Rexroth Bosch Group

Rexroth IndraLogic XLC IndraMotion MLC 14VRS Sequential Programming

R911341909
Edition 01

Library

Title Rexroth IndraLogic XLC
IndraMotion MLC 14VRS
Sequential Programming

Type of Documentation Library

Document Typecode DOK-XLCML*-SEQPROG*V14-LI01-EN-P

Internal File Reference RS-47f3e8ec40a037330a6846a50066a82b-1-en-US-6

Purpose of Documentation Documentation of the Sequential Programming Library

Record of Revisions

Edition	Release date	Notes
01	2014-04	First edition

Published by Bosch Rexroth AG

Bgm.-Dr.-Nebel-Str. 2 ■ D-97816 Lohr a. Main

Phone +49 9352 18 0 ■ Fax +49 9352 18 8400

http://www.boschrexroth.com/

System Development Printing and Converting, ChCl, FlZa (FrWe, UrWi)

Table of Contents

Table of Contents

1 About this Documentation

1.1 Validity of the Documentation

Overview on target groups and product phases

In the following illustration, the framed activities, product phases and target groups refer to the present documentation.

Example: In the product phase "Engineering", the target group "Programmers"can execute the activity "programming" using this documentation.

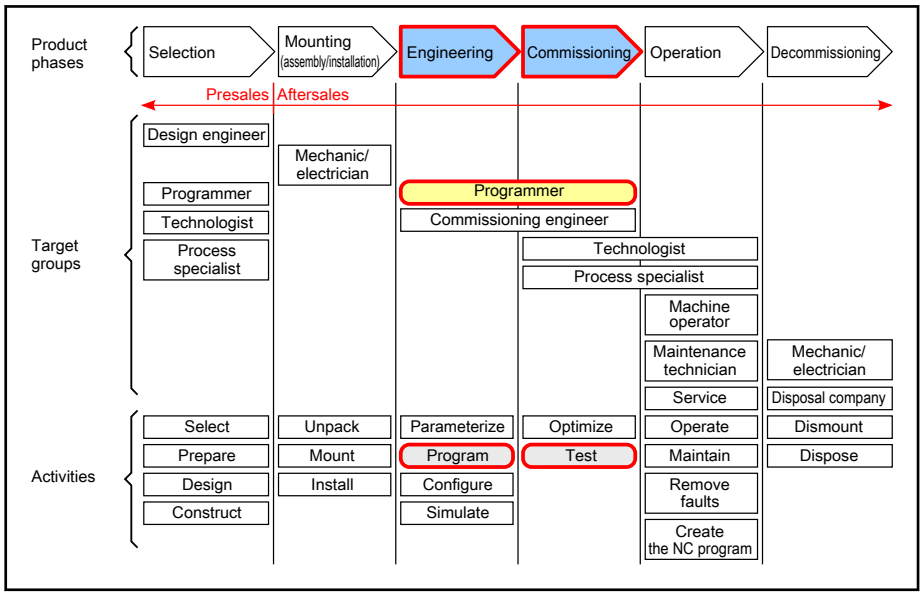

Fig. 1-1: Assigning this documentation to the target groups, product phases and target group activities

1.2 Required and Supplementing Documentations

Documentation titles with type codes and part numbers

System overview
Rexroth IndraLogic XLC 14VRS System Overview DOK-XLC***-SYSTEM**V14-PRRS-EN-P, R911341483 This documentation provides an overview on the possible hardware/software components of the automation system IndraLogic XLC of the mentioned version. The documentation facilitates assembling a system.
Rexroth IndraMotion MLC 14VRS System Overview DOK-MLC***-SYSTEM**V14-PRRS-EN-P, R911341504 This documentation provides an overview of the hardware/software components of the automation system IndraMotion MLC in the mentioned version. The documentation facilitates assembling a system.

Tab. 1-1: XCL/MLC documentation overview - System overview

About this Documentation

First steps
Rexroth IndraMotion MLC 14VRS First Steps
DOK-MLC***-F*STEP**V14-CORS-EN-P, R911341517
This documentation describes the first steps of the IndraMotion MLC and the RobotControl. It includes the hardware and software prerequisites as well as the creation of a project.
Rexroth IndraLogic XLC 14VRS First Steps
DOK-XLC***-F*STEP**V14-CORS-EN-P, R911341489
This documentation describes the first steps of the IndraLogic XLC. It includes the hardware and software prerequisites as well as the creation of a project.
Rexroth IndraMotion MLC 14VRS Project Conversion
DOK-XLCMLC-PROCONV*V14-APRS-EN-P, R911341494
This documentation describes the project conversion of the IndraMotion MLC.

Tab. 1-2: XCL/MLC documentation overview - First steps

Engineering
Rexroth IndraWorks 14VRS Software Installation
DOK-IWORKS-SOFTINS*V14-CORS-EN-P, R911344286
This documentation describes the IndraWorks installation.
Rexroth IndraWorks 14VRS Engineering
DOK-IWORKS-ENGINEE*V14-APRS-EN-P, R911343566
This documentation describes the application of IndraWorks in which the Rexroth Engineering tools are integrated. It includes instructions on how to work with IndraWorks and how to operate the oscilloscope function.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Function Description
DOK-XLCMLC-FUNC****V14-APRS-DE-P, R911341700
This documentation describes wizards, context menus, dialogs, control commissioning, device configuration and functionalities of the IndraMotion MLC.

Tab. 1-3: XCL/MLC documentation overview - Engineering

Diagnostics and service
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Diagnostics
DOK-XLCMLC-DIAG****V14-RERS-EN-P, R911341481
This documentation includes all control parameters implemented in the control systems IndraLogic XLC and IndraMotion MLC.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Parameters
DOK-XLCMLC-PARAM***V14-RERS-EN-P, R911341479
This documentation describes the parameters of the XLC/MLC systems as well as the interaction between parameterization and programming. It includes the axis parameters, control parameters, kinematic parameters, touch probe parameters and programmable limit switch parameters.

Diagnostics and service
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Commissioning DOK-XLCMLC-STARTUP*V14-CORS-EN-P, R911341502 This documentation describes the steps to commission and service the IndraMotion MLC and IndraLogic XLC systems. It includes checklists for frequent tasks and a detailed description of the steps.
Rexroth IndraWorks IndraMotion Service Tool DOK-IWORKS-IMST*******-APRS-DE-P, R911341383 This documentation describes the IndraMotion Service Tool (IMST). It is a web-based diagnostic tool to access the control systems IndraMotion MLC L25, L45 or 65 as well as the MLP via an Ethernet high-speed connection. The IMST allows OEMs, end users and service engineers to access and remotely diagnose a system. A PC with Internet Explorer version 6 or 7 is required.

Tab. 1-4: XLC/MLC documentation overview - Diagnostics and service

Drive Controllers
Rexroth IndraDrive Firmware for Drive Controllers MPH-, MPB-, MPD-, MPC-08 DOK-INDRV*-MP*-08VRS**-APxx-EN-P-DE-P, R911332643 This documentation describes all functional properties of the IndraDrive firmware in the variants MPH-08, MPB-08, MPD-08 and MPC-08.
Rexroth IndraDrive Drive Controllers MPx-02 to MPx-08 DOK-INDRV*-GEN-**VRS**-PAxx-EN-P, R911297317 This documentation describes all parameters implemented in the firmware for drive controllers of the IndraDrive family. It supports the parameterization of the drive controllers.
Rexroth IndraDrive MPx-02 to MPx-08 and HMV DOK-INDRV*-GEN-**VRS**-WAxx-EN-P, R911297319 This documentation describes all diagnostics implemented in the following firmwares: • Drive controller firmwares from MPx-02 to MPx-08 and • firmwares of the supply devices of the type "HMV". It supports the operating crew as well as the programmer during troubleshooting.
Rexroth IndraDrive Control Units CSB02, CSE02, CSH02, CDB02 DOK-INDRV*-CXX02******-PRRS-EN-P, R911338962
Rexroth IndraDrive MPx-18 Functions DOK-INDRV*-MP*-18VRS**-APRS-EN-P, R911338673 This documentation describes all functional properties of the IndraDrive firmware in the variants MPB-18, MPM-18, MPC-18 and MPE-18.

xx Corresponding edition

Tab. 1-5: XCL/MLC documentation overview - Drive controllers

About this Documentation

PLC
Rexroth IndraWorks 14VRS IndraLogic 2G PLC Programming System
DOK-IWORKS-IL2GPRO*V14-APRS-EN-P, R911343571
This documentation describes the PLC programming tool IndraLogic 2G and its usage. It includes the basic use, first steps, visualization, menu items and editors.
Rexroth IndraWorks 14VRS Basic Libraries IndraLogic 2G
DOK-IL*2G*-BASLIB**V14-LIRS-EN-P, R911343920
This documentation describes the system-comprehensive PLC libraries.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS PLCopen Libraries
DOK-XLCMLC-FUNLIB**V14-LIRS-EN-P, R911341491
This documentation describes the function blocks, functions and data types of the RIL_CommonTypes, ML_Base and ML_PLCopen libraries for the IndraLogic XLC/IndraMotion MLC. It also includes the error reactions of function blocks.

Tab. 1-6: XCL/MLC documentation overview - PLC

Field Buses
Rexroth IndraWorks 14VRS Field Buses
DOK-IWORKS-FB******V14-APRS-EN-P, R911341485
This documentation describes the field bus and local periphery connections supported by the IndraLogic XLC, IndraMotion MLC and IndraMotion MTX systems. The focus of this documentation is on the configuration, parameterization, commissioning and the diagnostics of the different periphery connections. This documentation is the basis for the online help.
Rexroth IndraWorks 14VRS Field Buses Libraries
DOK-IWORKS-FB*LIB**V14-LIRS-EN-P, R911343575
This documentation describes the field bus libraries: RIL_ProfibusDP_02, RIL_ProfibusDPSlave, RIL_ProfinetIO, RIL_ProfinetIODevice, RIL_EtherNetIPAdapter, RIL_MappingList, RIL_SERCOSIII, RIL_Inline including their diagnostics and error reactions of the function blocks.
Rexroth IndraWorks 12VRS FDT Container
DOK-IWORKS-FDT*CON*V12-APRS-EN-P, R911334398
This documentation describes the IndraWorks FDT Container functionality. It includes the activation of the functionality in the project and working with DTMs.
SERCOS System Manual for I/O Devices
DOK-CONTRL-ILS3*******-APxx-EN-P, R911333512
This documentation describes the configuration, parameterization, commissioning and diagnostics of I/O devices with a SERCOS interface.

xx Corresponding edition

Tab. 1-7: XCL/MLC documentation overview - Field buses

HMI
Rexroth IndraWorks 14VRS HMI
DOK-IWORKS-HMI*****V14-APRS-EN-P, R911343569
This documentation describes the functions, configuration and operation of the user interfaces IndraWorks HMI Engineering and IndraWorks HMI Operation.
Rexroth IndraWorks 14VRS WinStudio
DOK-IWORKS-WINSTUD*V14-APRS-EN-P, R911341585
This "User Manual and Technical Reference Book" facilitates working with the "Rexroth WinStudio" software for optimal results. This document provides technical information and step-by-step instructions to create web-enabled HMI/SCADA programs.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS HMI Connection
DOK-XLCMLC-HMI*****V14-APRS-EN-P, R911341497
This documentation describes the visualization systems supported by the IndraLogic XLC and IndraMotion MLC and their connection.

Tab. 1-8: XCL/MLC documentation overview - HMI

Technology
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Generic Application Template
DOK-XLCMLC-TF*GAT**V14-APRS-EN-P, R911341487
This documentation provides a structured template to the IndraLogic PLC programmer. This template can be used to add and edit the PLC programming code. It includes the template, the template wizard and example applications.
Rexroth IndraMotion MLC 14VRS Technology Libraries
DOK-MLC***-TF*LIB**V14-LIRS-EN-P, R911341511
This documentation describes the function blocks, functions and data types of the "ML_TechInterface.library", "ML_TechMotion.library", "RMB_TechCam.library" and "ML_TechBase.library". It also includes libraries for the winder functionality, register controller functionality and CrossCutter functionality.
Rexroth IndraWorks Energy Efficiency Management 14VRS
DOK-IWORKS-4EE*****V14-APRS-EN-P, R911339229
This documentation describes the HMI connection of the energy management system and the function blocks, functions and data types of the "RIL_4EE " library. The documentation also includes the error reactions of the function blocks.
Rexroth IndraMotion MLC 14VRS RegisterControl (Library)
DOK-MLC***-REGI*CO*V14-LIRS-EN-P, R911341509
This documentation describes the inputs and outputs of the individual function blocks and provides notes on their use.
Rexroth IndraMotion MLC 14VRS RegisterControl (Application Description)
DOK-MLC***-REGI*CO*V14-APRS-EN-P, R911341507
This documentation describes the application of the integrated register control for a rotogravure printing machine. The components of the mark stream sensor, the HMI application and the error recovery options are described. This instruction provides information on how to operate the register control, react to errors and query diagnostics. This documentation is written for machine setters and machine operators.
Rexroth IndraMotion MLC 14VRS Winder Function Application
DOK-MLC***-TF*WIND*V14-APRS-EN-P, R911341513
This application-related system documentation describes the application of the winder technology functions.
Rexroth IndraWorks 13VRS CamBuilder
DOK-IWORKS-CAMBUIL*V13-APRS-EN-P, R911336291
This documentation describes the basic principles and operation of the CamBuilder, the cam editing tool.

About this Documentation

Technology
Rexroth IndraMotion MLC 14VRS Robot Control V2
DOK-MLC***-ROCO****V14-RERS-EN-P, R911341588
This documentation provides information about the Robot Control V2. The focus is on PLCopen programming. The program structure, variables, functions, motion statements and the required system parameters are described. This documentation also includes the description of the "ML_Robot" library. Robot Control V2 is supported by a kinematic interface. The components of this interface are described in the "ML_KinTech" library.
Rexroth IndraMotion MLC 13VRS RCL Programming Instruction
DOK-MLC***-RCL*PRO*V13-APRS-EN-P, R911336297
This documentation provides information on the RobotControl. The programming language RCL (RobotControl Language) is focused. The program structure, variables, functions, motion statements and the required system parameters are described.

Tab. 1-9: XCL/MLC documentation overview - Technology

Open Core Engineering - OCE
Rexroth IndraLogic XLC IndraMotion MLC 14VRS First Steps MLPI
DOK-XLCMLC-MLPI****V14-CORS-EN-P, R911341033
This document describes the installation and the commissioning of the MLPI interface.
Rexroth IndraLogic XLC IndraMotion MLC 14VRS Automation Interface
DOK-XLCMLC-AUT*INT*V14-APRS-EN-P, R911341499
This documentation describes the script-based access to IndraWorks project data via the interface of the Automation Interface.

Tab. 1-10: XCL/MLC documentation overview - OCE

SafeLogic
Rexroth IndraWorks SafeLogic 14VRS First Steps
DOK-IWORKS-SL*STEP*V14-CORS-EN-P, R911341520
This documentation describes the initial commissioning of the Safety function module and its external Safety periphery. Taking the example of a project, all required steps to commission a simple Safety application are executed.
Rexroth IndraMotion MLC IndraLogic XLC 14VRS SafeLogic System Overview
DOK-XLCMLC-SL**SYS*V14-PRRS-EN-P, R911341696
This documentation describes the project planning, configuration, creation and commissioning of safety-related devices.
Rexroth IndraWorks 14VRS SafeLogic Project Configuration
DOK-IWORKS-SL**PRJ*V14-APRS-EN-P, R911341694
This documentation describes the configuration of Safety applications.
Rexroth IndraControl SafeLogic Function Module
DOK-CONTRL-SL**FM*****-ITRS-DE-P, R911336576
This documentation is provided with the hardware and describes the commissioning, the installation, the decommissioning and disposal of the hardware.

Tab. 1-11: XCL/MLC documentation overview - SafeLogic

Hardware
Rexroth IndraControl L45/L65/L85 Control DOK-CONTRL-ICL45L65L85-PRxx-EN-P, R911332116 This documentation describes the IndraControl L45/L65/L85 controls.
Rexroth IndraControl L25 DOK-CONTRL-IC*L25*****-PRxx-EN-P, R911328474 This documentation describes the IndraControl L25 controls.
Rexroth IndraControl Lxx 14VRS Function Modules DOK-CONTRL-FM*LXX**V14-APRS-EN-P, R911341583 This documentation describes all function modules of the Lxx controls including engineering and diagnostics.

xx Corresponding edition

Tab. 1-12: XCL/MLC documentation overview - Hardware

1.3 Using the Safety Instructions

1.3.1 Safety Instructions – Structure

The safety instructions are structured as follows:

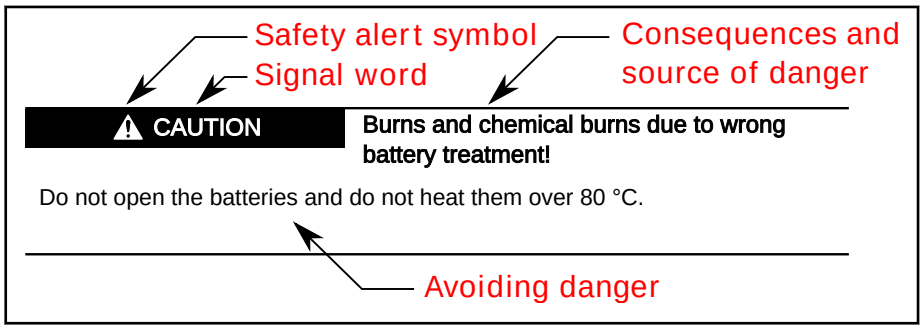

Fig. 1-2: Safety instructions – Structure

1.3.2 Explaining Signal Words and Safety Alert Symbol

The safety instructions in this documentation contain specific signal words (danger, warning, caution, notice) and, if necessary, a safety alert symbol (according to ANSI Z535.6-2006).

The signal word is meant to draw the reader's attention to the safety instruction and signifies the degree of danger.

The safety alert symbol (a triangle with an exclamation point), which precedes the signal words danger,warning and caution is used to alert the reader to personal injury hazards.

⚠ DANGER

In case of non-compliance with this safety instruction, death or serious injury **will** occur.

⚠ WARNING

In case of non-compliance with this safety instruction, death or serious injury **can** occur.

⚠ 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.

1.3.3 Symbols Used

Notes are displayed as follows:

 This is a note.

Tips are displayed as follows:

 This is a tip.

1.4 Names and Abbreviations

Term	Explanation
AxisInterface	The **AxisInterface** pools and extends PLCopen motion function blocks and provides an easy-to-operate interface for the drive functionality
ImcInterface	The **ImcInterface** provides an easy-to-operate interface for the control functionality It includes control signals and parameters for diagnostics and administration of the control
IndraMotion MLC	Compact Motion Logic system with Motion, Robot and Logic Control functionalities
IndraWorks Engineering Framework	Project planning and commissioning tool of Bosch Rexroth
IndraDrive	Drive control unit
HydraulicDrive	Hydraulic drive control unit
Wizard	Wizard guiding the user through several dialogs for an ergonomic data input
Function block	Provides application-specific functionalities
IEC	International Electrotechnical Commission
IAC	Integrated Axis Controller
sercos	sercos (serial realtime communication system) interface is a world-wide standardized interface for the communication between controls and drives

XML	Extensible Markup Language (XML). It is a markup language to represent hierarchically structured data as text data
FKM	Master communication

Tab. 1-13: Names and abbreviations used

1.5 Customer Feedback

Customer requests, comments or suggestions for improvement are of great importance to us. Please email your feedback on the documentations to Feedback.Documentation@boschrexroth.de. Directly insert comments in the electronic PDF document and send the PDF file to Bosch Rexroth.

2 Important Instructions on Use

2.1 Intended Use

2.1.1 Introduction

Bosch Rexroth products are developed and manufactured according to the state-of-the-art. The products are tested prior to delivery to ensure operating safety and reliability.

⚠ WARNING **Personal injury and damage to property due to incorrect use of products!**

The products have been developed for the use in an industrial environment and may only be used as intended. If the products are not used as intended, situations can occur that result in damage to property or personal injury due to an incorrect use of products.

☞ Bosch Rexroth as manufacturer shall not assume any warranty, liability or payment of damages in case of damage resulting from a non-intended use of the products; the user shall solely bear all risks arising from unintended use of the products.

Before using the Bosch Rexroth products, the following requirements have to be met to guarantee the intended use of the products:

- Anybody dealing with Bosch Rexroth products is obliged to read and consent to the relevant safety instructions and the intended use.
- Hardware products may not be altered and have to remain in their original state; i.e., no structural changes are permitted.
- The decompilation of software products or the alteration of source codes is not permitted
- Do not install or operate damaged or faulty products
- It has to be ensured that the products have been installed as described in the relevant documentation

2.1.2 Areas of Use and Application

The products ML_SeqProg of Bosch Rexroth are intended for Motion/Logic applications.

☞ The ML_SeqProg must only be used with the accessories and mounting parts listed in this documentation. Components that are not expressly mentioned must neither be attached nor connected. The same applies to cables and lines.

The operation must only be carried out with the component configurations and combinations that were expressly mentioned and with the software and firmware indicated and specified in the respective functional description.

The ML_SeqProg has been developed for Logic applications as well as for the use in single and multi-axes drive and control tasks. To allow for application-specific requirements of the ML_SeqProg, various device types with different performance and interfaces are provided.

Typical areas of application of the ML_SeqProg are:

- Handling and mounting systems
- Packaging and food machines
- Printing and paper-processing machines
- Presses
- Foundry and rolling technology
- Conveying engineering
- Building automation
- Special purpose machines
- Wood working machinery

The ML_SeqProg may only be operated under the mounting and installation conditions, position and ambient conditions (temperature, degree of protection, moisture, EMC, etc.) specified in this documentation.

2.2 Unintended Use

Using the ML_SeqProg outside of the above-referenced areas of application or under operating conditions other than described in the document and the technical data specified is defined as "unintended".

The ML_SeqProg must not be used if:

- subjected to ML_SeqProg operating conditions that do not comply with the specified ambient conditions. Operation under water, under extreme temperature fluctuations or under extreme maximum temperatures or in explosion-prone areas is prohibited.
- the intended applications have not expressly been released by Bosch Rexroth. Therefore, please read the information given in the general safety instructions!
- the machine or system is provided with an unsecure internet connection. The Ethernet interface of the ML_SeqProg is not securely protected against unauthorized access. Thus, only use the control in secure networks (security). Otherwise, malfunctions or a control failure can be caused due to unauthorized accesses

The operation is prohibited according to the following standards and guidelines:

- Lifts (Directive 95/16/EC)
- ATEX (94/9/EC)
- Burner control systems (EN 298)
- Pressure equipment (EN 764-7:2002/AC)

Furthermore, the use as safety-relevant part of controls according to DIN EN ISO 13849 is considered as unintended use. The functional safety has to be ensured by additional safety-relevant certified components.

3 General Information and Introduction

Introduction

Sequential programming facilitates sequence programming in IndraLogic. The IEC - 61131 (structured text) programming language is used for sequential commands. Thus, no additional programming environment is required, as the editor integrated in IndraWorks can also be used for sequential programming.

Requirements

The library requires the **HYD (hydraulics) function package** of the control.

Interaction with GAT

Sequential programming is completely integrated in the GAT (Generic Application Template) with a decentralized state machine. The GAT wizard was extended for the configuration of sequential programs. Sequential programs can be created, deleted or edited using the wizard. The corresponding axes to execute sequential commands are automatically prepared when a sequential program is created. The following screenshot shows the relevant part of the GAT wizard.

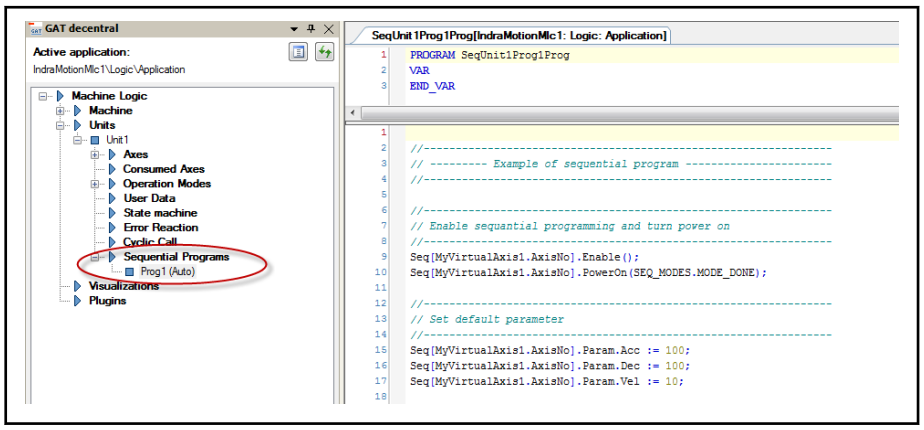

Fig. 3-1: GAT wizard

A sequential program is assigned to an operating mode or a unit.

The GAT wizard makes the following configuration when creating a sequential program:

- A folder is created under the operating mode name in which sequential programs can be filed.
- To be able to start and stop the sequential programs, an instance of the function block "SeqTaskControl" is generated in each case. The instances of this block are inserted in the Global Variable List Global_"UnitName" of the relevant unit. By default, the start and stop command of the sequential program is inserted when entering/exiting the respective operating mode. Consequently, the program is started or stopped when entering or exiting the operating mode.

General Information and Introduction

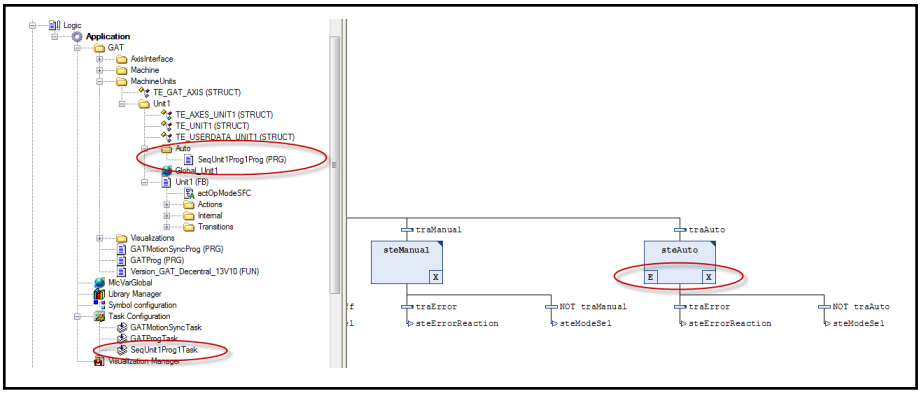

Fig. 3-2: Entering and exiting action by sequential programming

- The command "Seq[…].Disable()" is entered for each unit axis in the exiting step of the operating mode. This command ends the access of sequential programming to the axes when the operating mode is exited.
- Each sequential program is assigned to an individual task.

☞ Up to 20 tasks can be created for the MLC using this application. When creating new sequential programs/tasks, the limit of 20 tasks must not be exceeded. In addition, it has to be noted that all other PLC tasks (PlcTask, MotionTask, etc.) are added to the number of sequential tasks.

☞ Exiting of a sequential program can take several milliseconds. The more sequential programs are assigned to a unit operating mode, the greater the cycle time load when exiting the operating mode. For this reason, the cycle time and the watchdog of the "unit task" may have to be adjusted (by default GATProgTask).

---

4 Interface

4.1 Global Data

Sequential commands/data can be accessed using the global data access. The global data are provided by the ML_SeqProg library. The GAT wizard adds the library when creating a sequential program.

The following instances are relevant when using sequential programming:

Variable name	Type	Description
Seq	ARRAY [1..MB_MAX_AXIS_NUMBER] OF ML_SeqAxisCommand, page17	Sequential axis commands can be called using this instance. This variable has a field structure as each axis has its own commands. The IndraWorks axis number corresponds to the field instance index.
SeqBase	ML_SeqGeneralCommand, page 18	Access to axis-independent commands.
SeqKin	ARRAY [1..MB_MAX_KIN_NUMBER] OF ML_SeqKinCommand, page19	Sequential kinematics functions can be called using this instance. This variable has a field structure as each axis has its own commands. The IndraWorks kinematics number corresponds to the field instance index.
Seq**ProgX**Ctrl	ML_SeqTaskControl, page 19	Controlling of a sequential program "ProgX" is a sample name for a sequential program. Another instance is created by the GAT wizard for each program.

Tab. 4-1: Interfaces of sequential programming

4.2 Function Blocks

4.2.1 ML_SeqAxisCommand

Allocation of target/library

Target	Library
IndraMotion MLC as of version 14VRS Base package HYD	ML_SeqProg

Tab. 4-2: Reference table ML_SeqAxisCommand

The inputs and outputs of the ML_SeqAxisCommand function block are described in the following.

Interface

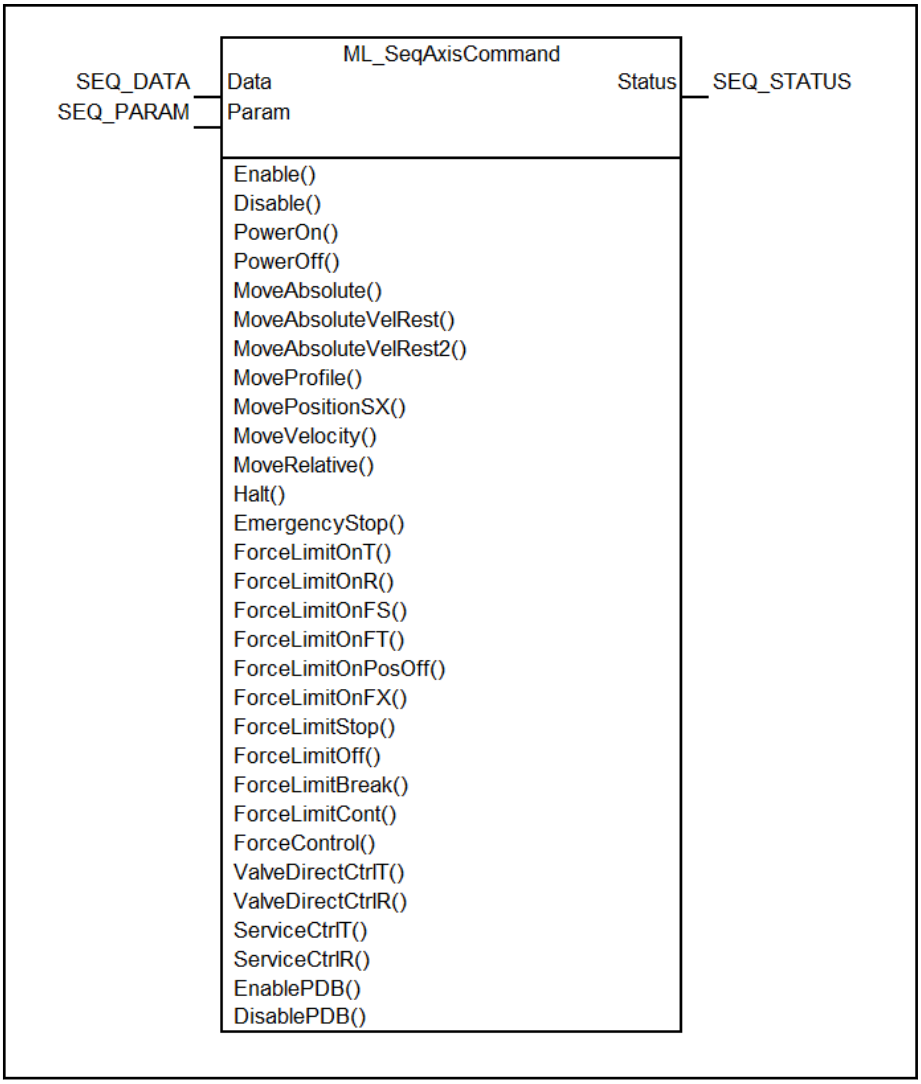

Fig. 4-1: Function block ML_SeqAxisCommand

Name	Data type	Description
Data	SEQ_DATA, page 20	Access to DV_Axis and AxisData structure.
Param	SEQ_PARAM, page 20	Access to parameters used across commands (velocity, accelerations, etc.).
Status	SEQ_STATUS, page 20	Access to axis states (InPos, InPosTarget, etc.)

Tab. 4-3: Data of the ML_SeqAxisCommand block

A detailed description of the sequential axis commands can be found in chapter 5 Overview of commands, page 27.

4.2.2 ML_SeqGeneralCommand

Allocation of target/library

Target	Library
IndraMotion MLC as of version 14VRS Base package HYD	ML_SeqProg

Tab. 4-4: Reference table ML_SeqGeneralCommand

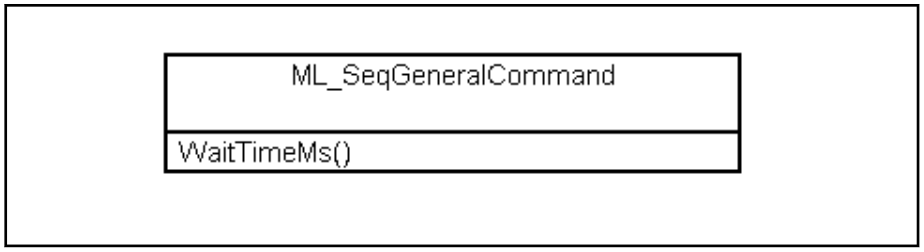

Fig. 4-2: Function block ML_SeqGeneralCommand

A detailed description of general sequential axis commands can be found in chapter 5 Overview of commands, page 27.

4.2.3 ML_SeqKinCommand

Allocation of target/library

Target	Library
IndraMotion MLC as of version 14VRS Base package HYD	ML_SeqProg

Tab. 4-5: Reference table ML_SeqKinCommand

The inputs and outputs of the ML_SeqKinCommand function block are described in the following.

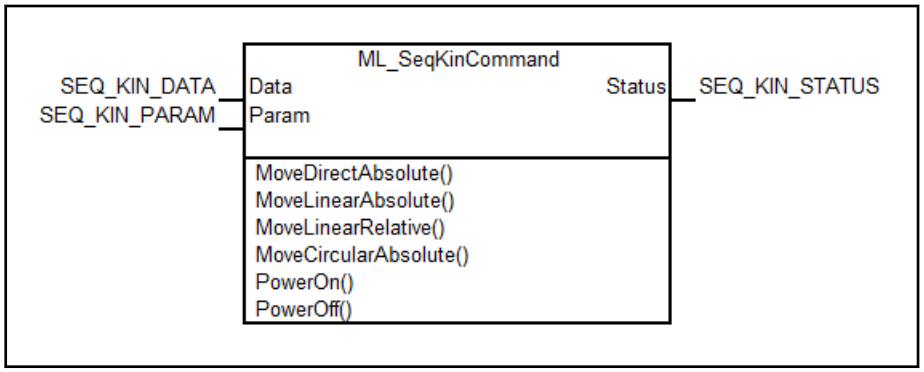

Fig. 4-3: Function block ML_SeqKinCommand

Name	Data type	Description
Data	SEQ_KIN_DATA, page 23	Access to DV_Kin and KinData structure.
Param	SEQ_KIN_PARAM, page 23	Access to parameters used across commands (velocity, accelerations, etc.).
Status	SEQ_KIN_STATUS, page 24	Access to kinematics states (InPosTarget), etc.

Tab. 4-6: Data of the ML_SeqKinCommand block

A detailed description of the sequential kinematics commands can be found in chapter 5 Overview of commands, page 27.

4.2.4 ML_SeqTaskControl

Allocation of target/library

Target	Library
IndraMotion MLC as of version 14VRS Base package HYD	ML_SeqProg

Tab. 4-7: Reference table ML_SeqTaskControl

Interface

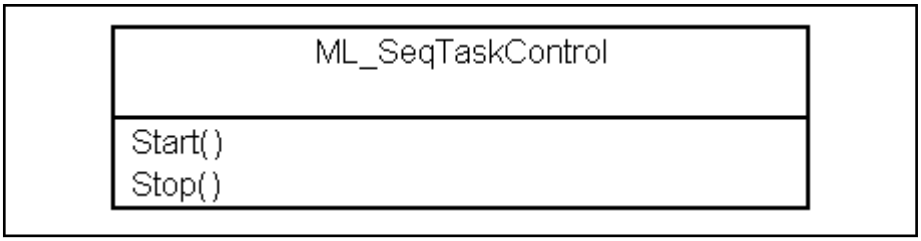

Fig. 4-4: Function block ML_SeqTaskControl

Each sequential program has an instance of this block. The "Start()" and "Stop()" commands are used to start or stop a program. The GAT wizard automatically generates command calls when creating a sequential program.

4.3 Structures

4.3.1 SEQ_DATA

Name	Data type	Description
DV	REFERENCE TO ML_DirectVarAxis	Access to DV_Axis structure.
AxisData	REFERENCE TO ML_AXISDATA_SM	Access to AxisData structure.

Tab. 4-8: SEQ_DATA structure

4.3.2 SEQ_PARAM

General command-independent parameters are saved in a superordinate data structure for sequential programming. The parameters are listed in the table below.

Name	Data type	Description [1]
Acc	REAL	Acceleration
Dec	REAL	Delay
Vel	REAL	Velocity
ForceMode	SEQ_FORCE_MODE, page 21	Force block settings (only for hydraulic axes).
PDB_Mode	SEQ_PDB_MODE, page 22	PDB mode settings (only for hydraulic axes).
PosSX	chapter 4.3.6 "SEQ_POS_SX" on page 22	Command settings MovePositionSX

Tab. 4-9: SEQ_PARAM structure

Implementation example:

Program:

```
// Parameter access acceleration
Seq[AxisNo].Param.Acc := 500;
```

4.3.3 SEQ_STATUS

The sequential programming generates status information for each axis:

Name	Data type	Description	Hydraulic axes only
InPos	BOOL	Actual position value in target window	
InPosTarget	BOOL	Position command value in target	

1) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Interface

ForceLimitOn	BOOL	Alternating control / force limitation active	x
InForceControl	BOOL	Force controller active	x
InForceTarget	BOOL	Force command value in target	x
InVelTarget	BOOL	Velocity command value in target	
InValveTarget	BOOL	Velocity command value in target	x
PowerIsOn	BOOL	Axis in closed-loop mode	
Standstill	BOOL	Axis at standstill	
EnabledPDB	BOOL	PDB status flag	
MoveProfileCurrentIndex	UINT	Active index (MoveProfile)	
VelRestActive	BOOL	Residual velocity active (MoveAbsoluteVelRest and MoveAbsoluteVelRest2)	
Error	BOOL	Error flag	
ErrorID	ERROR_CODE	Error number	
ErrorIdent	ERROR_STRUCT	IDent error	

Tab. 4-10: SEQ_STATUS structure

Implementation example: *Program:*

```
// Check status of axis
IF Seq[AxisNo].Status.InPos THEN

   // implementation...
END_IF
```

4.3.4 SEQ_FORCE_MODE

Name	Data type	Description
Ipo	INT / enumeration chapter 4.4.2 "SEQ_IPO_MODE " on page 24	Interpolation type of force blocks: IPO_CONTINUE (the position command value continues to be generated during force control), IPO_BREAK (the position command value is stopped during force control), IPO_JUMP (once the axis is in force control, the position command value jumps to the target).
Start	INT / enumeration chapter 4.4.3 "SEQ_START_MODES " on page 25	Trigger situation for force blocks: NOW (force blocks immediately start to generate command values) IN_FORCE_CONTROL (force blocks start to generate command values once the axis is in force control)

Interface

Direction	INT / enumeration chapter 4.4.4 "SEQ_FORCE_DIRECTION " on page 25	Direction of action of force blocks: POS_DIRECTION = effective in positive direction of force. Neg_DIRECTION = effective in negative direction of force. BIPOLAR_DIRECTION = effective in both directions of force.
TimeMode	INT / enumeration chapter 4.4.5 "SEQ_TIME_MODES " on page 25	Configuration of the time table of the ForceLimitOnFT force command RELATIVE = the values of the "arTime" input are interpreted as relative values. ABSOLUTE = the values of the "arTime" input are interpreted as absolute values.

Tab. 4-11: SEQ_FORCE_MODE structure

Implementation example: *Program:*

```
// Set force ipo settings
 Seq[AxisNo].Param.ForceMode.Ipo   := SEQ_IPO_MODES.IPO_CONTINUE;
 Seq[AxisNo].Param.ForceMode.Start := SEQ_START_MODES.NOW;
 Seq[AxisNo].Param.ForceMode.Direction := SEQ_FORCE_DIRECTION.POS_DIRECTION;
 Seq[AxisNo].Param.ForceMode.TimeMode := SEQ_TIME_MODE.RELATIVE;
```

4.3.5 SEQ_PDB_MODE

Name	Data type	Description [2)]
BreakStrength	REAL	PDB braking characteristic. Value range = 0-100 (0 = exponential function, 100 = root function)
VelRest	REAL	Residual velocity
PDB_Window	REAL	PDB window

Tab. 4-12: SEQ_PDB_MODE structure

Implementation example: *Program:*

```
// Set settings for position depending breaking
 Seq[AxisNo].Param.PDB_Mode.BreakStrength   := 50;
 Seq[AxisNo].Param.PDB_Mode.VelRest         := 5;
 Seq[AxisNo].Param.PDB_Mode.PDB_Window      := 2;
```

4.3.6 SEQ_POS_SX

Name	Data type	Description [3)]
Mode	MB_POS_MODE_SX	Edit mode: 0 = MB_POS_MODE_SX.ABSOLUTE 1 = MB_POS_MODE_SX.RELATIVE_S 2 = MB_POS_MODE_SX.RELATIVE_X 3 = MB_POS_MODE_SX.RELATIVE_SX
MaxVelocity	REAL	Maximum velocity. The velocity must be less than the motion limit values in parameters A-0-0032 and A-0-0033. [Velocity scaling] 0 = deactivated (default)

2) *The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.*

3) *The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.*

InvertPosition	BOOL	Inversion of position table TRUE = activated FALSE = deactivated
FilterTime	REAL	Filter for command position [ms]
DescendingValues	BOOL	Sorting of values in X-table: FALSE = X-values in ascending order (default) TRUE = X-values in descending order

Tab. 4-13: SEQ_POS_SX structure

Implementation example: *Program:*

```
// Set settings for MovePositionSX
Seq[AxisNo].Param.PosSX.Mode                := MB_POS_MODE_SX.ABSOLUTE; // Mode absolute
Seq[AxisNo].Param.PosSX.MaxVelocity         := 0; // Velocity limitation deactivated
Seq[AxisNo].Param.PosSX.FilterTime          := 0; // Filter deactivated
Seq[AxisNo].Param.PosSX.InvertPosition      := FALSE; // Position inversion deactivated
Seq[AxisNo].Param.PosSX.DescendingValues    := FALSE; // Ascending values in Table "InputXTable"
```

4.3.7 SEQ_KIN_DATA

Name	Data type	Description
DV	REFERENCE TO ML_DirectVarKin	Access to ML_DirectVarKin structure.
KinData	REFERENCE TO ML_KINEMATICDATA_SM	Access to KinData structure.

Tab. 4-14: SEQ_KIN_DATA structure

4.3.8 SEQ_KIN_PARAM

General command-independent kinematic parameters are saved in a superordinate data structure for sequential programming.

The parameters are listed in the table below:

Name	Data type	Description [4)]
Acc	REAL	Acceleration
Dec	REAL	Delay
Vel	REAL	Velocity
Jerk	REAL	Force block settings (only for hydraulic axes).
CoordSystem	ML_COORDSYS	The enumeration type "ML_COORDSYS" defines in which coordinate system the points have to be specified.
SlopeType	ML_SLOPE_TYPE	The enumeration type "ML_SLOPE_TYPE" defines the slope type of the motion.
SetMode	ML_SET_MODE	The enumeration type "ML_SET_MODE"defines whether a motion ends at the end point or whether it is blended.
BlendingRadius	REAL	Blending radius
CircMode	ML_CIRCMODES	The enumeration type "ML_CIRCMODES" defines the interpretation of the "AuxPoint" input

Tab. 4-15: SEQ_KIN_PARAM structure

4) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Interface

Implementation example: *Program:*

```
// Parameter access acceleration
SeqKin[KinNo].Param.Acc := 500;
```

4.3.9 SEQ_KIN_STATUS

The sequential programming generates status information for each kinematics:

Name	Data type	Description
InPosTarget	BOOL	Position command value in target
Standstill	BOOL	Kinematics in the "standstill" state
PowerIsOn	BOOL	Kinematics is active - "complete" state
Error	BOOL	Error flag
ErrorID	ERROR_CODE	Error number
ErrorIdent	ERROR_STRUCT	IDent error

Tab. 4-16: SEQ_KIN_STATUS structure

Implementation example: *Program:*

```
// Check status of axis
IF SeqKin[KinNo].Status.InPosTarget THEN

   // implementation...
END_IF
```

4.4 Enumerations

4.4.1 SEQ_MODES

Description **The enumeration contains the advancing conditions of the sequential commands:**

Member	Value (UINT)	Description
MODE_CONT	1	The command call has no delaying effect. No wait time until processing is completed.
MODE_DONE	2	The command call has a delaying effect. Wait time until processing is completed.

Tab. 4-17: Enumeration SEQ_MODES

4.4.2 SEQ_IPO_MODE

Description **The enumeration contains the interpolation type of the force blocks:**

Member	Value (INT)	Description
IPO_CONTINUE	0	The position command value continues to be generated during force control.
IPO_BREAK	1	During force control the position command value is stopped.
IPO_JUMP	2	When the axis changes over to force control, the position command value jumps to the target.

Tab. 4-18: Enumeration SEQ_IPO_MODE

4.4.3 SEQ_START_MODES

Description **The enumeration contains the starting conditions of the force blocks:**

Member	Value (UINT)	Description
NOW	0	Force blocks immediately start to generate the command values.
IN_FORCE_CONTROL	1	Force blocks start to generate command values once the axis is changing over to force control

Tab. 4-19: Enumeration SEQ_START_MODES

4.4.4 SEQ_FORCE_DIRECTION

Description **The enumeration contains the direction of action of the force blocks:**

Member	Value (INT)	Description
POS_DIRECTION	0	Effective in positive direction.
NEG_DIRECTION	1	Effective in negative direction
BIPOLAR_DIRECTION	2	Effective in positive and negative direction

Tab. 4-20: Enumeration SEQ_FORCE_DIRECTION

4.4.5 SEQ_TIME_MODES

Description **The enumeration contains the configuration of the time table of the ForceLimitOnFT force command:**

Member	Value (INT)	Description
RELATIVE	0	Values of the "arTime" input are interpreted as relative values.
ABSOLUTE	1	Values of the "arTime" input are interpreted as absolute values.

Tab. 4-21: Enumeration SEQ_TIME_MODES

5 Overview of Commands

5.1 Introduction

Requirements

Technology function blocks of the library "ML_SeqProg" require the firmware support of the target system. Specific requirements of the technology function blocks are documented in the chapter of the respective function blocks.

The library requires the **HYD (hydraulics) function package** of the control.

Structure of sequential programs

There are two basic concepts of how sequential programs can be structured. When selecting a concept, it is decisive how often the sequential program is to be executed. In the following, an example of a continued program is given. Programs that can only be executed once, do not require a particular type of handling. Processing is completed after the last sequential command has been executed. Programs that are delayed using a loop, continue to be active until the end criterion of the loop is reached or until they are exited using the command "SeqProgXCtrl.Stop()".

Program:

```
// Enable sequential programming and turn power on
 Seq[AxisNo].Enable();
 Seq[AxisNo].PowerOn();

// Set default parameter
 Seq[AxisNo].Param.Acc := 500;
 Seq[AxisNo].Param.Dec := 500;


 WHILE TRUE  DO

   // Move absolute position 1
   Seq[AxisNo].Param.Vel := 100;
   Seq[AxisNo].MoveAbsolute(200, SEQ_MODES.MODE_DONE);


   // Move absolute position 2
   Seq[AxisNo].Param.Vel := 200;
   Seq[AxisNo].MoveAbsolute(0, SEQ_MODES.MODE_DONE);

 END_WHILE
```

If a "WHILE" loop is used in a sequential program, it has to be ensured that at least **one** delaying command is used. A command has a delaying effect as soon as it is started with "MODE_DONE". The commands "__WAIT(..)" and SeqBase.WaitTimeMs(..) also have a delaying effect.

MODE_CONT handling:

If a sequential command is started in the continue mode, no delay occurs when executing a command. The "MODE_CONT" is used, when several commands have to be started simultaneously.

Program:

```
// Start several move absolutes at same time
Seq[AxisNo1].MoveAbsolute(100, SEQ_MODES.MODE_CONT);
Seq[AxisNo2].MoveAbsolute(200, SEQ_MODES.MODE_CONT);


// Wait until Axis1 and Axis2 are in position

__WAIT(Seq[AxisNo1].Status.InPos AND Seq[AxisNo2].Status.InPos);
```

Overview of Commands

5.2 General Commands

5.2.1 WaitTimeMs

Brief description This command delays the processing of subsequent command for the specified time.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
WaitTime	UDINT	Time in milliseconds.

Tab. 5-1: Inputs of Halt method

Implementation example: *Program:*

```
// Wait for one second
SeqBase.WaitTime(1000);
```

5.2.2 __WAIT

Brief description By means of the **__WAIT** command, waiting for one or several events is possible in a sequential program. The command delays processing of subsequent commands until the specified Boolean expression is fulfilled.

☞ The command call should exclusively be used for sequential programming.

Implementation example: *Program:*

```
// Wait until condition1 and condition2 is set
__WAIT(bCondition1 AND bCondition2);
```

5.3 Axis Commands

5.3.1 General Information

Enable

Brief Description The sequential programming is enabled using this command. The axis interface access to the axis is interrupted. If the axis is already connected at the time of activation and is not in state "Standstill", it is set to standstill (MC_Stop). If the axis is not connected at the time of activation, it is activated for sequential programming without being connected.

☞ The command call should exclusively be used for sequential programming.

Implementation example: *Program:*

```
// Enable sequential progamming
Seq[AxisNo].Enable();
```

Disable

Brief description This command disables the sequential programming. The axis interface access to the axis is restored. If the axis is already connected at the time of activation and is not in state Standstill, it is set to standstill (MC_Stop). If the ax-

is is not connected at the time of activation, it is activated without being connected.

☞ The command call should exclusively be used for sequential programming.

Implementation example: *Program:*

```
// Disable sequential progamming
Seq[AxisNo].Disable();
```

PowerOn

Brief Description Power on is activated using this command. Enabling the block is the precondition for each motion or force limit.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-2: MoveAbsolute inputs

Mode effects **The following table shows the mode effects on the command processing:**

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the axis reports status **PowerOn**.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-3: Mode variants

Implementation example: *Program:*

```
// Sequential command power on
Seq[AxisNo].PowerOn(SEQ_MODES.MODE_DONE);
```

PowerOff

Brief description The command disconnects the axis. The closed-loop control is disabled after the command has been executed.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-4: MoveAbsolute inputs

Overview of Commands

Mode effects

The following table shows the mode effects on the command processing:

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the axis reports status **PowerOff**.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-5: Mode variants

Implementation example:

Program:

```
// Sequential command power off
Seq[AxisNo].PowerOff(SEQ_MODES.MODE_DONE);
```

MoveAbsolute

Brief description

The command uses the absolute specified "position" as "target position" and approaches it. The user can set the method of absolute positioning using the parameters of the SEQ_PARAM, page20 data structure.

The following blocks are optionally used within the command:

- MC_MoveAbsolute
- MH_MoveAbsPDB (positioning with position-dependent braking)

  For detailed function descriptions, please refer to the respective block help.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment[1)]
Pos	REAL	Target position for movement (+/-)
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-6: MoveAbsolute input

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-7: Mode variants

Implementation example:

Program:

```
//-------------------------------------------------
// Settings position depending breaking
// Enable PDB - (optional)
//-------------------------------------------------
```

1) ***The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.***

```
Seq[AxisNo].Param.PDB_Mode.BreakStrength   := 50; // PDB break strength
Seq[AxisNo].Param.PDB_Mode.VelRest         := 5;  // PDB rest velocity
Seq[AxisNo].Param.PDB_Mode.PDB_Window      := 2;  // PDB target window
Seq[AxisNo].EnablePDB();                          // PDB enable


//-------------------------------------------------
// General settings - (optional)
//-------------------------------------------------
Seq[AxisNo].Param.Vel := 100; // Velocity
Seq[AxisNo].Param.Acc := 500; // Acceleration
Seq[AxisNo].Param.Dec := 500; // Deceleration

// Sequential command for absolute positioning
Seq[AxisNo].MoveAbsolute(Pos := 100, Mode := SEQ_MODES.MODE_DONE);
```

MoveAbsoluteVelRest

Brief description

Using the MoveAbsoluteVelRest method, the axis can be traversed to a specified position. After having reached this position, the axis can be moved to the position limit value of the axis (A-0-0030 for positive and A-0-0031 for negative direction) at the specified residual speed (VelRest). The command can be used for rapid speed and creep speed applications.

The following block is used within the command:

- MH_MoveAbsVRest

  For detailed function descriptions, please refer to the respective block help.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment[2)]
Pos	REAL	Target position 1 for movement (+/-)
VelRest	REAL	Velocity, at which the axis moves from target position 1 to the position limit value. (Creep speed) (always +)
Direction	INT	Direction of positioning. 1 = positive direction, -1 = negative direction, 0 = both directions
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-8: MoveAbsoluteVelRest inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-9: Mode variants

2) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Overview of Commands

Implementation example: *MoveAbsoluteVelRest*

```
//-----------------------------------------------
// Settings position depending breaking
// Enable PDB - (optional)
//-----------------------------------------------
 Seq[AxisNo].Param.PDB_Mode.BreakStrength   := 50; // PDB break strength
 Seq[AxisNo].Param.PDB_Mode.VelRest         := 5;  // PDB rest velocity
 Seq[AxisNo].Param.PDB_Mode.PDB_Window      := 2;  // PDB target window
 Seq[AxisNo].EnablePDB();                          // PDB enable


//-----------------------------------------------
// General settings - (optional)
//-----------------------------------------------
Seq[AxisNo].Param.Vel := 100; // Velocity 1 to position 1
Seq[AxisNo].Param.Acc := 500; // Acceleration
Seq[AxisNo].Param.Dec := 500; // Deceleration


// Sequential command for absolute positioning with rest velocity (VelRest = Velocity 2 to end position)
Seq[AxisNo].MoveAbsoluteVelRest(Pos := 100, VelRest := 50, Direction := 0, Mode := SEQ_MODES.MODE_DONE);
```

MoveAbsoluteVelRest2

Brief description

Using the MoveAbsoluteVelRest2 method, the axis can be traversed to a specified position (Pos1). After having reached this position, the axis can be moved to a second position (Pos2) at the specified residual speed (VelRest). The command can be used for rapid speed and creep speed applications.

The following block is used within the command:

- MH_MoveAbsVRest2

  For detailed function descriptions, please refer to the respective block help.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment[3)]
Pos	REAL	Target position 1 for movement (+/-)
Pos2	REAL	Target position 2 for movement (+/-)
VelRest	REAL	Velocity at which the axis moves from target position 1 to target position 2. (Creep speed) (always +)
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-10: MoveAbsoluteVelRest inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-11: Mode variants

3) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Implementation example: *MoveAbsoluteVelRest2*

```
//-----------------------------------------------
// Settings position depending breaking
// Enable PDB - (optional)
//-----------------------------------------------
 Seq[AxisNo].Param.PDB_Mode.BreakStrength   := 50; // PDB break strength
 Seq[AxisNo].Param.PDB_Mode.VelRest         := 5;  // PDB rest velocity
 Seq[AxisNo].Param.PDB_Mode.PDB_Window      := 2;  // PDB target window
 Seq[AxisNo].EnablePDB();                          // PDB enable


//-----------------------------------------------
// General settings - (optional)
//-----------------------------------------------
Seq[AxisNo].Param.Vel := 100; // Velocity 1 to position 1
Seq[AxisNo].Param.Acc := 500; // Acceleration
Seq[AxisNo].Param.Dec := 500; // Deceleration


// Sequential command for absolute positioning with rest velocity (VelRest = Velocity 2 to position 2)
Seq[AxisNo].MoveAbsoluteVelRest2(Pos := 100, Pos2 := 150, VelRest :=  50, Mode := SEQ_MODES.MODE_DONE);
```

MoveProfile

Brief description

The function block traverses the specified axis within a preset profile, which can contain up to ten positions. In conjunction with each position, a new velocity, acceleration and deceleration can be specified. There is a smooth transition between the target positions; i.e. the axis does not stop at the target position, but switches immediately over to the new velocity and moves to the next target until it reaches the end position of the profile.

The following block is used within the command:

- MH_MoveProfile

  For detailed function descriptions, please refer to the respective block help.

  The blending mode is firmly set to value 1 "BlendingLow". In this mode, the next velocity is already applied when the next profile position is reached. A detailed description of the blending mode can also be found in the function block help.

The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment[4)]
Count	UINT	Number of positions within the profile.
arPos	ARRAY (1..10) OF REAL	Field for profile positions.
arVel	ARRAY (1..10) OF REAL	Field for profile velocities.
arAcc	ARRAY (1..10) OF REAL	Field for profile accelerations.
arDec	ARRAY (1..10) OF REAL	Field for profile delays.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-12: MoveProfile inputs

4) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Overview of Commands

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-13: Mode variants

Implementation example:

Program:

```
VAR
  uiCounter   : UINT := 3;
  arPos       : ARRAY [1..10] OF REAL := [0,300,600]; // Position array
  arVel       : ARRAY [1..10] OF REAL := [300,400,500]; // Velocity array
  arAcc       : ARRAY [1..10] OF REAL := [500]; // Acceleration array
  arDec       : ARRAY [1..10] OF REAL := [500];    // Deceleration array
END_VAR

// Sequential command move profile
 Seq[AxisNo].MoveProfile(  Count := uiCounter,
                           arPos := arPos,
                           arVel := arVel,
                           arAcc := arAcc,
                           arDec := arDec,
                           Mode := SEQ_MODES.MODE_DONE
                         );
```

MovePositionSX

Brief description

The MovePositionSX command can be used to provide a command position for the axis in dependence on an input variable. The command positions are determined according to the stored tables "InputXTable" and "PositionTable". The command values are sent cyclically to the axis. Commanding takes place using the function block "ML_WriteCyclicPosition". The user can set further command parameters via the data structure chapter 4.3.2 "SEQ_PARAM" on page 20.

The following block is used within the command:

- MB_MovePositionSX

  For detailed function descriptions, please refer to the respective block help.

The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment[5)]
InputX	REFERENCE TO REAL	Input value X, for which the related command position is interpolated with the help of the position table.
InputTableX	POINTER TO REAL	Pointer to table with input values X. **Note:** Tables “InputXTable” and “PositionTable” must have the same number of elements. Changes made to one of the two tables during processing have a direct influence, because the input tables are used directly internally.
PositionTable	POINTER TO REAL	Pointer to table with actual position values [position scaling]. **Note:** Tables “InputXTable” and “PositionTable” must have the same number of elements. Changes made to one of the two tables during processing have a direct influence, because the input tables are used directly internally.
TableSize	UINT	Table size in bytes. Use, for example, SizeOf(“Variable PositionTable”)

Tab. 5-14: MovePositionSX inputs

Implementation example: *MovePositionSX*

```
VAR

  rValue            : REAL := 0;// Input value
  rInputXTable      : ARRAY [1..50] OF REAL := [0, 10, 20, 30, 40];// X table
  rPositionTable    : ARRAY [1..50] OF REAL := [0, 20, 40, 60, 80];// Position table

END_VAR


//-------------------------------------------------
// General settings - (optional)
//-------------------------------------------------
Seq[AxisNo].Param.PosSX.Mode               := MB_POS_MODE_SX.ABSOLUTE; // Mode absolute
Seq[AxisNo].Param.PosSX.MaxVelocity        := 0; // Velocity limitation deactivated
Seq[AxisNo].Param.PosSX.FilterTime         := 0; // Filter deactivated
Seq[AxisNo].Param.PosSX.InvertPosition     := FALSE; // Position inversion deactivated
Seq[AxisNo].Param.PosSX.DescendingValues   := FALSE; // Ascending values in Table "InputXTable"


// Sequential command for MovePositionSX
Seq[AxisNo].MovePositionSX(InputX        := rValue,
                           InputTableX   := ADR(rInputXTable),
                           PositionTable := ADR(rPositionTable),
                           TableSize     := SIZEOF(rInputXTable));
```

MoveVelocity

Brief description

The command starts an "endless" motion with a specified velocity and uses the acceleration and the deceleration of the data structure SEQ_PARAM20. To end the motion, it has to be interrupted by e.g. MC_Stop.

The following block is used within the command:

- MC_MoveVelocity

5) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Overview of Commands

For detailed function descriptions, please refer to the respective block help.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment[6)]
Vel	REAL	Maximum velocity value (does not necessarily have to be reached).
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-15: MoveVelocity inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command velocity** of the axis is in the target.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-16: Mode variants

Implementation example:

The following examples show commanding of absolute positioning with and without position-dependent braking (PDB):

Program:

```
//-----------------------------------------------
// General settings - (optional)
//-----------------------------------------------
Seq[AxisNo].Param.Acc := 500;
Seq[AxisNo].Param.Dec := 500;


// Sequential command for move velocity
Seq[AxisNo].MoveVelocity(50, SEQ_MODES.MODE_DONE);
```

MoveRelative

Brief Description

The command determines the new "target position" by adding the "distance" to the "actual position" and traverses it.

The following block is used in the command:

- MC_MoveRelative

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment [7)]
Distance	REAL	Relative distance for the motion (+/-).
Mode	SEQ_MODES	Mode for type of command execution

Tab. 5-17: MoveRelative inputs

6) The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

7) The measuring units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Mode effects

The following table shows the mode effects on the command processing

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-18: Mode variants

Implementation example:

Program:

```
//-------------------------------------------------
// General settings - (optional)
//-------------------------------------------------
Seq[AxisNo].Param.Vel := 100;
Seq[AxisNo].Param.Acc := 500;
Seq[AxisNo].Param.Dec := 500;


// Sequential command relative positioning
Seq[AxisNo].MoveRelative(50, SEQ_MODES.MODE_DONE);
```

Halt

Brief description

The command stops the axis in closed-loop control. In the case of hydraulic axes, alternating control is deactivated. If the force limitation has been active, the position command value is based on the actual position value.

The following blocks are used within the command:

- MC_Stop
- MH_ForceLimitOff

  For detailed function descriptions, please refer to the respective block help.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-19: Halt inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-20: Mode variants

Implementation example:

Program:

```
//-------------------------------------------------
```

Overview of Commands

```
// General settings - (optional)
//-----------------------------------------------
Seq[AxisNo].Param.Dec := 500;


// Sequential command relative positioning
Seq[AxisNo].Halt(SEQ_MODES.MODE_DONE);
```

EmergencyStop

Brief description

The command stops the axis in closed-loop control. In the case of hydraulic axes, alternating control is deactivated. If the force limitation has been active, the position command value is based on the actual position value. The maximum axis value is used as delay (A_0_0034).

The following blocks are used within the command:

- MC_Stop
- MH_ForceLimitPosOff

  For detailed function descriptions, please refer to the respective block help.

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-21: EmergencyStop inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command position** of the axis is in the target and the **actual position** is in the target window. The target window "(A-0-2795)" is configured via the axis dialogs.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-22: Mode variants

Implementation example:

Program:

```
// Sequential command relative positioning
Seq[AxisNo].EmergencyStop(SEQ_MODES.MODE_DONE);
```

ForceControl

Brief description

The command enables the force control of the axis. The operating mode cyclic force is executed.

The following block is used in the command:

- MC_TorqueControl

☞ The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment [8)]
Force	REAL	Command force to which it is controlled or blended.
RampValue	REAL	Force per second in which blending takes place.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-23: ForceControl inputs

Mode effects

The following table shows the mode effects on the command processing:

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **command force** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-24: Mode variants

Implementation example:

Program:

```
// Sequential command force control
Seq[AxisNo].ForceControl(50, 100, SEQ_MODES.MODE_DONE);
```

EnablePDB

Brief description

The command activates the mode "open loop with position-dependent braking" (PDB) of the axis. The German designation for PDB is "wegabhängiges Bremsen" (WAB). PDB triggers the drive to execute all positioning commands in the "Open Loop Mode" and to approach the position using position-dependent braking. This mode can also be enabled during a motion using this command.

☞ The command call should exclusively be used for sequential programming.

Implementation example:

Program:

```
// Settings position depending breaking (PDB)

 Seq[AxisNo].Param.PDB_Mode.BreakStrength   := 50; // PDB break strength
 Seq[AxisNo].Param.PDB_Mode.VelRest         := 5;  // PDB rest velocity
 Seq[AxisNo].Param.PDB_Mode.PDB_Window      := 2;  // PDB target window


// Enable PDB
Seq[AxisNo].EnablePDB();
```

DisablePDB

Brief description

The command deactivates the mode "open loop with position-dependent braking" (PDB) of the axis The German designation for PDB is "wegabhängiges Bremsen" (WAB). Deactivating PDB triggers the drive to execute all positioning commands in the "Closed Loop Mode". This mode can also be disabled during a motion using this command.

8) The measuring units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

☞ The command call should exclusively be used for sequential programming.

Implementation example: *Program:*

```
// Disable position dependent breaking (PDB)
Seq[AxisNo].DisablePDB();
```

5.3.2 Hydraulics

ForceLimitOnT

Brief description

The command activates alternating control (force limitation) and cross-fades the force command value to the target value according to the specified time The current force command value of the controller is used as initial value. The force command value is directly set as target value in case of a cross-fading time of 0.

The following block is used within the command:

- MH_ForceLimitOn

  For detailed function descriptions, please refer to the respective block help.

☞ The command call may exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment[9)]
Force	REAL	Command force, which is used for controlling or cross-fading.
TimeValue	REAL	Time in seconds, within which cross-fading is to take place.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-25: ForceLimitOnT inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The delay type can be parameterized via the "Seq[AxisNo].Param.ForceMode.Start" structure: • SEQ_START_MODES.NOW (force command value in target) • SEQ_START_MODES.IN_FORCE_CONTROL (force command value in target and force controller active)
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-26: Mode variants

9) The units of time, position, velocity, acceleration, deceleration and force depend on the effective scaling.

Implementation example: *Program:*

```
// Set force ipo settings (optional)
Seq[AxisNo].Param.ForceMode.Ipo   := SEQ_IPO_MODES.IPO_CONTINUE;
Seq[AxisNo].Param.ForceMode.Start := SEQ_START_MODES.NOW;
Seq[AxisNo].Param.ForceMode.Direction := SEQ_FORCE_DIRECTION.POS_DIRECTION;


// Sequential command activation force limit
Seq[AxisNo].ForceLimitOnT(Force := 50, TimeValue := 0, Mode := SEQ_MODES.MODE_DONE);
```

ForceLimitOnR

Brief description

The command activates alternating control (force limitation) and cross-fades the force command value to the target value according to the specified ramp. The current force command value of the controller is used as initial value. The force command value is directly set to the target value in case of a cross-fading time of 0.

The following block is used within the command:

- MH_ForceLimitOn

  For detailed function descriptions, please refer to the respective block help.

☞ The command call may exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment[10)]
Force	REAL	Command force, which is used for controlling or cross-fading.
RampValue	REAL	Force per second, in which cross-fading takes place.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-27: ForceLimitOnR inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The delay type can be parameterized via the "Seq[AxisNo].Param.ForceMode.Start" structure: • SEQ_START_MODES.NOW (force command value in target) • SEQ_START_MODES.IN_FORCE_CONTROL (force command value in target and force controller active)
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-28: Mode variants

10) The units of time, position, velocity, acceleration, deceleration and force depend on the effective scaling.

Overview of Commands

Implementation example: *Program:*

```
// Set force ipo settings (optional)
Seq[AxisNo].Param.ForceMode.Ipo   := SEQ_IPO_MODES.IPO_CONTINUE;
Seq[AxisNo].Param.ForceMode.Start := SEQ_START_MODES.NOW;
Seq[AxisNo].Param.ForceMode.Direction := SEQ_FORCE_DIRECTION.POS_DIRECTION;


// Sequential command activation force limit
Seq[AxisNo].ForceLimitOnR(Force := 50, RampValue := 0, Mode := SEQ_MODES.MODE_DONE);
```

ForceLimitOnFS

Brief description The function activates alternating control of the axis. A force command value table is generated depending on a force/position table.

The following block is used within the command:

- MH_ForceLimitOnFS

  For detailed function descriptions, please refer to the respective block help.

☞ The command call may exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment[11)]
arPos	ARRAY [1..50] OF REAL	Position values for X-axis of linear interpolation (input values)
arForce	ARRAY [1..50] OF REAL	Position values for Y-axis of linear interpolation (output values)

Tab. 5-29: ForceLimitOnFS inputs

Implementation example: *Program:*

```
VAR
  arPos      : ARRAY [1..50] OF REAL := [0,500,600]; // Position array
  arForce    : ARRAY [1..50] OF REAL := [0,100,200]; // Force array
END_VAR


  // Set force settings (optional)
  Seq[AxisNo].Param.ForceMode.Ipo   := SEQ_IPO_MODES.IPO_CONTINUE;
  Seq[AxisNo].Param.ForceMode.Direction := SEQ_FORCE_DIRECTION.POS_DIRECTION;


   // Sequential command activation force profile
   Seq[AxisNo].ForceLimitOnFS(arPos := arPos, arForce := arForce);
```

ForceLimitOnFT

Brief description The function activates alternating control of the axis. A force command value curve is generated depending on a force/time table.

The following block is used within the command:

- MH_ForceLimitOnFT

  For detailed function descriptions, please refer to the respective block help.

11) The units of time, position, velocity, acceleration, deceleration and force depend on the effective scaling.

The command call may exclusively be used for sequential programming.

The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment[12)]
arTime	ARRAY [1..50] OF REAL	Time values for X-axis of linear interpolation (input values) The values are given in seconds.
arForce	ARRAY [1..50] OF REAL	Position values for Y-axis of linear interpolation (output values)
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-30: ForceLimitOnFT inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The delay type can be parameterized via the "Seq[AxisNo].Param.ForceMode.Start" structure: • SEQ_START_MODES.NOW (force command value in target) • SEQ_START_MODES.IN_FORCE_CONTROL (force command value in target and force controller active)
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-31: Mode variants

Implementation example: *Program:*

```
VAR
  arTime      : ARRAY [1..50] OF REAL := [0,5,10];// Time array
  arForce     : ARRAY [1..50] OF REAL := [0,100,200];// Force array
END_VAR


  // Set force ipo settings (optional)
  Seq[AxisNo].Param.ForceMode.Ipo   := SEQ_IPO_MODES.IPO_CONTINUE;
  Seq[AxisNo].Param.ForceMode.Start := SEQ_START_MODES.NOW;
  Seq[AxisNo].Param.ForceMode.Direction := SEQ_FORCE_DIRECTION.POS_DIRECTION;
  Seq[AxisNo].Param.ForceMode.TimeMode := SEQ_TIME_MODE.RELATIVE;


  // Sequential command activation force profile
  Seq[AxisNo].ForceLimitOnFT(arTime := arTime, arForce := arForce, Mode := SEQ_MODES.MODE_DONE);
```

ForceLimitOnFX

Brief description

The function activates alternating control of the axis. A force command value curve is generated depending on a force/input variable table.

The following block is used within the command:

- MH_ForceLimitOnFX

  For detailed function descriptions, please refer to the respective block help.

12) The units of time, position, velocity, acceleration, deceleration and force depend on the effective scaling.

Overview of Commands

☞ The command call may exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment[13)]
ValueX	REFERENCE TO REAL	Reference to an optional input variable
arX	ARRAY [1..50] OF REAL	Values for X-axis of linear interpolation (input values)
arForce	ARRAY [1..50] OF REAL	Force values for Y-axis of linear interpolation (output values)

Tab. 5-32: ForceLimitOnFX inputs

Implementation example: *Program:*

```
VAR
  rValue       : REAL := 450;// Input value
  arX          : ARRAY [1..50] OF REAL := [0,400,500];// X array
  arForce      : ARRAY [1..50] OF REAL := [0,100,200];// Force array
END_VAR


  // Set force settings (optional)
  Seq[AxisNo].Param.ForceMode.Ipo   := SEQ_IPO_MODES.IPO_CONTINUE;
  Seq[AxisNo].Param.ForceMode.Direction := SEQ_FORCE_DIRECTION.POS_DIRECTION;


   // Sequential command activation force profile
   Seq[AxisNo].ForceLimitOnFX(ValueX := rValue, arX := arX, arForce := arForce);
```

ForceLimitStop

Brief description Stops the generation of the force command value. The alternating control remains active.

The following block is used in the command:

- MH_ForceLimitStop

☞ The command call should exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Implementation example: *Program:*

```
   // Sequential command for stopping force interpolation
   Seq[AxisNo].ForceLimitStop();
```

ForceLimitOff

Brief description The alternating control (force limit) is disabled and the generation of the force command value is stopped. If the force limit was active, the position command value is set to the actual position value.

The following block is used in the command:

- MH_ForceLimitOff

13) The units of time, position, velocity, acceleration, deceleration and force depend on the effective scaling.

A detailed function description is Project Explorer the block wizard.

☞ The command call should exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution

Tab. 5-33: ForceLimitStop input

Mode effects

The following table shows the mode effects on the command processing

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the axis reports status **ForceLimitOn**.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-34: Mode variants

Implementation example:

Program:

```
// Sequential command switching off force limit
Seq[AxisNo].ForceLimitOff(SEQ_MODES.MODE_DONE);
```

ForceLimitBreak

Brief description

Interrupts the generation of the force command value. The alternating control (force limit) remains active. The command value generation is started using the ForceLimitCont command. The following block is used in the command:

- MH_ForceLimitBreak

☞ The command call should exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Implementation example:

Program:

```
// Sequential command for interrupting force interpolation
Seq[AxisNo].ForceLimitBreak();
```

ForceLimitCont

Brief description

After the command value generation was interrupted (ForceLimitBreak), the force command value calculation can be continued using the ForceLimitCont command.

The following block is used in the command:

- MH_ForceLimitCont

☞ The command call should exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Overview of Commands

Implementation example: *Program:*

```
// Sequential command for continuing force interpolation
Seq[AxisNo].ForceLimitCont();
```

ValveDirectCtrlT

Brief description The method enables the operating mode valve direct control. The command value is crossfaded according to the specified time. The valve control value is directly set to the target value in case of a crossfading time of 0.

The following block is used in the command:

- MH_AnalogRamp

☞ The command call should exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment [14)]
AnalogValue	REAL	Analog control value
TimeValue	REAL	Time in seconds during which blending takes place.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-35: ValveDirectCtrlT inputs

Mode effects **The following table shows the mode effects on the command processing:**

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **valve control values** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-36: Mode variants

Implementation example: *Program:*

```
// Sequential command valve direct control
Seq[AxisNo].ValveDirectCtrlT(10, 0, SEQ_MODES.MODE_DONE);
```

ValveDirectCtrlR

Brief description The method enables the operating mode direct valve control. The command value is crossfaded according to the specified ramp. The valve control value is directly set to the target value in case of a blending ramp of 0.

The following block is used in the command:

- MH_AnalogRamp

☞ The command call should exclusively be used for sequential programming.

14) The measuring units of the analog control values depends on the active scaling. (Valve scaling)

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment [15]
AnalogValue	REAL	Analog control value
RampValue	REAL	Ramp used for blending. Valve unit per second.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-37: ValveDirectCtrlr inputs

Mode effects

The following table shows the mode effects on the command processing

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **valve control values** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-38: Mode variants

Implementation example:

Program:

```
// Sequential command valve direct control
Seq[AxisNo].ValveDirectCtrlR(10, 0, SEQ_MODES.MODE_DONE);
```

ServiceCtrlT

Brief description

The command describes the valve control value of a hydraulic controller axis. The command value is crossfaded according to the specified time. The valve control value is directly set to the target value in case of a crossfading time of 0.

The following block is used in the command:

- MH_ServiceCtrl

☞ The command call should exclusively be used for sequential programming.

☞ The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment [16]
AnalogValue	REAL	Analog control value
TimeValue	REAL	Time in seconds during which blending takes place.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-39: ServiceCtrlT inputs

15) The measuring units of the analog control values depends on the active scaling. (Valve scaling)

16) The measuring units of the analog control values depends on the active scaling. (Valve scaling)

Overview of Commands

Mode effects

The following table shows the mode effects on the command processing:

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **valve control values** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-40: Mode variants

Implementation example:

Program:

```
// Sequential command valve direct control (service)
Seq[AxisNo].ServiceCtrlT(10, 0, SEQ_MODES.MODE_DONE);
```

ServiceCtrlR

Brief description

The command describes the valve control value of a hydraulic controller axis. The command value is crossfaded according to the specified ramp. The valve control value is directly set to the target value in case of a blending ramp of 0.

The following block is used in the command:

- MH_ServiceCtrl

The command call should exclusively be used for sequential programming.

The command is only available for hydraulic controller axes.

Inputs

Name	Type	Comment [17)]
AnalogValue	REAL	Analog control value
RampValue	REAL	Ramp used for blending. Valve unit per second.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-41: ServiceCtrlR inputs

Mode effects

The following table shows the mode effects on the command processing:

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **valve control values** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-42: Mode variants

Implementation example:

Program:

```
// Sequential command valve direct control (service)
Seq[AxisNo].ServiceCtrlR(10, 0, SEQ_MODES.MODE_DONE);
```

17) The measuring units of the analog control values depends on the active scaling. (Valve scaling)

5.4 Kinematic Commands

5.4.1 PowerOn (Kinematics)

Brief description The command adds the configured axes to a kinematics.

The following block is used in the command:

- ML_GroupAllAxes

The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution

Tab. 5-43: PowerOn inputs

Mode effects **The following table shows the mode effects on the command processing:**

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the kinematics reports status **Incomplete**.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-44: Mode variants

Implementation example: *Program:*

```
// Sequential kinematic command power on
SeqKin[KinNo].PowerOn(SEQ_MODES.MODE_DONE);
```

5.4.2 PowerOff (Kinematics)

Brief description The command removes all kinematic axes.

The following block is used in the command:

- MC_UngroupAllAxes

The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment
Mode	SEQ_MODES	Mode for type of command execution

Tab. 5-45: PowerOff inputs

Mode effects **The following table shows the mode effects on the command processing:**

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the kinematics reports status **Complete**.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-46: Mode variants

Implementation example: *Program:*

```
// Sequential kinematic command power off
SeqKin[KinNo].PowerOff(SEQ_MODES.MODE_DONE);
```

5.4.3 MoveDirectAbsolute

Brief description The command moves a robot point by point from the current position of the TCP to a specified absolute target position in the coordinate system.

The following block is used in the command:

- MC_MoveDirectAbsolute

The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment [18)]
Position	ARRAY [1..16] OF REAL	Target position of the motion.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-47: MoveVelocity inputs

Mode effects **The following table shows the mode effects on the command processing:**

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **command positions** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-48: Mode variants

Implementation example: *MoveDirectAbsolute*

```
VAR

  arrPosition      : ARRAY [1..16] OF REAL := [300,300];

END_VAR


//-------------------------------------------------
// General settings - (optional)
//-------------------------------------------------
SeqKin[KinNo].Param.Acc := 500;
SeqKin[KinNo].Param.Dec := 500;
SeqKin[KinNo].Param.Vel := 10;


// Sequential command for move direct absolute
SeqKin[KinNo].MoveDirectAbsolute(arrPosition, SEQ_MODES.MODE_DONE);
```

5.4.4 MoveCircularAbsolute

Brief description The command moves a robot circularly from the current position of the TCP via an intermediate point to a specified absolute target position in the given coordinate system.

The following block is used within the command:

- MC_MoveCircularAbsolute

18) The measuring units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

For detailed function descriptions, please refer to the respective block help.

The command call may exclusively be used for sequential programming.

Inputs

Name	Type	Comment[19)]
AuxPoint	ARRAY[1..16] OF REAL	Intermediate position of movement
EndPoint	ARRAY[1..16] OF REAL	Target position of movement
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-49: MoveCircularAbsolute inputs

Mode effects

The following table shows the mode effects on command processing:

Name	Comment
MODE_DONE	The call has a delaying effect. The method delays processing until the **command positions** of the kinematics are in the target.
MODE_CONT	The call has no delaying effect. Command processing is continued without delay.

Tab. 5-50: Mode variants

Implementation example:

MoveCircularAbsolute

```
VAR

  arrAuxPoint      : ARRAY [1..16] OF REAL := [0,300,600];
  arrEndPoint      : ARRAY [1..16] OF REAL := [300,400,500];

END_VAR


//-------------------------------------------------
// General settings - (optional)
//-------------------------------------------------
SeqKin[KinNo].Param.Acc := 500;
SeqKin[KinNo].Param.Dec := 500;
SeqKin[KinNo].Param.Vel := 10;

// Sequential command for move circular absolute
SeqKin[KinNo].MoveCircularAbsolute(arrAuxPoint, arrEndPoint, SEQ_MODES.MODE_DONE);
```

5.4.5 MoveLinearAbsolute

Brief description

The command moves a robot linearly from the current position of the TCP to a specified absolute target position in the coordinate system.

The following block is used in the command:

- MC_MoveLinearAbsolute

The command call should exclusively be used for sequential programming.

19) *The units of time, position, velocity, acceleration and deceleration depend on the effective scaling.*

Overview of Commands

Inputs

Name	Type	Comment [20)]
Position	ARRAY [1..16] OF REAL	Absolute target position of the motion.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-51: MoveVelocity inputs

Mode effects **The following table shows the mode effects on the command processing:**

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **command positions** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-52: Mode variants

Implementation example: *MoveLinearAbsolute*

```
VAR
  arrPosition      : ARRAY [1..16] OF REAL := [300,300];
END_VAR

//-----------------------------------------------
// General settings - (optional)
//-----------------------------------------------
SeqKin[KinNo].Param.Acc := 500;
SeqKin[KinNo].Param.Dec := 500;
SeqKin[KinNo].Param.Vel := 10;

// Sequential command for move linear absolute
SeqKin[KinNo].MoveLinearAbsolute(arrPosition, SEQ_MODES.MODE_DONE);
```

5.4.6 MoveLinearRelative

Brief description The command moves a robot linearly from the last commanded position of the TCP to a specified relative target position in the coordinate system.

The following block is used in the command:

- MC_MoveLinearRelative

The command call should exclusively be used for sequential programming.

Inputs

Name	Type	Comment [21)]
Position	ARRAY [1..16] OF REAL	Relative target position of the motion.
Mode	SEQ_MODES	Mode for type of command execution.

Tab. 5-53: MoveVelocity inputs

20) The measuring units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

21) The measuring units of time, position, velocity, acceleration and deceleration depend on the effective scaling.

Mode effects

The following table shows the mode effects on the command processing

Name	Comment
MODE_DONE	The call is delayed. The method delays processing until the **command positions** of the axis is in the target.
MODE_CONT	The call is not delayed. Command processing is continued without delay.

Tab. 5-54: Mode variants

Implementation example: *MoveLinearRelative*

```
VAR

  arrPosition      : ARRAY [1..16] OF REAL := [300,300];

END_VAR


//-----------------------------------------------
// General settings - (optional)
//-----------------------------------------------
SeqKin[KinNo].Param.Acc := 500;
SeqKin[KinNo].Param.Dec := 500;
SeqKin[KinNo].Param.Vel := 10;


// Sequential command for move linear relative
SeqKin[KinNo].MoveLinearRelative(arrPosition, SEQ_MODES.MODE_DONE);
```

6 Service and Support

Our worldwide service network provides an optimized and efficient support. Our experts offer you advice and assistance should you have any queries. You can contact us **24/7**.

Service Germany

Our technology-oriented Competence Center in Lohr, Germany, is responsible for all your service-related queries for electric drive and controls.

Contact the **Service Helpdesk & Hotline** under:

Phone:	**+49 9352 40 5060**
Fax:	**+49 9352 18 4941**
E-mail:	service.svc@boschrexroth.de
Internet:	http://www.boschrexroth.com

Additional information on service, repair (e.g. delivery addresses) and training can be found on our internet sites.

Service worldwide

Outside Germany, please contact your local service office first. For hotline numbers, refer to the sales office addresses on the internet.

Preparing information

To be able to help you more quickly and efficiently, please have the following information ready:

- Detailed description of malfunction and circumstances resulting in the malfunction
- Type plate name of the affected products, in particular type codes and serial numbers
- Your contact data (phone and fax number as well as your email address)

Index

Index

Notes

Notes

Notes

Rexroth
Bosch Group

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Tel. +49 9352 18 0
Fax +49 9352 18 8400
www.boschrexroth.com/electrics

R911341909

DOK-XLCML*-SEQPROG*V14-LI01-EN-P