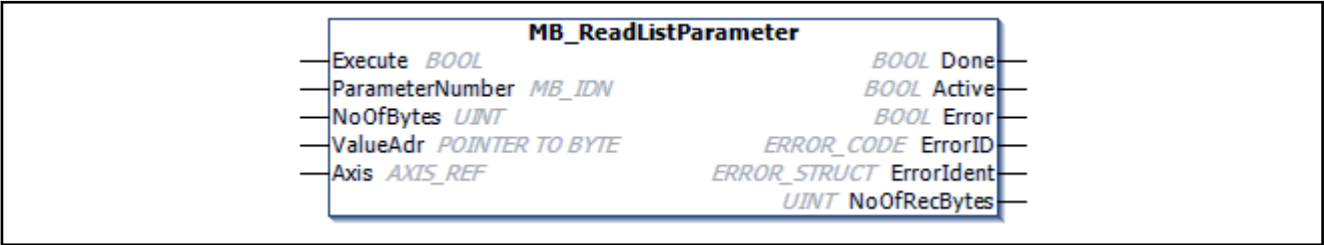


Fig. 13-51: "MB_ReadListParameter" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	IDN of the parameter to be read
NoOfBytes	UINT	The number of bytes to be read has to be indicated at the "NoOfBytes" input. Example: NoOfBytes:=SIZEOF(arFeld1)
ValueAdr	POINTER TO BYTE	The initial address of the field to which the bytes that have been read are to be written has to be indicated at the "ValueAdr" input. Example: ValueAdr := ADR(arFeld1) Note: The address has to match the memory alignment of the data that are accessed: • 1-byte lists: any address • 2-byte lists: any even address • 4-byte lists: any address that can be divided by 4

Tab. 13-115: Input variables of the "MB_ReadListParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the data value of the list parameter has been successfully read
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!

Output variable	Data type	Description
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
NoOfRecBytes	UINT	The "NoOfRecBytes" output signals how many bytes were actually read

* See note below
 Tab. 13-116: Output variables of the "MB_ReadListParameter" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-117: Input/output variable of the "MB_ReadListParameter" firmware function block

13.4.3 MB_ReadMaxRealValue

The "MB_ReadMaxRealValue" function block is used to read the maximum allowed value of a parameter. The value is returned in REAL format.

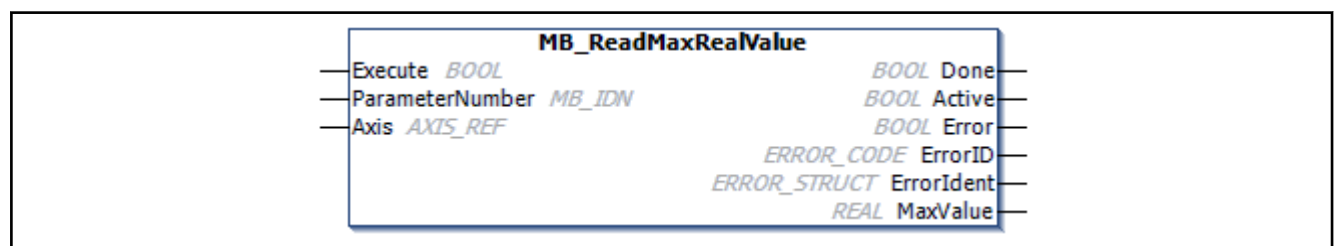


Fig. 13-52: "MB_ReadMaxRealValue" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter the maximum value of which is to be read

Tab. 13-118: Input variables of the "MB_ReadMaxRealValue" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
MaxValue	REAL	If "Done"=TRUE, "MaxValue" displays the maximum input value of the selected parameter. Note: According to the scaling, the input value of the parameter has decimal places that have to be taken into account and used for calculation

*

See note below

Tab. 13-119: Output variables of the "MB_ReadMaxRealValue" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-120: Input/output variable of the "MB_ReadMaxRealValue" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.4 MB_ReadMaxValue

The "MB_ReadMaxValue" function block is used to read the maximum allowed value of a parameter. The value is returned in DINT format.

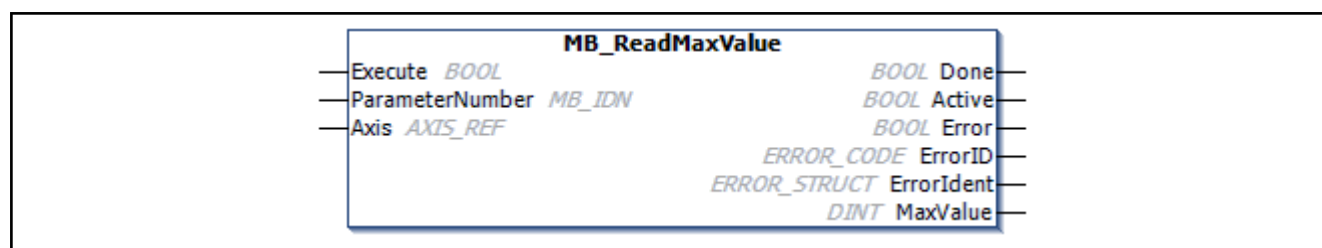


Fig. 13-53: "MB_ReadMaxValue" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter the maximum value of which is to be read

Tab. 13-121: Input variables of the "MB_ReadMaxValue" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
MaxValue	DINT	If "Done"=TRUE, "MaxValue" displays the maximum input value of the selected parameter. Note: According to the scaling, the input value of the parameter has decimal places that have to be taken into account and used for calculation

* See note below

Tab. 13-122: Output variables of the "MB_ReadMaxValue" firmware function block



NOTE: For the list of all possible errors, see: "Error Reference Lists"

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-123: Input/output variable of the "MB_ReadMaxValue" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.5 MB_ReadMinRealValue

The "MB_ReadMinRealValue" function block is used to read the minimum allowed value of a parameter. The value is returned in REAL format.



Fig. 13-54: "MB_ReadMinRealValue" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter the minimum value of which is to be read

Tab. 13-124: Input variables of the "MB_ReadMinRealValue" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
MinValue	REAL	If "Done"=TRUE, "MinValue" displays the minimum input value of the selected parameter. Note: According to the scaling, the input value of the parameter has decimal places that have to be taken into account and used for calculation

* See note below

Tab. 13-125: Output variables of the "MB_ReadMinRealValue" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-126: Input/output variable of the "MB_ReadMinRealValue" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

MX_PLCOpen

13.4.6 MB_ReadMinValue

The "MB_ReadMinValue" function block is used to read the minimum allowed value of a parameter. The value is returned in DINT format.

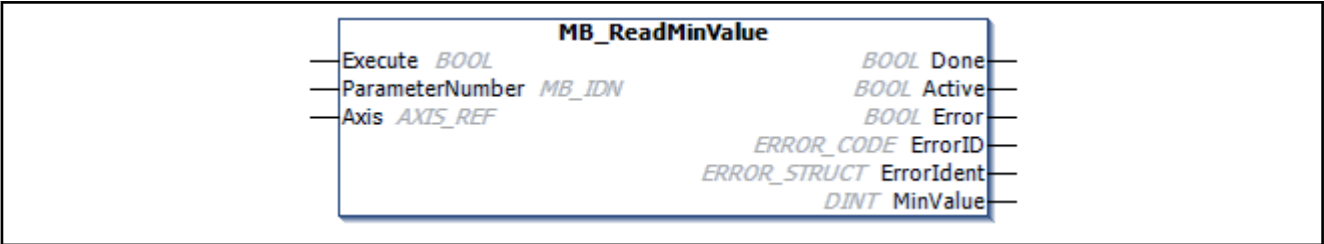


Fig. 13-55: "MB_ReadMinValue" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter the minimum value of which is to be read

Tab. 13-127: Input variables of the "MB_ReadMinValue" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
MinValue	DINT	If "Done"=TRUE, "MinValue" displays the minimum input value of the selected parameter. Note: According to the scaling, the input value of the parameter has decimal places that have to be taken into account and used for calculation

* See note below

Tab. 13-128: Output variables of the "MB_ReadMinValue" firmware function block



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-129: Input/output variable of the "MB_ReadMinValue" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.7 MB_ReadName

The "MB_ReadName" function block is used to read the name of a parameter.

The parameter number of the parameter the name of which is to be read must be applied to the "ParameterNumber" input. The function block processing starts with a rising edge at the "Execute" input. Error-free processing is signaled by "Done"=TRUE. If an error occurs during processing, this is signaled by "Error"=TRUE. In case of an error, a detailed description of the error is output at the "ErrorID" and "ErrorIdent" outputs.

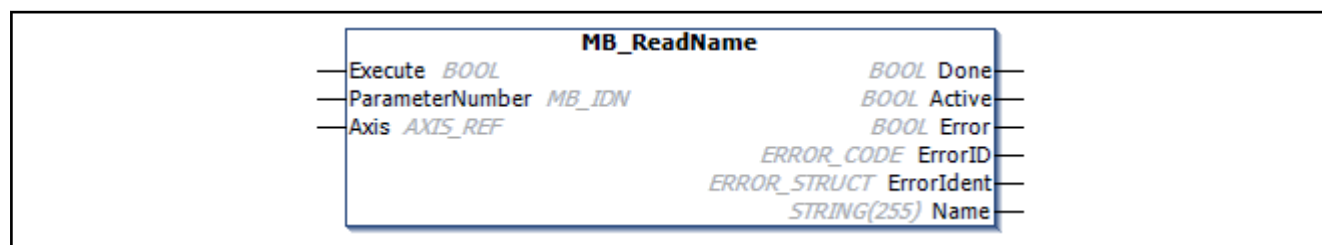


Fig. 13-56: "MB_ReadName" firmware function block

MX_PLCOpen

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter the name of which is to be read

Tab. 13-130: Input variables of the "MB_ReadName" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Name	STRING(255)	If "Done"=TRUE, "Name" displays the name of the selected parameter

* See note below

Tab. 13-131: Output variables of the "MB_ReadName" firmware function block

NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-132: Input/output variable of the "MB_ReadName" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.8 MB_ReadParameter

The "MB_ReadParameter" function block is used to read the current value of a parameter. The return value is DINT.



The input variables are only applied in case of a 0-1- edge in the "MB_ReadParameter" function block at the "Enable" input.



List and string parameters cannot be read with the function block.

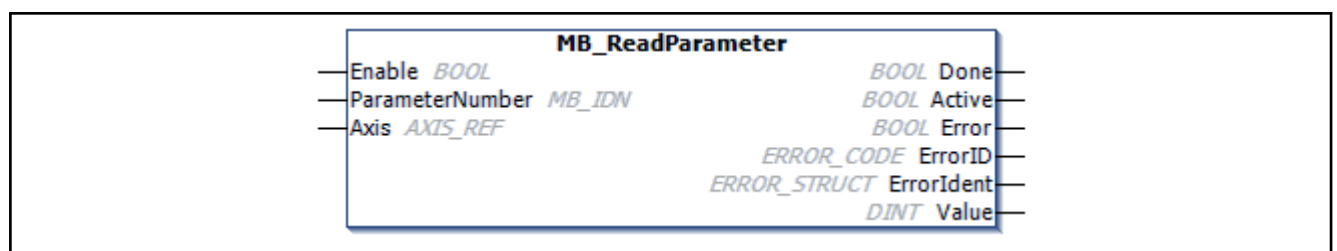


Fig. 13-57: "MB_ReadParameter" firmware function block

Input variable	Data type	Description
Enable	BOOL	The input variables are applied to the "Enable" input at a 0-1- edge
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter (e.g., FP_S_0_0052) the current value of which is to be read

Tab. 13-133: Input variables of the "MB_ReadParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Value	DINT	If "Done"=TRUE, "Value" displays the value of the selected parameter. According to the scaling, the parameter has decimal places that have to be taken into account and used for calculation

* See note below

Tab. 13-134: Output variables of the "MB_ReadParameter" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

MX_PLCOpen

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-135: Input/output variable of the "MB_ReadParameter" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.9 MB_ReadRealListParameter

The "MB_ReadRealListParameter" function block is used to read the current list of a list parameter.

The IDN of the list parameter to be read has to be applied at the "ParameterNumber" input. The number of bytes to be read has to be indicated at the "NoOfBytes" input. The initial address of the field to which the bytes that have been read are to be written has to be indicated at the "ValueAdr" input. The processing of the function block is started with a rising edge at the "Execute" ("Execute"=TRUE) input.



The "ActivateCounter" instance variable can be used to test the "MB_ReadRealListParameter" function block. It counts the positive edges of the Execute signal and thereby signals whether or not the function block is run with an edge.

Local axis

When the function block is used for a local axis, all list elements of the selected parameter are immediately read when the function block is called. After the list elements have been read, control is returned to the program and TRUE is set at the "Done" output.



Observe that the read process can take some time, depending on the volume of data to be read. Use a separate task, if necessary.

Example: Reading all list elements of parameter P-0-0072 (1,024 long words) takes several 100 µs.

Remote axes



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

When the function block is used for remote axes, the read task is returned to the service channel (Sercos III). The function block can only signal "Done"=TRUE at a later call.

The outputs "Done", "Active" and "Error" have to be polled cyclically.

The data transmission can take several seconds, depending on the volume.



Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

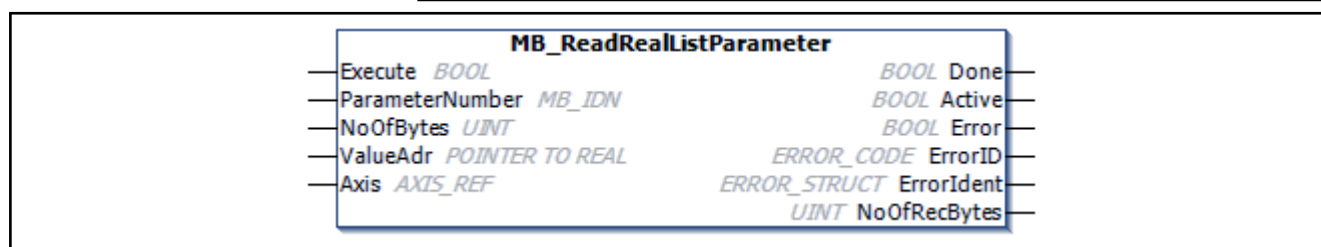


Fig. 13-58: "MB_ReadRealListParameter" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	IDN of the parameter to be read
NoOfBytes	UINT	The number of bytes to be read has to be indicated at the "NoOfBytes" input Example: NoOfBytes := SIZEOF(arFeld1)
ValueAdr	POINTER TO REAL	The initial address of the field to which the bytes that have been read are to be written has to be indicated at the "ValueAdr" input. Example: ValueAdr := ADR(arFeld1)

Tab. 13-136: Input variables of the "MB_ReadRealListParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the data value of the parameter has been successfully read cyclically
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.

MX_PLCOpen

Output variable	Data type	Description
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
NoOfRecBytes	UINT	The "NoOfRecBytes" output signals how many bytes were actually read

* See note below
Tab. 13-137: Output variables of the "MB_ReadRealListParameter" firmware function block

 NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the note below with regard to accessing parameters of remote axes!

Tab. 13-138: Input/output variable of the "MB_ReadRealListParameter" firmware function block

 Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.10 MB_ReadRealParameter

The "MB_ReadRealParameter" function block is used to read the current value of a parameter. The return value is REAL.

 List and string parameters cannot be read with the "MB_ReadRealParameter" function block.

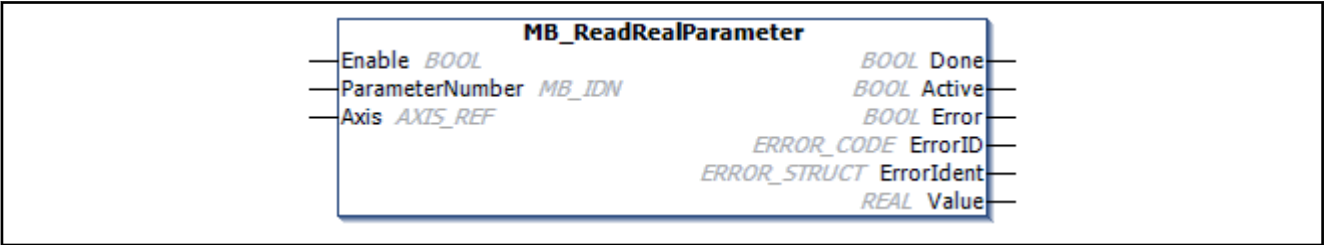


Fig. 13-59: "MB_ReadRealParameter" firmware function block

Input variable	Data type	Description
Enable	BOOL	The input variables are applied to the "Enable" input at a 0-1- edge
ParameterNumber	MB_IDN	At the "ParameterNumber" input apply the IDN of the parameter the current value of which is to be read. Any new parameter number will only be applied with the next positive edge at the "Enable" input

Tab. 13-139: Input variables of the "MB_ReadRealParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Value	REAL	If "Done"=TRUE, "Value" displays the value of the selected parameter

* See note below

Tab. 13-140: Output variables of the "MB_ReadRealParameter" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-141: Input/output variable of the "MB_ReadRealParameter" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.11
MB_ReadSercosAttribute

The "MB_ReadSercosAttribute" function block is used to read the Sercos attribute of a parameter.

The parameter number of the parameter the Sercos attribute of which is to be read has to be applied to the "ParameterNumber" input. The function block processing starts with a rising edge at the "Execute" input. Error-free processing is signaled by "Done"=TRUE. If an error occurs during processing, this is signaled by "Error"=TRUE. In case of an error, a detailed description of the error is output at the "ErrorID" and "ErrorIdent" outputs.

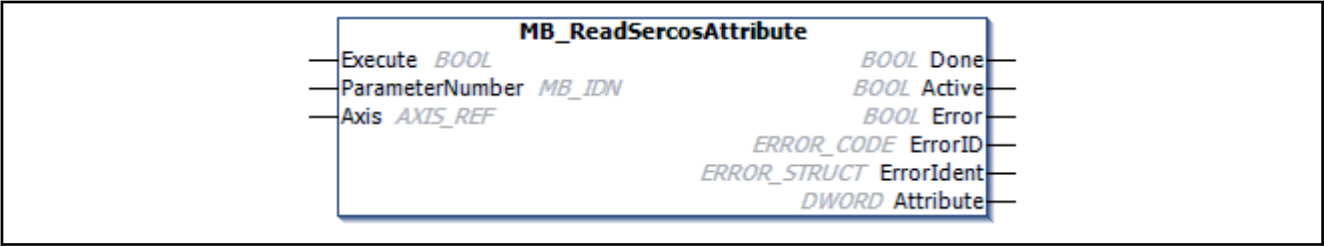


Fig. 13-60:
"MB_ReadSercosAttribute" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	The parameter number of the parameter the Sercos attribute of which is to be read has to be applied to the "ParameterNumber" input. NOTE: Any new parameter number will only be applied upon the next positive edge at the Execute input.

Tab. 13-142:
Input variables of the "MB_ReadSercosAttribute" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the Sercos attribute is successfully being cyclically read
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.

Output variable	Data type	Description
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Attributes	DWORD	If "Done"=TRUE, "Attribute" indicates the attribute of the selected parameter

* See note below
Tab. 13-143: Output variables of the "MB_ReadSercosAttribute" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-144: Input/output variable of the "MB_ReadSercosAttribute" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.12 MB_ReadSercosDataStatus

The "MB_ReadSercosDataStatus" function block is used to read the status of a parameter.

The parameter number of the parameter the status of which is to be read has to be applied to the "ParameterNumber" input.

Error-free processing is signaled by "Done"=TRUE. If an error occurs during processing, this is signaled by "Error"=TRUE. In the case of an error, a detailed description of the error is output at the "ErrorID" and

"ErrorIdent" outputs.

MX_PLCoPen

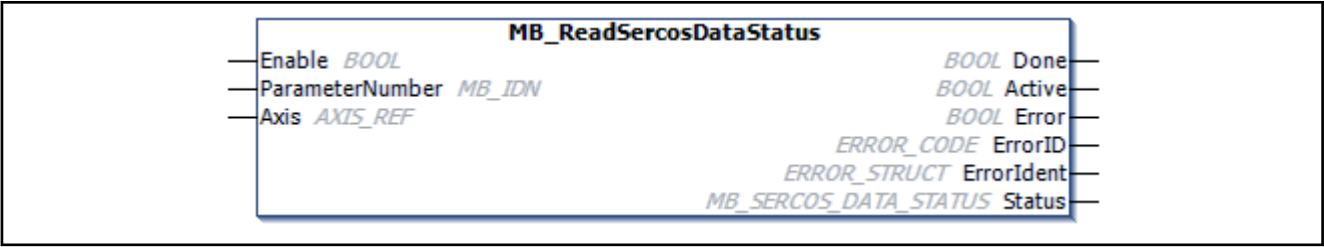


Fig. 13-61: "MB_ReadSercosDataStatus" firmware function block

Input variable	Data type	Description
Enable	BOOL	The function block is state-controlled. The input variables are applied with the positive edge of "Enable". New input values are only applied at another positive edge at "Enable".
ParameterNumber	MB_IDN	The parameter number of the parameter the status of which is to be read has to be applied to the "ParameterNumber" input

Tab. 13-145: Input variables of the "MB_ReadSercosDataStatus" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the function block has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Status	MB_SERCOS_DATA_STATUS	The output displays the current status of the parameter. Enum values for "Status" output: "MB_SERCOS_DATA_NOT_SET": Command has not been set "MB_SERCOS_DATA_SETTING": Command has been set "MB_SERCOS_DATA_END_OK": Command completed without error "MB_SERCOS_DATA_INTERRUPTED": Command interrupted "MB_SERCOS_DATA_IN_PROCESS": Command being processed "MB_SERCOS_DATA_BREAK_ERROR": Command aborted with error "MB_SERCOS_DATA_DATUM_FAULT": Invalid operating data

* See note below
Tab. 13-146: Output variables of the "MB_ReadSercosDataStatus" firmware function block



NOTE: For the list of all possible errors, see: "Error Reference Lists"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-147: *Input/output variable of the "MB_ReadSercosDataStatus" firmware function block*



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.13 MB_ReadStringParameter

The "MB_ReadStringParameter" function block is used to read a parameter the operating data of which is in ASCII format (string).

The parameter number of the parameter the operating data of which is to be read has to be applied to the "ParameterNumber" input. The function block processing starts with a rising edge at the "Execute" input.



The "ActivateCounter" instance variable can be used to test the function block. It counts the positive edges of the "Execute" signal and thereby signals whether or not the function block is run with an edge.

Local axis

When the function block is used for a local axis, all characters of the selected ASCII parameter are immediately read when the function block is called. Control is then returned to the program and TRUE is set at the "Done" output.



The read process can take some time, depending on the text length. A separate task should be used, if necessary.

Remote axes



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

MX_PLCOpen

When the function block is used for remote axes, the read task is returned to the service channel (Sercos III). The function block can only signal "Done" at a later call.

The outputs "Done", "Active" and "Error" have to be polled cyclically.

The data transmission can take several milliseconds, depending on the text length.

 Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

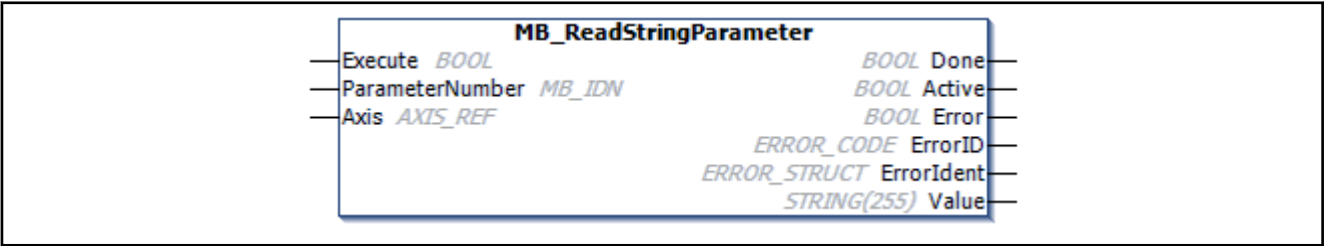


Fig. 13-62: "MB_ReadStringParameter" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	DINT	The parameter number of the parameter the operating data of which is to be read has to be applied to the "ParameterNumber" input

Tab. 13-148: Input variables of the "MB_ReadStringParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Value	STRING(255)	"Value" provides the operating data of the selected parameter

* See note below

Tab. 13-149: Output variables of the "MB_ReadStringParameter" firmware function block

 NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-150: Input/output variable of the "MB_ReadStringParameter" firmware function block

13.4.14 MB_ReadUnit

The "MB_ReadUnit" function block is used to read the unit of a parameter.

The parameter number of the parameter the unit of which is to be read has to be applied to the "ParameterNumber" input. The function block processing starts with a rising edge at the "Execute" input. Error-free processing is signaled by "Done"=TRUE. If an error occurs during processing, this is signaled by "Error"=TRUE. In case of an error, a detailed description of the error is output at the "ErrorID" and "ErrorIdent" outputs.

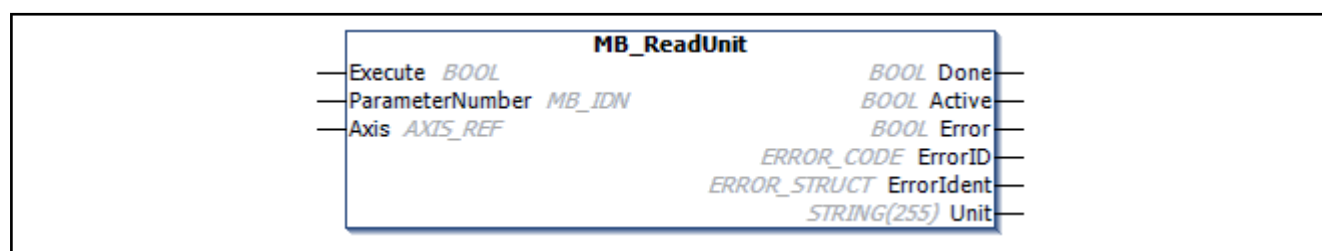


Fig. 13-63: "MB_ReadUnit" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	The parameter number of the parameter the unit of which is to be read has to be applied to the "ParameterNumber" input

Tab. 13-151: Input variables of the "MB_ReadUnit" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.

MX_PLCOpen

Output variable	Data type	Description
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Unit	STRING(255)	If "Done"=TRUE, "Unit" displays the unit of the selected parameter

* See note below

Tab. 13-152: Output variables of the "MB_ReadUnit" firmware function block



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-153: Input/output variable of the "MB_ReadUnit" firmware function block



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.15 MB_WriteListParameter

The "MB_WriteListParameter" function block is used to write defined values to a list parameter.

The IDN of the parameter to be written has to be applied to the "ParameterNumber" input.



Depending on the parameter "S-0-0269, Storage mode", the changed parameter value is saved in the device's internal, non-volatile memory ("flash") either immediately after every change or only after executing the command "S-0-0264, C2200 Backup working memory procedure command".

NOTICE

Destruction of fixed value parameter memory by constantly activating/executing the function block!

When the function block is frequently executed, set "S-0-0269, Storage mode" in such a way that changed parameter values are only saved in the device's internal, non-volatile memory ("flash") by executing the command "S-0-0264, C2200 Backup working memory procedure command".



The "ActivateCounter" instance variable can be used to test the "MB_WriteListParameter" function block. It counts the positive edges of the Execute signal and thereby signals whether or not the function block is run with an edge.

Local axis

When the function block is used for a local axis, all list elements for the applied array are immediately written when the function block is called. Control is then returned to the program and TRUE is set at the "Done" output.



Observe that the write process can take some time, depending on the volume of data to be written. Use a separate task, if necessary.

Example: Writing all list elements of parameter P-0-0072 (1,024 long words) takes several milliseconds.

It is better to only write the actually required elements; if, for example, 8 elements of parameter "P-0-0072, Cam table 1" are to be written (each element of parameter P-0-0072 consists of 4 bytes), then 32 (8*4) has to be applied to "NoOfBytes".

Remote axes



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

When the function block is used for remote axes, the write request is returned to the service channel (Sercos III). The function block can only signal "Done" at a later call.

The outputs "Done", "Active" and "Error" have to be polled cyclically.

The data transmission can take several seconds, depending on the volume.



Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

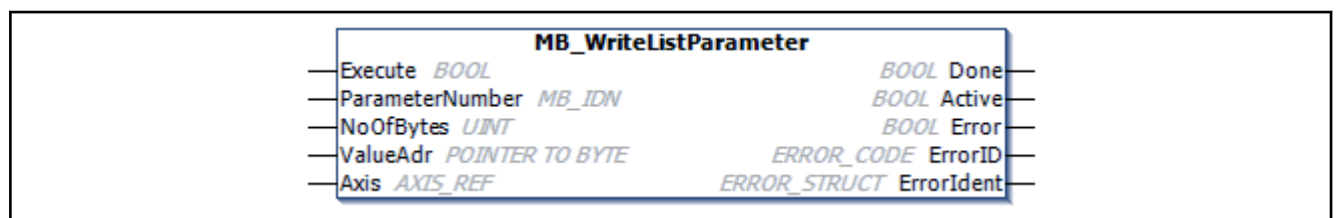


Fig. 13-64: "MB_WriteListParameter" firmware function block

MX_PLCOpen

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	The IDN of the list parameter to be written has to be applied to the "ParameterNumber" input
NoOfBytes	UINT	The number of bytes to be written has to be specified at the "NoOfBytes" input. Example: NoOfBytes:=SIZEOF(arFeld1)
ValueAdr	POINTER TO BYTE	The address of the field containing the data/elements to be written for the list parameter has to be specified at the "ValueAdr" input. Example: ValueAdr := ADR(arFeld1) Note: The address has to match the memory alignment of the data that are accessed: • 1-byte lists: any address • 2-byte lists: any even address • 4-byte lists: any address that can be divided by 4

Tab. 13-154: Input variables of the "MB_WriteListParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-155: Output variables of the "MB_WriteListParameter" firmware function block

NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-156: Input/output variable of the "MB_WriteListParameter" firmware function block

13.4.16 MB_WriteParameter

The "MB_WriteParameter" function block is used to write a defined value to a parameter. To do this, the parameter number of the parameter the value of which is to be written has to be applied to the "ParameterNumber" input. The value to be written to the parameter has to be applied at the "Value" input as DINT type.



List and string parameters cannot be written with "MB_WriteParameter".



According to the scaling, the parameter has decimal places that have to be taken into account and used for calculation!



Depending on the parameter "S-0-0269, Storage mode", the changed parameter value is saved in the device's internal, non-volatile memory ("flash") either immediately after every change or only after executing the command "S-0-0264, C2200 Backup working memory procedure command".

NOTICE

Destruction of fixed value parameter memory by constantly activating/executing the function block!

When the function block is frequently executed, set "S-0-0269, Storage mode" in such a way that changed parameter values are only saved in the device's internal, non-volatile memory ("flash") by executing the command "S-0-0264, C2200 Backup working memory procedure command".



The "ActivateCounter" instance variable can be used to test the "MB_WriteParameter" function block. It counts the positive edges of the "Execute" signal and thereby signals whether or not the function block is run with an edge.

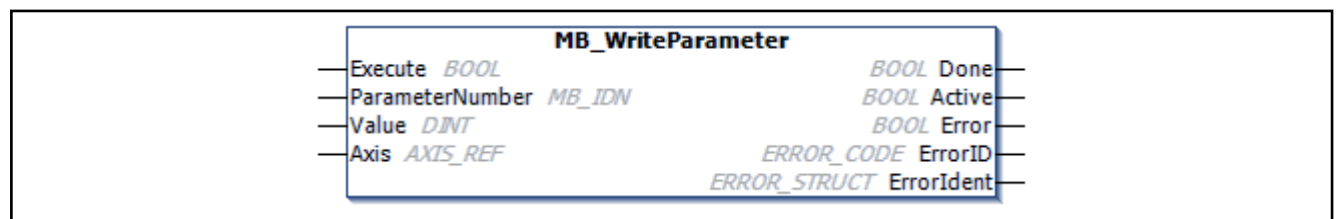


Fig. 13-65: "MB_WriteParameter" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	The parameter number of the parameter to be written has to be applied to the "ParameterNumber" input
Value	DINT	The value that is to be written to the parameter has to be applied to the "Value" input. Be sure to take the scaling of the selected parameter into account

Tab. 13-157: Input variables of the "MB_WriteParameter" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below
 Tab. 13-158: *Output variables of the "MB_WriteParameter" firmware function block*



NOTE: For the list of all possible errors, see: "[Error Reference Lists](#)"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-159: *Input/output variable of the "MB_WriteParameter" firmware function block*



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.17 MB_WriteRealListParameter

The "MB_WriteRealListParameter" function block is used to write defined values to a list parameter.

The IDN of the list parameter to be written has to be applied at the "ParameterNumber" input.



Depending on the parameter "S-0-0269, Storage mode", the changed parameter value is saved in the device's internal, non-volatile memory ("flash") either immediately after every change or only after executing the command "S-0-0264, C2200 Backup working memory procedure command".

NOTICE

Destruction of fixed value parameter memory by constantly activating/executing the function block!

When the function block is frequently executed, set "S-0-0269, Storage mode" in such a way that changed parameter values are only saved in the device's internal, non-volatile memory ("flash") by executing the command "S-0-0264, C2200 Backup working memory procedure command".



The "ActivateCounter" instance variable can be used to test the "MB_WriteRealListParameter" function block. It counts the positive edges of the Execute signal and thereby signals whether or not the function block is run with an edge.

The "NoOfBytes" and "ValueAdr" inputs should be used as follows:

Example:

```
inst_MB_WriteRealListParameter : MB_WriteRealListParameter;
arFeld1 : ARRAY[1..16] OF REAL;
inst_MB_WriteRealListParameter(...,
NoOfBytes:=SIZEOF(arFeld1), ValueAdr:=ADR(arFeld1), ...);
```

Local axis

When the function block is used for a local axis, all list elements for the applied array are immediately written when the function block is called. Control is then returned to the program and TRUE is set at the "Done" output.



Observe that the write process can take some time, depending on the volume of data to be written. A separate task should be used, if necessary.

Example: Writing all list elements of parameter P-0-0072 (1,024 long words) takes several milliseconds.

Remote axes



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

When the function block is used for remote axes, the write request is returned to the service channel (Sercos III). The function block can only signal "Done"=TRUE at a later call.

The outputs "Done", "Active" and "Error" have to be polled cyclically.

The data transmission can take several seconds, depending on the volume.



Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

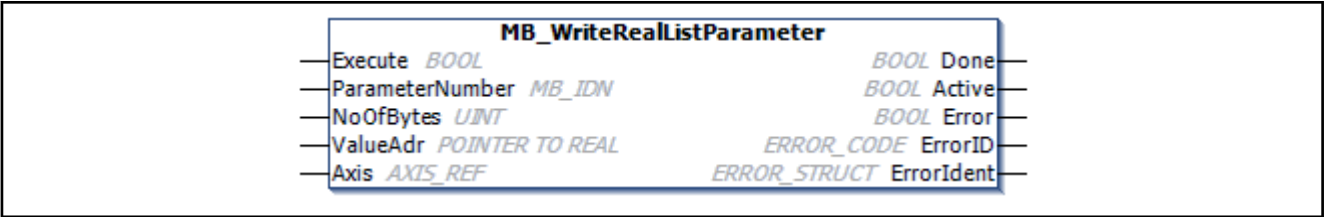


Fig. 13-66: "MB_WriteRealListParameter" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	The IDN of the list parameter to be written has to be applied to the "ParameterNumber" input
NoOfBytes	UINT	The number of bytes to be written has to be specified at the "NoOfBytes" input Example: NoOfBytes := SIZEOF(arFeld1); Note: Since the list elements are stored in a REAL array (REAL data type = 4-byte), the NoOfBytes is calculated as follows: NoOfBytes = 4 * (number of elements)
ValueAdr	POINTER TO REAL	The initial address of the field to which the bytes that have been read are to be written has to be indicated at the "ValueAdr" input. Example: ValueAdr := ADR(arFeld1); Note: The applied field has to be of the REAL data type.

Tab. 13-160: Input variables of the "MB_WriteRealListParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below
Tab. 13-161: Output variables of the "MB_WriteRealListParameter" firmware function block

 NOTE: For the list of all possible errors, see: "Error Reference Lists"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-162: Input/output variable of the "MB_WriteRealListParameter" firmware function block

13.4.18 MB_WriteRealParameter

The "MB_WriteRealParameter" function block is used to write a defined value to the parameter belonging to the parameter number. The data format of the parameter to be written has to be DEC_OV, DEC_MV or FLOAT; parameters of the data formats BIN, HEX and DIN cannot be written.



Depending on the parameter "S-0-0269, Storage mode", the changed parameter value is saved in the device's internal, non-volatile memory ("flash") either immediately after every change or only after executing the command "S-0-0264, C2200 Backup working memory procedure command".

NOTICE

Destruction of fixed value parameter memory by constantly activating/executing the function block!

When the function block is frequently executed, set "S-0-0269, Storage mode" in such a way that changed parameter values are only saved in the device's internal, non-volatile memory ("flash") by executing the command "S-0-0264, C2200 Backup working memory procedure command".



List and string parameters cannot be written with the "MB_WriteRealParameter" function block; please use the "MB_WriteListParameter" or "MB_WriteRealListParameter" function block to write list parameters.

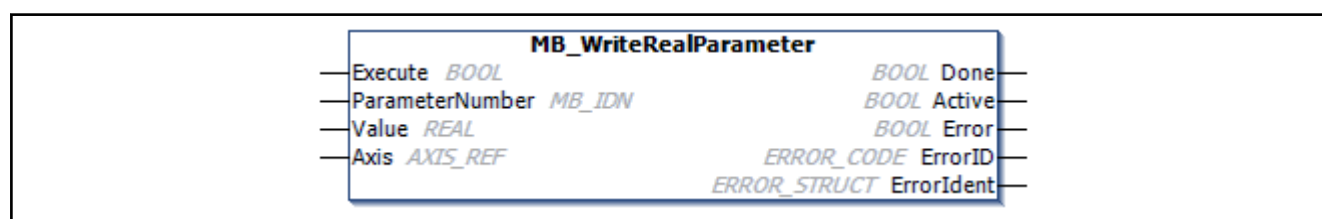


Fig. 13-67: "MB_WriteRealParameter" firmware function block

MX_PLCoPen

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	MB_IDN	The IDN of the list parameter to be written has to be applied to the "ParameterNumber" input
Value	REAL	The value that is to be written to the parameter has to be applied to the "Value" input. Be sure to take the scaling of the selected parameter into account

Tab. 13-163: *Input variables of the "MB_WriteRealParameter" firmware function block*

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-164: *Output variables of the "MB_WriteRealParameter" firmware function block*



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Please refer to the notes below with regard to accessing parameters of remote axes!

Tab. 13-165: *Input/output variable of the "MB_WriteRealParameter" firmware function block*



For remote axes, the task is returned to the service channel (Sercos III). The function block can only signal "Done"="TRUE" at a later call. This means the call times for this function block are always brief. The outputs "Done", "Active" and "Error" have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

13.4.19 MB_WriteStringParameter

The "MB_WriteStringParameter" function block is used to write a parameter the operating data of which is in ASCII format (string).

The parameter number of the parameter the operating data of which is to be written has to be applied to the "ParameterNumber" input. The function block processing starts with a rising edge at the "Execute" input.



Depending on the parameter "S-0-0269, Storage mode", the changed parameter value is saved in the device's internal, non-volatile memory ("flash") either immediately after every change or only after executing the command "S-0-0264, C2200 Backup working memory procedure command".

NOTICE

Destruction of fixed value parameter memory by constantly activating/executing the function block!

When the function block is frequently executed, set "S-0-0269, Storage mode" in such a way that changed parameter values are only saved in the device's internal, non-volatile memory ("flash") by executing the command "S-0-0264, C2200 Backup working memory procedure command".



The "ActivateCounter" instance variable can be used to test the function block. It counts the positive edges of the "Execute" signal and thereby signals whether or not the function block is run with an edge.

Local axis

When the function block is used for a local axis, all characters of the selected ASCII parameter are immediately written when the function block is called. Control is then returned to the program and TRUE is set at the "Done" output.



The write process can take some time, depending on the text length. A separate task should be used, if necessary.

Remote axes



Accessing parameters of remote axes is possible in the MLD-M system mode, in the CCD basic mode and in the CCD system mode.

MX_PLCOpen

When the function block is used for remote axes, the write request is returned to the service channel (Sercos III). The function block can only signal "Done" at a later call.

The outputs "Done", "Active" and "Error" have to be polled cyclically.

The data transmission can take several milliseconds, depending on the text length.

 Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

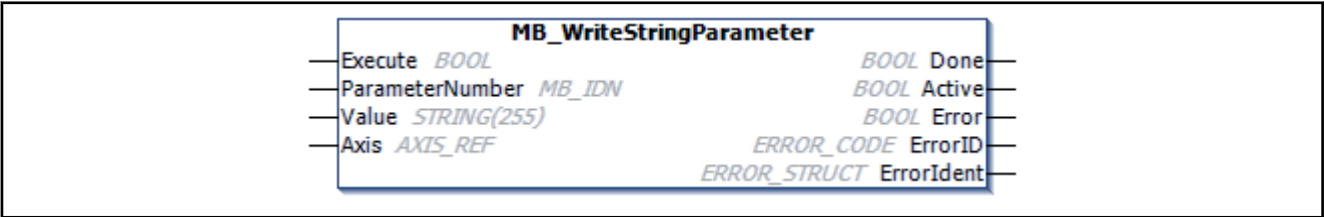


Fig. 13-68: "MB_WriteStringParameter" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
ParameterNumber	DINT	The parameter number of the parameter the operating data of which is to be written has to be applied at the "ParameterNumber" input

Tab. 13-166: Input variables of the "MB_WriteStringParameter" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Value	STRING(255)	"Value" provides the operating data of the selected parameter

* See note below

Tab. 13-167: Output variables of the "MB_WriteStringParameter" firmware function block

 NOTE: For the list of all possible errors, see: "Error Reference Lists"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...)

Tab. 13-168: Input/output variable of the "MB_WriteStringParameter" firmware function block

13.4.20 MX_Command

The "MX_Command" function block is used to execute a drive command.

The command is activated or deactivated with the "Enable" input. If TRUE has been applied to the input, the command is started by writing "3" (dec.). If FALSE, the command is aborted/terminated by writing "0".

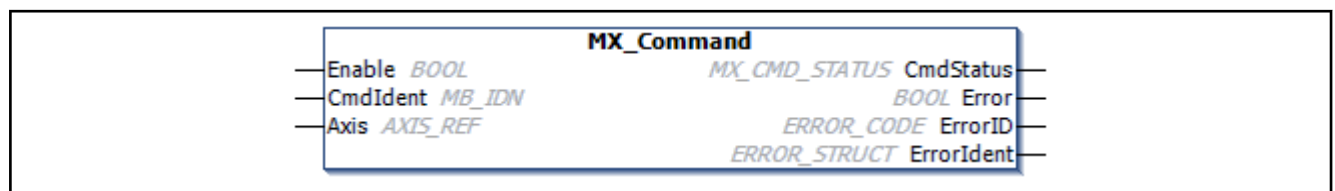


Fig. 13-69: "MX_Command" firmware function block

MX_PLCOpen

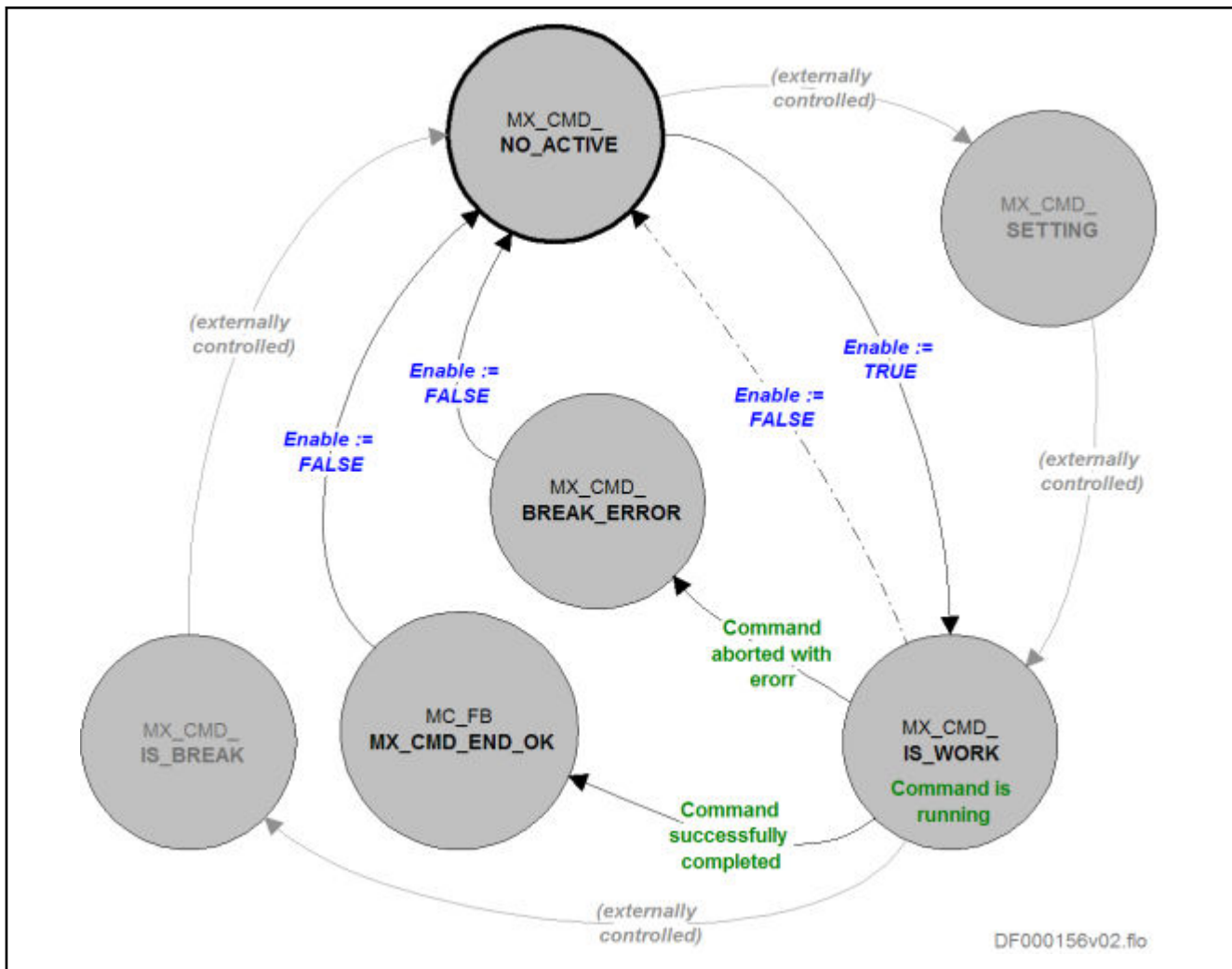


Fig. 13-70: Executing a drive command



For remote axes, the task is returned to the service channel (Sercos III). "CmdStatus" can only signal "MX_CMD_END_OK" at a later call. This means the call times for this function block are always brief. The "CmdStatus" and "Error" outputs have to be polled cyclically. The data transmission can take a few seconds, depending on the service channel load.

Data cannot be exchanged with remote axes if the Sercos ring is not yet in phase2. Any attempt to do so results in an error.

Input variable	Data type	Description
Enable	BOOL	The function block is state-controlled. The command is started with "Enable"=TRUE, it is cleared with "Enable"=FALSE. After the input has been set to TRUE, there is usually a delay until the output displays "CmdStatus"=MX_CMD_END_OK or "CmdStatus"=MX_CMD_BREAK_ERROR. After the input has been reset, wait until the "CmdStatus" output displays MX_CMD_NO_ACTIVE. (See also command description and "Signal time diagrams")
CmdIdent	MB_IDN	IDN of the command

Tab. 13-169: Input variables of the "MX_Command" firmware function block

Output variable	Data type	Description
CmdStatus	MX_CMD_STATUS	The output displays the current status of the command. Enum values for "CmdStatus" output: "MX_CMD_NO_ACTIVE": Command has not been set "MX_CMD_SETTING": Command has been set "MX_CMD_END_OK": Command completed without error "MX_CMD_IS_BREAK": Command interrupted "MX_CMD_IS_WORK": Command being processed "MX_CMD_BREAK_ERROR": Command aborted with error
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-170: Output variables of the "MX_Command" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Defines the axis. The following axes are possible: MPB (IndraDrive) and HDC (HydraulicDrive): Axis1 MPC (IndraDrive): - Axis1 (master) or - Axis2...Axis10 (a total of 9 slaves / remote axes) The remote axes contained in the list "P-0-1801.0.10, CCD: Addresses of projected drives" correspond to the order of Axis2 to Axis10 (Axis2 acts on the drive of list element 0 of P-0-1801.0.10,...) Note: The input is only applied at a rising edge in the Enable input. A separate instance should be used for each axis when multiple axes are to be controlled.

Tab. 13-171: Input/output variable of the "MX_Command" firmware function block

MX_PLCoPen

13.4.21 MX_fReadParamDINT

The "MX_fReadParamDINT" firmware function is used to read a parameter value and return it as DINT.

With this function, it is not immediately apparent whether or not the parameter value has been successfully read. The "MX_ReadParamDINT" function block is suited for this purpose.

The return value for unsigned 32-bit parameters can simply be converted with the "DINT_TO_UDINT" function.



Fig. 13-71: "MX_fReadParamDINT" firmware function

Input variable	Data type	Description
Ident	MB_IDN	The IDN of the parameter the value of which is to be read has to be applied at the "Ident" input
ListElem	INT	The value of a single parameter as well as the values of list parameters can be read. Apply "0" to "ListElem" to read a single parameter . For list parameters , the element 0 is read with "0", the first element with "1", the second element with "2", etc. Applying "-1" to "ListElem" reads the actual length, "-2" reads the maximum length of a list parameter
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-172: Input variables of the "MX_fReadParamDINT" firmware function

Return value	Data type	Description
MX_fReadParamDINT	DINT	Provides the value of the element specified by "Ident" and "ListElem". Note: The value "0" is returned if the parameter or list element cannot be read

Tab. 13-173: Return value of the "MX_fReadParamDINT" firmware function

13.4.22 MX_fReadStringParam

The "MX_fReadStringParam" firmware function is used to read a parameter the operating data of which is in ASCII format (string).

 A closing 0 character is copied in addition to "MaxLen". The return value returns the characters actually read (no 0).

When the function is called, the entire text is immediately read. Control is then returned to the program.

 The read process can take some time, depending on the text length. A separate task should be used, if necessary.



Fig. 13-72: "MX_fReadStringParam" firmware function

Input variable	Data type	Description
Ident	MB_IDN	IDN of the parameter
Text	STRING(255)	255-character text buffer
MaxLen	INT	Number of characters being read (no closing 0)
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-174: Input variables of the "MX_fReadStringParam" firmware function

Return value	Data type	Description
MX_fReadStringParam	INT	Returns the characters actually read (no 0)

Tab. 13-175: Return value of the "MX_fReadStringParam" firmware function

13.4.23 MX_fSetParamLimits

The "MX_fSetParamLimits" firmware function is used to set the limit values of the Global Register PLC parameters.



The limit values are reset to their default values by switching off or by "load defaults procedure for PLC" [C0730 Load def. proc. com. (load defaults procedure for PLC)].



This function mustn't be called from the system event "PositionLoop". When the function is called from "PositionLoop", a PLC exception occurs and the function returns the value "0".



Fig. 13-73: "MX_fSetParamLimits" firmware function

Input variable	Data type	Description
Ident	MB_IDN	A constant from the global variables can be transmitted here
MinValue	DINT	Minimum value to be set
MaxValue	DINT	Maximum value to be set

Tab. 13-176: Input variables of the "MX_fSetParamLimits" firmware function

Return value	Data type	Description
MX_fSetParamLimits	UINT	The function can return the following values: 0: Limit values were successfully set 1: Error, invalid IDN was specified

Tab. 13-177: Return value of the "MX_fSetParamLimits" firmware function

13.4.24 MX_fSetParamName

The "MX_fSetParamName" firmware function is used to set the names of the Global Register PLC parameters and of the PLC I/O parameters.

 By switching off, the names of the Global Register PLC parameters and of the PLC I/O parameters are reset to their default values.

If the names of the Global Register PLC parameters or of the PLC I/O parameters were changed, they are set to their English-language names by "load defaults procedure for PLC" [C0730 Load def. proc. com. (load defaults procedure for PLC)]. This is done for technical reasons.

 This function mustn't be called from the system event "PositionLoop". When the function is called from "PositionLoop", a PLC exception occurs and the function returns the value "0".



Fig. 13-74: "MX_fSetParamName" firmware function

Input variable	Data type	Description
Ident	MB_IDN	A constant from the global variables can be transmitted here
NewName	STRING (60)	Name to be set (maximum length: 60 characters)

Tab. 13-178: Input variables of the "MX_fSetParamName" firmware function

Return value	Data type	Description
MX_fSetParamName	UINT	The function can return the following values: • 0: Name was successfully set • 1: Error, invalid IDN was specified

Tab. 13-179: Return value of the "MX_fSetParamName" firmware function

13.4.25 MX_fSetParamUnit

The "MX_fSetParamUnit" firmware function is used to set the units of the Global Register PLC parameters.

 The units are reset to their default values by switching off or by "load defaults procedure for PLC" [C0730 Load def. proc. com. (load defaults procedure for PLC)].

 This function mustn't be called from the system event "PositionLoop". When the function is called from "PositionLoop", a PLC exception occurs and the function returns the value "0".



Fig. 13-75: "MX_fSetParamUnit" firmware function

Input variable	Data type	Description
Ident	MB_IDN	A constant from the global variables can be transmitted here
NewUnit	STRING (20)	Unit to be set (maximum length: 20 characters)

Tab. 13-180: Input variables of the "MX_fSetParamUnit" firmware function

Return value	Data type	Description
MX_fSetParamUnit	UINT	The function can return the following values: 0: Unit was successfully set 1: Error, invalid IDN was specified

Tab. 13-181: Return value of the "MX_fSetParamUnit" firmware function

13.4.26 MX_fWriteParamDINT

The "MX_fWriteParamDINT" firmware function is used to write a parameter. It is possible to write with or without buffering.



When writing with buffering, the parameter values are transferred to the non-volatile memory every time. Data should only be written with buffering when necessary, since this requires a lot of computing time.

With list parameters, only set "WriteBuffered" to TRUE for the last write access, since the entire list will be buffered with each access.

Any buffered access attempt while the parameter is buffering will fail (ACCESS_ERROR).

NOTICE

Destruction of device-internal, non-volatile memory ("flash") by constant buffered writing of the parameter values!

When the function block is frequently executed, only write parameter values in buffered form ("WriteBuffered"=TRUE) when necessary. For list parameters, only set "WriteBuffered" to TRUE for the last write access, for example.

With this function, it is not immediately apparent whether or not the parameter values were successfully written. The "MX_WriteParamDINT" function block is suited for this purpose.

The "Value" input value for unsigned 32-bit parameters can be converted with the "UDINT_TO_DINT" function.

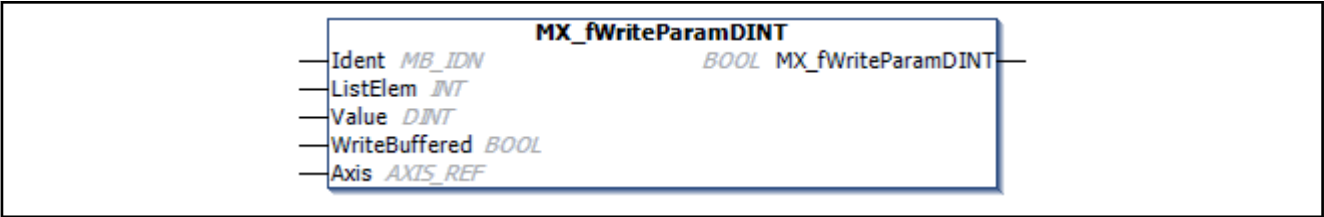


Fig. 13-76: "MX_fWriteParamDINT" firmware function

Input variable	Data type	Description
Ident	MB_IDN	IDN of the specified parameter
ListElem	INT	List element number: For list parameters starting with "0" and for single parameter always "0" "-1" for writing the actual length (only allowed for list parameters)
Value	INT	Value with which the selected element is to be written
WriteBuffered	BOOL	"WriteBuffered"=TRUE: Value is buffered; "WriteBuffered"=FALSE: Value is not buffered
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-182: Input variables of the "MX_fWriteParamDINT" firmware function

Return value	Data type	Description
MX_fWriteParamDINT	BOOL	Dummy output without significance

Tab. 13-183: Return value of the "MX_fWriteParamDINT" firmware function

13.4.27 MX_fWriteStringParam

The "MX_fWriteStringParam" firmware function is used to write a text parameter.
It is possible to write with or without buffering.



When writing with buffering, the parameter values are transferred to the non-volatile memory every time. Data should only be written with buffering when necessary, since this requires a lot of computing time.

Any buffered access attempt while the parameter is buffering will fail (ACCESS_ERROR).

When the function is called, the entire text is immediately written. Control is then returned to the program.



The write process can take some time, depending on the text length. A separate task should be used, if necessary.

NOTICE

Destruction of device-internal, non-volatile memory ("flash") by constant buffered writing of the parameter values!

When the function block is frequently executed, only write parameter values in buffered form ("WriteBuffered"=TRUE) when necessary. For list parameters, only set "WriteBuffered" to TRUE for the last write access, for example.

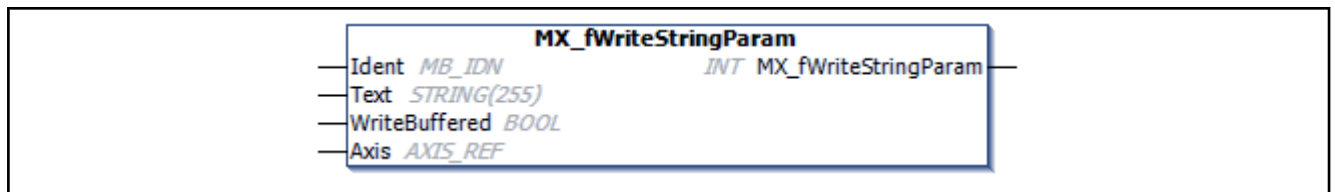


Fig. 13-77: "MX_fWriteStringParam" firmware function

Input variable	Data type	Description
Ident	MB_IDN	IDN of the specified parameter
Text	STRING(255)	Text string
WriteBuffered	BOOL	"WriteBuffered"=TRUE: Value is buffered "WriteBuffered"=FALSE: Value is not buffered
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-184: Input variables of the "MX_fWriteStringParam" firmware function

Return value	Data type	Description
MX_fWriteStringParam	INT	The return value returns the characters actually written (no 0)

Tab. 13-185: Return value of the "MX_fWriteStringParam" firmware function

13.4.28 MX_ReadParamDINT

The "MX_ReadParamDINT" function block is used to read a parameter value and return it as DINT.

The function block also indicates whether or not the parameter value was successfully read.

The return value for unsigned 32-bit parameters can be converted with the "DINT_TO_UDINT" function.

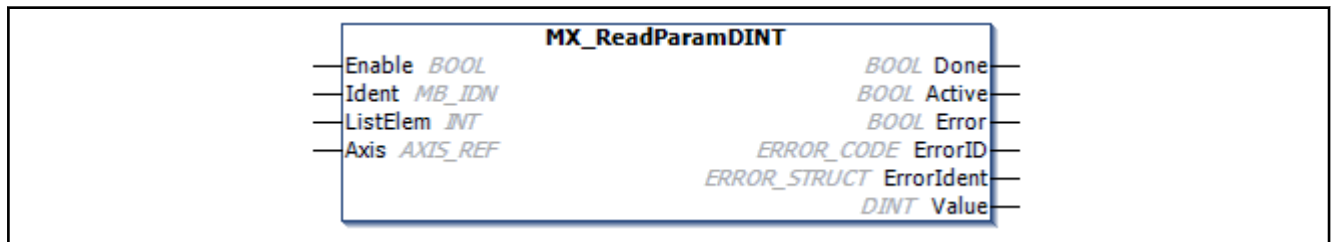


Fig. 13-78: "MX_ReadParamDINT" firmware function block

Input variable	Data type	Description
Enable	BOOL	The function block is activated with "Enable"=TRUE. All outputs are reset with "Enable"=FALSE
Ident	DINT	IDN of the parameter
ListElem	INT	List element numbers: For list parameters starting with "0" and for single parameter always "0" "-1" for reading the actual length (only allowed for list parameters) "-2" for reading the maximum length (only allowed for list parameters)

Tab. 13-186: Input variables of the "MX_ReadParamDINT" firmware function block

MX_PLCOpen

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.
Value	DINT	If the output "Done"=TRUE, "Value" displays the value of the selected element

* See note below

Tab. 13-187: Output variables of the "MX_ReadParamDINT" firmware function block



NOTE: For the list of all possible errors, see: ["Error Reference Lists"](#)

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-188: Input/output variable of the "MX_ReadParamDINT" firmware function block

13.4.29 MX_WriteParamDINT

The "MX_WriteParamDINT" firmware function block is used to write a parameter. It is possible to write with or without buffering.



When writing with buffering, the parameter values are transferred to the non-volatile memory every time. Data should only be written with buffering when necessary, since this requires a lot of computing time.

With list parameters, only set "WriteBuffered" to TRUE for the last write access, since the entire list will be buffered with each access.

Any buffered access attempt while the parameter is buffering will fail (ACCESS_ERROR).

NOTICE

Destruction of device-internal, non-volatile memory ("flash") by constant buffered writing of the parameter values!

When the function block is frequently executed, only write parameter values in buffered form ("WriteBuffered"=TRUE) when necessary. For list parameters, only set "WriteBuffered" to TRUE for the last write access, for example.

The "Value" input value for unsigned 32-bit parameters can be simply converted with the "UDINT_TO_DINT" function.

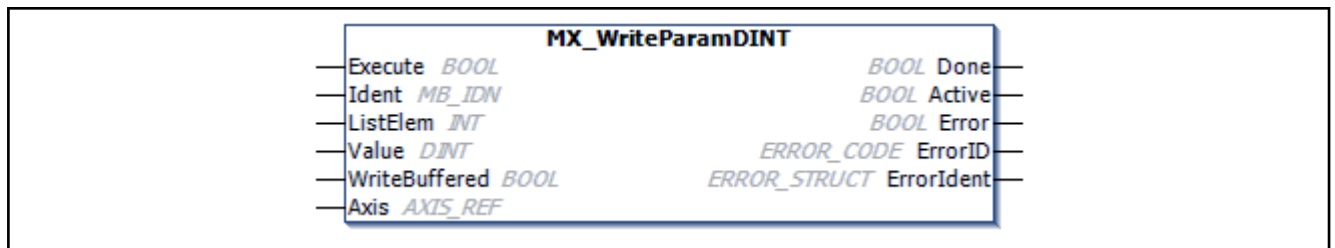


Fig. 13-79: "MX_WriteParamDINT" firmware function block

Input variable	Data type	Description
Execute	BOOL	See "Signal Time Diagrams"
Ident	MB_IDN	IDN of the parameter
ListElem	INT	The element that is to be written has to be specified with the "ListElem" input. List element number: For list parameters starting with "0" and for single parameter always "0" "-1" for writing the actual length (only allowed for list parameters)
Value	DINT	The value with which the parameter is to be written has to be applied at "Value". Be sure to take the scaling of the selected parameter into account. According to the scaling, the parameter has decimal places that have to be taken into account and used for calculation
WriteBuffered	BOOL	When "WriteBuffered"=TRUE, the value is written with buffering if the parameter is intended for this purpose. If FALSE, the parameter is always written without buffering

Tab. 13-189: Input variables of the "MX_WriteParamDINT" firmware function block

Output variable	Data type	Description
Done	BOOL	The function block signals with "Done"=TRUE that the processing of the task has been successfully completed
Active	BOOL	For remote axes, "Active"=TRUE signals that the function block has accepted the processing command; this output is not set for single axes!
Error	BOOL	With "Error"=TRUE the function block signals that an error is present
ErrorID *	ERROR_CODE	"ErrorID" gives a rough error description.
ErrorIdent *	ERROR_STRUCT	Via the elements: "Table", "Additional1" and "Additional2", "ErrorIdent" shows the exact cause of the error.

* See note below

Tab. 13-190: Output variables of the "MX_WriteParamDINT" firmware function block



NOTE: For the list of all possible errors, see: "Error Reference Lists"

Input/output variable (VAR_IN_OUT)	Data type	Description
Axis	AXIS_REF	Axis number (only the global variable "Axis1" can be transmitted here)

Tab. 13-191: Input/output variable of the "MX_WriteParamDINT" firmware function block

MX_PLCOpen

13.5 "Scaling" - Functions for Scaling

13.5.1 MX_fDINT_AccTo_REAL

The firmware function "MX_fDINT_AccTo_REAL" converts an acceleration value in the fixed point format of the DINT data type into a REAL value, according to the scaling which has been set.

If the axis reference is invalid, 0.0 is returned.

The function can be used for the virtual master axis / the master axis encoder and for the real axes "Axis1", "Axis2", etc.

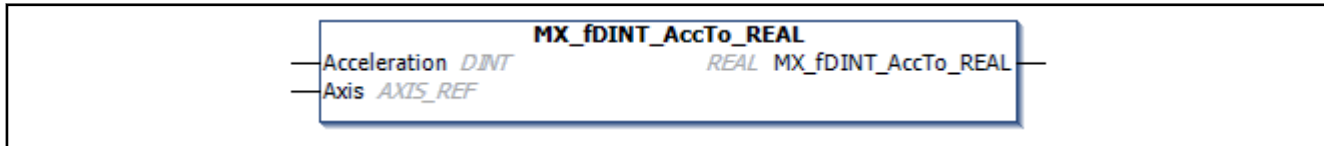


Fig. 13-80: Firmware Function "MX_fDINT_AccTo_REAL"

Input variable	Data type	Description
Acceleration	DINT	Acceleration value in fixed-point DINT format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-192: Input Variables of the Firmware Function "MX_fDINT_AccTo_REAL"

Return value	Data type	Description
MX_fDINT_AccTo_REAL	REAL	Returns an acceleration value in the REAL format, according to the scaling which has been set

Tab. 13-193: Return Value of the Firmware Function "MX_fDINT_AccTo_REAL"

13.5.2 MX_fDINT_DistTo_REAL

The firmware function "MX_fDINT_DistTo_REAL" converts a position value in the fixed point format of the DINT data type into a REAL value, according to the scaling which has been set.

If the axis reference is invalid, 0.0 is returned.

The function can be used for the virtual master axis / the master axis encoder and for the real axes "Axis1", "Axis2", etc.

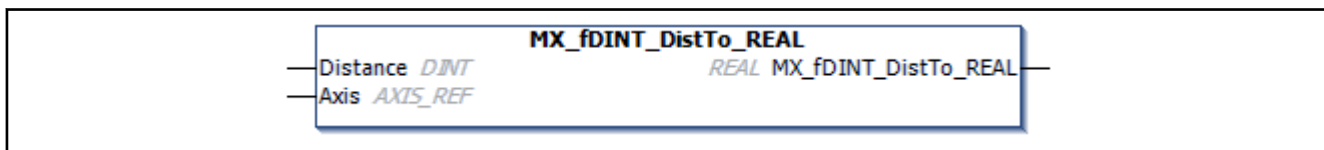


Fig. 13-81: Firmware Function "MX_fDINT_DistTo_REAL"

Input variable	Data type	Description
Distance	DINT	Position value in fixed-point DINT format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-194: Input Variables of the Firmware Function "MX_fDINT_DistTo_REAL"

Return value	Data type	Description
MX_fDINT_DistTo_REAL	REAL	Returns a position value in the REAL format, according to the scaling which has been set

Tab. 13-195: Return Value of the Firmware Function "MX_fDINT_DistTo_REAL"

13.5.3 MX_fDINT_VelTo_REAL

The firmware function "MX_fDINT_VelTo_REAL" converts a velocity value in the fixed point format of the DINT data type into a REAL value, according to the scaling which has been set.

If the axis reference is invalid, 0 is returned.

The function can be used for the virtual master axis / the master axis encoder and for the real axes "Axis1", "Axis2", etc.

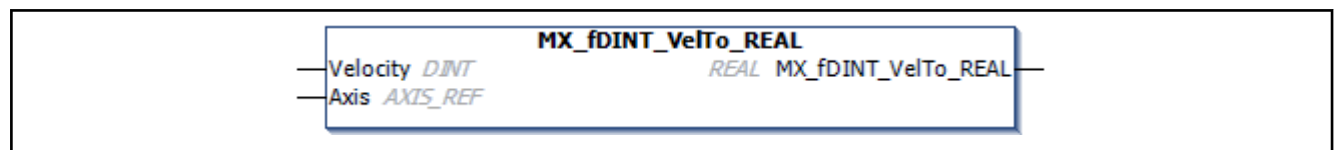


Fig. 13-82: Firmware Function "MX_fDINT_VelTo_REAL"

Input variable	Data type	Description
Velocity	DINT	Velocity value in fixed-point DINT format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-196: Input Variables of the Firmware Function "MX_fDINT_VelTo_REAL"

Return value	Data type	Description
MX_fDINT_VelTo_REAL	REAL	Returns a velocity value in the REAL format, according to the scaling which has been set

Tab. 13-197: Return Value of the Firmware Function "MX_fDINT_VelTo_REAL"

13.5.4 MX_fREAL_AccTo_DINT

The firmware function "MX_fREAL_AccTo_DINT" converts an acceleration value in the REAL format into a fixed-point DINT value, according to the scaling which has been set.

If the axis reference is invalid, 0.0 is returned.

The function can be used for the virtual master axis / the master axis encoder and for the real axes "Axis1", "Axis2", etc.

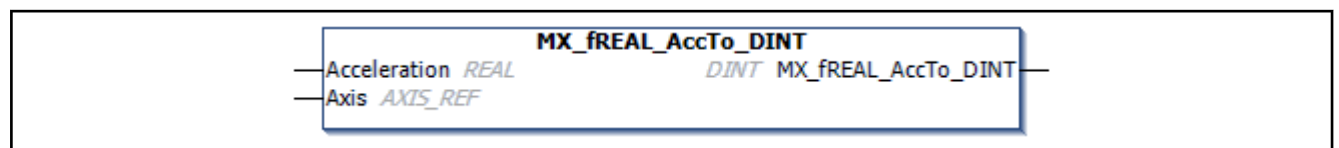


Fig. 13-83: Firmware Function "MX_fREAL_AccTo_DINT"

MX_PLCoPen

Input variable	Data type	Description
Acceleration	REAL	Acceleration value in the REAL format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-198: Input Variables of the Firmware Function "MX_fREAL_AccTo_DINT"

Return value	Data type	Description
MX_fREAL_AccTo_DINT	DINT	Returns an acceleration value in the fixed-point DINT format, according to the scaling which has been set

Tab. 13-199: Return Value of the Firmware Function "MX_fREAL_AccTo_DINT"

13.5.5 MX_fREAL_DistTo_DINT

The firmware function "MX_fREAL_DistTo_DINT" converts a position value in the REAL format into a fixed-point DINT value, according to the scaling which has been set.

If the axis reference is invalid, 0 is returned.

The function can be used for the virtual master axis / the master axis encoder and for the real axes "Axis1", "Axis2", etc.



Fig. 13-84: Firmware Function "MX_fREAL_DistTo_DINT"

Input variable	Data type	Description
Distance	REAL	Position value in the REAL format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-200: Input Variables of the Firmware Function "MX_fREAL_DistTo_DINT"

Return value	Data type	Description
MX_fDINT_DistTo_REAL	DINT	Returns a position value in the fixed-point DINT format, according to the scaling which has been set

Tab. 13-201: Return Value of the Firmware Function "MX_fREAL_DistTo_DINT"

13.5.6 MX_fREAL_TorqueRampTo_DINT

The firmware function "MX_fREAL_TorqueRampTo_DINT" converts a torque/force ramp value in the REAL format into a fixed-point DINT value, according to the scaling which has been set.

The function can be used for the real axes "Axis1", "Axis2", etc.



Fig. 13-85: Firmware Function "MX_fREAL_TorqueRampTo_DINT"

Input variable	Data type	Description
TorqueRamp	REAL	Torque/force ramp value in the REAL format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-202: Input Variables of the Firmware Function "MX_fREAL_TorqueRampTo_DINT"

Return value	Data type	Description
MX_fREAL_TorqueRampTo_DINT	DINT	Returns a torque/force ramp value in the fixed-point DINT format, according to the scaling which has been set

Tab. 13-203: Return Value of the Firmware Function "MX_fREAL_TorqueRampTo_DINT"

13.5.7 MX_fREAL_TorqueTo_DINT

The firmware function "MX_fREAL_TorqueTo_DINT" converts a torque/force value in the REAL format into a fixed-point DINT value, according to the scaling which has been set.

The function can be used for the real axes "Axis1", "Axis2", etc.

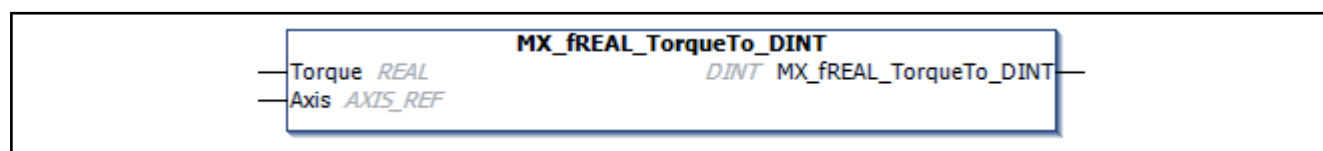


Fig. 13-86: Firmware Function "MX_fREAL_TorqueTo_DINT"

Input variable	Data type	Description
Torque	REAL	Torque/force value in the REAL format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-204: Input Variables of the Firmware Function "MX_fREAL_TorqueTo_DINT"

Return value	Data type	Description
MX_fREAL_TorqueTo_DINT	DINT	Returns a torque/force value in the fixed-point DINT format, according to the scaling which has been set

Tab. 13-205: Return Value of the Firmware Function "MX_fREAL_TorqueTo_DINT"

13.5.8 MX_fREAL_VelTo_DINT

The firmware function "MX_fREAL_VelTo_DINT" converts a velocity value in the REAL format into a fixed-point DINT value, according to the scaling which has been set.

If the axis reference is invalid, 0 is returned.

The function can be used for the virtual master axis / the master axis encoder and for the real axes "Axis1", "Axis2", etc.

MX_PLCOpen



Fig. 13-87: Firmware Function "MX_fREAL_VelTo_DINT"

Input variable	Data type	Description
Velocity	REAL	Velocity value in the REAL format
Axis	AXIS_REF	Axis address: The corresponding axis, e.g. "Axis1" from the global axis variables "AxisAddressing" of "MX_Base.lib", can be transmitted here

Tab. 13-206: Input Variables of the Firmware Function "MX_fREAL_VelTo_DINT"

Return value	Data type	Description
MX_fREAL_VelTo_DINT	DINT	Returns a velocity value in the fixed-point DINT format, according to the scaling which has been set

Tab. 13-207: Return Value of the Firmware Function "MX_fREAL_VelTo_DINT"

13.6 "Tools" - General functions

13.6.1 MX_fGetFreeTicks

The firmware function "MX_fGetFreeTicks" is used to determine the time of the last PLC time slice still available to the PLC. This PLC time slice is the fraction of a millisecond remaining after the drive controller and control tasks have been processed.

Timer ticks are returned that can be converted into microseconds with the function "MX_fHighResTimerTicks_to_us".



- When a freewheeling task is active, "0" is always returned in this case because this task works for the whole time slice.
- If "MX_fGetFreeTicks" is called in a task which has a clock slower than 1 ms, only the available time of the last ms time slice is displayed.
- When a cyclic task takes more time to be processed than the cycle time which was set, "MX_fGetFreeTicks" always returns the value "0".



Fig. 13-88: "MX_fGetFreeTicks" firmware function

Return value	Data type	Description
MX_fGetFreeTicks	DINT	Returns timer ticks. The timer ticks can be converted into μ s with the function "MX_fHighResTimerTicks_to_us"

Tab. 13-208: Return value of the firmware function "MX_fGetFreeTicks"

13.6.2 MX_fGetHighResTime

The firmware function "MX_fGetHighResTime" returns a growing timer value with high resolution.

Timer ticks are returned that can be converted into microseconds with the function "MX_fHighResTimerTicks_to_us".



Fig. 13-89: "MX_fGetHighResTime" firmware function

Return value	Data type	Description
MX_fGetHighResTime	UDINT	Returns timer ticks. The timer ticks can be converted into μ s with the function "MX_fHighResTimerTicks_to_us"

Tab. 13-209: Return value of the firmware function "MX_fGetHighResTime"

13.6.3 MX_fHighResTimerTicks_to_us

The firmware function "MX_fHighResTimerTicks_to_us" is used to convert timer ticks into microseconds and return the value as REAL.



Fig. 13-90: "MX_fHighResTimerTicks_to_us" firmware function

Input variable	Data type	Description
HighResTimerTicks	UDINT	Transfer values are timer ticks (see, for example, "MX_fGetHighResTime")

Tab. 13-210: Input variable of the "MX_fHighResTimerTicks_to_us" firmware function

Return value	Data type	Description
MX_fHighResTimerTicks_to_us	REAL	Returns the timer ticks in μ s

Tab. 13-211: Return value of the firmware function "MX_fHighResTimerTicks_to_us"

13.6.4 MX_IECTaskGetLoad

The function block "MX_IECTaskGetLoad" is used to activate the extended runtime measurement and displays information on the task load (see also Application Manual on IndraMotion MLD, chapter "Runtime Measurements").



As soon as a function block instance has been created in the project, the extended runtime measurement is activated in the drive.



The function block "MX_IECTaskGetLoad" can be called in the "own" task, but also in another (slower) task to save calculating time. The maximum task load is also measured, if the function block is not called in every cycle.

MX_PLCoPen

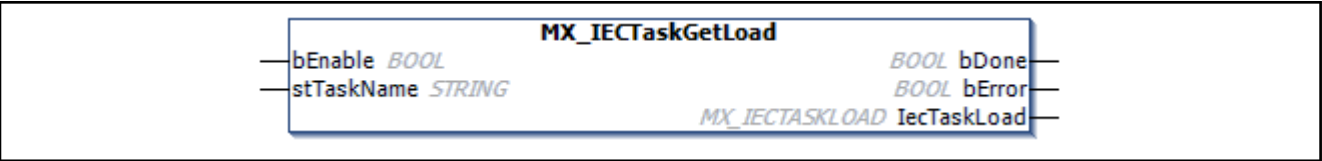


Fig. 13-91: "MX_IETaskGetLoad" firmware function block

Input variable	Data type	Description
bEnable	BOOL	At "bEnable"=TRUE the task information of the task named at the "stTaskName" input is returned.
stTaskName	STRING	At "stTaskName" indicate the name of the task the runtime information of which is to be displayed

Tab. 13-212: Input variables of the "MX_IETaskGetLoad" firmware function block

Output variable	Data type	Description
bDone	BOOL	"bDone" returns TRUE, if the task exists and the outputs are valid
bError	BOOL	The function block signals with "bError"=TRUE that the task was not found (wrong task name)
IecTaskLoad	MX_IETASKLOAD	Runtime information of the task Elements in "IecTaskLoad": "rLoad" displays the load of the task in the last cycle in percent, with reference to the cycle time "rLoadMax" displays the maximum load of the task in percent, with reference to the cycle time "rFreeTime" displays the remaining unused calculating time of the task up to its next start, from the last cycle in µs "rMinFreeTime" displays the minimum remaining unused calculating time of the task up to its next start in µs

Tab. 13-213: Output variables of the "MX_IETaskGetLoad" firmware function block

13.6.5 MX_PositionLoopEventInfo



The function block "MX_PositionLoopEventInfo" is for exclusive use by Bosch Rexroth staff.

14 Basic libraries

14.1 MX_CommonTypes

14.1.1 Introduction

The "MX_CommonTypes" library contains general data types and structures. ("MX_CommonTypes" was derived from "RIL_CommonTypes" for "IndraMotion MLD").

Only the data types and structures relevant to the user are described below.

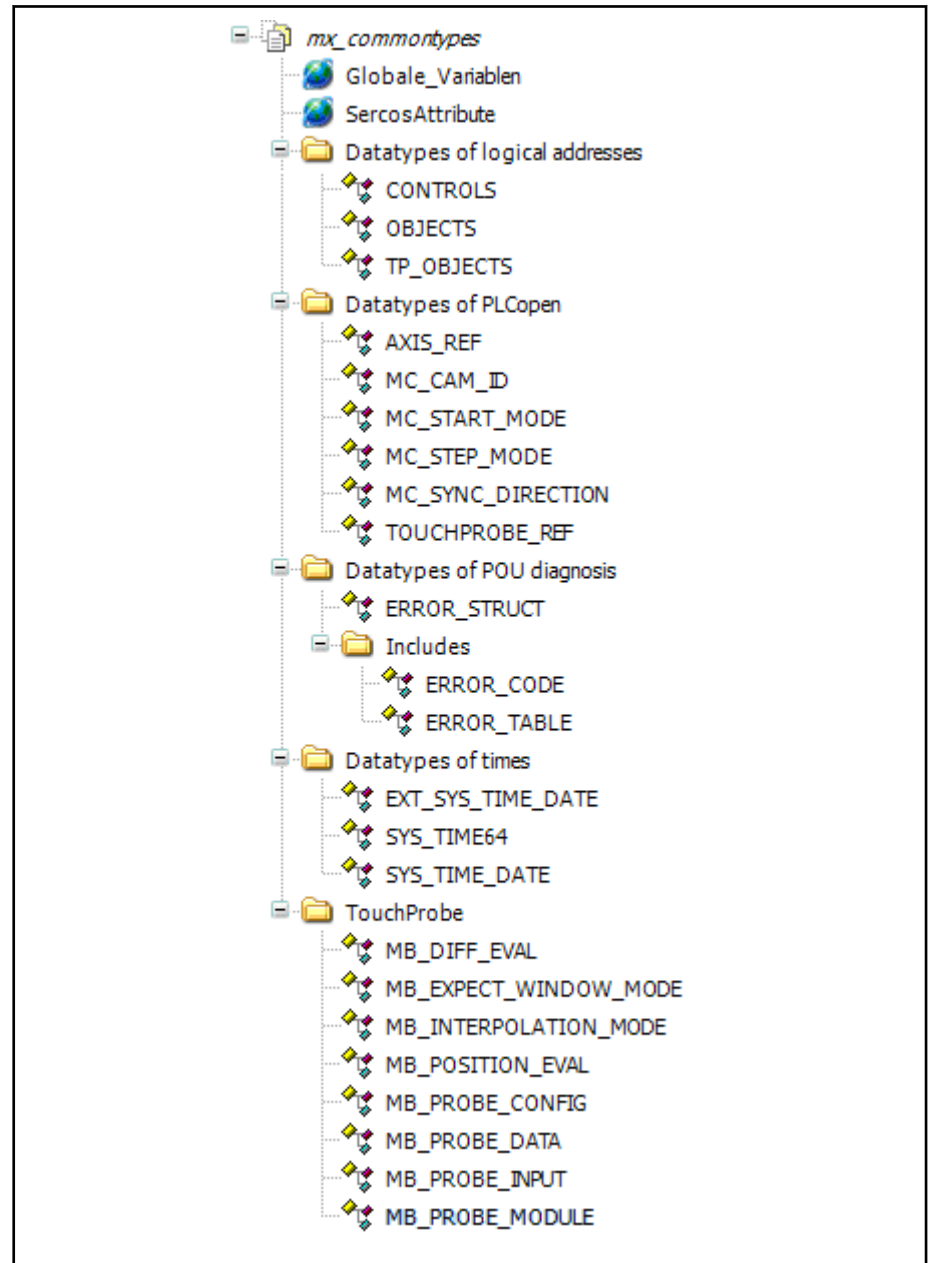


Fig. 14-1: Structure of the "MX_CommonTypes" library

Basic libraries

14.1.2 Global variables

IL_NULL

Brief description The global variable "IL_NULL" is a "Zero pointer" with data type "POINTER TO BYTE".

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/Global_Variables

Tab. 14-1: IL_NULL reference table

Type definition

```
VAR_GLOBAL CONSTANT
  IL_NULL: POINTER TO BYTE := 0;
END_VAR
```

14.1.3 Sercos attributes

See "[Sercos attribute](#)"

14.1.4 Data types of logical addresses

CONTROLS

Brief description Using the "CONTROLS" data type, the local control can be addressed via the IndraMotion function blocks. The local control can be accessed in two ways: either via the number "0" or via its own address "LOCAL_CTRL".

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of logical addresses

Tab. 14-2: CONTROLS reference table

Type definition

```
// Enumeration of valid logical control addresses
TYPE CONTROLS :
(
  LOCAL_CTRL := 0,    // Local control (default)
);
END_TYPE
```

OBJECTS

Brief description The enumeration of available axes contains all objects of an IndraMotion group that can be accessed by IndraMotion function blocks via "AXIS_REF".

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of logical addresses

Tab. 14-3: OBJECTS reference table

Type definition

```
// not specified object
NO_OBJECT := 0,    // no object specified
```

```
// drives 1..19
  AXIS_1 := 1,           // drive 1
  // ...
  AXIS_19 := 19,        // drive 19
// virtual master axis
  VMA_1 := 20,           // virtual master axis 1
  VMA_2 := 21,           // virtual master axis 2
  VMA_INT := 22,         // internal virtual master axis
// real master axis
  RMA_1 := 23,           // real master axis 1
  RMA_2 := 24,           // real master axis 2
  RMA_INT := 25,         // internal real master axis
// link axis
  MA_LINK_1 := 26,       // link axis 1
  MA_LINK_2 := 27,       // link axis 2
  MA_LINK_INT := 28,     // internal link axis
```



The number "0", "NO_OBJECT" (no object specified) is used for disabled axes.



The names and numbers are assigned upon creation of the respective axis or the object in the IndraWorks Suite.

TP_OBJECTS

Brief description The enumeration of the available probes (Touch Probe) contains objects of an IndraMotion group that can be accessed by probe function blocks.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of logical addresses

Tab. 14-4: TP_OBJECTS reference table

Type definition

```
(* Enumeration of valid logical object addresses *)
TYPE TP_OBJECTS :
(
  (* not specified object *)
  NO_TOUCHPROBE := 0, (* no object specified *)

  (* Touch Probe 1..100, realized on controller *)
  TOUCHPROBE_1 := 01, (* Touch Probe 1 *)
  (* ... *)
  TOUCHPROBE_100 := 100 (* Touch Probe 100 *)

  (* Touch Probe 1001..1002, realized on drive *)
  TOUCHPROBE_DRIVE_1 := 1001 (* Touch Probe Drive 1 *)
  TOUCHPROBE_DRIVE_2 := 1002 (* Touch Probe Drive 2 *)
);
END_TYPE
```



The names and the numbers are assigned upon manual creation of the respective probe in the IndraWorks Suite.



The number of probes that can be realized depends on the control type and the used firmware version.

14.1.5 Data types of PLCopen

AXIS_REF

Brief description The structure contains information on the desired control and the object controlled by it (axis, position switch, oscilloscope...).

Basic libraries

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of PLCopen

Tab. 14-5: *AXIS_REF* reference table
Type definition

```
(* Structure of PLCopen element AXIS_REF. The AXIS_REF is a structure that contains *)
(* information on the corresponding object. *)
(* It is used as a VAR_IN_OUT CONSTANT in all Motion Control Function Blocks. *)
TYPE AXIS_REF :
STRUCT
  CntrlNo: CONTROLS := LOCAL_CNTRL; (* control reference number *)
  AxisNo: OBJECTS := NO_OBJECT; (* Steuerungsnummer, Init: lokale Steuerung *)
  (* object reference number *)
  (* Achsreferenznummer, Init: kein Objekt *)
END_STRUCT
END_TYPE
```

Functional description

From version 14VRS, the information is transmitted to the respective PLCopen function blocks using **VAR_IN_OUT CONSTANT**.

Die *AXIS_REF* structure contains two 16-bit address elements:

- the control address (element of [chapter "CONTROLS" on page 186](#)) and
- the object address (element of [chapter "OBJECTS" on page 186](#))

The elements can be addressed using either the logical or the physical address. In case of a logical address, the element points to a parameter structure with a physical address (e.g. Ethernet address of the control, Sercos address of the drive etc.).



Information was transferred up to 13VRS via **VAR_IN_OUT**.
In 14VRS and higher, information is transferred via **VAR_IN_OUT CONSTANT**.

MC_START_MODE

Brief description This parameter defines the type of automatic dynamic synchronization in case of phase synchronization.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of PLCopen

Tab. 14-6: MC_START_MODE reference table

Type definition

```

TYPE MC_START_MODE :
(
  ABSOLUTE      := 0, (* synchronization with absolute angle reference, *)
                  (* only for first design *)
  RELATIVE      := 1, (* synchronization to synchronous velocity, *)
                  (* only for first design *)
  ABSOLUTE_RAMPIN := 2, (* synchronization with absolute angle reference, *)
                  (* for every design *)
  RELATIVE_RAMPIN := 3 (* synchronization to synchronous velocity, *)
                  (* for every design *)
);
END_TYPE
    
```

Absolute synchronization The slave axis position in the zero-crossing of the master axis position is unambiguously defined and does not change with every call of MB_MotionProfile/ MC_CamIn.

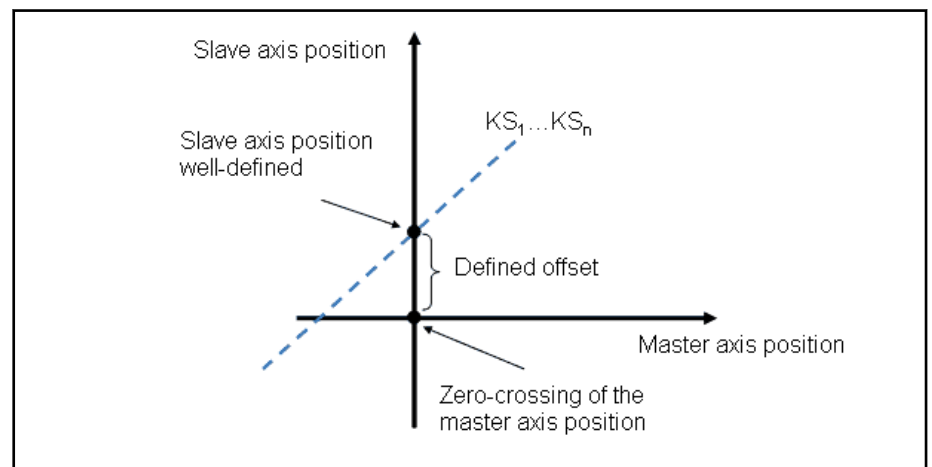


Fig. 14-2: Absolute synchronization

Relative synchronization The slave axis position in the zero-crossing of the master axis position is arbitrary and changes with every call of MB_MotionProfile/ MC_CamIn.

Basic libraries

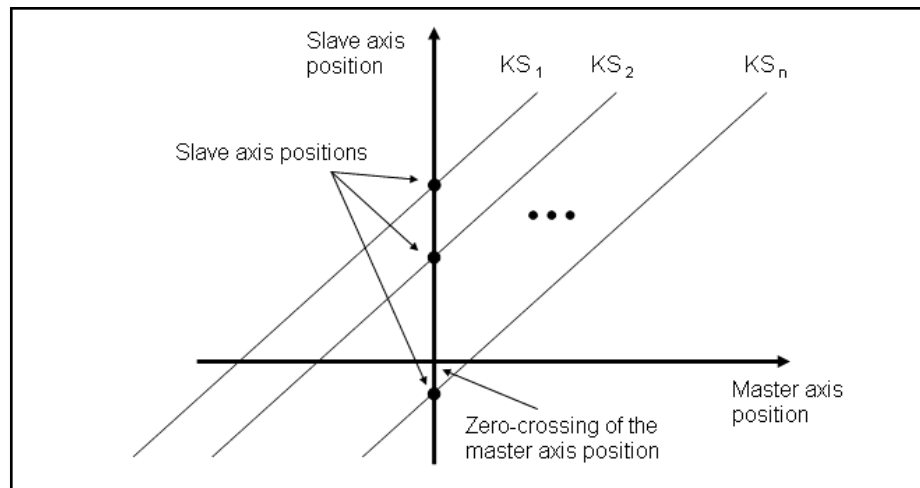


Fig. 14-3: Relative synchronization

"Hard" synchronization

- StartMode= 0: Absolute synchronization, "hard changeover" in case of a change of profile,
- StartMode= 1: Relative synchronization, "hard changeover" in case of a change of profile

Switching takes place after the master axis passed the master axis switching position (P-0-0700).

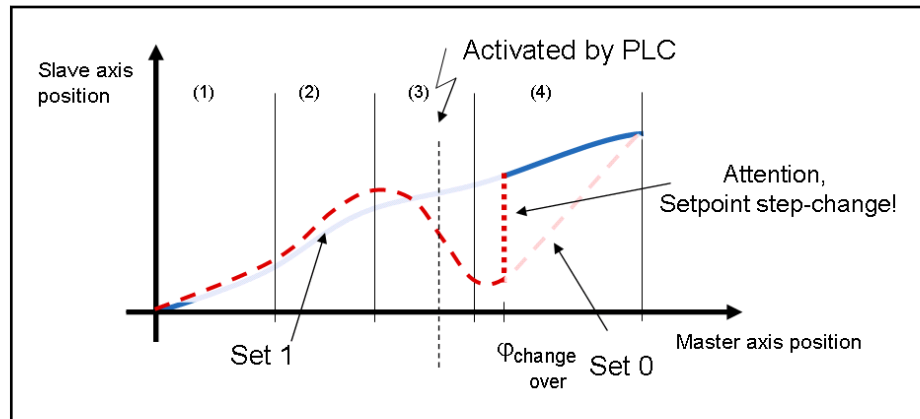


Fig. 14-4: Hard changeover

"Soft" synchronization

- StartMode= 2: Absolute synchronization, "soft changeover" in case of a change of profile,
- StartMode= 3: Relative synchronization, "soft changeover" in case of a change of profile.

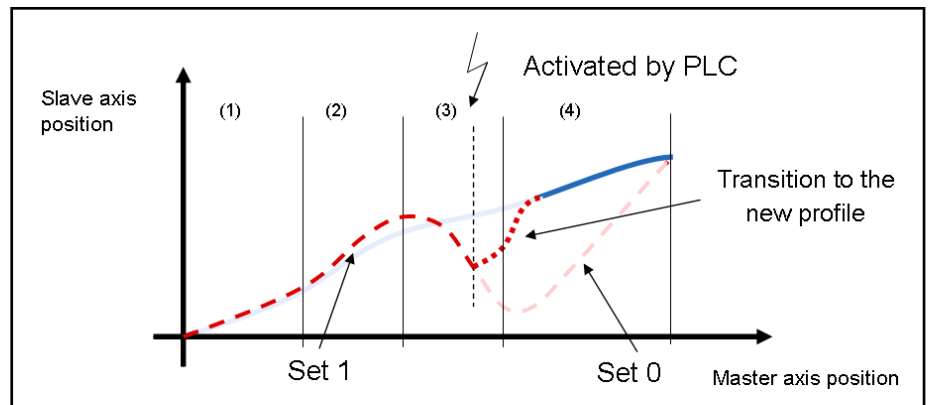


Fig. 14-5: Soft changeover

Switching takes place immediately after the function block started. The system does not wait for the master axis to pass the master axis switching position (P-0-0700).



The "soft synchronization" requires the calculation on the control.

MC_SYNC_DIRECTION

Brief description

For modulo axes, the motion of the axis can be in positive or negative direction.

The direction of synchronization defines the direction of the axis motion.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G	MX_CommonTypes/data types of
PLC basic package	PLCopen

Tab. 14-7: MC_SYNC_DIRECTION reference table

Type definition

```

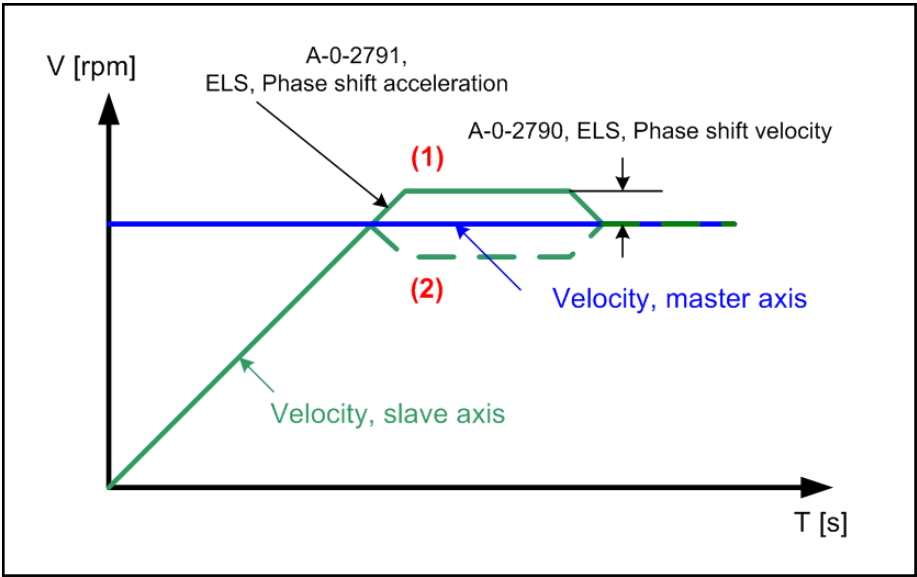
TYPE MC_SYNC_DIRECTION :
(
    SHORTEST_WAY    := 0,    (* kürzester Weg /shortest distance *)
    CATCH_UP        := 1,    (* immer vorwärts / always forward *)
    SLOW_DOWN       := 2
);
END_TYPE
    
```



If the shortest distance to the absolute synchronization is less than the "Synchronization window" for modulo format, the axis travels the shortest distance and ignores the specified synchronization direction.

Basic libraries

Example MLC:



- ① The slave axis follows the master axis
- ② The master axis follows the slave axis

Fig. 14-6: Transition from velocity-synchronous to position-synchronous

In the example, the slave axis first accelerates to the same velocity as the master axis. Subsequently, the phase difference between the master and the slave axis has to be reduced.

The parameters P-0-0142 (Synchronization acceleration/deceleration) and P-0-0143 (Synchronization velocity) take effect on the synchronization process.

The process is completed when both parameters run velocity-synchronously and position-synchronously.

TOUCHPROBE_REF

Brief description The structure contains information on the desired control and the object (probe) controlled by it.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of PLCopen

Tab. 14-8: TOUCHPROBE_REF reference table

Type definition

```
(* Structure of PLCopen element TOUCHPROBE_REF. The TOUCHPROBE_REF is a structure that *)
(* contains information on the corresponding object. *)

TYPE TOUCHPROBE_REF :
STRUCT
  CntrlNo:      CONTROLS    := LOCAL_CNTRL;  (* control reference number *)
              (* control number, Init: local control *)
END_STRUCT
END_TYPE
```

Functional description From version 14VRS, the information is transmitted to the respective PLCopen function blocks using **VAR_IN_OUT CONSTANT**.

The TOUCHPROBE_REF structure contains a 16-bit address element of the control address (element of [chapter "CONTROLS" on page 186](#)).



Information was transferred up to 13VRS via **VAR_IN_OUT**.

In 14VRS and higher, information is transferred via **VAR_IN_OUT CONSTANT**.

14.1.6 "ERROR_STRUCT" structure

The "ERROR_STRUCT" structure is used for the "ErrorIdent" output. "ERROR_STRUCT" contains information describing the error details.

The elements of the "ERROR_STRUCT" structure are:

- **Table**

"Table" is of type "ERROR_TABLE". The value of the "Table" element contains information on which error table [INDRV_TABLE, SERCOS_TABLE or MLD_TABLE] has to be used to decode the error code of the "Additional1" element.

- **Additional1**

"Additional1" is of type "DWORD". The "Additional1" element contains the error code for precisely specifying the error that has occurred.

- **Additional2**

"Additional2" is of type "DWORD". "Additional2" contains additional information (if available)

14.1.7 Data types to return the date and the absolute time specifications in IndraMotion function blocks

Returning the date, absolute time specifications, introduction and overview

Three structures are available to return the date and other absolute time specifications:

- [SYS_TIME64, page 194](#), system date and system time in microseconds since 01/01/1970, 00:00.
- [EXT_SYS_TIME_DATE, page 194](#), system date and system time in microseconds since 01/01/1970, 00:00, additional data: year, month, day, hours, minutes, seconds, milliseconds and day of the week are calculated from the system time.
- [EXT_SYS_TIME_DATE, page 194](#), year, month, day, hours, minutes, seconds, milliseconds and day of the week are calculated from the system in microseconds since 01/01/1970, 00:00.

Basic libraries

SYS_TIME64

Brief description Structure to return the system time in microseconds since 01/01/1970, 00:00.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of times

Tab. 14-9: SYS_TIME64 reference table

Type definition

```
(* Structure of system time - Microseconds since 1970 *)
TYPE SYS_TIME64 :
STRUCT
  LowValue:      DWORD;      (* Low value [us]      *)
  HighValue:     DWORD;      (* High value [us]     *)
END_STRUCT
END_TYPE
```

EXT_SYS_TIME_DATE

Brief description Structure to return the system time in microseconds since 01/01/1970, 00:00 and additional data: year, month, day, hours, minutes, seconds, milliseconds and day of the week are calculated from the system time.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of times

Tab. 14-10: EXT_SYS_TIME_DATE reference table

Type definition

```
(* Extended structure of system time *)
TYPE EXT_SYS_TIME_DATE :
STRUCT
  LowValue: DWORD;      (* Low value [us]      *)
  HighValue: DWORD;     (* High value [us]     *)
  Year : UINT;          (* Year                *)
  Month : UINT;         (* Month               *)
  Day : UINT;           (* Day                 *)
  Hour : UINT;          (* Hour                *)
  Minute : UINT;        (* Minute              *)
  Second : UINT;        (* Second              *)
  Milliseconds : UINT;  (* Milliseconds        *)
  DayOfWeek: UINT;      (* Day of week         *)
END_STRUCT
END_TYPE
```

SYS_TIME_DATE

Brief description Structure to return the data: year, month, day, hours, minutes, seconds, milliseconds and day of the week calculated from the system time in microseconds since 01/01/1970, 00:00.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of times

Tab. 14-11: SYS_TIME_DATE reference table

Type definition

```
(* Structure of system time *)
TYPE SYS_TIME_DATE :
STRUCT
    Year : UINT;          (* Year *)
    Month : UINT;         (* Month *)
    Day : UINT;           (* Day *)
    Hour : UINT;          (* Hour *)
    Minute : UINT;        (* Minute *)
    Second : UINT;        (* Second *)
    Milliseconds : UINT;  (* Milliseconds *)
    DayOfWeek: UINT;      (* Day of week *)
END_STRUCT
END_TYPE
```

14.1.8 Data types to support probe functions

Data types to support probe functions, general information

The section contains data types supporting the probe functions during processing in the drive as well as in the control:

- [chapter "MB_DIFF_EVAL" on page 196](#), setting the methods for difference measurement
- [chapter "MB_EXPECT_WINDOW_MODE" on page 196](#), configuring and activating the expectation window
- [chapter "MB_INTERPOLATION_MODE" on page 197](#), activating the interpolation
- [chapter "MB_POSITION_EVAL" on page 198](#), selecting and activating the trigger signal
- [chapter "MB_PROBE_DATA" on page 199](#), selecting the probe reference signal source
- [chapter "MB_PROBE_MODULE" on page 200](#), selecting the trigger source for the probe
- [chapter "MB_PROBE_INPUT" on page 199](#), selecting the desired input on the selected module

The data of a probe function are summarized in the following structure:

- [chapter "MB_PROBE_CONFIG" on page 198](#)

Basic libraries

MB_DIFF_EVAL

Brief description The "MB_DIFF_EVAL" enumeration type supports the setting of the methods for difference measurement.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/TouchProbe

Tab. 14-12: MB_DIFF_EVAL reference table

Type definition

```
TYPE MB_DIFF_EVAL :  
(  
  DIFF_DISABLED      := 0, (* diff.measurement is disabled *)  
  DIFF_STANDARD MODE := 1, (* diff. last two measured values*)  
  DIFF_NEG_TO_NEG_EDGE:= 2, (* neg. edge to neg. edge diff. *)  
  DIFF_POS_TO_NEG_EDGE:= 3, (* pos. edge to neg. edge diff. *)  
  DIFF_NEG_TO_POS_EDGE:= 4, (* neg. edge to pos. edge diff. *)  
  DIFF_POS_TO_POS_EDGE:= 5 (* pos. edge to pos. edge diff. *)  
);  
END_TYPE
```

Functional description The following settings are supported:

- 0 - Difference measurement is switched off, difference signals for ML_TouchProbe and/or ML_TouchProbe_Continuous are not calculated.
- 1 - Difference between the last two measured values, depending on the activated edge.
Example: Both edges are activated, the difference pos/neg edge, then neg/pos edge, pos/neg edge is registered.
- In the following figure, the extended methods 2..5 of the difference measurement are displayed:

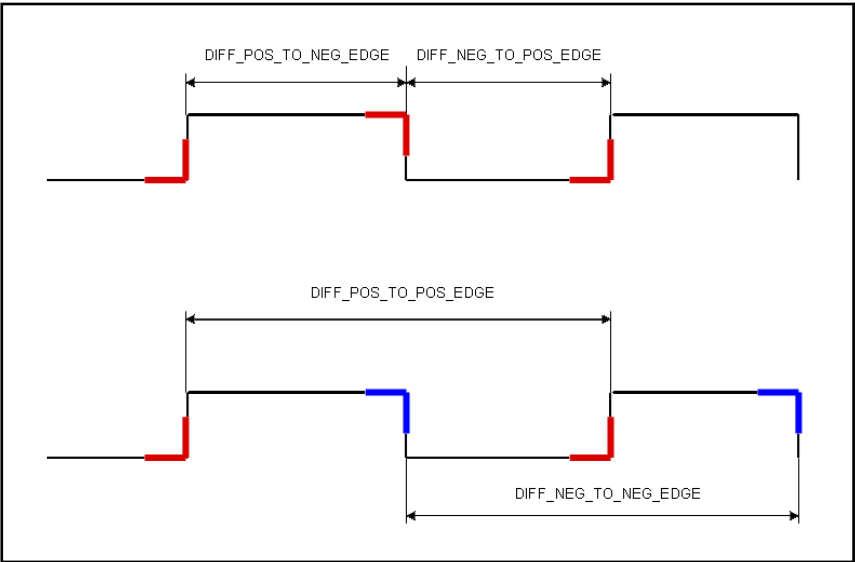


Fig. 14-7: Difference measurement methods

MB_EXPECT_WINDOW_MODE

Brief description The "MB_EXPECT_WINDOW_MODE" enumeration type supports the configuration and activation of the expectation window.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/TouchProbe

Tab. 14-13: MB_EXPECT_WINDOW_MODE reference table

Type definition

```

TYPE MB_EXPECT_WINDOW_MODE :
(
  EXPECT_WINDOW_DISABLED:= 0, (* expectation window disabled *)
  EXPECT_WINDOW_ABS      := 1, (* expectation window absolute *)
  EXPECT_WINDOW_REL      := 2, (* expectation window relative *)
                                (* to "ExpectWindowSetPoint" *)
);
END_TYPE

```

Functional description

The following decisions can be made separately for the positive and the negative edge of the probe value of an expectation window.

If the user reconsiders and choses to use an expectation window, the expectation window boundaries can be defined in two different ways:

1. Expectation window disabled:
The respective probe signal edge is unfiltered
2. Expectation window absolute defined:
The starting and end position of the expectation window are to be specified as absolute. The center point value is not decisive in this case.
3. Expectation window relative defined:
The center point value of the expectation window are to be specified as absolute. The starting and end position specify the distance to the center point value.

MB_INTERPOLATION_MODE

Brief description

The "MB_INTERPOLATION_MODE" enumeration type supports the activation of the interpolation.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/TouchProbe

Tab. 14-14: MB_INTERPOLATION_MODE reference table

Type definition

```

TYPE MB_INTERPOLATION_MODE:
(
  NO_INTERPOLATION      := 0, (* no interpolation *)
  LINEAR_INTERPOLATION  := 1, (* linear interpolation *)
  CUBIC_3_INTERPOLATION := 2, (* cubic interpolation with 3 setpoints *)
  CUBIC_4_INTERPOLATION := 3, (* cubic interpolation with 4 setpoints *)
);
END_TYPE

```

Basic libraries

MB_POSITION_EVAL

Brief description The "MB_POSITION_EVAL" enumeration type supports the selection and activation of the trigger signal.

Using these enumeration values, it can be selected whether triggering is to take place to the positive or the negative edge or to both edges.



The default value "NULL" blocks the trigger action!

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G	MX_CommonTypes/TouchProbe
PLC basic package	

Tab. 14-15: MB_POSITION_EVAL reference table

Type definition

```

TYPE MB_POSITION_EVAL :
(
  POSITION_EVAL_DISABLED:= 0, (* disabled *)
  POSITION_POS_EDGE      := 1, (* positive edge is evaluated *)
  POSITION_NEG_EDGE      := 2, (* negative edge is evaluated *)
  POSITION_POS_NEG_EDGE  := 3, (* pos.and neg. edge are evaluated *)
);
END_TYPE

```

MB_PROBE_CONFIG

Brief description The "MB_PROBE_CONFIG" structure is used for registering the data of a probe function.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G	MX_CommonTypes/TouchProbe
PLC basic package	

Tab. 14-16: MB_PROBE_CONFIG reference table

Type definition

```

TYPE MB_PROBE_CONFIG :
STRUCT
  ModuloValueDrive      :REAL;          (* Modulo value *)
  ProcessingFormat      :BOOL;          (* Weighting position, abs./modulo *)
  ProbePositionEval     :MB_POSITION_EVAL; (* Enabling trigger signal, def.: Off *)
                                     (* pos., neg., both edges *)
  DifferenzMeasurment   :MB_DIFF_EVAL;  (* difference measurement method, def.: Off *)
  ExpectWindowPosMode   :MB_EXPECT_WINDOW_MODE; (* Expectation window mode pos. edge *)
                                     (* def.: Off, absolute / relative to the setpoint *)
  ExpectWindowNegMode   :MB_EXPECT_WINDOW_MODE; (* Expectation window mode neg. edge *)
                                     (* def.: Off, absolute / relative to the setpoint *)
  ExpectWindowPosStart  :REAL;          (* Start pos. expectation window pos. edge *)
  ExpectWindowPosEnd    :REAL;          (* End pos. expectation window pos. edge *)
  ExpectWindowNegStart  :REAL;          (* Start pos. expectation window neg. edge *)
  ExpectWindowNegEnd    :REAL;          (* End pos. expectation window neg. edge *)
END_STRUCT
END_TYPE

```

"MB_PROBE_CONFIG" uses the following data types

- [chapter "MB_DIFF_EVAL" on page 196](#)
- [chapter "MB_EXPECT_WINDOW_MODE" on page 196](#)
- [chapter "MB_EXPECT_WINDOW_MODE" on page 196](#)

MB_PROBE_DATA

Brief description The "MB_PROBE_CONFIG" enumeration type supports the selection of the probe reference signal source.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/TouchProbe

Tab. 14-17: MB_PROBE_DATA reference table

Type definition

```

TYPE MB_PROBE_DATA :
(
    TP_ACT_POS_FEEDBACK1      := 0, // S-0-0051
    TP_ACT_POS_FEEDBACK2      := 1, // S-0-0053
    TP_ACT_POSIN_ACT_CYCLE    := 2, // P-0-0753
    TP_SYNC_CMD_POS           := 3, // P-0-0778
    TP_PROBE_TIME              := 4, // Time measurement
    TP_ACT_MEASURING_ENC_POS   := 5, // P-0-0052
    TP_RES_MASTER_AXIS_POS     := 6, // P-0-0775
    TP_EFF_MASTER_AXIS_POS     := 7, // P-0-0776
    TP_CAM_SHAFT_PROF_ACCESS_ANGLE := 8, // P-0-0227
);
END_TYPE
    
```

MB_PROBE_INPUT

Brief description The "MB_PROBE_INPUT" enumeration type supports the selection of the desired input on the selected module.



The inputs 1..8 are available for the OnBoard module.

The inputs 1..16 are the trigger source of the fast I/O modules.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/TouchProbe

Tab. 14-18: MB_PROBE_INPUT reference table

Type definition

```

TYPE MB_PROBE_INPUT :
(
    D_INPUT_1  := 1, (* digital input 1 *)
    D_INPUT_2  := 2, (* digital input 2 *)
    D_INPUT_3  := 3, (* digital input 3 *)
    D_INPUT_4  := 4, (* digital input 4 *)
    D_INPUT_5  := 5, (* digital input 5 *)
    D_INPUT_6  := 6, (* digital input 6 *)
    D_INPUT_7  := 7, (* digital input 7 *)
    D_INPUT_8  := 8, (* digital input 8 *)
    D_INPUT_9  := 9, (* digital input 9 *)
    D_INPUT_10 := 10, (* digital input 10 *)
    D_INPUT_11 := 11, (* digital input 11 *)
    D_INPUT_12 := 12, (* digital input 12 *)
    D_INPUT_13 := 13, (* digital input 13 *)
    D_INPUT_14 := 14, (* digital input 14 *)
    D_INPUT_15 := 15, (* digital input 15 *)
    D_INPUT_16 := 16, (* digital input 16 *)
);
END_TYPE
    
```

Basic libraries

MB_PROBE_MODULE

Brief description The "MB_PROBE_MODULE" enumeration type supports the selection of the trigger source for the probe.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/TouchProbe

Tab. 14-19: MB_PROBE_MODULE reference table

Type definition

```

TYPE MB_PROBE_MODULE :
(
  ONBOARD_MODULE      := 0, (* Onboard module MLC,      Inputs 1...8          *)
  FUNCTION_MODULE_1 := 1, (* Function module 1 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
  FUNCTION_MODULE_2 := 2, (* Function module 2 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
  FUNCTION_MODULE_3 := 3, (* Function module 3 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
  FUNCTION_MODULE_4 := 4, (* Function module 4 MLC, Inputs 1..16 (CFL-01.1-E2, Fast IO) *)
);
END_TYPE

```

14.1.9 Data types to support cams, MotionProfiles and FlexProfiles

Data type to switch between cams, general information

Rexroth IndraDrive supports up to 8 cam tables which can be divided into the following groups:

- 4 cam tables with 3 to 1024 data points (P-0-0072, P-0-0092, P-0-0780, P-0-0781)
- 4 cam tables with 3 to 128 data points (P-0-0782, P-0-0783, P-0-0784, P-0-0785)

Cam table switching is possible at runtime. Select and switch via the function block MC_CamIn or via P-0-0088.

MC_CAM_ID

Brief description The enumeration type supports the selection of the cam table.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of PLCopen

Tab. 14-20: MC_CAM_ID reference table

Type definition

```

TYPE MC_CAM_ID :
(
  CAM_TABLE_1 := 1, (* Cam table 1 *)
  CAM_TABLE_2 := 2, (* Cam table 2 *)
  (* ... *)
  CAM_TABLE_8 := 8, (* Cam table 8 *)
);
END_TYPE

```

Functional description Cam table switching is possible at runtime. Select and switch via the function block MC_CamIn or via P-0-0088.

MC_STEP_MODE

Brief description The MC_STEP_MODE enumeration type describes the dependency of the slave axis position on the master axis during a motion step.

The dependency can be a functional dependency or a cam table.

Target/library assignment

Target	Library/folder
IndraMotion MLD 2G PLC basic package	MX_CommonTypes/data types of PLCopen

Tab. 14-21: MC_STEP_MODE reference table

Type definition

```

TYPE MC_STEP_MODE :
(
    REST_IN_REST_INCLINEDSINE      := 16#0000, (* R-R: Rest in rest, inclined sine curve *)
    REST_IN_REST_POLY5              := 16#0100, (* R-R: Rest in rest, 5th order polynomial *)
    REST_IN_VELOCITY_POLY5          := 16#0200, (* R-V: Rest in velocity, 5th order polynomial *)
    REST_IN_VELOCITY_POLY7          := 16#0300, (* R-V: Rest in velocity, 7th order polynomial *)
    VELOCITY_IN_REST_POLY5          := 16#0400, (* V-R: Velocity in rest, 5th order polynomial *)
    VELOCITY_IN_REST_POLY7          := 16#0500, (* V-R: Velocity in rest, 7th order polynomial *)
    CONSTANT_VELOCITY               := 16#0600, (* V: Constant velocity *)
    VELOCITY_IN_VELOCITY_POLY5      := 16#0700, (* V-V: Velocity in velocity, 5th order polynomial *)

    REST_IN_REST_POLY5_VLIM         := 16#1200, (* MotionProfile, FlexProfile *)

    (* Cam tables *)
    CAMTABLE_1                      := 1,        (* Cam table 1, P-0-0072 *)
    CAMTABLE_2                      := 2,        (* Cam table 2, P-0-0092 *)
    CAMTABLE_3                      := 3,        (* Cam table 3, P-0-0780 *)
    CAMTABLE_4                      := 4,        (* Cam table 4, P-0-0781 *)
    CAMTABLE_5                      := 5,        (* Cam table 5, P-0-0782 *)
    CAMTABLE_6                      := 6,        (* Cam table 6, P-0-0783 *)
    CAMTABLE_7                      := 7,        (* Cam table 7, P-0-0784 *)
    CAMTABLE_8                      := 8,        (* Cam table 8, P-0-0785 *)
);
END_TYPE
    
```

Functional description Different standard profiles or a cam table can be defined as mode of a step.

The following modes are available:

- R-R: Rest in rest, 5th order polynomial
- R-R: Rest in rest, inclined sine curve
- R-V: Rest in velocity, 5th order polynomial
- V-R: Velocity in rest, 5th order polynomial
- V: Constant velocity
- V-V: Velocity in velocity, 5th order polynomial
- Cam tables 1 to 8

R-R: Rest in rest For the "rest in rest" profile, there are ten profiles available. It is possible to choose between the standardized motion laws "5th order polynomial" and "inclined sine curve" ...

These profiles are defined by the distance and the master axis range. The final velocity of the step is determined by the following step.

The profile for the "5th order polynomial" is determined by the subsequent function:

Basic libraries

$$f(z) = 6z^5 - 15z^4 + 10z^3$$

Fig. 14-8: Rest in rest, 5th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-22: Boundary values for the motion law - 5th order polynomial

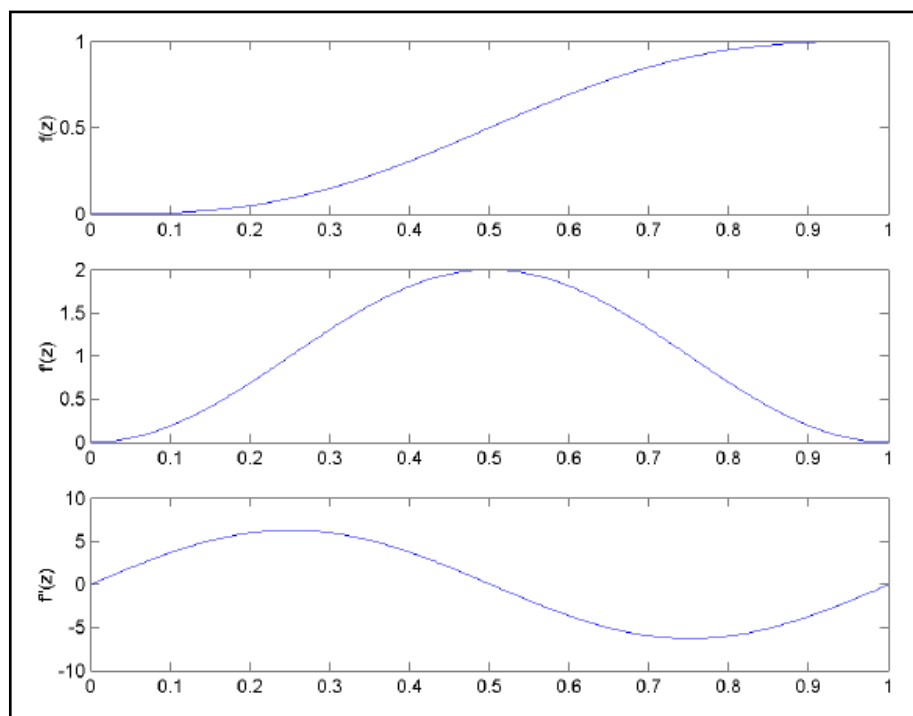


Fig. 14-9: Rest in rest, 5th order polynomial

The following formula applies to the "7th order polynomial":

$$f(z) = -20z^7 + 70z^6 - 84z^5 - 35z^4$$

Fig. 14-10: Rest in rest, 7th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	0	0

Tab. 14-23: Boundary values for the motion law - 7th order polynomial

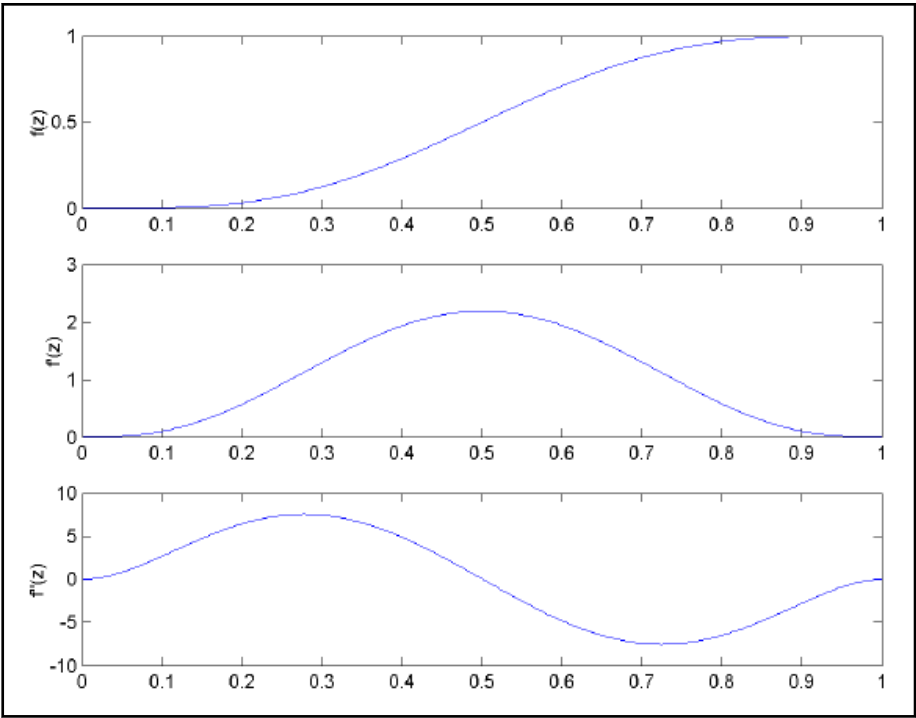


Fig. 14-11: Rest in rest, 7th order polynomial

The profile for the "Inclined sine curve" is determined by the subsequent function:

$$f(z) = z - \frac{1}{2\pi} \sin(2\pi z)$$

Fig. 14-12: Rest in rest, inclined sine curve

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-24: Boundary values for motion law - Inclined sine curve

Basic libraries

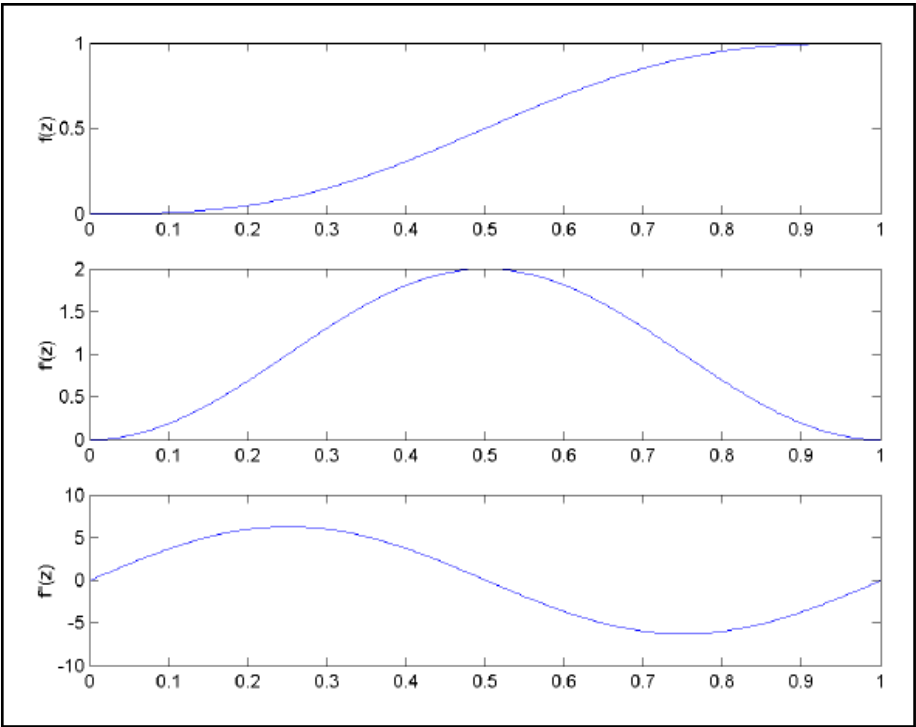


Fig. 14-13: Rest in rest, inclined sine curve (BESTEHORN)

The profile for the **"Sine curve (simple sinusoids)"** is determined by the subsequent function:

$$f(z) = \frac{1}{2} \left[1 - \cos(\pi z) \right]$$

Fig. 14-14: Rest in rest, sine curve

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	Automatically	Automatically
3rd order (jerk)	Automatically	Automatically

Tab. 14-25: Boundary values for the motion law sine curve

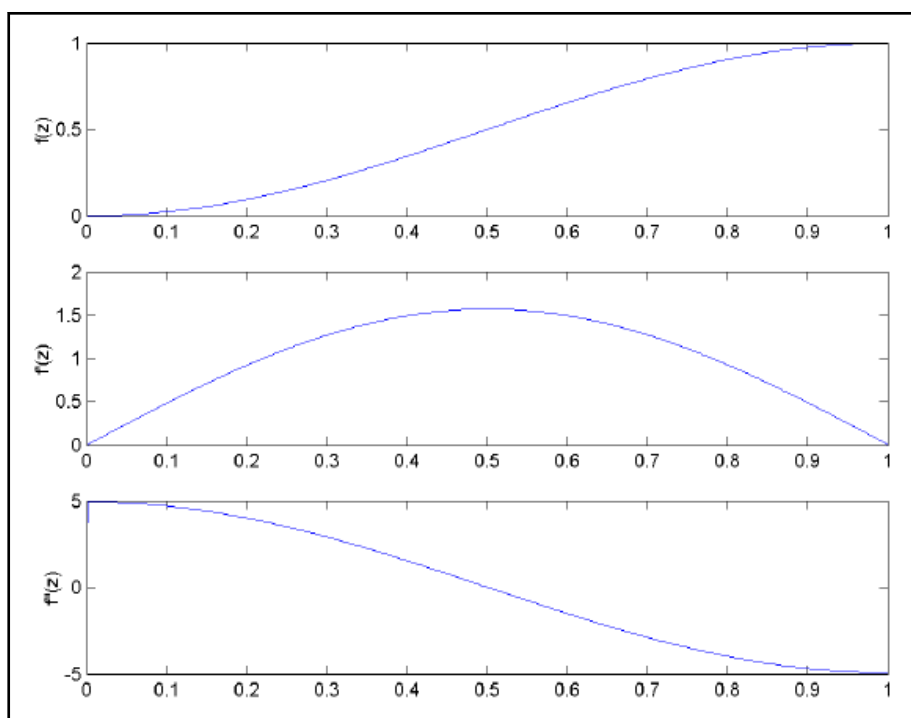


Fig. 14-15: Rest in rest, sine curve (simple sinusoids)

The profile for the "Linear rest" is determined by the subsequent function:

$$f(z) = 0$$

Fig. 14-16: Linear rest

Possible boundary values for the motion law:

Boundary value	$z=0$	$z=1$
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	0	0

Tab. 14-26: Boundary values for motion laws - Linear rest

Basic libraries

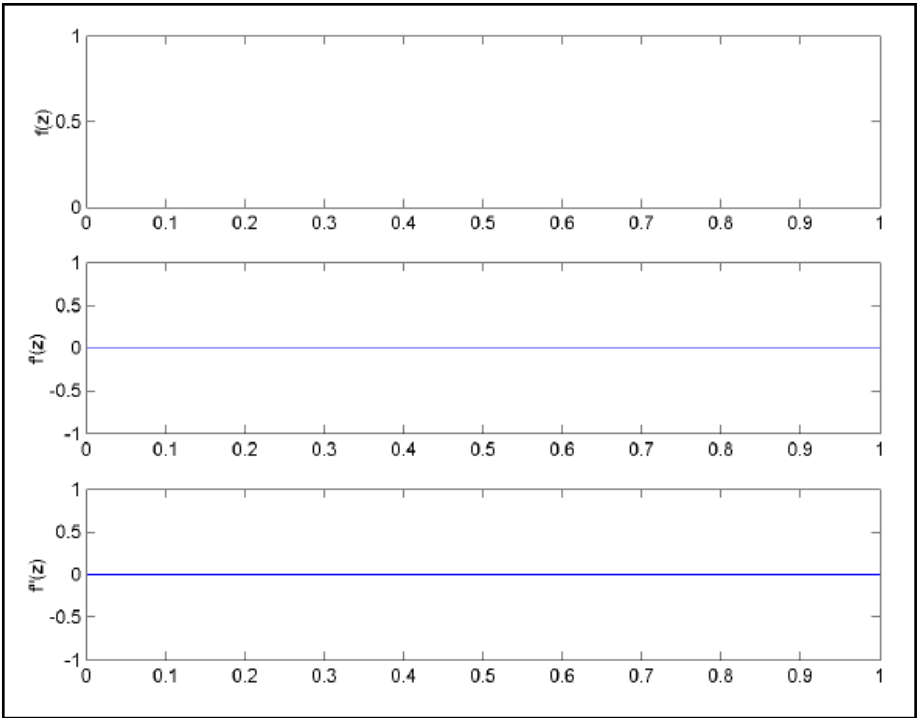


Fig. 14-17: Linear rest

The profile for the **"Gutman sinusoid"** is determined by the subsequent function:

$$f(z) = z - \frac{15}{32\pi} \sin(2\pi z) - \frac{1}{96\pi} \sin(6\pi z)$$

Fig. 14-18: Gutman sinusoid

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-27: Boundary values for motion law - Gutman sinusoid

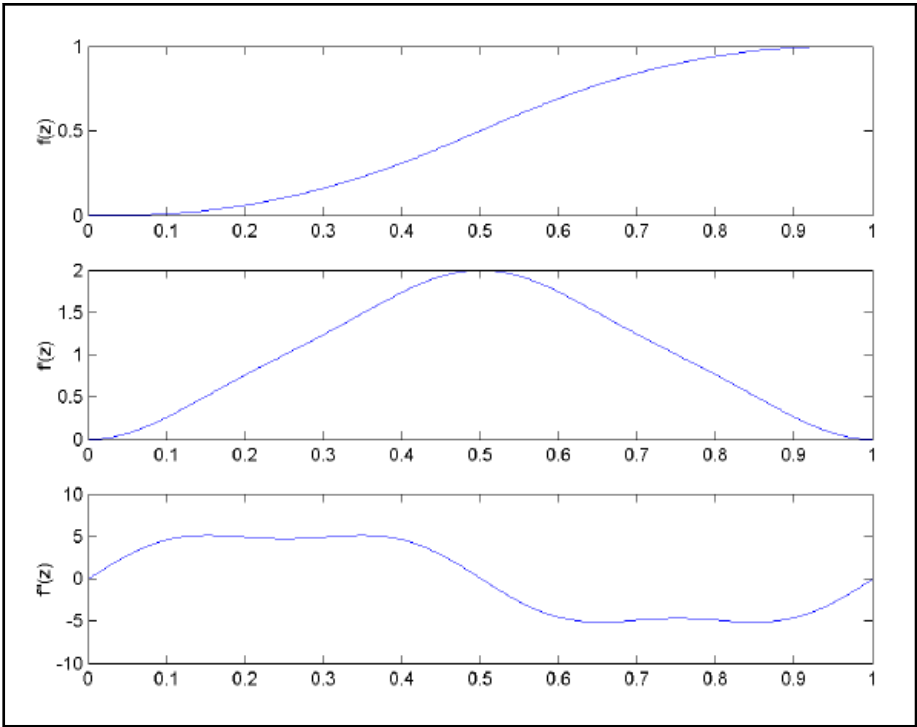


Fig. 14-19: Rest in rest, Gutman sinusoid

The profile for the **"Acceleration-optimum inclined sine curve"** is determined by the subsequent function:

$$f(z) = z - \frac{1}{2\pi} \sin(2\pi z)$$
$$g(z) = z - \frac{0.41}{2\pi} \sin(2\pi z)$$

Fig. 14-20: Acceleration-optimum inclined sine curve

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-28: Boundary values for the motion law - Acceleration-optimum inclined sine curve

Basic libraries

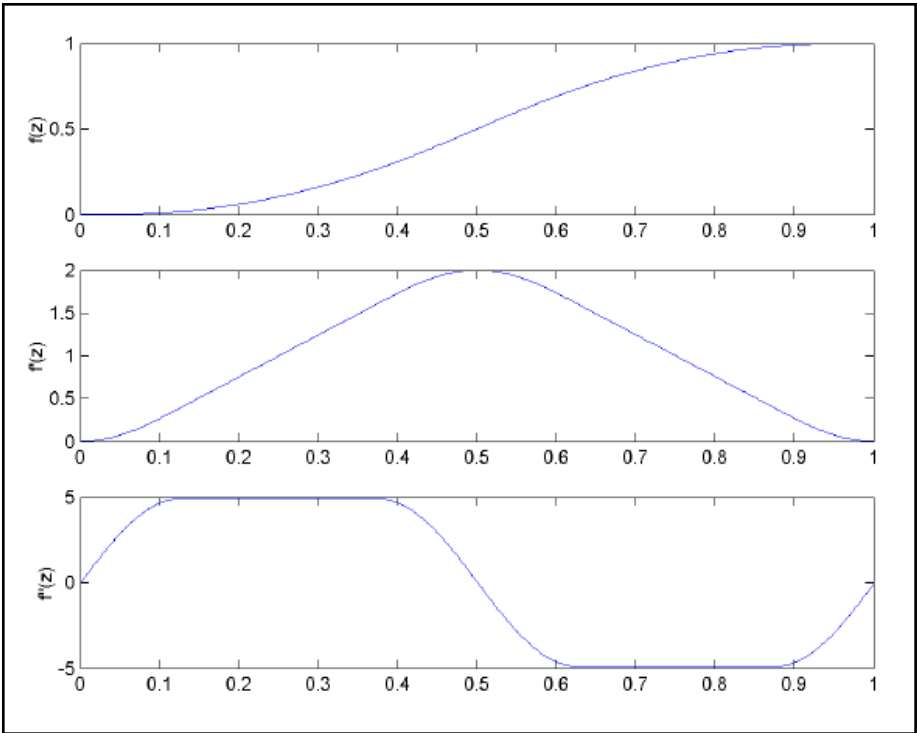


Fig. 14-21: *Rest in rest, acceleration-optimum inclined sine curve*

The profile for the **"Torque-optimum inclined sine curve"** is determined by the subsequent function:

$$f(z) = z - \frac{1}{2\pi} \sin(2\pi z)$$
$$g(z) = z - \frac{0.134}{2\pi} \sin(2\pi z)$$

Fig. 14-22: *Torque-optimum inclined sine curve*

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-29: *Boundary values for the motion law - Torque-optimum inclined sine curve*

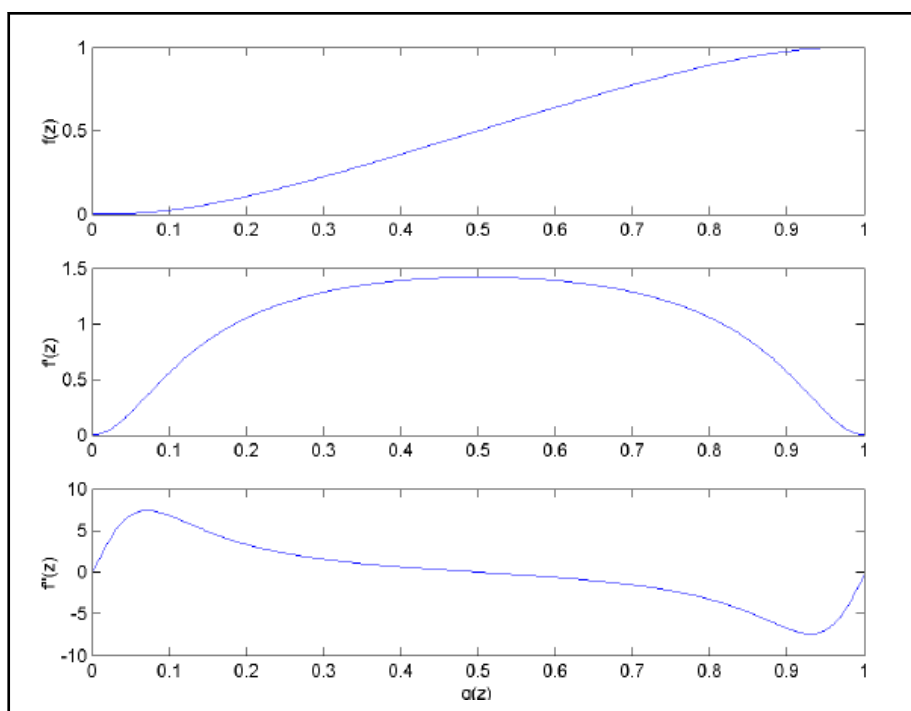


Fig. 14-23: Rest in rest, torque-optimum inclined sine curve

The profile for the **"Modified sinusoids"** is determined by the subsequent function:

$$\begin{aligned}
 0 \leq z \leq \frac{1}{8}: f(z) &= \frac{1}{4+\pi} \left[\pi z - \frac{1}{4} \sin(4\pi z) \right] \\
 \frac{1}{8} \leq z \leq \frac{7}{8}: f(z) &= \frac{1}{4+\pi} \left[2 + \pi z - \frac{4}{9} \sin\left(\frac{4\pi}{3}z + \frac{\pi}{3}\right) \right] \\
 \frac{7}{8} \leq z \leq 1: f(z) &= \frac{1}{4+\pi} \left[4 + \pi z - \frac{1}{4} \sin(4\pi z) \right]
 \end{aligned}$$

Fig. 14-24: Modified sinusoids

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-30: Boundary values for the motion law - Modified sinusoid

Basic libraries

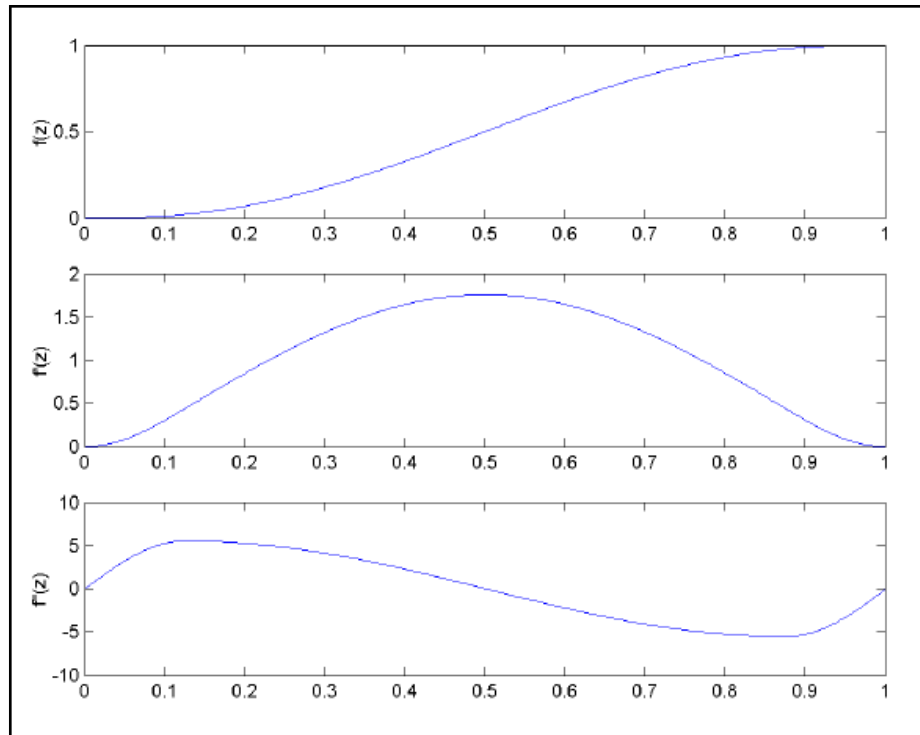


Fig. 14-25: Rest in rest, modified sinusoid

The profile for the **"Modified acceleration trapezoid"** is determined by the subsequent function:

$$\begin{aligned}
 0 \leq z \leq \frac{1}{8}: f(z) &= \frac{1}{2+\pi} \left[2z - \frac{1}{2\pi} \sin(4\pi z) \right] \\
 \frac{1}{8} \leq z \leq \frac{3}{8}: f(z) &= \frac{1}{2+\pi} \left[-\frac{1}{2\pi} + 2z + 4\pi \left(z + \frac{1}{8} \right)^2 \right] \\
 \frac{3}{8} \leq z \leq \frac{1}{2}: f(z) &= \frac{1}{2+\pi} \left[-\frac{\pi}{2} + 2(1+\pi)z - \frac{1}{2\pi} \sin(4\pi z - \pi) \right] \\
 \frac{1}{2} \leq z \leq 1: f(z) &= 1 - f(1-z)
 \end{aligned}$$

Fig. 14-26: Modified acceleration trapezoid

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	0
2nd order (acceleration)	0	0
3rd order (jerk)	Automatically	Automatically

Tab. 14-31: Boundary values for the motion law sine curve - Modified acceleration trapezoid

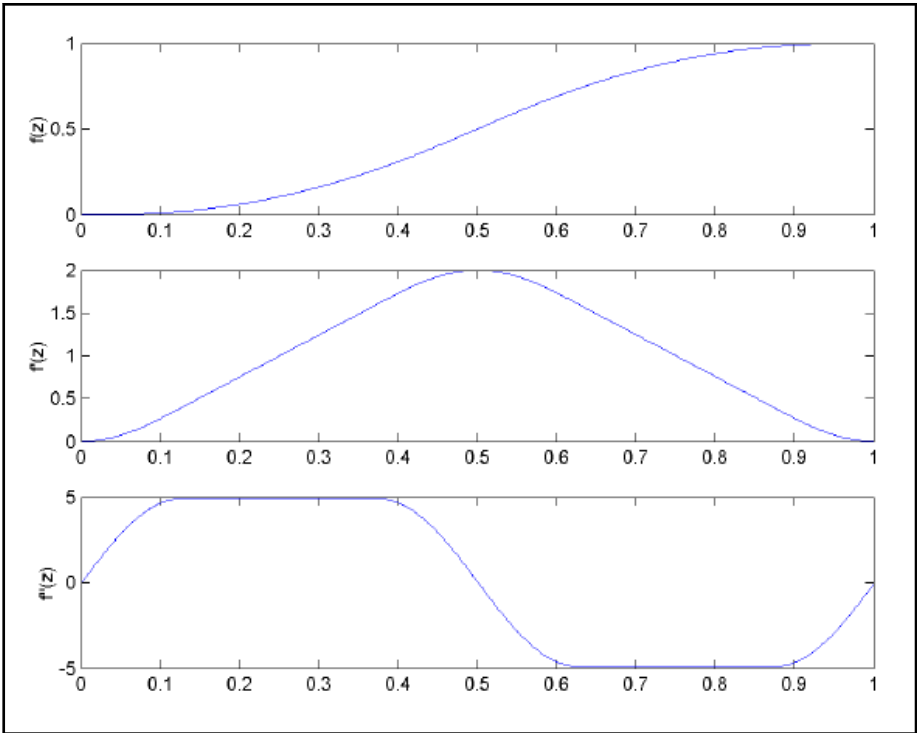


Fig. 14-27: Rest in rest, modified acceleration trapezoid

R-V: Rest in velocity

This profile is used for transition from standstill to a certain velocity.
The profile is calculated with a "5th order polynomial":

$$f(z) = A_5z^5 + A_4z^4 + A_3z^3$$

Fig. 14-28: 5th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	Selectable
2nd order (acceleration)	0	Selectable
3rd order (jerk)	Automatically	Automatically

Tab. 14-32: Boundary values for the motion law - 5th order polynomial

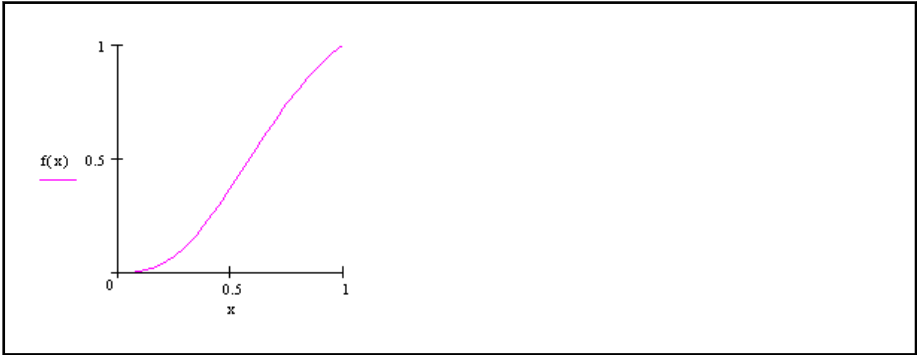


Fig. 14-29: Rest in velocity, 5th order polynomial

This profile is used for transition from standstill to a certain velocity.
The profile is calculated with a "7th order polynomial":

Basic libraries

$$f(z) = A_7z^7 + A_6z^6 + A_5z^5 + A_4z^4 + A_3z^3$$

Fig. 14-30: 7th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	0	Selectable
2nd order (acceleration)	0	Selectable
3rd order (jerk)	Automatically	Selectable

Tab. 14-33: Boundary values for the motion law - 7th order polynomial

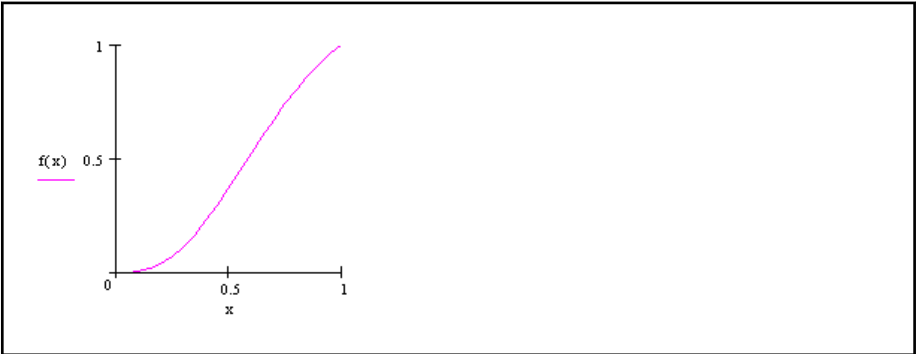


Fig. 14-31: Rest in velocity, 7th order polynomial

V-R: Velocity in rest

The profile is used for transition from a defined velocity to standstill.

The path traveled with this motion profile is defined by the distance for this motion step.

The profile is calculated with a "5th order polynomial":

$$f(z) = A_5z^5 + A_4z^4 + A_3z^3 + A_1z$$

Fig. 14-32: Velocity in rest, 5th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	0
2nd order (acceleration)	Selectable	0
3rd order (jerk)	Automatically	0

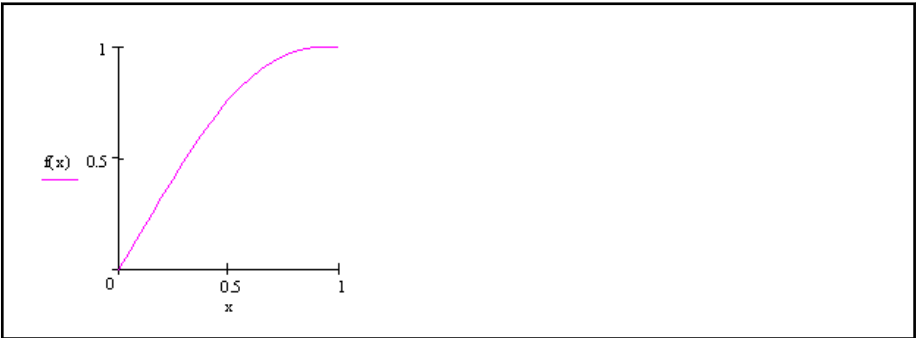


Fig. 14-33: Velocity in rest, 5th order polynomial

The profile is used for transition from a defined velocity to standstill.

The profile is calculated with a "7th order polynomial":

$$f(z) = A_7 z^7 + A_6 z^6 + A_5 z^5 + A_4 z^4 + A_3 z^3 + A_2 z^2 + A_1 z$$

Fig. 14-34: Rest in velocity, 7th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	0
2nd order (acceleration)	Selectable	0
3rd order (jerk)	Selectable	0

Tab. 14-34: Boundary values for the motion law - 7th order polynomial

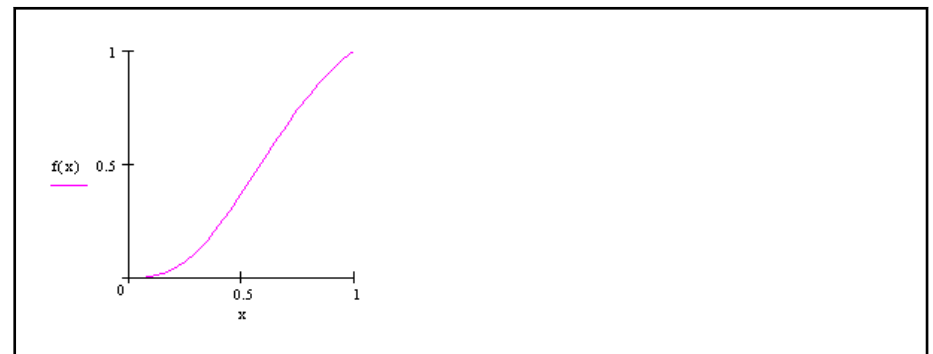


Fig. 14-35: Rest in velocity, 7th order polynomial

V: Constant velocity

The profile of this motion step is a "straight line". The velocity within this step is constant. The path traveled is defined by the distance for this motion step.

The standardized formula for the profile is:

$$f(z) = z.$$

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Automatically	Automatically
2nd order (acceleration)	Automatically	Automatically
3rd order (jerk)	Automatically	Automatically

Basic libraries

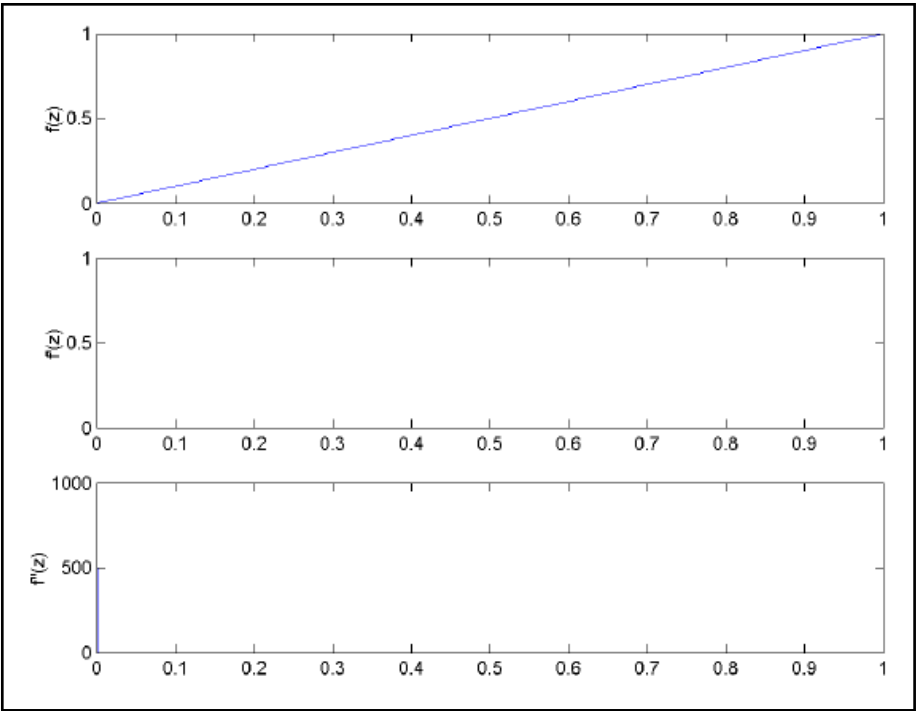


Fig. 14-36: *Constant velocity*

The slave axis velocity is determined from the values preset for this motion step:

- Master axis range
- Distance
- Master axis velocity

It is not necessary to enter a value for the slave axis velocity.

V-V: Velocity in velocity

The profile for the "5th order polynomial" is determined by the subsequent function:

$$f(z) = A_5z^5 + A_4z^4 + A_3z^3 + A_2z^2 + A_1z$$

Fig. 14-37: *5th order polynomial*

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	Selectable
2nd order (acceleration)	Selectable	Selectable
3rd order (jerk)	Automatically	Automatically

Tab. 14-35: *Boundary values for the motion law - 5th order polynomial*

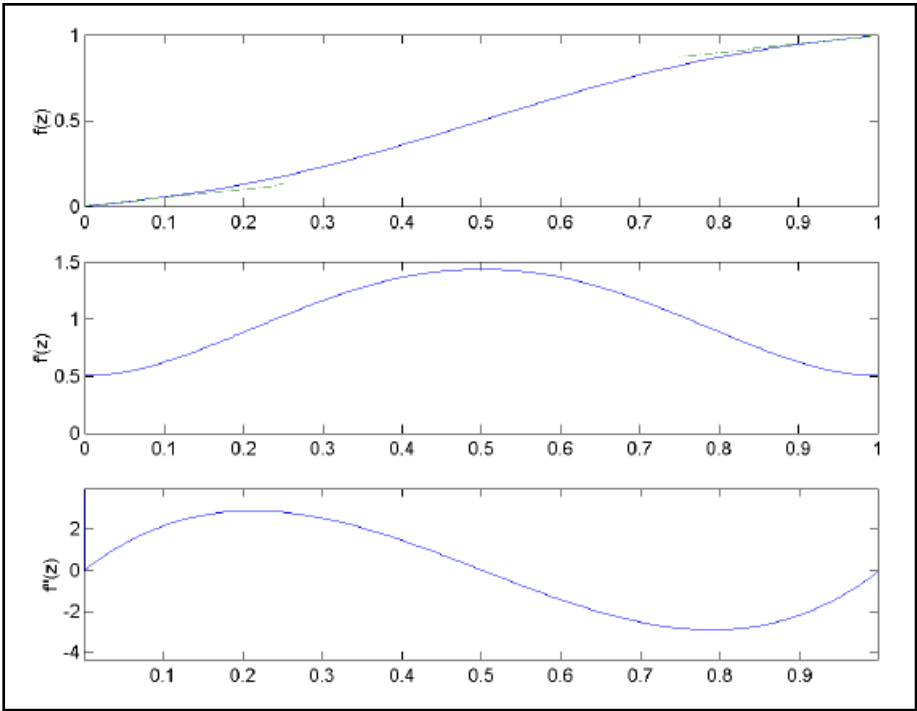


Fig. 14-38: Velocity in velocity, 5th order polynomial

The profile for the "7th order polynomial" is determined by the subsequent function:

$$f(z) = A_7 z^7 + A_6 z^6 + A_5 z^5 + A_4 z^4 + A_3 z^3 + A_2 z^2 + A_1 z$$

Fig. 14-39: 7th order polynomial

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	Selectable
2nd order (acceleration)	Selectable	Selectable
3rd order (jerk)	Selectable	Selectable

Tab. 14-36: Boundary values for the motion law - 7th order polynomial

Basic libraries

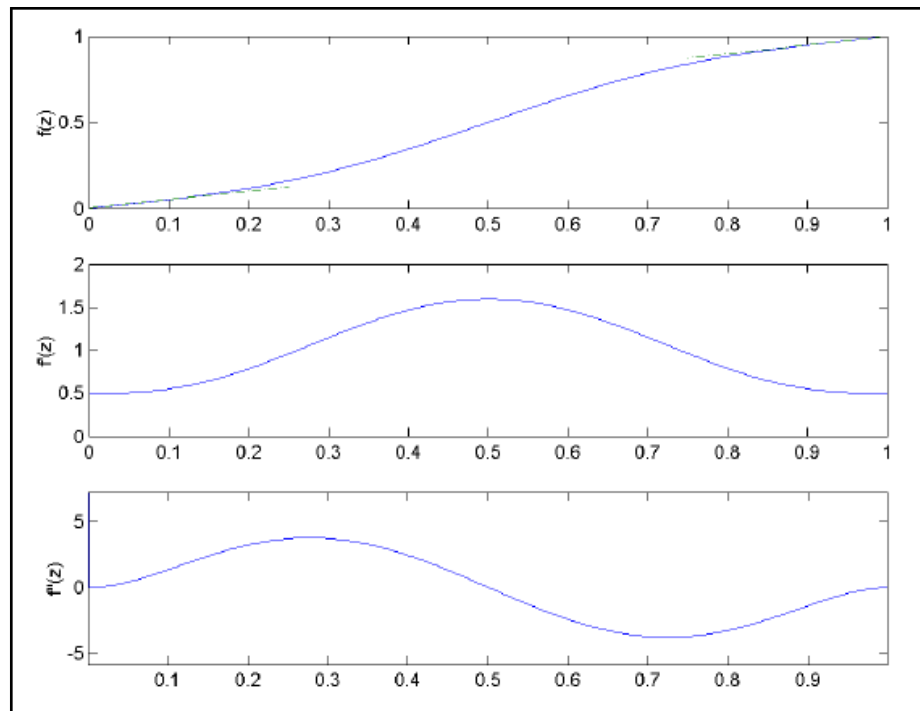


Fig. 14-40: Velocity in velocity, 7th order polynomial

The profile for the **"Modified sine curve"** is determined by the subsequent function:

$$\begin{aligned}
 0 \leq z \leq \frac{\lambda}{8}: f'(z) &= C_a \frac{\lambda}{2\pi} \left(z - \frac{\lambda}{2\pi} \sin \left(2\pi \frac{z}{\lambda} \right) \right) + f'(0)z \\
 \frac{\lambda}{8} \leq z \leq \lambda: f'(z) &= C_a \frac{\lambda}{2\pi} \left(z - \frac{9\lambda}{2\pi} \sin \left(2\pi \frac{z + \frac{\lambda}{2}}{3\lambda} \right) + \frac{4\lambda}{\pi} \right) + f'(0)z \\
 \lambda \leq z \leq 1 - \frac{1-\lambda}{4}: f'(z) &= C_b \frac{1-\lambda}{2\pi} \left(z - \frac{9(1-\lambda)}{2\pi} \sin \left(2\pi \frac{z + \frac{3-5\lambda}{3(1-\lambda)}}{3\lambda} \right) + \frac{4\lambda}{\pi} \right) \\
 &+ f'(1) + (f'(0) - f'(1)) \left(\frac{3}{4} - \frac{1}{\pi} \right) + \lambda \\
 1 - \frac{1-\lambda}{4} \leq z \leq 1: f'(z) &= -C_b \frac{1-\lambda}{2\pi} \left(z - \frac{1-\lambda}{2\pi} \sin \left(2\pi \frac{z + \lambda}{1-\lambda} \right) - 1 \right) \\
 &+ 1 + f'(1) + (z-1)
 \end{aligned}$$

Fig. 14-41: Modified sine curve

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	Selectable
2nd order (acceleration)	0	0
3rd order (jerk)	0	0

Tab. 14-37: Boundary values for the motion law - Modified sine curve

Extended motion laws

The extended motion laws are special motion laws. They differ in new degrees of freedom from the motion laws described in the VDI guidelines. While a standard motion law is clearly defined by specifying left and right boundary values, the motion laws described in this documentation support other different setting options, such as a required maximum velocity. In this case, the master axis range of the motion step does not have to be specified. This section results from the parameterization of the motion law and the specification of characteristic values. The acceleration-limited motion facilitates a distance specification, the specification of a starting and end velocity but not the specification of a master axis range. The master axis range and thus the duration of a motion step results from the values of a maximum acceleration and a maximum velocity that can also be specified.

	Distance	Master axis range	$v_{\max}$	$a_{\max}$	$j_{\max}$
Overshooting-free 5th order polynomial	Selectable	Selectable	Selectable	Automatically	Automatically
Velocity-limited 5th order polynomial	Selectable	Selectable	Selectable	Automatically	Automatically
Acceleration-limited motion	Selectable	Automatically	Selectable	Selectable	Automatically
Acceleration-limited sinusoid profile	Selectable	Automatically	Selectable	Selectable	Automatically
Jerk-limited motion	Selectable	Automatically	Selectable	Selectable	Selectable

Tab. 14-38: Extended motion laws

The profile for **"Velocity-limited 5th order polynomial"** can be described as follows:

The 5th order polynomial described in the VDI guideline does not support any verification of the maximum travel velocity. To solve this problem, the motion law is divided into three sections. The center section corresponds to a straight line with the required velocity. This section is surrounded by two 5th order polynomials. They ensure a smooth transition from starting or end velocity to the travel velocity. The intersection points between the first and second section as well as the second and third section are to be determined. These points are selected so that there is no inflection point for both polynomials.

Basic libraries

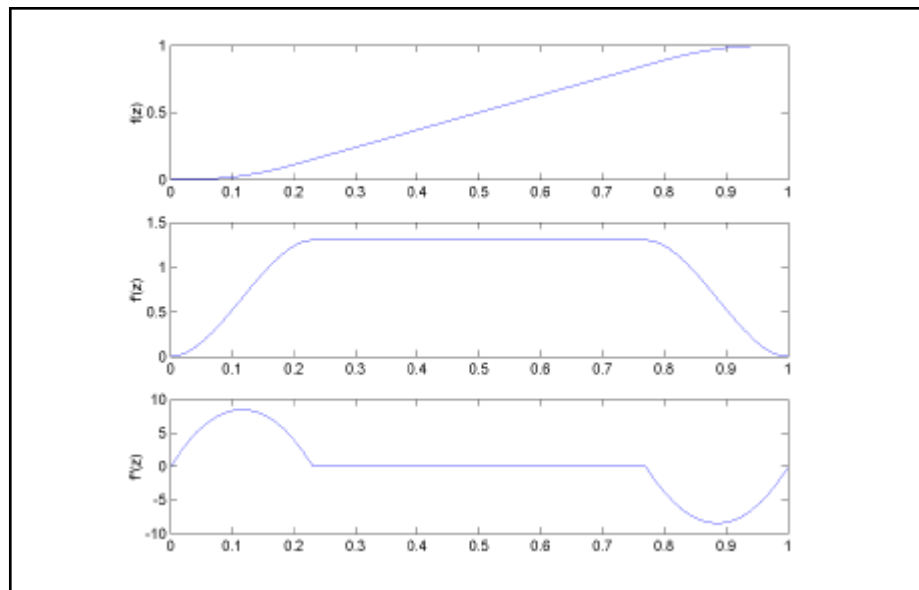


Fig. 14-42: Velocity-limited 5th order polynomial

The profile for **"Overshooting-free 5th order polynomial"** can be described as follows:

The shown motion law detects and prevents a potential overshooting of the polynomial. How it is changed depends on the combination of the standardized velocity boundary values. There are four cases. This motion law cannot be used for combinations in which the standardized initial velocity " v_0 " and the standardized end velocity " v_1 " have different signs. This would indicate that the motion path has a break point in a boundary point if the motion path would not be allowed to leave the standardized range of $0 \leq z \leq 1$ and $0 \leq f(z) \leq 1$.

- Case 1:
 $V_0 \leq 1$ and $V_1 \leq 1$

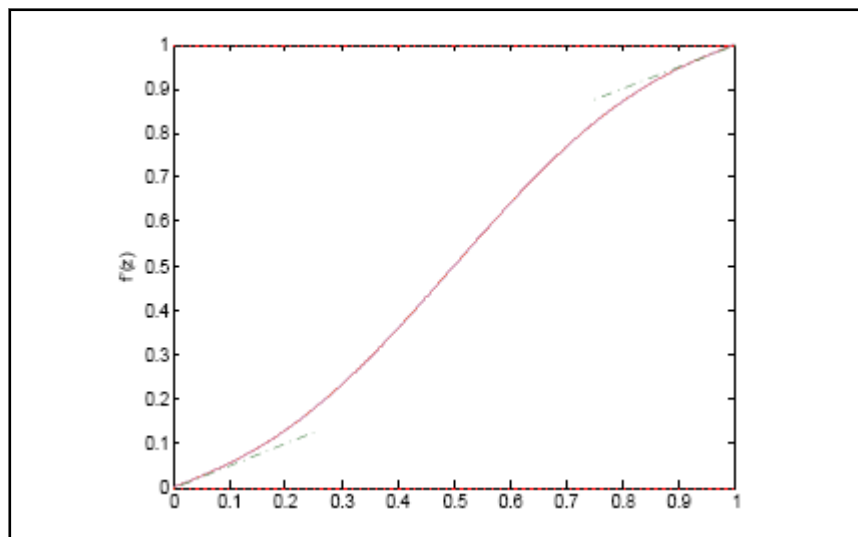


Fig. 14-43: Case 1

- Case 2:
 $V_0 \leq 1$ and $V_1 > 1$

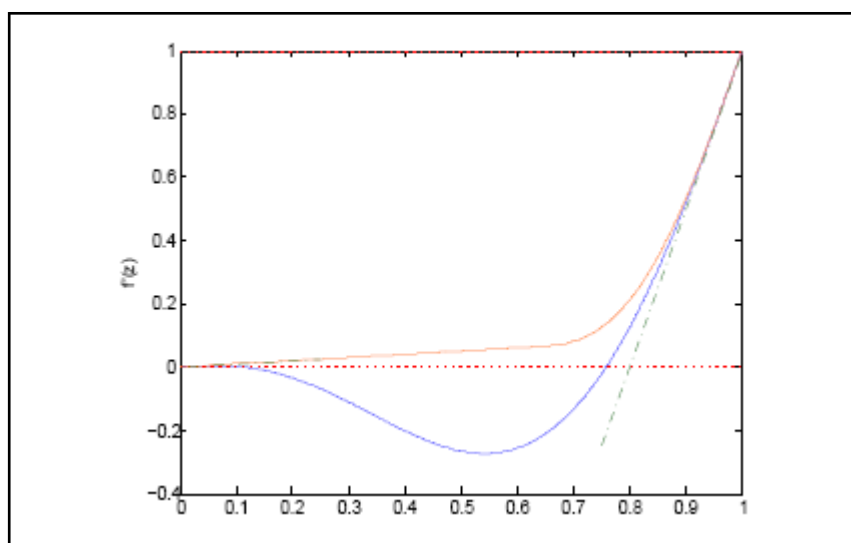


Fig. 14-44: Case 2

- Case 3:
 $V_0 > 1$ and $V_1 \leq 1$

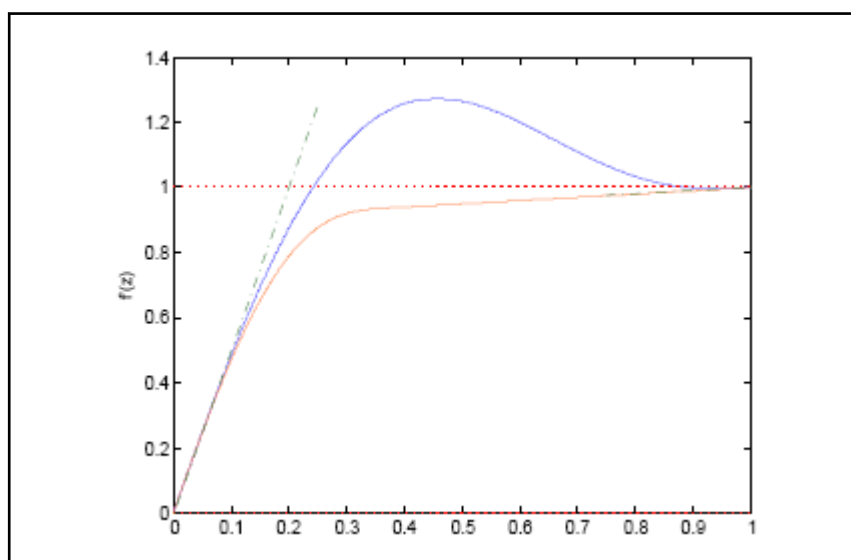


Fig. 14-45: Case 3

- Case 4:
 $V_0 > 1$ and $V_1 > 1$

Basic libraries

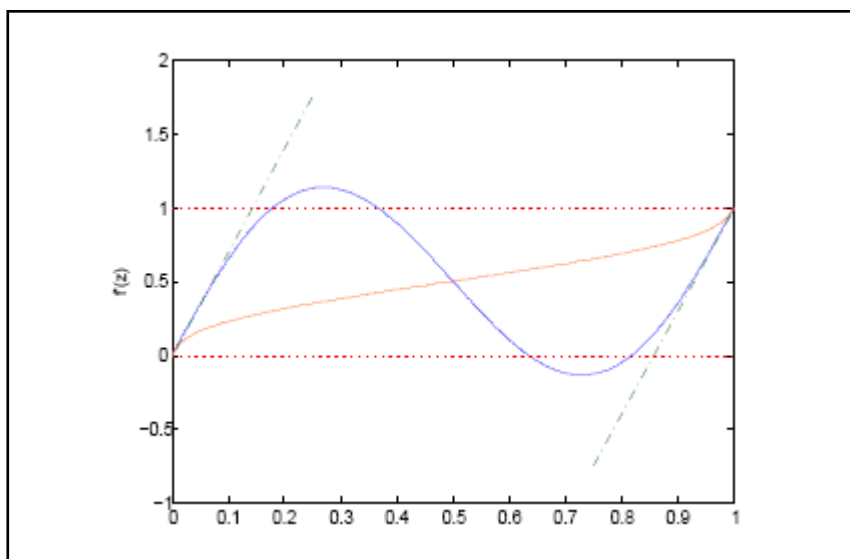


Fig. 14-46: Case 4

The profile for **"Acceleration-limited motion"** can be described as follows:

The simplest type of path interpolation is the trapezoid profile, also referred to as ramp profile. If a path distance "s" is to be traversed, the tool path can be divided into three phases. A first phase in which it is accelerated to a specified travel velocity with a specified acceleration value, a second phase with constant velocity and subsequently a third deceleration phase in which it is decelerated again at a specified deceleration. The path interpolation calculates the starting and end times t_1 , t_2 and t_3 or the individual traversing times T_1 , T_2 and T_3 from these specified acceleration values a_1 and a_3 as well as the desired velocity v_m and the traveled path "s". Furthermore, the required starting velocity v_0 as well as the end velocity v_3 are taken into consideration in this variant.

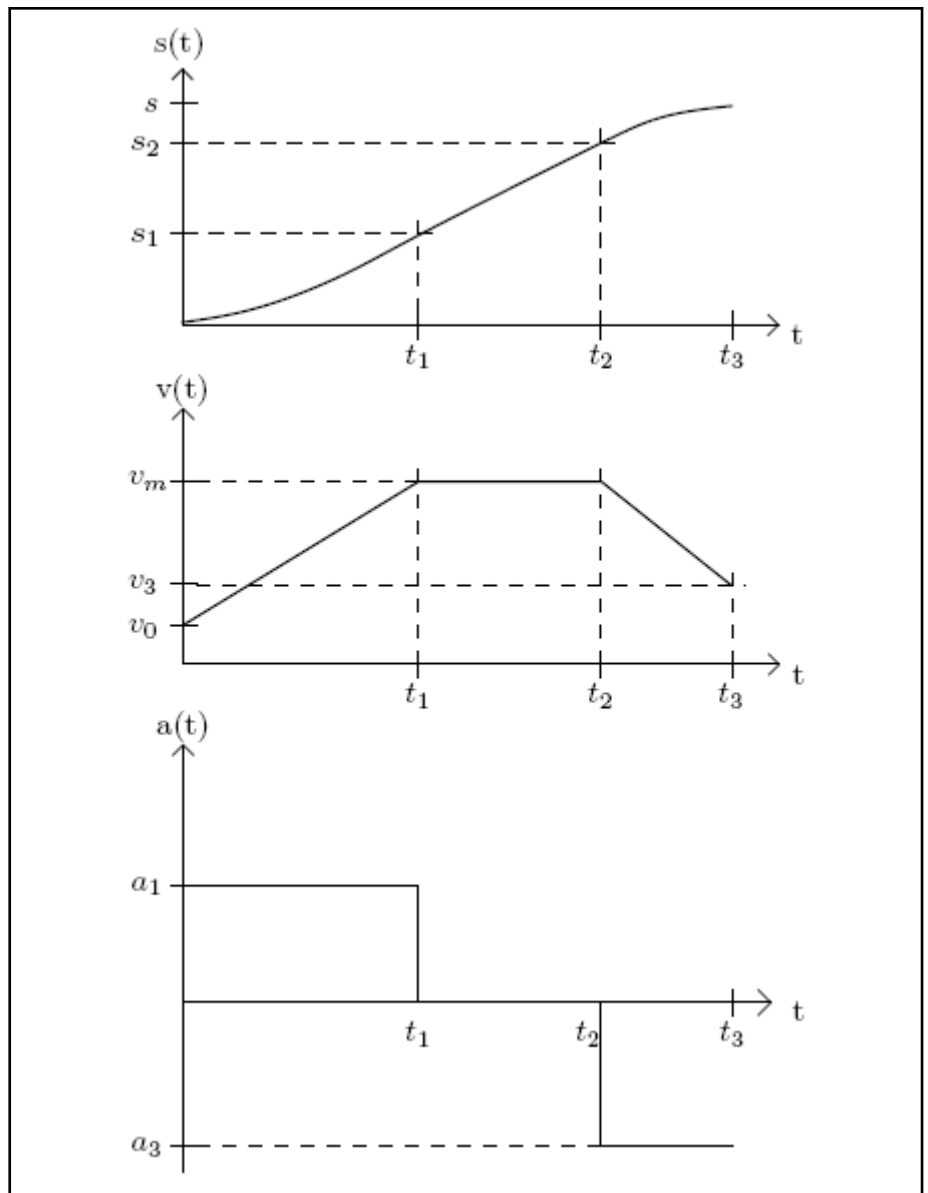


Fig. 14-47: Acceleration-limited motion

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	Selectable
2nd order (acceleration)	Automatically	Automatically
3rd order (jerk)	Automatically	Automatically

Tab. 14-39: Boundary values for motion law - Acceleration-limited motion

The profile for **"Acceleration-limited motion with sinusoid profile"** can be described as follows:

The starting and end velocities can be unequal to 0 in the sinusoid profile. It can also have different acceleration variables a_1 and a_3 for the acceleration and deceleration phase. Furthermore, special cases are also taken into consideration if the starting velocity v_0 is already larger than the desired travel velocity v_m . The same applies to the end velocity v_3 .

Basic libraries

Possible boundary values for the motion law:

Boundary value	z=0	z=1
1st order (velocity)	Selectable	Selectable
2nd order (acceleration)	Automatically	Automatically
3rd order (jerk)	Automatically	Automatically

Tab. 14-40: Boundary values for the motion law - Acceleration-limited motion with sinusoid profile

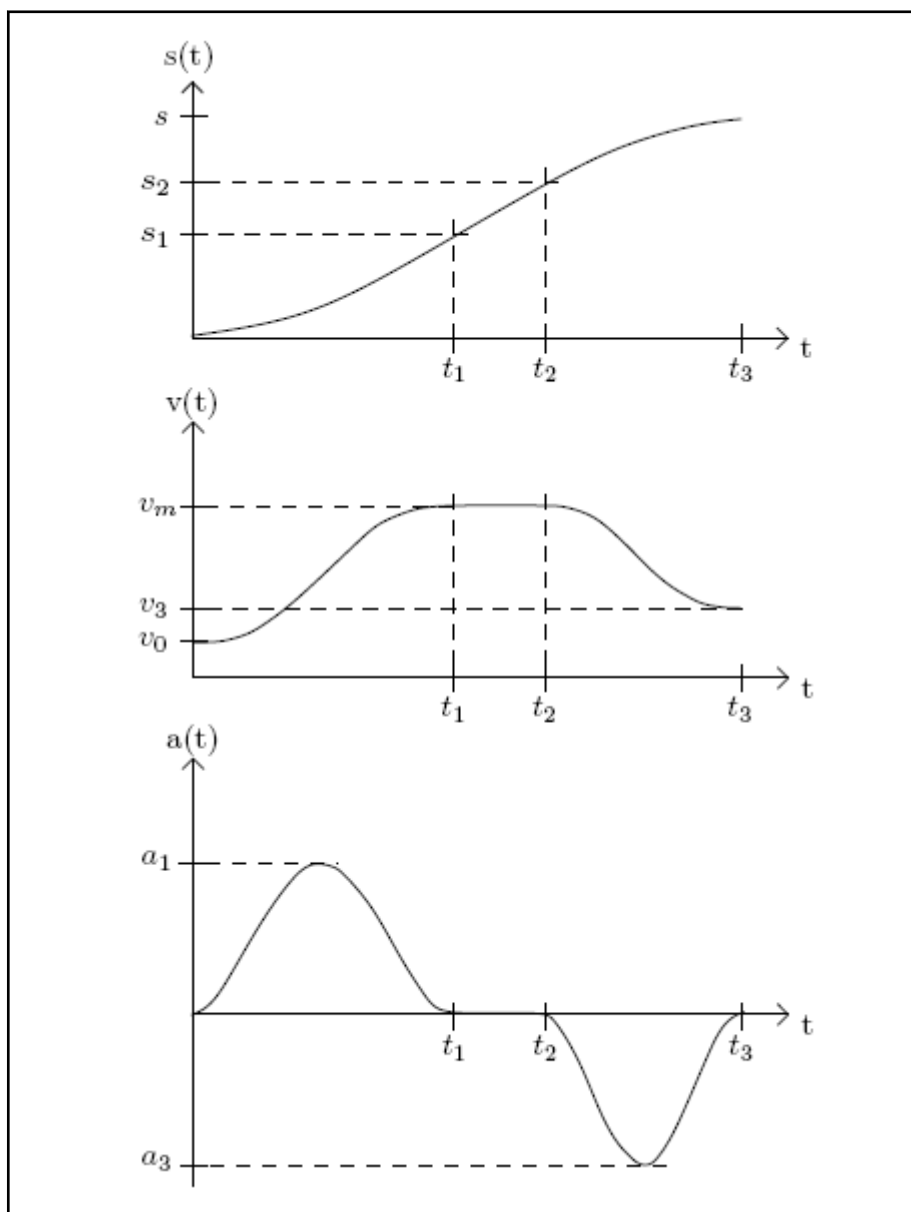


Fig. 14-48: Acceleration-limited motion with sinusoid profile

The profile for "**Jerk-limited motion**" can be described as follows:

An acceleration with jerk limitation applies to many demanding motion tasks. For example, for the transport of liquids or also in case of easily oscillating machines.

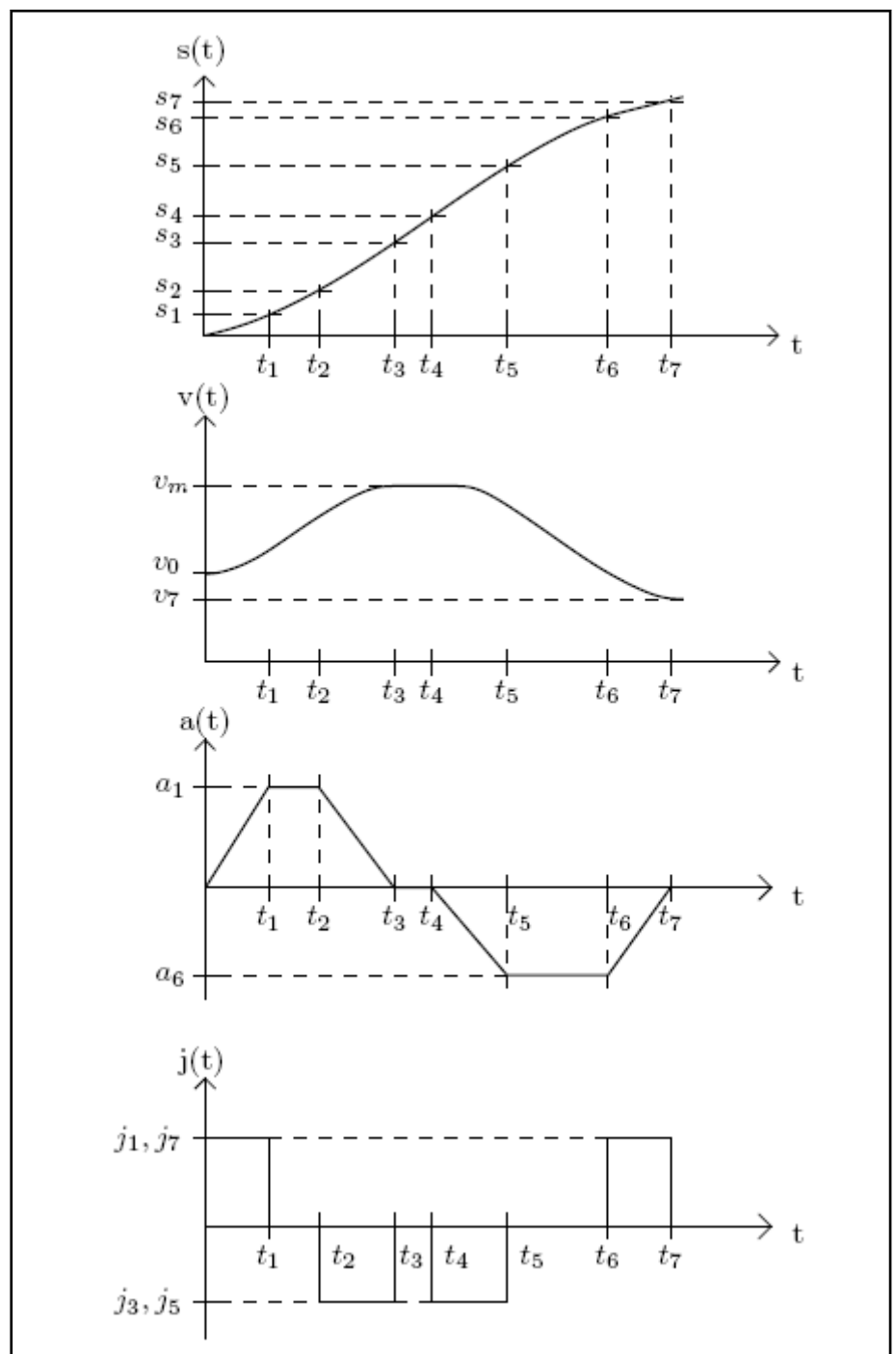


Fig. 14-49: Jerk-limited motion

14.2 RIL_HMI_Utility

The "RIL_HMI_Utility" library contains function blocks supporting the HMI devices.

The library is supported by several targets and therefore is described in the documentation "Basic Libraries, IndraLogic 2G".

Basic libraries

14.3 RIL_LoopControl

The "RIL_LoopControl" library provides basic elements and controllers of control technology.

The "RIL_CommonTypes" and "RIL_Uutilities" libraries have to be included in the project in order to use this library.

The library is supported by several targets and therefore described in the "Basic Libraries, IndraLogic 2G" documentation.

14.4 RIL_ModbusTCP

The field bus protocol "Modbus TCP" is used to exchange data between Ethernet devices which support the Modbus protocol.

In this case, the drive-internal PLC assumes the role of the "Modbus TCP" server (master) to which the "Modbus TCP" clients (slaves) can connect.

Using the function blocks contained in the "RIL_ModbusTCP" library, both a "Modbus TCP" server and a "Modbus TCP" client can be implemented.

The library is supported by several targets and therefore described in the "Basic Libraries, IndraLogic 2G" documentation.

14.5 RIL_SercosIII

14.5.1 General information

The "RIL_SercosIII" library is the interface between the PLC programming environment and Sercos III nodes. This interface allows, for example, the Sercos III I/O nodes to be diagnosed or Sercos parameters to be acyclically exchanged.

The library is supported by several targets and is therefore described in the "Field Bus Libraries" documentation.

14.6 RIL_SocketComm

The "RIL_SocketComm" library contains functions and function blocks which allow using the TCP/UDP communication.

The library is supported by several targets and therefore described in the "Basic Libraries, IndraLogic 2G" documentation.



Due to the operating system of drives, the "IL_Ping" function and the "IL_PingAsync" function block cannot be used. They map the ICMP protocol and collide with the realization of the ICMP protocol in the operating system. As a replacement, the "SysSockPing" function of the "SysSocket" library can be used which synchronously transmits the ping to the operating system.

14.7 RIL_Uutilities

The "RIL_Uutilities" library contains functions and function blocks for converting and influencing different data types.

The library is supported by several targets and therefore is described in the documentation "Basic Libraries, IndraLogic 2G".

14.8 RMB_SercosIII_Util

The "RMB_SercosIII_Util" library provides the following functions:

- Functions for simplified axis control of drives via the Sercos bus.

Note: The drive in the control has to be created as PLC device for axis control via Sercos bus.

- Functions to operate a frequency converter of the "EFC" type in the "frequency control" mode.

The library is supported by several targets and is therefore described in the "Field Bus Libraries" documentation.

15 System libraries

15.1 General information

The below-mentioned system libraries are supported by Rexroth IndraMotion MLD with MPx-18 (IndraDrive) / HDx-18 (HydraulicDrive) and above.

NOTICE

Property damage caused when non-supported libraries are included.

Only the libraries listed below are allowed to be included.

15.2 Standard

As a standard, the "Standard.library" library is made available with the IndraLogic 2G development system.

It contains all functions and function blocks that are required by IEC61131-3 as standard blocks for an IEC programming system. The difference between a standard function and an operator is that the operator is implicitly known to the programming system while the standard blocks must be included in the project as a library.

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.3 Util

The "Util.library" library contains function blocks which can be used for BCD conversion, bit/byte functions, mathematical auxiliary functions, as controllers, signal generators, function manipulators and for analog value processing.

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.4 SysDir

15.4.1 General information

With the functions of the "SysDir" library it is possible to synchronously access a file directory system on the target. Directory entries can be read.

The library can be added via the library manager (category "System"- "SysLibs").



Restriction: The "SysDirGetCurrent" and "SysDirSetCurrent" functions are not supported.

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.4.2 Instructions for use

General information

Only the "User" directory on the external memory card can be accessed. Any access from the user program to the root directory is redirected to the "User" directory.

System libraries

When listing the directories, only the content of the "User" directory and possibly contained subdirectories are displayed. The "User" directory itself is not displayed, as if "User" were a separate partition. Accessing the root branch from the user's point of view is possible without specifying the directory, with single "\" and with double "\\" (Windows-compatible).

Runtime behavior When synchronous file system access is used in a cyclic MLD task, the time behavior of this task might possibly be changed. Therefore, it is advisable to carry out the file system access from a separate, freewheeling task.

15.5 SysFile

15.5.1 General information

The functions of the "SysFile" library support **synchronous** file access. "Synchronous file access" means that a function is called and only returns after the output has been completed.

With the library functions, files can be

- opened
- closed
- deleted
- renamed
- In addition, it is possible to write data to or read data from files.
- Other functions make it possible
 - to determine the file size
 - the point of time of the last access, and
 - to determine or change the current offset

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.5.2 Instructions for use

General information Only the "User" directory on the external memory card can be accessed. Any access from the user program to the root directory is redirected to the "User" directory.

When listing the directories, only the content of the "User" directory and possibly contained subdirectories are displayed. The "User" directory itself is not displayed, as if "User" were a separate partition. Accessing the root branch from the user's point of view is possible without specifying the directory, with single "\" and with double "\\" (Windows-compatible).

Runtime behavior When synchronous file system access is used in a cyclic MLD task, the time behavior of this task might possibly be changed. Therefore, it is advisable to carry out the file system access from a separate, freewheeling task.

15.6 SyslecTasks23

From the "SyslecTasks23" library, only the "SysIECTaskGetConfig" function may be used; all other functions are not supported by IndraMotion MLD 2G and must not be used.

By means of the "SysIECTaskGetConfig" function, the task configuration can be read.

The tasks are listed according to their priorities, the cycle time is displayed in milliseconds.

15.7 SysMem

The "SysMem" library is used for memory management.

The library functions can be used to reserve (allocate) memory space and de-allocate it again, to define memory, to compare two memory ranges, to copy or displace data from one memory to the other, or to swap.

Processing takes place synchronously.

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.8 SysCallback23

The "SysCallback23" library contains functions for activating defined callback functions for runtime events.

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.9 SysSem

The "SysSem" library provides functions for generating and using semaphores for task synchronization. Semaphores serve as a means to avoid simultaneous access to critical data used by several tasks.

Processing takes place synchronously.



In the case of a PLC reset, no semaphores are unlocked. The user has to make sure to unlock the semaphores.

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

15.10 SysSockets

The "SysSockets" library provides functions supporting the access to sockets for communication via TCP/IP and UDP.

To begin with, please check whether the "RIL_SocketComm" library can fulfill your requirements.

If supported by the target, the corresponding functions of the operating system can be called via the library functions. For detailed information on the operating system functions, please see the relevant Help.

Please note:

- Depending on the target, the behavior of the functions can differ.
- Opening/closing sockets can take a lot of time when many sockets are opened/closed simultaneously.



The "SysSockets" library is incompatible with the "SysLibSockets" library of IndraMotion MLD 1G.

Although the "SysSockets23" library provides the programming interface of the "SysLibSockets" library of IndraMotion MLD 1G, the entire functionality is not covered. **Therefore, this library must not be used!**

The library is supported by several targets and therefore is described in the documentation "PLC System Libraries, IndraLogic 2G".

System libraries

15.11 SysStr23

15.11.1 Overview

Comparing Strings

The "SysStr23" library provides functions for handling strings.

- [SysStrCmp, page 230](#)
- [SysStrCmpl, page 230](#)
- [SysStrCmpN, page 230](#)
- [SysStrCmpNI, page 231](#)

Copying Strings

[SysStrCpy, page 231](#)

Determining the String Length

[SysStrLen, page 231](#)

The functions are processed synchronously.

15.11.2 SysStrCmp

With the "SysStrCmp" function of the DINT type, is it possible to compare two strings lexicographically.

Input variable	Data type	Description
sString1	STRING	First string
sString2	STRING	Second string

The function provides one of the following results:

Return value < 0	String1 smaller than String2
Return value = 0	String1 = String2
Return value > 0	String1 greater than String2

15.11.3 SysStrCmpl

With the "SysStrCmpl" function of the DINT type, is it possible to check whether two strings are identical.

Input variable	Data type	Description
sString1	STRING	First string
sString2	STRING	Second string

The function provides one of the following results:

Return value < 0	String1 smaller than String2
Return value = 0	String1 = String2
Return value > 0	String1 greater than String2

15.11.4 SysStrCmpN

The "SysStrCmpN" function of the DINT type allows comparing the size of two strings; in doing so, a defined character position starting from the beginning of the string is examined.

Input variable	Data type	Description
sString1	STRING	First string
sString2	STRING	Second string
diChars	DINT	Number of characters for which - counting starts at the beginning of the string - the size of the strings is to be compared

The function provides one of the following results:

Return value < 0	String1 smaller than String2
Return value = 0	String1 = String2
Return value > 0	String1 greater than String2

15.11.5 SysStrCmpNI

With the "SysStrCmpNI" function of the DINT type, is it possible to compare whether the starting ranges of two strings are identical. The number of characters for which counting starts at the beginning of the string must be set with "diChars".

Input variable	Data type	Description
sString1	STRING	First string
sString2	STRING	Second string
diChars	DINT	Number of characters for which counting and checking starts at the beginning of the string to find out whether the contents of the two strings are identical.

The function provides one of the following results:

Return value < 0	String1 smaller than String2
Return value = 0	String1 = String2
Return value > 0	String1 greater than String2

15.11.6 SysStrCpy

The "SysStrCpy" function of the DWORD type allows copying one string (sString2) to another string (sString1).

Input variable	Data type	Description
sString1	STRING	String to which the data are to be copied (target)
sString2	STRING	String that is to be copied (source)

The return value obtained is a pointer to the target string (sString1).

15.11.7 SysStrLen

With the "SysStrLen" function of the DINT type, is it possible to determine the length of a string.

Input variable	Data type	Description
sString1	STRING	String the length of which is to be checked

The return value obtained is the number of characters, "terminal NULL" not being counted.

System libraries

15.12 SysTimeRTC.library

15.12.1 Introduction

Firmware version 20 and above support the following functions of the "SysTimeRTC.library" system library:

- SysTimeRtcGet
- SysTimeRtcSet
- SysTimeRtcHighResGet
- SysTimeRtcHighResSet

Provides functions to read out or write the system time of the device in the IEC 61588 format ("S-0-1305.0.1, System time").

15.12.2 SysTimeRtcGet

Brief description Use this function to read the current RTC system time of the device in **seconds**.

Interface description

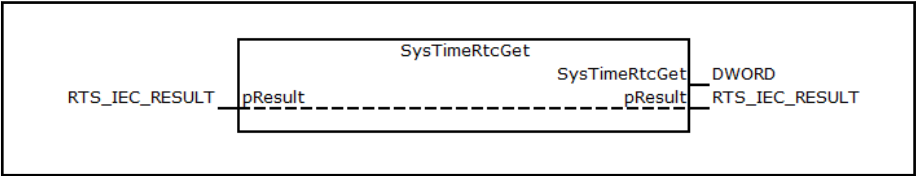


Fig. 15-1: FUN SysTimeRtcGet

Output variable	Data type	Description
SysTimeRtcGet	DWORD	RTC time in seconds since 01/01/1970 00:00:00

Input/output variable (VAR_IN_OUT)	Data type	Description
pResult	RTS_IEC_RESU LT	"ERR_OK" (16#0000) signals a successfully completed operation

15.12.3 SysTimeRtcSet

Brief description Use this function to set the current RTC system time of the device in **seconds**.

Interface description

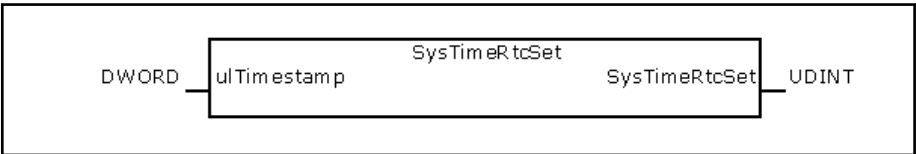


Fig. 15-2: FUN SysTimeRtcSet

Input variable	Data type	Description
ulTimestamp	DWORD	RTC time in seconds since 01/01/1970 00:00:00

Output variable	Data type	Description
SysTimeRtcSet	UDINT	ERR_OK (16#0000): Ok ERR_FAILED 16#0001): General internal error

15.12.4 SysTimeRtcHighResGet

Brief description Use this function to read the current RTC system time of the device in seconds.

Interface description

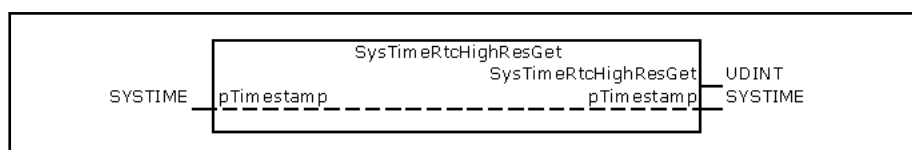


Fig. 15-3: FUN SysTimeRtcHighResGet

Output variable	Data type	Description
SysTimeRtcHighResGet	UDINT	"ERR_OK" (16#0000) signals a successfully completed operation

Input/output variable (VAR_IN_OUT)	Data type	Description
pTimestamp	SYSTIME	RTC time in milliseconds since 01/01/1970 00:00:00

15.12.5 SysTimeRtcHighResSet

Brief description Use this function to set the current RTC system time of the device in milliseconds.

Interface description

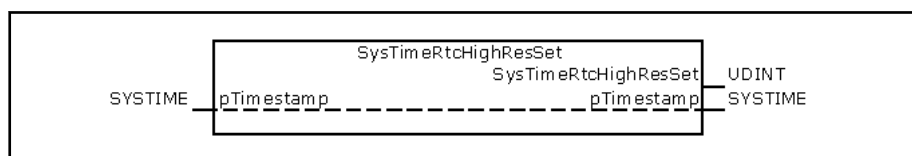


Fig. 15-4: FUN SysTimeRtcHighResSet

Input/output variable VAR_IN_OUT	Data type	Description
pTimestamp	SYSTIME	Time in milliseconds since 01/01/1970 00:00:00 to be set

Output variable	Data type	Description
SysTimeRtcHighResSet	UDINT	"ERR_OK" (16#0000) signals a successfully completed operation

15.13 IoDrvEtherNetIP.library

Description version/library availability

- Firmware: MPC-20V12
- Engineering: IndraWorks MLD 14V16
- MLD system configuration / compiler: CoDeSys 3.5.6.40
- Library: IoDrvEtherNetIP.library 3.5.7.0

The "IoDrvEtherNetIP.library" library provides an EtherNet/IP scanner. Using this scanner, other EtherNet/IP devices can be integrated as I/O slaves (e.g., barcode reader, RFID, flow rate meter, laser distance meter, I/O, HMI,...).

The "IoDrvEtherNetIP.library" library was loaded to the MLD with the application.

The configuration is realized via standard I/O mechanisms in which the scanner and the adapters (I/O slaves) are added to the project as devices

System libraries

(also see "[Configuring an EtherNet/IP scanner](#)"). Scanner and adapter are stored in the device database. Missing adapters can be added using the EDS description files. It is also possible to configure an adapter via a generic, freely configurable device.

Due to its implementation in the IEC code and the technical properties of the drive, the EtherNet/IP scanner has a restricted scope of application that has to be considered when configuring the EtherNet/IP scanner.



The EtherNet/IP scanner runs in an MLD application task, referred to as bus cycle task in the following. Thus, the scanner cannot fulfill the requirements on hard real-time. Furthermore, a noticeable cycle time load and an increased memory requirement during the MLD configuration due to the EtherNet/IP scanner has to be taken into consideration.

In the following chapters, the characteristics to be taken into consideration in the MLD context are described, followed by an overview of the editors available for EtherNet/IP scanner and adapter configuration.

15.13.1 Licensing and operation

The EtherNet/IP scanner requires a license available in the firmware in binary format. The license is activated via the device key (enabling functional packages) "MLD Advanced with EtherNet/IP" ("**ME**" design).



If the license is missing, the stack runs for 30 minutes until it stops. Execute a "warm" reset at the MLD to start (loading the test license).

In devices in which the EtherNet/IP scanner operation is generally not supported, an error because of unresolved references is generated during the download as soon as an EtherNet/IP scanner is configured in the MLD application.

15.13.2 MLD performance and cycle times

The processing of the cyclic data causes a load of typically about 2ms on the MLD bus cycle task with 4 simultaneously operated EtherNet/IP adapters. In the case of a high network load, the processing in the bus cycle task can sporadically have an additional load of up to 5ms.

- The cyclic data should not be configured faster than with a RPI=10 ms
- With an RPI=10 ms, the bus cycle task should exclusively be used for EtherNet/IP communication.

When configuring the watchdog for the bus cycle task, take the sporadic load on the bus cycle time due to network peaks into account in MLD. Besides, it is recommended to switch on the watchdog only after the EtherNet/IP scanner has been initialized via `IecTaskEnableWatchdog()`.

15.13.3 Resource requirement in the boot project

As data and code of the EtherNet/IP scanner and the subordinate adapter are stored in IEC code, the resource consumption during the system configuration has to be taken into consideration. Apart from the code and data memory, sockets are assigned for the IP-based communication.

The resource demand results from the following specifications:

EtherNet/IP scanner

- Code: approx. 300 kBytes

- Global data: approx. 35 kBytes
- Dynamically allocated memory: approx. 5 kByte
- Sockets: 3

EtherNet/IP adapter

- Code: approx. 0.5 kBytes
- Global data: approx. 8 kBytes
- Dynamically allocated memory: approx. 1.5 kByte
- Sockets: 1 (in case of acyclic communication)



The EtherNet/IP scanner can be operated independently of a μ SD card. However, the μ SD card is normally required for the MLD boot project, due to the high memory requirements of the EtherNet/IP scanner.

15.13.4 Supported interfaces and coexistence with Sercos

As the EtherNet/IP scanner is based on the socket library, the scanner use is not connected to a certain Ethernet interface. However, the interface selection affects the scanner performance as the performance of the Ethernet interfaces differs. Of the Ethernet interfaces available at the drive, generally only the Master Sercos Interface (X24 and X25) is suitable for connecting EtherNet/IP adapters. This has to be taken into account for the project planning of the installation.

Relevant Master Sercos Interface type data:

Interface	Available memory		Number of IP messages that can be buffered	
	Receive buffer	Transmit buffer	Receive buffer	Transmit buffer
Sercos master	6 kByte (per port, segmented)	6 kByte (shared, segmented)	8 (per port)	8 (per port)

Tab. 15-1: Master Sercos Interface type data



The number of EtherNet/IP adapters that can be simultaneously operated is generally limited to 8 adapters per port (X24 and X25).

Using the Sercos and EtherNet/IP master communication simultaneously is only possible in a line topology and under one of the following two conditions:

1. „Single Wire Coexistence“ (SWC, Support of Industrial Ethernet protocols via UCC) is used, and the EtherNet/IP adapters are connected at the last inactive port of the Sercos slaves. SWC has to be enabled in the CCD group settings.
2. Each protocol uses a port (X24 or X25); this means one port is exclusively used for Sercos at the Sercos interface and one port exclusively for EtherNet/IP.

This is the preferred alternative, as SWC is currently only supported by a few Sercos nodes and makes specific demands on the wiring.

15.13.5 Requirements on the IP network

If the interface used for EtherNet/IP is also used for other protocols (Engineering, data transfer, etc.), this has an effect on the availability of the transmit and receive buffers of the interface for EtherNet/IP communication. The number of adapters that can be simultaneously operated can thus be

System libraries

reduced. The restrictions depend on the use case and have to be assessed by the user during evaluation. The following effects have to be taken into consideration:

1. It is possible that only a part of the transmit and receive buffers of the interface is available to the EtherNet/IP communication.
 - In case of load peaks, sporadically, no receive buffer is available to receive telegrams which can result in loss of data.
 - Sporadic load peaks can occur as the EtherNet/IP adapters are not synchronized.
 - To transmit compressed bus cycle task data, the transmit buffer in the interface has to have the capacity to contain many IP messages. If this is not the case, delays in the bus cycle task can occur if the transmit buffer in the interface has already been assigned by other protocols.

Thus, the cycle time of the bus cycle task in the MLC can increase.

2. Due to the socket-based realization of the EtherNet/IP scanner, the scanner uses the TCP/IP stack in the drive. Due to simultaneous accesses by other protocols, delays can occur. In particular if the data volume transferred via other protocols is considerably high. It is to be taken into consideration that all drive interfaces are connected by a common TCP/IP stack.

Thus, the cycle time of the bus cycle task in the MLC can increase.

15.13.6 Restrictions of the functional scope of the library

Note the following restrictions in the scope of functions when using the library "IoDrvEtherNetIP.library" in the MLD context:

1. Not more than 4 EtherNet/IP adapters can be configured and operated. Depending on the network load, the number of adapters that can simultaneously be operated can be further reduced. For details, refer to ["chapter 15.13.4 "Supported interfaces and coexistence with Sercos" on page 235"](#) and ["chapter 15.13.5 "Requirements on the IP network" on page 235"](#).



The EtherNet/IP adapter can be disconnected from or connected to the bus during a running control via the device diagnostics. Enable the device diagnostics in the PLC settings. Consequently, the "CAA Device Diagnostics" library is used. For more details, refer to the description of the "CAA Device Diagnostics" library.

2. Only point-to-point connections are supported. Multicast messages can be transmitted but not received by the drive.
3. The IP address of the scanner results from the interface selection that has to be entered in the configuration editor of the scanner. The IP addresses of the adapter also have to be set correctly in the project. All IP addresses have to be manually entered in the EtherNet/IP scanner settings and the adapters.
4. Due to the restrictions and the fact that an EtherNet/IP scanner does not have to be certified, the scanner is not certified in the MLD.

NOTICE

The EtherNet/IP communication is not suited for safe communication. The sporadic loss of cyclic data can be tolerated. Therefore, the EtherNet/IP specification specifies timeout monitoring with a time multiplier of no less than 4.

However, safe operation without telegram losses is to be expected if the above mentioned type data are taken into consideration.

15.13.7 EtherNet/IP scanner configuration

The EtherNet/IP scanner is an IEC-based I/O driver and can thus be configured using the standard editors for the configuration of inputs and outputs.

Adding the device branches



Since the Ethernet branch is only required for Ethernet-based I/O protocols, it is variable and has to be added manually. Furthermore, the branch is not created with the standard project.

Steps to add an MLD project, an EtherNet/IP scanner and an EtherNet/IP target (or an EtherNet/IP adapter):

Condition: A compatible MLD project (MPC20) was created.

1. Right-click the MLD branch in the Project Explorer to call the context menu.
2. Select **Add ► Ethernet**.

The "Ethernet (Ethernet)" branch is inserted below the MLD branch.

3. Call the context menu at the "Ethernet (Ethernet)" branch.
4. Select **Add ► Add device ► MLD EtherNet/IP scanner**.

The "MLD_Ethernet_IP_Scanner (MLD Ethernet/IP scanner)" branch is inserted below the MLD branch.



The library "IoDrvEtherNetIP.library" is automatically added by creating the branch for the EtherNet/IP scanner. The library version is determined by the device version. The version can subsequently be adjusted via the library placeholder.

5. Call the context menu at the "MLD_Ethernet_IP_Scanner (MLD Ethernet/IP scanner)" branch.
6. Select **Add ► Add device ► Communications adapter**. Select the device.
(In preparation: In the project, individual adapters can be enabled or disabled in the device tree via the selection boxes. Use the mouse to expand the selection boxes under the device branch symbol.)

Task configuration

By creating the scanner branch, a task for cyclic I/O data ("ENIPScannerIOTask") and a task for acyclic communication with explicit messages can be created ("ENIPScannerServiceTask") (also see "[Acyclic communication \(Explicit Messaging Service\)](#)"). User-defined calls can be

System libraries

added to these tasks. Furthermore, it is possible to rename the tasks. As usual, up to 4 tasks are available in the application.



Use Drag&Drop to move the automatically added calls of the created tasks to other user tasks.

If the tasks are deleted by the user, they can only be restored again by deleting the "MLD EtherNet/IP scanner" device branch and creating it again.

IP addressing

The basis for using the EtherNet/IP scanner is an interface (port). Via this port, communication via TCP and UDP (unicast) is possible. This is generally the case for all IndraDrive ports, however, it is only recommended for a CCD port. The IP address of the scanner results from the port selection that has to be entered in the configuration editor of the scanner. The IP addresses of the adapter also have to be set correctly in the project. All IP addresses have to be manually entered in the EtherNet/IP scanner settings and the adapters and can thus not be dynamically assigned during operation.



The adapters also have to be configured so that the IP addresses are assigned in a fixed way. Disable "BOOTP" and "DHCP" in the adapters. Otherwise, it can sporadically occur that no communication can be established to the adapters. For the procedure, refer to the respective adapter documentation.



Setting the IP address of the EtherNet/IP scanner: See "[MLD EtherNet/IP scanner](#)"

Setting the IP address of the EtherNet/IP adapter: See "[EtherNet/IP target](#)"

I/O configuration

The procedure in case of I/O configuration corresponds to the procedure for other inputs and outputs. The I/O addresses are automatically assigned as for the Sercos I/O in the adapter image when adding the adapters. In the adapter properties, defined variables can be assigned (mapping) to the inputs and outputs in the application or to other I/O addresses.



Behavior of the outputs in stop is determined by the bus. For Ethernet/IP, it has been specified that the RUN-IDLE bit is set to IDLE (0) in the case of PLC Stop. The adapter evaluates the bit and sets the outputs according the image or uses individual substitute values, irrespective of its own configuration. The PLC setting "I/O at stop" thus refers to the output mapping and does not necessarily have to match the outputs at the individual adapters.

If the bus cycle task is also stopped in stop, no communication with the adapter takes place and the adapter sets its outputs depending on its configuration.



A maximum of 511 byte per connection is possible. As only one "Exclusive Owner Connection" (= connection with inputs and outputs) can exist per adapter, the maximum size of the inputs and outputs is restricted to 511 bytes per adapter. Background: The field for the connection size is only 9 bit. The EtherNet/IP extension "Large Forward Open" is not supported anymore.

User parameters

Apart from the cycle I/O data exchange, there is the option, to define additional parameters - user parameters - that are transmitted to the adapter once upon start of the cyclic data exchange (bus system). The scope of available parameters is determined by the I/O device. User parameters are created in the adapter properties. If parameters are described in the device description file, a selection of these parameters is offered in the editor upon creation. Otherwise, they have to be created manually.

15.13.8 Editors for the Ethernet/IP configuration

An EtherNet/IP network consists of an EtherNet/IP scanner and one or several EtherNet/IP targets. The scanner maps the master and the slave targets in the network.

MLD only supports the scanner function. Configure the master communication accordingly to use the drive as adapter.

IndraWorks provides dialogs to set the parameters as well as to map the input/outputs of the variables.

Ethernet

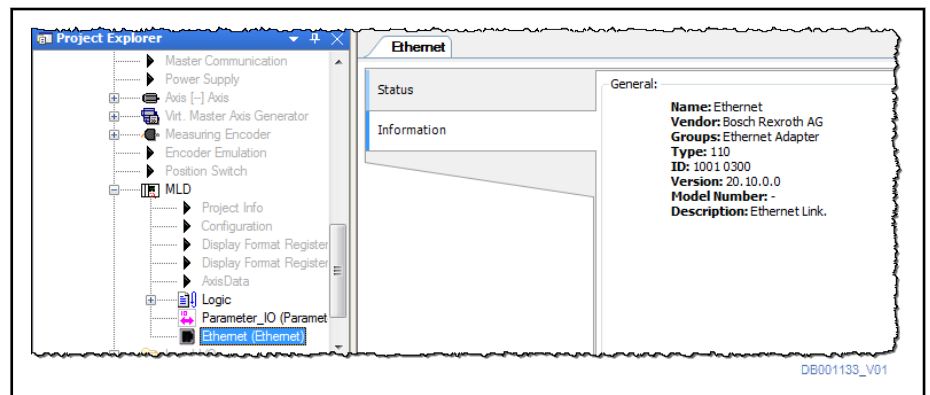


Fig. 15-5: "Ethernet" dialog

Status The "Status" tab displays status information to the device and the function. The status is always "Running" in the Ethernet device.

Information The "Information" tab, displays general information of the device description file:

- Name
- Vendor
- Categories
- Type
- ID
- Version
- Order number
- Description

System libraries

MLD EtherNet/IP scanner

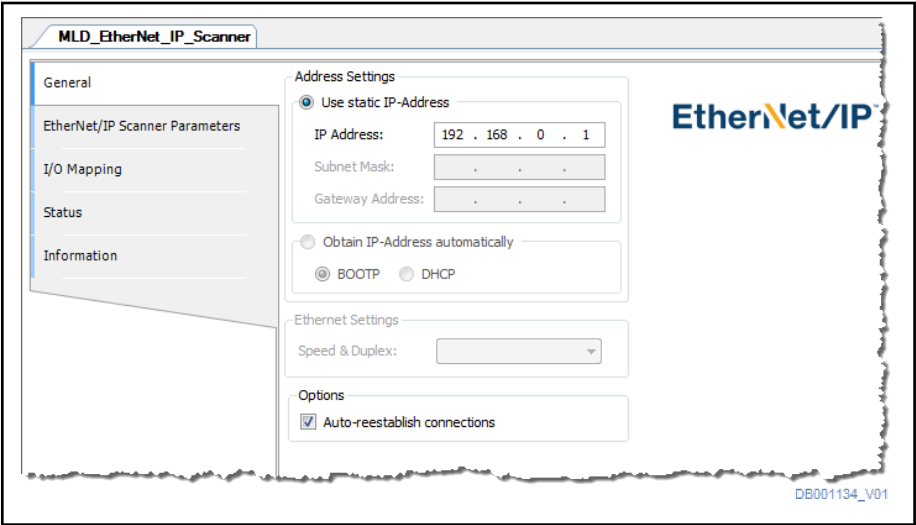


Fig. 15-6: "MLD EtherNet/IP scanner" dialog

General information

The "General" tab in the configurator of the EtherNet/IP scanner contains the basic settings for the communication in the network.

Address settings

Use static IP addresses.	
IP address: Subnet mask: Gateway address:	Each of these entries is assigned to four bytes. The entries are used to identify the scanner within the network environment. The "Subnet mask" settings and the "Gateway address" are not supported.
Obtain IP address automatically: <i>The option is not supported.</i>	
BOOTP	IP address assignment by a server using the Bootstrap Protocol (BOOTP)
DHCP	Automatic network settings configuration by the host using the Dynamic Host Configuration Protocol (DHCP)

Ethernet settings

Speed & duplex <i>The option is not supported.</i>	Transmission bit rate In case of "autonegotiation" the highest available bit rates are automatically selected.
---	---

Options

Automatically connections	reestablish If this option is selected, the scanner automatically tries to reestablish the connection.
------------------------------	---

EtherNet/IP scanner parameter

The "EtherNet/IP scanner parameters" tab in the configuration of the EtherNet/IP scanner corresponds to the generic configuration dialog. The dialog shows the "ConfigVersion". This value was generated automatically and shows the plugin version in IndraWorks used to configure the scanner.

If the option "Show generic configuration editors" option is enabled, the dialog is extended by another dialog, showing the device-specific parameters as

lists. This dialog is intended for test purposes. Only experts should change values.

I/O image

The "I/O image" automatically displays the name of the generated adapter instance under "IEC objects" in the "Variable" column. Thus, access to this instance via the application is possible. Thus, a restart, stop or a scanner status check can be executed via an application.

No project variables can be mapped to inputs and outputs in the EtherNet/IP scanner.

Bus cycle options

Bus cycle task	The drop-down list provides all tasks defined in the task configuration of the active application. The parent branch settings are applied when selecting "Use parent bus cycle settings". If no task is defined, the task selected in the PLC settings applies. If no task is selected in the PLC settings either, the highest priority task is used as bus cycle task.
----------------	---

Status

The "Status" tab displays status information (e.g. "Running", "Stopped") and specific diagnostic messages. The status information is displayed separately for the device and the bus system. Acknowledgeable diagnostic messages can be acknowledged.

Information

The "Information" tab, displays general information of the device description file:

- Name
- Vendor
- Categories
- Type
- ID
- Version
- Order number
- Description

System libraries

EtherNet/IP target

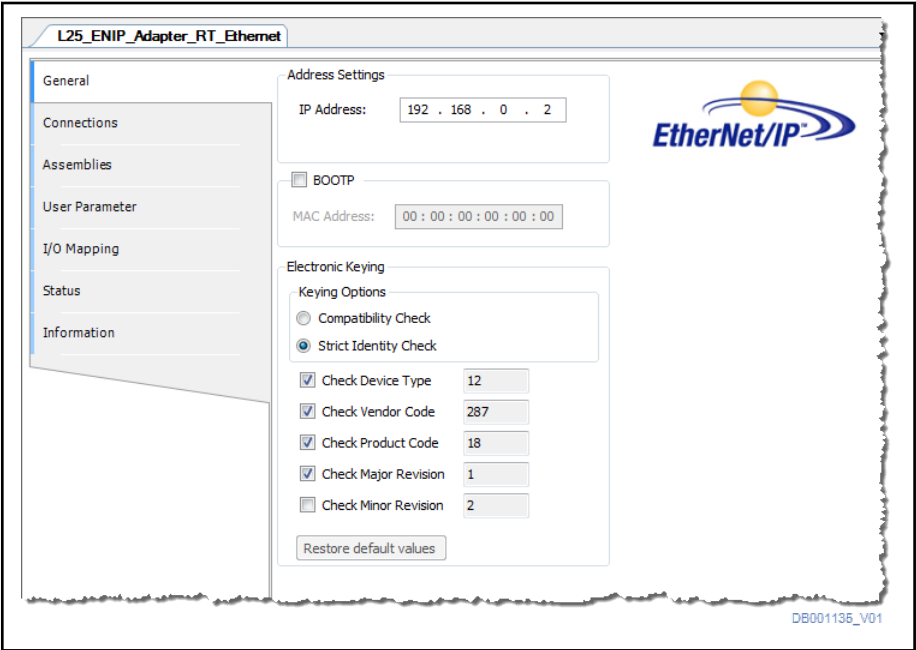


Fig. 15-7: EtherNet/IP target dialog

General information

The "General" tab in the configuration editor of the EtherNet/IP target contains the basic settings for the communication in the network.

Address settings

IP address	Address to identify the EtherNet/IP target device.
------------	--



The adapters also have to be configured so that the IP addresses are assigned in a fixed way. Disable "BOOTP" and "DHCP" in the adapters. Otherwise, it can sporadically occur that no communication can be established to the adapters. For the procedure, refer to the respective adapter documentation.

BOOTP

Bootstrap Protocol (BOOTP) <i>This option is not supported by the scanner.</i>	
MAC address	Device-specific MAC address of the slave

Electronic keying

Keying options	Compatibility check: Target (adapter) has to execute a compatibility check of the keying values of the device data sheet (*.EDS) with its individual keying values. Strict identity check: Target (adapter) has to execute an exact check of the keying values of the device data sheet (*.EDS) with its individual keying values. If the check fails, no I/O connection is established to the device and an error message is output on the status page.
----------------	---

System libraries

Check device type	If the option is selected, the information for the compatibility and identity check is taken into consideration. These values can be changed in case of generic devices.
Check vendor code	
Check product code	
Check the major revision	
Check the minor revision	

Connections

The "Connections" tab displays a list of all configured connections in the upper part. If the "Exclusive owner" connection in the device data sheet (*.EDS) is available it is automatically added when adding the target. The configuration data of the connections can be changed in the lower dialog part.

The configuration data are defined in the device data sheet (*.EDS). They are transmitted when opening the connection to the adapter.

RPI (ms)	Requested package interval: Exchange interval of input and output data
O -> T size (bytes)	Size of producer data from scanner to adapter (originator --> target)
T -> O size (bytes)	Size of consumer data form adapter to scanner (T --> O)
Config #1 size (bytes)	Configuration data 1 size
Config #2 size (bytes)	Configuration data 2 size
Connection path	Address of the - Configuration objects - Input objects - Output objects
Add connection...	Click to open the "New connection" dialog. In the dialog, the parameters for the new connection can be specified.
Delete connection	Clicking the button deletes the selected connection from the list.
Edit connection...	Click to open the "Edit connection" dialog. In the dialog, the parameters for the existing connection can be changed.

Configuration data

Symbolic values	If this option is selected, the value is displayed as text, if defined in the device data sheet (*.EDS).
Presettings	Click the button to reset the values to default.
Value	Double-click a value to change it.

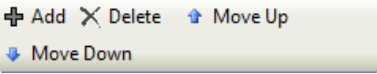
Assemblies

The tab displays a list of all configured connections in the upper part. When selecting a connection, the corresponding assemblies are shown in the lower dialog part.

The columns are described under "[Connections](#)" of the EtherNet/IP target description.

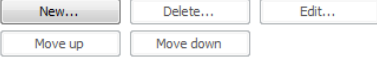
System libraries

Output assembly, input assembly

	Click the Add button to open the "Select parameters" dialog.
	Click the Delete button to delete all parameters that have been selected.
	Click the Up/Down button to move all selected parameters within the list.
	The list order determines the order in the I/O image.
Name	Double-click a value to change it.
Bit length	
Help string	

User parameters

The "User parameters" tab shows all additional parameters that are transmitted to the bus system once in the phase of the starting sequence intended for this purpose.

	Click the New... button to open the "Select parameters" dialog and add a new parameter. The new parameter is inserted in front of the selected line.
	Click the Edit... button to open the "Select parameters" dialog and change an existing parameter.
	Click the Delete... button to open the "Select parameters" dialog and delete an existing parameter.
	Click the Up/Down button to change the user parameter order. The parameter order in the list corresponds to the initialization order.
Value	The value of the respective parameter can be directly changed by double-clicking on the value. A list field with possible values is opened.
Cancel in case of error	If an error occurs, the entire parameter transmission is canceled.
In the case of error jump to line	In case of error, it is continued with the line specified in the "Next line" column. Thus, an entire block can be skipped during the initialization or a return can be defined. Note: A return can result in an infinite loop if it is never possible to write a certain parameter.

EtherNet/IP parameters

A "EtherNet/IP scanner parameters" tab in the configurator of the EtherNet/IP scanner corresponds to the generic configuration dialog. It is displayed if the option "Show generic configuration editors" is active in the IndraWorks options.

Device-specific parameters are displayed in a list. This dialog is intended for test purposes. Only experts should change values.

I/O image

The "I/O image" tab automatically displays the name of the generated target instance under "IEC objects" in the "Variable" column. Thus, access to this instance via the application is possible. Thus, a restart, stop or a target status check can be executed via an application.

In the tab, project variables are mapped to inputs and outputs of the target.

This description corresponds to the description of the general I/O image dialog.

System libraries

Always update variables	If this option is selected, all variables are updated in each cycle of the bus cycle task irrespective of whether they are used or mapped on an input or output channel.
Bus cycle task	The drop-down list provides all tasks defined in the task configuration of the active application, such as "MainTask". The parent branch settings are applied when selecting "Use parent bus cycle settings". If no task is defined, the task selected in the PLC settings applies. If no task is selected in the PLC settings either, the highest priority task is used as bus cycle task.

Status

The "Status" tab displays status information (e.g. "Running", "Stopped") and specific diagnostic messages. Acknowledgeable diagnostic messages can be acknowledged.

Information

The "Information" tab, displays general information of the device description file:

- Name
- Vendor
- Categories
- Type
- ID
- Version
- Order number
- Description

"Select parameters" dialog

"Select parameters" dialog

Show parameter groups	If this option is selected, the dialog shows all parameters of the device data sheet (*.EDS) summarized in groups. If this option is not selected, the dialog shows all parameters of the device data sheet (*.EDS) in a flat structure. You can select individual parameters of this list and add them to the assemblies list with OK.
Generic parameter	If this option is selected, generic parameters can be added. The individual parameter values can be edited.

15.13.9 Acyclic communication (Explicit Messaging Service)

The "EtherNet/IP services" library contains function blocks to implement an acyclic communication with the adapter. They implement the functions and attributes of CIP objects, as described in the CIP specification of Open DeviceNet Vendor Association (ODVA):

- Get_Attribute_Single
- Set_Attribute_Single
- Get_Attributes_All
- Set_Attributes_All
- Reset
- NOP (No Operation)

System libraries

- Start
- Stop
- Apply_Attributes

"Ethernet/IP services" library**Get_Attribute_Single**

Use this function block to query the attribute of a certain CIP object instance.

Inputs

Name	Data type	Description
xExecute	BOOL	Rising edge: The action starts Falling edge: Resetting the outputs If a falling edge occurs before the function block completed its action, the outputs are only reset if either the action is completed or if an error occurred. In this case, the corresponding output values (xDone, xError, eError) are present at the output for exactly one cycle.
itfEtherNetIPDevice	IEtherNetIPService	EtherNet/IP device which implemented the EtherNet/IP service interface
eClass	WORD	Class that is to carry out the service
dwlInstance	DWORD	Instance that is to carry out the service (0: Class level, 1..x: Instance level)
pData	POINTER TO BYTE	Data buffer in which the attribute data are to be loaded
udiDataSize	USINT	Buffer size
wAttribute	Word	Attribute that is queried

Tab. 15-2: Inputs of the "Get_Attribute_Single" function block

Outputs

Name	Data type	Description
xDone	BOOL	Operation completed successfully
xBusy	BOOL	Function block enabled
xError	BOOL	0: No error 1: Error occurred, function block cancels the action
eError	Error	Error (0-255: CIP error; 256-x: library error)

Tab. 15-3: Inputs of the "Get_Attribute_Single" function block

Example

This instance call queries the host name (= name of device in the network) of the DEVICE_123_ETHERNET device as soon as xExecute is TRUE.

Program:

```

getAttributeSingle(
    xExecute:=,
    (* instance of the device (instance is found in the I/O Mapping of the device) *)
    itfEtherNetIPDevice:= DEVICE_123_ETHERNET,
    (* cip class which contains the desired attribute *)
    eClass:= ENIP.CIPClass.TCPIPInterfaceObject,
    (* value of 0 is class level, range from 1..x is instance level *)
    usiInstance:= 1,
    (* data buffer *)
    pData:= ADR(strHostnameToGet),
    (* size of the data buffer *)
    udiDataSize:= sizeof(strHostnameToGet) ,

```

```
(* attribute no. 6 of the tcp/ip interface object is the Hostname *)
(* (CIP Spec. Vol.2 Chapter 5-3.3.2.6) *)
usiAttribute:= 6,
xDone=> ,
xBusy=> ,
xError=> ,
eError=> ,
(* if the request was successfull, udiReceivedData contains the number of bytes *)
(* which have been received *)
udiReceivedDataSize=> udiReceivedData);
```

Set_Attribute_Single

The function block is used to set the attribute of a certain CIP object instance. The significance of the inputs/outputs is described in "Get_Attribut_Single" or in the documentation of the library in the library manager.

Example

The instance call sets the host name (= name of device in the network) of the DEVICE_123_ETHERNET device as soon as xExecute is TRUE.

Program:

```
strHostnameToSet.uiLength := INT_TO_UINT(Standard.LEN(strHostnameToSet.sString));
setAttributeSingle(
  xExecute:= ,
  (* instance of the device (instance is found in the I/O Mapping of the device) *)
  itfEtherNetIPDevice:= DEVICE_123_ETHERNET,
  (* cip class which contains the desired attribute *)
  eClass:= ENIP.CIPClass.TCPIPInterfaceObject,
  (* value of 0 is class level, range from 1..x is instance level *)
  usiInstance:= 1,
  (* data to be assigned to the attribute *)
  pData:= ADR(strHostnameToSet),
  (* size of the data *)
  (* here: size of the Length-Parameter PLUS the length of the string *)
  udiDataSize:= SIZEOF(strHostnameToSet.uiLength)
  + strHostnameToSet.uiLength,
  (* attribute no. 6 of the tcp/ip interface object is the Hostname *)
  (* (CIP Spec. Vol.2 Chapter 5-3.3.2.6) *)
  usiAttribute:= 6,
  xDone=> ,
  xBusy=> ,
  xError=> ,
  eError=> );
```

Get_Attributes_All

The function block is used to **query** all attributes of a certain instance of a CIP object.

The significance of the inputs/outputs is described in "Get_Attribut_Single" or in the documentation of the library in the library manager.

Set_Attributes_All

The function block is used to **set** all attributes of a certain instance of a CIP object.

The significance of the inputs/outputs is described in "Get_Attribut_Single" or in the documentation of the library in the library manager.

Reset

The function block is used to call the reset service of a certain CIP object instance. The effect of this service depends on the CIP object (see CIP specification volume 1 and volume 2).

The significance of the inputs/outputs is described in "Get_Attribut_Single" or in the documentation of the library in the library manager.

Example

System libraries

The instance call resets the DEVICE_123_ETHERNET device as soon as xExecute is TRUE.

Program:

```
reset(  
  xExecute:= ,  
  (* instance of the device *)  
  (* (instance is found in the I/O Mapping of the device) *)  
  itfEtherNetIPDevice:= DEVICE_123_ETHERNET,  
  (* cip class which contains the desired attribute *)  
  eClass:= ENIP.CIPClass.IdentityObject,  
  (* value of 0 is class level, range from 1..x is instance level *)  
  usiInstance:= 1,  
  xDone=> ,  
  xBusy=> ,  
  xError=> ,  
  eError=> );
```

NOP (No Operation)

The function block is used to call the "NOP" service of a certain CIP object instance. This service is usually used to check if the adapter still exists in the network.

The significance of the inputs/outputs is described in "[Get_Attribut_Single](#)" or in the documentation of the library in the library manager.

Start

The function block is used to call the "Start" service of a certain CIP object instance. The effect of this service depends on the CIP class (see CIP specification volume 1 and volume 2).

The significance of the inputs/outputs is described in "[Get_Attribut_Single](#)" or in the documentation of the library in the library manager.

Stop

The function block is used to call the "Stop" service of a certain CIP object instance. The effect of this service depends on the CIP class (see CIP specification volume 1 and volume 2).

The significance of the inputs/outputs is described in "[Get_Attribut_Single](#)" or in the documentation of the library in the library manager.

Apply_Attributes

The function block is used to call the "Apply_Attributes" service of a certain CIP object instance. Thus, all adapter attributes previously set with "Get_Attribut_Single" or "Get_Attribut_All" are applied and saved.

The significance of the inputs/outputs is described in "[Get_Attribut_Single](#)" or in the documentation of the library in the library manager.

16 Technology libraries

16.1 Introduction

The technology libraries are available in firmware version 20 and above. The libraries extend the basic functionality of the target and provide application-specific functions.

16.2 MX_TechInterface

16.2.1 Overview

Introduction to the "TechInterface"

The "AxisInterface" function is provided in the "MX_TechInterface.library".

The axis interface bundles and enhances PLCopen motion function blocks and provides an easy-to-use interface for drive functionality. Less code and more efficient commands speed up the programming of applications.

The axis interface contains control signals and parameters for the various operation modes of the master axis and slave axis, as well as adjustment options for selected process values.

The "MX_TechInterface" library is described in the online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology libraries".

Comparison between IndraMotion XLC/MLC and IndraMotion MLD

	IndraMotion XLC/MLC	IndraMotion MLD
FlexProfile	✓	–
MotionProfile	✓	✓
Parameterization level (ParamLevel)	✓	✓
Diagnostic messages	A-parameters	S-0-0390 / S-0-0095
Element "arAxisStatus_gb[].Admin.AxisFeatures"	✓	–



Other features that should be noted when using the axis interface in the IndraMotion MLD system are described in chapter ["Using the axis interface in the IndraMotion MLD system"](#).

16.2.2 Using the axis interface in the IndraMotion MLD system

Introduction

This chapter describes the features that should be noted when using the axis interface in the IndraMotion MLD system.

The axis interface cannot be used in every firmware variant:

- MPC: Use as MLD-M with up to 10 real axes and one virtual axis
- MPB: Use as MLD-S for one axis and one virtual axis

The additive functional package "ML - IndraMotion MLD with technology functions" has to be enabled to use the axis interface. The functional package "MA - enhanced technology function blocks" is not required.

Parameterization

Select the following parameter setting to use the axis interface on MLD:

Technology libraries

- PLC has permanent control over drive: "P-0-1367, PLC configuration", bit 4 set
- AxisData structure supported: "P-0-1367, PLC configuration", bit 6 set

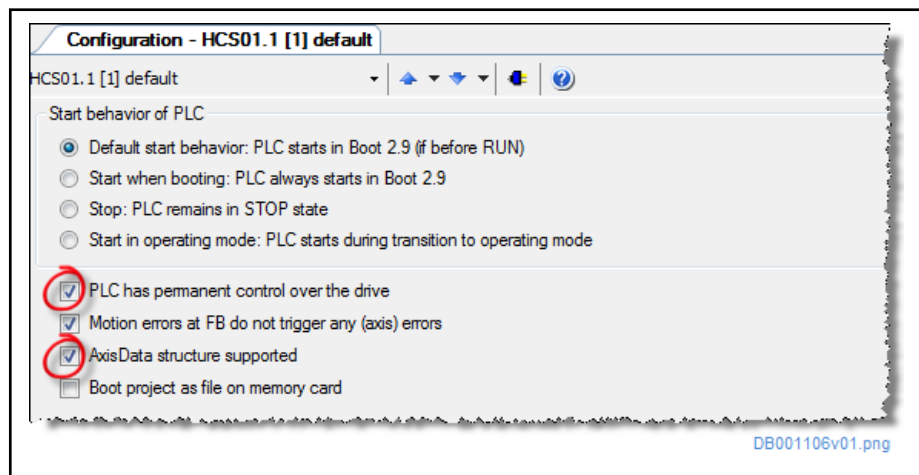


Fig. 16-1: Parameterizing MLD for axis interface

The boot project does not require a control panel with optional memory card. The memory card expands the usable memory capacity for the boot project.

The following settings are required to use the master axis generator as a virtual axis for the axis interface:

- Activating master axis generator: "P-0-0917, Control word of master axis generator", bit 0 set
If the master axis generator is activated in the CCD group, the following parameter applies: "P-0-1820.0.1, CCD: Master axes configuration list"
- Master axis generation: "P-0-0916, Master axis format converter signal selection", P-0-0758 entered
If the master axis generation is activated in the CCD group, the following parameter applies: "P-0-1822.0.1, CCD: Default master axis selection for the IndraDrive axes"
- This is an example of the settings for a rotary virtual axis that can rotate in both directions:
 - Rotary modulo scaling: "P-0-0756, Virtual master axis, scaling type", bit 2/1/0="010", bit 7 set
 - Shortest distance: "P-0-0769, Virtual master axis, command value mode", bit 1/0="10"

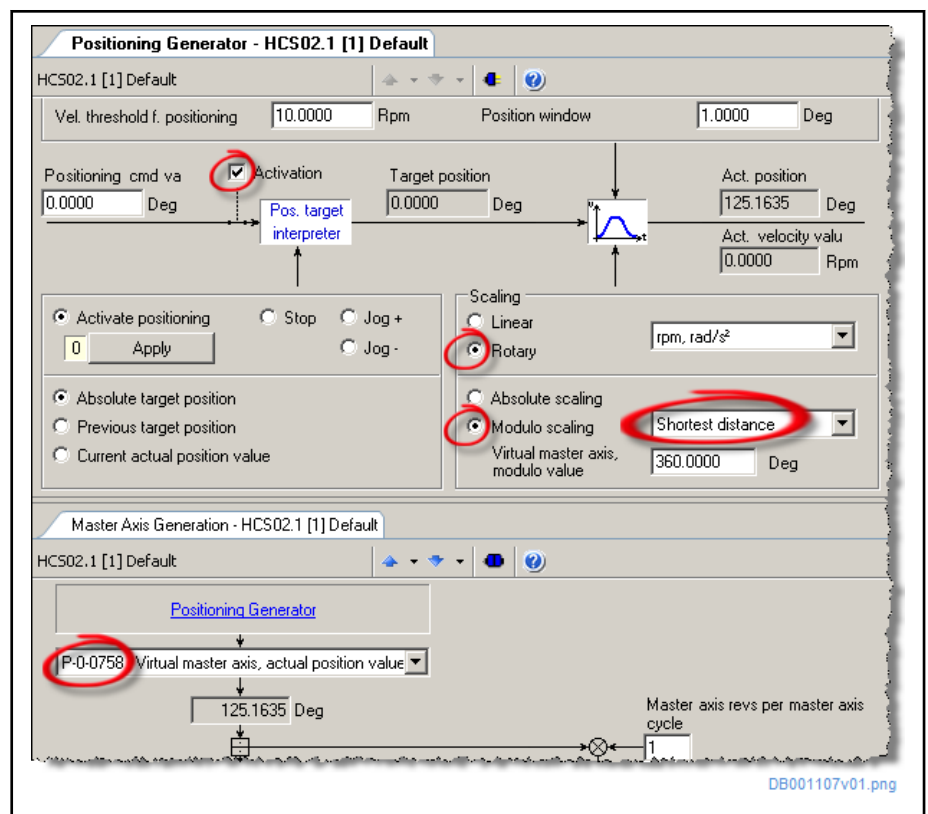


Fig. 16-2: Parameterizing the MLD master axis generator as a virtual axis

If the above mentioned parameterization is not performed, the exemplary AxisInterface project will not fully initialize and the user will have to make adjustments.

PLC programming

The axis interface structures are defined for 11 axes:

- Index1 to Index10: real axes
- Index20: virtual axis (=master axis generator) addressed via define "VMA_1"



Since the axis interface structures are defined for precisely 11 axes, indexing these structures with the predefined axis references "VmAxisInt.AxisNo" or "VmAxisExt.AxisNo" will result in the error message 2021 "Array limit exceeded".

Information on the virtual axis

- The axis interface structures are accessed with define "VMA_1 (Index 20)"
- The index can be determined on MLC and MLD regardless of the system by using the "MB_GetAxisInterfaceIndex()" function.
- For the virtual axis, the axis interface emulates some of the information provided in the "AxisData" structure for real axes. A pointer in the "AxisStatus" structure can be used to access this information.

Technology libraries



This pointer is initialized by the "MX_AxisInitType01" axis interface initialization function block. The pointer and with it the information can only be accessed after initialization has been successfully completed.

Information available for the virtual axes

The information of the real axes is available for virtual axes.

Information available for the virtual axes (this information is always set to "TRUE" in operating mode!)

Available information	Significance
arAxisStatus[VMA_1].PtrAxisData^.Axis_Inbb;	"Drive status bb" is displayed
arAxisStatus[VMA_1].PtrAxisData^.Axis_InAb;	"Drive status Ab" is displayed
arAxisStatus[VMA_1].PtrAxisData^.Axis_Power;	"Power on" is displayed
arAxisStatus[VMA_1].PtrAxisData^.Axis_Homed;	"Axis homed" is displayed

16.3 MX_TechBase

The "MX_TechBase" library contains basic functionalities such as control functions [IndraMotion Control (IMC) Interface], functions to query system information and trigonometric functions.

The "MX_TechBase" library is described in the IndraWorks online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology Libraries".

16.4 MX_TechMotionBase

The "MX_TechMotionBase" library contains functionalities such as probe function, position monitoring, position switch and functions to manage the cyclic data channel.

The "MX_TechMotionBase" library is described in the IndraWorks online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology Libraries".

16.5 MX_TechMotion

The "MX_TechMotion" library contains the "MB_TensionControlLoadCellType01" function block. Using a function block, a tension controller is provided, controlling the tension to a command value by evaluating the measured value of a force transducer.

The "MX_TechMotion" library is described in the IndraWorks online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology Libraries".

16.6 MX_TechWinder

The "MX_TechWinder" library contains function blocks providing functions for winding and unwinding fabric webs (winders).

The "MX_TechWinder" library is described in the IndraWorks online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology Libraries".

16.7 MX_TechCrosscutCrossseal

The "MX_TechCrosscutCrossseal" library contains function blocks for cross sealer and cross cutter applications.

The "MB_CrossCutterCalcType05" function block can be used with IndraMotion MLD (2G). The function block calculates a cam for the application as cross cutter. The cross cutter functionality can be configured.

The "MX_TechCrosscutCrossseal" library is described in the IndraWorks online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology Libraries".

16.8 RMB_TechCam

The "RMB_TechCam" library contains function blocks for creating cam tables.

The function blocks are available for different applications, such as feed motion, cross cutter, slotter, etc.

The "RMB_TechCam" library is described in the IndraWorks online help and in the printed documentation "Rexroth IndraMotion MLC 14VRS, Technology Libraries".

17 Free PLC library from "OSCAT"

The open-source (free) "Basic" PLC library by "OSCAT" ("Open Source Community for Automation Technology") contains many useful functions in the areas of "automation technology" and "building automation".

"OSCAT" libraries do not depend on the target or manufacturer.

The "Basic" library also contains function blocks for the following topics:

- Mathematical calculations
- String, list and buffer processing
- Logic modules
- Date and time functions
- Control technology
- Signal processing
- Network and communication functions
- Metrology and sensor technology
- Device drivers for standard peripherals

OSCAT libraries can be found online at www.oscat.de. Please note that these are not Rexroth products; we can therefore not be held liable for their completeness, functionality and operational reliability, and we do not provide any support for these libraries.



Libraries can be managed in various "repositories" (directories, storage locations) on the local system. Before a library can be included in a project, it has to be installed in one of these repositories on the local system. This is done in IndraLogic 2G using the "Library repository" dialog.



To avoid compiling errors due to names overlapping with those of other libraries, only include in a project the function blocks and data types from the OSCAT libraries that will actually be used. Thus, make sure that the **"Publish all IEC symbols to that project ..."** checkbox is **not** selected in the properties of the OSCAT libraries in the library manager.

Index

A

Analog value processing.....	227
AXIS_REF.....	187

B

BCD conversion.....	227
Bit/byte functions.....	227
Buffer processing.....	255
Bus cycle task	
IoDrvEtherNetIP.....	234

C

CheckBounds.....	53
CheckDivLReal.....	53
CheckDivReal.....	53
CheckPointer.....	53
CheckRangeSigned.....	53
CheckRangeUnsigned.....	53
CheckRtv.....	53
Communication via TCP/IP.....	229
Communication via UDP.....	229
Comparing strings.....	230
CONTROLS.....	186
Copying strings.....	230

D

Data types to return the date and the absolute time specifications in IndraMotion function blocks.....	193
Data types to support cams, MotionProfiles and FlexProfiles.....	200
Data types to support probe functions.....	195
Date functions.....	255
Determining the string length.....	230
Directory access.....	227

E

Error.....	23
Error handling.....	23
ERROR_CODE.....	23
ERROR_STRUCT.....	23, 193
ErrorID.....	23
ErrorIdent.....	23
EtherNet and Sercos	
Coexistence.....	235
EtherNet/IP	
IP network requirements.....	235
EtherNet/IP adapter	
Disconnect from bus.....	236
enabling/disabling.....	237
EtherNet/IP and Sercos	
Topology.....	235
EtherNet/IP scanner	
Interface.....	235
Resource requirement.....	234

Task configuration.....	237
EXT_SYS_TIME_DATE.....	194

F

File access	
Asynchronous.....	228
Synchronous.....	228

I

IL_NULL.....	186
INDRV_TABLE.....	29
Input signals "Execute" / "Enable".....	17
IoDrvEtherNetIP	
Introduction.....	233
Licensing.....	234
Performance and cycle time.....	234
Scope of functions.....	236
Test license.....	234
Watchdog.....	234
IoDrvEtherNetIP.library	
add.....	237
Change version.....	237

L

Library	
Debug.....	55
RIL_SercosIII.....	224
SysStr23.....	230
List processing.....	255
Logic modules.....	255

M

Master Sercos Interface	
Type data.....	235
Mathematical calculations.....	255
MB_ChangeCamData.....	61
MB_ChangeProfileSet.....	64
MB_ChangeProfileStep.....	66
MB_ClearAllError.....	69
MB_Command.....	130
MB_DIFF_EVAL.....	196
MB_EXPECT_WINDOW_MODE.....	196
MB_GearInPos.....	70
MB_GetAxisInterfaceIndex.....	251
MB_Home.....	75
MB_INTERPOLATION_MODE.....	197
MB_MotionProfile.....	76
MB_Phasing.....	79
MB_PhasingSlave.....	81
MB_POSITION_EVAL.....	198
MB_PreSetMode.....	83
MB_PROBE_CONFIG.....	198
MB_PROBE_DATA.....	199
MB_PROBE_INPUT.....	199
MB_PROBE_MODULE.....	200

Index

MB_ReadListParameter.....	135
MB_ReadMaxRealValue.....	137
MB_ReadMaxValue.....	139
MB_ReadMinRealValue.....	140
MB_ReadMinValue.....	142
MB_ReadName.....	143
MB_ReadParameter.....	145
MB_ReadRealListParameter.....	146
MB_ReadRealParameter.....	148
MB_ReadSercosAttribute.....	150
MB_ReadSercosDataStatus.....	151
MB_ReadStringParameter.....	153
MB_ReadUnit.....	155
MB_SetPositionControlMode.....	84
MB_Stop.....	86
MB_WriteListParameter.....	156
MB_WriteParameter.....	159
MB_WriteRealListParameter.....	160
MB_WriteRealParameter.....	163
MB_WriteStringParameter.....	165
MC_CAM_ID	200
MC_CamIn.....	88
MC_CamOut.....	91
MC_GearIn.....	92
MC_GearOut.....	94
MC_Jog.....	95
MC_MoveAbsolute.....	98
MC_MoveAdditive.....	100
MC_MoveRelative.....	103
MC_MoveVelocity.....	105
MC_Power.....	108
MC_ReadStatus.....	109
MC_Reset.....	111
MC_START_MODE	189
MC_STEP_MODE.....	201
MC_Stop.....	112
MC_SYNC_DIRECTION	191
MC_TorqueControl.....	114
Means of representation	
Conventions of notation.....	7
Notations.....	7
Notes.....	8
Memory management.....	229
MLD_TABLE.....	24
MX_CanL2.....	41
MX_CANL2_FifoReceive.....	41
MX_CANL2_FifoSend.....	42
MX_CANL2_Init.....	42
MX_CANL2_SetLED.....	44
MX_Command.....	167
MX_fCANL2_MBoxReceive.....	45
MX_fCANL2_MBoxRecStatus.....	47
MX_fCANL2_MBoxSend.....	46
MX_fCANL2_MBoxSndStatus.....	47
MX_fCANL2_Status.....	46
MX_fCANtoHl.....	49
MX_fCANtoHs.....	49
MX_fCheckCmdRequest.....	57

MX_fDINT_AccTo_REAL.....	178
MX_fDINT_DistTo_REAL.....	178
MX_fDINT_VelTo_REAL.....	179
MX_fGetDriveWarning.....	57
MX_fGetFreeTicks.....	182
MX_fGetHighResTime.....	183
MX_fHighResTimerTicks_to_us.....	183
MX_fHtoCANl.....	50
MX_fHtoCANs.....	50
MX_fReadParamDINT.....	170
MX_fReadStringParam.....	170
MX_fREAL_AccTo_DINT.....	179
MX_fREAL_DistTo_DINT.....	180
MX_fREAL_TorqueRampTo_DINT.....	180
MX_fREAL_TorqueTo_DINT.....	181
MX_fREAL_VelTo_DINT.....	181
MX_fSetDriveError.....	58
MX_fSetDriveErrorText.....	58
MX_fSetDriveWarning.....	59
MX_fSetDriveWarningText.....	59
MX_fSetParamLimits.....	171
MX_fSetParamName.....	172
MX_fSetParamUnit.....	172
MX_fWriteParamDINT.....	173
MX_fWriteStringParam.....	174
MX_IECTaskGetLoad.....	183
MX_MoveAbsolute.....	116
MX_MoveAdditive.....	118
MX_MoveRelative.....	121
MX_PositionLoopEventInfo.....	184
MX_ReadParamDINT.....	175
MX_SetCmdState.....	60
MX_SetControl.....	124
MX_SetDeviceMode.....	126
MX_SetOpMode.....	127
MX_WriteParamDINT.....	176

O

OBJECTS.....	186
--------------	-----

P

Parameter structure.....	33
Sercos attribute.....	33
Structure.....	33
Performance and cycle time	
IoDrvEtherNetIP.....	234
Processing	
Synchronous.....	230
Processing commands in a PLCopen func-	
tion block.....	17

R

RIL_SercosIII.....	224
Runtime errors	
Signaling / checking.....	53
Runtime event.....	229

S

Scaling.....	178
Semaphores.....	229
SERCOS_TABLE.....	27
Signal generators.....	227
Signal time diagrams.....	17
Sockets.....	229
String processing.....	255
Strings.....	230
Synchronous file access.....	228
SYS_TIME_DATE.....	195
SYS_TIME64.....	194
SysStr23.....	230
SysStrCmp.....	230
SysStrCmpl.....	230
SysStrCmpN.....	230
SysStrCmpNI.....	231
SysStrCpy.....	231
SysStrLen.....	231
System libraries.....	227
SysTimeRtcGet.....	232
SysTimeRtcHighResGet.....	233
SysTimeRtcHighResSet.....	233
SysTimeRtcSet.....	232

T

Time functions.....	255
TOUCHPROBE_REF.....	192
TP_OBJECTS.....	187

W

Watchdog	
IoDrvEtherNetIP.....	234

Notes

Notes

Bosch Rexroth AG

Electric Drives and Controls

P.O. Box 13 57

97803 Lohr, Germany

Bgm.-Dr.-Nebel-Str. 2

97816 Lohr, Germany

Phone +49 9352 18 0

Fax +49 9352 18 8400

www.boschrexroth.com/electrics



R911338916