Rexroth
Bosch Group

Rexroth IndraMotion MLC 12VRS Technology Libraries

R911333868
Edition 02

Library

Title
Rexroth IndraMotion MLC
12VRS
Technology Libraries

Type of Documentation
Library

Document Typecode
DOK-MLC***-TF*LIB**V12-LI02-EN-P

Internal File Reference
RS-ed9bd9627e6498430a6846a000f81475-2-en-US-4

Purpose of Documentation
This documentation describes the function blocks, functions and data types of the libraries

- ML_TechBase.compiled-library
- ML_TechMotion.compiled-library
- RMB_TechCam.compiled-library

as well as libraries for the winder functionality, register controller functionality and CrossCutter functionality.

Record of Revision

Edition	Release Date	Notes
Edition 01	03.2011	First edition for 12VRS
Edition 02	08.2011	Supplements, corrections

Published by
Bosch Rexroth AG
Bgm.-Dr.-Nebel-Str. 2 ■ 97816 Lohr a. Main, Germany
Phone +49 (0)93 52/ 40-0 ■ Fax +49 (0)93 52/ 40-48 85
http://www.boschrexroth.com/
System Development Automation Motion Logic Control, DT (KaWa/MePe)

Note
This document has been printed on chlorine-free bleached paper.

Table of Contents

Table of Contents

Table of Contents

Page

Table of Contents

Table of Contents

Page

Table of Contents

Table of Contents

Page

Table of Contents

1 About this Documentation

1.1 Validity of the Documentation

Target group

This documentation describes the usage of functions and function blocks from the technology libraries.

This documentation supports the user during the following phases:

- Engineering and
- Commissioning

1.2 Documentation Structure

The first part of the document includes important instructions on use and safety (chapter 2 "Important Instructions on Use" on page 17 and chapter 3 "Safety Instructions for Electric Drives and Controls" on page 19).

chapter 4 "Overview " on page 31 includes a list of the libraries in this documentation.

chapter 5 "ML_TechBase.library" on page 33 includes descriptions of the technology function blocks.

chapter 6 "ML_TechMotion.library" on page 159 includes descriptions of the TechMotion function blocks.

chapter 7 "ML_TechRegi.library/MX_TechRegi.lib" on page 285 describes the function blocks available for the register control.

chapter 8 "ML_TechCrank.library" on page 371 describes functions for crank and bell-crank kinematics.

chapter 9 "RMB_TechWinder.library/MX_TechWinder.lib" on page 409 describes the function blocks used for winding and unwinding webs (winders).

chapter 10 "RMB_TechCrossCutCrossSeal.library" on page 497 describes the function blocks for the CrossSealer and CrossCutter functionality.

chapter 11 "RMB_TechCam.library" on page 527 describes the function blocks to create cam tables.

chapter 12 "RIL_ParameterChannel.library" on page 549 includes a description of the function block for the acyclic communication via a cyclic channel of a field bus connection.

chapter 13 "ML_TechTensionAdvanced.library" on page 571 describes the function blocks for multi-axis tension controllers with decoupling network that are **subject to license**.

chapter 14 "Service and Support" on page 597 includes information on the Bosch Rexroth customer service help desk.

1.3 Required and Supplementing Documentations

Documentation titles with type designation codes and parts numbers

IndraWorks		MLC	XLC
/36/	**Rexroth IndraWorks 12VRS Software Installation** DOK-IWORKS-SOFTINS*V12-COxx-EN-P, R911334396 This documentation describes the IndraWorks installation.	X	X

About this Documentation

/5/	**Rexroth IndraWorks 12VRS Engineering** DOK-IWORKS-ENGINEE*V12-APxx-EN-P, R911334388 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.	X	X
/20/	**Rexroth IndraMotion MLC 12VRS Functional Description** DOK-MLC***-FUNC****V12-APxx-EN-P, R911333848 This documentation describes wizards, context menus, dialogs, control commissioning, device configuration and functionalities of the IndraMotion MLC.	X	
/20/	**Rexroth IndraLogic XLC 12VRS Functional Description** DOK-XLC***-FUNC****V12-APxx-EN-P, R911333878 This documentation describes wizards, context menus, dialogs, control commissioning, device configuration and functionalities of the IndraLogic XLC.		X
/7/	**Rexroth IndraWorks 12VRS CamBuilder** DOK-IWORKS-CAMBUIL*V12-APxx-EN-P, R911333842 This documentation describes the basic principles and operation of the CamBuilder, the cam editing tool.	X	X
/37/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS Automation Interface** DOK-XLCMLC-AUT*INT*V12-APxx-EN-P, R911334178 This documentation describes the script-based access to IndraWorks project data via the interface of the Automation Interface.	X	X
/38/	**Rexroth IndraWorks 12VRS FDT Container** DOK-IWORKS-FDT*CON*V12-APxx-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.	X	X
/29/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS Project Conversion** DOK-XLCMLC-PROCONV*V12-APxx-EN-P, R911334187 This documentation described the project conversion of IndraLogic 04VRS and IndraMotion MLC04VRS on IndraWorks Version 12 with IndraLogic 2G. It especially focuses on changes in the field of Motion and PLC.	X	X
/28/	**Rexroth IndraMotion MLC 12VRS Commissioning** DOK-MLC***-STARTUP*V12-COxx-EN-P, R911333858 This documentation describes the steps required for commissioning and performing service on the IndraMotion MLC system. It includes checklists for tasks to be frequently performed and a detailed description of the steps.	X	

Motion		**MLC**	**XLC**
/23/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS PLCopen Libraries** DOK-XLCMLC-FUNLIB**V12-LIxx-EN-P, R911334182 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.	X	X

/27/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS Generic Application Template** DOK-XLCMLC-TF*GAT**V12-APxx-EN-P, R911334191 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.	X	X
/31/	**Rexroth IndraMotion MLC 12VRS RCL Programming Instruction** DOK-MLC***-RCL*PRO*V12-APxx-EN-P, R911333852 This documentation provides information on the RobotControl. It is given most importance to the programming language RCL (RobotControl Language). The program structure, variables, functions, motion statements and the required system parameters are described.	X	
/21/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS Parameters** DOK-XLCMLC-PARAM***V12-RExx-EN-P, R911334176 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.	X	X
/10/	**Rexroth IndraDrive; Firmware for Drive Controllers MPH, MPB, MPD, MPC-07** DOK-INDRV*-MP*-07VRS**-FKxx-EN-P, R911328670		
/11/	**Rexroth IndraDrive MPx-16 Functions** DOK-INDRV*-MP*-16VRS**-APxx-EN-P, R911326767		

Field buses		**MLC**	**XLC**
/39/	**Rexroth IndraMotion MLC 11VRS PLCopen Field Bus** DOK-IM*ML*-PLCFBUS*V11-APxx-EN-P, R911333896 This documentation describes the creation of field bus drives in an IndraWorks project, function blocks, functions and data types of the libraries RIL_CommonTypes.library (excerpt for field bus drives), RMB_PLCopenFieldBus.library, RIL_Utilities.library (excerpt for field bus drives). It also includes the error reactions of function blocks.	X	X
/4/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS Field Buses** DOK-XLCMLC-FB******V12-APxx-EN-P, R911334394 This documentation describes the supported field buses and their diagnostic function blocks.	X	X

HMI		**MLC**	**XLC**
/8/	**Rexroth IndraWorks 12VRS HMI** DOK-IWORKS-HMI*****V12-APxx-EN-P, R911334392 This documentation describes the functions, configuration and operation of the user interfaces IndraWorks HMI Engineering and IndraWorks HMI Operation.	X	X
/6/	**Rexroth IndraWorks 12VRS WinStudio** DOK-IWORKS-WINSTUD*V12-APxx-EN-P, R911333844 This documentation describes the installation of the software, working with WinStudio and the creation and operation of applications.	X	X
/50/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS HMI Connection** DOK-XLCMLC-HMI*****V12-APxx-EN-P, R911334184 This documentation describes the visualization systems supported by the IndraLogic XLC and IndraMotion MLC and their connection.	X	X

About this Documentation

PLC		MLC	XLC
/3/	**Rexroth IndraWorks 12VRS IndraLogic 2G Programming Instruction** DOK-IWORKS-IL2GPRO*V12-APxx-EN-P, R911334390 This documentation describes the PLC programming tool IndraLogic 2G and its usage. It includes the basic usage, first steps, visualization, menu items and editors.	X	X
/33/	**Rexroth IndraWorks 12VRS, Basic Libraries, IndraLogic 2G** DOK-IL*2G*-BASLIB**V12-LIxx-EN-P, R911333835 This documentation describes the system-comprehensive PLC libraries.	X	X

Technology			
/30/	**Rexroth IndraMotion MLC 12VRS Technology Libraries** DOK-MLC***-TF*LIB**V12-LIxx-EN-P, R911333868 This documentation describes the function blocks, functions and data types of the libraries "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.	X	
/60/	**Rexroth IndraMotion MLC 12VRS RegisterControl (Library)** DOK-MLC***-REGI*CO*V12-LIxx-EN-P, R911333856 This documentation describes the inputs and outputs of the individual function blocks and provides notes on their usage.	X	
/62/	**Rexroth IndraMotion MLC 12VRS RegisterControl (Application Manual)** DOK-MLC***-REGI*CO*V12-APxx-EN-P, R911333854 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 on errors and query diagnostics. This documentation is written for machine setters and machine operators.	X	
/49/	**Rexroth IndraMotion MLC 12VRS Application of Winder Functions** DOK-MLC***-TF*WIND*V12-APxx-EN-P, R911333870 This application-related system documentation describes the usage of the winder technology functions.	X	

Hardware		MLC	XLC
/1/	**Rexroth IndraControl L45/L65** DOK-CONTRL-IC*L45*L65*-PRxx-EN-P, R911324661	X	X
/2/	**Rexroth IndraControl L25** DOK-CONTRL-IC*L25*****-PRxx-EN-P, R911328474	X	X
/24/	**Rexroth IndraControl Lxx 12VRS Function Modules** DOK-CONTRL-FM*LXX**V12-APxx-EN-P, R911333830 This documentation describes all function modules of the Lxx controls including engineering and diagnostics.	X	X
/12/	**Rexroth IndraDrive Drive Controllers MPx-02 to MPx-07** DOK-INDRV*-GEN-**VRS**-PAxx-EN-P, R911297317		

/13/	**Rexroth IndraDrive MPx-02 to MPx-07 and HMV** DOK-INDRV*-GEN-**VRS**-WAxx-EN-P, R911297319		
/35/	**Rexroth IndraDrive Drive Controller Control Sections CSB01, CSH01, CDB01** DOK-INDRV*-CSH********-PR08-EN-P, R911295012		

Diagnostics and Service		**MLC**	**XLC**
/26/	**Rexroth IndraMotion MLC/XLC 11VRS Service Tool** DOK-IM*ML*-IMST****V11-RExx-EN-P, R911331940	X	X
/22/	**Rexroth IndraLogic XLC IndraMotion MLC 12VRS Diagnostics** DOK-XLCMLC-DIAG****V12-RExx-EN-P, R911334180 This documentation includes all control parameters implemented in the control systems IndraLogic XLC and IndraMotion MLC.	X	X

System Overview		**MLC**	**XLC**
/48/	**Rexroth IndraMotion for Printing 12VRS System Overview** DOK-IM*PR*-SYSTEM**V11-PRxx-EN-P, R911333840 This documentation describes the product IndraMotion for Packaging. It introduces the control systems, drive systems and I/O systems as well as the commissioning and programming.	X	
/48/	**Rexroth IndraMotion for Packaging 12VRS System Overview** DOK-IM*PA*-SYSTEM**V12-PRxx-EN-P, R911333838 This documentation describes the product IndraMotion for Packaging. It introduces the control systems, drive systems and I/O systems as well as the commissioning and programming.	X	
/9/	**Rexroth IndraMotion MLC 12VRS System Overview** DOK-MLC***-SYSTEM**V12-PRxx-EN-P, R911333860 This documentation provides an overview on the possible hardware/software components of the automation system IndraMotion MLC of the named version. It helps assembling a system.	X	
/9/	**Rexroth IndraLogic XLC 12VRS System Overview** DOK-XLC***-SYSTEM**V12-PRxx-EN-P, R911333880 This documentation provides an overview on the possible hardware/software components of the automation system IndraLogic XLC of the named version. It helps assembling a system.		X

First Steps		**MLC**	**XLC**
/25/	**Rexroth IndraMotion MLC 12VRS First Steps** DOK-MLC***-F*STEP**V12-COxx-EN-P, R911333846 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.	X	
/25/	**Rexroth IndraLogic XLC 12VRS First Steps** DOK-XLC***-F*STEP**V12-COxx-EN-P, R911333876 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.		X

xx Respective edition

Fig.1-1: XCL/MLC documentation overview

1.4 Representing Information

1.4.1 Safety Instructions

The safety instructions, if contained in the documentation, contain certain signal words("Danger", "Warning", "Caution", "Note") and sometimes a signal symbol (according to ANSI Z535.6-2006).

The signal word should direct the attention to the safety instructions. It indicates the severity of the hazard.

The signal symbol (triangular safety reflector with three exclamation marks), preceding the signal words "Danger", "Warning", "Caution" indicates a hazard for persons.

Description of safety notes in this documentation:

⚠ 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 **could** 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 damages can occur.

1.4.2 Used Symbols

Representation The representation of different symbols can be found under chapter 3.4 Explanation of Signal Words and the Safety Alert Symbol, page 28.

1.4.3 Names an Abbreviations

Term	Explanation
Libraries	Includes functions and function blocks
Tech-FB	Technology function block
PLS	Program Limit Switch
PLCopen	was founded in 1992 and is an international, manufacturer- and product-independent interest group of control manufacturers, software companies and institutes. Mandatory to the PLC standard IEC 61131-3, different standards are defined in technical committees facilitating a significant increase in efficiency in the engineering of the application software.

Target system	Control (e.g. IndraMotion MLC)
SYNAX	Automation systems for printing and paper-processing machinery
VisualMotion	Programmable multi-axis motion control system
IndraMotion MLC	Compact Motion Logic system with Motion, Robot and Logic Control functionality
IndraLogic XLC	Compact PLC system with integrated motion functionality
IndraMotion MLD	Drive-based Motion Logic system
IndraWorks	Configuration and commissioning tool by Bosch Rexroth
IndraDrive	Drive and control device
MA	IndraMotion MLD Advanced
MLD-S	Motion Logic Drive (single drive)
MLD-M	Motion Logic Drive (multi drive)
I.e.	This means that
e.g.,	for example
n.def.	not defined
cf.	compare
s	Second
ms	Millisecond
m	meter
mm	millimeter
%	percent

Fig.1-2: Names and abbreviations

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 they are not used as intended, situations occur that can result in damage to property and injury to persons.

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

- Anybody dealing with Bosch Rexroth Products in any way 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 Scope of Use and Application

The products IndraMotion MLC/IndraLogic XLC of Bosch Rexroth are intended for Motion/Logic applications.

☞ The IndraMotion MLC/IndraLogic XLC must only be used together with the accessories and attachment parts given in this documentation. Components that are not expressly mentioned must neither be attached nor connected. The same is valid for 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 IndraMotion MLC/IndraLogic XLC has been developed for Logic applications as well as for the usage in single and multi-axes drive and control tasks.

Important Instructions on Use

To allow for application-specific requirements of the IndraMotion MLC/ IndraLogic XLC, various device types with different performance and interfaces are provided.

Typical areas of application of the IndraMotion MLC/IndraLogic XLC are:

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

The IndraMotion MLC/IndraLogic XLC 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

Usage of the IndraMotion MLC/IndraLogic XLC in applications areas other than those specified or described in the documentation and technical data is considered as "unintended".

The IndraMotion MLC/IndraLogic XLC must not be used if

- they are subjected to operating conditions not corresponding to 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!

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 Safety Instructions for Electric Drives and Controls

3.1 Definitions of Terms

Application Documentation Application documentation comprises the entire documentation used to inform the user of the product about the use and safety-relevant features for configuring, integrating, installing, mounting, commissioning, operating, maintaining, repairing and decommissioning the product. The following terms are also used for this kind of documentation: User Guide, Operation Manual, Commissioning Manual, Instruction Manual, Project Planning Manual, Application Manual, etc.

Component A component is a combination of elements with a specified function, which are part of a piece of equipment, device or system. Components of the electric drive and control system are, for example, supply units, drive controllers, mains choke, mains filter, motors, cables, etc.

Control System A control system comprises several interconnected control components placed on the market as a single functional unit.

Device A device is a finished product with a defined function, intended for users and placed on the market as an individual piece of merchandise.

Electrical Equipment Electrical equipment encompasses all devices used to generate, convert, transmit, distribute or apply electrical energy, such as electric motors, transformers, switching devices, cables, lines, power-consuming devices, circuit board assemblies, plug-in units, control cabinets, etc.

Electric Drive System An electric drive system comprises all components from mains supply to motor shaft; this includes, for example, electric motor(s), motor encoder(s), supply units and drive controllers, as well as auxiliary and additional components, such as mains filter, mains choke and the corresponding lines and cables.

Installation An installation consists of several devices or systems interconnected for a defined purpose and on a defined site which, however, are not intended to be placed on the market as a single functional unit.

Machine A machine is the entirety of interconnected parts or units at least one of which is movable. Thus, a machine consists of the appropriate machine drive elements, as well as control and power circuits, which have been assembled for a specific application. A machine is, for example, intended for processing, treatment, movement or packaging of a material. The term "machine" also covers a combination of machines which are arranged and controlled in such a way that they function as a unified whole.

Manufacturer The manufacturer is an individual or legal entity bearing responsibility for the design and manufacture of a product which is placed on the market in the individual's or legal entity's name. The manufacturer can use finished products, finished parts or finished elements, or contract out work to subcontractors. However, the manufacturer must always have overall control and possess the required authority to take responsibility for the product.

Product Examples of a product: Device, component, part, system, software, firmware, among other things.

Project Planning Manual A project planning manual is part of the application documentation used to support the sizing and planning of systems, machines or installations.

Qualified Persons In terms of this application documentation, qualified persons are those persons who are familiar with the installation, mounting, commissioning and operation of the components of the electric drive and control system, as well as with the hazards this implies, and who possess the qualifications their work

requires. To comply with these qualifications, it is necessary, among other things,

1) to be trained, instructed or authorized to switch electric circuits and devices safely on and off, to ground them and to mark them

2) to be trained or instructed to maintain and use adequate safety equipment

3) to attend a course of instruction in first aid

User A user is a person installing, commissioning or using a product which has been placed on the market.

3.2 General Information

3.2.1 Using the Safety Instructions and Passing Them on to Others

Do not attempt to install and operate the components of the electric drive and control system without first reading all documentation provided with the product. Read and understand these safety instructions and all user documentation prior to working with these components. If you do not have the user documentation for the components, contact your responsible Bosch Rexroth sales partner. Ask for these documents to be sent immediately to the person or persons responsible for the safe operation of the components.

If the component is resold, rented and/or passed on to others in any other form, these safety instructions must be delivered with the component in the official language of the user's country.

Improper use of these components, failure to follow the safety instructions in this document or tampering with the product, including disabling of safety devices, could result in property damage, injury, electric shock or even death.

3.2.2 Requirements for Safe Use

Read the following instructions before initial commissioning of the components of the electric drive and control system in order to eliminate the risk of injury and/or property damage. You must follow these safety instructions.

- Bosch Rexroth is not liable for damages resulting from failure to observe the safety instructions.
- Read the operating, maintenance and safety instructions in your language before commissioning. If you find that you cannot completely understand the application documentation in the available language, please ask your supplier to clarify.
- Proper and correct transport, storage, mounting and installation, as well as care in operation and maintenance, are prerequisites for optimal and safe operation of the component.
- Only qualified persons may work with components of the electric drive and control system or within its proximity.
- Only use accessories and spare parts approved by Bosch Rexroth.
- Follow the safety regulations and requirements of the country in which the components of the electric drive and control system are operated.
- Only use the components of the electric drive and control system in the manner that is defined as appropriate. See chapter "Appropriate Use".
- The ambient and operating conditions given in the available application documentation must be observed.
- Applications for functional safety are only allowed if clearly and explicitly specified in the application documentation "Integrated Safety Technolo-

gy". If this is not the case, they are excluded. Functional safety is a safety concept in which measures of risk reduction for personal safety depend on electrical, electronic or programmable control systems.

- The information given in the application documentation with regard to the use of the delivered components contains only examples of applications and suggestions.

  The machine and installation manufacturers must

  – make sure that the delivered components are suited for their individual application and check the information given in this application documentation with regard to the use of the components,

  – make sure that their individual application complies with the applicable safety regulations and standards and carry out the required measures, modifications and complements.

- Commissioning of the delivered components is only allowed once it is sure that the machine or installation in which the components are installed complies with the national regulations, safety specifications and standards of the application.
- Operation is only allowed if the national EMC regulations for the application are met.
- The instructions for installation in accordance with EMC requirements can be found in the section on EMC in the respective application documentation.

  The machine or installation manufacturer is responsible for compliance with the limit values as prescribed in the national regulations.

- The technical data, connection and installation conditions of the components are specified in the respective application documentations and must be followed at all times.

National regulations which the user must take into account

- European countries: In accordance with European EN standards
- United States of America (USA):
  – National Electrical Code (NEC)
  – National Electrical Manufacturers Association (NEMA), as well as local engineering regulations
  – Regulations of the National Fire Protection Association (NFPA)
- Canada: Canadian Standards Association (CSA)
- Other countries:
  – International Organization for Standardization (ISO)
  – International Electrotechnical Commission (IEC)

3.2.3 Hazards by Improper Use

- High electrical voltage and high working current! Danger to life or serious injury by electric shock!
- High electrical voltage by incorrect connection! Danger to life or injury by electric shock!
- Dangerous movements! Danger to life, serious injury or property damage by unintended motor movements!
- Health hazard for persons with heart pacemakers, metal implants and hearing aids in proximity to electric drive systems!

Safety Instructions for Electric Drives and Controls

- Risk of burns by hot housing surfaces!
- Risk of injury by improper handling! Injury by crushing, shearing, cutting, hitting!
- Risk of injury by improper handling of batteries!
- Risk of injury by improper handling of pressurized lines!

3.3 Instructions with Regard to Specific Dangers

3.3.1 Protection Against Contact With Electrical Parts and Housings

☞ This section concerns components of the electric drive and control system with voltages of **more than 50 volts**.

Contact with parts conducting voltages above 50 volts can cause personal danger and electric shock. When operating components of the electric drive and control system, it is unavoidable that some parts of these components conduct dangerous voltage.

High electrical voltage! Danger to life, risk of injury by electric shock or serious injury!

- Only qualified persons are allowed to operate, maintain and/or repair the components of the electric drive and control system.
- Follow the general installation and safety regulations when working on power installations.
- Before switching on, the equipment grounding conductor must have been permanently connected to all electric components in accordance with the connection diagram.
- Even for brief measurements or tests, operation is only allowed if the equipment grounding conductor has been permanently connected to the points of the components provided for this purpose.
- Before accessing electrical parts with voltage potentials higher than 50 V, you must disconnect electric components from the mains or from the power supply unit. Secure the electric component from reconnection.
- With electric components, observe the following aspects:

  Always wait **30 minutes** after switching off power to allow live capacitors to discharge before accessing an electric component. Measure the electrical voltage of live parts before beginning to work to make sure that the equipment is safe to touch.
- Install the covers and guards provided for this purpose before switching on.
- Never touch electrical connection points of the components while power is turned on.
- Do not remove or plug in connectors when the component has been powered.
- Under specific conditions, electric drive systems can be operated at mains protected by residual-current-operated circuit-breakers sensitive to universal current (RCDs/RCMs).
- Secure built-in devices from penetrating foreign objects and water, as well as from direct contact, by providing an external housing, for example a control cabinet.

High housing voltage and high leakage current! Danger to life, risk of injury by electric shock!

- Before switching on and before commissioning, ground or connect the components of the electric drive and control system to the equipment grounding conductor at the grounding points.

- Connect the equipment grounding conductor of the components of the electric drive and control system permanently to the main power supply at all times. The leakage current is greater than 3.5 mA.
- Establish an equipment grounding connection with a minimum cross section according to the table below. With an outer conductor cross section smaller than 10 mm^2 (8 AWG), the alternative connection of two equipment grounding conductors is allowed, each having the same cross section as the outer conductors.

Cross section outer conductor	Minimum cross section equipment grounding conductor Leakage current ≥ 3.5 mA	
	1 equipment grounding conductor	2 equipment grounding conductors
1,5 mm^2 (AWG 16)	10 mm^2 (AWG 8)	2 × 1,5 mm^2 (AWG 16)
2,5 mm^2 (AWG 14)		2 × 2,5 mm^2 (AWG 14)
4 mm^2 (AWG 12)		2 × 4 mm^2 (AWG 12)
6 mm^2 (AWG 10)		2 × 6 mm^2 (AWG 10)
10 mm^2 (AWG 8)		-
16 mm^2 (AWG 6)	16 mm^2 (AWG 6)	-
25 mm^2 (AWG 4)		-
35 mm^2 (AWG 2)		-
50 mm^2 (AWG 1/0)	25 mm^2 (AWG 4)	-
70 mm^2 (AWG 2/0)	35 mm^2 (AWG 2)	-
...	...	...

Fig.3-1: Minimum Cross Section of the Equipment Grounding Connection

3.3.2 Protective Extra-Low Voltage as Protection Against Electric Shock

Protective extra-low voltage is used to allow connecting devices with basic insulation to extra-low voltage circuits.

On components of an electric drive and control system provided by Bosch Rexroth, all connections and terminals with voltages between 5 and 50 volts are PELV ("Protective Extra-Low Voltage") systems. It is allowed to connect devices equipped with basic insulation (such as programming devices, PCs, notebooks, display units) to these connections.

Danger to life, risk of injury by electric shock! High electrical voltage by incorrect connection!

If extra-low voltage circuits of devices containing voltages and circuits of more than 50 volts (e.g., the mains connection) are connected to Bosch Rexroth products, the connected extra-low voltage circuits must comply with the requirements for PELV ("Protective Extra-Low Voltage").

3.3.3 Protection Against Dangerous Movements

Dangerous movements can be caused by faulty control of connected motors. Some common examples are:

- Improper or wrong wiring or cable connection
- Operator errors
- Wrong input of parameters before commissioning
- Malfunction of sensors and encoders
- Defective components
- Software or firmware errors

These errors can occur immediately after equipment is switched on or even after an unspecified time of trouble-free operation.

The monitoring functions in the components of the electric drive and control system will normally be sufficient to avoid malfunction in the connected drives. Regarding personal safety, especially the danger of injury and/or property damage, this alone cannot be relied upon to ensure complete safety. Until the integrated monitoring functions become effective, it must be assumed in any case that faulty drive movements will occur. The extent of faulty drive movements depends upon the type of control and the state of operation.

Dangerous movements! Danger to life, risk of injury, serious injury or property damage!

A **risk assessment** must be prepared for the installation or machine, with its specific conditions, in which the components of the electric drive and control system are installed.

As a result of the risk assessment, the user must provide for monitoring functions and higher-level measures on the installation side for personal safety. The safety regulations applicable to the installation or machine must be taken into consideration. Unintended machine movements or other malfunctions are possible if safety devices are disabled, bypassed or not activated.

To avoid accidents, injury and/or property damage:

- Keep free and clear of the machine’s range of motion and moving machine parts. Prevent personnel from accidentally entering the machine’s range of motion by using, for example:
  - Safety fences
  - Safety guards
  - Protective coverings
  - Light barriers
- Make sure the safety fences and protective coverings are strong enough to resist maximum possible kinetic energy.
- Mount emergency stopping switches in the immediate reach of the operator. Before commissioning, verify that the emergency stopping equipment works. Do not operate the machine if the emergency stopping switch is not working.
- Prevent unintended start-up. Isolate the drive power connection by means of OFF switches/OFF buttons or use a safe starting lockout.
- Make sure that the drives are brought to safe standstill before accessing or entering the danger zone.

- Additionally secure vertical axes against falling or dropping after switching off the motor power by, for example,
  - mechanically securing the vertical axes,
  - adding an external braking/arrester/clamping mechanism or
  - ensuring sufficient counterbalancing of the vertical axes.
- The standard equipment **motor holding brake** or an external holding brake controlled by the drive controller is **not sufficient to guarantee personal safety**!
- Disconnect electrical power to the components of the electric drive and control system using the master switch and secure them from reconnection ("lock out") for:
  - Maintenance and repair work
  - Cleaning of equipment
  - Long periods of discontinued equipment use
- Prevent the operation of high-frequency, remote control and radio equipment near components of the electric drive and control system and their supply leads. If the use of these devices cannot be avoided, check the machine or installation, at initial commissioning of the electric drive and control system, for possible malfunctions when operating such high-frequency, remote control and radio equipment in its possible positions of normal use. It might possibly be necessary to perform a special electromagnetic compatibility (EMC) test.

3.3.4 Protection Against Magnetic and Electromagnetic Fields During Operation and Mounting

Magnetic and electromagnetic fields generated by current-carrying conductors or permanent magnets of electric motors represent a serious danger to persons with heart pacemakers, metal implants and hearing aids.

Health hazard for persons with heart pacemakers, metal implants and hearing aids in proximity to electric components!

- Persons with heart pacemakers and metal implants are not allowed to enter the following areas:
  - Areas in which components of the electric drive and control systems are mounted, commissioned and operated.
  - Areas in which parts of motors with permanent magnets are stored, repaired or mounted.
- If it is necessary for somebody with a heart pacemaker to enter such an area, a doctor must be consulted prior to doing so. The noise immunity of implanted heart pacemakers differs so greatly that no general rules can be given.
- Those with metal implants or metal pieces, as well as with hearing aids, must consult a doctor before they enter the areas described above.

3.3.5 Protection Against Contact With Hot Parts

Hot surfaces of components of the electric drive and control system. Risk of burns!

- Do not touch hot surfaces of, for example, braking resistors, heat sinks, supply units and drive controllers, motors, windings and laminated cores!
- According to the operating conditions, temperatures of the surfaces can be **higher than 60 °C** (140 °F) during or after operation.
- Before touching motors after having switched them off, let them cool down for a sufficient period of time. Cooling down can require **up to 140 minutes**! The time required for cooling down is approximately five times the thermal time constant specified in the technical data.
- After switching chokes, supply units and drive controllers off, wait **15 minutes** to allow them to cool down before touching them.
- Wear safety gloves or do not work at hot surfaces.
- For certain applications, and in accordance with the respective safety regulations, the manufacturer of the machine or installation must take measures to avoid injuries caused by burns in the final application. These measures can be, for example: Warnings at the machine or installation, guards (shieldings or barriers) or safety instructions in the application documentation.

3.3.6 Protection During Handling and Mounting

Risk of injury by improper handling! Injury by crushing, shearing, cutting, hitting!

- Observe the relevant statutory regulations of accident prevention.
- Use suitable equipment for mounting and transport.
- Avoid jamming and crushing by appropriate measures.
- Always use suitable tools. Use special tools if specified.
- Use lifting equipment and tools in the correct manner.
- Use suitable protective equipment (hard hat, safety goggles, safety shoes, safety gloves, for example).
- Do not stand under hanging loads.
- Immediately clean up any spilled liquids from the floor due to the risk of falling!

3.3.7 Battery Safety

Batteries consist of active chemicals in a solid housing. Therefore, improper handling can cause injury or property damage.

Risk of injury by improper handling!

- Do not attempt to reactivate low batteries by heating or other methods (risk of explosion and cauterization).
- Do not attempt to recharge the batteries as this may cause leakage or explosion.
- Do not throw batteries into open flames.
- Do not dismantle batteries.
- When replacing the battery/batteries, do not damage the electrical parts installed in the devices.
- Only use the battery types specified for the product.

☞ Environmental protection and disposal! The batteries contained in the product are considered dangerous goods during land, air, and sea transport (risk of explosion) in the sense of the legal regulations. Dispose of used batteries separately from other waste. Observe the national regulations of your country.

3.3.8 Protection Against Pressurized Systems

According to the information given in the Project Planning Manuals, motors and components cooled with liquids and compressed air can be partially supplied with externally fed, pressurized media, such as compressed air, hydraulics oil, cooling liquids and cooling lubricants. Improper handling of the connected supply systems, supply lines or connections can cause injuries or property damage.

Risk of injury by improper handling of pressurized lines!

- Do not attempt to disconnect, open or cut pressurized lines (risk of explosion).
- Observe the respective manufacturer's operating instructions.
- Before dismounting lines, relieve pressure and empty medium.
- Use suitable protective equipment (safety goggles, safety shoes, safety gloves, for example).
- Immediately clean up any spilled liquids from the floor due to the risk of falling!

☞ Environmental protection and disposal! The agents (e.g., fluids) used to operate the product might not be environmentally friendly. Dispose of agents harmful to the environment separately from other waste. Observe the national regulations of your country.

3.4 Explanation of Signal Words and the Safety Alert Symbol

The Safety Instructions in the available application documentation contain specific signal words (DANGER, WARNING, CAUTION or NOTICE) and, where required, a safety alert symbol (in accordance with ANSI Z535.6-2006).

The signal word is meant to draw the reader's attention to the safety instruction and identifies the hazard severity.

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.

Safety Instructions for Electric Drives and Controls

⚠ WARNING

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

4 Overview

4.1 Overview and Target Systems

The following table list all supported target systems for technology libraries.

Library	Target system IndraMotion MLC	Target system IndraLogic XLC	Target system IndraMotion MLD	Described in
ML_TechBase.compiled-library	X	X		(chapter 5 "ML_TechBase.library" on page 33)
ML_TechMotion.compiled-library	X			(chapter 6 "ML_TechMotion.library" on page 159)
ML_TechRegi.compiled-library MX_TechRegi.lib	X		X	(chapter 7 "ML_TechRegi.library/MX_TechRegi.lib" on page 285)
ML_TechCrank.compiled-library	X			(chapter 8 "ML_TechCrank.library" on page 371)
ML_TechTensionAdvanced.compiled-library	X			(chapter 13 "ML_TechTensionAdvanced.library" on page 571)
RMB_TechWinder.compiled-library MX_TechWinder.lib	X		X	(chapter 9 "RMB_TechWinder.library/MX_TechWinder.lib" on page 409)
RMB_CrosscutCrossseal.compiled-library	X			(chapter 10 "RMB_TechCrossCutCrossSeal.library" on page 497)
RMB_TechCam.compiled-library	X			(chapter 11 "RMB_TechCam.library" on page 527)
RIL_ParameterChannel.compiled-library	X	X		(chapter 12 "RIL_ParameterChannel.library" on page 549)

Fig.4-1: Target systems supported by the technology libraries

4.2 Special Features of the IndraMotion MLD

4.2.1 General

The IndraMotion MLD is an "OnBoard" control integrated into the hardware of the IndraDrive control unit. The PLC firmware is part of the drive firmware. In contrast to the IndraMotion MLC, the PLC engineering environment is the IndraLogic 1.X.

Overview

4.2.2 Drive Firmware, Function Package

The drive firmware is available in the following variants starting from 07VRS:

- MPC firmware: Multi-axis MotionControl IndraMotion MLD-M for up to 8 axes
- MPH firmware: Single-axis control IndraMotion MLD-S for 1 axis
- MPD firmware: No IndraMotion MLD functionality possible
- MPB firmware: Single-axis control IndraMotion MLD-S for 1 axis

Function packages

The drive firmware functionality is divided in several function packages. The function packages have to be ordered and licensed according to the application. The required function packages are then enabled.

☞ The "IndraMotion MLD" function packages must be enabled for the IndraMotion MLD.

The additional package "IndraMotion MLD Advanced" ("MA") has to be enabled for extended technology function blocks.

☞ The additional package "IndraMotion MLD Advanced" is only available for MPC firmware and MPH firmware (not for MPB firmware).

Technology libraries requiring, e.g. the additional package "IndraMotion MLD Advanced":

- MX_TechWinder
- MX_TechRegi

Multi-axis IndraMotion MLD-M

The MPC drive firmware is required for the multiple axis control IndraMotion MLD-M starting from the drive firmware 07VRS. The CSH01.3 control unit is necessary for this firmware.

Set additionally in the PLC configuration that the PLC has "permanent controls" (P-0-1367 "PLC configuration" Bit 4 set or dialog for the "PLC configuration") of the drive.

☞ The CSH01.3 control unit is necessary for the MPC firmware starting from MPX07.

5 ML_TechBase.library

5.1 Introduction and Overview

General

Technology function blocks (Tech FB) extend the basic functionality of the target systems IndraMotion MLC/IndraLogic XLC and IndraMotion MLD and provide application-specific functionalities. The library described includes basic functionalities that can be used for several applications or as stand-alone functionality.

The function blocks described are provided to the IndraMotion MLC/ IndraLogic XLC via the "ML_TechBase.compiled-library".

This documentation describes the functionality as well as the inputs and outputs of the technology function blocks.

Prerequisites

Technology function blocks of the "ML_TechBase.compiled-library" library 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.

Alias

There is a "FP_Aliases" entry in the "Global variables" directory in the library. These are identifiers behind the parameters. The parameters are system-dependent. Users should use these identifiers to be independent from the selected target system. For example, the identifier "FP_MODULO" is set on the axis parameter A-0-0045 in an IndraMotion MLC/IndraLogic XLC. In an IndraMotion MLD system it is set to the SERCOS parameter S-0-0103. Thus, the system used can be abstracted better at application level and generally valid code can be written.

Overview on Function Blocks

Function block	Description
Jogging variables	
MB_ContinuousAdjustType01	Continuous adjustment of a REAL variable via binary inputs
MB_ContinuousAdjustType02	Continuous adjustment of a DINT variable to be influenced via binary inputs
MB_IncrementalAdjustType01	Incremental adjustment of a REAL variable via binary inputs
Two-position controller	
ML_TwoPosCtrlType01	implements a two-position controller with hysteresis
Mathematical Functions	
ML_InterpolationLinear	cyclically calculates the y-value from the x-value according to the data point table
ML_MaxValue	Calculates the present maximum value from the time characteristics of an input signal
Application "Programmable limit switch"	
MC_DigitalCamSwitch	Implements an arbitrarily programmable limit switch with up to 32 switches on 8 outputs
Temperature controller	
IL_TempControlType01	Controls temperature including the comfort functions such as monitoring, pause and manual operation.
Touch Probe Function Blocks	

ML_TechBase.library

Function block	Description
MB_InitTouchProbe	Initialization of the touch probe functionality
MB_TouchProbe	Recording an axis position, master axis position, time or single-shot measurements
MB_TouchProbeContinuous	Recording an axis position, master axis position or time at continuous measurement
MB_WriteExpectWindow	Configures the expectation window for the touch probe functionality.
MB_AbortTrigger	Canceling the MB_TouchProbe function block
Trigonometric functions	
ML_DegToInc	converts angle from degree to corresponding incremental value
ML_DegToRad	converts angle in degree to angle in radian measure incl. modulo
ML_IncToDeg	converts angle specified in increments to degree
ML_IncToRad	converts angle specified in increments to the corresponding radian measure
ML_RadToDeg	converts angle in radian measure to corresponding angle in degree incl. modulo
ML_RadToInc	converts angle in radian measure to corresponding incremental value
Administration "cyclic data channels"	
MB_AllocCyclicParameter	Checking and reviewing configuration of parameter to be read or written cyclically in the cyclic data channel
MB_FreeCyclicParameter	Enabling an already configured parameter to be read or written cyclically in cyclic data channel
MB_GetCyclicChannelConfig	reads configuration of cyclic read and write data channels and saves them in a data structure
MB_GetCyclicParameterHandle	searches for a cyclic parameter to be read or written in the cyclic data channel and stores all required information in a handle
MB_ReadCyclicParameter	reads a cyclical parameter via handle and returns the current content as DINT
MB_ReadCyclicRealParameter	reads a cyclical parameter via Handle and returns the current content as REAL
MB_WriteCyclicParameter	writes cyclic parameters via handle, number of decimals are converted according to SERCOS
MB_WriteCyclicRealParameter	writes cyclic parameters via handle, number of decimals are rounded
MB_GetLastCyclicParameterError	returns the last occurred error when reading or writing a cyclical value for a handle and deletes the error
Backing up and restoring axes	
ML_AxisBackup	Parameter backup of an axis on a Compact Flash card

Function block	Description
ML_AxisRestore	Parameter restoration of an axis on a Compact Flash card
Logic applications	
IL_LinkedList;	Managing a list linked twice
Calculating modulo values	
MB_GetModuloType01	Returns modulo value (value range between 0 and + modulo value) (data type REAL)
MB_GetModuloType02	Returns a modulo value (value range between 0 and + modulo value) (data type DINT)
MB_GetModuloType03	Returns modulo value (value range between - modulo value and +modulo value)
MB_GetModuloType04	Returns modulo value (value range is between - modulo/2 and + modulo/2)
Auxiliary functions	
MB_GetSystemInfo	Checks function packages and system options
Downloading cam tables	
MB_LoadCamData	Downloads cams to the drive and control to encapsulate the existing detail differences.

Fig.5-1: Function Blocks in the ML_TechBase Library

5.2 Function Blocks for Application "Jogging Variables"

5.2.1 Introduction and Overview

With the following function blocks, PLC variables can be continuously or incrementally adjusted (jogged) via binary inputs:

- chapter 5.2.2 "MB_ContinuousAdjustType01" on page 38
- chapter 5.2.3 "MB_ContinuousAdjustType02" on page 42
- chapter 5.2.4 "MB_IncrementalAdjustType01" on page 45

When using the above mentioned function blocks to jog PLC variables, the following requirements have to be taken into account:

- The operation must be stopped via a binary input ("Enable")
- The variable to be influenced can be continuously adjusted using the function blocks MB_ContinuousAdjustType01 and MB_ContinuousAdjustType02 (similar to the long jog for SYNAX)
- The variable to be influenced can be incrementally adjusted using the function block MB_IncrementalAdjustType01 (similar to the short jog for SYNAX)
- The selected variable can either be changed in incremented or decremented steps within the limit values
- When reaching a limit value "HighLimitActive" = TRUE or "LowLimitActive" = TRUE), the continuous adjustment is deactivated, i. e. the function block does no longer influence the variable. In this case, the respective limit value is displayed at the input "HighLimit" or "LowLimit"

Basics

If the limits "HighLimit" and "LowLimit" have the same value for a specified modulo value, "LowLimit" is set to zero and "HighLimit" is set to the modulo value. Thus, it is possible to adjust the limit values in the entire modulo range.

The change velocity describes the rate by which the influenced variable changes during the adjustment. This rate depends on the step width and the number of steps per second. The change velocity is calculated as follows, for example, when changing a position:

$$AlterationVelocity\left[\frac{Degree}{s}\right] = Increments\left[Degree\right] * Increments\ per\ Second\left[\frac{1}{s}\right]$$

Fig.5-2: Change rate when changing a position

If, for example, the speed of the virtual master axis is changed, the change velocity is calculated as follows:

$$AlterationVelocity\left[\frac{r.p.m.}{s}\right] = Increments\left[r.p.m.\right] * Increments\ per\ Second\left[\frac{1}{s}\right]$$

Fig.5-3: Change rate when changing the axis speed

As long as the control signal is present, the influenced variable is changed with the defined velocity.

Adjustment Limits

The variable can only be adjusted via the corresponding inputs within the determined limit values.

Exception:

If the initial value of the variable to be adjusted is outside the specified limits, the influenced variable can only be adjusted in the direction of the target range. If a modulo value is specified, the influenced variable can be adjusted in both directions. After reaching the target range, it is no longer possible to leave this range.

Preset

The output variable can be preassigned to a definite value using the inputs "Preset" and "PresetValue". The "PresetAck" is set, after the "PresetValue" was applied. The output variable remains the "PresetValue" as long as the "Preset" input is set.

Basic rules

- If both control signals ("Inc"/"Dec") are TRUE at the same time, they are evaluated as FALSE
- An control signal changes the polarity and causes an immediate change of direction. Furthermore, the current adjustment is not completed
- If the adjustment velocity is changed during a motion, the behavior of the function blocks

  MB_ContinuousAdjustType01/MB_ContinuousAdjustType02 and

  MB_IncrementalAdjustType01

  differs.

  The change becomes immediately effective for MB_ContinuousAdjustType01 and MB_ContinuousAdjustType02. In contrast, the motion for the MB_IncrementalAdjustType01 is completed before the new step velocity is applied A subsequent adjustment procedure is then performed with the adjusted variables.

Furthermore, the following rules apply to the function block MB_IncrementalAdjustType01:

- The inputs "Inc" and "Dec" are evaluated with regard to the edges. A rising edge causes an adjustment by the specified step width ("StepWidth")

- A new control signal (positive edge) with the same polarity detected during a running motion causes another motion following the current motion

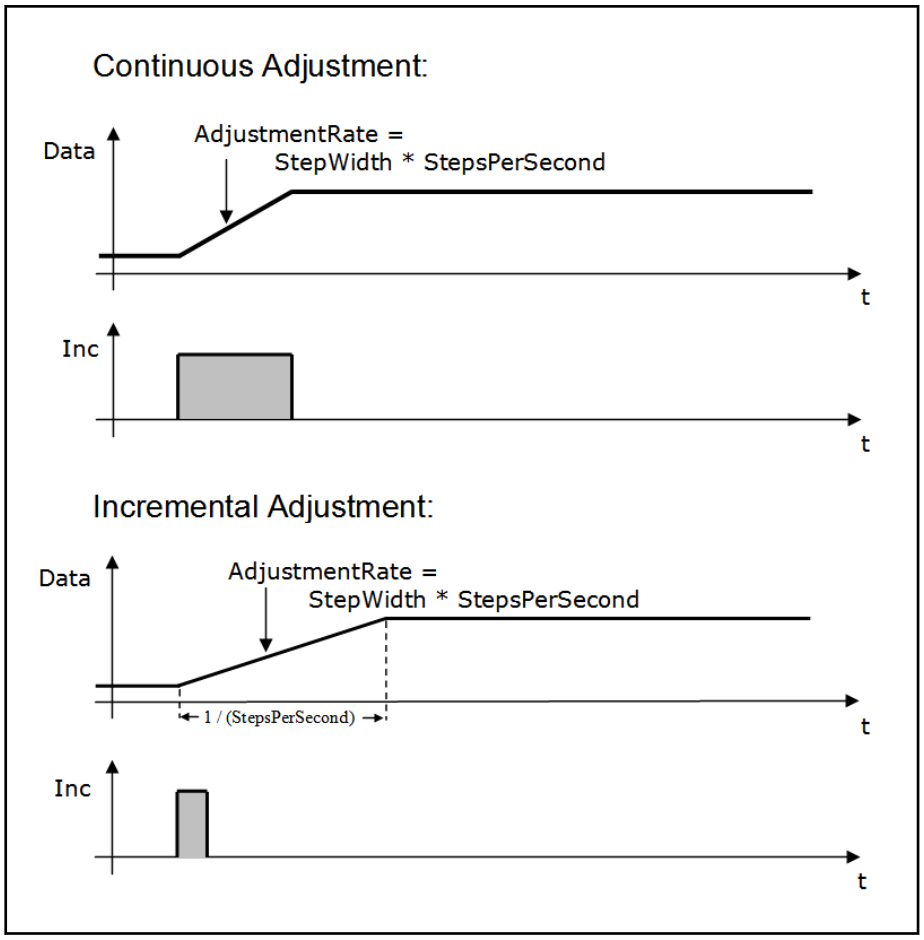


Fig.5-4: Comparison of the behavior between a continuous and an incremental adjustment

During the first activation ("Enable" = TRUE), the function blocks MB_ContinuousAdjustType01 and MB_ContinuousAdjustType02 initialize the variables required for determining the cycle time (irrespective of the present inputs). In the second cycle, a calculated cycle time is available required for calculating the correct change velocity. Subsequently, the cycle time is updated continuously.

By specifying a modulo value, the function block provides an output variable between zero and the specified modulo value. When exceeding the modulo value, the output variable is reset to zero. However, the adjustment process continues. The specified valid range can thus be between zero and the modulo value. The range can be adjusted by the zero value.

ML_TechBase.library

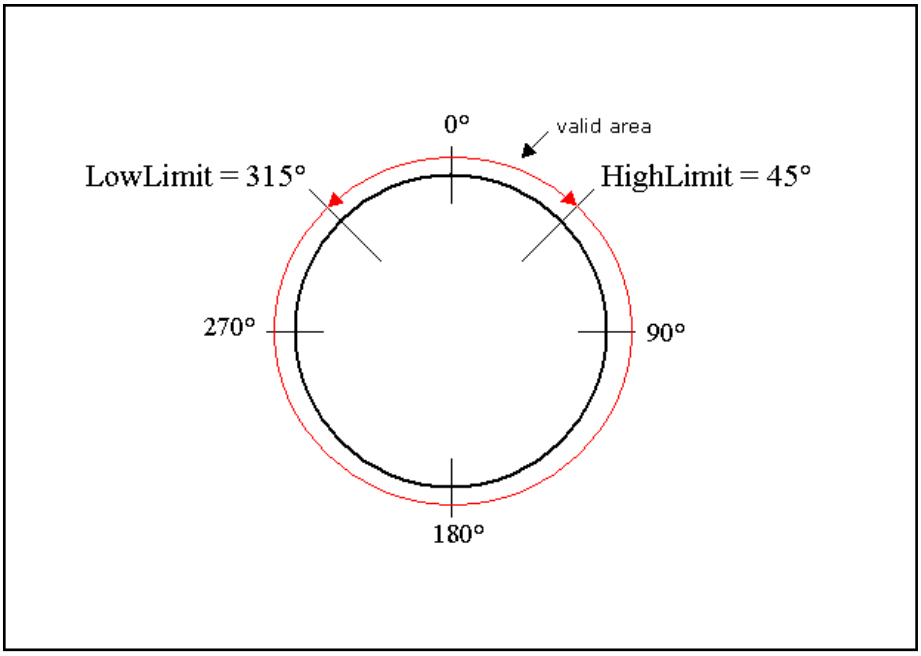


Fig.5-5: Possible allocation of a valid range when specifying a modulo value of 360°

In addition, the function block MB_ContinuousAdjustType01 or MB_ContinuousAdjustType02 provides the possibility to adjust the change velocity depending on the time by specifying scaling factors and time intervals.

5.2.2 MB_ContinuousAdjustType01

Brief Description

The MB_ContinuousAdjustType01 function block allows a continuous adjustment of a REAL variable via binary inputs.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD MPx18 or higher (in preparation)	In preparation

Fig.5-6: Reference table of the MB_ContinuousAdjustType01 function block

Interface Description

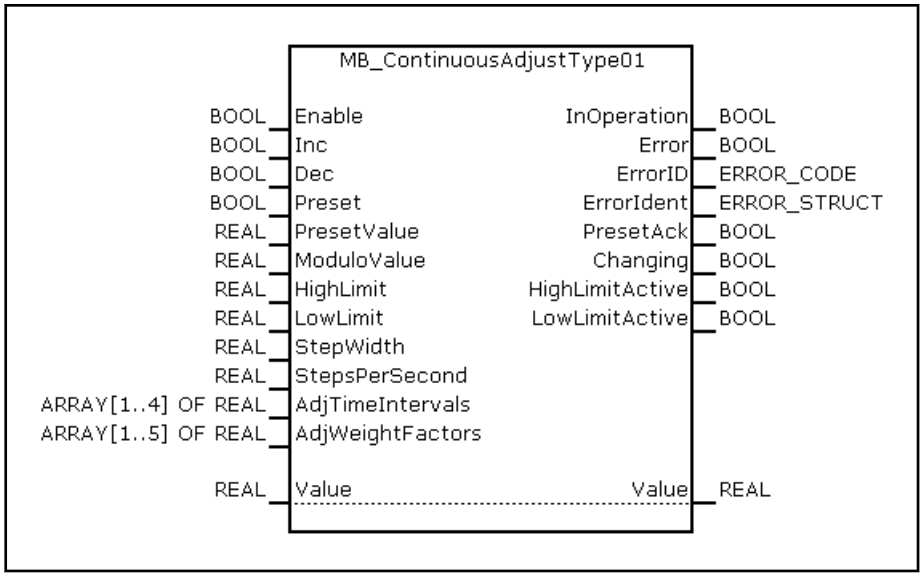


Fig.5-7: MB_ContinuousAdjustType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enabling of function block (cyclic, state-controlled)
	Inc	BOOL	Increase influenced variable
	Dec	BOOL	Decrease influenced variable
	Preset	BOOL	Set the Preset-Value (PresetValue)
	PresetValue	REAL	Specified value for the preset value
	ModuloValue	REAL	Modulo value (if 0, then absolute processing)
	HighLimit	REAL	Maximum output value of the variable to be influenced
	LowLimit	REAL	Minimum output value of the variable to be influenced
	StepWidth	REAL	Step width
	StepsPerSecond	REAL	Number of steps per second
	AdjTimeIntervals	ARRAY [1..4] OF REAL	Time intervals in seconds to scale the change velocity
	AdjWeightFactors	ARRAY [1.0,5] OF REAL	Scaling factors of the change velocity
VAR_OUTPUT	InOperation	BOOL	Function block enabled. The outputs "HighLimitActive" and "LowLimitActive" are valid
	Error	BOOL	Calculation of the variable "Value" to be influenced was completed with an error. The output variable "ErrorIdent" is valid
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed description of the diagnostics in case of an error
	PresetAck	BOOL	The value of the "PresetValue" input was applied
	Changing	BOOL	Calculation of the variable to be influenced was executed. Output variable "Value" is valid.

ML_TechBase.library

I/O type	Name	Data type	Description
	HighLimitActive	BOOL	Maximum output value of the variable to be influenced was reached
	LowLimitActive	BOOL	Minimum output value of the variable to be influenced was reached
VAR_IN_OUT	Value	REAL	The variable to be influenced

Fig.5-8: Interface variables of the MB_ContinuousAdjustType01 function block

Signal-time diagram

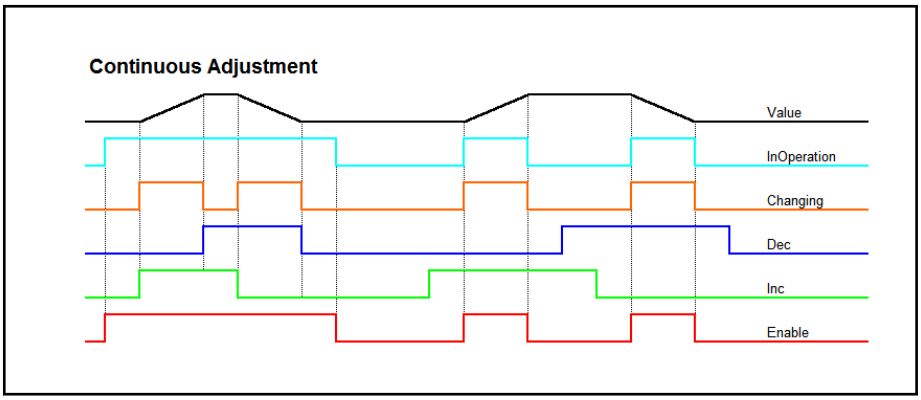


Fig.5-9: Continuous Parameter Adjustment

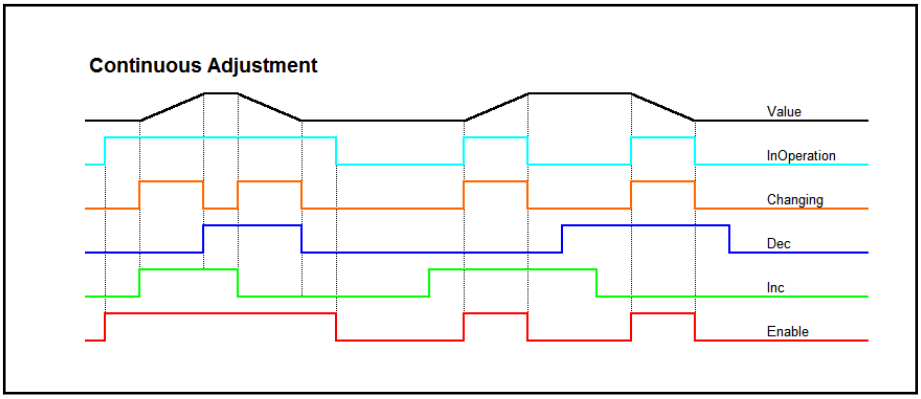


Fig.5-10: Continuous Parameter Adjustment

Functional Description

The adjustment of the influenced variable "Value" can be carried either in positive or negative direction. A valid range is specified for influencing the variable by the limit values. The specification of a modulo value allows the adjustment in order to be within the reference system of the axis. By specifying the step width and the number of steps per second, the change velocity of the influenced variable is determined. Additionally, a specified value ("PresetValue") in the permissible operating range can be directly written to the influenced variable.

After the enabling of the processing via "Enable", the variable to be influenced can be changed via the inputs "Inc" (in positive direction) and "Dec" (in negative direction). As long the "Inc" input or the "Dec" input are set, the adjustable variable can be continuously increased or decreased ("StepsPerSecond" * "StepWidth").

When initializing the function block, it is to be ensured that the input variables "ModuloValue", "HighLimit" and "LowLimit" are provided with valid values.

By specifying the scaling factors ("AdjWeightFactors[ ]") and their respective time intervals ("AdjTimeIntervals[ ]"), the change velocity can be modified with regard to time. The scaling factors and the time intervals have to be set

additionally to the inputs "StepsPerSecond" and "StepWidth". Depending on the current time interval, the change velocity is multiplied with the respective scaling factor. The figure below shows the behavior of the block with an example configuration: The "AdjTimeIntervals" input variables ([T1; T2; T3; T4]) are given in seconds. T5 is the time interval starting after the completion of the time interval T4 up to the abortion of the adjustment process. The "AdjWeightFactors" input variables ([0,5; 1; 1,5; 2; 4]) determine the scaling of the change velocity. The last "AdjWeightFactors" value ("AdjWeightFactors [5]") is assigned to the time interval T5.

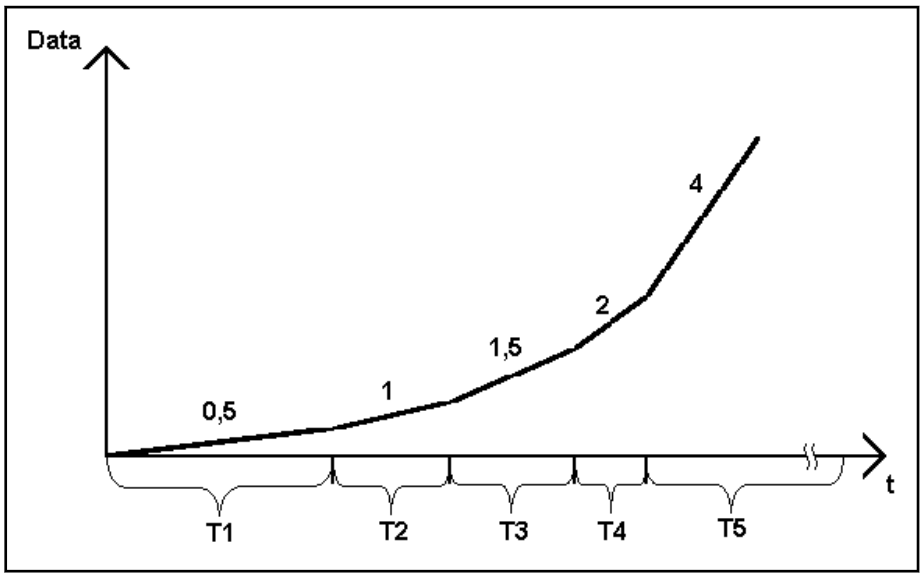


Fig.5-11: Scaled adjustment

☞ If a time interval of 0 is selected, the following scaling factor is retained till the adjustment process is aborted.

Example:

If "AdjTimeIntervals[2]" = 0, the scaling factor of "AdjWeightFactors[3]" is retained till the abortion of the process.

The calculation of the time elapsed to define the correct time interval of the scaling factors is reset with each new adjustment process. If the inputs "AdjWeightFactors" and "AdjTimeIntervals" are not set, a continuous adjustment is executed with the change velocity given in the inputs "StepsPerSecond" and "StepWidth".

☞ Positive values have to be specified for the input values of the time intervals ("AdjTimeIntervals"). The scaling factors can also contain negative factors.

The variable to be influenced can only be set to the specified preset value if it is within the valid range specified by "HighLimit" and "LowLimit".

The "InOperation" output signalizes that the function block is operating. However, an adjustment is not executed. The "Changing" indicates that the adjustment process was processed without errors. The value of the adjusted variable "Value" is valid.

The "Error" output indicates if an error occurs during the adjustment. For detailed information on the error, refer to the "ErrorIdent" structure.

ML_TechBase.library

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the **F_RELATED_TABLE**, 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs are not within the valid range
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block

Fig.5-12: Error codes of the MB_ContinuousAdjustType01 function block

5.2.3 MB_ContinuousAdjustType02

Brief Description

The MB_ContinuousAdjustType02 function block allows a continuous adjustment of a DINT variable to be influenced via binary inputs.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD MPx18 or higher (in preparation)	In preparation

Fig.5-13: Reference table of the MB_ContinuousAdjustType02 function block

Interface Description

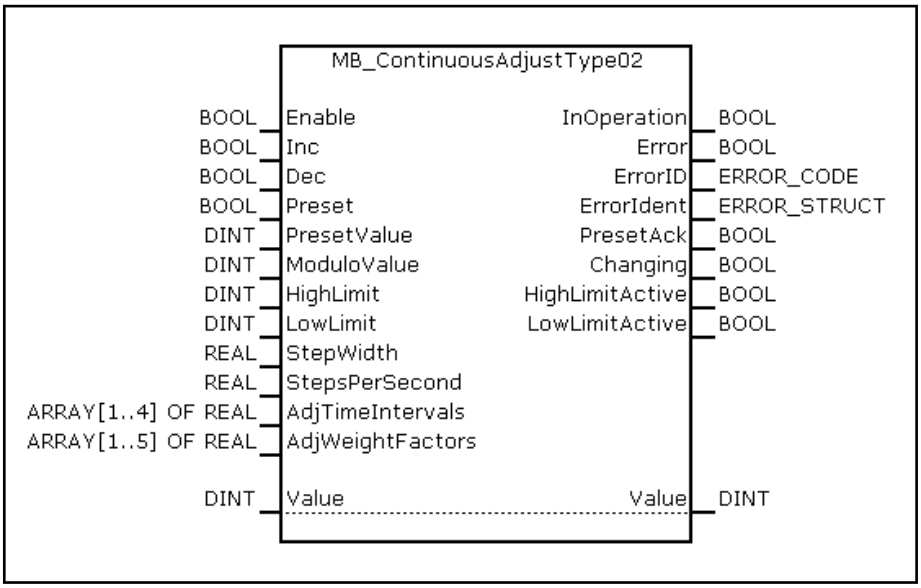


Fig.5-14: MB_ContinuousAdjustType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enabling of function block (cyclic, state-controlled)
	Inc	BOOL	Increase influenced variable
	Dec	BOOL	Decrease influenced variable
	Preset	BOOL	Set the Preset-Value (PresetValue)
	PresetValue	DINT	Specified value for the preset value
	ModuloValue	DINT	Modulo value (if 0, then absolute processing)
	HighLimit	DINT	Maximum output value of the variable to be influenced
	LowLimit	DINT	Minimum output value of the variable to be influenced

I/O type	Name	Data type	Description
	StepWidth	REAL	Step width
	StepsPerSecond	REAL	Steps per second
	AdjTimeIntervals	ARRAY [1..4] OF REAL	Time intervals in seconds to scale the change velocity
	AdjWeightFactors	ARRAY [1.0,5] OF REAL	Scaling factors of the change velocity
VAR_OUTPUT	InOperation	BOOL	Function block enabled. The outputs "HighLimitActive" and "LowLimitActive" are valid
	Error	BOOL	Calculation of the variable "Value" to be influenced was completed with an error. The output variable "ErrorIdent" is valid
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed description of the diagnostics in case of an error
	PresetAck	BOOL	The value of the "PresetValue" input was applied
	Changing	BOOL	Calculation of the variable to be influenced was executed. The output variable "Value" is valid
	HighLimitActive	BOOL	Maximum output value of the variable to be influenced was reached
	LowLimitActive	BOOL	Minimum output value of the variable to be influenced was reached
VAR_IN_OUT	Value	DINT	The variable to be influenced

Fig.5-15: Interface variables of the MB_ContinuousAdjustType02 function block

Signal-time diagram

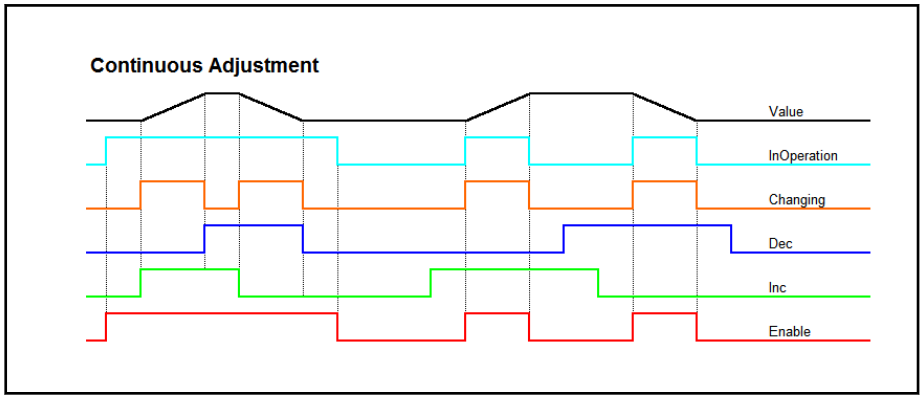


Fig.5-16: Continuous Parameter Adjustment

Functional Description

The adjustment of the influenced variable "Value" can be carried either in positive or negative direction. A valid range is specified for influencing the variable by the limit values. The specification of a modulo value allows the adjustment in order to be within the reference system of the axis. By specifying the step width and the number of steps per second, the change velocity of the influenced variable is determined. Additionally, a specified value (PresetValue) in the permissible operating range can be written directly to the influenced variable.

After the enabling of the processing via "Enable", the variable to be influenced can be changed via the inputs "Inc" (in positive direction) and "Dec" (in negative direction). As long the "Inc" input or the "Dec" input are set, the ad-

justable variable can be continuously increased or decreased ("StepsPerSecond" * "StepWidth").

When initializing the function block, it is to be ensured that the input variables "ModuloValue", "HighLimit" and "LowLimit" are provided with valid values.

By specifying the scaling factors ("AdjWeightFactors[ ]") and their respective time intervals ("AdjTimeIntervals[ ]"), the change velocity can be modified with regard to time. The scaling factors and the time intervals have to be set additionally to the inputs "StepsPerSecond" and "StepWidth". Depending on the current time interval, the change velocity is multiplied with the respective scaling factor. The figure below shows the behavior of the block with an example configuration: The "AdjTimeIntervals" input variables ([T1; T2; T3; T4]) are given in seconds. T5 is the time interval starting after the completion of the time interval T4 up to the abortion of the adjustment process. The "AdjWeightFactors" input variables ([0,5; 1; 1,5; 2; 4]) determine the scaling of the change velocity. The last "AdjWeightFactors" value ("AdjWeightFactors [5])" is assigned to the time interval T5.

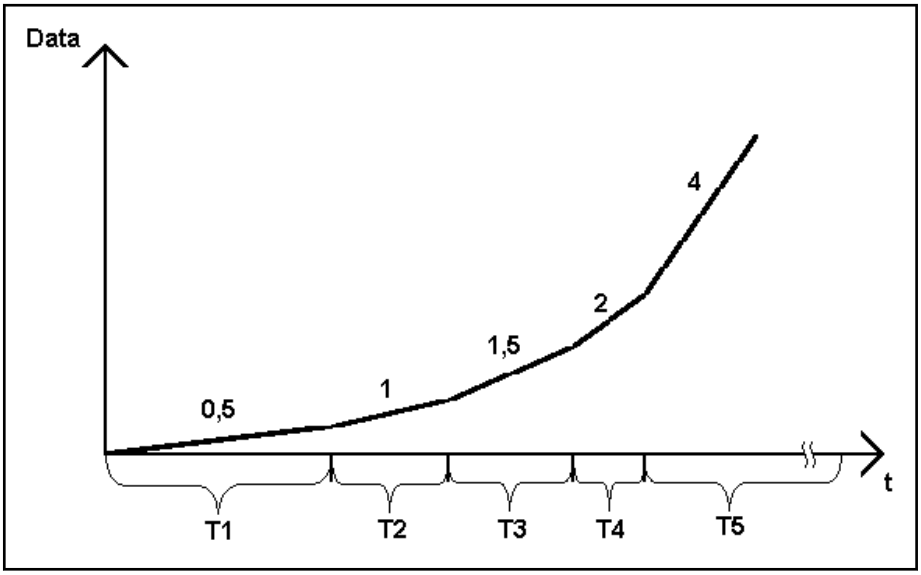


Fig.5-17: Scaled adjustment

☞ If a time interval of 0 is selected, the following scaling factor is retained till the adjustment process is aborted.

Example:

If "AdjTimeIntervals[2]" = 0, the scaling factor of "AdjWeightFactors[3]" is retained till the abortion of the process.

The calculation of the time elapsed to define the correct time interval of the scaling factors is reset with each new adjustment process. If the inputs "AdjWeightFactors" and "AdjTimeIntervals" are not set, a continuous adjustment is executed with the change velocity given in the inputs "StepsPerSecond" and "StepWidth".

☞ Positive values have to be specified for the input values of the time intervals (AdjTimeIntervals). The scaling factors can also contain negative factors.

The variable to be influenced can only be set to the specified preset value if it is within the valid range specified by "HighLimit" and "LowLimit".

The "InOperation" output signalizes that the function block is operating. However, an adjustment is not executed. The "Changing" indicates that the adjustment process was processed without errors. The value of the adjusted variable "Value" is valid.

The "Error" output indicates if an error occurs during the adjustment. For detailed information on the error, refer to the "ErrorIdent" structure.

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs are not within the valid range
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block

Fig.5-18: Error codes of the MB_ContinuousAdjustType02 function block

5.2.4 MB_IncrementalAdjustType01

Brief Description

The MB_IncrementalAdjustType01 function block allows an incremental adjustment of a REAL variable via binary inputs.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD MPx18 or higher (in preparation)	In preparation

Fig.5-19: Reference table of the MB_IncrementalAdjustType01 function block

Interface Description

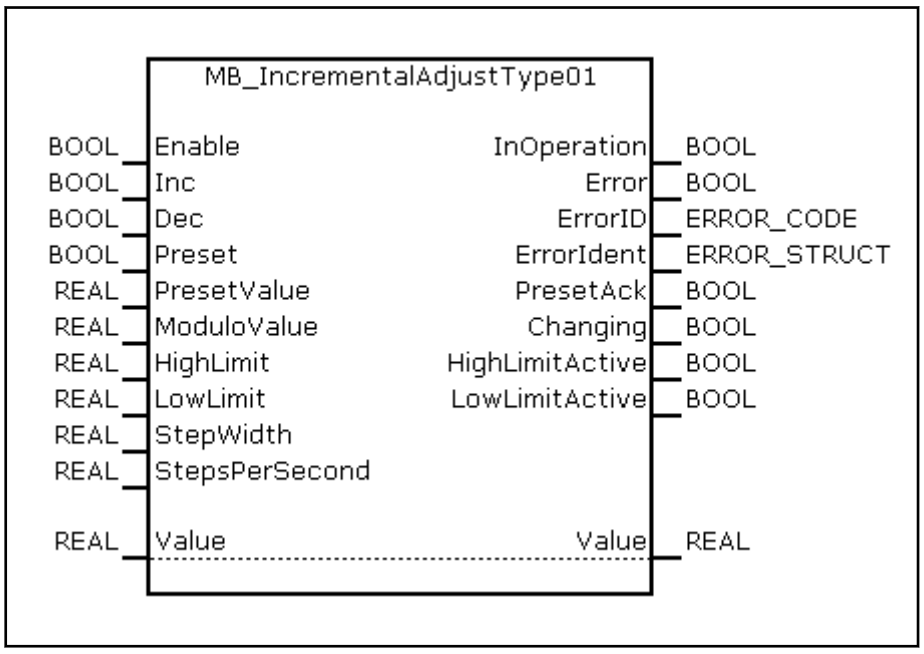


Fig.5-20: MB_IncrementalAdjustType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enabling of function block (cyclic, state-controlled)
	Inc	BOOL	Increase influenced variable

ML_TechBase.library

I/O type	Name	Data type	Description
	Dec	BOOL	Decrease influenced variable
	Preset	BOOL	Set the Preset-Value (PresetValue)
	PresetValue	REAL	Specified value for the preset value
	ModuloValue	REAL	Modulo value for rotary calculation
	HighLimit	REAL	Maximum output value of the variable to be influenced
	LowLimit	REAL	Minimum output value of the variable to be influenced
	StepWidth	REAL	Step width
	StepsPerSecond	REAL	Steps per second
VAR_OUTPUT	InOperation	BOOL	Function block enabled. The outputs "HighLimitActive" and "LowLimitActive" are valid
	Error	BOOL	Calculation of the variable "Value" to be influenced was completed with an error. The output variable "ErrorIdent" is valid
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed description of the diagnostics in case of an error
	PresetAck	BOOL	The value of the "PresetValue" input was applied
	Changing	BOOL	Calculation of the variable to be influenced was executed. Output variable "Value" is valid.
	HighLimitActive	BOOL	Maximum output value of the variable to be influenced was reached
	LowLimitActive	BOOL	Minimum output value of the variable to be influenced was reached
VAR_IN_OUT	Value	REAL	The variable to be influenced

Fig.5-21: Interface variables of the MB_IncrementalAdjustType01 function block

Signal-time diagram

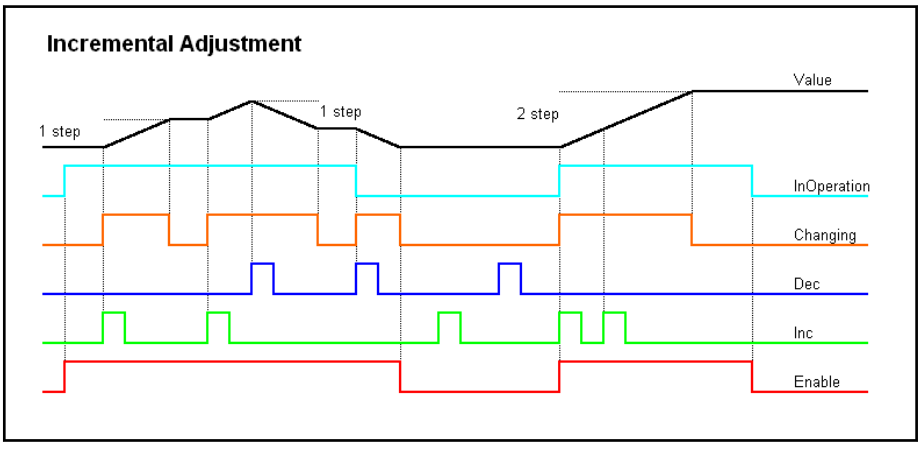


Fig.5-22: Incremental parameter adjustment

Functional Description

The adjustment of the influenced variable "Value" can be carried either in positive or negative direction. A valid range is specified for influencing the variable by the limit values. The specification of a modulo value allows the adjustment in order to be within the reference system of the axis. By specifying the step width and the number of steps per second, the change velocity of

the influenced variable is determined. Additionally, a specified value ("PresetValue") in the permissible operating range can be directly written to the influenced variable.

After the enabling of the processing via "Enable", the variable to be influenced can be changed via the inputs "Inc" (in positive direction) and "Dec" (in negative direction). As long the "Inc" input or the "Dec" input are set, the adjustable variable can be continuously increased or decreased ("StepsPerSecond" * "StepWidth").

The variable to be influenced can only be set to the specified preset value if it is within the valid range specified by "HighLimit" and "LowLimit".

The "InOperation" output signalizes that the function block is operating. However, an adjustment is not executed. The "Changing" indicates that the adjustment process was processed without errors. The value of the adjusted variable "Value" is valid.

If an error occurs during the function block processing, this is signalized by the output "Error" = TRUE. For detailed information on the error, refer to the "ErrorIdent" output structure.

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs are not within the valid range
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block

Fig.5-23: Error codes of the MB_IncrementalAdjustType01 function block

5.2.5 Parameterization of the Function Blocks to Jog Variables

Required Hardware

The following Rexroth hardware components are required:

- IndraMotion MLC/IndraLogic XLC hardware

Required Firmware

The following control firmware has to be used together with the above mentioned Rexroth hardware components:

- IndraMotion MLC: FWA-CMLXX*-MLC-12VRS
- IndraLogic XLC: FWA-CMLXX*-XLC-12VRS

XX - Device variant, e.g. CML65

Required software

The following PC software has to be used:

- IndraWorks IndraMotion MLC 12VRS
- IndraWorks IndraLogic XLC 12VRS

Parameterization of the Function Blocks for Adjusting Variables

No special parameterization is required.

5.3 Function Block for Two-Position Controller

5.3.1 Introduction

The two-position controller has a hysteresis and is appropriate for simple closed-loop controls.

ML_TechBase.library

5.3.2 ML_TwoPosCtrlType01

Brief Description

The function block ML_TwoPosCtrlType01 implements a two-position controller with hysteresis.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-24: Reference table of ML_TwoPosCtrlType01 function block

Interface Description

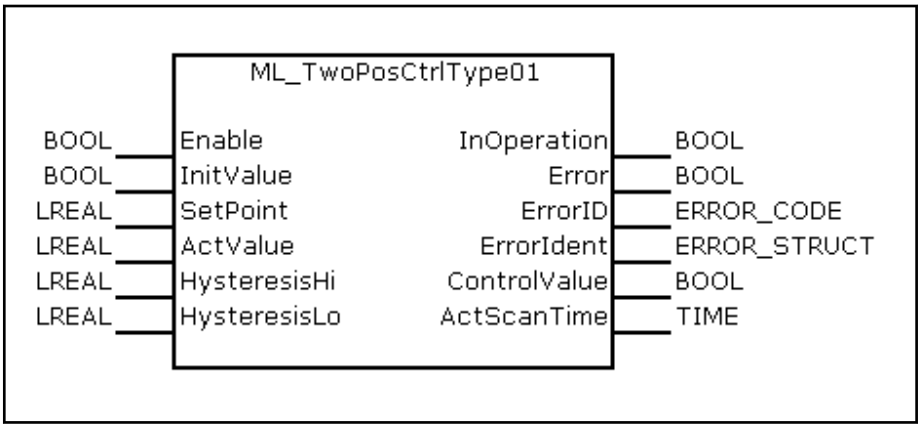


Fig.5-25: ML_TwoPosCtrlType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing enabled for function block (cyclic, state-controlled)
	InitValue	BOOL	Specified value for the control variable y
	SetPoint	LREAL	Reference variable w (command value)
	ActValue	LREAL	Controlled variable x (actual value)
	HysteresisHi	LREAL	Upper response threshold of the hysteresis
	HysteresisLo	LREAL	Lower response threshold of the hysteresis
VAR_OUTPUT	InOperation	BOOL	Calculation of the control variable y completed without error, output variable "ControlValue" valid
	Error	BOOL	Calculation of the control variable "ControlValue" completed with error, output variable "ErrorIdent" valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	Describing diagnostics in case of error
	ControlValue	BOOL	Controller output value – control variable y
	ActScanTime	TIME	Controller cycle time [ms]

Fig.5-26: Interface variables of the ML_TwoPosCtrlType01 function block

Specification

The function block ML_TwoPosCtrlType01 can be adjusted in its characteristics via the following input variables:

"HysteresisHi":

- The input has to be preset with a positive value or zero, the value zero switches off the positive hysteresis.

"HysteresisLo":

- The input has to be preset with a negative value or zero, the value zero switches off the negative hysteresis.

When calculating the control variable "ControlValue", "FALSE" is dominant, i.e. with "HysteresisLo" == "HysteresisHi" == zero with a control difference of zero ("SetPoint" == "ActualValue"), the output "ControlValue" is always "FALSE".

Functional Description

After the function block was enabled for processing, the function block ML_TwoPosCtrlType01 uses "Enable" to preset the internal controller state in the first controller cycle as well as the controller output to the preset value given at the input "InitValue" and cyclically calculates the control variable "ControlValue" in the following cycles.

If no error occurs during the function block processing, the validity of the control variable is signalized by the "InOperation" output and the elements of the output structures "ErrorID" and "ErrorIdent" are not updated in this case.

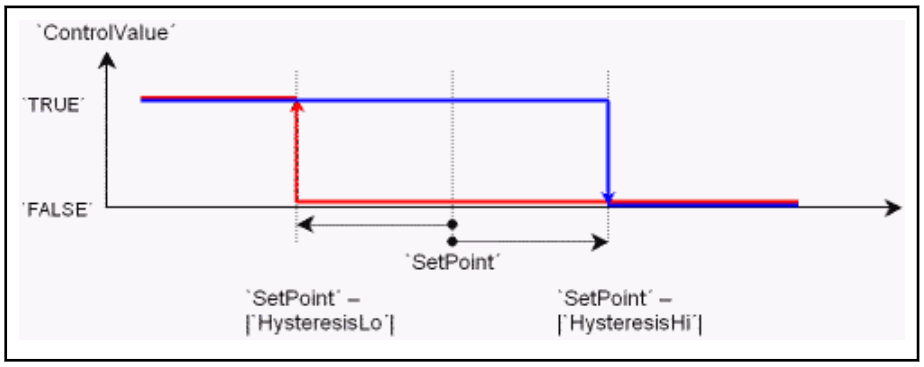


Fig.5-27: Hysteresis of the ML_TwoPosCtrlType01 function block

If an error occurs in the course of the processing of the function block, this is signalized by the output "Error" and specified by the elements of the output structure "ErrorID" and "ErrorIdent" which are updated in this case. The outputs "ControlValue" and "ActScanTime" are not updated in case of an error.

Error Handling

The function block uses the error table **MLC_TABLE**, 16#0030. In Additional1 and Additional2 it can generate the following error messages:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#F0260030	0	Inputs "HysteresisHi" or "HysteresisLo" are invalid
STATE_MACHINE_ERROR (16#0005)	16#F0260031	xx	Invalid state of the state machine xx indicates the invalid state

Fig.5-28: Error codes of the ML_TwoPosCtrlType01 function block

5.4 Function Blocks for Mathematical Functions

5.4.1 Introduction

The following blocks provide mathematical functions. The function block ML_InterpolationLinear page, 50 can be used to calculate mid-points and linearization, for example.

Each time the block ML_MaxValue, page, 52 is called, it indicates at its output the maximum value of all previously used values.

ML_TechBase.library

5.4.2 ML_InterpolationLinear

Brief Description

The function block ML_InterpolationLinear cyclically calculates the y-value associated to a given x-value by linear interpolation based on a defined X-Y node table.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-29: Reference table of ML_InterpolationLinear function block

Interface Description

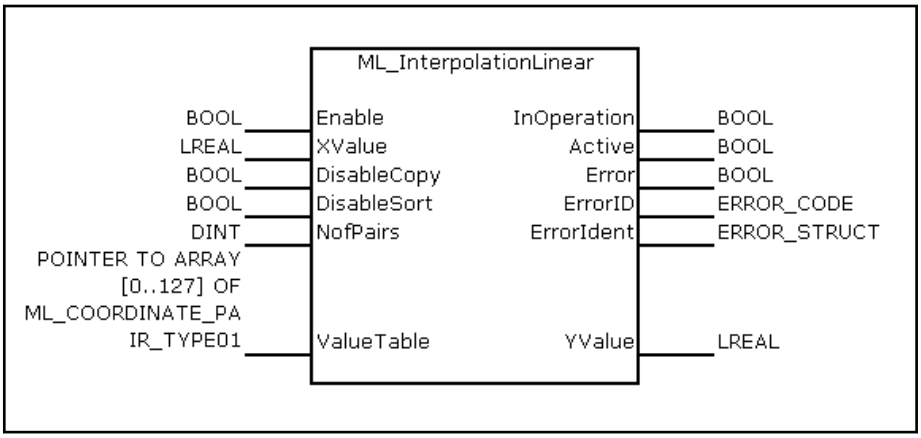


Fig.5-30: ML_InterpolationLinear function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enabling of function block (cyclic, state-controlled)
	XValue	LREAL	X value
	DisableCopy	BOOL	Deactivating copying of the data point table
	DisableSort	BOOL	Deactivating sorting of the data point table
	NofPairs	DINT	Number of the connected value pairs in the data point table
	ValueTable	POINTER TO ARRAY [0...127] OF ML_CO-ORDINATE_ PAIR_TYPE01	Address of the data point table
VAR_OUTPUT	InOperation	BOOL	Cyclic calculation of a Y value completed without error, output variable "YValue" valid
	Active	BOOL	Calculation of a Y value active, output variables invalid
	Error	BOOL	Calculation of a Y value completed with error, error outputs "ErrorID" and "ErrorIdent" valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, the output contains detailed error information
	YValue	LREAL	Y value

Fig.5-31: Interface variables of the ML_InterpolationLinear function block

Functional Description

The function block ML_InterpolationLinear cyclically calculates the Y value "YValue" associated with the input value "XValue" with "Enable" on basis of a

specified data point table after the release for processing of the function block.

The data point table is saved on an internal data array depending on the inputs "DisableCopy" and "DisableSort" at a rising edge and sorted in an ascending order afterwards. The time for calculating an associated y-value "YValue" can be minimized if a data point table already sorted in an ascending order is used when the function block is enabled for processing and this is marked by the input "DisableSort". In this case, the data array is not sorted. If in a new processing of the function block, the internally available and already used data point table should be used again, the input "DisableCopy" as well as the input "DisableSort" have to be set to enabled.

The Y value to be calculated is stored on the output "YValue" in case of error-free processing of the function block, whereupon the validity of the value is signalized by the output "InOperation".

If it is not possible to provide the Y value to be calculated immediately after the activation of the function block, this is indicated by the output "Active".

If an error occurs while processing the function block, this is signaled by the "Error" output. The output "YValue" is not updated in case of an error.

Influence of the inputs "DisableCopy" und "DisableSort" on the run time behavior of the function block

DisableCopy	DisableSort	Use case
FALSE	FALSE	An unsorted data point table is specified which has not yet been used in this way in the previous calculation. The data point table is copied and sorted
FALSE	TRUE	A data point table which is already sorted in an ascending order is specified which has not yet been used in this way in the previous calculation. The data point table is copied but not sorted internally
TRUE	FALSE	This combination is not a reasonable state
TRUE	TRUE	The data point table has already been used in previous calculations. Therefore, it is already known internally and sorted in an ascending order

Fig.5-32: Assignment of the inputs DisableCopy and DisableSort

☞ The sorting of the data point table after the activation of the function block might lead to the fact that the function block cannot deliver the desired output value in the first or the following cycle due to the restriction to a maximally admissible run time. This duration at the beginning of the activation is signalized by "Active".

Error Handling

The function block uses the error table **MLC_TABLE**, 16#0030. In Additional1 and Additional2 it can generate the following error messages:

ML_TechBase.library

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#F0260030	0	Inputs invalid (XValue not within the data points)
STATE_MACHINE_ERROR (16#0005)	16#F0260031	xx	Invalid state of the state machine. xx indicates the invalid state

Fig.5-33: Error codes of the ML_InterpolationLinear function block

5.4.3 ML_MaxValue

Brief Description

The function block ML_MaxValue determines the actual maximum value from the time dependent course of the input signal.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-34: Reference table ML_MaxValue function block

Interface Description

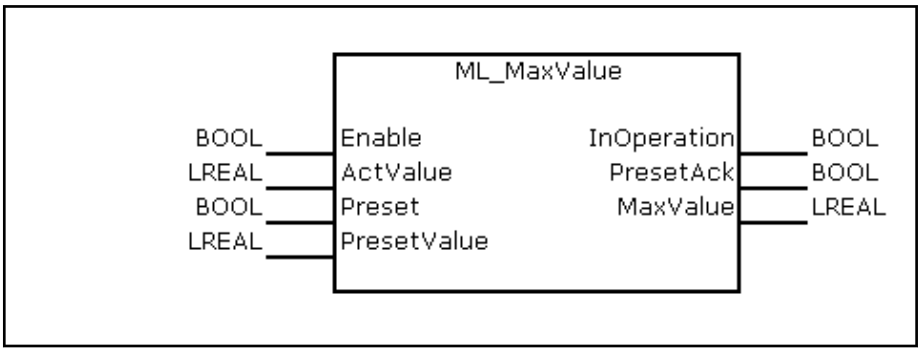


Fig.5-35: ML_MaxValue function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing enabled for function block (cyclic, state-controlled)
	ActValue	LREAL	Current value
	Preset	BOOL	Activation of the preset value
		LREAL	Default value
VAR_OUTPUT	InOperation	BOOL	Determination of the maximum value completed
	PresetAck	BOOL	Confirmation for the takeover of the "PresetValue" value
	MaxValue	LREAL	Maximum value

Fig.5-36: Interface variables of the ML_MaxValue function block

Functional Description

After the release of the function block, the function block ML_MaxValue uses "Enable" to compare the current value of the input variable "ActValue" to the value in "MaxValue". It copies the respectively higher value to the output variable "MaxValue".

If the input "Preset" is activated for the run time of the function block, the value of the input variable "PresetValue" is copied to the output variable "MaxValue". The input is state-controlled, i.e. as long as the "Preset" input is active, the "PresetValue" is copied to the "MaxValue" output variable.

> If a "PresetValue" value is to be specified at the same time the function is activated, the "Preset" input must be set simultaneously with the "Enable" input.

Error codes The function block ML_MaxValue does not generate any errors.

5.5 Function Block for the Application "Programmable Limit Switch"

5.5.1 Introduction and Overview

The function block for the application "Programmable limit switch" chapter 5.5.5 "MC_DigitalCamSwitch" on page 54 is a PLCopen-compliant block. Up to 8 outputs with up to 32 switches can be arbitrarily programmed with it.

The data structures used

- chapter 5.5.4 "MC_OUTPUT_REF" on page 54
- chapter 5.5.2 "MC_CAMSWITCH_REF with ML_BASIC_CAMSWITCH_REF" on page 53
- chapter 5.5.3 "MC_TRACK_REF with ML_BASIC_TRACK_REF" on page 53

also fulfil the PLCopen specifications.

5.5.2 MC_CAMSWITCH_REF with ML_BASIC_CAMSWITCH_REF

Type definitions The following definition describes data types used in the function block chapter 5.5.5 "MC_DigitalCamSwitch" on page 54.

The type definitions are included in the library.

"MC_CAMSWITCH_REF" contains an array of 32 basic structures each of which describes a single switch.

Values are given in units [u]. These relate to the corresponding axis. If e.g. a switch-on position is at 20u and the corresponding axis is scaled in degrees, the switch-on position correspondingly is 20°.

The basic structure "ML_BASIC_CAMSWITCH_REF" is as follows:

Structure element	Type	Description
TrackNumber	INT	"TrackNumber" is the reference for the switch track
FirstOnPosition [u]	REAL	Lower limit at which the output is ON (switch-on position)
LastOnPosition [u]	REAL	Upper limit at which the output is ON (turn-off position)
AxisDirection	INT	Direction in which the programmable limit switch works: both (=0; Default); positive (1); negative (2)
CamSwitchMode	INT	position-related (=0; Default); time-related (=1)
Duration	TIME	Coupled with time-related "CamSwitchMode". I.e. the output remains on for the specified time

Fig.5-37: Definition of the basic elements of the ML_BASIC_CAMSWITCH_REF structure

5.5.3 MC_TRACK_REF with ML_BASIC_TRACK_REF

Type Definitions The following definition describes data types which are used in the function block chapter 5.5.5 "MC_DigitalCamSwitch" on page 54. The data type definitions are contained in the library.

A cam disc is an element describing the characteristics of the cam switch related to an output.

The "MC_TRACK_REF" contains an array with 8 basic structures (i.e. 8 cam discs which are allocated to 8 outputs). The basic structure ("ML_BASIC_TRACK_REF") of a cam disc is as follows:

Structural element	Type	Description
OnCompensation	TIME	Lead time of the cam disc related to the time of turn-on.
OffCompensation	TIME	Lead time of the cam disc related to the time of turn-off.
Hysteresis [u]	REAL	Hysteresis can be set in order to filter noise-afflicted input signals.
TimeDef	WORD	Time definition: Bit 0 set: "OnCompensation" time is negative; Bit 1 set: "OffCompensation" time is negative;

Fig.5-38: Definition of the basic elements of the MC_TRACK_REF structure

Several cams can be allocated to every cam disc.

5.5.4 MC_OUTPUT_REF

Type Definitions

The following definition describes data types which are used in the function block chapter 5.5.5 "MC_DigitalCamSwitch" on page 54. The data type definitions are contained in the library.

The data type describes the 8 outputs of the cam switch.

"MC_OUTPUT" contains an ARRAY with 8 boolean variables which are assigned to the outputs and it looks as follows:

Structural element	Type	Description
OUTPUT	ARRAY [0..7] OF BOOL	Outputs of the cam switch.

Fig.5-39: Definition of the basic elements of the MC_OUTPUT_REF structure

5.5.5 MC_DigitalCamSwitch

Brief Description

The function block MC_DigitalCamSwitch implements an user-defined programmable limit switch with up to 32 switches on 8 outputs.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-40: Reference table of the MC_DigitalCamSwitch function block

Interface Description

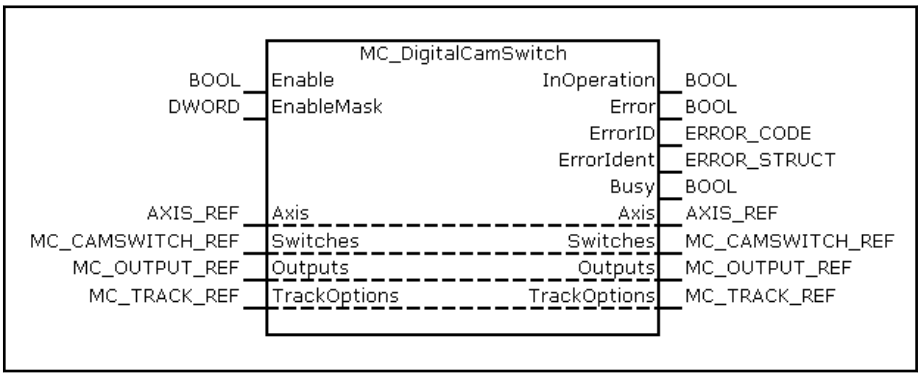


Fig.5-41: MC_DigitalCamSwitch function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis which is assigned to the programmable limit switch
	Switches	MC_CAMSWITCH_REF	Reference to the switch structure containing all definitions for all switches. See also chapter 5.5.2 "MC_CAMSWITCH_REF with ML_BASIC_CAMSWITCH_REF" on page 53
	Outputs	MC_OUTPUT_REF	Reference to the signal outputs directly related to the associated switch tracks (max. 8 per function block, first output = first "TrackNumber"). See also chapter 5.5.4 "MC_OUTPUT_REF" on page 54
	TrackOptions	MC_TRACK_REF	Reference to the structure containing the associated switch track characteristics, e.g. the on/off lead time per output/switch track. See also chapter 5.5.3 "MC_TRACK_REF with ML_BASIC_TRACK_REF" on page 53
VAR_INPUT	Enable	BOOL	Activates the switching outputs
	EnableMask	DWORD	32 bits of the type BOOL. Enables the different switch tracks. The lowest-order bit is allocated to the lowest "TrackNumber". If the value is set to 1 or to TRUE, the allocated "TrackNumber" is enabled. Since only the 8 lowest-order bits are available, the number is restricted to 8 switch tracks per function block. If the bit is not set, the corresponding output is set to 0
VAR_OUTPUT	InOperation	BOOL	The enabled switch tracks are calculated and the outputs are updated
	Error	BOOL	It indicates that an error occurred in the function block
	ErrorID	ERROR_CODE	Error ID
	ErrorIdent	ERROR_STRUCT	Error structure with further subdivision of the errors. See also "Error codes" on page 68
	Busy	BOOL	Is set when the function block is enabled (not in idle mode)
Note: A detailed description of the different inputs/outputs is given later in this document.			

Fig.5-42: Interface variables of the MC_DigitalCamSwitch function block

Functionality

The following function block MC_DigitalCamSwitch provides a PLCopen-specific (part 2) interface for a programmable limit switch. The function block calculates 8 binary switch track outputs according to the specification below.

Every binary output is allocated to a switch track. According to the following specification, a switch track can contain several on/off positions. A maximum of 32 switches, which can be arbitrarily distributed to the 8 switch tracks, is available.

ML_TechBase.library

Terminology

Description	Description
IndraMotion MLC/IndraLogic XLC	BRC rack-based MotionLogic control
Digital cam switch	Programmable limit switch (PLS)

Fig.5-43: Terminology of the programmable limit switch

Settings

The units for position and velocity have to be set accordingly for the axis allocated to the programmable limit switch. If e.g. the positions of the axis are defined in degrees, the programmable limit switch positions are also interpreted in degrees.

Data Types

The function block "MC_DigitalCamSwitch" uses the following data types:

A switch track is an element describing the characteristics of the programmable limit switch related to an output.

- chapter 5.5.3 "MC_TRACK_REF with ML_BASIC_TRACK_REF" on page 53 contains an array with 8 basic structures (i.e. 8 switch tracks allocated to 8 outputs). Several switches can be allocated to each switch track
- chapter 5.5.2 "MC_CAMSWITCH_REF with ML_BASIC_CAMSWITCH_REF" on page 53 contains an array of 32 basic structures each of which describes a single switch
- chapter 5.5.4 "MC_OUTPUT_REF" on page 54 It contains an array with 8 Boolean variables which are allocated to the outputs of the programmable limit switch

Example program

A short example explains how the function block is programmed. The different structure elements are explained in detail later.

It is assumed that a single switch should be created:

- with number 4 on switch track 3
- with a "FirstOnPosition" of 35 and
- a "LastOnPosition" of 55
- the switch is intended to react 10ms earlier and the switch track should have a hysteresis of 3
- The switch should switch on/off with regard to the position and not to the time and it should only be effective in positive direction

The variable declaration and the function block call are as follows:

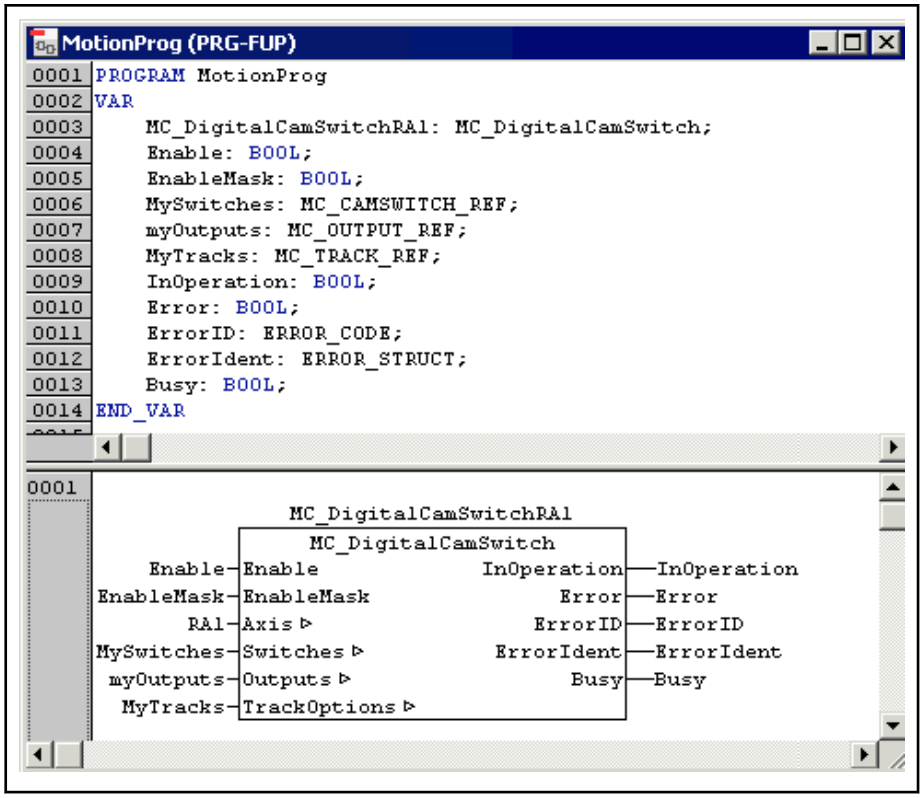


Fig.5-44: Variable declaration and function block call

Refer to the three input structures "MySwitches", "MyOutputs" and "MyTracks" which define all switch characteristics, switch tracks and outputs. In order understand these structures better, it is necessary to have a closer look at the data contents of the structures provided for the desired results of the example mentioned above:

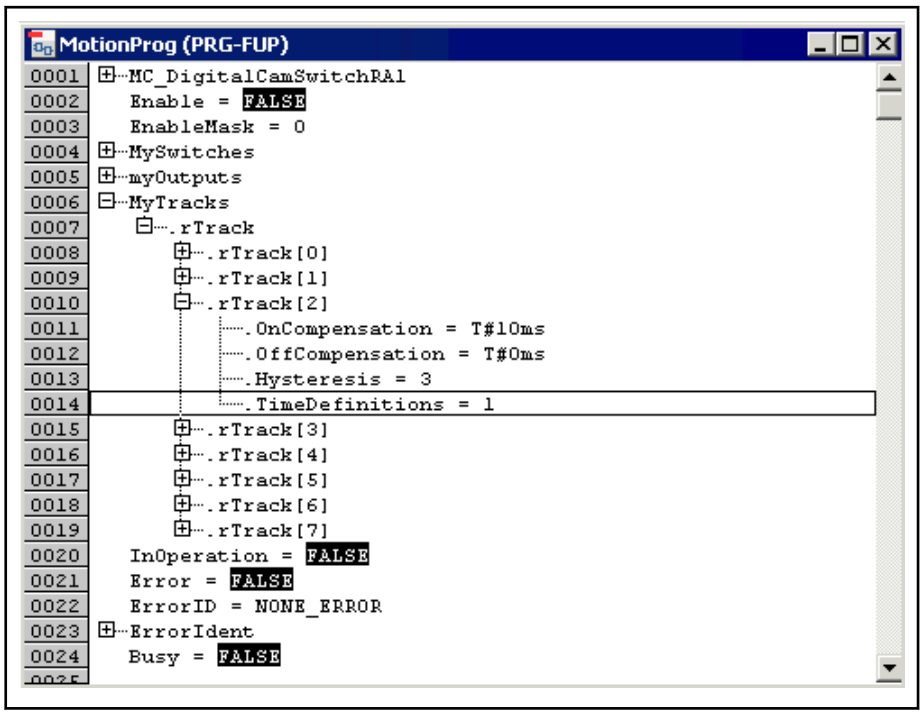


Fig.5-45: Content of the MyTracks structure

First, the switch track is defined (figure above).

1. Since switch track 3 should be used, rTrack[2] has to be initialized since the array starts with 0.
2. The OnCompensation time is set to the desired 10ms. To ensure that the switch switches earlier and not later, bit 0 has to be set in "TimeDefinitions" defining the "OnCompensation" time as negative.
3. Finally, the hysteresis has to be set to the desired value 3.

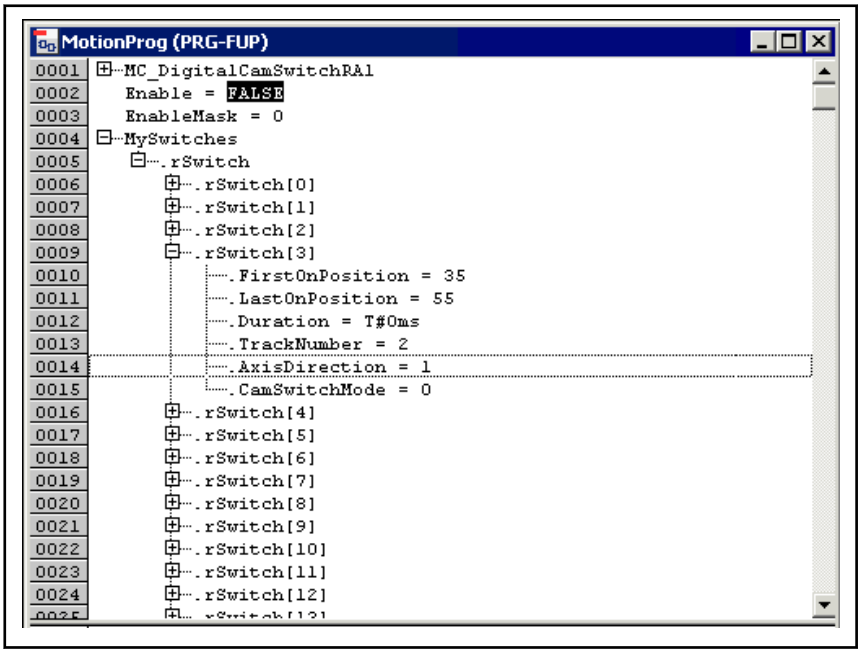


Fig.5-46: Content of the MySwitches structure

Now, the switch is defined (figure above).

1. The initialization of the switch structure has to be carried out with index 3 to initialize switch 4 because the array starts with index 0.
2. "FirstOnPosition" is set to the desired value 35, "LastOnPosition" is set to 55 and the "AxisDirection" is set to 1, which means that the switch is only effective in positive direction.
3. In order to allocate the switch to a switch track, the TrackNumber is set to 2. This determines that this switch is allocated to the switch track with index 2 in the switch track structure.
4. Finally, the "CamSwitchMode" has to be set to 0 in order to switch on the position-related mode.

☞ The individual elements of the REF structures can be accessed according to the following pattern:

- MyTracks.rTrack[2].TimeDefinitions
- MySwitches.rSwitches[3].AxisDirection
- MyOutputs.Output[4]

The function block start is prepared:

To start the function block, the remaining variables have to be added to the function block and their values have to be set accordingly. The "Enable" input has to be set and the "EnableMask" has to set the corresponding bit:

4 or more graphically 2#...00000100, bit 2 for the third switch track "**rTrack[2]**".

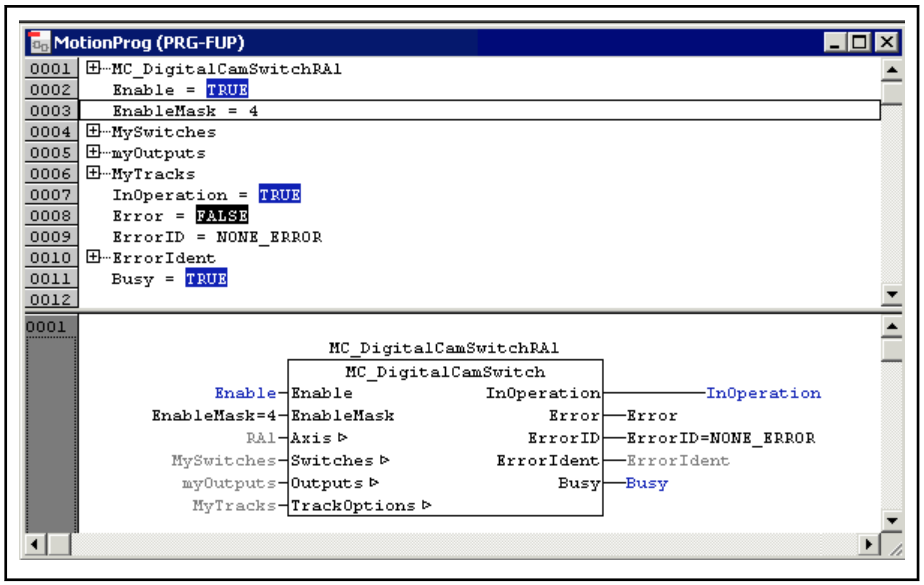


Fig.5-47: Running function block

Functional Description

The **Axis input** defines the position and velocity necessary for calculations. If the "Axis" input changes while the "Enable" input is not yet set, the "Busy" output remains set and the "InOperation" output becomes inactive. This happens because some internal values have to be updated before the calculation continues. This requires several PLC cycles. After this initialization sequence, the "InOperation" output becomes active again.

When the **Enable** input is set for the first time, the "Busy" output is also set, but the "InOperation" output remains inactive. This also happens because some internal values have to be updated before the calculation continues. This requires several PLC cycles. After this initialization sequence, the "InOperation" output becomes active again.

With the **EnableMask** input, outputs can be quickly enabled or blocked. The lowest-order bit is allocated to the lowest "TrackNumber". If the value is set to 1 or TRUE, the corresponding "TrackNumber" is enabled. Due to the restriction to the 8 lowest-order bits, only 8 switch tracks are available as per function block. If the bit is not set, the respective output is set to 0, regardless whether the current position is located within the area of a switch or not. If "EnableMask" is 0, "InOperation" becomes inactive since none of the outputs is enabled.

Every switch has a start position and an end position. This allows every user to specify a **FirstOnPosition** and a **LastOnPosition** (or time) for each switch. In addition to a mechanical switch, there is the possibility to activate the switch for a certain time and to allocate a lead time and a hysteresis to it for a certain time. If ("FirstOnPosition" > "LastOnPosition"), the switch behaves inversely and is only switched off within these positions.

Area/modulo calculations

Most likely, the PLS is only used for rotary processes. However, the PLS could also be used for a linear axis which runs forwards and backwards. In both cases, the PLS units are based on the units defined for the axis.

AxisDirection indicates in which motion direction the switch becomes active. If the "AxisDirection" is 0 (default), the switch becomes active in both motion directions. If the "AxisDirection" is 1, the output is only set if the traveling

takes place in positive direction. If the "AxisDirection" is set to 2, the output is only set if the traveling takes place in negative direction.

CamSwitchMode switches between the indication of position or time. "CamSwitchMode" = 0 refers to the position mode. The output enabled at "FirstOnPosition" and disabled at "LastOnPosition". If "CamSwitchMode" = 1, the output enabled at the "FirstOnPosition" and disabled after the specified "Duration".

Duration: duration for which a time-controlled switch is switched on. If the position is still within the switch area after this time has expired, the output is not set anew until the position has switched off the switch. The maximum possible duration depends on the hardware used. In case of excess switch-on duration, the error code 16#F0260006 is generated.

The lead time (**OnCompensation** and **OffCompensation**) can be positive or negative. Negative means that the output switches before the switching position is reached. The lead time is indicated in ms (also refer to "TimeDef").

Hysteresis: This parameter prevents the output from switching too often if the axis is near the switching point and the current position varies around the switching point. The hysteresis is part of "MC_TRACK_REF" which means that an individual hysteresis can be set for every switch track.

TimeDef: This parameter has been added to allow a differentiation between positive and negative times for the "On/OffCompensation". If bit 0 is set, the "OnCompensation" time is negative, and if bit 1 is set, the "OffCompensation" time is negative.

Example:

"OffCompensation" is set to 300, "TimeDef" is set to 2.

This means that the lead time is "-300ms".

The following figure shows the different variables and structures and their respective use:

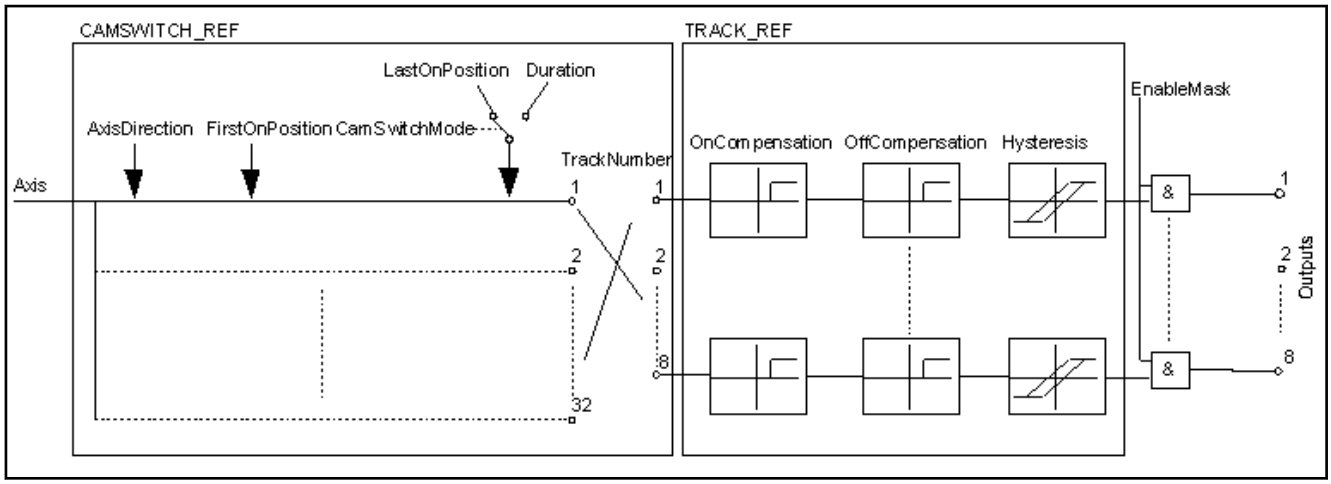


Fig.5-48: Relation of the structures and the corresponding functionality of the MC_DigitalCamSwitch function block

The following figure shows the mechanical equivalent of the digital programmable limit switch.

Please note that not all the functions were shown since it is either not possible or hardly possible to show them.

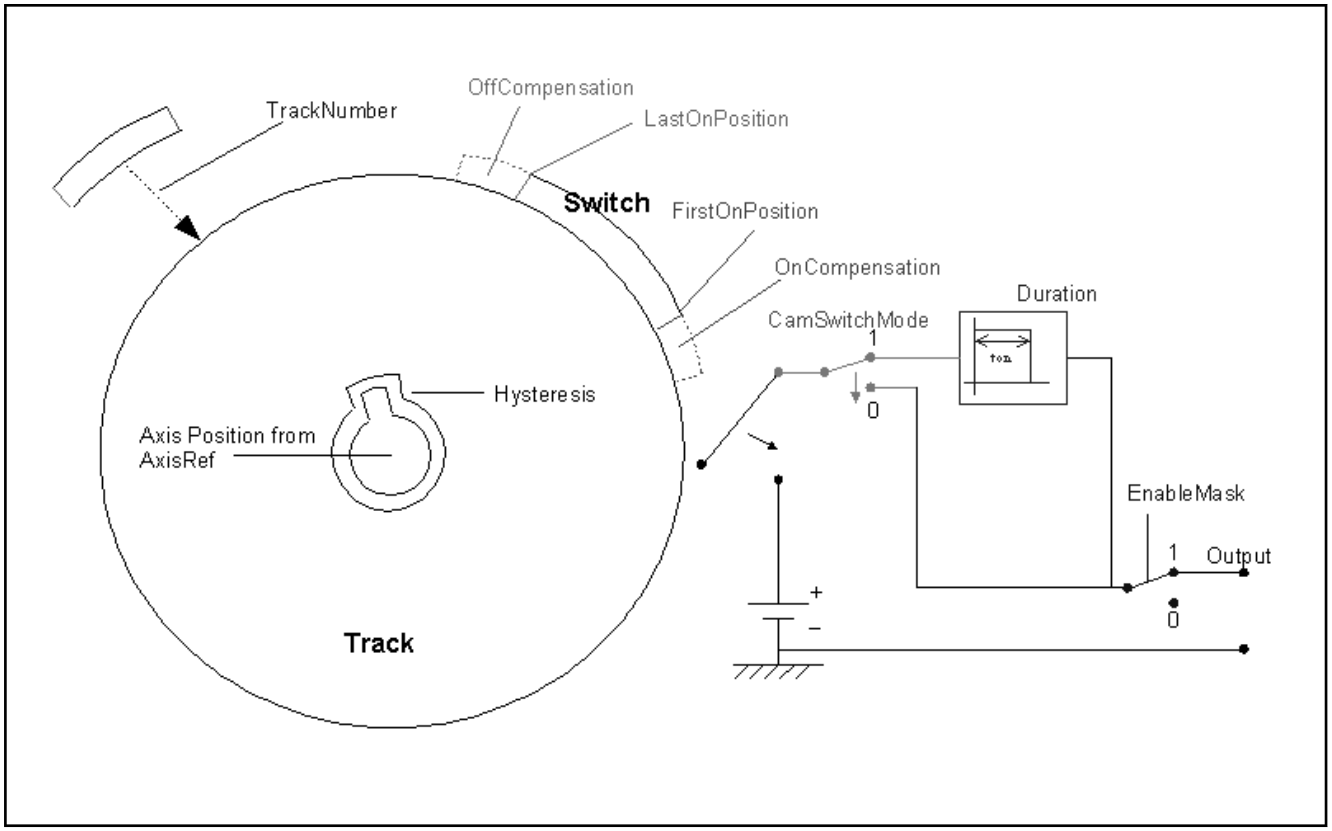


Fig.5-49: Mechanical equivalent of the digital programmable limit switch

Time diagram

The following example shows the use of the "MC_CAMSWITCH_REF" structure and of the time-based switches. In the following list, you find a predetermined configuration of values:

Parameters	Type	Switch01	Switch02	Switch03	Switch04	...	SwitchN
TrackNumber	INT	1	1	1	2		
FirstOnPosition [u]	REAL	2000	2500	4000	3000		
LastOnPosition [u]	REAL	3000	3000	1000	-		
AxisDirection	INT	1=Pos	2=Neg	0=Both	0=Both		
CamSwitchMode	INT	0=Position	0=Position	0=Position	1=Time		
Duration	TIME	-	-	-	1350		

Fig.5-50: Example MC_CAMSWITCH_REF structure

This example uses the values from the above example "MC_CAMSWITCH_REF". It neither uses "On/OffCompensation" nor "Hysteresis".

This is the behavior of the outputs when the axis continuously moves in the positive direction. The axis is a modulo axis with a modulo length of 5,000u (units).

ML_TechBase.library

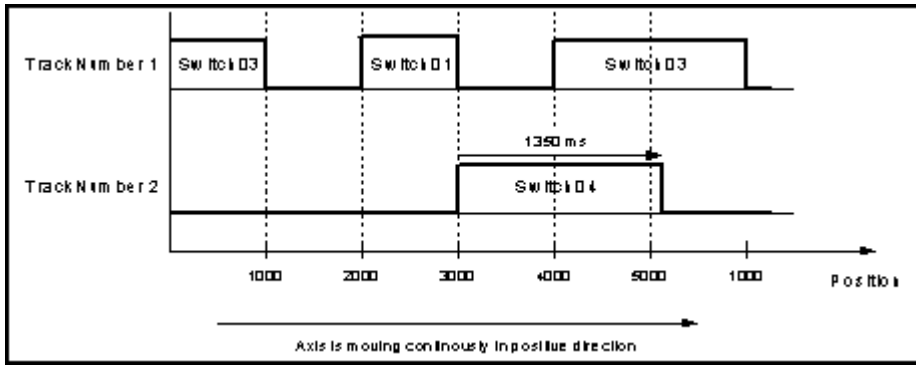


Fig.5-51: Behavior of the outputs in the example

Detailed description of the switch "Switch01"

This example additionally uses "OnCompensation"= -125ms and "OffCompensation"= +250ms.

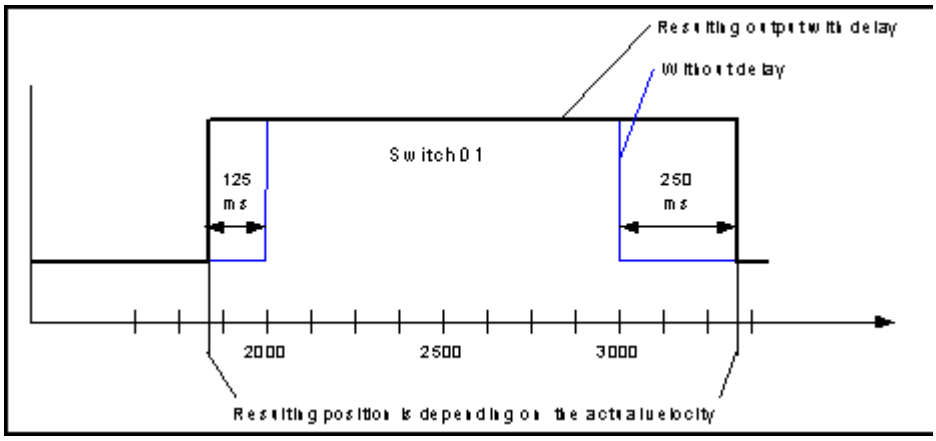


Fig.5-52: Detailed behavior of Switch01 when moving in the positive direction

This is the behavior of the outputs when the axis continuously moves in the negative direction without "On" or "OffCompensation" and without "Hysteresis":

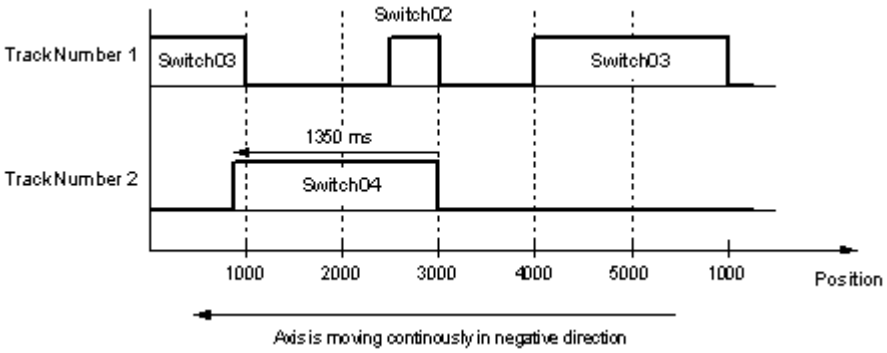


Fig.5-53: Detailed behavior of the switches when moving in the negative direction

Realization Details

The PLCopen extension specifies the interface of the PLS functionality. However, some application details are not specified. Therefore, these were defined for an implementation with a better runtime behavior.

OnCompensation/OffCompensation

PLCopen defines the "On/OffCompensation" only for motions in positive direction. In this case, the "OnCompensation" relates to the "FirstOnPosition" while the "OffCompensation" relates to the "LastOnPosition". How can this be defined for motions in negative direction?

Based on applications for which the lead times are used to compensate for the on and off delays with actuators, this means that the "OnCompensation" is responsible for the rising edge of the signal. In other words: if the traveling takes place in positive direction, the "OnCompensation" relates to the "**First**OnPosition", and if the traveling takes place in negative direction, the "OnCompensation" relates to the "**Last**OnPosition". The "OffCompensation" is defined accordingly

The following figure shows the behavior for "OnCompensation"= -0.1s and "OffCompensation"= +0.2s:

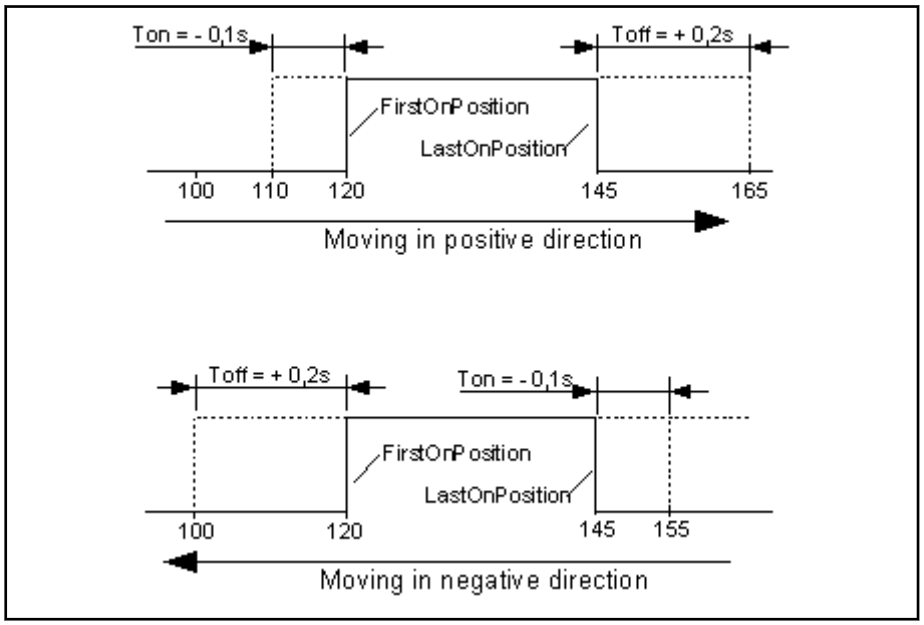


Fig.5-54: Definition of the OnCompensation and OffCompensation time

Duration

The "Duration" is defined as the time during which the switch remains switched on after it has been switched on. The question is what happens when the duration has expired but the current position is still within the "FirstOnPosition" and "LastOnPosition". Does the output have to be switched on immediately or do you have to wait until the current position has moved out of the "FirstOnPosition" and "LastOnPosition"?

It has been decided to wait for a rising edge of the switch output. In other words: the current position has to be beyond the "FirstOnPosition" and "LastOnPosition" before the "Duration" can be started anew.

The same applies to the "OnCompensation"/"OffCompensation" where the "Duration" begins with the rising edge of a switch, i.e. at the "LastOnPosition" when moving in negative direction.

The situation is explained in the following figure:

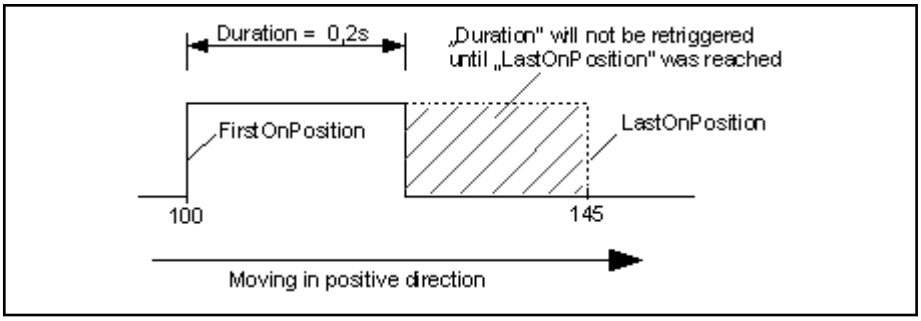


Fig.5-55: Definition of the duration

Combination of OnCompensation/OffCompensation and Duration

Should these be considered when a time-based switch with "On/OffCompensation" times is used? The "OnCompensation" is taken into consideration. This leads to an earlier/later activation of the output. The "OffCompensation" time however is more difficult since the output is disabled after the lead time ("Duration"). The "OffCompensation" time is only relevant when it has to be determined that the output has to be retriggered (hatched area in the figure above).

In this application, both the "OnCompensation" and the "OffCompensation" were taken into consideration.

How a switch is not used

The PLS has up to 32 switches. The element "TrackNumber" in the MC_CAMSWITCH_REF structure determines if a switch is used or not.

This variable contains the switch track number. Valid values are from 0 to 7 since there is a maximum of 8 switch tracks. The default setting of this variable is 8, it is marked as not used. Every invalid number (8 to 65535) is interpreted as an unused switch and is therefore not evaluated by the firmware.

Restrictions of the PLS

The restrictions of the PLS are discussed in this section. The restrictions are only definitions which do however not cause any error. The system behavior is determined with these definitions. Therefore, the user has to expect different behaviors.

" Lost" switching actions

The sampling theorem indicates that the sampling rate is twice as high as the highest signal frequency which is needed in order to recognize the signal. This also applies to the PLS. The PLS samples the position data every time it is called by the PLC. Usually, this is the Motion-synchronous task or every other cyclic task. Every switch which is switched on for a time shorter than the sampling rate, is not identified correctly. The following equations show how the velocity or the size of switch, with which switching actions are lost, are calculated.

Rotary systems:

1. $\omega = 2\pi n$

n indicates the revolutions as per second.

2. $\omega = \rho / t$

ρ is the angle in rad and t is the time in seconds.

3. Conversion factor of degree to rad:

$\pi / 180$

If one takes equations 1 and 2, one gets

$$\rho[rad] = 2\pi nt$$

for the angle, or

$$n\left[\frac{revolutions}{second}\right] = \frac{\rho}{2\pi t}$$

If the factors are adapted for common units, one gets:

equation **A1.)**

$$\rho[°] = n[rpm] * t[ms] * 0.006$$

or for the velocity:

equation **B1.)**

$$n[rpm] = \frac{\rho[°]}{t[ms]} * 166.\bar{6}$$

A1.) and B1.) provide the two necessary equations to the user which are required to determine whether switching actions can be lost. For example, if the maximum rate of rotations of 1,000 rpm is known and the PLS is called for in a cyclic task with a cycle time of 2ms. What is the minimum size of a switch to ensure that it is recognized by the PLS? If one takes equation A1.), this results in:

$$\rho[°] = 1000rpm * 2ms * 0.006 = 12$$

I.e. a switch has to be of a size of 12° at least in order to ensure that it will surely be recognized at a rate of rotations of max. 1,000rpm.

Linear systems:

1. $v = s/t$,

s is the distance in m and t is the time in seconds.

If one takes the cycle time of the PLC as the sampling time for the PLS and adds the corresponding factors to adjust the units, one gets the following equations:

equation **A2.)**

$$s[mm] = v\left[\frac{mm}{min}\right] * t[ms] * 1,6\bar{6} * 10^{-5}$$

or for the velocity:

equation **B2.)**

$$v\left[\frac{mm}{min}\right] = \frac{s[mm]}{t[ms]} * 60000$$

.

These equations also apply for systems having inch as the unit for the distance:

equation **A2.)**

$$s[inch] = v\left[\frac{inch}{min}\right] * t[ms] * 1,6\bar{6} * 10^{-5}$$

or for the velocity:

equation **B2.)**

$$v\left[\frac{inch}{min}\right] = \frac{s[inch]}{t[ms]} * 60000$$

Example: A switch in a linear system is 0.02 inches long. The PLS is located in an PLC task with a cycle time of 2ms. Which maximum velocity is possible so that the switch is surely be recognized?

If one takes equation B2.), the following results:

$$v\left[\frac{inch}{min}\right] = \frac{0,02inch}{2ms} * 60000\frac{1}{min} = 600$$

Switches seem to have the wrong size

The previous chapter was about lost switching actions. Even if the conditions for lost switches are not fulfilled, the switching times might be inexact. The PLS can only change the state of its outputs when it is called for by the PLC. If e.g. a switch should be closed for 3.7ms with a specified velocity but the PLS is only called for every 2ms, the output is set to "On" either for 2ms (case B) or for 4ms (case A), depending on the sampling time of the PLS. A more detailed explanation can be found in the following figure

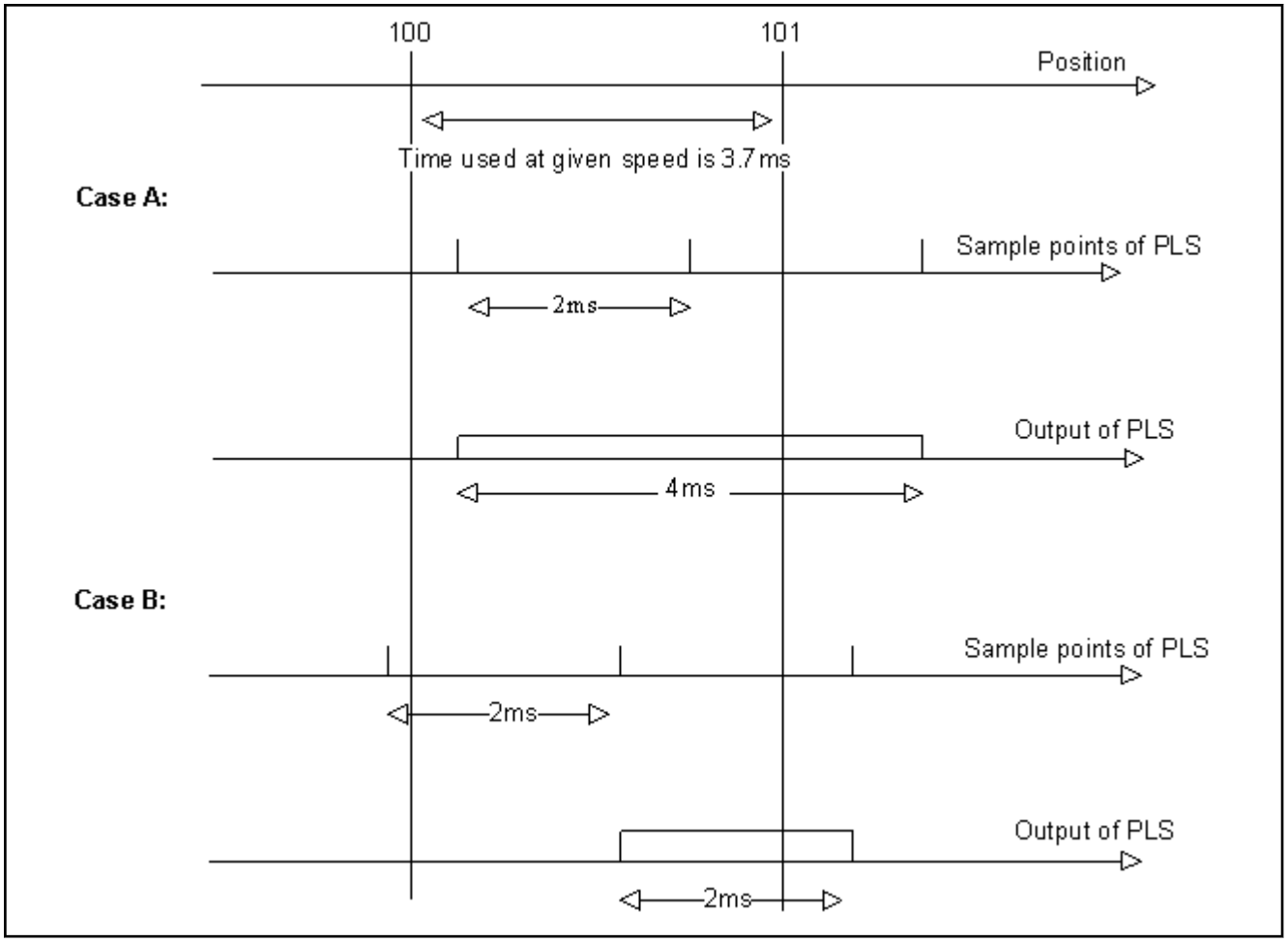


Fig.5-56: Inexact switch sizes

Switching takes place twice during acceleration/deceleration

If "OnCompensation" and "OffCompensation" are used for a switch, it is possible that it is switched on and off several times when the PLS master position accelerates or decelerates. This happens because the PLS requires a constant velocity which is however not guaranteed in certain situations. The bigger problem occurs during deceleration. Hereinafter, you find some options when a switch can be activated at several point in time.

The following equations can be established:

1.) $s_v = v_0 * t_{lead}$

s_v is the distance which is travelled at the specified velocity v_0 in the "OnCompensation" time t_{lead}.

2.) $s_b = v_0 * t - \frac{1}{2} a t^2$

s_b is the distance necessary for deceleration at a specified start velocity v_0 and deceleration a in the time t.

Furthermore, the general equations are:

3.) $v = a * t$ or $t = v / a$

The "OnCompensation" time of a programmable limit switch "moves" the switch in the direction of lower positions in accordance with equation 1.). The distance necessary in order to come to a standstill at a specified velocity can be determined with equation 2.).

Therefore, the general rule is as follows:

If the position is reached with v = v_0 so that the output of a programmable limit switch can be switched on with "OnCompensation" time. And the input position is decelerated in a way which ensures that the stop position is the switch-on position of the programmable limit switch without "OnCompensation" time (v=0), the output in between is not switched off.

If we equate the equations 1.) and 2.), we get the equation which ensures that the switch is only switched on once. If we decelerate a little bit faster, the switch is switched on several times.

$$s_b = s_v \Rightarrow v_0 t - \frac{1}{2} a t^2 = v_0 t_{lead} \Rightarrow v_0 \frac{v_0}{a} - \frac{1}{2} a \frac{v_0^2}{a^2} = v_0 t_{lead}$$

$$\Rightarrow \frac{v_0^2}{2a} = v_0 t_{lead}$$

4.) $$\Rightarrow v_0 = 2 t_{lead} a$$

In equation 4.) one can see that the start velocity v_0 with which it is possible to switch a limit switch on several times increases when the "OnCompensation" time t_{lead} or the deceleration a are increased.

There is no solution for this problem. One can only keep the "OnCompensation" time and the deceleration for this application as low as possible.

Another possibility is to use the "Hysteresis" in order to also take the variations due to the acceleration/deceleration in consideration.

The same considerations can be used for the acceleration or negative velocities.

Error codes

The function block uses the error table **MLC_TABLE**, 16#0030. In Additional1/Additional2, it can generate the following error codes:

ErrorID	Additional1	Additional2	Description
SYSTEM_ERROR (16#7FFF)	16#F0260001	0	Error while initializing the function block. No connection could be established to the parameter system
COMMUNICATION_ERROR (16#0002)	16#F0260004	xx	Runtime error in the function block: Parameter could not be read xx refers to the parameter number (A-parameter)
INPUT_INVALID_ERROR(16#0001)	16#F0260005	0	Runtime error in the function block: Invalid combination of units for velocity, time and position
INPUT_INVALID_ERROR(16#0001)	16#F0260006	0	Runtime error in the function block: The value "Duration" of a switch is too high

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#F0260007	0	Runtime error in the function block: Invalid state of the state machine
RESSOURCE_ERROR	16#F0260040	0	Reference to the basic system could not be found
RESSOURCE_ERROR	16#F0260041	0	The function block could not be instantiated
SYSTEM_ERROR	16#F0260042	0	Function block may not be within the RETAIN data array

Fig.5-57: Error codes of the MC_DigitalCamSwitch function block

5.6 Temperature Controller Function Block

5.6.1 IL_TempControlType01

Brief Description

The IL_TempControlType01 function block controls the temperature. It can determine the controller parameters either independently or with specifications. Special functions like monitoring, pause and manual operation can also be activated.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx18 (in preparation)	In preparation

Fig.5-58: Reference table of IL_TempControlType01 function block

Interface Description

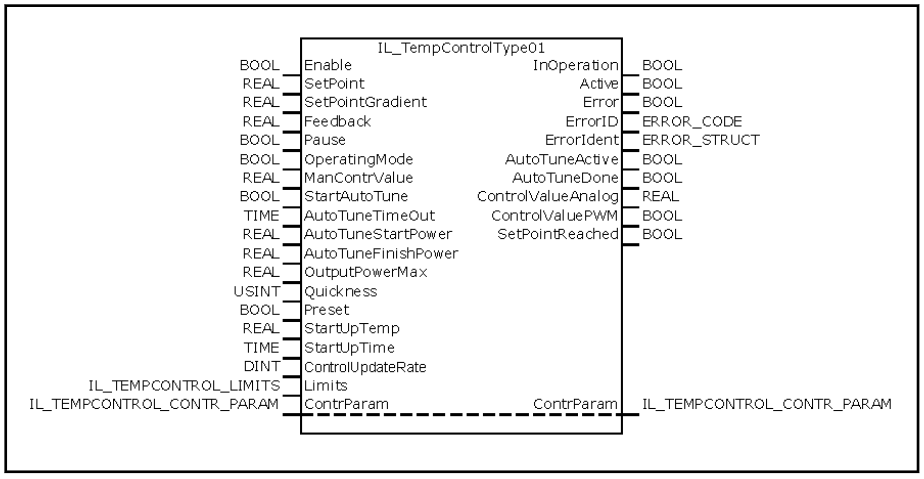


Fig.5-59: IL_TempControlType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	The function block is active if this input is "TRUE". The function block is reset to its initial values with a falling edge. The input is read in each cycle
	SetPoint	REAL	Command temperature Unit in [u] The input is read in each cycle

ML_TechBase.library

I/O type	Name	Data type	Description
	SetPointGradient	REAL	Maximum adjustment of the parameter "SetPoint" in [rps] The input is read in each cycle. The preset value is "0 [rps]" (not active)
	Feedback	REAL	Input for the actual temperature Unit in [u] The input is read in each cycle
	Pause	BOOL	This input "freezes" the control variable and the controller calculation. The input is read in each cycle. The preset value is "FALSE" (not active)
	OperatingMode	BOOL	Changes the controller mode from Automatic mode to Manual mode: AUTO:=FALSE, MANUAL:=TRUE. The preset value is "FALSE" (AUTO). The input is read in each cycle
	ManContrValue	REAL	It specifies the control variable in [%] for the manual operation but will only be actively read if "OperatingMode:=1". The input is read in each cycle. The preset value is "0%" (no control variable)
	StartAutoTune	BOOL	A rising edge starts the identification process. The PID controller parameters is calculated subsequently but not actively until a falling edge is present. The input is read in each cycle. The preset value is "FALSE" (not active)
	AutoTuneTimeOut	TIME	Watchdog responsible for the time monitoring during the identification process The preset value is "10000 • T_A". The input is read by "StartAutoTune" at a rising edge
	AutoTuneStartPower	REAL	Control variable in [%] representing the lower limit of the identification process. The preset value is "0%". The input is read by "StartAutoTune" at a rising edge
	AutoTuneFinishPower	REAL	Control variable in [%] representing the upper limit of the identification process. The preset value is "50%". The input is read by "StartAutoTune" at a rising edge
	OutputPowerMax	REAL	Maximum output control variable in [%] The preset value is "100%". The input is read in each cycle

ML_TechBase.library

I/O type	Name	Data type	Description
	Quickness	USINT	Quickness of the controller The preset value is "3" (very slight overshoot). The input can be selected in the range from "1" (quick control, strong overrun) to "5" (slow control, no overrun). The input is read by "StartAutoTune" at a rising edge
	Preset	BOOL	A rising edge applies the controller parameters of the "ContrParam" structure. The preset value is "FALSE" (not active). The input is read during each cycle and becomes active only in case of "Enable" = TRUE
	StartUpTemp	REAL	Startup temperature in [u] The input is read by "Enable" at a rising edge. The preset value is "0u" (not active). Only becomes active if a value not equal to "0" is assigned to the "StartUpTime" input
	StartUpTime	TIME	Startup time, i. e. the period during which the startup temperature ("StartUpTemp") is maintained until the "normal" closed-loop control is started. The preset value is "T#0s" (not active). The input is read by "Enable" at a rising edge
	ControlUpdateRate	DINT	This factor specifies how much slower the controller works compared to the task of the function block. The sampling time results from multiplying the cycle time (= task interval time) with "ControlUpdateRate"
	Limits	IL_TEMPCONTROL_LIMITS	Structure containing the alarm limits in [u] (units). The preset value is always "0u" (not active). The input is read in each cycle. If values are assigned to the structure, Limits.UpperAlarm has to be > Limits.LowerAlarm and (Setpoint + Limits.UpperAtSetpoint) has to be >= (Setpoint - Limits.LowerAtSetpoint). Otherwise, an error is generated
VAR_OUTPUT	InOperation	BOOL	The temperature control is active in case of a set output
	Active	BOOL	This output is set if "Enable" is set and if the controller is active
	Error	BOOL	An error has occurred during operation. The control variable is set to "0%"
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	In case of a set "Error" output, this output contains a detailed error information
	AutoTuneActive	BOOL	The identification process is active

ML_TechBase.library

I/O type	Name	Data type	Description
	AutoTuneDone	BOOL	The identification process has been successfully completed
	ControlValueAnalog	REAL	Analog value of the control variable in %
	ControlValuePWM	BOOL	Digital, pulse-width modulated control variable value, either "0" or "1"
	SetPointReached	BOOL	A status output indicates whether the actual value is within the limits ("Setpoint" + "Limits.UpperAtSetpoint") >= actual value >= (Setpoint - "Limits.LowerAtSetpoint")
VAR_IN_OUT	ContrParam	IL_TEMPCONTROL_CONTR_PARAM	This structure contains the current parameters K_P , K_I and K_D of the controller as well as the dead time T_t

Fig.5-60: Interface variables of the IL_TempControlType01 function block

Data structure interface description

The following table lists the data structures of the L_TempControlType01 function block.

Name of the structure	Structure element	Type
IL_TEMPCONTROL_LIMITS	UpperAlarm	REAL
	LowerAlarm	REAL
	UpperAtSetpoint	REAL
	LowerAtSetpoint	REAL
IL_TEMPCONTROL_CONTR_PARAM	KP	REAL
	KI	REAL
	KD	REAL
	Tt	TIME

Fig.5-61: Data Structures

Time diagram

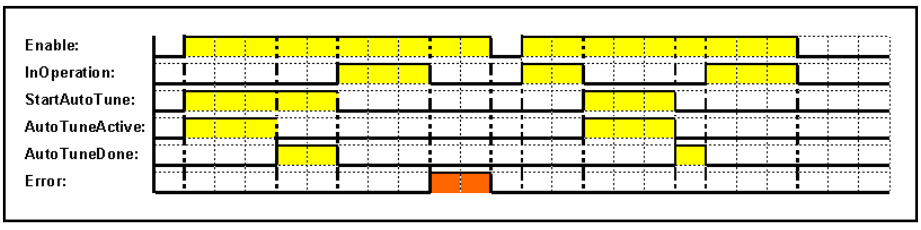


Fig.5-62: Time diagram

Error description

ErrorID	Table	Additional1	Additional2	Description
STATE_MACHINE_ERROR	F_RELATED_TABLE	16#1600	Current status	Error in the state machine, invalid state
INPUT_RANGE_ERROR	F_RELATED_TABLE	16#1601	16#0001	Input values for a maximum output value ("OutputPowerMax", "AutoTuneStartPower" or "AutoTuneFinishPower") not within the valid range. Value must be higher than/equal to 0 [%] up to max. 100 [%]
			16#1602	Input values for alarm limits invalid (Condition: Limits.UpperAlarm > Limits.LowerAlarm)
			16#1603	Input value for manual control variable invalid. Value must be in the range from 0 [%] to 100 [%]
			16#0004	Controller parameter invalid
			16#000A	Start value and end value for identification are too close to each other. The minimum difference is 5 [%]
			16#000B	The "SetpointGradient" input is invalid. Value has to be higher than 0
			16#000C	Input values invalid for status limits (Condition: (Setpoint + Limits.UpperAtSetPoint) > (Setpoint - Limits.LowerAtSetPoint)
			16#000D	Time for identification is too long ("AutoTuneTimeOut")
			16#000E	Value for "Quickness" is not within the valid limits (1 - 5)
			16#000F	Value for "ControlUpdateRate" is not within the valid limits (1 - 1000)
SYSTEM_ERROR	F_RELATED_TABLE	16#1602	Sampling time	Jittering the sampling time > 20 [%]. Error does not become active before 2 successive cycle timeouts
			16#0005	Gradient of actual value is too high, possibly due to sensor error (diagnostics)
			16#0006	Error in the control loop, sensor possibly displaced or destroyed

ML_TechBase.library

ErrorID	Table	Additional1	Additional2	Description
CALCULATION_ERROR	F_RELATED_TABLE	16#1603	16#0007	Identified controlled system parameters are invalid, self-identification of controlled system failed
			16#0008	The identified process model is not correct/invalid
ACCESS_ERROR	F_RELATED_TABLE	16#1604	Field number of the array	Incorrect access to internal variable
OTHER_ERROR	F_RELATED_TABLE	16#1605	16#0009	Time elapsed during identification (watchdog)
DEVICE_ERROR	F_RELATED_TABLE	16#1606	Temperature	Temperature error limit exceeded, temperature not within the error limits

Fig.5-63: Error codes of the IL_TempControlType01 function block

Information on specific errors

- Error

  "Gradient of the actual value is too high"

  (Additional1 = 16#1602; Additional2 = 16#0005):

  Gradient of the actual value is monitored. If the difference between two consecutive actual values is higher than 10 r./ms, this error is generated. This may be the case when using an inline temperature model, since error analyses are displayed as measured values with values higher than 32768 (bit 15 set). If this analysis is not evaluated by the user, this error message may be generated.

- Error

  "Error in the control loop, sensor possibly displaced or destroyed"

  (Additional1 = 16#1602; Additional2 = 16#0006):

  It is monitored, whether the actual value approaches the command value in case of a maximum output performance. For this purpose, time monitoring is started which, when the I-value is active, is equal to

  2000/Ki + Tt and without the I-value equals

  2000 * sampling time + Tt

  If the actual value does not approach the command value during this period, the error is generated. The sensor might be defective/displaced or the heating/cooling system does not work. The error may also occur if no dead time (Tt) is specified when specifying the controller parameters via the structure "`IL_TEMPCONTROL_CONTR_PARAM`". In this case, the value of the dead time should be estimated and the controller parameters should be set again. This is accomplished automatically when identifying the controlled system.

- Error

  "Identified controlled system parameters are invalid, self-identification of controlled system failed"

  (Additional1 = 16#1603; Additional2 = 16#0007):

  The controlled system parameters resulted in invalid values. This may be due to an incorrect sampling time or due to incorrect actual values (sensor not connected). See also "Recommendations for selecting the sampling time" on page 80.

- Error

  "**The identified process model is not correct/invalid**"

  (Additional1 = 16#1603; Additional2 = 16#0008):

  When identifying the controlled system, the calculated model is compared to the measured values. The comparison did not show any satisfying results. The model shows a deviation of more than 20 % from the measured values. The sampling time might not have been chosen correctly, or the controlled system does not show an aperiodic behavior. Disturbance variables occurring during the identification process can also lead to this effect.

- Error

  "**Time elapsed during identification (watchdog)**"

  (Additional1 = 16#1605; Additional2 = 16#0009):

  When identifying the controlled system, the system waits until the actual value is in steady state. Subsequently, a command value step change is performed and the system waits again until the actual value is in the steady state. If this requires too much time, the error is generated. The reason is either a strongly varying actual value or an incorrectly selected sampling time (see also next section).

- Error "**Time for identification is too long ('AutoTuneTimeOut')**"

  (Additional1 = 16#1601; Additional2 = 16#000D):

  When identifying the controlled system, the time monitoring can be set. This value can have a maximum of 10000 * sampling time. In this case, the time monitoring has to be set to 0 (maximum possible time for identification in case of a given sampling time) or the sampling time must be increased correspondingly.

Application

The function block can be used for IndraLogic XCL and IndraMotion MLC with the help of inline temperature modules (IL TEMP 2 RTD). Currently, the inline modules are not yet available for IndraMotion MLD. In general, however, each value can be used as temperature value since the temperature controller does not directly access the hardware. Thus, it is also possible to obtain temperature values, for example, via field bus connections.

Identification

The automatic identification, also known as "AutoTune", identifies the parameters of the controlled system to determine the parameters of the controller. First, certain model parameters are calculated. Subsequently, the model is compared with the real controlled system to avoid incorrect parameterizations of the controller parameters.

Before the jump is activated, it must be ensured that the controlled system is in passive state. Therefore, a time monitoring is started. The preset value for this time monitoring is 5000, multiplied with the root of the sampling time in [ms]. The actual value must not exceed the tolerance limits of the defined tolerances prior to the end of the time monitoring. Otherwise, the time monitoring is restarted and the system waits until the specified time has elapsed. The limit for the temperature window containing the actual value for the time of the time monitoring is calculated as follows:

$$\left|\mathrm{IdentTemp}-\mathrm{ActTemp}\right| < \sqrt{\left|\frac{\mathrm{IdentTemp}}{10}\right|}+0.5$$

The variable "IdentTemp" is the value for which the tolerance window of the actual temperature is created. Waiting for the stationary final value is performed in the same way.

The quality of the determined model is estimated. If the deviation is too big, the function block displays the error

"16#1603" (CALCULATION_ERROR)

with the identifier "16#0008" (invalid process model).

5.6.2 Controller

Setting the controller parameter

Equation for the controlled system:

$$G_S(s) = \frac{K_S}{(1+sT_{SUM})} e^{-sT_t} \approx \frac{K_S}{(1+sT_t)(1+sT_{SUM})}$$

Equation for the controller:

$$G_R(p) = K_p(1+\frac{1}{sT_N}+T_V s) = K_p + \frac{K_I}{s} + \frac{K_D s}{1+sT_f}$$

Setting to optimum value:

$$T_N = T_{SUM}$$

$$T_V = T_t$$

$$K_p = \frac{T_N}{a \cdot K_S \cdot T_t}$$

or

$$K_I = \frac{K_p}{T_{SUM}}$$

$$K_D = T_t \cdot K_p$$

$$K_p = \frac{T_{SUM}}{Quickness \cdot K_S \cdot T_t}$$

Quickness = 1...5;

1 = very fast with large overrun,

5 = very slow without overshoot;

Preset value is quickness = 3.

Setting the filter time for the difference:

$$T_f = T_V / 10$$

This value is fixed and cannot be changed. The calculated controller parameters is entered into the provided structure "IL_TEMPCONTROL_CONTR_PARAM".

The entry

KP corresponds to the proportional gain factor

KI corresponds to the integration constant

KD corresponds to the differentiation constant and

Tt corresponds to the dead time

If these parameters are specified by the user, an estimated value should at least be entered for the dead time to avoid the error

"Error in the control loop, sensor possibly displaced or destroyed"

(Additional1 = 16#1602; Additional2 = 16#0006),

to be avoided.

Controller algorithm

An additive $PIDT_1$ algorithm with anti-wind-up of the I-part and PT_1 filtering of the D-part has been implemented. The additive form allows to deactivate certain parts of the controller so that, for example, the structure of a PI-controller can be used by parameterizing the D value to "zero".

The I-part is limited with the help of an anti-wind-up limiter which has been internally preset to a value of 90% of the control variable.

The saturation limit of the controller is the maximum output value (VAR_IN OutputPowerMax) of the controller. The preset value is 100%.

Controller structure

In the following, the $PIDT_1$ controller structure with anti-wind-up limitation and limitation of the maximum control variable are displayed. Due to the additive structure, the parts of the controller can be individually deactivated. Thus, pure P-, P-I or PDT_1-controller can be parameterized as well. However, the individual parts have to be deactivated by the user. For this purpose, the desired values have to be entered into the structure "contrParam". At a rising edge at the "Preset" input, these new parameters are activated in the controller. By default, the controller works with the complete $PIDT_1$ structure.

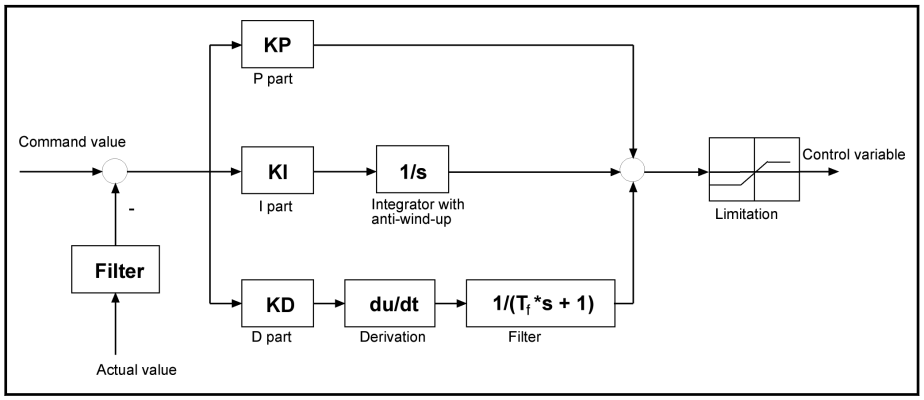


Fig.5-64: Controller structure

The actual temperature value is smoothed with a PT_1 filter both during the identification and the control process to avoid stochastic failures. The time constant of the filter is fixed to 20 times the cycle time.

Control loop monitoring

The function block is provided with a control loop monitoring option. In case of a control variable output of "100%" (full power), the temperature has to change in the direction of the set sign in K_P. Otherwise, a timer is activated displaying an error when the time has elapsed. The timer is reset once the temperature moves into the correct direction. The values for the timer are set as follows:

In case of an existing I-value, a value of $(2000/K_I)$ + dead time is used. If no I-value is used, the value 2000 * sample time is only preset. The timer is reset even if the control variable becomes smaller than "100%". It is important to

specify the dead time (Tt) in the structure for the controller parameters as well if the controller parameters are specified by the input "Preset". In case of a high K_I value and a high dead time value, the monitoring process could be started. Due to this monitoring process, sensor errors (e. g. by being shifted) or actuating element errors can be detected.

In addition, the function block also monitors the gradient of the actual value; however, this monitoring function becomes active only in the case of large deviations. These deviations can occur, for example, if the sensor displays an error code and if this error code is not recorded/evaluated by the user. The gradient is fixed to a value of 10 rev./ms. A single deviation is tolerated before an error is generated.

5.6.3 Using the Function Block

The function block should be operated in a cyclic task. The function block automatically detects the task interval time (= cycle time) and adjusts the control algorithm accordingly. The controller works with the sampling time resulting from the task interval time multiplied with the input variable "ControlUpdateRate". If "ControlUpdateRate" = 1, the sampling time corresponds to the task interval time. The sampling time is used for the identification of the controlled system as well as in case of an activated controller. All other functions of the function block (read actual values, gradient monitoring, PWM output, manual closed-loop control) use the task interval time. Thus, it is possible to separate the control/identification process from other functions of the function block (with regard to time). This is especially important to separate the output of the control variable from the control/identification process with the help of PWM.

The task interval time forms the basis for the pulse-width modulation (PWM), i. e. the task interval time is the smallest possible time slice which can be set to "ON" or "OFF" in case of pulse-width modulation. One period of the PWM is 20 times longer than the task interval time. Thus, the control variable can be changed in steps of 5% between 0% and 100%.

Example:

The task interval time is parameterized to 20 ms for the cyclic task. This means that a new value for the control variable can be transferred to the PWM every 400 ms. Thus, the possible activation or deactivation times for the PWM are:

Control variable [%]	Switch-on duration [ms]	Switch-off period [ms]
0	0	400
5	20	380
10	40	360
...		
50	200	200
...		
95	380	20
100	400	0

Fig.5-65: Example for activation/deactivation times

The task interval time and "ControlUpdateRate" always have to be selected depending on the control element and the controlled system. If the task interval times are too short, the control element will wear too quickly (for example,

relay modules). If the task interval times/sampling times are too long, the accuracy of the closed-loop control is reduced.

Furthermore, the sampling time is monitored, i. e. if the sampling time exceeds/falls below a value of more than 20%, an error is generated and the output "Error" is set to TRUE.

Recommendations for selecting the task interval time

The task interval time/cycle time should be set in a way so that the PWM does not cause any visible temperature changes within a PWM period of 20*task interval time. If, in case of a control variable of 20%, the temperature of the controlled system strongly increases during the heating process and if the temperature is reduced while the heating is switched off, this means that the task interval time is too long. This can be seen in the following figure where the upper curve displays the temperature curve and the lower curve displays the heating phases (TRUE). It is clearly visible that the controlled system is heated with every heating process and that the controlled system cools down while the heating is deactivated.

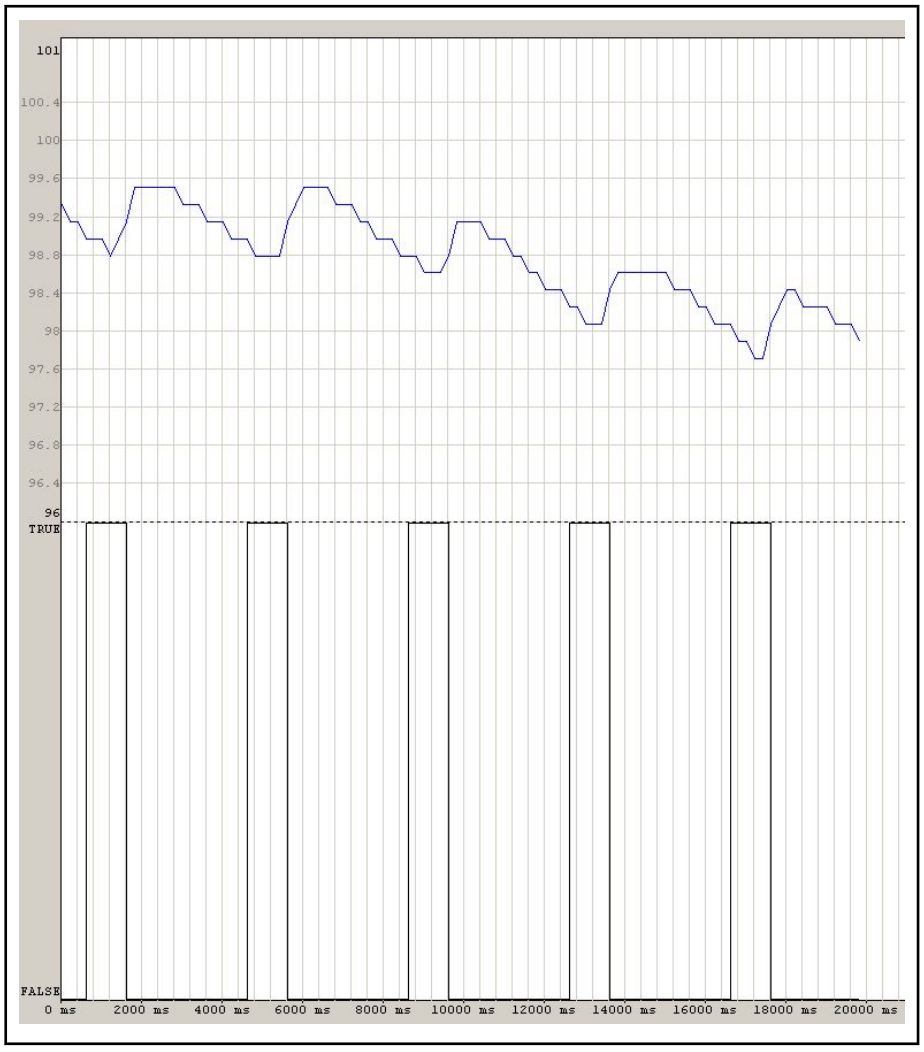


Fig.5-66: Task interval time

In contrast, it is also not recommended to choose a too short task interval time since the control element is then subject to higher loads (e. g. the relay module has to perform a lot of switching processes).

An appropriate task interval time has to be determined with the help of certain tests.

Recommendations for selecting the sampling time

First, the task interval time should be selected as described in the previous section. Then, the sampling time can be set to the desired value by using the input "ControlUpdateRate".

The following recommendation applies to the sampling time:

- 10 ms for quick/dynamic controlled systems (time constant for the controlled system from 200ms to 2s)
- 100 ms for average controlled systems (time constant for the controlled system from 2s to 20s)
- 1s to 30s for slow/static controlled systems (time constant for the controlled system from 20s to 100 min).

The sampling time should be approximately 20 to 200 times faster than the "dominating time constant" of the controlled system. The "dominating time constant" is approximately the time a controlled system requires to pass through 63% of the change of a jump. In principle, however, the sampling time can be freely selected by the user.

If the sampling time has been selected too short, the error "Time elapsed during identification (watchdog)" (Additional1 = 16#1605; Additional2 = 16#0009) may occur during the identification process. In this case, a longer sampling time has to be selected since the controlled system was not in steady state when the time monitoring process had expired.

If the sampling time selected is too long, the error "Identified controlled system parameters invalid, self-identification of the controlled system failed" (Additional1 = 16#1603; Additional2 = 16#0007) can occur during the identification process. In this case, a shorter sampling time has to be selected since the intervals for measuring the behavior of the controlled system were too long.

Recommendation for the identification

For identification, some rules are necessary to achieve good results. These are explained more thoroughly here.

In general, temperature systems are not linear. For low temperatures, less power is required to achieve a certain temperature increase than at higher temperatures. At low temperatures, a controller needs different settings than at high temperatures.

For this reason, it is particularly important to carry out the identification even in the temperature range in which the controller is to function later. There are two possibilities here:

1. For very precise closed-loop control in a small temperature range the power required for manual operation should first be specified such that the desired temperature is achieved. Then an identification can be carried out in which the value for "AutoTuneStartPower" is somewhat lower than the calculated value and the value for "AutoTuneFinishPower" is somewhat higher than the calculated value. Here the controller parameters are determined precisely for this temperature range. However, outside of the temperature range the closed-loop control is less precise.
2. For a wide closed-loop control range the power required for the lowest and highest temperature in manual operation should be determined. These then become the values for "AutoTuneStartPower" and "AutoTuneFinishPower". The closed-loop control is then averaged over the entire temperature range. In general, this means a somewhat more dynamic control at low temperatures and more restrained closed-loop control at higher temperatures.

5.6.4 Description of the Individual Functions, Examples

With regard to the following use cases, the following variables have to be specified in the declaration section of the user program for the configuration of the function block.

Program:

```
VAR
 fb_TempControl:IL_TempControlType01;       (* Instance of the function blocks *)
 strContrParam: IL_TEMPCONTROL_CONTR_PARAM; (* Structure containing the controller parameters *)
 strLimits: IL_TEMPCONTROL_LIMITS;          (* Alarm/status limits *)
 rActualTemp:REAL;       (* Actual temperature*)
 rActualPower:REAL;      (* Control variable, analog *)
 bActualPower:BOOL;      (* Control variable, digital (PWM) *)
 bAutoTuneDone:BOOL;     (* Identification state *)
 bSetPointReached:BOOL;  (* Actual value is within the command value window *)
 bError:BOOL;            (* State of the error output *)
 iState:INT;             (* Value in SFC *)
END_VAR
```

To make the description more simple, no errors are described in the following examples.

Simple closed-loop control with a controller parameter specification

In the structure "strContrParam", the function block is parameterized with the desired controller parameters. In the case of a rising edge at the input "Preset", new controller parameters can also be parameterized during the closed-loop control process. The command value ("Setpoint") has to be preset and the actual value ("Feedback") has to be returned. The control variable can either be retrieved at the analog output ("ControlValueAnalog") or at the digital output ("ControlValuePWM").

Program:

```
(* Setting the controller parameters *)
strContrParam.KP := 1.5;
strContrParam.KI := 2;
strContrParam.KD := 0.6;
strContrParam.Tt := T#1s;


(* Starting the function block at simultaneous transfer *)
(* of the controller parameters *)
fb_TempControl(   Enable:=TRUE,
                  Setpoint:=50,
                  Feedback:=rActualTemp,
                  Preset:=TRUE,
                  ContrParam:=strContrParam,
                  ControlValueAnalog=>rActualPower,
                  ControlValuePWM=>bActualPower);
```

Thus, the controller parameters will only be transferred once and the controller works with the specified values.

Simple identification of a controlled system with a subsequent transition to the closed loop control

First, the function block is set to the identification or AutoTune mode. After having successfully completed the identification, the closed-loop control is started by generating a falling edge at the input "StartAutoTune".

The calculated controller parameters are saved in the structure "strContrParam" and are thus available to the user. Thus, these controller parameters can be saved in the retain area of the control by the user (see also the example "Retaining the controller parameters" on page 86). Another identification process is thus no longer required when starting the closed-loop control again but the simple control with parameter specification can be started immediately.

With the optional input "Quickness", a slightly quicker setting process has been specified for the controller (quicker control but higher disposition for overshoots).

The identification is performed with a jump from 0% ("AutoTuneStartPower") to 10% ("AutoTuneFinishPower") output power.

Program:

```
CASE iState OF
1: (* Identification  *)
      fb_TempControl(   Enable:=TRUE,
                        StartAutoTune := TRUE
                        Feedback:=rActualTemp,
                        AutoTuneStartPower:= 0,
                        AutoTuneFinishPower:= 10,
                        Quickness:=2,
                        ContrParam:=strContrParam,
                        ControlValueAnalog=>rActualPower,
                        ControlValuePWM=>bActualPower,
                        AutoTuneDone=>bAutoTuneDone);
       IF (bAutoTuneDone = TRUE) THEN
             iState := 2;
       END_IF


2: (* Transition to closed-loop control *)
      fb_TempControl(   Enable:=TRUE,
                        StartAutoTune := FALSE,
                        Setpoint:=50,
                        Feedback:=rActualTemp,
                        ContrParam:=strContrParam,
                        ControlValueAnalog=>rActualPower,
                        ControlValuePWM=>bActualPower);
```

Identification of the controlled system in the operating point

During a process control, another identification in the operating point may be desired. In the following, the corresponding procedure is described based on the previous example "Simple identification of a controlled system with subsequent transition to a closed-loop control". By setting the input "AutoTuneStart", the identification is started. The initial and final parameters of the control variable parameters are transferred to "AutoTuneStartPower" and "AutoTuneFinishPower". After a successful identification, the new controller parameters are available in the structure "strContrParam".

In the example, the operating range is calculated based on the actual value of the control variable (in %). The prerequisite is, of course, that the controlled system is in steady (stationary) state. In the example, the operating range has been selected from "control variable -10%" to "control variable +10%". The identification in the operating point starts with changing to state "3".

Program:

```
CASE iState OF
2: (* Closed-loop control *)
      fb_TempControl(   Enable:=TRUE,
                         StartAutoTune := FALSE,
                         Setpoint:=50,
                         Feedback:=rActualTemp,
                         ContrParam:=strContrParam,
                         ControlValueAnalog=>rActualPower);
      If (bIdent = TRUE) THEN
           iState := 3;
      End_IF;


3: (* Identification in the operating point *)
         fb_TempControl(   Enable:=TRUE,
                         StartAutoTune := TRUE,
                         AutoTuneStartPower:=rActualPower-10,
                         AutoTuneFinishPower:=rActualPower+10,
                         Feedback:=rActualTemp,
                         ContrParam:=strContrParam,
                         ControlValueAnalog=>rActualPower,
                         AutoTuneDone=>bAutoTuneDone);


      IF (bAutoTuneDone = TRUE) THEN
         iState := 2;
      END_IF
```

Definition of a start-up ramp

If a process prior to the actual temperature control requires the system to wait at a certain temperature for a certain time (start-up ramp), this function is available via the inputs "StartUpTemp" and "StartUpTime".

First, the controller controls until the start-up temperature. If this is not achieved, the start-up time "StartUpTime" expires and the controller maintains a constant start-up temperature. After the start-up time "StartUpTime" has elapsed, the start-up ramp is completed and the controller shows its normal behavior. The start-up ramp is performed once after each change from "Enable" to "TRUE".

In the example, the controller controls to a temperature of "100", then it waits for 5 minutes and 30 seconds and finally it controls to the desired temperature of "150" (see fig. 5-68 "Using a command value gradient and a start-up ramp" on page 85).

Program:

```
fb_TempControl(   Enable:=TRUE,
                    Setpoint:=150,
                    Feedback:=rActualTemp,
                    Preset:=TRUE,
                    StartUpTemp:=100,
                    StartUpTime:=t#5m30s,
                    ContrParam:=strContrParam,
                    ControlValueAnalog=>rActualPower);
```

Definition of alarm and status limits

Alarm and status limits can be set for process monitoring purposes.

The structure "Limits" contains the values "UpperAlarm" and "LowerAlarm". When exceeding the alarm limits, an error is generated. The monitoring function only becomes active if the current process temperature ranges between the values of "UpperAlarm" and "LowerAlarm" so that no error is generated during the start-up of the process. These values are absolute values.

The structure "Limits" also contains the values "UpperAtSetPoint" and "LowerAtSetPoint". The output "SetPointReached" is set as soon as the condition ("Setpoint – Limits.LowerAtSetPoint") ≤ actual value ≤ ("Setpoint + Limits.UpperAtSetPoint") has been fulfilled. These values are relative values related to the "Setpoint". In this way, the "Setpoint" can be modified without having to change "Limits.LowerAtSetPoint" and "Limits.UpperAtSetPoint". Based on this status display it can be determined if the desired temperature has already been reached.

Program:

```
strLimits.UpperAlarm:=250;
strLimits.LowerAlarm:=70;
strLimits.UpperAtSetpoint:=20;
strLimits.LowerAtSetPoint:=20;

fb_TempControl(   Enable:=TRUE,
                   Setpoint:=150,
                   Feedback:=rActualTemp,
                   Limits:=strLimits,
                   ContrParam:=strContrParam,
                   ControlValueAnalog=>rActualPower,
                   SetPointReached=>bSetPointReached);
```

ML_TechBase.library

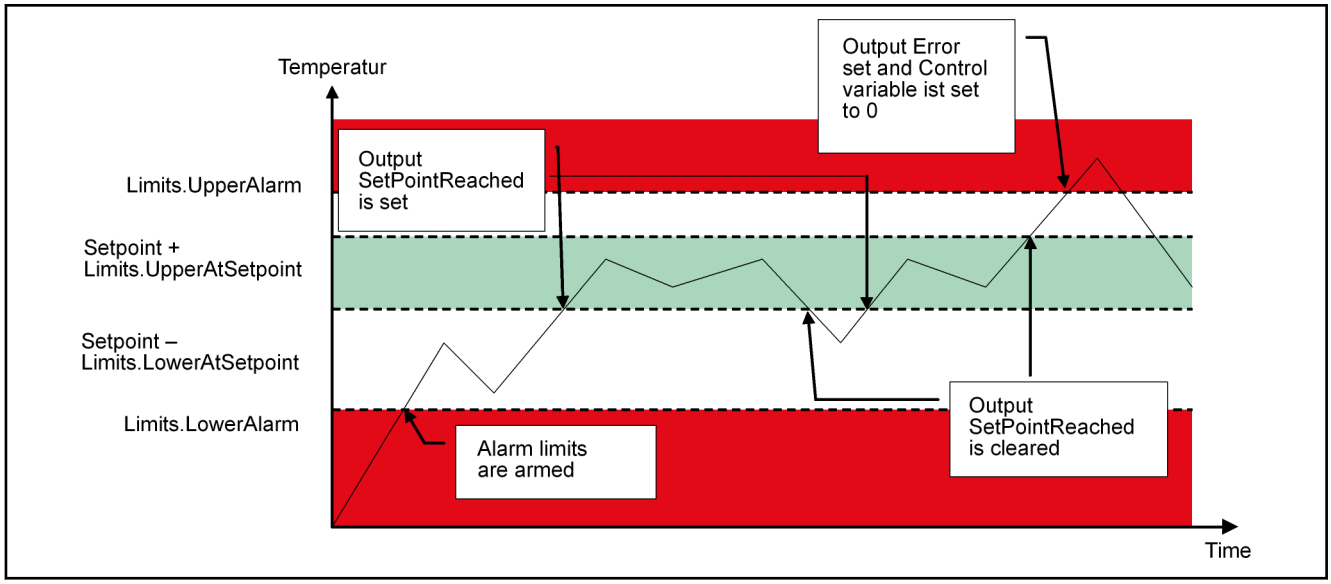


Fig.5-67: Mode of operation of the alarm and status limits

Pausing the closed-loop control (PAUSE mode)

Pausing the closed-loop control can be implemented via the "Pause" input. The control variable remains constant and the controller as well as all its values are "frozen". After having reset the input "Pause", the closed-loop control continues.

In the example, the temperature is controlled to "150" in state "4". By switching to state 5, the closed-loop control is "frozen" and no new control variables are output. By returning to state "4", the closed-loop control is continued and new control variables are output again. A jump may occur if the actual value has strongly changed in the meantime.

Program:

```
CASE iState OF
4:
fb_TempControl(   Enable:=TRUE,
                    Pause:=FALSE,
                    Setpoint:=150,
                    Feedback:=rActualTemp,
                    Preset:=TRUE,
                    ContrParam:=strContrParam,
                    Limits:=strLimits,
                    ControlValueAnalog=>rActualPower);

5:
fb_TempControl(   Enable:=TRUE,
                    Pause:=TRUE,
                    Setpoint:=150,
                    Feedback:=rActualTemp,
                    Preset:=TRUE,
                    ContrParam:=strContrParam,
                    Limits:=strLimits,
                    ControlValueAnalog=>rActualPower);
```

Analog/digital control variable output

The function block is provided with two control variable outputs: one analog output ("ControlValueAnalog") which displays REAL values between "0%" (no power) and "100%" (full power), and one digital output ("ControlValuePWM") displaying the BOOL values "FALSE" (no power) or "TRUE" (full power). The digital output can be used for pulsing the power in the case of relay modules. The pulsing process is performed with the help of a pulse-width modulation whose cycle time is 20 times slower than the time of the cyclical task in which the function block is running. Thus, control variables can be modulated in steps of 5% with a resolution between 0% to 100%.

The analog output can be used for an analog output or for monitoring the current control variable.

Program:

```
fb_TempControl(   Enable:=TRUE,
                  Pause:=FALSE,
                  Setpoint:=150,
                  Feedback:=rActualTemp,
                  Preset:=TRUE,
                  ContrParam:=strContrParam,
                  Limits:=strLimits,
                  ControlValueAnalog=>rActualPower
                  ControlValuePWM=>bActualPower);
```

Specifying a command value gradient

The function block provides the possibility to limit (to ramp) the increase of the command value. This reduces the overrun and prevents the unit from damages. For each new specification of the input "SetPoint", the command value is internally changed in the form of ramps (not jumps) until the desired command value is reached. "SetPointGradient" is specified in [rps].

In the example, the command temperature is controlled to "150" with an increase of "0.5 per second".

Program:

```
fb_TempControl(   Enable:=TRUE,
                  Pause:=FALSE,
                  Setpoint:=150,
                  Setpointgradient:=0.5,
                  Feedback:=rActualTemp,
                  Preset:=TRUE,
                  ContrParam:=strContrParam,
                  Limits:=strLimits,
                  ControlValueAnalog=>rActualPower);
```

A ramping process with command value gradient and an additional start-up ramp would show the following development of command value and actual value. Here, a target temperature ("Setpoint:=200") of "200°C" has been controlled with a gradient ("Setpointgradient:=1") of "1°C/s". The start-up temperature ("StartUpTemp:=100") is "100°C" and the start-up time ("StartUpTime:=T#50s") is "50s".

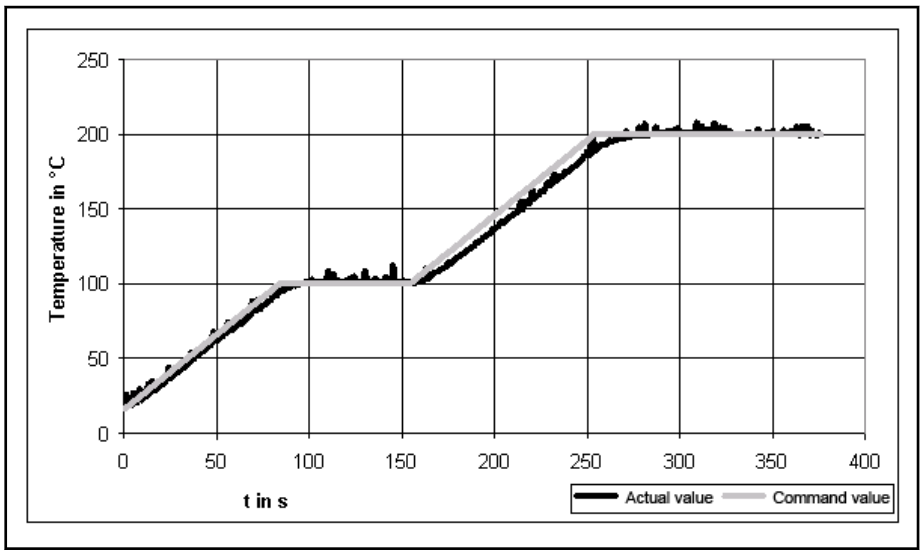


Fig.5-68: Using a command value gradient and a start-up ramp

Specifying a manual control variable

A manual control variable can be defined to control the temperature manually. At the same time, this variable represents the control variable at the output of the function block. A manual control variable may be required to record

manual testing functions or to activate an emergency mode in the case of a sensor error.

In the example, the function block is manually controlled with a control variable of "50%".

Program:

```
fb_TempControl(   Enable:=TRUE,
                  Pause:=FALSE,
                  OperatingMode:=TRUE,
                  ManContrValue:=50,
                  ContrParam:=strContrParam,
                  Limits:=strLimits,
                  ControlValueAnalog=>rActualPower);
```

Retaining the controller parameters

To ensure that the controller parameters are safely stored even in the case of a power failure, they can be saved in the retain area. By doing so, no identification process has to be performed when restarting the controller but the closed-loop control can be started immediately.

For this purpose, the user has to create the structure "strContrParam" as retain variables ("VAR RETAIN") which then contains the values of the individual controller parameters. At the same time, the user has to create a mechanism to differentiate at the initial start-up if an identification process has already been performed.

Additional retain variables have to be created in the header.

Program:

```
VAR RETAIN
   strContrParam:  STRUCT_CONTR_PARAM;
   bIdentification: BOOL := FALSE;
END_VAR

If (bIdentification = FALSE) THEN
     fb_TempControl(   Enable:=TRUE,
                        StartAutoTune := TRUE,
                        Feedback:=rActualTemp,
                        Quickness:=2,
                        ContrParam:=strContrParam,
                        ControlValueAnalog=>rActualPower,
                        ControlValuePWM=>bActualPower,
                        AutoTuneDone=>bAutoTuneDone);
  IF (bAutoTuneDone = TRUE) THEN
     bIdentification:= TRUE;
  END_IF
ELSE
     fb_TempControl(   Enable:=TRUE,
                        StartAutoTune := FALSE
                        Preset:=TRUE,
                        Setpoint:=150,
                        Feedback:=rActualTemp,
                        ContrParam:=strContrParam,
                        Limits:=strLimits,
                        ControlValueAnalog=>rActualPower);
END_IF;
```

In the above example, the values of the controller are stored in the retain variables in each cycle. After a deactivation/power failure, the code checks with the help of the Boolean variable "bIdentification" if there has already been an identification. If this is not the case, an identification is performed. Otherwise, the controller can be restarted with the saved (old) parameters. A rising edge has to be present at the input "Preset".

Troubleshooting with the substitute value strategy

By default, the function block displays the control variable "0%" (no power) in the case of an error. This is done for the safety of the system. If the user wants to use a control with a certain power, for example, to avoid a strong cool down of a system, it is recommended to use the substitute value strategy described in the following example.

Errors can be checked in state "6" with "bError". If this condition is "TRUE", the function block is reset ("Enable:=FALSE") and state "7" is activated.

In state "7", the function block is restarted with a rising edge at the input "Enable" and operated in Manual mode (see "Specifying a manual control variable" on page 85).

In the example, a control variable of "50%" is output.

Further information on the error is displayed at the outputs "ErrorID" and "ErrorIdent" which were not analyzed in this example.

Program:

```
CASE iState OF:
6:
fb_TempControl(   Enable:=TRUE,
                  Setpoint:=50,
                  Feedback:=rActualTemp,
                  Preset:=TRUE,
                  ContrParam:=strContrParam,
                  Limits:=strLimits,
                  ControlValueAnalog=>rActualPower,
                  Error=>bError);

IF (bError = TRUE) THEN
 fb_TempControl(  Enable:=FALSE,
                  ContrParam:=strContrParam,
                  Limits:=strLimits);
 iState := 7;
END_IF

7:
fb_TempControl(   Enable:=TRUE,
                 Pause:=FALSE,
                 OperatingMode:=TRUE,
                 ManContrValue:=50,
                 ContrParam:=strContrParam,
                 Limits:=strLimits,
                 ControlValueAnalog=>rActualPower);
```

It has to be stated that the control variable output in the above example is set to 0 for one cycle (if the error has been detected and if Enable:=FALSE). In state 7, the manual value is then used as control variable.

5.7 Touch Probe Function Blocks

5.7.1 Introduction and Overview

The function blocks MB_InitTouchProbe, MB_TouchProbe, MB_WriteExpectWindow and MB_TouchProbeContinuous control and manage the touch probe functionality supported by the Bosch Rexroth IndraDrive drives.

The touch probe functionality of the control hardware is provided via the following function blocks of the ML_PLCopen library:

Function block	Description
ML_InitTouchProbe	It enables and configures the touch probe in the control
ML_WriteExpectWindow	It configures the expectation window for the touch probe functionality
ML_TouchProbe	It enables the touch probe selected, analyzes the status and provides the measured values after the trigger event

Function block	Description
ML_TouchProbeContinuous	It enables the touch probe selected, analyzes the status and provides the measured values after the trigger event
ML_AbortTrigger	It cancels an active measurement

Fig.5-69: Function blocks for the touch probe functionality of the control hardware

The touch probe functionality of the Rexroth IndraDrive drives is provided via the following function blocks of the ML_TechBase library:

Function block	Description
MB_InitTouchProbe	It enables and configures the touch probe in the drive
MB_WriteExpectWindow	It configures the expectation window for the touch probe functionality
MB_TouchProbe	It enables the touch probe selected, analyzes the status and provides the measured values one time after the trigger event
MB_TouchProbeContinuous	It enables the touch probe selected, analyzes the status and provides the measured values continuously after the trigger event
MB_AbortTrigger	It cancels an active measurement

Fig.5-70: Function blocks for the touch probe functionality of the Rexroth IndraDrive

The following documentation describes the function blocks for the touch probe functionality of the Rexroth IndraDrive. The function blocks for the touch probe functionality of the control are described in the documentations "Rexroth IndraMotion MLC 12VRS Functional Description", "Rexroth IndraLogic XLC 12VRS Functional Description" and "Rexroth IndraMotion MLC IndraMotion IndraLogic XLC PLCopen Function Blocks and Data Types".

The IndraDrive touch probe functionality permits the detection of positions or of the internal system time in [us] for the instance of a fast digital input signal. The touch probe function block uses this drive functionality and provides a user interface. The following steps are required to use the touch probe function blocks:

5.7.2 Touch Probe Function Blocks - Components and Parameterization

Hardware

The following Bosch Rexroth hardware components are required:

- IndraMotion MLC/IndraLogic XLC
- IndraDrive C or IndraDrive M with advanced or basic performance

  The following controls are supported:

  – ADVANCED (type designation code: CSH01.XC-...)

  – BASIC SERCOS (single-axis; type designation code: CSB01.XN-SE-...)

– BASIC PROFIBUS (single-axis; type designation code: CSB01.XN-PB-...)
– BASIC UNIVERSAL (single-axis; type designation code: CSH01.XC-...)

Firmware

The following firmware is required and used with the above mentioned Bosch Rexroth hardware components:

- Control firmware 12VRS
- Drive firmware MPx 07VRS or higher
- A drive function package including the touch probe function

Parameterization of the Touch Probe Function Blocks IndraMotion MLC/IndraLogic XLC and IndraMotion MLD

The following drive parametrization steps are required in order to use the MB_InitTouchProbe, MB_TouchProbe and MB_TouchProbeContinuous technology function blocks:

IndraMotion MLC/IndraLogic XLC configuration

The following drive parameter settings are required when using the touch probe functionality in an IndraMotion MLC/IndraLogic XLC:

Open IndraWorks and go online with the relevant project.

- Include the configured touch probe control bits in the signal control word.
- **IndraWorks ▸ Project ▸ Motion ▸ Rea Axes ▸ Richt click on axis: Communication ▸ Signal control word**
  – Touch probe 1: S-0-0405, Bit 0
  – Touch probe 2: S-0-0406, Bit 0

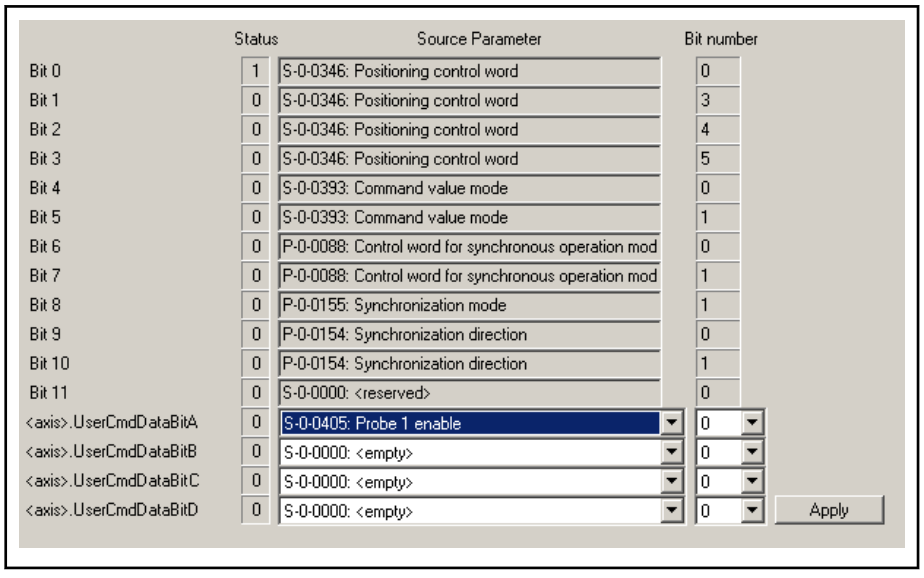


Fig.5-71: Signal control word

- Include the configured touch probe status bits in the signal status word.

  IndraWorks ▸ Project ▸ Motion ▸ Rea Axes ▸ Richt click on axis: Communication ▸ Signal status word
  – Touch probe 1, positively detected: S-0-0409, Bit 0

– Touch probe 1, negatively detected: S-0-0410, Bit 0
– Touch probe 2, positively detected: S-0-0411, Bit 0
– Touch probe 2, negatively detected: S-0-0412, Bit 0

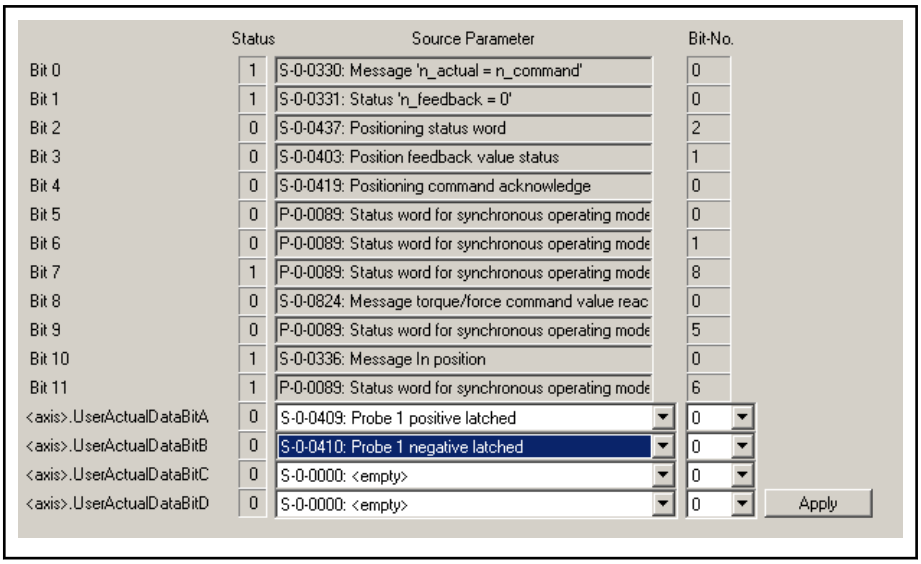


Fig.5-72: Signal status word

- Include the configured touch probe values in the optional cyclic AT telegram.

  IndraWorks ▸ Project ▸ Motion ▸ Rea Axes ▸ Richt click on axis: Communication ▸ Cyclic SERCOS Channel

  – Touch probe 1, positive edge: S-0-0130
  – Touch probe 1, negative edge: S-0-0131
  – Touch probe 2, positive edge: S-0-0132
  – Touch probe 2, negative edge: S-0-0133
  – Touch probe 1, number of marker failures: P-0-0224
  – Touch probe 2, number of marker failures: P-0-0225
  – Touch probe 1, difference value of the touch probe: P-0-0202
  – Touch probe 2, difference value of the touch probe: P-0-0203

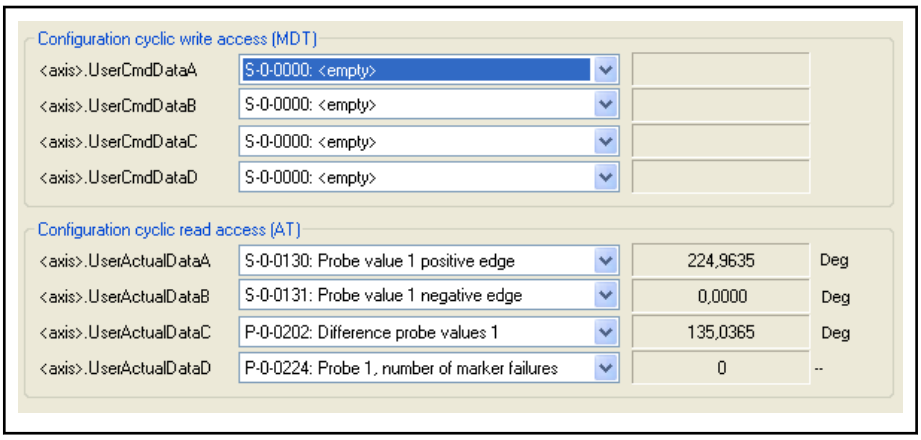


Fig.5-73: Configuring a cyclic AT telegram

Configuring the IndraMotion MLD-S or IndraMotion MLD-M (master axis)

The following drive parameter settings are required to use the touch probe functionality in an IndraMotion MLD-S or IndraMotion MLD-M as master axis:

- Include the configured touch probe control bits in the signal control word.

  IndraWorks ▸ Project ▸ MLD ▸ Right-click on: AxisData

  – Touch probe 1: S-0-0405, Bit 0

  – Touch probe 2: S-0-0406, Bit 0

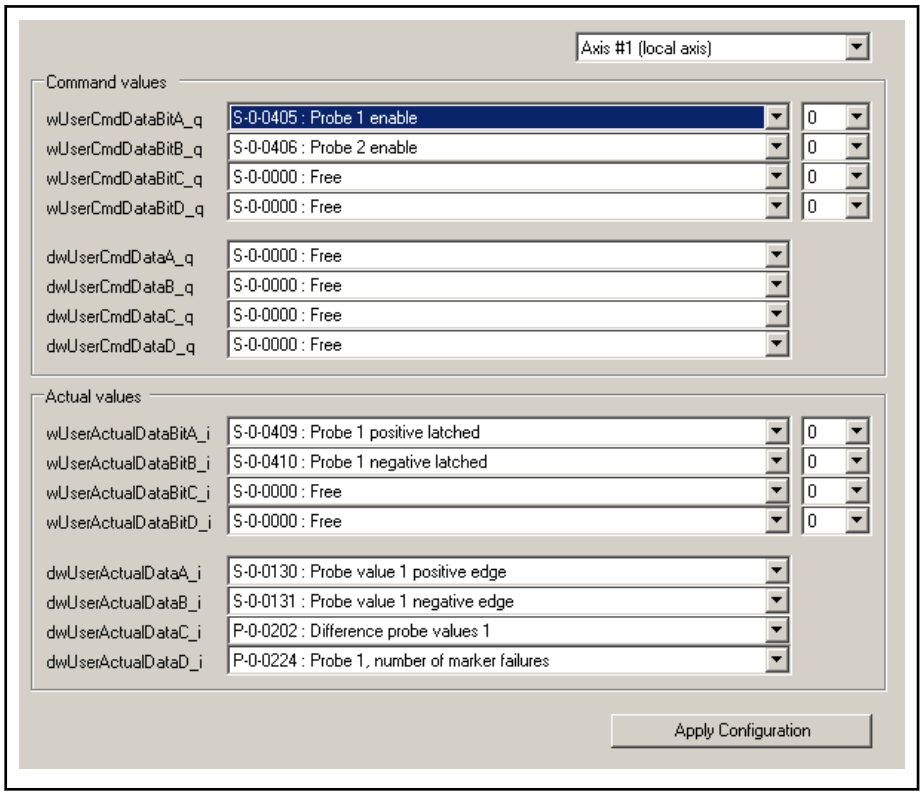


Fig.5-74: Command value data bits

- Include the configured touch probe status bits in the signal status word.

  IndraWorks ▸ Right click on SERCOS III ▸ Cross Communication Drives ▸ Signal status word

  – Touch probe 1, positively detected: S-0-0409, Bit 0

  – Touch probe 1, negatively detected: S-0-0410, Bit 0

  – Touch probe 2, positively detected: S-0-0411, Bit 0

  – Touch probe 2, negatively detected: S-0-0412, Bit 0

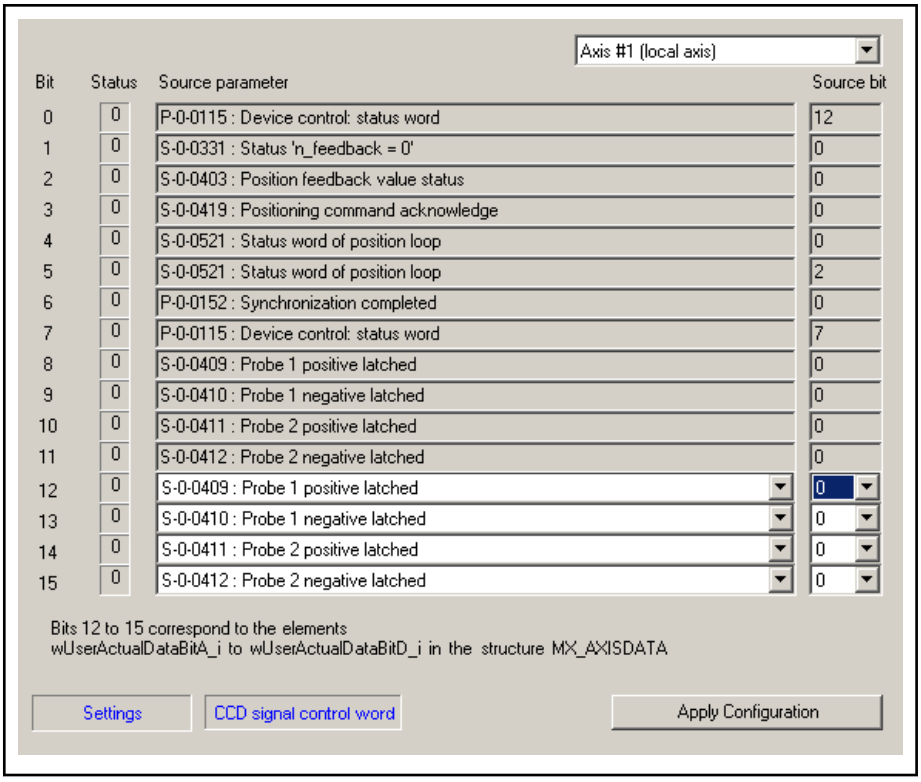


Fig.5-75: Actual value data bits

- Include the configured touch probe values in the optional cyclic AT telegram.

  IndraWorks ▸ Project ▸ MLD ▸ Right-click on: AxisData

  - Touch probe 1, positive edge: S-0-0130
  - Touch probe 1, negative edge: S-0-0131
  - Touch probe 2, positive edge: S-0-0132
  - Touch probe 2, negative edge: S-0-0133
  - Touch probe 1, number of marker failures: P-0-0224
  - Touch probe 2, number of marker failures: P-0-0225
  - Touch probe 1, difference value of the touch probe: P-0-0202
  - Touch probe 2, difference value of the touch probe: P-0-0203
  - Include the following parameters in the actual values data container for each configured, measured touch probe signal:
    - TP_ACT_POS_FEEDBACK1 => No additional data is required
    - TP_ACT_POS_FEEDBACK2 => No additional data is required
    - TP_ACT_POSIN_ACT_CYCLE => No additional data is required
    - TP_SYNC_CMD_POS => P-0-0779
    - TP_PROBE_TIME => No additional data is required
    - TP_ACT_MEASURING_ENC_POS => P-0-0332
    - TP_RES_MASTER_AXIS_POS => P-0-0764

– TP_EFF_MASTER_AXIS_POS => P-0-0777
– TP_CAM_SHAFT_PROF_ACCESS_ANGLE => P-0-0777

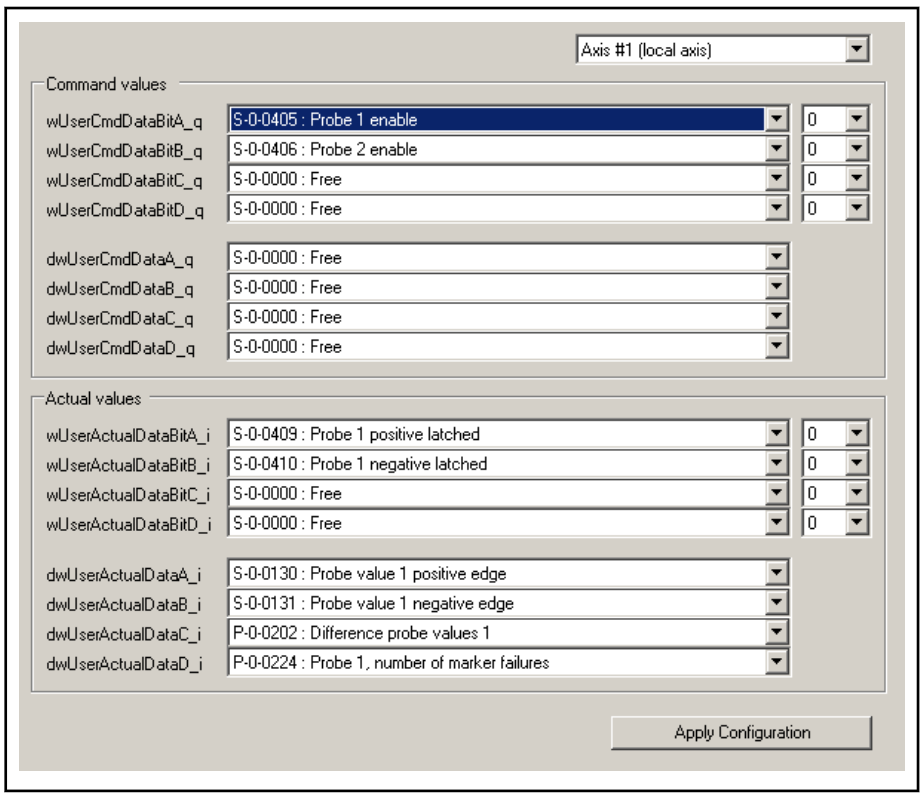


Fig.5-76: Command value data bits

- Set P-0-1367, Bit 6 (enable update of the axis data structure used, only IndraMotion MLD-M)

Configuring IndraMotion MLD-M (slave axis)

The following drive parameter settings are required when using the touch probe functionality in an IndraMotion MLD-M as slave axis:

- Include the configured touch probe control bits in the signal control word.

  IndraWorks ▸ Right click on SERCOS III ▸ Cross Communication Drives ▸ Signal control word

  – Touch probe 1: S-0-0405, Bit 0
  – Touch probe 2: S-0-0406, Bit 0

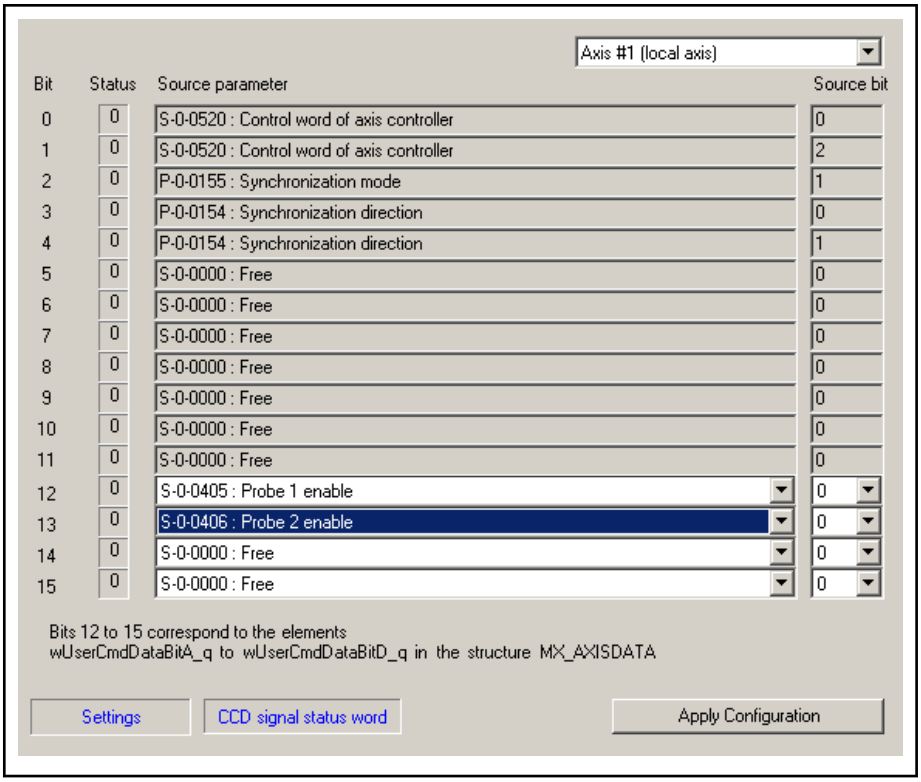


Fig.5-77: Signal control word

- Include the configured touch probe status bits in the signal status word.

  IndraWorks ▸ Right click on SERCOS III ▸ Cross Communication Drives ▸ Signal status word

  – Touch probe 1, positively detected: S-0-0409, Bit 0
  – Touch probe 1, negatively detected: S-0-0410, Bit 0
  – Touch probe 2, positively detected: S-0-0411, Bit 0
  – Touch probe 2, negatively detected: S-0-0412, Bit 0

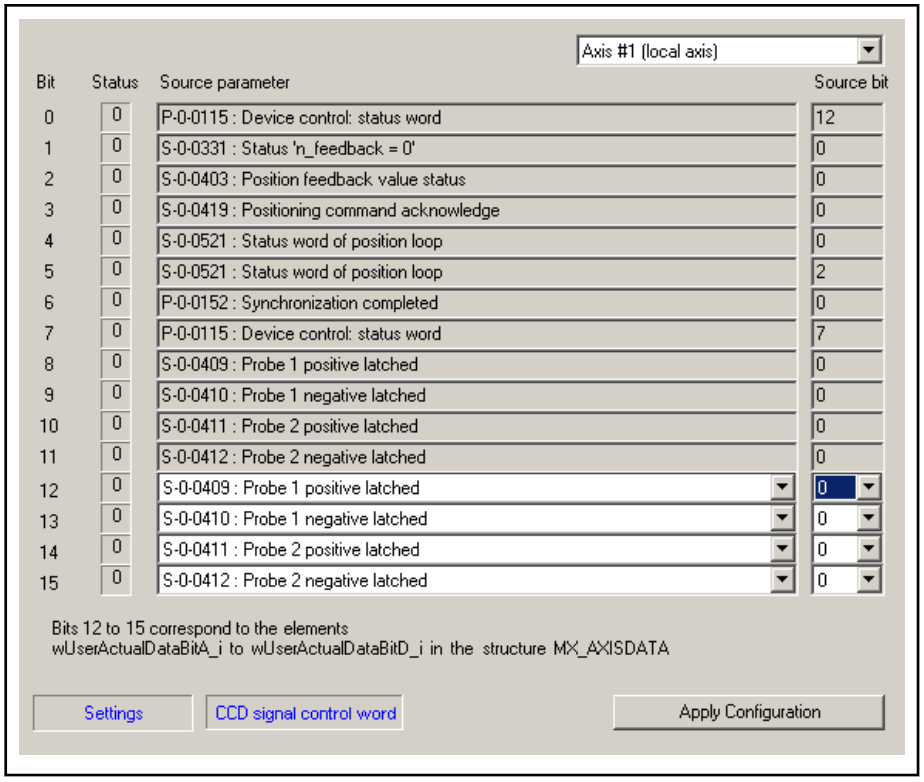


Fig.5-78: Signal status word

- Include the configured touch probe values in the optional cyclic AT telegram.

  IndraWorks ▸ Right click on SERCOS III ▸ Cross Communication Drives ▸ Cyclic SERCOS channel

  – Touch probe 1, positive edge: S-0-0130
  – Touch probe 1, negative edge: S-0-0131
  – Touch probe 2, positive edge: S-0-0132
  – Touch probe 2, negative edge: S-0-0133
  – Touch probe 1, number of marker failures: P-0-0224
  – Touch probe 2, number of marker failures: P-0-0225
  – Touch probe 1, difference value of the touch probe: P-0-0202
  – Touch probe 2, difference value of the touch probe: P-0-0203
  – Include the following parameters in the optional cyclic AT for each configured, measured touch probe signal:
    – TP_ACT_POS_FEEDBACK1 => No additional data is required
    – TP_ACT_POS_FEEDBACK2 => No additional data is required
    – TP_ACT_POSIN_ACT_CYCLE => No additional data is required
    – TP_SYNC_CMD_POS => P-0-0779
    – TP_PROBE_TIME => No additional data is required
    – TP_ACT_MEASURING_ENC_POS => P-0-0332
    – TP_RES_MASTER_AXIS_POS => P-0-0764

– TP_EFF_MASTER_AXIS_POS => P-0-0777
– TP_CAM_SHAFT_PROF_ACCESS_ANGLE => P-0-0777

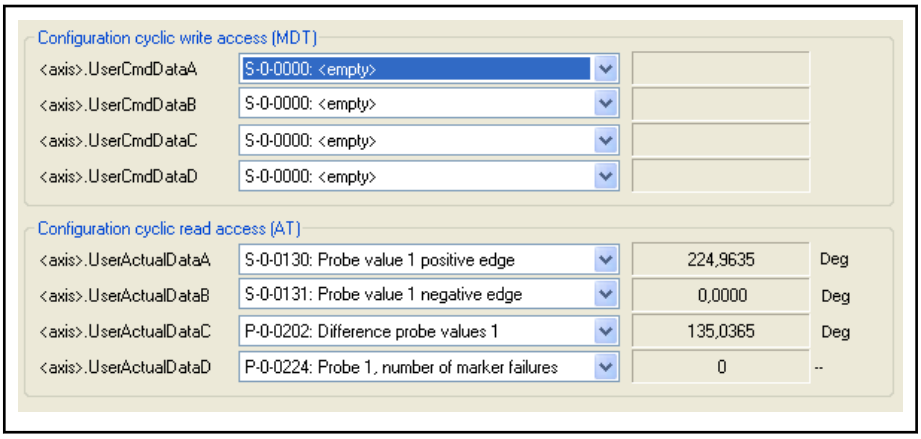


Fig.5-79: Configuring a cyclic AT telegram

- Set P-0-1367, Bit 6 (enable update of the axis data structure used, only IndraMotion MLD-M)

Configuration: Touch probe 1

Touch probe 1	AT container	Control bit	Status bit
Positive edge	S-0-0130	S-0-0405.0	S-0-0409.0
Negative edge	S-0-0131	S-0-0405.0	S-0-0410.0
Positive and negative edge, difference, marker failure	S-0-0130,S-0-0131, P-0-0224 and P-0-0202	S-0-0405.0	S-0-0409.0, S-0-0410.0

Fig.5-80: Configuration: Touch probe 1

Configuration: Touch probe 2

Touch probe 2	AT container	Control bit	Status bit
Positive edge	S-0-0132	S-0-0406.0	S-0-0411.0
Negative edge	S-0-0133	S-0-0406.0	S-0-0412.0
Positive and negative edge, difference, marker failure	S-0-0132, S-0-0133, P-0-0225 AND P-0-0203	S-0-0406.0	S-0-0411.0, S-0-0412.0

Fig.5-81: Configuration: Touch probe 2

Example configuration of the signal status word, signal control word and cyclic SERCOS channel

- IndraDrive touch probe 1
- Negative edge
- Expectation window disabled
- Marker failure disabled
- Difference measurement disabled

1. Configuring a signal control word

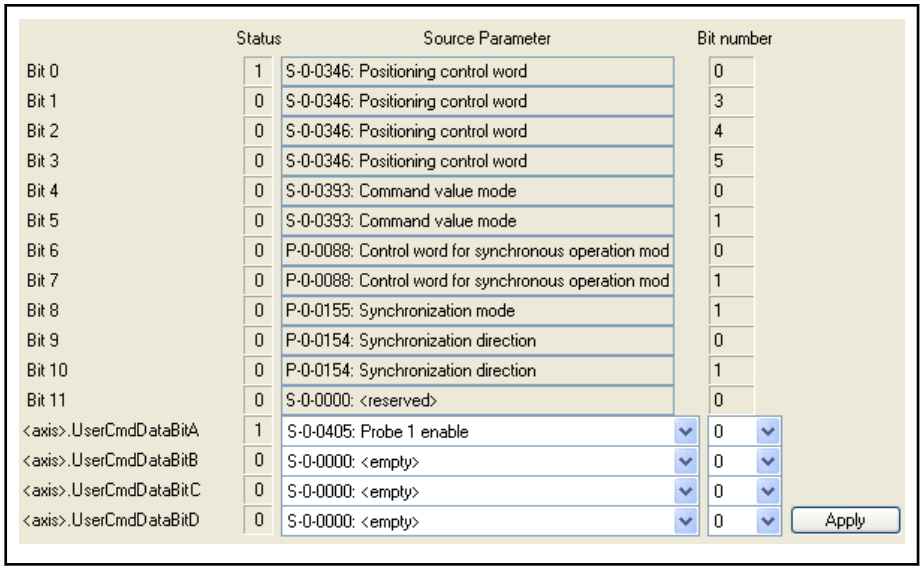


Fig.5-82: Example of a signal control word

2. Configuring a signal status word

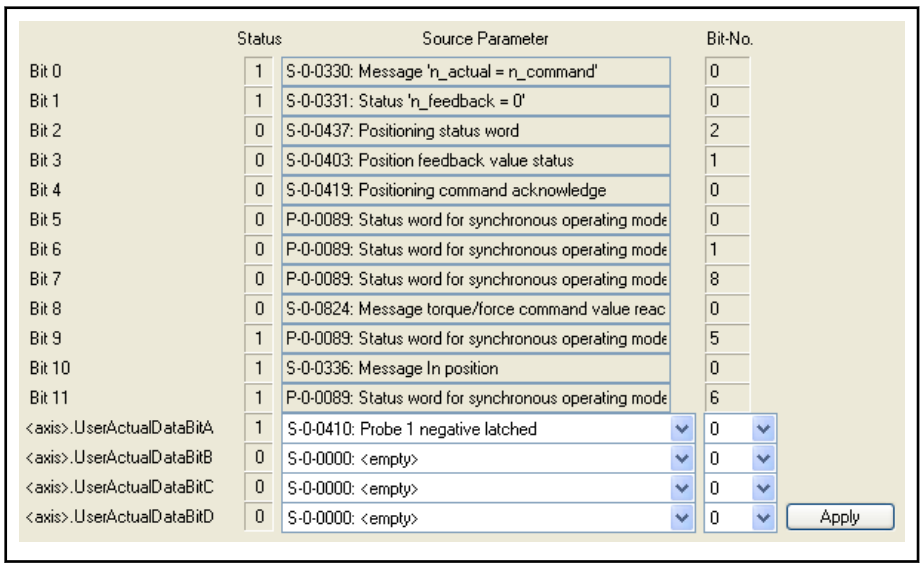


Fig.5-83: Example of a signal status word

3. Configuration of a cyclic SERCOS channel

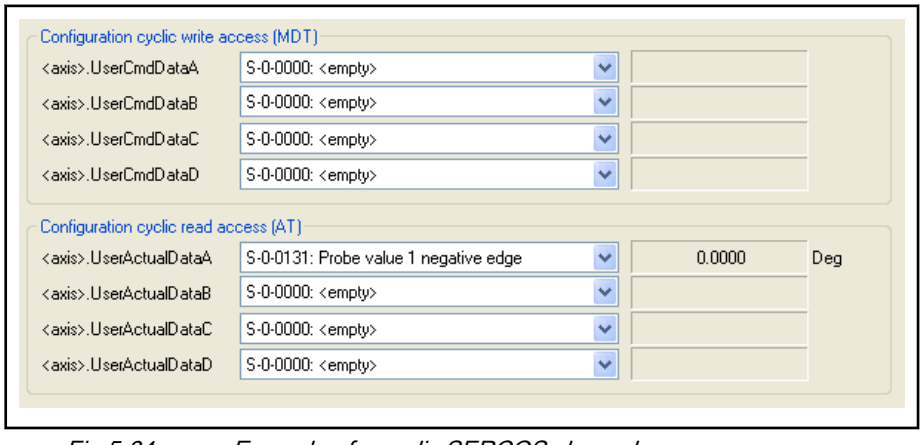


Fig.5-84: Example of a cyclic SERCOS channel

ML_TechBase.library

5.7.3 MB_InitTouchProbe

Brief Description

The The MB_InitTouchProbe function block initializes the touch probe functionality.

Interface Description

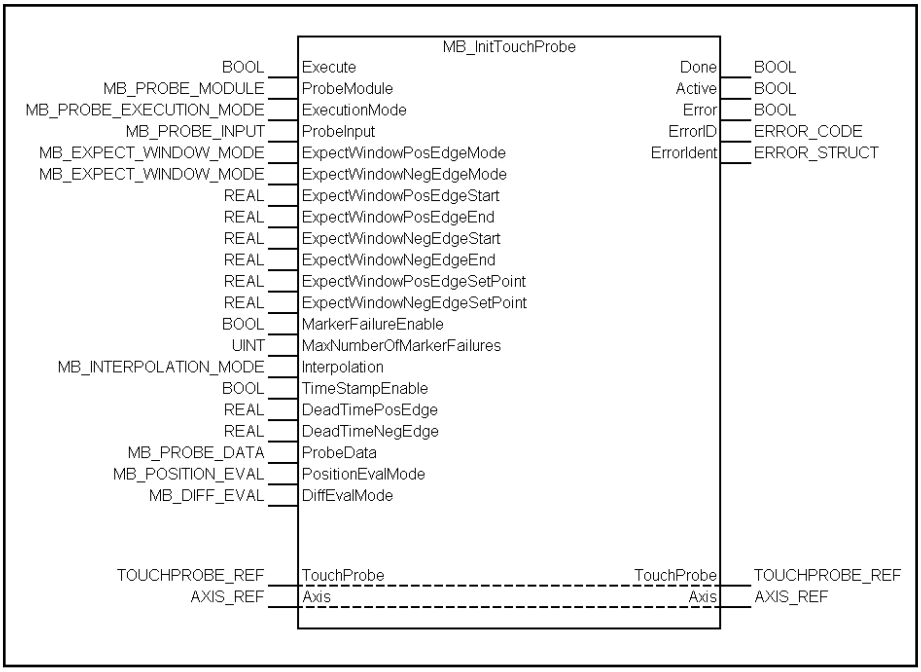


Fig.5-85: MB_InitTouchProbe function block

I/O type	Name	Data type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis
	TouchProbe	TOUCHPROBE_REF	Reference to the signal trigger source
VAR_INPUT	Execute	BOOL	It starts the initialization of the touch probe functionality at a rising edge
	ProbeModule	MB_PROBE_MODULE	Position of the touch probe on the MLC (onboard, function module)
	ExecutionMode	MB_EXECUTION_MODE	Selection if single-shot or continuous measurement
	ProbeInput	MB_PROBE_INPUT	It selects the digital Input for the MLC touch probe
	ExpectWindowPosEdgeMode	MB_EXPECT_WINDOW_MODE	Mode of expectation window if the positive edge is evaluated
	ExpectWindowNegEdgeMode	MB_EXPECT_WINDOW_MODE	Mode of expectation window if the negative edge is evaluated
	ExpectWindowPosEdgeStart	REAL	Start of expectation window if the positive edge is evaluated
	ExpectWindowPosEdgeEnd	REAL	End of expectation window if the positive edge is evaluated
	ExpectWindowNegEdgeStart	REAL	Start of expectation window if the negative edge is evaluated
	ExpectWindowNegEdgeEnd	REAL	End of expectation window if the negative edge is evaluated

I/O type	Name	Data type	Comment
	ExpectWindowPosEdgeSetPoint	REAL	Command value for relative expectation window if the positive edge is evaluated
	ExpectWindowNegEdgeSetPoint	REAL	Command value for relative expectation window if the negative edge is evaluated
	MarkerFailureEnable	BOOL	Enable marker failure
	MaxNumberOfMarkerFailure	UINT	Threshold for the registered number of marker failure
	Interpolation	MB_INTERPOLATION_MODE	Interpolation for the MLC touch probe
	TimeStampEnable	BOOL	Enabling time stamp for MLC touch probe
	DeadTimePosEdge	REAL	Dead time compensation for positive edge if evaluated
	DeadTimeNegEdge	REAL	Dead time compensation for negative edge if evaluated
	ProbeData	MB_PROBE_DATA	Measured touch probe signal
	PositionEvalMode	MB_POSITION_EVAL	It selects the edge to be evaluated
	DiffEvalMode	MB_DIFF_EVAL	Edge selection for difference measurement
VAR_OUTPUT	Done	BOOL	Initialization done
	Active	BOOL	The function block is currently active.
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

Fig.5-86: Interface variables of the MB_InitTouchProbe function block

Min./max. and default values

The following table lists the min./max. and default values of the function block input.

Name	Type	Min. value	Max. value	Default values	Effective
Execute	BOOL			FALSE	Continuous
ProbeModule	MB_PROBE_MODULE			ONBOARD_MODULE	Rising edge at "Execute"
ExecutionMode	MB_PROBE_EXECUTION_MODE			CONTINUOUS	Rising edge at "Execute"
ProbeInput	MB_PROBE_INPUT			D_INPUT_1	Rising edge at "Execute"
ExpectWindowPosEdgeMode	MB_EXPECT_WINDOW_MODE			DISABLED_EXP	Rising edge at "Execute"
ExpectWindowNegEdgeMode	MB_EXPECT_WINDOW_MODE			DISABLED_EXP	Rising edge at "Execute"
ExpectWindowPosEdgeStart	REAL	0.0	ModuloValue	0.0	Rising edge at "Execute"

ML_TechBase.library

Name	Type	Min. value	Max. value	Default values	Effective
ExpectWindow-PosEdgeEnd	REAL	0.0	ModuloValue	0.0	Rising edge at "Execute"
ExpectWindow-NegEdgeStart	REAL	0.0	ModuloValue	0.0	Rising edge at "Execute"
ExpectWindow-NegEdgeEnd	REAL	0.0	ModuloValue	0.0	Rising edge at "Execute"
ExpectWindow-PosEdgeSetPoint	REAL	0.0	ModuloValue	0.0	Rising edge at "Execute"
ExpectWindow-NegEdgeSetPoint	REAL	0.0	ModuloValue	0.0	Rising edge at "Execute"
MarkerFailureEnable	BOOL			FALSE	Rising edge at "Execute"
MaxNumberOfMarkerFailure	UINT	0		0	Rising edge at "Execute"
TimeStampEnable	BOOL			FALSE	Rising edge at "Execute"
DeadTimePosEdge	REAL	0.0		0.0	Rising edge at "Execute"
DeadTimeNegEdge	REAL	0.0		0.0	Rising edge at "Execute"
ProbeData	MB_PROBE_DATA			TP_ACT_POS_FEEDBACK	Rising edge at "Execute"
PositionEvalMode	MB_POSITION_EVAL			DISABLED_EVAL	Rising edge at "Execute"
DiffEvalMode	MB_DIFF_EVAL			DISABLED_DIFF	Rising edge at "Execute"

Fig.5-87: Min./max. and default values for the MB_TouchProbeContinuous inputs

Functional description of the MB_InitTouchProbe

The inputs defined in this section must follow the data types described in chapter "MB_PROBE_EXECUTION_MODE" on page 115.

The MB_InitTouchProbe function block must be executed in SERCOS phase 4 before the MB_TouchProbe and MB_TouchProbeContinuous function blocks can be used.

TouchProbe, ProbeModule, ProbeInput

The "TouchProbe" input defines the control and touch probes used. The "CntrlNo" element has to be set to "LOCAL_CNTRL". The "TouchProbeNo" element can either assume the value "TOUCHPROBE_DRIVE_1" or "TOUCHPROBE_DRIVE_2".

The inputs "ProbeModule" and "ProbeInput" are currently not evaluated.

ExecutionMode

At the "ExecutionMode" input, the touch probe function for single-shot measurement or continuous measurement is set. If single-shot measurement is

used, the MB_TouchProbe function block must be used. For continuous measurement, the MB_TouchProbeContinuous function block is used.

ExpectWindow, PositionEvalMode

"ExpectWindow" describes the expectation window of the touch probe. Currently, the expectation window of the drive does not support a separate expectation window for the negative and positive edges at the same time. If only the positive edge is evaluated ("PositionEvalMode"=POSITION_POS_EDGE), the inputs "ExpectWindowPosEdgeMode", "ExpectWindowPosEdgeStart","ExpectWindowPosEdgeStart" and "ExpectWindowPosEdgeSetPoint" are evaluated.

If only the negative edge is evaluated ("PositionEvalMode"=POSITION_NEG_EDGE), "ExpectWindowNegEdgeMode", "ExpectWindowNegEdgeStart","ExpectWindowNegEdgeStart" and "ExpectWindowNegEdgeSetPoint" are evaluated.

If both edges are evaluated ("PositionEvalMode"=POSITION_POS_NEG_EDGE), only the input "ExpectWindowPosEdgeMode", "ExpectWindowPosEdgeStart","ExpectWindowPosEdgeEnd", "ExpectWindowPosEdgeSetPoint" for the expectation window of the drive are evaluated. The expectation window applies to both evaluated edges.

DeadTimePosEdge, DeadTimeNegEdge

The inputs "DeadTimePosEdge" and "DeadTimeNegEdge" determine the dead-time compensation of the edges. The "DeadTimePosEdge" input is valid if the positive edge is evaluated. The "DeadTimeNegEdge" input is valid if the negative edge is evaluated.

ProbeData, DiffValMode

The "ProbeData" input defines the measured signal of the touch probe function. The following signals are available for selection:

Signal	Description
TP_ACT_POS_FEEDBACK1	Measured value, positive edge
TP_ACT_POS_FEEDBACK2	Measured value, negative edge
TP_SYNC_CMD_POS	Synchronous position command value
TP_PROBE_TIME	Time stamp
TP_ACT_MEASURING_ENC_POS	Actual position value of the measuring encoder
TP_RES_MASTER_AXIS_POS	Resulting master axis position
TP_EFF_MASTER_AXIS_POS	Effective master axis position
TP_CAM_SHAFT_PROF_ACCESS_ANGLE	Access angle of the cam table

Fig.5-88: Possible values of the "ProbeData" input

The "DiffValMode" input selects the difference measurement of the touch probe function. Only "STANDARD_MODE" is currently supported.

ModuloValue

The "ModuloValue" output displays the modulo value of the measured signal.

Interpolation

The "Interpolation" indicates how the position values should be interpolated. The MB_InitTouchProbe function block currently only supports the values

"NO_INTERPOLATION" and "LINEAR_INTERPOLATION". It is not evaluated further.

Time diagram

Time diagram according to the PLCopen specification.

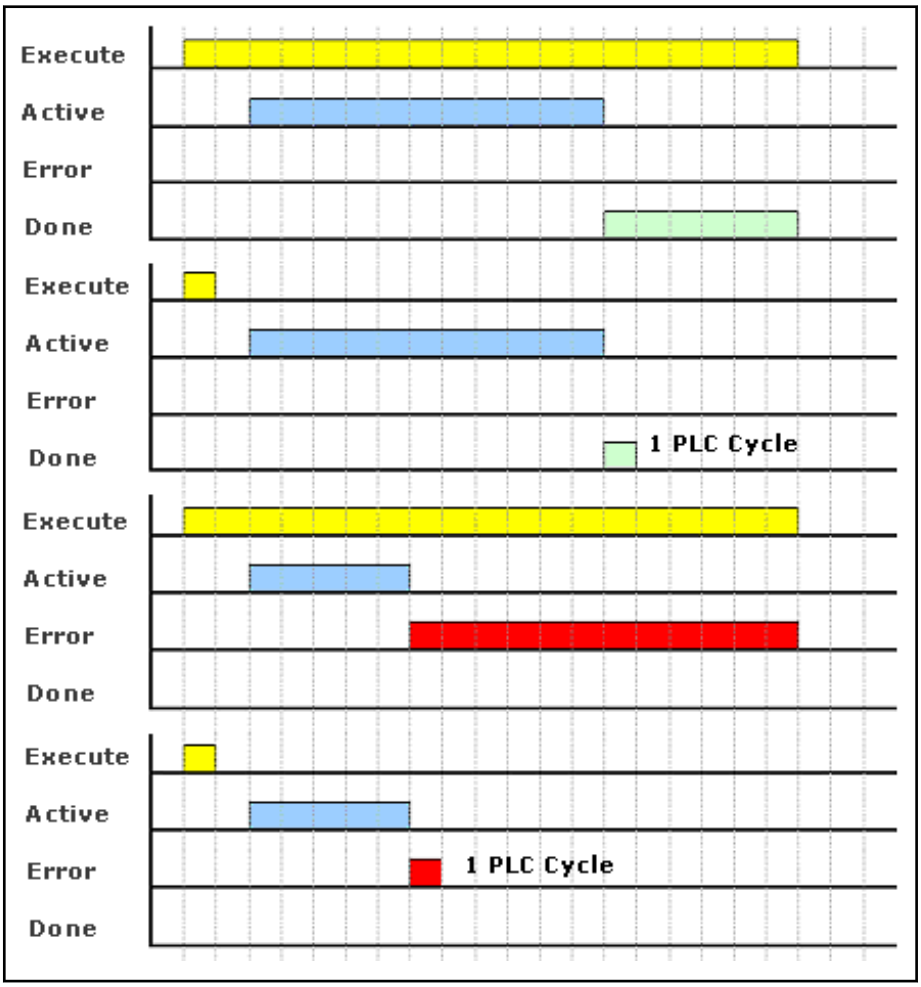


Fig.5-89: Time diagram of the MB_InitTouchProbe function block

Error Handling

The function block generates the following error messages in Additional1/Additional2 for the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"AxisNo" for "Axis" is not within the valid range
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis (Axis_Ref) was changed while the function block was in operation
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0001	The "ProbeModule" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0002	The "ExecutionMode" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0003	The "ProbeInput" input is invalid

ML_TechBase.library

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0004	The"DeadTimePosEdge" input is invalid (check if < 0)
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0005	The"DeadTimeNegEdge" input is invalid (check if < 0)
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0006	The "ProbeData" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0007	The "PositionEvalMode" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0008	The "DiffEvalMode" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0404	16#0009	The"MaxNumberOfMarkerFailure" input is invalid (check if < 0)
INPUT_INVALID_ERROR(16#0001)	16#0405	16#0001	The "DelExpectWindowPosEdge-Modeay" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0405	16#0002	The "ExpectWindowNegEdge-Mode" input is invalid
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0003	The "ExpectWindowPosEdgeStart" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0004	The "ExpectWindowPosEdgeEnd" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0005	The "ExpectWindowNegEdgeStart" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0006	The "ExpectWindoNegEdgeEnd" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0007	The "ExpectWindowPosEdgeSet-Point" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0008	The "ExpectWindowNegEdgeSet-Point" input is not within the valid range
RESOURCE_ERROR (16#0003)	16#0403	16#0001	Required touch probe control bit (S-0-0405, Bit 0) is not configured in the signal control word
RESOURCE_ERROR (16#0003)	16#0403	16#0002	Required touch probe control bit (S-0-0406, Bit 0) is not configured in the signal control word
RESOURCE_ERROR (16#0003)	16#0403	16#0003	Required touch probe control bit (S-0-0405, Bit 0) is not configured in signal status word
RESOURCE_ERROR (16#0003)	16#0403	16#0004	Required touch probe control bit (S-0-0410, Bit 0) is not configured in the signal status word
RESOURCE_ERROR (16#0003)	16#0403	16#0005	Required touch probe control bit (S-0-0411, Bit 0) is not configured in the signal status word

ML_TechBase.library

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0403	16#0006	Required touch probe control bit (S-0-0412, Bit 0) is not configured in the signal status word
RESOURCE_ERROR (16#0003)	16#0403	16#0007	Required touch probe value S-0-0130 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#0008	Required touch probe value S-0-0131 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#0009	Required touch probe value S-0-0132 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#000A	Required touch probe value S-0-0133 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#000B	Required touch probe parameter P-0-0224 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#000C	Required touch probe parameter P-0-0225 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#000D	Required touch probe parameter P-0-0202 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#000E	Required touch probe parameter P-0-0203 is not cyclically configured in the AT
RESOURCE_ERROR (16#0003)	16#0403	16#00XX	P-0-1367, Bit 6 of the MLD-M master axis is not set to TRUE. This is required for the update of the AxisData structure

Fig.5-90: Error codes of the MB_InitTouchProbe function block

Parameterized parameters

The MB_InitTouchProbe function block parameterizes the following parameters:

- P-0-0226 touch probe, extended control word
- S-0-0169, touch probe control parameter
- S-0-0170, Command touch probe cycle
- S-0-0426, Signal selection, touch probe 1
- S-0-0427, Signal selection, touch probe 2

5.7.4 MB_TouchProbe

Brief Description

The MB_TouchProbe function block is used to record an axis position, a master position or time in case of an event trigger. This is a function block for the single-shot measurement. The function block has to be executed again to detect another signal. The signal to be measured is defined at the "TouchProbe" input of the MB_InitTouchProbe function block.

Interface Description

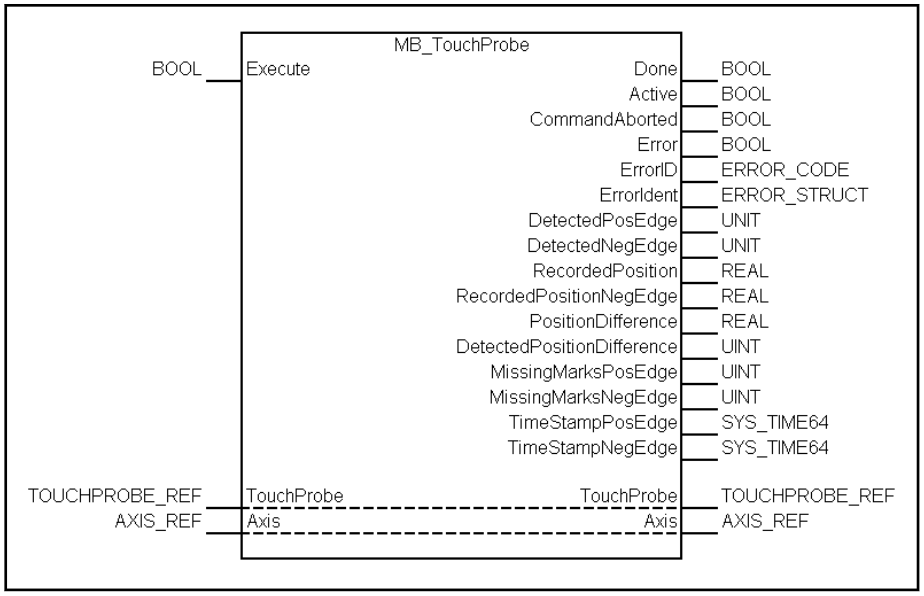


Fig.5-91: MB_TouchProbe function block

I/O type	Name	Data type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis
	TouchProbe	TOUCHPROBE_REF	Reference to the signal trigger source
VAR_INPUT	Execute	BOOL	It starts the touch probe recording at a rising edge
VAR_OUTPUT	Done	BOOL	Trigger event recorded
	Active	BOOL	The function block is currently active.
	CommandAborted	BOOL	Command is aborted by MB_AbortTrigger
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	DetectedPosEdge	UINT	Positive edge detected
	DetectedNegEdge	UINT	Negative edge detected
	RecordedPositionPosEdge	REAL	Position at which the positive edge occurred (in technical units)
	RecordedPositionNegEdge	REAL	Position at which the negative edge occurred (in technical units)
	PositionDifference	REAL	It indicates the difference between the touch probe values depending on "DiffEvalMode"
	DetectedPositionDifference	UINT	Difference measurement detected
	MissingMarksPosEdge	UINT	Counter of missing marks if positive edge is evaluated
	MissingMarksNegEdge	UINT	Counter of missing marks if negative edge is evaluated

ML_TechBase.library

I/O type	Name	Data type	Comment
	TimeStampPosEdge	SYS_TIME64	Point in time when the positive edge occurred (MLC/ XLC touch probe only)
	TimeStampNegEdge	SYS_TIME64	Point in time when the negative edge occurred (MLC/ XLC touch probe only)

Fig.5-92: Interface variables of the MB_TouchProbe function block

Default value

The following table lists the default value of the function block input.

Name	Type	Min. value	Max. value	Default values	Effective
Enable	BOOL	n/a	n/a	FALSE	Continuous

Fig.5-93: Default value for the MB_TouchProbe input

Functional description of the MB_TouchProbe

The MB_TouchProbe function block executed a single-shot measurement. The function block is enabled at a rising edge at the "Execute" input.

DetectedPosEdge, DetectedNegEdge

If the touch probe function is set to evaluate a positive/negative edge of the input signal, the "DetectedPosEdge"/"DetectedNegEdge" output increases when a positive edge (negative edge) is detected.

RecordedPositionPosEdge, RecordedPositionNegEdge

The outputs "RecordedPositionPosEdge" and "RecordedPositionNegEdge" contain the position at which the positive or negative edge occurred.

PositionDifference, DetectedPositionDifference

The "PositionDifference" output contains the difference of the two latest measured values of the same touch probe in case of an activated difference measurement. At each measurement, the "DetectedPositionDifference" output is increased.

The following calculation applies if the drive touch probe function is activated for the "PositionDifference" output:

$$\text{PositionDifference} = |(\text{S-0-0130}) - (\text{S-0-0131})|$$

$$\text{PositionDifference} = |(\text{S-0-0132}) - (\text{S-0-0133})|$$

Fig.5-94: Calculating the position difference for axes

MissingMarksPosEdge, MissingMarksNegEdge

The "MissingMarksPosEdge" or "MissingMarksNegEdge" outputs are only evaluated if the expectation window is active and marker failure was enabled. They count the marks outside the expectation window. The outputs are set to 0 every time a marker is detected within the expectation window. If the touch probe is set to evaluate both edges, only the "MissingMarksPosEdge" output is valid. Currently, the drive touch probe function does not support individual marker failures for each edge at the same time.

Time diagram

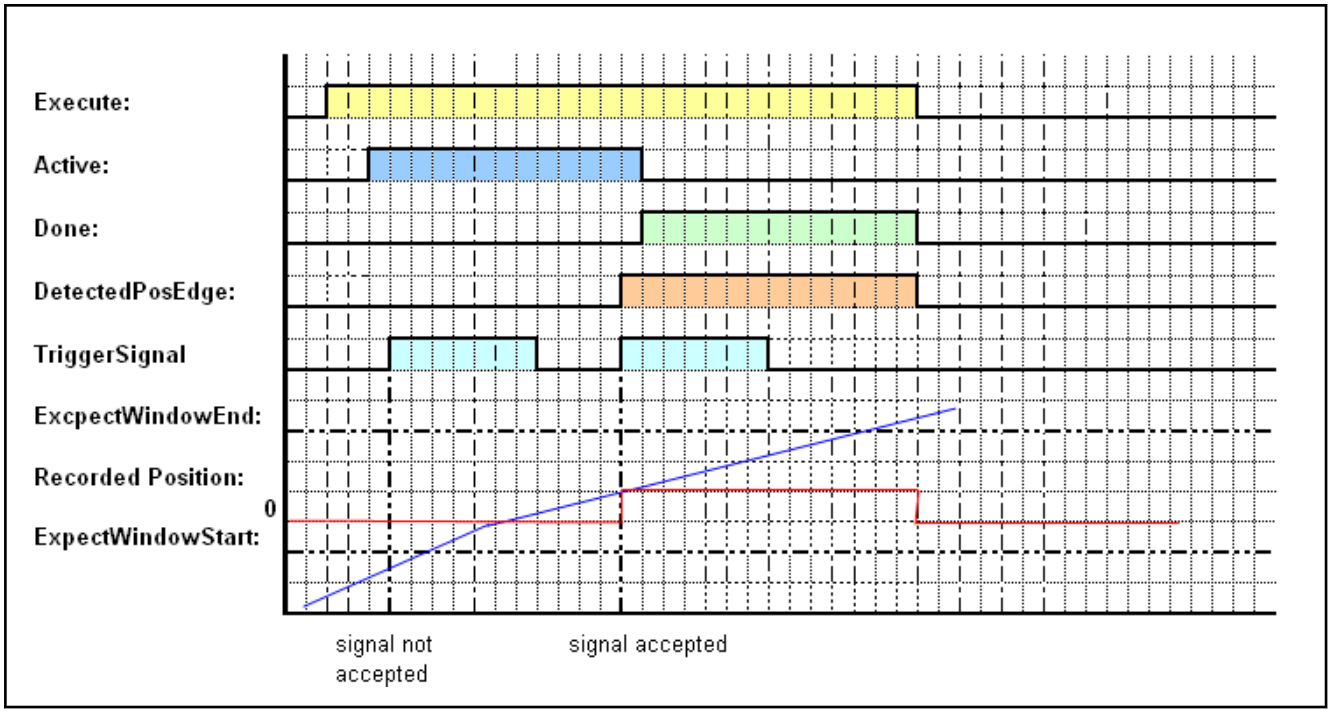


Fig.5-95: Time diagram of the MB_TouchProbe function block

Error Handling The function block generates the following error messages in Additional1/Additional2 for the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	AxisNo of "Axis" is not within the valid range
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis (Axis_Ref) was changed while the function block was in operation
ACCESS_ERROR (16#0003)	16#0406	16#0001	Empty handle for ATPosEdgeVal
ACCESS_ERROR (16#0003)	16#0406	16#0002	Empty handle for ATNegEdgeVal
ACCESS_ERROR (16#0003)	16#0406	16#0003	Empty handle for ATMarkerFailure
ACCESS_ERROR (16#0003)	16#0406	16#0004	Empty handle for ATDiffProbeVal
ACCESS_ERROR (16#0003)	16#0406	16#0005	Empty handle for ATDeadTime-Comp
ACCESS_ERROR (16#0003)	16#0406	16#0006	Empty handle for SCWEnableP-robe
ACCESS_ERROR (16#0003)	16#0406	16#0007	Empty handle for SSWPosLatched
ACCESS_ERROR (16#0003)	16#0406	16#0008	Empty handle for SSWNegLatched

Fig.5-96: Error codes of the MB_TouchProbe function block

ML_TechBase.library

5.7.5 MB_TouchProbeContinuous

Brief Description

The MB_TouchProbeContinuous function block is used to record an axis position, a master position or time in case of an event trigger. It executes a continuous measurement as long as the "Enable" input is TRUE. The signal to be detected is defined at the "TouchProbe" input of the MB_InitTouchProbe function block.

Interface Description

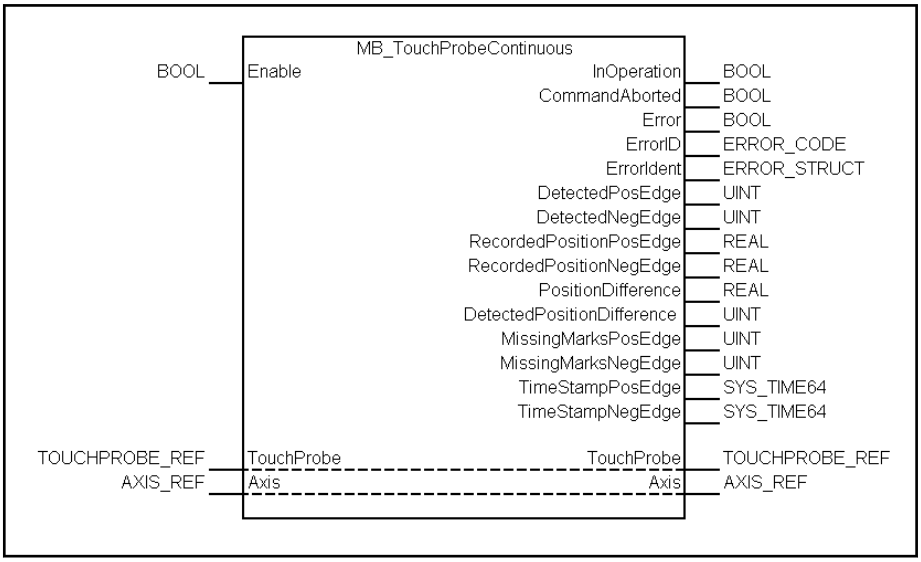


Fig.5-97: MB_TouchProbeContinuous function block

I/O type	Name	Data type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis
	TouchProbe	TOUCHPROBE_REF	Reference to the signal trigger source
VAR_INPUT	Enable	BOOL	It starts the touch probe recording at TRUE
VAR_OUTPUT	InOperation	BOOL	Touch probe in operation
	CommandAborted	BOOL	Command is aborted by MB_AbortTrigger
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	DetectedPosEdge	UINT	Positive edge detected
	DetectedNegEdge	UINT	Negative edge detected
	RecordedPositionPosEdge	REAL	Position at which the positive edge occurred (in technical units)
	RecordedPositionNegEdge	REAL	Position at which the negative edge occurred (in technical units)
	PositionDifference	REAL	It indicates the absolute value of the difference between touch probe values
	DetectedPositionDifference	UINT	Difference measurement detected
	MissingMarksPosEdge	UINT	Counter of missing marks for the positive edge if evaluated

I/O type	Name	Data type	Comment
	MissingMarksNegEdge	UINT	Counter of missing marks for the negative edge if evaluated
	TimeStampPosEdge	MB_TIME_STAMP	Point in time when the positive edge occurred (MLC/ XLC touch probe only)
	TimeStampNegEdge	MB_TIME_STAMP	Point in time when the negative edge occurred (MLC/ XLC touch probe only)

Fig.5-98: Interface variables of the MB_TouchProbeContinuous function block

Default value

The following table lists the default value of the function block input.

Name	Type	Min. value	Max. value	Default values	Effective
Enable	BOOL			FALSE	Continuous

Fig.5-99: Default value for the MB_InitTouchProbe input

Functional description of the MB_TouchProbeContinuous

The MB_TouchProbeContinuous function block is exclusively used for continuous measurement. The function block is enabled as long as the "Enable" input is TRUE.

DetectedPosEdge, DetectedNegEdge

If the touch probe function is set to evaluate a positive/negative edge of the input signal, the "DetectedPosEdge"/"DetectedNegEdge" output increases when a positive edge (negative edge) is detected.

RecordedPositionPosEdge, RecordedPositionNegEdge

The outputs "RecordedPositionPosEdge" and "RecordedPositionNegEdge" contain the position at which the positive or negative edge occurred.

PositionDifference, DetectedPositionDifference

The "PositionDifference" output contains the difference of the two latest measured values of the same touch probe in case of an activated difference measurement. At each measurement, the "DetectedPositionDifference" output is increased.

The following calculation applies if the drive touch probe function is activated for the "PositionDifference" output:

$$\text{PositionDifference} = |(\text{S-0-0130}) - (\text{S-0-0131})|$$

$$\text{PositionDifference} = |(\text{S-0-0132}) - (\text{S-0-0133})|$$

Fig.5-100: Calculating the position difference for axes

MissingMarksPosEdge, MissingMarksNegEdge

The "MissingMarksPosEdge" or "MissingMarksNegEdge" outputs are only evaluated if the expectation window is active and marker failure was enabled. They count the marks outside the expectation window. The outputs are set to 0 every time a marker is detected within the expectation window. If the touch probe is set to evaluate both edges, only the "MissingMarksPosEdge" output is valid. Currently, the drive touch probe function does not support individual marker failures for each edge at the same time.

Time diagram

ML_TechBase.library

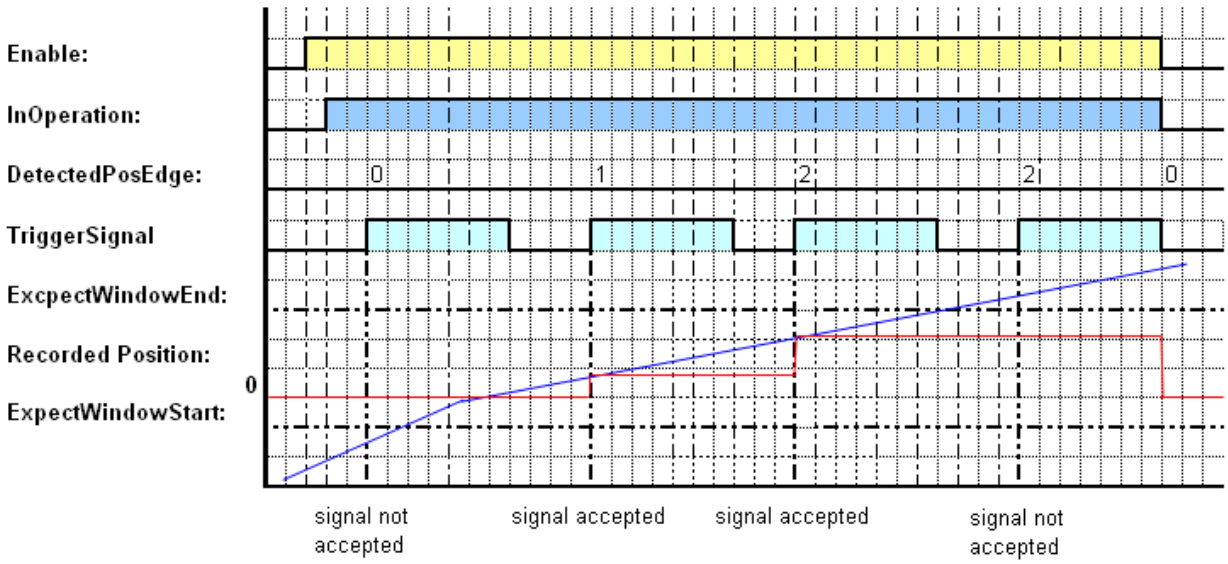


Fig.5-101: Time diagram of the MB_TouchProbeContinuous function block

Error Handling The function block generates the following error messages in Additional1/Additional2 for the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"AxisNo" for "Axis" is not within the valid range
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis (Axis_Ref) was changed while the function block was in operation
ACCESS_ERROR (16#0003)	16#0406	16#0001	Empty handle for ATPosEdgeVal
ACCESS_ERROR (16#0003)	16#0406	16#0002	Empty handle for ATNegEdgeVal
ACCESS_ERROR (16#0003)	16#0406	16#0003	Empty handle for ATMarkerFailure
ACCESS_ERROR (16#0003)	16#0406	16#0004	Empty handle for ATDiffProbeVal
ACCESS_ERROR (16#0003)	16#0406	16#0005	Empty handle for ATDeadTime-Comp
ACCESS_ERROR (16#0003)	16#0406	16#0006	Empty handle for SCWEnableP-robe
ACCESS_ERROR (16#0003)	16#0406	16#0007	Empty handle for SSWPosLatched
ACCESS_ERROR (16#0003)	16#0406	16#0008	Empty handle for SSWNegLatched

Fig.5-102: Error codes of the MB_TouchProbeContinuous function block

5.7.6 MB_WriteExpectWindow

Brief Description The MB_WriteExpectWindow function block configures the expectation window for the touch probe functionality.

Interface Description

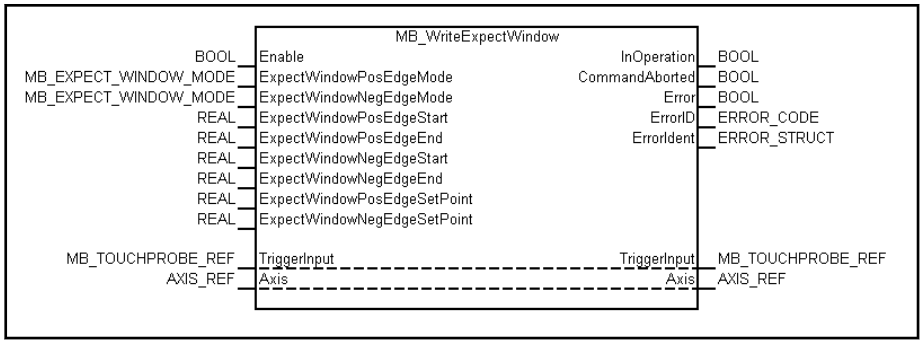


Fig.5-103: MB_WriteExpectWindow function block

I/O type	Name	Data type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis
	TouchProbe	TOUCHPROBE_REF	Reference to the signal trigger source
VAR_INPUT	Enable	BOOL	It enables the function block
	ExpectWindowPosEdgeMode	MB_EXPECT_WINDOW_MODE	Mode of expectation window if the positive edge is evaluated
	ExpectWindowNegEdgeMode	MB_EXPECT_WINDOW_MODE	Mode of expectation window if the negative edge is evaluated
	ExpectWindowPosEdgeStart	REAL	Start of expectation window if the positive edge is evaluated
	ExpectWindowPosEdgeEnd	REAL	End of expectation window if the positive edge is evaluated
	ExpectWindowNegEdgeStart	REAL	Start of expectation window if the negative edge is evaluated
	ExpectWindowNegEdgeEnd	REAL	End of expectation window if the negative edge is evaluated
	ExpectWindowPosEdgeSetPoint	REAL	Command value for relative expectation window if the positive edge is evaluated
	ExpectWindowNegEdgeSetPoint	REAL	Command value for relative expectation window if the negative edge is evaluated
VAR_OUTPUT	InOperation	BOOL	Function block in operation
	CommandAborted	BOOL	Function block aborted
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

Fig.5-104: Interface variables of the MB_WriteExpectWindow function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

ML_TechBase.library

Name	Type	Min. value	Max. value	Default values	Effective
Enable	BOOL			FALSE	Continuous
ExpectWindow-PosEdgeMode	MB_EX-PECT_WIN-DOW_MODE			DISABLED_EXP	Rising edge at "Enable"
ExpectWindow-NegEdgeMode	MB_EX-PECT_WIN-DOW_MODE			DISABLED_EXP	Rising edge at "Enable"
ExpectWindow-PosEdgeStart	REAL	0.0	ModuloValue	0.0	Continuous
ExpectWindow-PosEdgeEnd	REAL	0.0	ModuloValue	0.0	Continuous
ExpectWindow-NegEdgeStart	REAL	0.0	ModuloValue	0.0	Continuous
ExpectWindow-NegEdgeEnd	REAL	0.0	ModuloValue	0.0	Continuous
ExpectWindow-PosEdgeSetPoint	REAL	0.0	ModuloValue	0.0	Continuous
ExpectWindow-NegEdgeSetPoint	REAL	0.0	ModuloValue	0.0	Continuous

Fig.5-105: Min./max. and default values for inputs of the MB_WriteExpectWindow function block

Functional description of the drive touch probe

Currently, the expectation window of the drive does not support a separate expectation window for the negative and positive edges at the same time. If only the positive edge is evaluated, the inputs "ExpectWindowPosEdge-Mode","ExpectWindowPosEdgeStart","ExpectWindowPosEdgeStart" and "ExpectWindowPosEdgeSetPoint" are evaluated.

If the negative edge is evaluated, "ExpectWindowNegEdgeMode", "Expect-WindowNegEdgeStart","ExpectWindowNegEdgeStart" and "ExpectWindow-NegEdgeSetPoint" are evaluated.

if both edges are evaluated, only the inputs "ExpectWindowPosEdge-Mode","ExpectWindowPosEdgeStart","ExpectWindowPosEdgeEnd" and "Ex-pectWindowPosEdgeSetPoint" are evaluated for the expectation window of the drive. The expectation window applies to both evaluated edges.

Time diagram

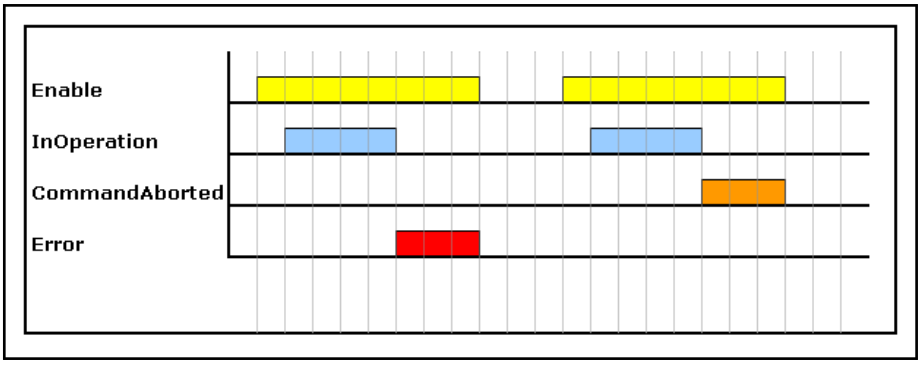


Fig.5-106: Time diagram of the MB_WriteExpectWindow function block

Error Handling

The function block generates the following error messages in Additional1/Ad-ditional2 for the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"AxisNo" for "Axis" is not within the valid range
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis (Axis_Ref) was changed while the function block was in operation
INPUT_INVALID_ERROR(16#0001)	16#0405	16#0001	The "ExpectWindowPosEdge-Mode" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0405	16#0002	The "ExpectWindowNegEdge-Mode" input is invalid
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0003	The "ExpectWindowPosEdgeStart" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0004	The "ExpectWindowPosEdgeEnd" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0005	The "ExpectWindowNegEdgeStart" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0006	The "ExpectWindoNegEdgeEnd" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0007	The "ExpectWindowPosEdgeSet-Point" input is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0405	16#0008	The "ExpectWindowNegEdgeSet-Point" input is not within the valid range
RESOURCE_ERROR (16#0003)	16#0403	16#000F	P-0-1367, Bit 6 of the MLD-M master axis is not set to TRUE. This is required for the update of the "Axis-Data" structure.

Fig.5-107: Error codes of the MB_WriteExpectWindow function block

5.7.7 MB_AbortTrigger

Brief Description

The MB_AbortTrigger function block aborts the MB_TouchProbe function block.

Interface Description

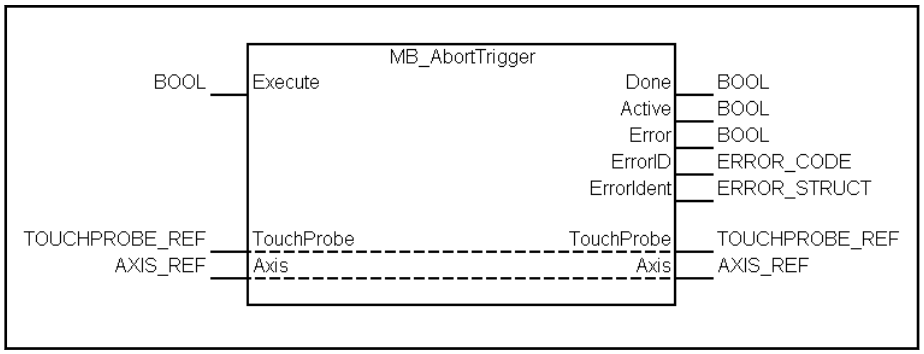


Fig.5-108: MB_AbortTrigger function block

ML_TechBase.library

I/O types	Name	Data type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis
	TouchProbe	TOUCHPROBE_REF	Reference to the signal trigger source
VAR_INPUT	Execute	BOOL	It starts the function block at a rising edge
VAR_OUTPUT	Done	BOOL	Touch probe aborted
	Active	BOOL	The function block is currently active.
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

Fig.5-109: Interface variables of the MB_AbortTrigger function block

Default value The following table lists the default value of the function block input.

Name	Type	Min. value	Max. value	Default values	Effective
Execute	BOOL			FALSE	Continuous

Fig.5-110: Default value for the input of the MB_AbortTrigger function block

Functional Description The MB_AbortTrigger function block disables the touch probe function. It can only be used together with the MB_TouchProbe function block. To disable the continuous measurement, the "Enable" input of the MB_TouchProbeContinuous function block has to be set to FALSE.

Time diagram Time diagram according to the PLCopen specification.

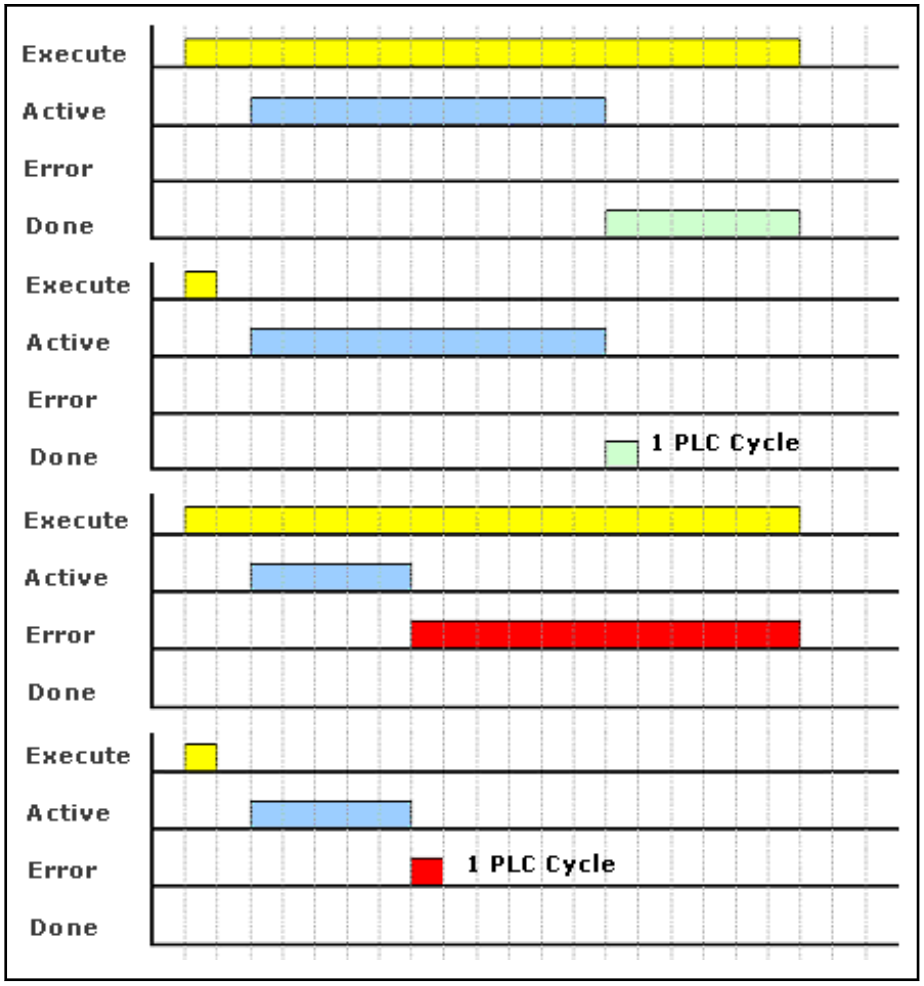


Fig.5-111: Time diagram of the MB_AbortTrigger function block

Error Handling

The function block generates the following error messages in Additional1/Additional2 for the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"AxisNo" for "Axis" is not within the valid range
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis (Axis_Ref) was changed while the function block was in operation
ACCESS_ERROR (16#0003)	16#0406	16#0006	Empty handle for SCWEnableProbe

Fig.5-112: Error codes of the MB_AbortTrigger function block

5.7.8 Touch Probe Data Types

MB_PROBE_EXECUTION_MODE

The following data types define the touch probe function mode either as single-shot measurement or continuous measurement:

ML_TechBase.library

Program:

```
TYPE MB_PROBE_EXECUTION_MODE :
(
 CONTINUOUS :=0,  (* touch probe function continuous measurement *)
 SINGLE_SHOT := 1 (* touch probe function single shot measurement *)
);
END_TYPE
```

MB_EXPECT_WINDOW_MODE

If the input "ExpectWindowMode" is set to DISABLED_EXP, the expectation window is disabled.

If the input "ExpectWindowMode" is set to EXPECT_WIN_ABS, the beginning and the end of the expectation window are absolute.

If the input ""ExpectWindowMode"" is set to EXPECT_WIN_REL, the beginning and the end of the expectation window are relative to the variable value of the ExpectWindowPosEdgeSetPoint(ExpectWindowPosEdgeSetPoint).

The relative expectation window for the positive edge is [ExpectWindowPosEdgeSetPoint – ExpectWindowPosEdgeStart.. ExpectWindowPosEdgeSetPoint + ExpectWindowPosEdgeEnd].

For the negative edge, it is [ExpectWindowNegEdgeSetPoint – ExpectWindowNegEdgeStart .. ExpectWindowNegEdgeSetPoint + ExpectWindowNegEdgeEnd].

Program:

```
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
```

MB_PROBE_DATA

The following data type defines the measured touch probe signal and configures the parameters S-0-0426/S-0-0427 depending on the touch probe selected:

Program:

```
 TYPE MB_PROBE_DATA :
(
 TP_ACT_POS_FEEDBACK1:=1,  (* S-0-0051 A-0-0100 Actual position 1 value *)
 TP_ACT_POS_FEEDBACK2:=2,  (* S-0-0053 A-0-0100 Actual position 2 value *)
 TP_ACT_POSIN_ACT_CYCLE:=3,(* P-0-0753 Actual position value in actual value cycle *)
 TP_SYNC_CMD_POS :=4,      (* P-0-0778 A-0-2653 Synchronous position command value *)
 TP_PROBE_TIME:=5,         (* time measurement *)
 TP_ACT_MEASURING_ENC_POS:=6,(* P-0-0052 Actual position value of measuring encoder *)
 TP_RES_MASTER_AXIS_POS:=7,  (* P-0-0775 A-0-2650 Resulting master axis position *)
 TP_EFF_MASTER_AXIS_POS:=8,  (* P-0-0776 A-0-2651 Effective master axis position *)
 TP_CAM_SHAFT_PROF_ACCESS_ANGLE:=9(*P-0-0227 A-0-2656 Cam shaft profile, access angle*)
);
END_TYPE
```

MB_POSITION_EVAL

The following data type selects the edge to be evaluated:

Program:

```
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  (* positive and negative edge is evaluated *)
);
END_TYPE
```

MB_DIFF_EVAL

The following data type selects the difference measurement:

- DIFF_DISABLED: Difference measurement disabled
- DIFF_STANDARD_MODE: The difference of both the values measured last of the same touch probe (IndraMotion MLC/IndraLogic XLC/drive touch probe)
- DIFF_NEG_TO_NEG_EDGE: The difference of the two latest evaluated negative edge values of the same touch probe (IndraMotion MLC/ IndraLogic XLC/only touch probe)
- DIFF_POS_TO_NEG_EDGE: The difference of the last positive and last negative evaluated edge value of the same touch probe (IndraMotion MLC/IndraLogic XLC/only touch probe)
- DIFF_NEG_TO_POS_EDGE: The difference of the last negative and last positive evaluated edge value of the same touch probe (IndraMotion MLC/IndraLogic XLC/only touch probe)
- DIFF_POS_TO_POS_EDGE: The difference of the two latest evaluated positive edge values of the same touch probe (IndraMotion MLC/ IndraLogic XLC/only touch probe)

Program:

```
TYPE MB_DIFF_EVAL :
(
 DIFF_DISABLED         := 0, (* difference measurement is disabled        *)
 DIFF_STANDARD_MODE    := 1, (* difference of two measured values         *)
 DIFF_NEG_TO_NEG_EDGE  := 2, (* negative edge to negative edge difference *)
 DIFF_POS_TO_NEG_EDGE  := 3, (* positive edge to negative edge difference *)
 DIFF_NEG_TO_POS_EDGE  := 4, (* negative edge to positive edge difference *)
 DIFF_POS_TO_POS_EDGE  := 5  (* positive edge to positive edge difference *)
);
END_TYPE
```

TOUCHPROBE_REF

The following data type defines the configuration used for the control and for the touch probe:

Program:

```
TYPE TOUCHPROBE_REF :
 STRUCT CntrlNo:
     CONTROLS     := LOCAL_CNTRL;    (* logical control address *)
     TouchProbeNo: TP_OBJECTS   := NO_TOUCHPROBE;  (* touch probe number           *)
 END_STRUCT
END_TYPE
```

5.8 Trigonometric Functions

5.8.1 Introduction

The following functions allow users to makes conversions among various units. Because these are functions, no errors can be output. If invalid values are used, a 0 value is output.

5.8.2 ML_DegToInc

Brief Description The function ML_DegToInc converts the angle indicated in degrees (0°<=x<360°) to the corresponding increment value (0<=x<2^20).

ML_TechBase.library

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-113: Reference table of the ML_DegToInc function

Interface Description

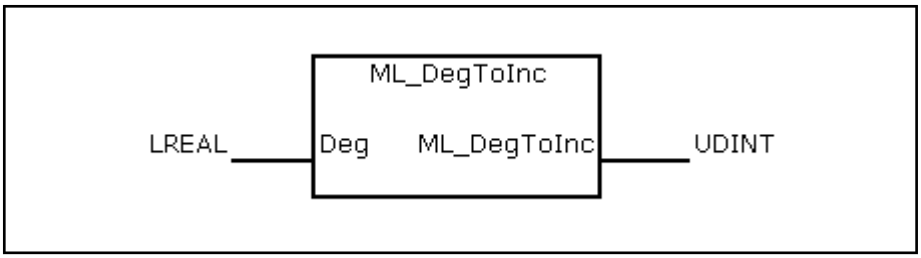


Fig.5-114: ML_DegToInc function

I/O type	Name	Data type	Description
VAR_INPUT	Deg	LREAL	Angle in degrees [DEG]
RETURN	ML_DegToInc	UDINT	Angle in increments

Fig.5-115: Interface variables of the ML_DegToInc function

Min./max. and default value

The following table lists the min./max. and default values of the function block input.

Name	Type	Min. value	Max. value	Default value
Deg	LREAL	-2^32 * 360.0	2^32 * 360.0	0.0

Fig.5-116: Min./max. and default value for the ML_DegToInc function

Functional Description

The function ML_DegToInc converts the angle indicated in degrees (0°<=x<360°) to the corresponding increment value (0<=x<2^20). If an angle is specified as larger than or equal to 360°, it is converted to the respective angle within the valid value area (modulo 360).

If a value outside of the valid value range is specified, 0 is output.

Error codes

The function ML_DegToInc does not generate any errors.

5.8.3 ML_DegToRad

Brief Description

The function ML_DegToRad converts an angle value in degrees to the corresponding radian incl. modulo.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-117: Reference table of the ML_DegToRad function

Interface Description

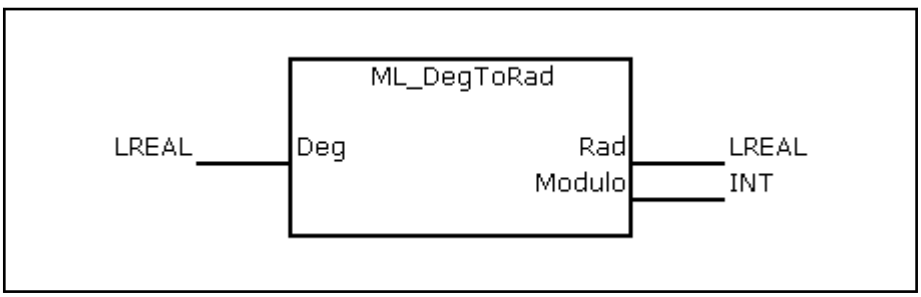


Fig.5-118: ML_DegToRad function

I/O type	Name	Data type	Description
VAR_INPUT	Deg	LREAL	Angle [DEG]
VAR_OUTPUT	Rad	LREAL	Angle [RAD]
	Modulo	INT	Modulo

Fig.5-119: Interface variables of the ML_DegToRad function

Min./max. and default value

The following table lists the min./max. and default values of the function block input.

Name	Type	Min. value	Max. value	Default value
Deg	LREAL	-2^15 * 360.0	2^15 * 360.0	0.0

Fig.5-120: Min./max. and default value for the ML_DegToRad function

Functional Description

At an instance call, the ML_DegToRad function cyclically converts the indicated angle in degrees to the respective radian incl.Modulo.

If a value outside of the valid value range is specified, 0 is output.

Error codes

The function ML_DegToRad does not generate any errors.

5.8.4 ML_IncToDeg

Brief Description

The function ML_IncToDeg converts the angle indicated in increments (0<=x<2^20) to the corresponding angle in degrees (0°<=x<360°).

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-121: Reference table of the ML_IncToDeg function

Interface Description

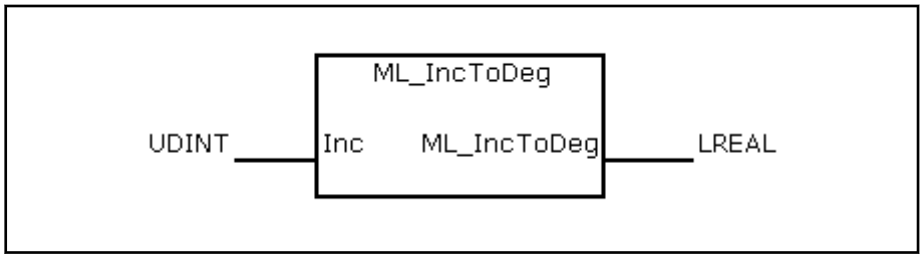


Fig.5-122: ML_IncToDeg function

I/O type	Name	Data type	Description
VAR_INPUT	Inc	UDINT	Angle in increments
RETURN	ML_IncToDeg	LREAL	Angle in degrees [DEG]

Fig.5-123: Interface variable of the ML_IncToDeg function

Functional Description

The function ML_IncToDeg converts the angle indicated in increments (0<=x<2^20) to the corresponding angle in degrees (0°<=x<360°). If an angle is specified larger than or equal to 2^20 increments, it is converted to the respective angle within the valid value area (modulo 2^20).

Error codes

The function ML_IncToDeg does not generate any errors.

5.8.5 ML_IncToRad

Brief Description

The function ML_IncToRad converts the angle indicated in increments (0<=x<2^20) to the corresponding angle in radian (0<=x<2*PI).

ML_TechBase.library

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-124: Reference table of the ML_IncToRad function

Interface Description

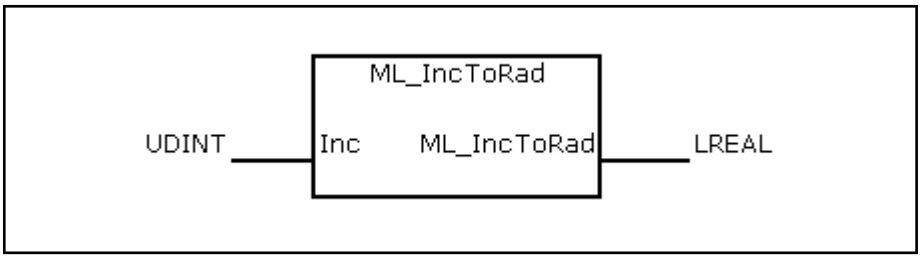


Fig.5-125: ML_IncToRad function

I/O type	Name	Data type	Description
VAR_INPUT	Inc	UDINT	Angle in increments
RETURN	ML_IncToRad	LREAL	Angle in radian [RAD]

Fig.5-126: Interface variables of the ML_IncToRad function

Functional Description

The function ML_IncToRad converts the angle indicated in increments (0<=x<2^20) to the corresponding angle in radian (0<=x<2*PI). If an angle is specified larger than or equal to 2^20 increments, it is converted to the respective angle within the valid value area (modulo 2^20).

Error codes

The function ML_IncToRad does not generate any errors.

5.8.6 ML_RadToDeg

Brief Description

The function ML_RadToDeg converts an angle value in radian to the corresponding degrees incl. Modulo.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-127: Reference table of the ML_RadToDec function

Interface Description

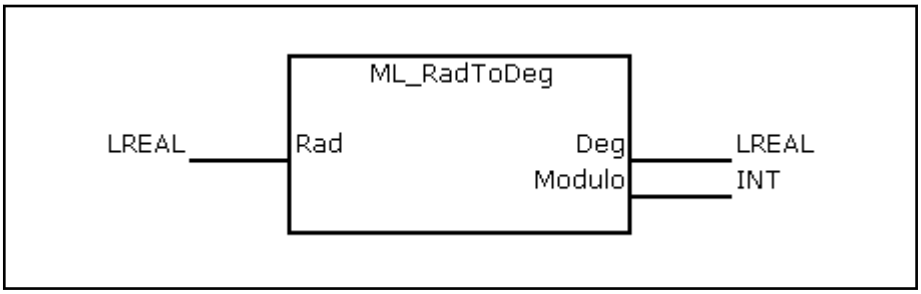


Fig.5-128: ML_RadToDeg function

I/O type	Name	Data type	Description
VAR_INPUT	Rad	LREAL	Angle [RAD]
VAR_OUTPUT	Deg	LREAL	Angle [DEG]
	Modulo	INT	Modulo

Fig.5-129: Interface variable of the ML_RadToDeg function

Min./max. and default value

The following table lists the min./max. and default values of the function block input.

Name	Type	Min. value	Max. value	Default value
Rad	LREAL	-2 * PI * 2^15	2 * PI * 2^15	0.0

Fig.5-130: Min./max. and default value for the ML_RadToDeg function

Functional Description

At an instance call, the function ML_RadToDeg cyclically converts the indicated angle in radian to the respective degrees incl. Modulo.

If a value outside of the valid value range is specified, 0 is output for "Deg" and "Modulo".

Error codes

The function ML_RadToDeg does not generate any errors.

5.8.7 ML_RadToInc

Brief Description

The function ML_RadToInc converts the angle indicated in radian (0<=x<2*PI) to the corresponding increment value (0<=x<2^20).

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-131: Reference table of the ML_RadToInc function

Interface Description

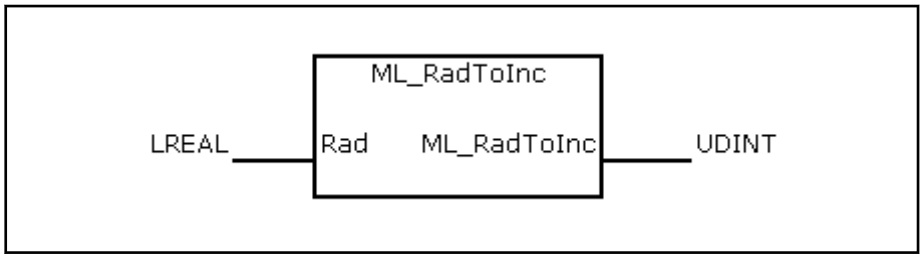


Fig.5-132: ML_RadToInc function

I/O type	Name	Data type	Description
VAR_INPUT	Rad	LREAL	Angle in radian [RAD]
RETURN	ML_RadToInc	UDINT	Angle in increments

Fig.5-133: Interface variable of the ML_RadToInc function

Min./max. and default value

The following table lists the min./max. and default values of the function block input.

Name	Type	Min. value	Max. value	Default value
Rad	LREAL	-2 * PI * 2^32	2 * PI *2^32	0.0

Fig.5-134: Min./max. and default value for the ML_RadToInc function

Functional Description

The function ML_RadToInc converts the angle indicated in radian (0<=x<2*PI) to the corresponding increment value (0<=x<2^20). If an angle is specified larger than or equal to 2*PI, it is converted to the respective angle within the valid value area (modulo 2*PI).

If a value outside of the valid value range is specified, 0 is output.

Error codes

The function ML_RadToInc does not generate any errors.

5.9 Function Blocks and Functions for Maintenance "Cyclic Data Channels"

5.9.1 Introduction and Overview

With the SERCOS interface, it is possible to apply freely configurable cyclic data. These data is cyclically transferred in each SERCOS cycle by the control to the drives and from the drives to the control. Four parameters are available in read and write direction for each axis. In addition, four bits can be read or written. The cyclic data is configured in the parameterization mode (PM, SERCOS phase 2) and cannot be changed in the operation mode (BB, SERCOS phase 4).

In the parameterization mode (PM, SERCOS phase 2), a cyclic data channel for a parameter to be transferred can be requested by the MB_AllocCyclicDataParameter function block. If the parameter has already been applied in the cyclic data channel, the configuration is not changed. Otherwise, a free data channel is browsed for and the respective parameter is applied.

If the configuration changed, a cyclic parameter applied once can be enabled again with chapter 5.9.3 "MB_FreeCyclicParameter" on page 125. Thus, this channel is available for a new configuration. The channel must only be explicitly enabled if the data channels are newly configured during runtime from the PLC.

The configuration can also be executed in IndraWorks via AxisData or via the cyclic actual and command value dialog. A configuration defined by IndraWorks is used automatically. If some parameters are configured via IndraWorks and if other parameters are requested later during runtime with chapter 5.9.2 "MB_AllocCyclicParameter" on page 123, the new parameters are distributed to free channels. The current configuration of the cyclic applied parameters can be read with chapter 5.9.4 "MB_GetCyclicChannelConfig" on page 127.

In the operating mode (BB), the function block chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128 browses for a cyclically applied parameter in the configuration of one axis and writes all required information into a handle. With this handle, the parameter can be read or written in each SERCOS cycle with chapter 5.9.6 "MB_ReadCyclicParameter" on page 130 and chapter 5.9.8 "MB_WriteCyclicParameter" on page 132. The functions chapter 5.9.7 "MB_ReadCyclicRealParameter" on page 131 and chapter 5.9.9 "MB_WriteCyclicRealParameter" on page 132 are available for cyclic parameters with decimal places. Errors that occurred while accessing the cyclic data channels can be retrieved via the function chapter 5.9.10 "MB_GetLastCyclicParameterError" on page 133.

The cyclically applied parameters are also available in the AxisData structure of the respective axis. The actual values are specified in UserActualDataX. The command values are written via UserCmdDataX. However, the correct data type according to the applied parameter has to be described. This is not required when using the function blocks MB_ReadCyclicParameter and MB_WriteCyclicParameter.

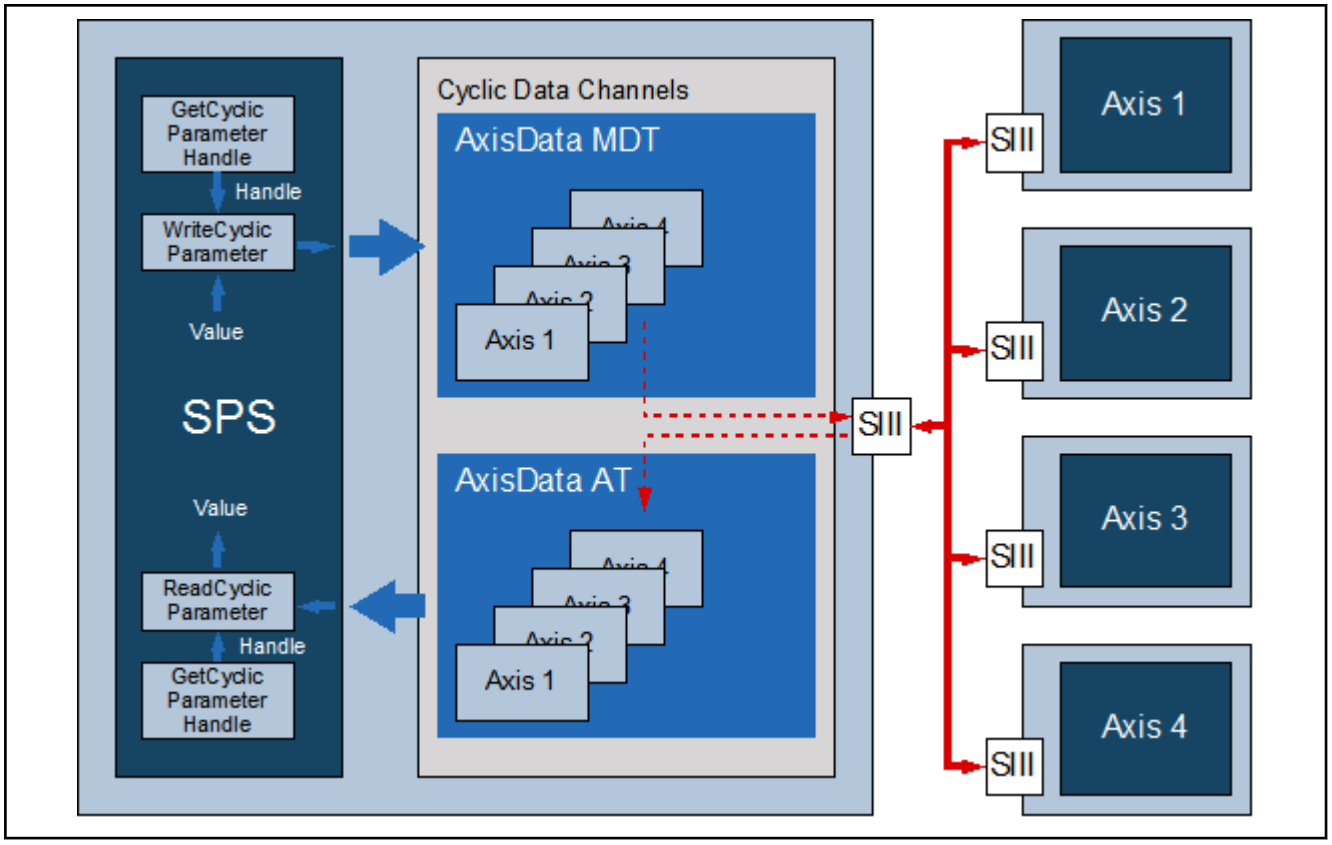


Fig.5-135: Access to cyclic data using the "CyclicParameter" function blocks and functions

5.9.2 MB_AllocCyclicParameter

Brief Description

This function block checks the configuration of a parameter to be read or written to on a cyclic basis in the cyclic data channel of an axis and configures the cyclic data channel if required. Assigned parameters are stored in a global data structure.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-136: Reference table

ML_TechBase.library

Interface Description

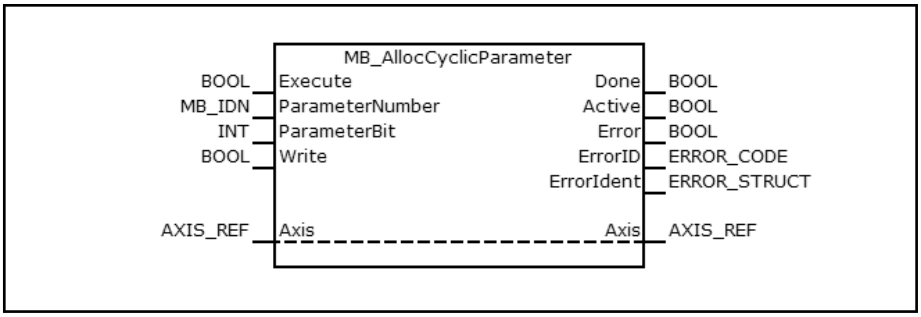


Fig.5-137: Function block MB_AllocCyclicParameter

I/O type	Name	Type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Parameter axis
VAR_INPUT	Execute	BOOL	Starts the parameter configuration
	Parameter number	MB_IDN	Parameter to be applied
	ParameterBit	INT	Internal bit of the parameter; "CYCLIC_ALLBITS" for the entire parameter
	Write	BOOL	FALSE: Read parameter TRUE: Write parameter
VAR_OUTPUT	Done	BOOL	Parameter was applied
	Active	BOOL	The function block is active
	Error	BOOL	Indicates that an error has occurred
	ErrorID	ERROR_CODE	Basic error reason
	ErrorIdent	ERROR_STRUCT	Error structure with further fragmentation of the errors, see "Error codes" on page 125

Fig.5-138: Interface variables of the MB_AllocCyclicParameter function block

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
Parameter number	DINT	0	n.def.	0	Rising edge at "Execute"
ParameterBit	INT	-1	31	CYCLIC_ALLBITS	Rising edge at "Execute"
Write	BOOL			FALSE	Rising edge at "Execute"

Fig.5-139: Input behavior of the MB_AllocCyclicParameter

Functional Description

This function block only works in SERCOS phase 2. At a rising edge at "Execute", the inputs "ParameterNumber", "ParameterBit" and "Write" are applied. The actual data (UserActualData) and the command data (UserCmdData) are differentiated using the "Write" input. "ParameterBit" specifies the internal bit number of the parameter. If "ParameterBit" is between 0 and 31, a bit data channel is used. If "ParameterBit" is equal to "CYCLIC_ALLBITS", the entire parameter is applied. "ParameterNumber" specifies the parameter to be applied.

In the first step, MB_AllocCyclicParameter checks if the parameter has already been stored in the current configuration. If this is not the case, a free

cyclic channel is browsed for and the parameter is applied there. Subsequently, the configuration is stored in an internal static variable. If a bit number if specified in "ParameterBit", a free bit channel is preferred. If all bit channels are assigned, a parameter channel is used. After having successfully finished the configuration of the cyclic data channel, the output "Done" is set to TRUE for at least one cycle. "Done" remains TRUE as long as the input "Execute“ is active. If there is no free cyclic channel available, a corresponding error message is displayed.

Error characteristics If the parameter cannot be applied in the cyclic data channel or if another error occurs, "Error" is set to TRUE.

Error codes The function block generates the following error messages of the "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#000D	AxisNo	AxisRef is not within the valid range
ACCESS_ERROR	16#0010	16#0000	SERCOS phase 2 is not active
ACCESS_ERROR	16#1504	16#0000	AxisData structure not activated
INPUT_INVALID_ERROR	16#F212	16#0000	Parameter type structure not supported
INPUT_INVALID_ERROR	16#F213	16#0000	Parameter type function not supported
INPUT_INVALID_ERROR	16#F214	16#0000	Parameter type not supported
INPUT_INVALID_ERROR	16#F215	16#0000	Parameter is not writable
RESSOURCE_ERROR	16#F216	16#0000	No free cyclic channel
INPUT_INVALID_ERROR	16#F217	16#0000	Parameter cannot be configured in the cyclic data channel

Fig.5-140: Error codes of the MB_AllocCyclicParameter function block

5.9.3 MB_FreeCyclicParameter

Brief Description This function block enables an already configured cyclic parameter to be read or written in the cyclic data channel of an axis. The enabling of the parameter is stored in a global structure.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-141: Reference table

ML_TechBase.library

Interface Description

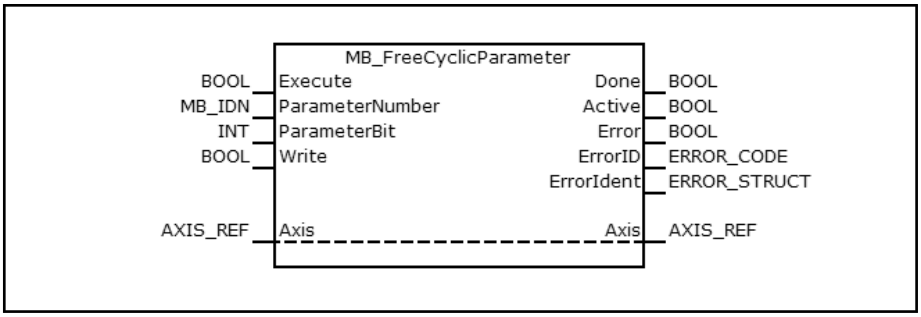


Fig.5-142: Function block MB_FreeCyclicParameter

I/O type	Name	Data type	Description
VAR_IN_OUT	Axis	AXIS_REF	Parameter axis
VAR_INPUT	Execute	BOOL	Starts the enabling of the parameter
	Parameter number	MB_IDN	Number of the parameter to be enabled
	ParameterBit	INT	Internal bit of the parameter; "CYCLIC_ALLBITS" for the entire parameter
	Write	BOOL	FALSE: Read parameter TRUE: Write parameter
VAR_OUTPUT	Done	BOOL	Parameter was applied
	Active	BOOL	The function block is active
	Error	BOOL	Indicates that an error has occurred
	ErrorID	ERROR_CODE	Basic error reason
	ErrorIdent	ERROR_STRUCT	Error structure with further fragmentation of the errors, see "Error codes" on page 127

Fig.5-143: Interface variables of the MB_FreeCyclicParameter function block

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
Parameter number	DINT	0	n.def.	0	Rising edge at "Execute"
ParameterBit	INT	-1	31	CYCLIC_ALLBITS	Rising edge at "Execute"
Write	BOOL			FALSE	Rising edge at "Execute"

Fig.5-144: Input behavior of the MB_FreeCyclicParameter function block

Functional Description

This function block only works in SERCOS phase 2. At a rising edge at "Execute", the inputs "ParameterNumber", "ParameterBit" and "Write" are applied. The actual data (UserActualData) and the command data (UserCmdData) are differentiated using the "Write" input. "ParameterBit" specifies the internal bit number of the parameter. If "ParameterBit" is between 0 and 31, a bit data channel is used. If "ParameterBit" is equal to "CYCLIC_ALLBITS", the entire parameter is applied. "ParameterNumber" specifies the parameter to be enabled.

In the first step, MB_FreeCyclicParameter checks if the parameter has already been stored in the current configuration. If this is not the case, an error

message is displayed. Subsequently, the enabling is stored in an internal static variable. MB_FreeCyclicParameter does not change the actual configuration of the cyclic data channels. The enabling is only stored in the internal control structure. After having successfully finished the enabling of the cyclic data channel, the output "Done" is set to TRUE for at least one cycle. "Done" remains TRUE as long as the input "Execute" is active.

Error characteristics If the parameter is not applied in the cyclic data channel. If another error occurs, "Error" is set to TRUE.

Error codes The function block generates the following error messages of the "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#000D	16#0000	AxisRef is not within the valid range
ACCESS_ERROR	16#0010	16#0000	SERCOS phase 2 is not active
ACCESS_ERROR	16#1504	16#0000	AxisData structure not activated
ACCESS_ERROR	16#F211	16#0000	Parameter not found in the cyclic data channel

Fig.5-145: Error codes of the MB_FreeCyclicParameter function block

5.9.4 MB_GetCyclicChannelConfig

Brief Description The MB_GetCyclicChannelConfig function block reads the configuration of the cyclic data channels of an axis. Read channels as well as write channels are read and stored in a data structure. The number of entries depend on the device and is output in an individual entry.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-146: Reference table of the MB_GetCyclicChannelConfig function block

Interface Description

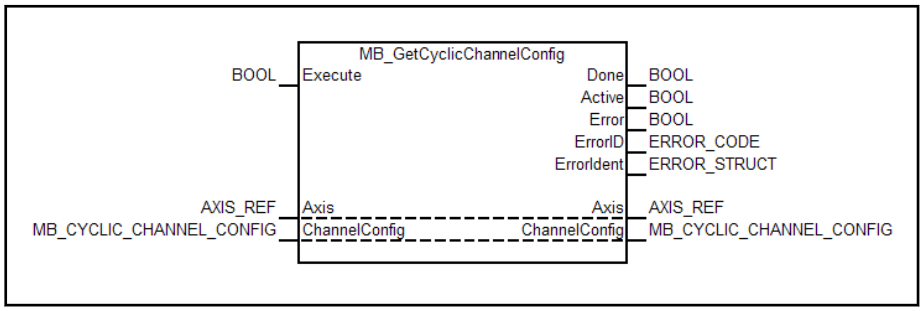


Fig.5-147: MB_GetCyclicChannelConfig function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Axis	AXIS_REF	Parameter axis
	ChannelConfig	MB_CYCLIC_CHANNEL_CONFIG	Read configuration, see chapter 5.9.14 "MB_CYCLIC_CHANNEL_CONFIG" on page 135
VAR_INPUT	Execute	BOOL	Starts the parameter configuration

ML_TechBase.library

I/O type	Name	Data type	Description
VAR_OUTPUT	Done	BOOL	Parameter was applied
	Active	BOOL	The function block is active
	Error	BOOL	Indicates that an error has occurred
	ErrorID	ERROR_CODE	Basic error reason
	ErrorIdent	ERROR_STRUCT	Error structure with further fragmentation of the errors, see "Error codes" on page 128

Fig.5-148: Interface variables of the MB_GetCyclicChannelConfig function block

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous

Fig.5-149: Input behavior of the MB_GetCyclicChannelConfig

Functional Description

This function block only operates in the SERCOS phases 2 and 4. Cyclic data channels only exist for real axes. Thus, a function block can only be used for real axes. At a rising edge at "Execute", the created axis is checked and the configuration of the cyclic data channels is read. This operation requires several cycles. During this period, the "Active" output is set. "Active" is reset and "Done" is set after a successful completion of the reading process. The configuration is now available at the "ChannelConfig“ output. The "ChannelConfig" output is only valid after the "Done" output was set.

The number of available data channels depends on the axis and is stored in the "…Count" elements of the "ChannelConfig" structure. The elements "ReadChannel", "WriteChannel", "ReadBitChannel" and "WriteBitChannel" contain the assigned parameter numbers. If no parameter is assigned, 0 is entered. The elements "ReadBitChannelBit" and "WriteBitChannelBit" contain the active bit number of the parameter.

Error characteristics

If an error occurs during the processing, the outputs "Active" and "Done" are reset and "Error" is set. "ErrorID" and "ErrorIdent" contain more information on the error if the "Error" output is set.

Error codes

The function block generates the following error messages of the "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#000D	16#0000	AxisRef is not within the valid range

Fig.5-150: Error codes of the MB_GetCyclicChannelConfig function block

5.9.5 MB_GetCyclicParameterHandle

Brief Description

This function block browses for a parameter to be cyclically read or written in the cyclic data channel of an axis and stores all required information in a handle.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib

IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-151: Reference table

Interface Description

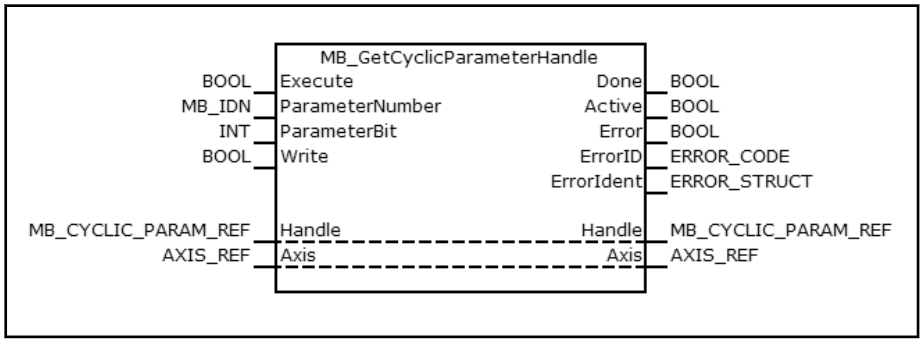


Fig.5-152: MB_GetCyclicParameterHandle function block

I/O type	Name	Type	Comment
VAR_IN_OUT	Handle	MB_CYCLIC_PARAM_REF	It contains information on the access on the cyclic parameter
	Axis	AXIS_REF	Parameter axis
VAR_INPUT	Execute	BOOL	Starts the parameter search
	Parameter number	MB_IDN	Number of the parameter cyclically applied
	ParameterBit	INT	Internal bit of the parameter; "CYCLIC_ALLBITS" for the entire parameter
	Write	BOOL	FALSE: Read parameter TRUE: Write parameter
VAR_OUTPUT	Done	BOOL	Parameter was found and "Handle" is valid
	Active	BOOL	The function block is active
	Error	BOOL	It indicates that an error occurred
	ErrorID	ERROR_CODE	Basic error reason
	ErrorIdent	ERROR_STRUCT	Error structure with further fragmentation of the errors, see "Error codes" on page 130

Fig.5-153: Interface variables of the MB_GetCyclicParameterHandle function block

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
Parameter number	DINT	0	n.def.	0	Rising edge at "Execute"
ParameterBit	INT	-1	31	CYCLIC_ALLBITS	Rising edge at "Execute"
Write	BOOL			FALSE	Rising edge at "Execute"

Fig.5-154: Input behavior of the MB_GetCyclicParameterHandle

ML_TechBase.library

Functional Description

This function block is used in the operating mode (SERCOS phase 4). After a rising edge at "Execute", the inputs "ParameterNumber", "ParameterBit" and "Write" are read. The input "Write" decides if the command values of the MDT or if the actual values of the AT are browsed for. If "ParameterBit" is "CYCLIC_ALLBITS", the complete parameter is applied. If a bit number is specified in "ParameterBit", the bit channel is browsed first. If the parameter is not found in the bit channels and if this parameter is an actual value, the parameter channels are browsed. After having successfully finished the processing, "Done" is active and the data in the created "Handle" is valid.

Error characteristics

If the parameter is not found in the cyclic data channel or if another error occurs, "Error" is set to TRUE and "Handle" is not changed.

Error codes

The function block generates the following error messages of the "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#000D	16#0000	AxisRef is not within the valid range
ACCESS_ERROR	16#1504	16#0000	AxisData structure not activated
ACCESS_ERROR	16#F211	16#0000	Parameter not found in the cyclic data channel
INPUT_INVALID_ERROR	16#F212	16#0000	Parameter type structure not supported
INPUT_INVALID_ERROR	16#F213	16#0000	Parameter type function not supported
INPUT_INVALID_ERROR	16#F214	16#0000	Parameter type not supported
INPUT_INVALID_ERROR	16#F215	16#0000	Parameter is not writable

Fig.5-155: Error codes of the MB_GetCyclicParameterHandle function block

5.9.6 MB_ReadCyclicParameter

Brief Description

This function reads a cyclical parameter via "Handle" and returns the current content as DINT. Values with decimal places are converted according to the scaling stored in the SERCOS attribute. This function can also be used for bit values.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-156: Reference table

Interface Description

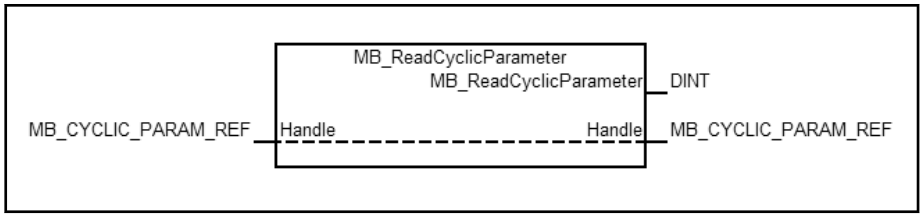


Fig.5-157: Function MB_ReadCyclicParameter

Input variable	Data type	Description
Handle	MB_CYCLIC_PARAM_REF	It contains information on the access on the cyclic parameter
Return value	**Data type**	**Description**
MB_ReadCyclicParameter	DINT	Value of the parameter

Fig.5-158: Interface variables of the MB_ReadCyclicParameter function

5.9.7 MB_ReadCyclicRealParameter

Brief Description

This function reads a cyclical parameter via "Handle" and returns the current content as REAL. If the data type of the parameter is not REAL, the type is converted. This function cannot be used for bit values.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-159: Reference table

Interface Description

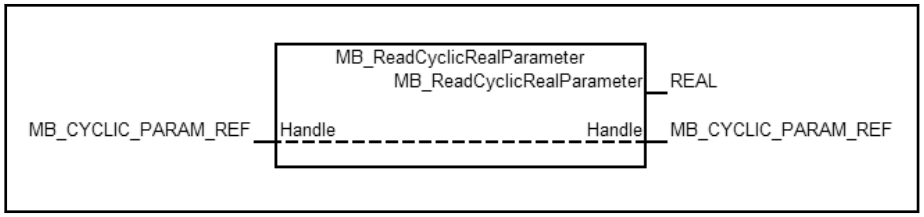


Fig.5-160: Function MB_ReadCyclicRealParameter

Input variable	Data type	Description
Handle	MB_CYCLIC_PARAM_REF	It contains information on the access on the cyclic parameter
Return value	**Data type**	**Description**
MB_ReadCyclicRealParameter	REAL	Value of the parameter

Fig.5-161: Interface variables of the MB_ReadCyclicRealParameter function

ML_TechBase.library

5.9.8 MB_WriteCyclicParameter

Brief Description

This function writes a cyclical parameter via "Handle". If the target data type has decimal places, the value is converted according to the decimal places specified in the SERCOS attribute. This function can also be used for bit values.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-162: Reference table

Interface Description

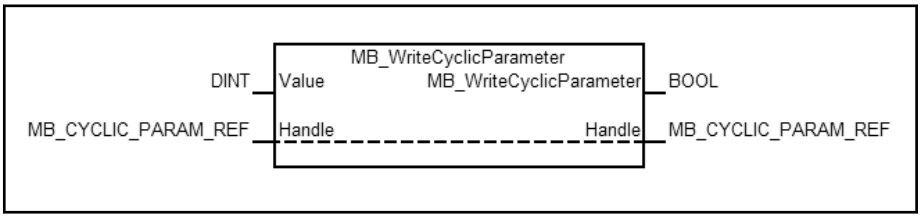


Fig.5-163: Function MB_WriteCyclicParameter

Input variable	Type	Description
Handle	MB_CYCLIC_PARAM_REF	It contains information on the access on the cyclic parameter
Value	DINT	Value to be written
Return value	**Type**	**Description**
MB_WriteCyclicParameter	BOOL	Dummy output without meaning; is always TRUE

Fig.5-164: Interface variables of the MB_WriteCyclicParameter function

5.9.9 MB_WriteCyclicRealParameter

Brief Description

This function writes a cyclical parameter via "Handle". Integral target data types without decimal places are rounded. This function cannot be used for bit values.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-165: Reference table

Interface Description

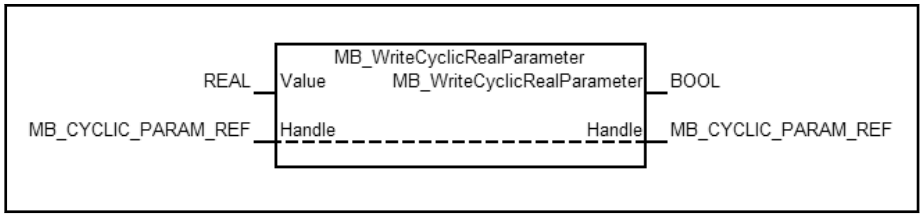


Fig.5-166: Function MB_WriteCyclicRealParameter

Input variable	Type	Description
Handle	MB_CYCLIC_PARAM_REF	It contains information on the access on the cyclic parameter
Value	REAL	Value to be written
Return value	**Type**	**Description**
MB_WriteCyclicRealParameter	BOOL	Dummy output without meaning; is always TRUE

Fig.5-167: Interface variables of the MB_WriteCyclicRealParameter function

5.9.10 MB_GetLastCyclicParameterError

Brief Description

This function returns the last occurred error when reading or writing a cyclical value for a "handle" and deletes the error.

This function is optional and does not have to be called. However, it is recommended to check the cyclical data communication for occurred errors on a regular basis.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-168: Reference table

Interface Description

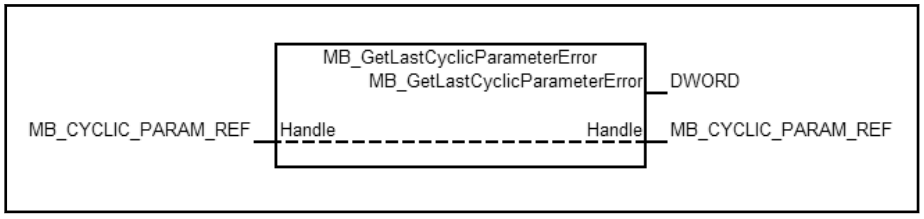


Fig.5-169: Function MB_GetLastCyclicParameterError

Input variable	Data type	Description
Handle	MB_CYCLIC_PARAM_REF	It contains information on the access on the cyclic parameter

ML_TechBase.library

Input variable	Data type	Description
Return value	**Data type**	**Description**
MB_GetLastCyclicParameterError	DWORD	Error code of the last reading or writing action

Fig.5-170: Interface variables of the MB_GetLastCyclicParameterError function

Error codes

The MB_GetLastCyclicParameterError function generates the following errors:

ErrorID	Description
16#0001	Cannot write parameter - parameter only configured for reading
16#0002	Wrong type for bit access
16#0003	Wrong type for parameter access
16#0004	Invalid axis number
16#0005	Invalid data channel
16#0006	Invalid handle

Fig.5-171: Error codes of the function GetLastCyclicParameterError

5.9.11 MB_CYCLIC_PARAM_REF

Type definitions

The following definition describes data types used in the function block chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128.

The type definitions are included in the library.

The structure "MB_CYCLIC_PARAM_REF" may be read but must not be written. Changing the content can lead to malfunctions. The structure can change in later versions. The structure should only be read for debugging purposes.

The structure "MB_CYCLIC_PARAM_REF" is as follows:

Structure element	Type	Description
Parameter number	MB_IDN	FP-number of the parameter
AxisNo	OBJECTS	Logic axis number
ParamType	MB_CYCLIC_TYPES	Parameter data type
Bit	INT	Internal bit number of the parameter
AccessChannel	MB_CYCLIC_CHANNELS	Data channel in which the parameter was found
ParamAdr	POINTER TO SM_TYPES	Pointer to the cyclic parameter
BitParamAdr	POINTER TO WORD	Pointer to the cyclic bit parameter
Factor	DINT	Used to convert REAL to DINT
ErrorCode	DWORD	Error code of the last error, see "Error codes" on page 134

Fig.5-172: Definition of the basic elements of the structure MB_CYCLIC_PARAM_REF

5.9.12 MB_CYCLIC_TYPES

The data type "MB_CYCLIC_TYPES" is an enumeration type with the following entries:

Element	Value
tINT	1
tUINT	2
tDINT	3
tUDINT	4
tREAL	5
tBIT	6

Fig.5-173: Definition of the enumeration type MB_CYCLIC_TYPES

5.9.13 MB_CYCLIC_CHANNELS

The data type "MB_CYCLIC_CHANNELS" is an enumeration type with the following entries:

Element	Value
UserCmdDataA	0
UserCmdDataB	1
UserCmdDataC	2
UserCmdDataD	3
UserCmdDataBitA	32
UserCmdDataBitB	33
UserCmdDataBitC	34
UserCmdDataBitD	35
UserActualDataA	256
UserActualDataB	257
UserActualDataC	258
UserActualDataD	259
UserActualDataBitA	288
UserActualDataBitB	289
UserActualDataBitC	290
UserActualDataBitD	291

Fig.5-174: Definition of the enumeration type MB_CYCLIC_CHANNELS

5.9.14 MB_CYCLIC_CHANNEL_CONFIG

The "MB_CYCLIC_CHANNEL_CONFIG" structure is described by the chapter 5.9.4 "MB_GetCyclicChannelConfig" on page 127 function block and contains the applied parameters in the cyclic data channel of the axis.

The structure "MB_CYCLIC_CHANNEL_CONFIG" is as follows:

Structure element	Type	Min. value	Max. value	Default value	Description
ReadChannelCount	UINT	0	10	0	Number of read channels
ReadChannel	ARRAY[1..10] OF MB_IDN	n.def.	n.def.	S-0-0000	Configured parameters of the read channels

ML_TechBase.library

Structure element	Type	Min. value	Max. value	Default value	Description
WriteChannelCount	UINT	0	10	0	Number of write channels
WriteChannel	ARRAY[1..10] OF MB_IDN	n.def.	n.def.	S-0-0000	Configured parameters of the write channels
ReadBitChannelCount	UINT	0	10	0	Number of bit read channels
ReadBitChannel	ARRAY[1..10] OF MB_IDN	n.def.	n.def.	S-0-0000	Configured parameters of the bit read channels
ReadBitChannelBit	ARRAY[1..10] OF UINT	0	63	0	Configured bits of the bit read channels
WriteBitChannelCount	UINT	0	10	0	Number of bit write channels
WriteBitChannel	ARRAY[1..10] OF MB_IDN	n.def.	n.def.	S-0-0000	Configured parameters of the bit write channels
WriteBitChannelBit	ARRAY[1..10] OF UINT	0	63	0	Configured bits of the bit write channels

Fig.5-175: Definition of the MB_CYCLIC_CHANNEL_CONFIG structure

5.10 Function Blocks to Backup and Restore Drive Parameters

5.10.1 Introduction and Overview

Drives are normally parameterized via IndraWorks. At running systems, it is often required to exchange drives without using IndraWorks and to parameterize them again.

This use case is covered by two function blocks. ML_AxisBackup creates a parameter backup and stores it as text file on the CompactFlash card of the control. ML_AxisRestore transfers the parameter backup back to the axis.

To restore parameters, the drive has to be in Parameterization mode (P2). This is also recommended for the parameter backup.

5.10.2 ML_AxisBackup

Brief Description

The ML_AxisBackup function block backs up the parameters of an axis in a text file on the CompactFlash card of the control.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-176: Reference table of the ML_AxisBackup function block

Interface Description

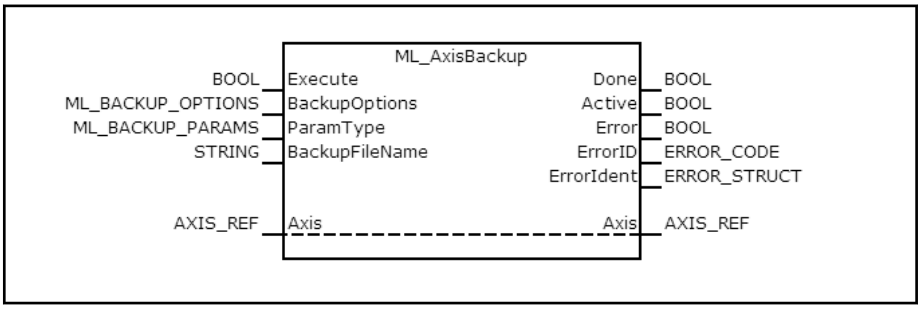


Fig.5-177: ML_AxisBackup function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (edge-controlled)
	BackupOptions	ML_BACKUP_OPTIONS	Extend of the parameter backup
	ParamType	ML_BACKUP_PARAMS	Selection of the parameters to be backed up (all, A, S and P)
	BackupFileName	STRING	File name on the CompactFlash card of the control
VAR_OUTPUT	Done	BOOL	Processing completed, output data valid
	Active	BOOL	Processing running, output data valid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Error diagnostics (see next table Error codes, page 138)
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis

Fig.5-178: Interface variables of the ML_AxisBackup function block

Min./max. and default values of the inputs

The following value ranges and default values can be applied at the function block:

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
BackupOptions	ML_BACKUP_OPTIONS			TO_SAVE_PARAMS	Rising edge at "Execute"
ParamType	ML_BACKUP_PARAMS			ASP_PARAMS	Rising edge at "Execute"
BackupFileName	STRING			''	Rising edge at "Execute"

Fig.5-179: Min./max. and default values of the ML_AxisBackup function block

Functional Description

The ML_AxisBackup function block backs up the parameter of an axis in a text file on the user partition of the CompactFlash card of the control. If the file already exists, it is overwritten. The file can be read again by the IL_AxisRestore function blocks and by IndraWorks.

BackupOptions

The parameters to be backed up are selected via the "BackupOptions" input. The following selections are possible:

- ALL_PARAM: All parameters are backed up. This option is primarily required to check the configuration
- MODIFIED_PARAM: The parameters deviating from the default values are backed up
- TO_SAVE_PARAM: The parameters required are backed up for restoration. This is the presetting

☞ The "BackupOptions" is currently only evaluated for S- and P-parameters. When exporting A-parameters, all parameters are always exported.

ParamType

The parameter types to be backed up are selected via the "ParamType" input. The following options are available for selection:

- ASP_PARAMS: Parameters in the drive and control-internal axis parameters
- A_PARAMS: Control-internal axis parameters
- SP_PARAMS: Parameters in the drive

BackupFileName

Via "BackupFileName", the file name of the CompactFlash card of the control can be specified. It is possible to use a path specification. The path has to be available on the CompactFlash card.

Example:

- "Drive1.par" (stores the file "Drive1.par" on the user partition of the CompactFlash card.)
- "Params/drive1.par" (stores the file "Drive1.par" in the "Params" directory on the user partition of the CompactFlash card.)
- "/ata0a/drive2.par" (stores the file "drive2.par" on the OEM partition of the control)

Time response The function block ML_AxisBackup works asynchronously. After the start due to a rising edge at the "Execute" input, it reads the inputs "BackupOptions", "ParamType" and "BackupFileName". "Active" is set as long as the function block is operating. After the operation has been successfully completed, the "Done" output is set. If an error occurs, the "Error" output is set.

Error codes The function block uses the error table **F_RELATED_TABLE**. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#0000000D	16#00000000	AxisRef is not within the valid range
INPUT_RANGE_ERROR	16#00001E00	16#00000000	Invalid "BackupOptions" input
INPUT_RANGE_ERROR	16#00001E01	16#00000000	Invalid "ParamType" input
RESOURCE_ERROR	16#00001E02	16#00000000	The parameter type selected in "ParamType" is not supported by this axis
RESOURCE_ERROR	16#00001E03	16#xxxxxxxx	Error when reading a parameter
ACCESS_ERROR	16#00001E10	16#00000000	Error when creating the backup file
ACCESS_ERROR	16#00001E12	16#00000000	Error when writing the backup file
ACCESS_ERROR	16#00001E14	16#00000000	Error when closing the backup file
STATE_MACHINE_ERROR	16#00000006	16#0000FFFx	Invalid state of the function block

Fig.5-180: Error codes of the ML_AxisBackup function block

5.10.3 ML_AxisRestore

Brief Description The ML_AxisRestore function block loads the parameters from a text file on the CompactFlash card of the control to an axis.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-181: Reference table of the ML_AxisRestore function block

Interface Description

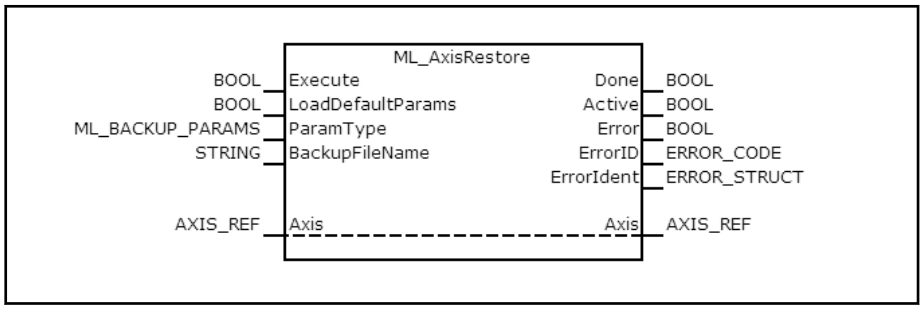


Fig.5-182: ML_AxisRestore function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (edge-controlled)
	LoadDefaultParams	BOOL	Loading default parameters before loading the parameters from the file
	ParamType	ML_BACKUP_PARAMS	Selection of the parameters to be restored (all, A, S and P)
	BackupFileName	STRING	File name on the CompactFlash card of the control
VAR_OUTPUT	Done	BOOL	Processing completed, output data valid
	Active	BOOL	Processing running, output data valid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Error diagnostics (see table Error codes, page 140).
VAR_IN_OUT	Axis	AXIS_REF	Reference to the axis

Fig.5-183: Interface variables of the ML_AxisRestore function block

Min./max. and default values of the inputs

The following value ranges and default values can be applied at the function block:

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
LoadDefaultParams	BOOL			FALSE	Rising edge at "Execute"
ParamType	ML_BACKUP_PARAMS			ASP_PARAMS	Rising edge at "Execute"
BackupFileName	STRING			''	Rising edge at "Execute"

Fig.5-184: Min./max. and default values of the ML_AxisRestore function block

Functional Description

The ML_AxisRestore function block loads the parameters from a text file on the CompactFlash card of the control to an axis. After the parameter file has been loaded successfully, the configuration is saved on the drive or the axis.

It is thus also available after deactivation. The parameter file can be created by the ML_AxisBackup function block or by IndraWorks.

LoadDefaultParams

If the "LoadDefaultParams" input is set, the default parameters of the drive or the axis are loaded before loading the parameters. This is especially reasonable if the parameter file was created using the options "MODIFIED_PARAM" or "TO_SAVE_PARAM".

☞ Loading the default parameters takes several seconds per axis.

ParamType

The parameter types to be restored are selected via the "ParamType" input. The following options are available for selection:

- ASP_PARAMS: Parameters in the drive and control-internal axis parameters
- A_PARAMS: Control-internal axis parameters
- SP_PARAMS: Parameters in the drive

BackupFileName

Via "BackupFileName", the file name of the CompactFlash card of the control can be specified. It is possible to use a path specification. The path has to be available on the CompactFlash card.

Example:

- "Drive1.par" (reads the file "Drive1.par" from the user partition of the CompactFlash card.)
- "Params/drive1.par" (reads the file "Drive1.par" from the "Params" directory on the user partition of the CompactFlash card.)
- "/ata0a/drive2.par" (reads the file "drive2.par" from the OEM partition of the control)

Time response

The function block ML_AxisRestore operates asynchronously. After the start due to a rising edge at the "Execute" input, it reads the inputs "LoadDefaultParams", "ParamType" and "BackupFileName". "Active" is set as long as the function block is operating. After the operation has been successfully completed, the "Done" output is set. If an error occurs, the "Error" output is set.

Error codes

The function block uses the error table **F_RELATED_TABLE**. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#0000000D	16#00000000	AxisRef is not within the valid range
INPUT_RANGE_ERROR	16#00001E01	16#00000000	Invalid "ParamType" input
RESOURCE_ERROR	16#00001E02	16#00000000	The parameter type selected in "ParamType" is not supported by this axis
RESOURCE_ERROR	16#00001E04	16#xxxxxxxx	Error when writing a parameter
ACCESS_ERROR	16#00001E11	16#00000000	Error when opening the backup file
ACCESS_ERROR	16#00001E13	16#00000000	Error when reading the backup file
ACCESS_ERROR	16#00001E14	16#00000000	Error when closing the backup file

ErrorID	Additional1	Additional2	Description
OTHER_ERROR	16#00001E15	16#xxxxxxxx	Error when parsing the backup file
STATE_MACHINE_ERROR	16#00000006	16#0000FFFx	Invalid state of the function block

Fig.5-185: Error codes of the ML_AxisRestore function block

5.10.4 ML_BACKUP_OPTIONS

Type definitions

The following definition describes the enumeration type ML_BACKUP_OPTIONS used in the function block chapter 5.10.2 "ML_AxisBackup" on page 136.

The type definitions are included in the library.

The data type "ML_BACKUP_OPTIONS" is defined as follows:

Enumeration element	Value	Description
ALL_PARAM	1	All parameters
MODIFIED_PARAM	2	Parameters deviating from default value
TO_SAVE_PARAM	3	Parameters required for restoration

Fig.5-186: Definition of the enumeration elements of the ML_BACKUP_OPTIONS data type

5.10.5 ML_BACKUP_PARAMS

Type definitions

The following definition describes the enumeration type ML_BACKUP_PARAMS used in the function blocks chapter 5.10.2 "ML_AxisBackup" on page 136 and chapter 5.10.3 "ML_AxisRestore" on page 138.

The type definitions are included in the library.

The data type "ML_BACKUP_OPTIONS" is defined as follows:

Enumeration element	Value	Description
ASP_PARAMS	1	Parameters in the drive and control-internal axis parameters
A_PARAMS	2	Control-internal axis parameters
SP_PARAMS	3	Parameters in the drive

Fig.5-187: Definition of the enumeration elements of the ML_BACKUP_PARAMS data type

5.11 Function Blocks for Logic Applications

5.11.1 Introduction

The following function block was designed for applications at which a double-linked list is required. Elements can be added and deleted. The function block manages the list. Thus, ring buffers, FIFOs and LIFOs can be designed.

5.11.2 IL_LinkedList

Brief Description

The function block maintains a double-linked list. The list contents can be specified by the user.

ML_TechBase.library

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx18 (in preparation)	In preparation

Fig.5-188: Reference table of the IL_LinkedList function block

Interface Description

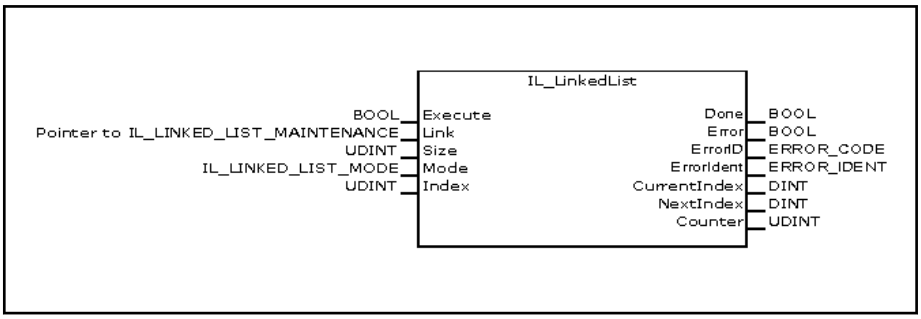


Fig.5-189: IL_LinkedList function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	The function block starts its functionality at a rising edge
	Link	Pointer to IL_LINKED_LIST_MAINTENANCE	Pointer to an array of IL_LINKED_LIST_MAINTENANCE structures required for maintaining the list
	Size	UDINT	Number of list elements to be maintained
	Mode	IL_LINKED_LIST_MODE	Mode that executed the function block, e.g. adding or removing an element
	Index	DINT	Index on which the operation should be applied
VAR_OUTPUT	Done	BOOL	If this output is TRUE, the function block has successfully completed its work
	Error	BOOL	Indicates that an error occurred in the function block. The output data is invalid
	ErrorID	ERROR_CODE	Error ID (see table Error codes, page 146).
	ErrorIdent	ERROR_STRUCT	Error structure with further subdivision of error classes
	CurrentIndex	DINT	Current index
	NextIndex	DINT	Next index
	Counter	UDINT	Number of elements in the list

Fig.5-190: Interface variables of the IL_LinkedList function block

Min./max. and default values of the inputs

The following value ranges and default values can be applied at the function block:

Name	Type	Min. value	Max. value	Default value
Link	0	0	16#FFFFFFFF	0
Size	UDINT	0	16#FFFFFFFF	0
Mode	IL_LINKED_LIST_MODE	ModeInitClear	ModeCheckIndex	ModeInitClear
Index	UDINT	0	16#FFFFFFFF	0

Fig.5-191: Min./max. and default values of the IL_LinkedList function block

Functional Description

The function block maintains a double-linked list. This list is stored in an array in which only the indices must be maintained. To maintain the list, the user must create an array of type IL_LINKED_LIST_MAINTENANCE (see definition IL_LINKED_LIST_MAINTENANCE structure, page 143) that must be as large as its own data.

The list is manipulated by calling the function block. The manipulation type can be defined by the "Mode" input (see Definitions for the Mode input, page 143).

IL_LINKED_LIST_MAINTENANCE structure

The data structure IL_LINKED_LIST_MAINTENANCE is used by the function block and the user may not overwrite it. Because users can determine the length of the linked list, they must create the maintain structures for the list as well. This can be done by defining an array of type IL_LINKED_LIST_MAINTENANCE. The internal structure looks like the following and can be used for debugging purposes. "Prev" is a pointer for the previous list element, "Next" for the subsequent element. If the value is 0, it is the end of the array, the beginning or an unused element.

Program:

```
TYPE IL_LINKED_LIST_MAINTENANCE :
STRUCT
    Prev: POINTER TO DINT;
    Next: POINTER TO DINT;
END_STRUCT
END_TYPE
```

Definitions for the "Mode" input

The user can use the "Mode" input to define the action to be executed. The following enumerations are available:

Enumeration	Description
ModeInitClear	The linked list is initialized. All entries are deleted
ModeAddTail	An element is added at the tail of the list. If the list is empty, the first element is created
ModeAddHead	An element is added at the head of the list. If the list is empty, the first element is created
ModeRemoveTail	The last element in the list is removed
ModeRemoveHead	The first element in the list is removed
ModeRemoveIndex	The element in the index "Index" is removed
ModeGetHead	Returns the index of the head of the list
ModeGetTail	Returns the index of the tail of list
ModeGetNext	Returns the subsequent element in the list
ModeGetPrev	Returns the previous element in the list
ModeGetEmptyIndex	Returns the next free index in the list
ModeCheckIndex	Checks whether the specified index is used in the linked list. The result is returned in the "CurrentIndex": 1: Index is used -1: Index is not used

Fig.5-192: Enumerations and their meaning in the function block IL_LinkedList

Example

Example for using the function block:

Defining variables:

Program:

```
(* Maximum array size *)
uiMAX_ARRAY_SIZE: INT := 10;
(* My data structure array *)
arMyData:     ARRAY [1..uiMAX_ARRAY_SIZE] OF MY_STRUCTURE;
(* List management array*)
arLinkedList: ARRAY [1..uiMAX_ARRAY_SIZE] OF IL_LINKED_LIST_MAINTEANANCE;
(* Function block instance *)
fbMyList:     IL_LinkedList;
(* Index variable*)
udiIndex:     DINT;
```

Call for initialization:

Program:

```
fbMyList(Execute:= TRUE,
    Link:= ADR(arLinkedList),
    Size:= uiMAX_ARRAY_SIZE,
    Mode:= ModeInitClear,
    Index:= ,
    Done=> ,
    Error=> ,
    ErrorId =>,
    ErrorIdent=> ,
    CurrentIndex=> ,
    NextIndex=>);
```

Adding a head to the list:

Program:

```
(* Reset "Execute" input first (value of mode is not important) *)
fbMyList(Execute:= FALSE,
          Link:= ADR(arLinkedList),
          Size:= uiMAX_ARRAY_SIZE,
          Mode:= ModeInitClear,
          Index:= ,
          Done=> ,
          Error=> ,
          ErrorId =>,
          ErrorIdent=> ,
          CurrentIndex=> ,
          NextIndex=>);

(* Add head *)
fbMyList(Execute:= TRUE,
          Link:= ADR(arLinkedList),
          Size:= uiMAX_ARRAY_SIZE,
          Mode:= ModeAddHead,
          Index:= ,
          Done=> ,
          Error=> ,
          ErrorId =>,
          ErrorIdent=> ,
          CurrentIndex=> udiIndex,
          NextIndex=>);

(* Return value in CurrentIndex contains index for list head or
   -1 if an error occurred. *)
IF udiIndex <> -1 THEN
    arMyData[udiIndex].StructureMember := MyValue;
END_IF;
```

Scroll through a list:

Program:

```
(* Reset "Execute" input first (value of mode is not important) *)
fbMyList(Execute:= FALSE,
          Link:= ADR(arLinkedList),
          Size:= uiMAX_ARRAY_SIZE,
       Mode:= ModeGetHead,
          Index:= ,
          Done=> ,
          Error=> ,
          ErrorId =>,
```

```
       ErrorIdent=> ,
          CurrentIndex=> ,
          NextIndex=>);

(* Determine head *)
fbMyList(Execute:= TRUE,
          Link:= ADR(arLinkedList),
       Size:= uiMAX_ARRAY_SIZE,
          Mode:= ModeGetHead,
          Index:= ,
          Done=> ,
          Error=> ,
          ErrorId =>,
          ErrorIdent=> ,
          CurrentIndex=> udiIndex,
       NextIndex=>);

(* Check return value (-1 means error) *)
WHILE (udiIndex > 0) DO
          (* Return value in CurrentIndex contains index in list *)
          arMyData[udiIndex].StructureMember := MyValue;

          (* Reset "Execute" input again *)
          fbMyList(Execute:= FALSE,
                    Link:= ADR(arLinkedList),
              Size:= uiMAX_ARRAY_SIZE,
              Mode:= ModeGetNext,
              Index:= ,
                    Done=> ,
                    Error=> ,
                    ErrorId =>,
                 ErrorIdent=> ,
                    CurrentIndex=> ,
                 NextIndex=>);

          (* Get next element*)
    fbMyList(Execute:= TRUE,
                    Link:= ADR(arLinkedList),
                 Size:= uiMAX_ARRAY_SIZE,
                    Mode:= ModeGetNext,
                    Index:= ,
                    Done=> ,
                 Error=> ,
                    ErrorId =>,
                    ErrorIdent=> ,
                 CurrentIndex=> udiIndex,
                    NextIndex=>);
     END_WHILE
```

Output assignment depending on the "Mode" input

Depending on the "Mode" input assignment, the "Index" input and the "CurrentIndex" and "NextIndex" are configured accordingly. The following table lists the respective values. An X stands for a value that is unchanged with respect to the previous call. If an error occurs, -1 is returned for "CurrentIndex".

Mode	Index	CurrentIndex	NextIndex
ModeInitClear	X	-1	Next free index
ModeAddTail	X	Tail index	Next free index
ModeAddHead	X	Head index	Next free index
ModeRemoveTail	X	New tail index	Next free index
ModeRemoveHead	X	New head index	Next free index
ModeRemoveIndex	Index for removal	Previous element index	Next free index
ModeGetHead	X	Head index	X
ModeGetTail	X	Tail index	X
ModeGetNext	Previous index	Subsequent element index	X
ModeGetPrev	Next index	Previous element index	X

ML_TechBase.library

Mode	Index	CurrentIndex	NextIndex
ModeGetEmptyIndex	X	X	Next free index
ModeCheckIndex	Testing index	1: found -1: not found	-1

Fig.5-193: Function block outputs with respect to "Mode"

Error codes

The function block uses the error table **F_RELATED_TABLE**. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#1800	16#0	The "Index" input is not within the valid range
INPUT_INVALID_ERROR	16#1801	16#0	"Link" input is a zero pointer
ACCESS_ERROR	16#1802	16#xxx	An index is still linked in the list although it is empty (internal error)
ACCESS_ERROR	16#1803	16#0	An element in the list is to be cleared, but the list is already empty
RESOURCE_ERROR	16#1804	16#0	List is full
STATE_MACHINE_ERROR	16#1805	16#0	"Mode" input has an invalid value

Fig.5-194: Error codes of the IL_LinkedList function block

5.12 Functions for Calculating Modulo Values

5.12.1 Introduction

The following functions are used to calculate a modulo value. Depending on function selected, different value ranges are used.

5.12.2 MB_GetModuloType01

Brief Description

The function returns the modulo value. The value range returned is between -modulo value and + modulo value.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx18	MX_Technology07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-195: Reference table of the MB_GetModuloType01 function

Interface Description

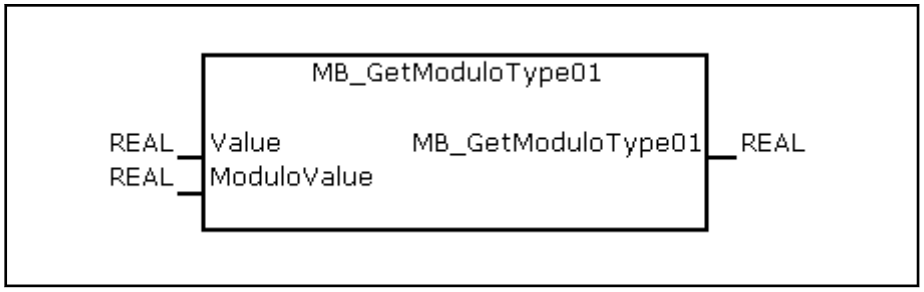


Fig.5-196: MB_GetModuloType01 function

I/O type	Name	Data type	Description
VAR_INPUT	Value	REAL	Value at which the modulo function should be used
	ModuloValue	REAL	Modulo value to be used for the modulo operation

Fig.5-197: Interface variables of the MB_GetModuloType01 function

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Value	REAL	- 1.175494e-38	3.402823e+38	0.0	With every call
ModuloValue	REAL	- 1.175494e-38	3.402823e+38	360.0	With every call

Fig.5-198: Min./max. and default values of the MB_GetModuloType01 function

Functional Description

The function returns the modulo value. The value range returned is between 0.0 and + modulo value. Positive and negative limit value are not part of the value range (also see the following diagram which shows the value transferred to the function on the x-coordinates and the value returned from the function on the y-coordinates).

If a value of 0.0 is transferred for "ModuloValue", the function returns the value "Value".

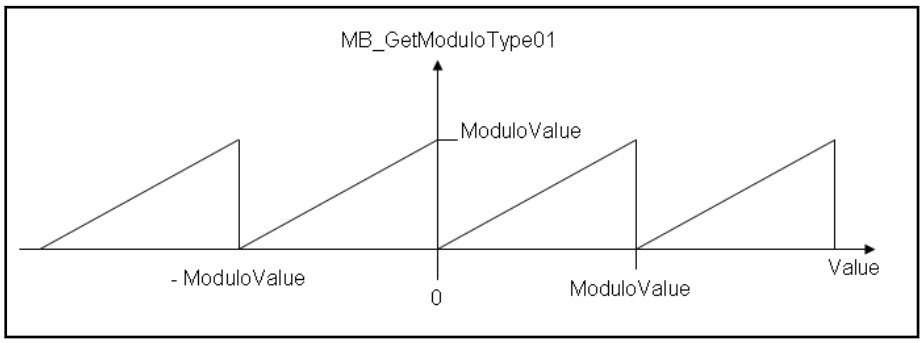


Fig.5-199: Representation of the modulo function

Error Handling

The function does not generate any errors.

5.12.3 MB_GetModuloType02

Brief Description

The function returns the modulo value. The value range returned is between - modulo value and + modulo value.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib

ML_TechBase.library

IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-200: Reference table

Interface Description

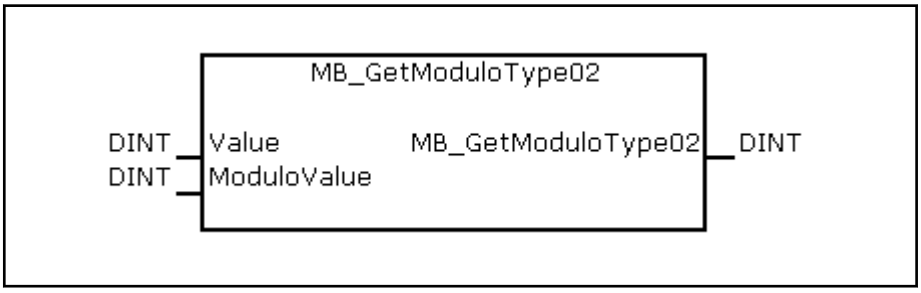


Fig.5-201: MB_GetModuloType02 function

I/O type	Name	Data type	Description
VAR_INPUT	Value	DINT	Value at which the modulo function should be used
	ModuloValue	DINT	Modulo value to be used for the modulo operation

Fig.5-202: Interface variables of the MB_GetModuloType02 function

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Value	DINT	- 2147483648	2147483647	0	With every call
ModuloValue	DINT	- 2147483648	2147483647	360	With every call

Fig.5-203: Min./max. and default values of the MB_GetModuloType02 function

Functional Description

The function returns the modulo value. The value range returned is between 0 and + modulo value. Positive and negative limit value are not part of the value range (also see the following diagram which shows the value transferred to the function on the x-coordinates and the value returned from the function on the y-coordinates).

If a value of 0 is transferred for "ModuloValue", the function returns the value "Value".

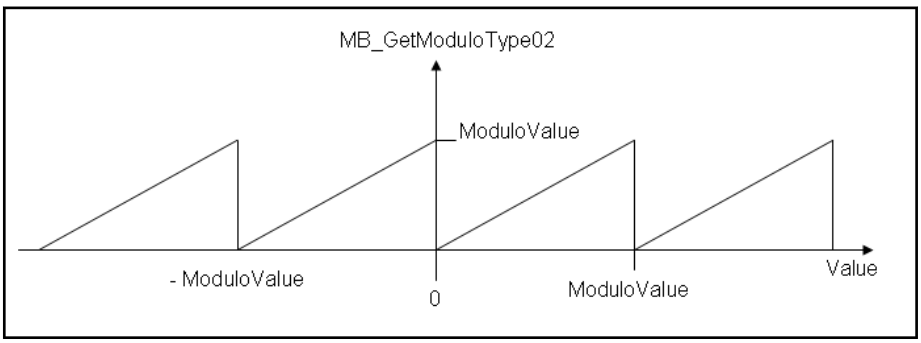


Fig.5-204: Representation of the modulo function

Error Handling

The function does not generate any errors.

5.12.4 MB_GetModuloType03

Brief Description

The function returns the modulo value. The value range returned is between - modulo value and + modulo value.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx18	MX_Technology07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-205: Reference table of the MB_GetModuloType03 function

Interface Description

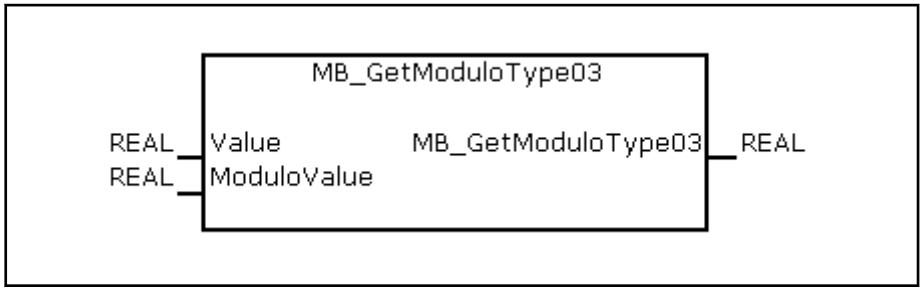


Fig.5-206: MB_GetModuloType03 function

I/O type	Name	Data type	Description
VAR_INPUT	Value	REAL	Value at which the modulo function should be used
	ModuloValue	REAL	Modulo value to be used for the modulo operation

Fig.5-207: Interface variables of the MB_GetModuloType03 function

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Value	REAL	- 1.175494e-38	3.402823e+38	0.0	With every call
ModuloValue	REAL	- 1.175494e-38	3.402823e+38	360.0	With every call

Fig.5-208: Min./max. and default values of the MB_GetModuloType03 function

Functional Description

The function returns the modulo value. The value range returned is between - modulo value and + modulo value. Positive and negative limit value are not part of the value range (also see the following diagram which shows the value transferred to the function on the x-coordinates and the value returned from the function on the y-coordinates).

If a value of 0.0 is transferred for "ModuloValue", the function returns the value "Value".

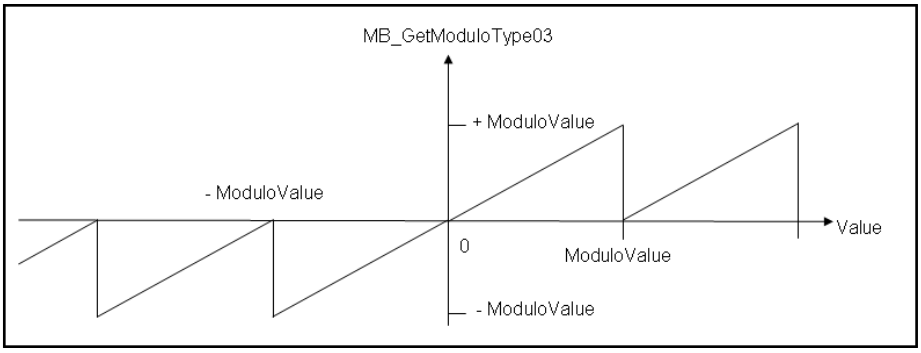


Fig.5-209: Representation of the modulo function

Error Handling The function does not generate any errors.

5.12.5 MB_GetModuloType04

Brief Description The function returns the modulo value. The value range returned is between - modulo / 2 and + modulo / 2.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx18	MX_Technology07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-210: Reference table of the MB_GetModuloType04 function

Interface Description

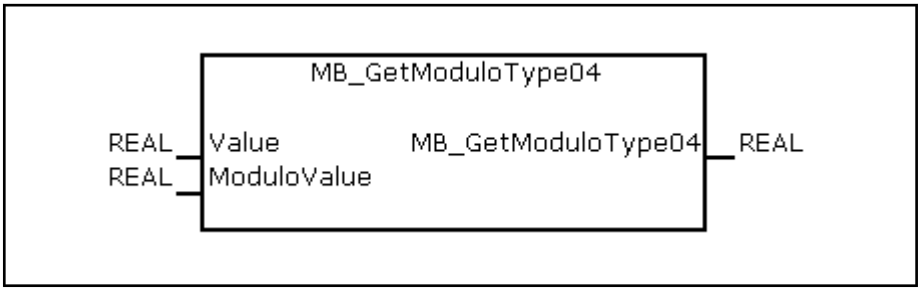


Fig.5-211: MB_GetModuloType04 function

I/O type	Name	Data type	Description
VAR_INPUT	Value	REAL	Value at which the modulo function should be used
	ModuloValue	REAL	Modulo value to be used for the modulo operation

Fig.5-212: Interface variables of the MB_GetModuloType04 function

Min./max. and default values The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Value	REAL	- 1.175494e-38	3.402823e+38	0.0	With every call
ModuloValue	REAL	- 1.175494e-38	3.402823e+38	360.0	With every call

Fig.5-213: Min./max. and default values of the MB_GetModuloType04 function

Functional Description

The function returns the modulo value. The value range returned lies between - modulo * 0.5 and + modulo * 0.5. Positive and negative limit value are part of the value range (also see the following diagram which shows the value transferred to the function on the x-coordinates and the value returned from the function on the y-coordinates).

If a value of 0.0 is transferred for "ModuloValue", the function returns the value "Value".

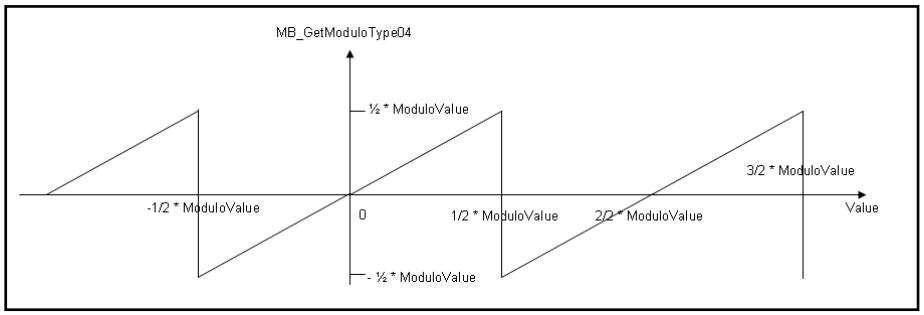


Fig.5-214: Representation of the modulo function

Error Handling

The function does not generate any errors.

5.13 Auxiliary Functions

5.13.1 MB_GetSystemInfo

Brief Description

This function block checks function packages and system options and displays all activated function packages and system options together with the system type.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.5-215: Reference table

Interface Description

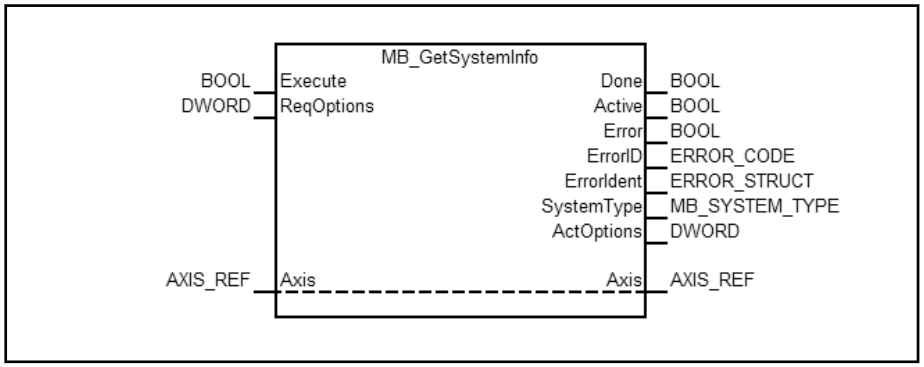


Fig.5-216: MB_GetSystemInfo function block

ML_TechBase.library

I/O type	Name	Data type	Description
VAR_IN_OUT	Axis	AXIS_REF	Axis to be checked
VAR_INPUT	Execute	BOOL	Starts the checking process
	ReqOptions	DWORD	Required system options
VAR_OUTPUT	Done	BOOL	Checking performed without errors
	Active	BOOL	The function block is active
	Error	BOOL	Indicates that an error has occurred
	ErrorID	ERROR_CODE	Basic error reason
	ErrorIdent	ERROR_STRUCT	Error structure with further fragmentation of the errors, see "Error codes" on page 153
	SystemType	MB_SYSTEM_TYPE	System type
	ActOptions	DWORD	Actually activated system options

Fig.5-217: Interface variables of the MB_GetSystemInfo function block

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
ReqOptions	DWORD	0	n.def.	0	Rising edge at "Execute"

Fig.5-218: Input behavior of the ML_GetSystemInfo

Functional Description

This function block works in SERCOS phase 2 or 4. At a rising edge at "Execute", the input "ReqOptions" are applied. Then, the system is checked. If options or function packages are set in "ReqOptions" which are not available, a corresponding error (see "Error codes" on page 153) is generated.

After successful completion, the system type is displayed in "SystemType" and all activated options are displayed in "ActOptions" and "Done" is set to TRUE.

Optional	IndraMotion MLC/IndraLogic XLC	IndraMotion MLD	Value	Description
MB_OPEN_LOOP	Axis	Axis	16#00000001	Basic package Open Loop
MB_CLOSED_LOOP	Axis	Axis	16#00000002	Basic package CloseddLoop
MB_SERVO	Axis	Axis	16#00000010	Function package Servo
MB_SYNCHRONIZATION	Axis	Axis	16#00000020	Function package Synchronization
MB_MAIN_SPINDLE	Axis	Axis	16#00000040	Function package main spindle
MB_ML	Always set	First axis	16#00000200	Option technology function
MB_TF	Always set	First axis	16#00000200	Option advanced technology function
MB_MA	Always set	First axis	16#00000300	
MB_SAFETY			16#00010000	Safety technology active in the axis

Optional	IndraMotion MLC/IndraLogic XLC	IndraMotion MLD	Value	Description
MB_AXISDATA	Local axes	First axis	16#00100000	AxisData activated
MB_REALAXIS	Real Axis	Always set	16#01000000	
MB_VIRTUALAXIS	Virtual axis		16#02000000	
MB_LINKAXIS	Axes via cross link		16#04000000	
MB_LOCAL_INTERPOL	Interpolation in the control		16#10000000	
MB_INDRADRIVE	IndraDrive	Always set	16#20000000	
MB_INDRADRIVE_1X	IndraDrive 2nd generation	IndraDrive 2nd generation	16#40000000	

Fig.5-219: Possible basic packages, function packages and options in ReqOptions and ActOptions

Required options should be specified in "ReqOptions". If at least one option is not available on the specified axis, a corresponding error message is generated. If all options requested in "ReqOptions" are available, "Done" becomes active. It is also possible to specify no required options and to perform the evaluation independently.

Checking the options requires several function block calls. During this period, the "Active" output is set.

System type	Description
IL_SYSTEM_UNKNOWN	Unknown system
IL_SYSTEM_MLC	IndraMotion MLC
IL_SYSTEM_XLC	IndraLogic XLC
IL_SYSTEM_MLD	IndraMotion MLD

Fig.5-220: Possible system types at the output SystemType

Error characteristics

If an option specified in "ReqOptions" is not available or if another error occurs, "Error" is set to TRUE.

Error codes

The function block generates the following error messages of the "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
ACCESS_ERROR	16#1503	16#0000	AxisData structure not activated
RESOURCE_ERROR	16#000F	16#0001	Required option TF not enabled
RESOURCE_ERROR	16#000F	16#0002	Required option ML not enabled
RESOURCE_ERROR	16#000F	16#0003	Required option MA not enabled
RESOURCE_ERROR	16#000F	16#0010	Required basic package OpenLoop not active
RESOURCE_ERROR	16#000F	16#0011	Required basic package CloseddLoop not active
RESOURCE_ERROR	16#000F	16#0020	Required function package Synchronization is not active

ML_TechBase.library

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#000F	16#0021	Required function package Servo is not active
RESOURCE_ERROR	16#000F	16#0022	Required functional package Main Spindle is not active

Fig.5-221: Error codes of the MB_GetSystemInfo function block

5.14 Function Block to Download Cams

5.14.1 Introduction

The following function block downloads cams to the drive and control to encapsulate the existing detail differences. The following "MB_CAMPROF" enumeration and the "MB_CAMTABLE" structure are used in the function blocks to create cam tables.

5.14.2 MB_LoadCamData

Brief Description

MB_LoadCamData downloads cams to the drive and control to encapsulate the existing detail differences.

Assignment: Target system/library

IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechBase.compiled-library

Fig.5-222: Reference table of the MB_LoadCamData

Interface Description

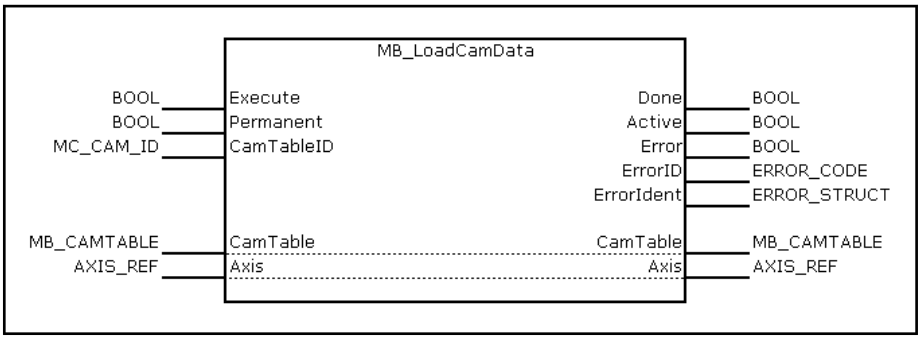


Fig.5-223: MB_LoadCamData function block

I/O type	Name	Type	Comment
VAR_IN_OUT	CamTable	MB_CAMTABLE	Structure with the calculated cam table elements and information on the number of elements
	Axis	AXIS_REF	Reference selection (drive or control)
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	Permanent	BOOL	Setting option whether the cam should be written permanently to the parameter or only temporarily to the volatile memory
	CamTableID	MC_CAM_ID	Selection of the desired cam table of the drive or control

I/O type	Name	Type	Comment
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid, error outputs valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, it contains a detailed error description (see "Error Handling" on page 156)

Fig.5-224: Interface variables of the MB_LoadCamData function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
Permanent	BOOL			FALSE	Rising edge at "Execute"
CamTableID	MC_CAM_ID			CAM_TABLE_1	Rising edge at "Execute"

Fig.5-225: Min./max. and default values of the MB_LoadCamData function block

The enumeration "MC_CAM_ID" supports the selection of the desired cam table.

Element	Value	Description
CAM_TABLE_1	1	Cam table 1
CAM_TABLE_2	2	Cam table 2
.....	.....	.....
CAM_TABLE_255	255	Cam table 255

Fig.5-226: Elements of the enumeration type MC_CAM_ID

The structure chapter 5.14.4 "MB_CAMTABLE" on page 157 is provided with the "CamTable" array containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

Functional Description

After processing enable via "Execute", the function block checks the type of format it can be written on the selected cam via "CamTableID".

The cam table in the "MB_CAMTABLE" structure is then converted to this format and loaded to the control or drive using a "WriteParameter" function block.

The selection of the memory location (control or drive) of the cam table is determined by the "Axis" input.

With the "Permanent" input, the decision can either be made for the permanent cam storage or the temporary storage (RAM) of the device. The different types of storages are implemented on the control by two different "WriteParameter" function blocks. The parameter S-0-0269 on the drive is responsible for the differentiation of the permanent and temporary storage.

The number of valid cam points included in the "CamTable" array can be seen via the "NumberOfElements" element of the "MB_CAMTABLE" structure. If a drive cam with less than 1,024 data points is selected via "CamTableID", but if "NumberOfElements" signalizes more elements, an error is generated.

The converted cam table is transferred to the "WriteParameter" function block in case of a correct function block processing and loaded to the control or drive. The validity of the conversion and the loading is signaled by the "Done" output.

If an error occurs while processing the function block, this is signaled by the "Error" output. The elements of the "CamTable" structure and "Axis" are not updated in case of error.

Cam characteristics

The cam table of the ML_LoadCamData function block provides the following characteristic:

- The algorithm to convert the cam data points is distributed across several cycles, i.e. calls of the instance to avoid a considerable effect on the PLC cycle time

	Cam in the control		Cam in the drive	
Axis	CntrlNo = LOCAL_CNTRL AxisNo = NO_OBJECT		CntrlNo = LOCAL_CNTRL AxisNo = AXIS_1..AXIS_99	
Permanent	FALSE (temporary cam)	TRUE (permanent cam)	FALSE (temporary cam)	TRUE (permanent cam)
CamTableID	1..99 (3-1,024 data points)		1..4 (3-1,024 data points); 5..8 (3-128 data points)	

Fig.5-227: Input assignments of the MB_LoadCamData function block to load the cam to the drive or control

Error Handling

The function block uses the F_RELATED_TABLE, 16#0170x error table. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#00000006	16#00000000	Invalid state of the function block
INPUT_RANGE_ERROR	16#00002200	16#00000001	The "NumberOfElements" input is lower than the minimum or higher than the maximum
INPUT_INVALID_ERROR	16#00002201	16#00000001	Address of the "CamTable" array is zero
INPUT_INVALID_ERROR	16#00002210	16#00000001	"CamTableID" input too high to select "Axis" input
INPUT_INVALID_ERROR	16#00002210	16#00000002	"NumberOfElements" input too high for selected "CamTableID"

Fig.5-228: Error codes of the MB_LoadCamData function block

5.14.3 MB_CAMPROF

Type definitions

The enumeration is used to select between different cam profiles. The selections of CAMPROF_STANDARD and CAMPROF_ADDITIVE contain the old

format at which the last element of the cam table is the data point φ = 360° - dφ. The selections of CAMPROF_STANDARD and CAMPROF_ADDITIVE contain the old format at which the last element of the cam table is the data point φ = 360° - dφ.

The format used can be selected for cam tables in the drive via the parameter P-0-0086. For cam tables in the control, only the new format is available.

Element	Value	Description
CAMPROF_STANDARD	0	Cam profile without reduction, cannot be segmented (old format)
CAMPROF_ADDITIVE	1	Superimposition profile with reduction, cannot be segmented (old format)
CAMPROF_ADDITIVE_SEGMENTABLE	2	Superimposition profile with reduction, can be segmented (new format)

Fig.5-229: Elements of the enumeration type MB_CAMPROF

5.14.4 MB_CAMTABLE

The structure is provided with the array "CamTable" containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

Name	Type	Min. value	Max. value	Default value	Effective	Description
CamTable	ARRAY [0...1023] OF LREAL				Rising edge at "Execute"	Array with the calculated cam points
NumberOfElements	UINT	3	1024		Rising edge at "Execute"	Number of valid cam points of the "CamTable" array

Fig.5-230: Min./max. and default values of the MB_CAMTABLE structure

6 ML_TechMotion.library

6.1 Overview on Function Blocks

The function blocks of the ML_TechMotion

Function block	Description
Function block of the "FlyingShear" application	
ML_FlyingShear and MX_FlyingShear	Contains the typical procedure of a "FlyingShear"
Function Blocks for Application "Tension Controller"	
MB_TensionControlLoadCellType01	Depending on the measured value of a load cell, the tension controller controls the tension to the desired command value
Function blocks for the "Sag Control" application	
ML_SagControlA	Contains a sag controller with analog actual value measurement for the sag
ML_SagControlC	Contains a sag controller with binary minimum and maximum contacts
Function blocks for cycle-synchronous lock-on and lock-off	
MB_CamLock	Used to enable the Run, LockOn and LockOff cam profiles calculated and stored by the MB_PrepareCams function block
MB_PrepareCams	Uses a polynomial 5th order to calculate and to generate 3 cam profiles and to load them into parameters specified by the inputs MC_CAM_ID. The resulting MotionProfiles contain boundary conditions for position and velocity
Function Blocks for general usage	
MB_DistanceAccumulator	Sums up the relatively traveled distance using an encoder and stops when the summed up distance is higher or equal to the **TotalDistance** input value. The current encoder position is entered using the **CurrentPosition** input
MB_AxisDistanceAccumulator	Sums up the relatively traveled distance using an encoder and stops when the summed up distance is higher or equal to the **TotalDistance** input value. This function block uses the axis reference input (AXIS_REF) to query the current encoder position automatically
MB_ReachVelocityType02	Calculates the respective jerk, acceleration and target velocity to travel along specified jerk-limited velocity characteristics (e.g. S-curve) and allows up to three changes in the target velocities
Function blocks for the "Position Monitoring" application	
MB_PositionMonitoring_GantryCant	Monitors the relative position of two axes with regard to the tilting to each other. The retraction motion is possible with the help of the tolerance band functionality where the direction of the motion out of the tilting is continuously monitored
MB_PositionMonitoring_PosCollision	Monitors the absolute position and the relative position of two axes with regard to indentation and collision
Function blocks for the implementation of a "feed machine"	
MB_SmartBeltType01	Executes a phase correction at a product using a triangular velocity profile
MB_MagicBeltType01	Contains functionalities to control a chain of a MagicBelt application

ML_TechMotion.library

Function block	Description
Function blocks to create cam tables for technology functions	
MB_PilgrimStepCalcType01	Calculates a cam table with up to 1,024 data points as well as the distances of the forward and backward transport as well as stop positions for the step-back operation.
MB_PrintLengthCorrCalcType01	Calculates a cam table with up to 1,024 data points for the technology function "print length correction" to change the length of a print on the printing material (web or shed)

Fig.6-1: ML_TechMotion.lib

6.2 Function Block for the "FlyingShear" Application Application

6.2.1 Introduction and Overview

With the application "FlyingShear", a machining station is synchronized with regard to a continuously passing material (like, for example, metal, plastic, etc.), and the machining is executed in the synchronous run of the machining station.

The material position is usually detected via a measuring wheel and transferred via an optional encoder input to the drive controller. Then, the drive controller synchronizes the connected servo motor on the continuously blending through material when the specified cut length was blending through. The function block chapter 6.2.2 "MX(L)_FlyingShear " on page 160 or chapter 6.2.3 "ML_FlyingShear - Special Features of IndraMotion MLC" on page 167 is responsible for the synchronization and sets the output "InSync" once the machining station is running synchronously. In this state, the material can be machined since now the machining station runs synchronously to the material.

After having finished the machining process, the synchronous run can be interrupted with the input "MoveReturn", and the machining station returns to its specified starting position.

For test purposes, a virtual axis can be used instead of the measuring wheel. With this virtual axis, it is then possible to simulate the measuring wheel.

6.2.2 MX(L)_FlyingShear

Brief Description

The function block MX(L)_FlyingShear contains the typical procedure of a FlyingShear and performs the following steps once the input "Start" was set to TRUE:

- Move the slave axis to the starting position "ReturnPos" and wait until the axis is in position
- Synchronize the slave axis on the master axis via a Lock On cam profile
- Set the output "InSync" once the slave axis (machining station) was synchronized on the material
- Return the slave axis to its starting position if the corresponding signal is set at the input "MoveReturn"

The function block operation required special boundary condition (e.g. interpolation in the drive, cyclic function block call in the Motion task...).

Please note the commissioning instructions chapter 6.2.3 "ML_FlyingShear - Special Features of IndraMotion MLC" on page 167

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_Technology07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.6-2: Reference table

Interface description

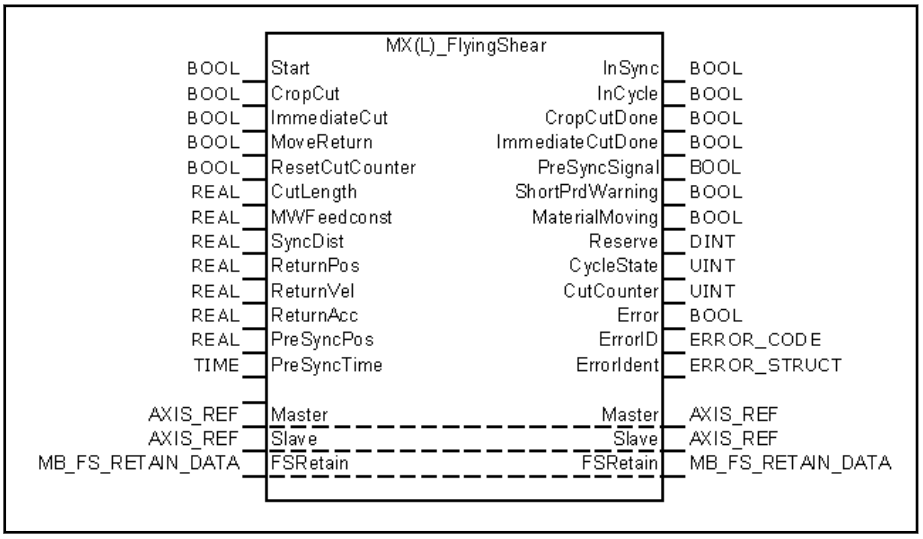


Fig.6-3: Function block MX(L)_FlyingShear

I/O type	Name	Data type	Explanation
VAR_IN_OUT	Master	AXIS_REF	Reference to the master axis
	Slave	AXIS_REF	Reference to the slave axis (machining station)
	FSRetain	MB_FS_RETAIN_DATA*[1]	Reference to the required retain data of this function block

ML_TechMotion.library

I/O type	Name	Data type	Explanation
VAR_INPUT	Start	BOOL	Start=TRUE: Starts the FlyingShear function block. Depending on the states of the inputs "ImmediateCut" and "CropCut", the function block performs the following functions: Start=TRUE and positive edge at the input "ImmediateCut": The machining station is synchronized immediately and performs the cutting length (Cutlength) once the immediate cut (ImmediateCut) has been performed. Start=TRUE and positive edge at the input "CropCut": The machining station is synchronized after the cutting length (Cutlength) has passed the machine and performs the cutting length (Cutlength) once the crop cut (CropCut) has been performed. Start=TRUE and no signal "ImmediateCut" and "CropCut": The machining station is synchronized after the cutting length (Cutlength) has passed the machine (measured from the last cut)
	CropCut	BOOL	At a rising edge, the slave axis starts the synchronization process after the cutting length has passed the machine. If a cut cycle is currently performed (InCycle=TRUE), the crop cut is performed in the next cut cycle
	ImmediateCut	BOOL	Start = TRUE: This case is intended for an immediate cut on the running material. The immediate cut (ImmediateCut) is activated by a positive edge. The slave axis is synchronized immediately. If a cut is currently performed (InCycle = TRUE), the immediate cut is performed in the next cut cycle. If (Start = FALSE) and ($V_{measuring\_wheel}$ < 5 rpm): This case is intended for an immediate cut material standstill. The immediate cut (ImmediateCut) is activated by a positive edge. The slave axis remains in its standstill position and is not synchronized. The output "InSync" is set immediately, and the function block expects the input signal "MoveReturn" to finish the process. After having finished this process, the FlyingShear is referenced to the material
	MoveReturn	BOOL	Completes the synchronous run of the slave axis with the material and returns the slave axis to its starting position "ReturnPos"
	ResetCutCounter	BOOL	The positive edge resets the cut counter
	Cutlength	REAL	Specified material cutting length[*2*3]

I/O type	Name	Data type	Explanation
	MWFeedconst	REAL	Feed constant of the measuring wheel per revolution *2 *3
	SyncDist	REAL	Synchronization distance required by the slave axis to synchronous the master axis*2*3. The shorter the synchronization distance, the higher the synchronization acceleration.
	ReturnPos	REAL	The slave axis travels to this starting position at the beginning and after the synchronization and the input MoveReturn is set to TRUE *2*3
	ReturnVel	REAL	The slave axis travels to the starting position "ReturnPos" with the velocity ReturnVel *2*3
	ReturnAcc	REAL	The slave axis returns to the starting position "ReturnPos" with the acceleration ReturnAcc *2*3
	PreSyncPos	REAL	The output signal "PreSyncSignal" is set before the synchronous position "PreSyncPos" with the distance given prepare before the synchronous run*2
	PreSyncTime	TIME	Duration of "PreSyncSignal"
VAR_OUTPUT	InSync	BOOL	The slave axis runs synchronously with the material. It is in its synchronization window (see drive parameter S-0-0228)
	InCycle	BOOL	The slave axis currently performs a cut cycle. An immediate or crop cut is performed in the next cycle (CutCycle = 0)
	CropCutDone	BOOL	The crop cut has been performed
	ImmediateCutDone	BOOL	The immediate cut has been performed
	PreSyncSignal	BOOL	First marking before the synchronous position "PreSyncPos"
	ShortPrdWarning	BOOL	The slave axis does not have enough time to reach the starting position "ReturnPos" for the next cut. The return is interrupted and the synchronization ramp starts again. This case should be prevented since the drive might be overloaded and the machining station might move to the carriage bed end. Countermeasures: • Increase return velocity (function block input "ReturnVel") • Increase return acceleration (function block input "ReturnAcc") • Reduce synchronization distance (function block input "SyncDist") • Reduce material feed velocity
	MaterialMoving	BOOL	The speed of the material encoder is higher than 5 rpm. An immediate cut in the case of a material standstill is only possible if MaterialMoving=FALSE

ML_TechMotion.library

I/O type	Name	Data type	Explanation
	Reserve	DINT	Reserved increments of the synchronization procedure for analysis purposes. A synchronization is not possible if Reserve ≤ 0 -> In this case, the function block returns an error message
	CycleState	UINT	Current cut cycle state: 0: Standstill and waiting phase 1: Acceleration phase 2: Synchronous run phase 3: Return phase
	CutCounter	UINT	With each cut, the cut counter "CutCounter" is increased. With (Start = FALSE) or a positive edge at the input "ResetCutCounter", the counter is reset
	Error	BOOL	Indicates an error. Cleared if Start = FALSE
	ErrorID	INT (Enum)	ERROR_CODE: Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

*1: STRUCT (bCutNotCompleted: BOOL, diMasterSyncPosition: DINT, iRevCounter:INT)

*2: Measuring units according to drive scaling, e. g. mm

*3: New input values are transferred with the transition from the synchronous run phase to the return phase. Due to this reason, a new cutting length which is to be activated in the next cycle has to be copied to the function block input before the axis is moved to its return position.

Fig.6-4: Interface variables of the function block MX(L)_FlyingShear

Signal-time diagram

The following diagram shows the signal flow with immediate cut and stopped material (Start = FALSE):

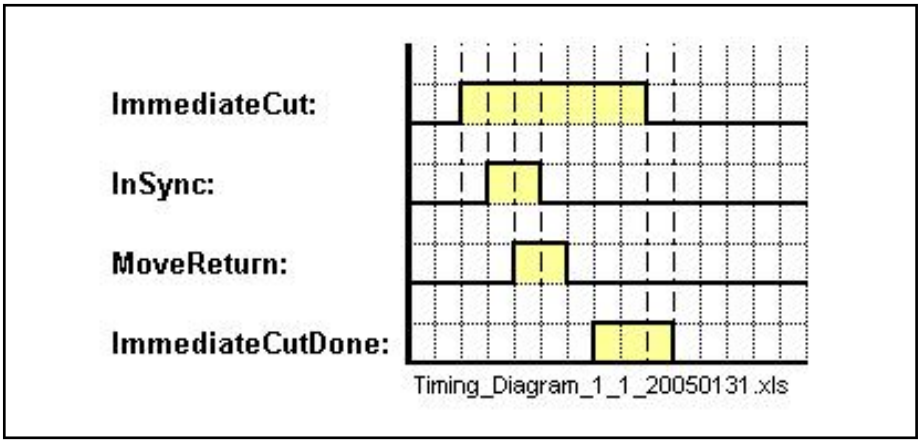


Fig.6-5: Signal-time diagram: Immediate cut at material standstill

The following chart shows the complete signal flow of the function block "FlyingShear" with continuously passing material and immediate cut:

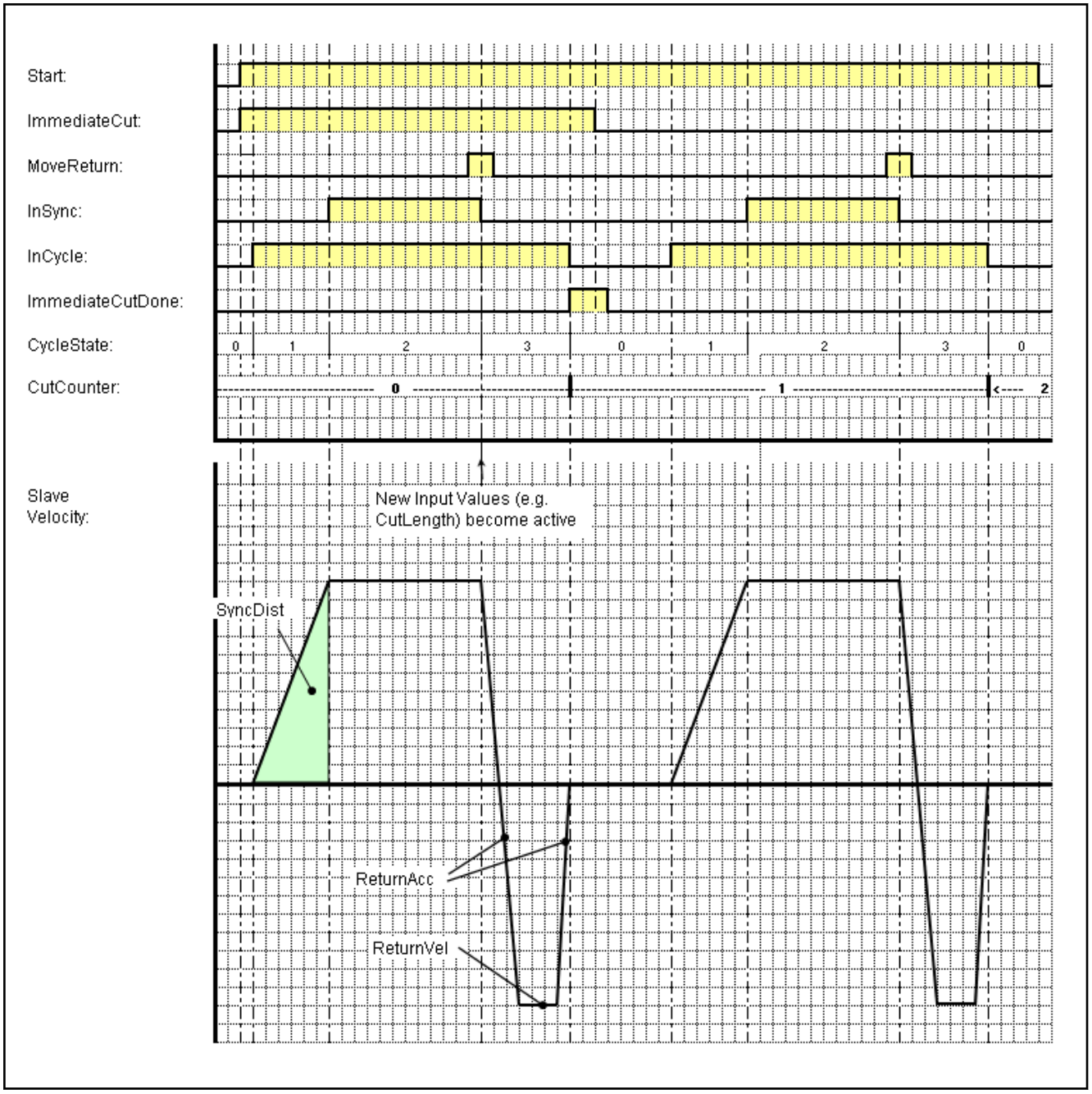


Fig.6-6: Complete signal flow of the function block "FlyingShear" with immediate cut

Error handling

The function block "MX(L)_FlyingShear" generates the following error messages in Additional1/Additional2 for the table "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive not enabled or drive error
ACCESS_ERROR (16#0004)	16#0003	16#0000	The function block was aborted by another function block
ACCESS_ERROR (16#0004)	16#0004	16#0000	Not supported drive firmware

ML_TechMotion.library

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis (Axis_Ref) was changed during the function block processing
INPUT_RANGE_ERROR(16#0006)	16#0201	16#0001	CutLength <= 0
INPUT_RANGE_ERROR(16#0006)	16#0201	16#0002	MWFeedconst <= 0
INPUT_RANGE_ERROR(16#0006)	16#0201	16#0003	SyncDist <= 0
INPUT_RANGE_ERROR(16#0006)	16#0201	16#0004	ReturnAcc <= 0
INPUT_RANGE_ERROR(16#0006)	16#0201	16#0005	ReturnVel <= 0
SYSTEM_ERROR (16#7FFF)	16#0202	16#0000	Synchronous position is too far away. Thus, synchronization is not possible. Reset error and initiate immediate cut
ACCESS_ERROR (16#0004)	16#0203	16#0001	S-0-0048 is not configured in the optional cyclical MDT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0002	P-0-0054 is not configured in the optional cyclical MDT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0003	P-0-0691 is not configured in the optional cyclical MDT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0004	P-0-0692 is not configured in the optional cyclical MDT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0005	P-0-0052 is not configured in the optional cyclical AT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0006	P-0-0053 is not configured in the optional cyclical AT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0007	P-0-0761 is not configured in the optional cyclical AT channel of the slave axis
ACCESS_ERROR (16#0004)	16#0203	16#0008	P-0-1367 bit 6 is not TRUE (the structure AxisData is not updated).
OTHER_ERROR (16#00FE)	16#0204	16#0001	The ELS configuration word (P-0-0086) of the slave axis has been configured incorrectly. The configuration has to be: P-0-0086 = 2#xxxx01000
OTHER_ERROR (16#00FE)	16#0204	16#0002	The parameter P-0-0060 of the slave axis has been configured incorrectly. The value has to be: P-0-0060=0

ErrorID	Additional1	Additional2	Description
OTHER_ERROR (16#00FE)	16#0204	16#0003	The parameter P-0-0693 of the slave axis has been configured incorrectly. The value has to be: P-0-0693=0
OTHER_ERROR (16#00FE)	16#0204	16#0004	The parameter P-0-0750 of the slave axis has been configured incorrectly. The value has to be: P-0-0750=0
OTHER_ERROR (16#00FE)	16#0204	16#0005	The parameter P-0-0090 of the slave axis has been configured incorrectly. Bit 2 of P-0-0090 has to be FALSE
SYSTEM_ERROR (16#7FFF)	16#0205	16#0000	Immediate cut (in standstill) was requested although the material was moving
OTHER_ERROR (16#00FE)	16#0206	16#0000	The parameter A-0-0200 of the slave axis does not correspond to the logical axis number of the master axis.

Fig.6-7: Error codes of the MX(L)_FlyingShear function block

6.2.3 ML_FlyingShear - Special Features of IndraMotion MLC

Required hardware

- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive C or M
- Additional encoder interface card to read in the measuring wheel position if a measuring encoder is to be used (can be omitted if a virtual or real axis is used)
- Additional encoder to read in the measuring wheel position (according to IndraDrive planning) if a measuring encoder is to be used (can be omitted if a virtual or real axis is used)

☞ It is recommended to use a high-resolution measuring encoder with a resolution of at least 4096 increments/revolution (possibly sinus encoder). Roughly resolved encoder signals cause low cut accuracies as well as running noises in the synchronous run.

Required firmware

- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- Drive firmware MPx07
- The following function packages have to be enabled in the drive:
  – Closed Loop
  – Synchronization

Required software

IndraWorks 12VRS for the IndraMotion MLC

Required parameterization

The following parameterization has to be accomplished in the drive to ensure a proper functionality of the function block "MX(L)_FlyingShear".

1. Create the slave axis in IndraWorks

The function block requires the command value generation in the drive. Thus, when creating a slave axis in IndraWorks, the checkbox "Interpolation in the drive" may not be deselected.

2. Loading the parameter file "FlyingShearSettings.par"

   By loading the parameter file "FlyingShearSettings.par", important configurations as well as the cams are automatically loaded on the FlyingShear axis. The parameter file is provided via the IndraWorks MLC installation CD 3 in the path "Add ons\ Tech FB_Parameterfile".

   The parameter file contains the following information:
   - P-72 Lock On Cam Poly 5
   - P-92 Run Cam
   - P-750=1
   - P-693=0
   - P-60=0
   - P-86=0x0000.0000.0000.1000
   - P-90=0x0000.0000.0000.0000
   - A-500=P-691
   - A-502=P-692

3. Parameterization of the default master axis

   Enter the logical axis number of the master axis into the parameter A-0-0200 of the FlyingShear slave axis

   Write on the parameter A-0-0200 of the slave axis with the logical axis number of the master axis using the IndraWorks parameter editor.

4. Parameterization of the FlyingShear axis

   Parameterization of the FlyingShear axis according to the application.

   Scaling should be linear and absolute.

5. Parameterization of the optional encoder

   IndraWorks ▸ Add an encoder axis to the project

   Scaling should be set to rotary modulo 360°.

   Setting the encoder parameters according to encoder data

6. Ensuring the encoder motion direction

   In normal operation, the optional encoder (measuring wheel) has to move in the positive direction. The motion direction can be observed via the IndraWorks dialog of the encoder axis. The motion sense has to be inverted if the encoder moves in the negative direction.

7. Parameterize the synchronization acceleration (P-0-0142) and synchronization velocity (P-0-0143) with preferably high values to ensure a dynamic synchronization process

   IndraWorks ▸ Parameter Editor.

8. Parameterize the synchronous run window (S-0-0228) preferably small to obtain a possibly high cut accuracy

   IndraWorks ▸ Parameter Editor.

Required IndraLogic steps

- The function block "MX(L)_FlyingShear" contains high-prior Motion commands and should therefore be cyclically called in a high-prior, cyclical PLC task with a cycle time ≤ 4ms (MotionTask)
- Before starting the function block, "MC_Power" has to be processed without errors

6.3 Function Blocks for Application "Tension Controller"

6.3.1 Introduction and Overview

Depending on the measured value of a load cell, the tension controller (chapter 6.3.2 " MB_TensionControlLoadCellType01" on page 169) controls the tension to the desired command value. The actual value has to be provided to the controller and is normally measured via an analog or field bus input.

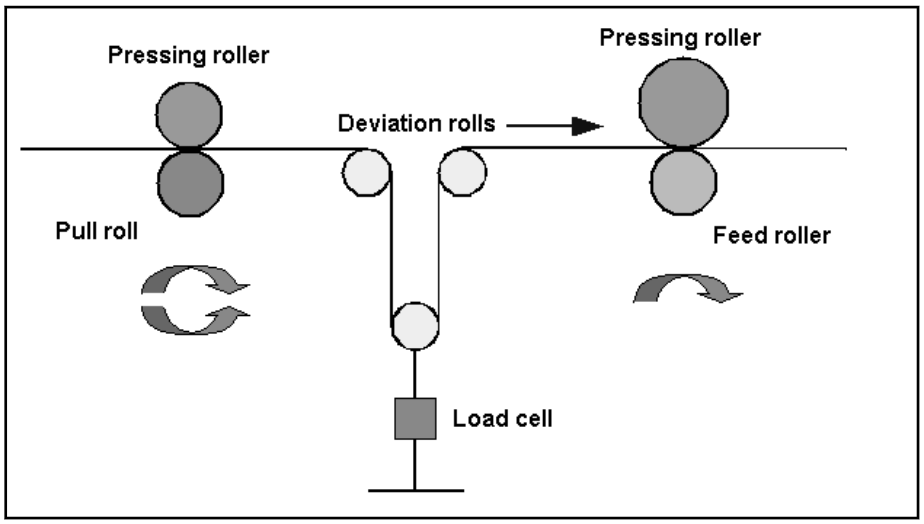


Fig.6-8: Schematic diagram for the closed-loop tensile stress control

The tension controller is designed as a PI controller (optionally adaptive). The controller output affects the gear fine adjustment of the slave axis (pull roll). In addition, the controller can hold the tension at standstill or tension the web.

The slave axis can either be operated in the velocity-synchronous or in the phase-synchronous operation mode.

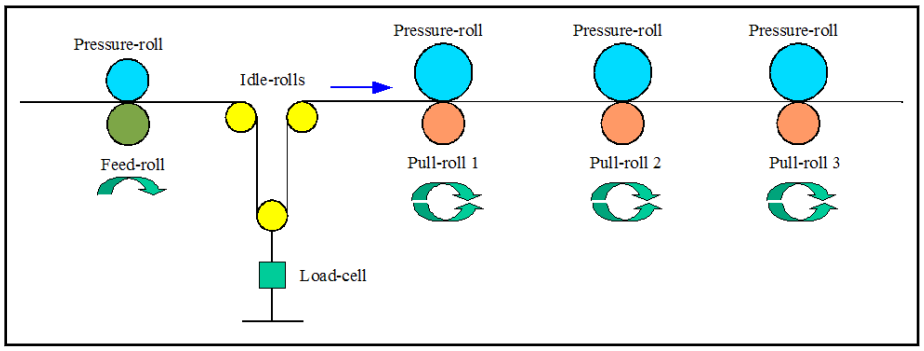


Fig.6-9: Multi-axis tension controller

6.3.2 MB_TensionControlLoadCellType01

Brief Description

Depending on the measured value of a load cell, the tension controller regulates the tension to the desired command value.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib

ML_TechMotion.library

IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.6-10: Reference table

☞ IndraMotion MLC 12VRS or higher also supports slave axes with interpolation on the control.

Interface description of the

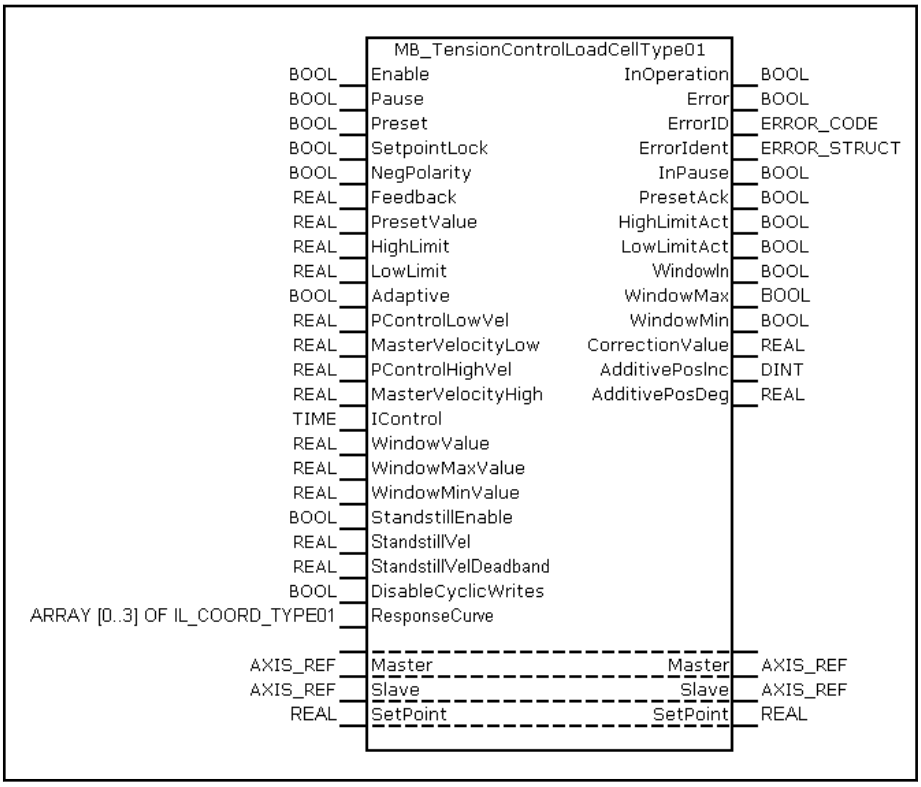


Fig.6-11: MB_TensionControlLoadCellType01 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Master	AXIS_REF	Reference to the master axis
	Slave	AXIS_REF	Reference to the slave axis
	Setpoint	REAL	Command value of the tension This is generally set by the user but can also be changed by the function block (see "SetpointLock")
VAR_INPUT	Enable	BOOL	If this input is set, the function block operates
	Pause	BOOL	If this input is set, the tension controller is paused and the I-value of the output value is frozen. The "InPause" output signalizes an active "Pause"
	Preset	BOOL	At the rising edge of this input, the control output is set to the value parameterized in "PresetValue" and if successful, the "PresetAck" output is set

I/O type	Name	Data type	Description
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly to the optional cyclic data container, but only displayed at the output. The value is only evaluated at a rising edge at "Enable"
	SetpointLock	BOOL	At a rising edge at the input, the current actual tension value at "Feedback" is accepted as new command value for the tension controller
	NegPolarity	BOOL	If this input is set, the control variable "CorrectionValue" and "AdditivePosInc" are provided with a negative sign
	Feedback	REAL	Current actual tension
	PresetValue	REAL	The controller output is set to this value if the "Preset" input is set
	HighLimit	REAL	Maximum value of the controller output. If the control variable calculated by the controller is greater than this value, the control variable is limited and the I-value of the controller is not summed up anymore
	LowLimit	REAL	Minimum value of the controller output. If the control variable calculated by the controller is lower than this value, the control variable is limited and the I-value of the controller is not summed up anymore
	Adaptive	BOOL	If this input is set, an adaptive P-gain is used for the closed-loop control
	PControlLowVel	REAL	P-gain of the controller if "Adaptive" = FALSE Otherwise, the value corresponds to the P-gain at low master axis velocities
	MasterVelocityLow	REAL	Only effective if "Adaptive" = TRUE. Master axis velocity up to which "PControlLowVel" is used
	PControlHighVel	REAL	P-gain of the controller at high master axis velocities. If "Adaptive" = FALSE, this input has no effect
	MasterVelocityHigh	REAL	Only effective if "Adaptive" = TRUE. Master axis velocity from which "PControlHighVel" is used
	IControl	TIME	Integral action time of the I-controller If the value is set to 0s, the I-controller is not effective
	WindowValue	REAL	Process variable window
	WindowMaxValue	REAL	Maximum value of the process variable
	WindowMinValue	REAL	Minimum value of the process variable
	StandstillEnable	BOOL	If this input is set, the additive velocity specified in "StandstillVel" is effective
	StandstillVel	REAL	Additive velocity for standstill control

ML_TechMotion.library

I/O type	Name	Data type	Description
	StandstillVelDeadband	REAL	This value determines the controller deviation value. Below this value, the tension velocity for the standstill control is reduced
VAR_OUTPUT	InOperation	BOOL	The tension controller operates and the outputs are valid
	Error	BOOL	Indicates an error If "Enable" = FALSE, the error is cleared
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	For a detailed error description according to the error table (see Error handling on page 180).
	InPause	BOOL	"Pause" is active
	PresetAck	BOOL	If this output is active, the "Preset" input was analyzed and the "CorrectionValue" output assumed the "PresetValue" value
	HighLimitAct	BOOL	This output is set if the control variable calculated by the controller is greater than the value specified in "HighLimit"
	LowLimitAct	BOOL	This output is set if the control variable calculated by the controller is lower than the value specified in "LowLimit"
	WindowIn	BOOL	If TRUE is set, the process variable is in the parameterized window
	WindowMax	BOOL	If TRUE is set, the process variable exceeded the parameterized maximum value
	WindowMin	BOOL	If TRUE is set, the process variable is below the parameterized minimum value
	CorrectionValue	REAL	Control variable for the tension controller
	AdditivePosInc	DINT	Additive velocity in increments per SERCOS cycle

Fig.6-12: Interface variables of the MB_TensionControlLoadCellType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
SetPointLock	BOOL			FALSE	Continuous
NegPolarity	BOOL			FALSE	Continuous
Feedback	REAL	n.def.	n.def.	0.0	Continuous
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"

Name	Type	Min. value	Max. value	Default value	Effective
HighLimit	REAL	>LowLimit	n.def.	200.0	Continuous
LowLimit	REAL	-95.0	<HighLimit	-50	Continuous
Adaptive	BOOL			FALSE	Continuous
PControlLowVel	REAL	>0.0	n.def.	1.0	Continuous
MasterVelocity-Low	REAL	>0.0	<=MasterVelocity-High	1.0	Continuous
PControlHighVel	REAL	>0.0	n.def.	1.0	Continuous
MasterVelocity-High	REAL	>=MasterVelocity-Low	n.def.	1.0	Continuous
IControl	TIME	0 s	n.def.	0 s	Continuous
WindowValue	REAL	0.0	100.0	0.0	Continuous
WindowMaxValue	REAL	>=WindowMinVal-ue	100.0	0.0	Continuous
WindowMinValue	REAL	0.0	<=WindowMax-Value	0.0	Continuous
StandstillEnable	BOOL			FALSE	Continuous
StandstillVel	REAL	0.0	n.def.	0.0	Continuous
StandstillDead-band	REAL	0.0	n.def.	1.0	Continuous

Fig.6-13: Input behavior of the MB_TensionControlLoadCellType01

Functional Description

The following explains the individual functions of the function block inputs in detail. In general, all outputs are continuously analyzed if the "Enable" input is set. That means that changes to the inputs become immediately effective without toggling the "Enable" input again.

An exception is the "PresetValue" input which is only accepted at a rising edge at the "Preset" input. Likewise, the "DisableCyclicWrites" input is only adopted at a rising edge at the "Enable" input.

The complete mode of operation of the tension controller is shown in the following flowcharts. Depending on the "StandstillEnable" input, either the "In-Velocity" path or "Standstill Control" is effective.

ML_TechMotion.library

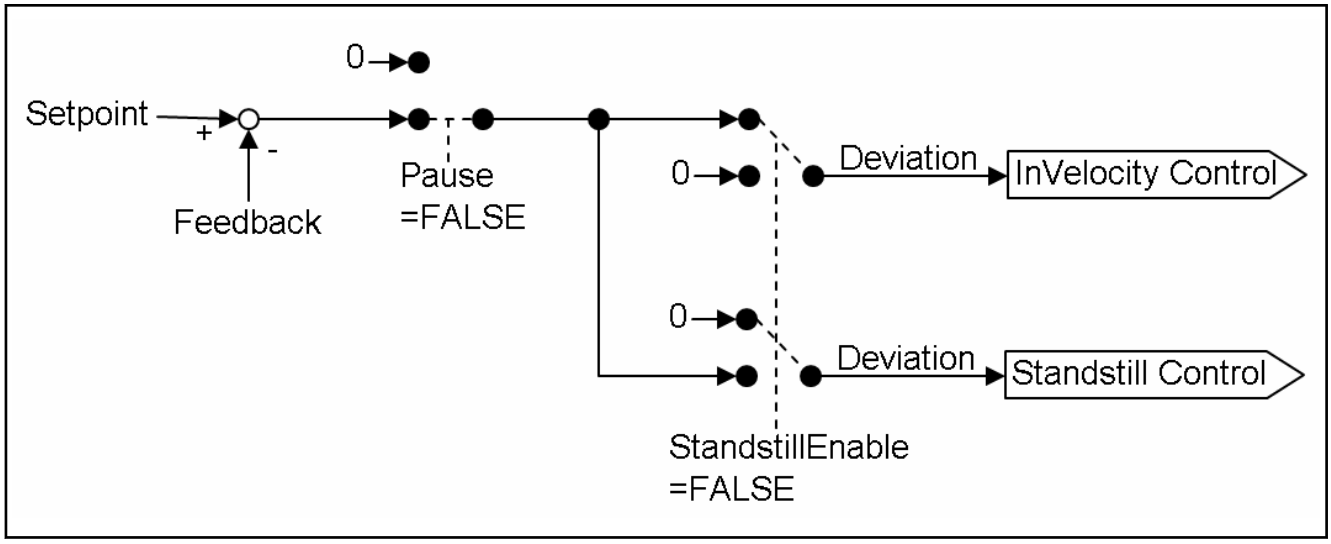


Fig.6-14: Flowchart of the tension controller with branch for the control at standstill/velocity

The "InVelocity Control" path is effective if the "StandstillEnable" input is not set. The tension is controlled via the parameter "Gear ratio fine adjustment" P-0-0694 (slave axis with interpolation in the drive/IPO Drive) or A-0-2605 (slave axis with interpolation on the control/IPO Control). It is directly written on the parameters if the "DisableCyclicWrites" input is not set. For real axes with interpolation in the drive, the configuration of the parameter P-0-0694 in the optional cyclic SERCOS data (MDT) is thus required.

The control variable is additionally output via the "CorrectionValue" output.

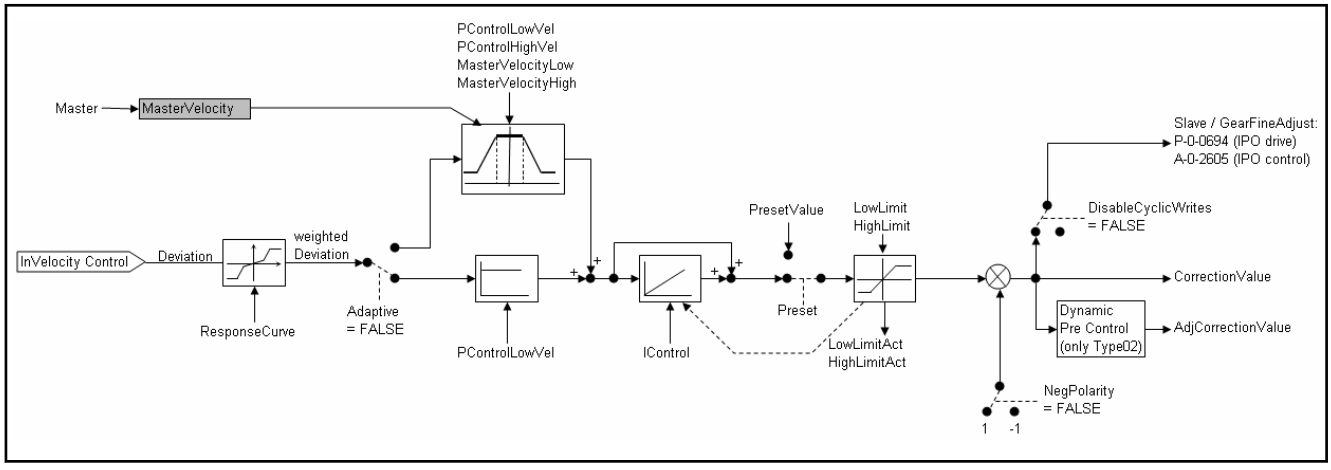


Fig.6-15: Flowchart of the tension controller for the "InVelocity Control" branch

For the standstill control, another path can be activated to which the additive master axis position (P-0-0692) can be written as long as "DisableCyclic-Writes" is inactive. At the same time, the value is always available via the "AdditivePosInc" output.

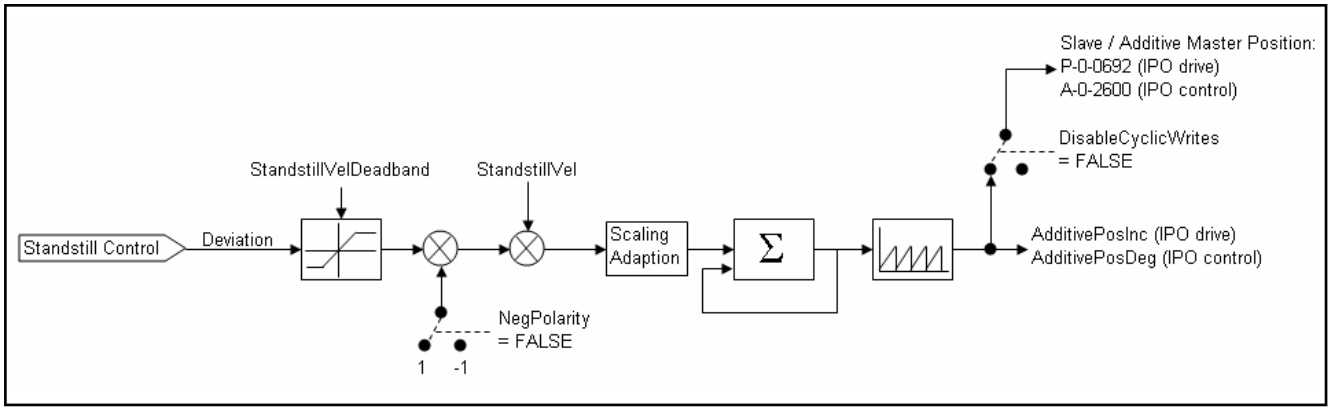


Fig.6-16: Flowchart of the tension controller for the closed-loop control at standstill (Standstill Control)

To ensure user-friendliness, the flowchart does not contain the "WindowValue", "WindowMaxValue" or "WindowMinValue" limitations and their status displays.

Commissioning

If "DisableCyclicWrites" is inactive, both the P-0-0694 (Gear ratio fine adjustment) and P-0-0692 (Additive master axis position) parameters have to be applied in the cyclic command value channel of the slave axis. At a rising edge at "Enable", the cyclic data channels are checked. If one of the two parameters is not found, a respective error message is returned at the error outputs of the function block. If "DisableCyclicWrites" is enabled, the configuration of the cyclic command value data channels is not checked.

The axes used have to be in synchronous operation mode before the "Enable" input is activated. This can be executed by the PLCopen function blocks "GearIn" or "GearInPos".

If the tension controller is used in a drive-integrated IndraMotion MLD, the AxisData structure has to be enabled in the drive.

Preset/PresetValid

A positive edge at the "Preset" input specifies an initial value for the control variable using the "PresetValue" variable. At the same time, the internal variables of the controller are set in a way that the controller is approached without any jerk. "Preset" acts only on the gear fine adjustment in the path "InVelocity Control".

The "Preset" input is only evaluated when "Enable" is TRUE. At a rising edge at "Preset", the new value is accepted at "PresetValue" and "PresetAck" is set. A falling edge at "Preset" or "Enable" deletes "PresetAck". The "Preset" inputs dominates "Pause". That means that the value at "PresetValue" is applied at the output even though "Pause" is set. If "Preset" is active at the rising edge at "Enable", "PresetValue" is immediately assumed as control variable.

Execute the "Preset" function standstill (StandstillEnable=TRUE). Otherwise, velocity jumps can result due to the setting of the ControlValue to the PresetValue. The value of the jump is proportional to the velocity.

The "Preset" input is only effective at the rising edge.

DisableCyclicWrites

The "DisableCyclicWrites" input determines whether the values calculated by the function block are to be written directly via the cyclic channel (FALSE) or whether they are only available at the function block output (TRUE).

Thus, it is possible to cascade several function blocks if "DisableCyclicWrites" is set to TRUE. Then, only the outputs ("CorrectionValue" and "AdditivePosInc") can be used and modified later on. However, the user still has to write the values into the optional cyclic channel.

If "DisableCyclicWrites" is set to TRUE, it is not checked whether the optional cyclic parameters are also configured.

Adaptive, PControlLowVel, PControlHighVel, MasterVelocityLow, MasterVelocityHigh

The P-gain can either be specified as constant P-gain or as adaptive P-gain. Via the "Adaptive" input, either an adaptive or a constant P-gain is selected. Setting "Adaptive" to TRUE selects the adaptive P-gain.

If a constant P-gain is selected, only the value in "PControlLowVel" affects the P-gain.

If an adaptive P-gain is selected, the "PControlLowVel", "PControlHighVel", "MasterVelocityLow" and "MasterVelocityHigh" inputs have to be assigned. The characteristic curve looks as follows:

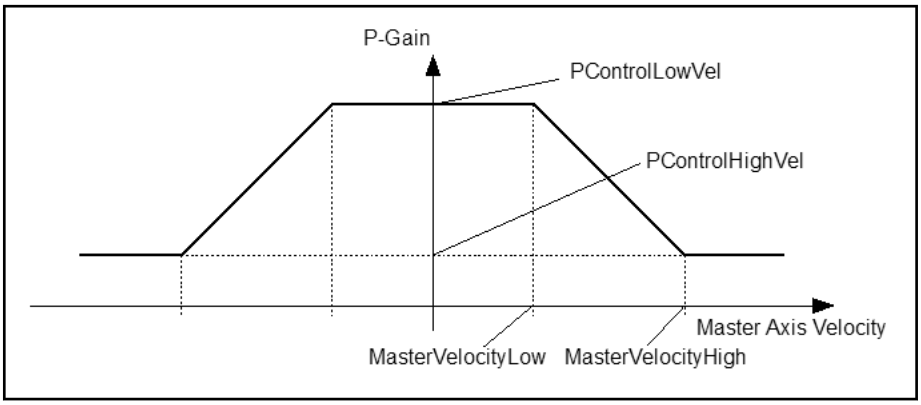


Fig.6-17: Characteristic curve for an adaptive P-gain

The individual inputs describe the characteristic curve for positive master axis velocities. The characteristic curve for negative master axis velocities results from its mirror image on the axis of the P-gain. This means that only symmetric characteristics are possible. The P-gain between the master axis velocities "MasterVelocityLow" and "MasterVelocityHigh" is linearly interpolated. This results in some conditions for the value ranges of the inputs:

- The values for "MasterVelocityLow" and "MasterVelocityHigh" have to be positive
- The values for "MasterVelocityLow" and "MasterVelocityHigh" may not be identical
- The "MasterVelocityLow" value has to be lower than the "MasterVelocityHigh" value

If one of these conditions is not met, an error is generated at the output of the function block.

SetpointLock

Setting the "SetpointLock" input applies the current actual tension value as the new command value for the tension controller. For example, the function allows the command value of the tension to obtain a reasonable initial value that reflects the current state of the machine.

Pause

The "Pause" input enables the tension controller to pause. If this input is set to TRUE, the control deviation is set to "0", i.e. the control output is frozen at the current I-value. If "Pause" is active, the "InPause" output is set.

NegPolarity

The "NegPolarity" input affects the control direction. Depending on the position of the pull roll with regard to the blending direction of the material and the position of the feed roll, the control direction has to be either positive or nega-

tive. If this input is FALSE, the control direction is positive and a positive control deviation results in a positive control motion. The input also affects the standstill control.

Feedback This input represents the actual value of the tension. It can be read via an analog input at the drive or at the Inline I/O for example. The user is responsible for a reasonable scaling.

PresetValue Setting the "Preset" input specifies an initial value for the control variable using the "PresetValue" variable. At the same time, the internal variables of the controller are set in a way that the controller is approached without any jerk.

HighLimit, HighLimitAct This input limits the control variable of the controller to a maximum value. If the limitation becomes effective, the I-value of the controller is no longer summed up ("Anti-reset-windup"). The maximum value is determined by the maximum value for the gear ratio fine adjustment defined in the drive. If a limitation becomes effective, the "HighLimitAct" output is set.

LowLimit, LowLimitAct This input limits the control variable of the controller to a minimum value. If the limitation becomes effective, the I-value of the controller is no longer summed up ("Anti-reset-windup"). The minimum value is -95 irrespective of possible lower values in the drive (also refer to section "Gear ratio fine adjustment" on page 178). If a limitation becomes effective, the "LowLimitAct" output is set.

PControlLowVel This value corresponds to the proportional gain of the tension controller as long as the "Adaptive" input is not active. Otherwise, it represents the lower gain of an adaptive P-gain.

WindowValue, WindowIn With this value, the user can define a symmetric window containing the current actual tension value. The window is defined as follows:

$$\left(SetPoint - |SetPoint| \cdot \frac{WindowValue}{100\%}\right) \leq Feedback \leq \left(SetPoint + |SetPoint| \cdot \frac{WindowValue}{100\%}\right)$$

Fig.6-18: WindowValue, WindowIn

If this condition is met, the "WindowIn" output is set. If this condition is not met, the output is not set and no limits become effective.

WindowMaxValue, WindowMax This value can be used to set the upper value for the actual tension value. The value is evaluated as follows:

$$Feedback > \left(SetPoint + |SetPoint| \cdot \frac{WindowMaxValue - 100}{100}\right)$$

Fig.6-19: WindowMaxValue, WindowMax

If this condition is met, the "WindowMax" output is set. If this condition is not met, the output is not set and no limits become effective.

WindowMinValue, WindowMin This value can be used to specify a lower value for the actual tension value. The value is evaluated as follows:

$$Feedback < \left(SetPoint + |SetPoint| \cdot \frac{WindowMinValue - 100}{100}\right)$$

Fig.6-20: WindowMinValue, WindowMin

If this condition is met, the "WindowMin" output is set. If this condition is not met, the output is not set and no limits become effective.

StandstillEnable

By setting the "StandstillEnable" input, the velocity specified in "StandstillVel" is additively added to the axis to be controlled. This allows a closed-loop tension control at standstill as well.

☞ The lower the velocity, the lower the influence of the gear ratio fine adjustment. There is even an open control loop at master axis standstill, since the gear ratio fine adjustment has lost all its influence. Thus, the "StandstillEnable" input has to be set when reaching the standstill window in order not to increase the I-value up to the limit in the "InVelocity Control" path by the open control loop.

☞ For axes with interpolation on the control, the "AdditivePosInc" output is given in modulo format 2^{20} (12VRS or higher).

☞ If this input is used, it must be ensured that the tension controller is only used in the SERCOS synchronous task. If this is not the case, the motor cannot run smoothly or oscillate around the standstill position.

StandstillVel

This value indicates the additive velocity sent to the drive when the "StandstillEnable" input is set. The units used for the velocity correspond to the scaling set in the drive.

☞ At standstill, the velocity is generated in jumps. Since the velocity has to be reached within one SERCOS cycle, the values specified cannot be indefinitely high. If the value selected for the standstill velocity is too high, oscillation of the drive can be caused. The maximum value depends on the scaling of the velocity, the maximum acceleration set and the mass connected to the drive. To avoid big velocity jumps, a filter time constant can be defined in the drive parameter P-0-0693. This smoothes the jump.

Background information

Gear ratio fine adjustment

The tension controller generally operates with velocities greater than 0. The gear ratio fine adjustment is an appropriate method to keep the tension. However, there are also some boundary conditions to be explained.

The gear ratio fine adjustment is internally limited to a minimum of -95%. If a lower value is entered, the function block outputs an error.

The reason for this limitation is the calculated output value defined as follows for a given input value:

$$Output\,value = Input\,value * \left(1 + \frac{Gear\ fine\ adjust}{100}\right)$$

Based on the equation, it can be seen that when the gear ratio fine adjustment approaches 100%, the output value approaches 0. Thus, the control loop practically opens since a modification of the input value does not cause any response at the output. It is even worse at a gear ratio fine adjustment of less than -100% because the sign of the output value reverses. In the control loop, positive feedback would occur that can result in an unstable system.

StandstillEnable, StandstillVel, StandstillVelDeadband

The standstill control can be executed using the "StandstillEnable" and "StandstillVel" inputs. This adds an additive velocity. To achieve better control quality, it is reduced by the command value. This can be executed with a modified signum function provided with the following transfer function:

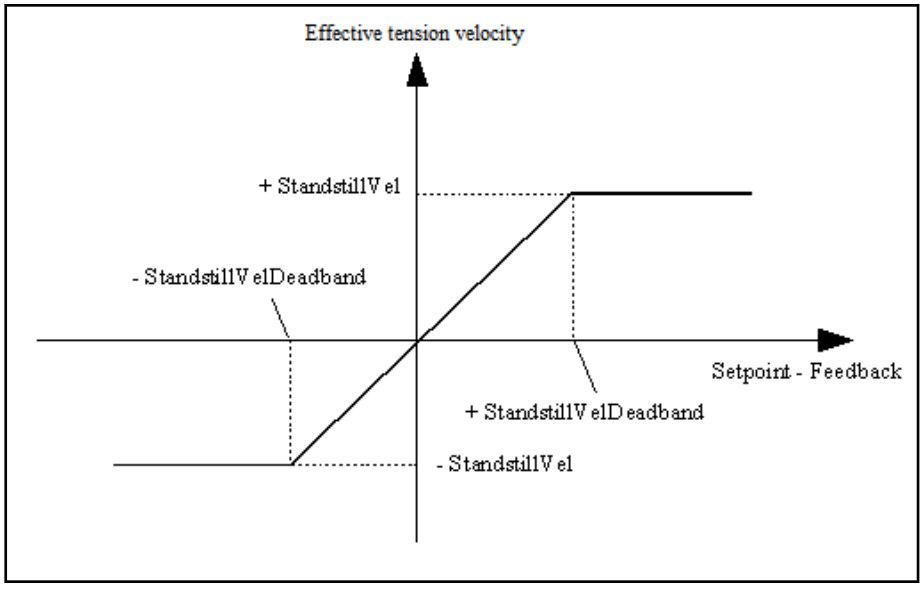


Fig.6-21: Modified signum function to limit the added "StandstillVel"

The control deviation is relativized in the "Setpoint" range. If the "Feedback" approaches the "Setpoint", the closed-loop control becomes "more sensitive" than a simple two-position controller. The value responsible for the limitation can be set at the "StandstillVelDeadband" input. The values are provided with the same units as the "Setpoint" input or the "Feedback" input. Linear interpolation occurs between the two extreme values +"StandstillVel" and -"StandstillVel" so that the transfer function above results for control deviation and tension velocity.

Then, the factor determined is multiplied with the specified "StandstillVel". The velocity is processed as speed with the scaling set in the drive. If the web velocity should be entered instead of the speed, it is to be calculated before.

To reverse the direction of motion for the standstill control, set the "NegPolarity" input.

Subsequently, the result is multiplied by a correction factor. This correction factor results from the gear ratio. However, the gear ratio fine adjustment is not taken into consideration. In practice, the gear ratio fine adjustment is generally very small and is therefore not important. The gear ratio is only accepted at a rising edge at the "Enable" input.

The flow chart shows that the velocity specified is converted into increments. A few rules are to be considered:

- The specified velocity cannot be unlimited because the drive has to be able to reach the velocity within one SERCOS cycle. Thus, only very small values are allowed. The value size depends on the scaling used and the mass connected to the motor. The increments calculated are specified using parameter P-0-0692. This parameter can also be smoothed using the filter (P-0-0693) available in the drive allow larger jumps as well.
- The increments calculated are added before the gear. The gear ratio (A-0-2720 and A-0-2721 or P-0-0156 and P-0-0157) is included in the calculation as well. The gear ratio fine adjustment (P-0-0083 and P-0-0694) is not taken into consideration because it is generally very small. This can result in the axis moving at a different velocity than the velocity set.

- The units are scaled. In addition, the velocity value entered is not checked against the maximum and minimum values. This can result in very large values for the number of increments per SERCOS cycle. If the calculated interim value exceeds the maximum values by ± half of a modulo value, it is automatically limited. This results in a less additive velocity than specified at the "StandstillVel" input.
- At very low velocities, it also has to be considered that the current velocity can deviate from the specified velocity. This can be caused due to the increments granulated per SERCOS cycle: e.g. if 1 increment corresponds to a velocity of 0.03 rpm, only velocities that are multiples of 0.03 rpm can be set. If this is not the case, it is rounded up to the next value.
- The calculated increments are processed with each SERCOS cycle. Thus, when using the standstill control, the function block has to be processed in the SERCOS synchronous task. If this is not the case, the drive moves one SERCOS cycle around the calculated number of increments and in the following SERCOS cycles it comes again to a standstill which results in a jerky motion. The filter in P-0-0693 can also be helpful.

Error handling The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**, 16#0170:

ErrorID	Additional1	Additional2	Description
ACCESS_ERROR (16#0004)	16#00000A00	16#00000001	The parameter P-0-0694 is not parameterized in the optional cyclic telegram of the slave axis
ACCESS_ERROR (16#0004)	16#00000A00	16#00000002	The parameter P-0-0692 is not parameterized in the optional cyclic telegram of the slave axis
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000001	The master axis velocity in the lower operating point ("MasterVelocityLow") is < 0
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000002	The master axis velocity in the upper operating point ("MasterVelocityHigh") is < 0
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000003	The master axis velocity in the lower operating point ("MasterVelocityLow") and upper operating point ("MasterVelocityHigh") are provided with the same values
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000004	The value of the master axis velocity in the lower operating point ("MasterVelocityLow") is greater than the value in the upper operating point ("MasterVelocityHigh")
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000005	The value of the "LowLimit" input is lower than the minimum allowed (-95.0)
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000006	The value of the inputs "PGainLowVel" or "PGainHighVel" is lower than or equal to 0

ErrorID	Additional1	Additional2	Description
DEVICE_ERROR (16#0008)	16#00000A02	16#00000007	No power is added to the drive specified in "Slave". Closed-loop tension control is therefore not possible
DEVICE_ERROR (16#0008)	16#00000A02	16#00000008	The drive specified in the "Slave" is not in a synchronous operation mode
RESOURCE_ERROR (16#0003)	16#00001503	16#00000000	AxisData is not active in the control. This error occurs only on the IndraMotion MLD

Fig.6-22: Error codes of the MB_TensionControlLoadCellType01 function block

6.4 Function Blocks for the Application "Sag Control"

6.4.1 Introduction and Overview

Sag controls are used to decouple system components.

The following belt system is a typical use case.

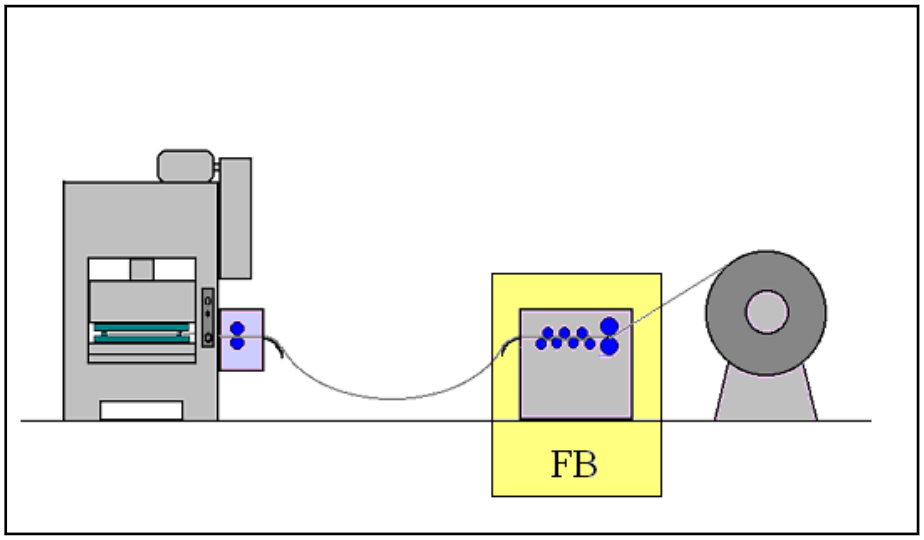


Fig.6-23: Belt system, concept sketch

The sag is located between the punching machine feed rolls and the upstream feed system. It is used to decouple both motion systems (start/stop or continuously). The sag depends on the roll feed data.

In normal operating mode one area of the belt system runs continuously at an average velocity v specified by the operator.

The other system area runs in start-stop operation. While the sheet is formed, this part of the system goes into stop mode and the sag accumulates. When the material is tracked for the next process, the sag contents are removed again. In one work cycle, the sag always move between the min./max. contacts.

Prerequisites for using the function block for sag control:

When setting up the belt system, the new sheet belt must be fed from the coil through the straightener to the punching machine such that a sag already exists between the min./max. contacts (or min./max. analog values).

- ML_SagControlC, page 182, controls the sag with min./max. contacts.

- ML_SagControlA, page 185, controls the sag with analog switch signals.

6.4.2 ML_SagControlC

Brief Description

The "ML_SagControlC" function block contains a sag controller with binary minimum and maximum contacts.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-24: Reference table

Interface Description

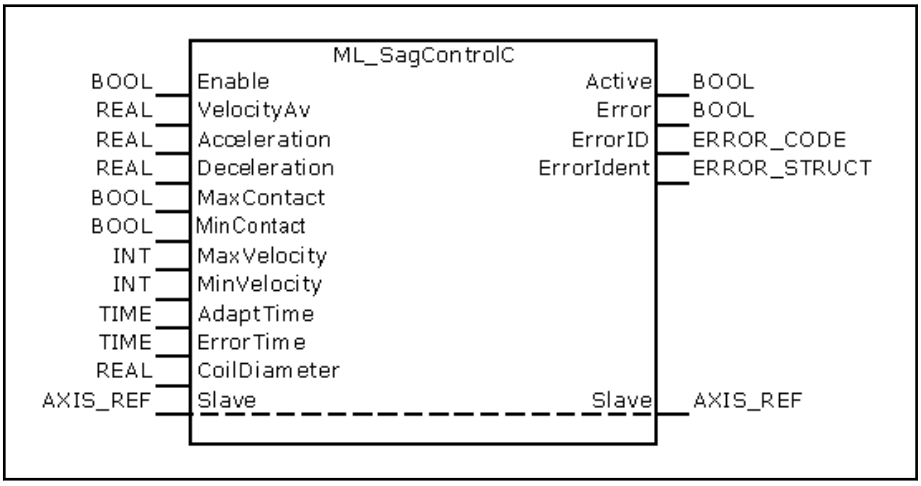


Fig.6-25: ML_SagControlC function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Slave	AXIS_REF	Slave axis
VAR_INPUT	Enable	BOOL	Activates/deactivates the closed-loop velocity control
	VelocityAv	REAL	Average velocity of the belt system
	Acceleration	REAL	Acceleration for velocity changes
	Deceleration	REAL	Deceleration for velocity changes
	MaxContact	BOOL	Maximum contact of the sag
	MinContact	BOOL	Minimum contact of the sag
	MaxVelocity	INT	Maximum velocity in % based on the average velocity of the belt system
	MinVelocity	INT	Minimum velocity in % based on the average velocity of the belt system
	AdaptTime	TIME	Adaptation time (Ta)
	ErrorTime	TIME	Error time (Te)
	CoilDiameter	REAL	Coil diameter ⇒ Evaluation of the velocity in %. If the input is not set, a default value of 100% is set
VAR_OUTPUT	Active	BOOL	Sag control is running
	Error	BOOL	Indicates an error. "Enable" = FALSE clears the error

I/O type	Name	Data type	Description
	ErrorID	ERROR_CODE	INT (Enum): Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description depending on the error table

Fig.6-26: Interface variables of the ML_SagControlC function block

Functional Description

The input "Enable" activates/deactivates the closed-loop velocity control. The first active velocity is the average velocity of the system specified by the operator.

If the "Enable" input is reset, the drive to be controlled is set in stop mode.

The function block for the sag control function exclusively affects the drive for material feed. This is an attempt to keep the sag in a defined window (min./max. value) by adjusting the current velocity of the belt system based on each deviation, with an optional sensor for detecting the coil diameter.

When the minimum contact is reached, the velocity of the material feed system increases to the maximum velocity associated with the function block to build up a greater sag. If movement away from the minimum contact towards the maximum contact occurs again within the error reaction time "Te", the greater velocity is used for operation for the rest of the adaptation time "Ta" so that the operation continues at the average velocity specified. If the sag becomes so big that the maximum contact responds, the velocity is reduced to the minimum velocity applied to the function block. If it moves away from the maximum contact within the error reaction time "Te", the drive continues to operate at the reduced velocity for the remaining time "Ta". Afterwards, the operation continues at the specified average velocity (see v-t diagram).

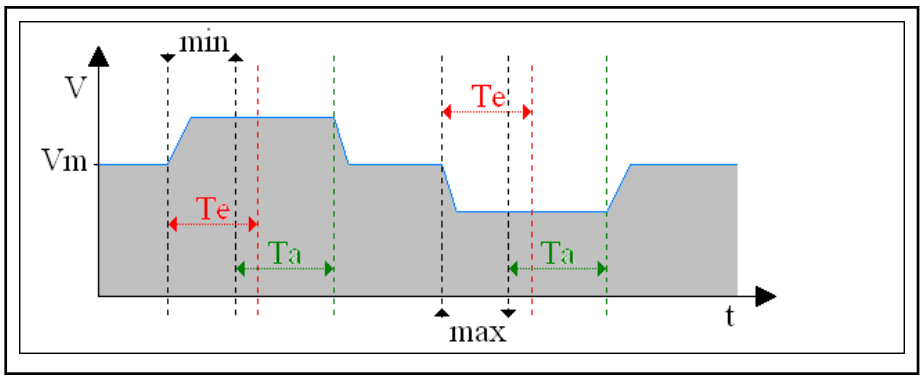


Fig.6-27: Correct function block behavior

Error Handling

If the minimum contact is pending for a period longer than "Te", an error is generated and the operation continues at the same velocity. This state remains until the contact is left again. Afterwards, the drive still operates at the increased velocity for the rest of the adaptation time "Ta" and then at the average velocity specified. However, if the maximum contact is pending for the period "Te", velocity is reduced to zero. If the maximum contact switches again, the drive remains stopped for the period "Ta" and then accelerates again to the specified average velocity "Vm" and continues to operate at this velocity.

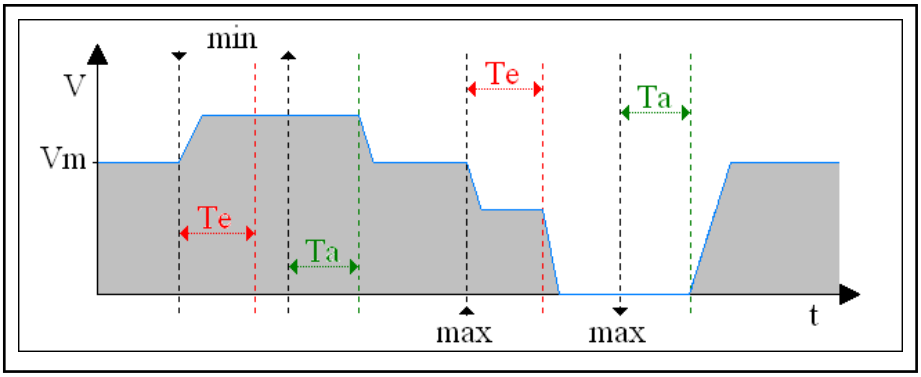


Fig.6-28: Incorrect function block behavior

The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**, 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs outside of the valid range: Velocity <= 0 Acceleration <= 0 Deceleration <= 0 Max. velocity. <= Min. velocity Adaptation time <= 0 Error reaction time <= 0 Coil diameter <= 0
SYSTEM_ERROR (16#7FFF)	16#0501	16#0000	Velocity adaptation failed, adaptation time and error time have elapsed, maximum contact is still present
SYSTEM_ERROR (16#7FFF)	16#0502	16#0000	Velocity adaptation failed, adaptation time and error time have elapsed, minimum contact is still present
SYSTEM_ERROR (16#7FFF)	16#0503	16#0000	Maximum and minimum contacts are present at the same time. Contact problem ?
SYSTEM_ERROR (16#7FFF)	16#0504	16#0000	The predefined velocity could not be reached. Check velocity specifications

Fig.6-29: Error codes of the ML_SagControlC function block

Firmware, software, hardware requirements

The following firmware, software and hardware is required:

- Hardware - IndraDrive C or M
- Firmware - Drive firmware (function package - Closed Loop)
- Software - IndraWorks Engineering 12VRS

Application example 1

In the simplest form 2 contacts, 1 minimum and 1 maximum contact are available as closed-loop control parameters.

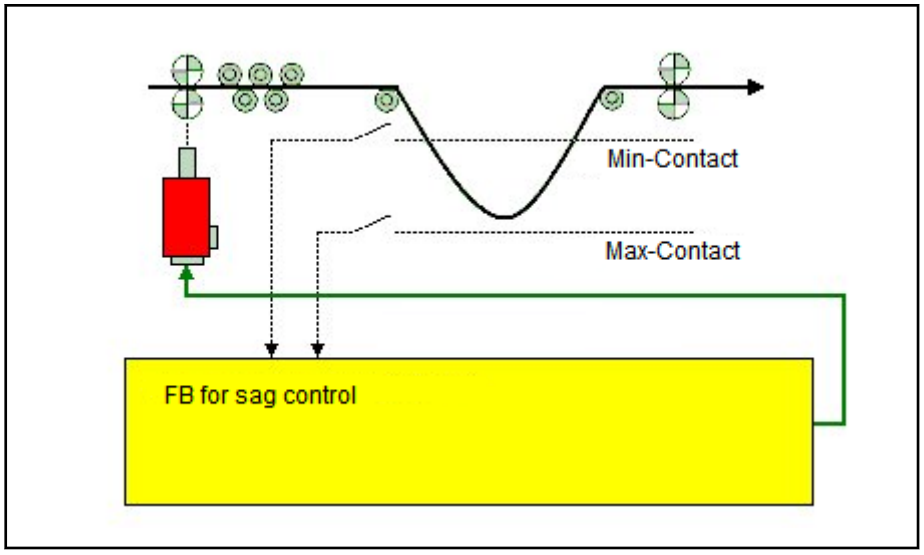


Fig.6-30: Application 1, ML_SagControlC

In normal operation mode, the belt system runs at an average velocity "Vm" specified by the operator. The function block for the sag control now attempts to keep the sag within a defined window (min./max. contact) by adjusting the current velocity of the belt system based on each deviation.

Application example 2

In this variant, the drive to be controlled is the coil itself. The decreasing coil diameter is used in addition to the min./max. contacts as a controlled variable.

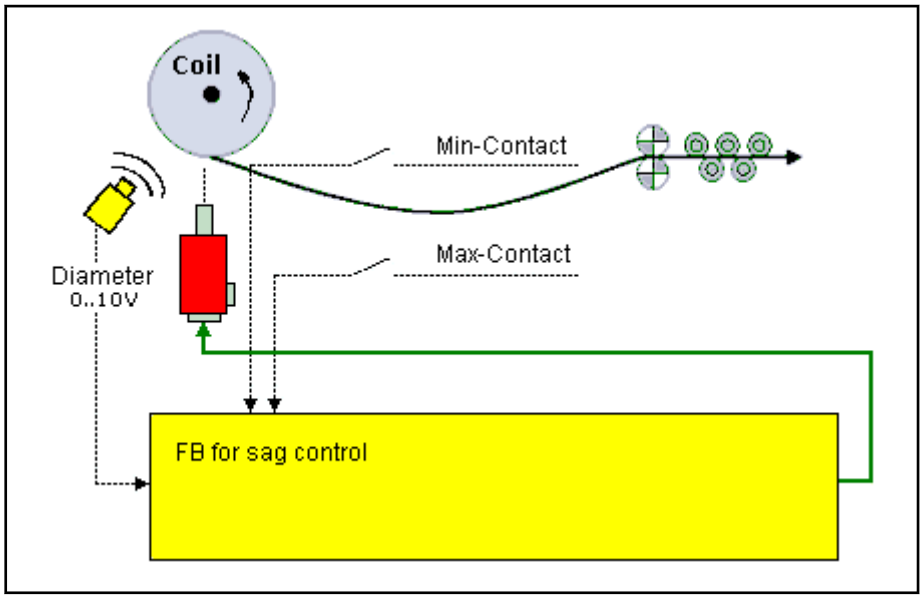


Fig.6-31: Application 2, ML_SagControlC

6.4.3 ML_SagControlA

Brief Description

The function block "ML_SagControlA" contains a sag controller with an analog actual value measurement for the sag.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-32: Reference table

ML_TechMotion.library

Interface Description

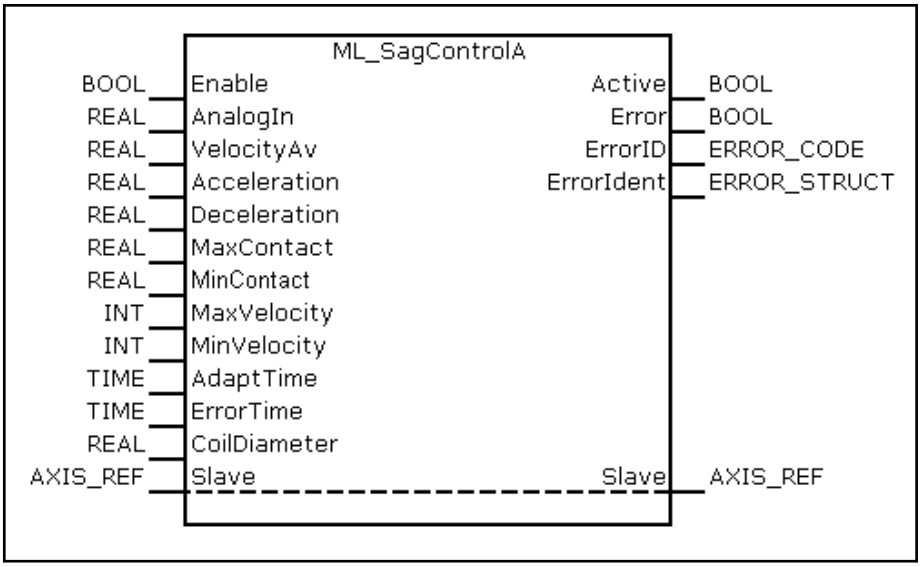


Fig.6-33: ML_SagControlA function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Slave	AXIS_REF	Slave axis
VAR_INPUT	Enable	BOOL	Activates/deactivates the closed-loop velocity control
	AnalogIn	REAL	Analog value for sag (0 - 10)
	VelocityAv	REAL	Average velocity of the belt system
	Acceleration	REAL	Acceleration for velocity changes
	Deceleration	REAL	Deceleration for velocity changes
	MaxContact	REAL	Maximum contact of the sag
	MinContact	REAL	Minimum contact of the sag
	MaxVelocity	INT	Maximum velocity in % based on the average velocity of the belt system
	MinVelocity	INT	Minimum velocity in % based on the average velocity of the belt system
	AdaptTime	TIME	Adaptation time (Ta)
	ErrorTime	TIME	Error time (Te)
	CoilDiameter	REAL	Coil diameter ⇒ Evaluation of the velocity in %. If the input is not set, a default value of 100% is set
VAR_OUTPUT	Active	BOOL	Sag control is running
	Error	BOOL	Indicates an error. "Enable" = FALSE clears the error
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description depending on the error table

Fig.6-34: Interface variables of the ML_SagControlAfunction block

Functional Description

The "Enable" input activates/deactivates the closed-loop velocity control. The first active velocity is the average velocity of the system specified by the operator.

If the "Enable "input is reset, the drive to be controlled is set in stop mode.

The function block for the sag control function exclusively affects the drive for material feed. This is an attempt to keep the sag in a defined window (sensor analog value) by adjusting the current velocity of the belt system based on each deviation, with an optional sensor for detecting the coil diameter.

When the minimum contact is reached, the velocity of the material feed system increases to the maximum velocity associated with the function block to build up a greater sag. If movement away from the minimum contact towards the maximum contact occurs again within the error reaction time "Te", the greater velocity is used for operation for the rest of the adaptation time "Ta" so that the operation continues at the average velocity specified. If the sag becomes so big that the maximum contact responds, the velocity is reduced to the minimum velocity applied to the function block. If it moves away from the maximum contact within the error reaction time "Te", the drive continues to operate at the reduced velocity for the remaining time "Ta". Afterwards, the operation continues at the specified average velocity (see v-t diagram).

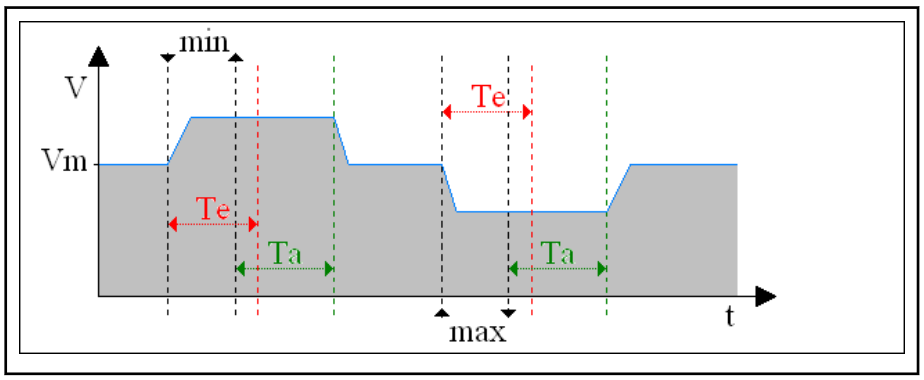


Fig.6-35: Correct function block behavior

Error Handling

If the minimum contact is pending for a period longer than "Te", an error is generated and the operation continues at the same velocity. This state remains until the contact is left again. Afterwards, the drive still operates at the increased velocity for the rest of the adaptation time "Ta" and then at the average velocity specified. However, if the maximum contact is pending for the period "Te", velocity is reduced to zero. If the maximum contact switches again, the drive remains stopped for the period "Ta" and then accelerates again to the specified average velocity "Vm" and continues to operate at this velocity.

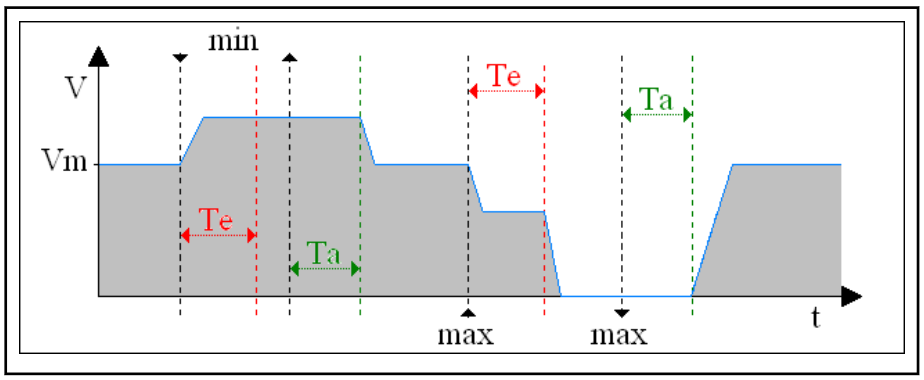


Fig.6-36: Incorrect function block behavior

The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**, 16#0170:

ML_TechMotion.library

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs outside of the valid range: Velocity <= 0 Acceleration <= 0 Deceleration <= 0 Max. velocity. <= Min. velocity Adaptation time <= 0 Error reaction time <= 0 Coil diameter <= 0
SYSTEM_ERROR (16#7FFF)	16#0501	16#0000	Velocity adaptation failed, adaptation time and error time have elapsed, maximum contact is still present
SYSTEM_ERROR (16#7FFF)	16#0502	16#0000	Velocity adaptation failed, adaptation time and error time have elapsed, minimum contact is still present
SYSTEM_ERROR (16#7FFF)	16#0504	16#0000	The predefined velocity could not be reached. Check velocity specifications

Fig.6-37: Error codes of the ML_SagControlA function block

Firmware, software, hardware requirements

The following firmware, software and hardware is required:

- Hardware - IndraDrive C or M
- Firmware - Drive firmware (function package - Closed Loop)
- Software - IndraWorks Engineering 12VRS

Application example 1

In this variant, a sensor reproduces the sage position as an analog input (0-10V) which is added to the closed-loop velocity control.

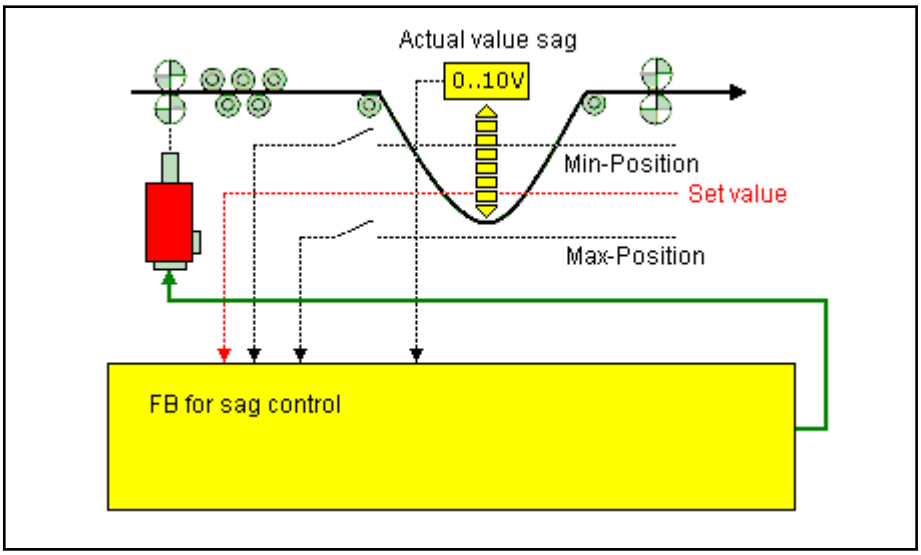


Fig.6-38: Application 1, ML_SagControlA

In normal operation mode, the belt system runs at an average velocity Vm specified by the operator. The function block for the sag control now attempts

to keep the sag within a defined window (min./max. position) by adjusting the current velocity of the belt system based on each deviation.

Application example 2

In this variant, the drive to be controlled is the coil itself. The decreasing coil diameter is used as a controlled variable in addition to the analog input.

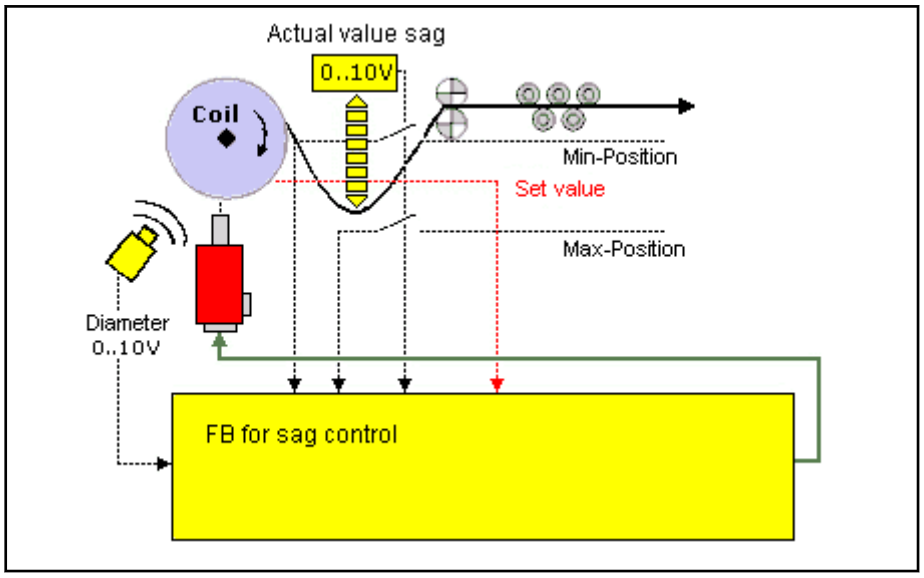


Fig.6-39: Application 2, ML_SagControlA

6.5 Function Blocks for Synchronous Lock-on and Lock-off

6.5.1 Overview

Starting with firmware version 3, Bosch Rexroth IndraMotion MLC controls were supplemented with the availability of CamLock technology function blocks regarding the "Lock On/Lock Off" functionality.

The **MB_PrepareCams** function block uses a polynomial 5th order to calculate, build and download three cam profiles to a slave axis. These cam profiles are designated as Lock On, Lock Off and one-to-one. They allow a synchronized slave axis to disengage (Lock Off) from its master and stop at a predefined position until it is once again synchronized (Lock On) to the master. The **MB_CamLock** function block is used to activate the Lock On/Lock Off functionality. A user cam profile is also supported.

While a slave axis is synchronized to the master, it follows the master using the one-to-one cam. A phase offset between the master and slave position can be defined. When the Lock Off cam is enabled (via the MB_CamLock function block), the slave axis transitions off of the master position to a predefined lock off position and comes to a stop. When the Lock Off cam is disabled (Lock On), the slave axis transitions from its stopped position and re-synchronizes back to the master. Once synchronized, the slave axis switches back to the One-to-One cam and continues following the master input position.

6.5.2 CamLock - Application Example

Normally, this functionality is used in packaging machines, provided that the incoming products are spaced equally before they can be packaged. If the distance is too large, the packaging process in the machine is locked off for one or several cycles by the master until the product is detected. This state is called "No Product, No Seal". As soon as the product is detected, the pack-

aging process is re-synchronized to the master (Lock On) and the packaging of the products continues.

These unequal distances require the slave axis to be accelerated and slowed down in order to synchronize it to the master. The following figure shows a typical packaging machine for filling and sealing:

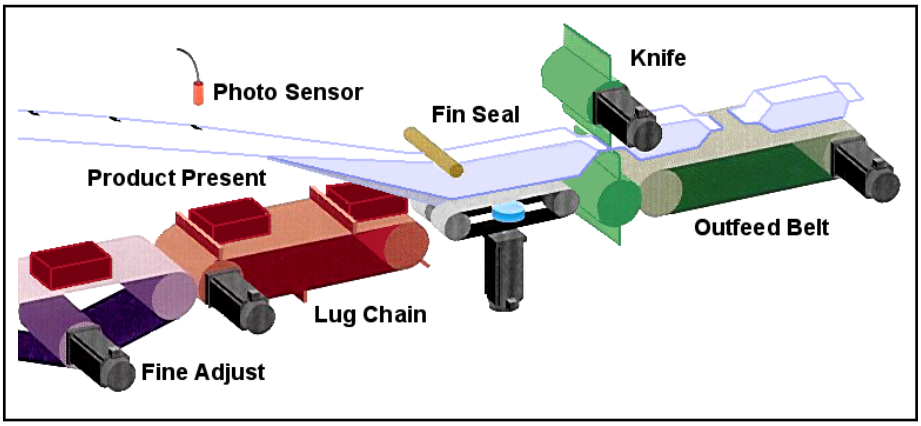


Fig.6-40: Horizontal version of a packaging machine for filling and sealing

The following diagram shows the processes of processing, slowing down, stopping, accelerating, and re-processing a slave axis follows during the Lock On/Lock Off process.

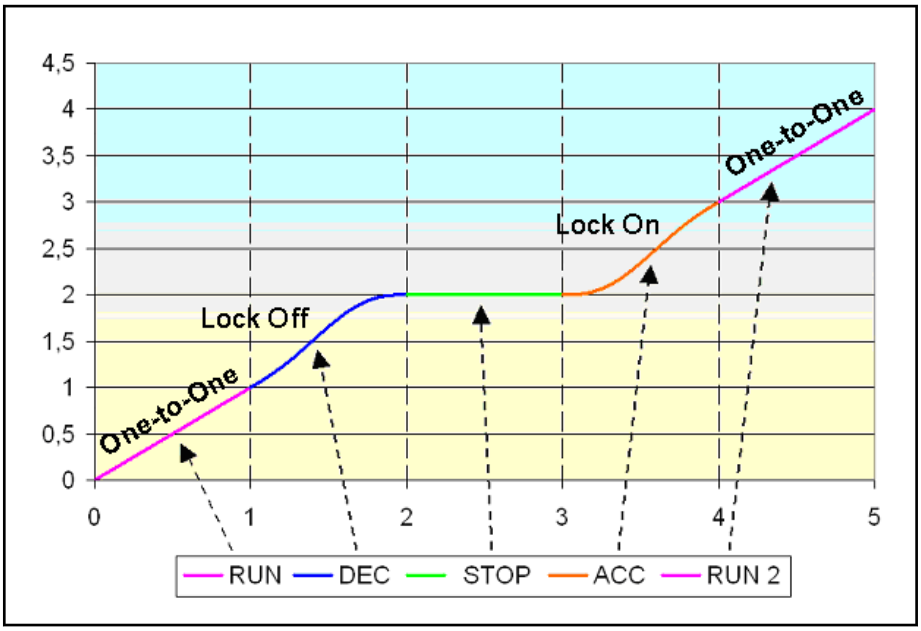


Fig.6-41: CamLock cam profile

One-to-one cam profile "Synchronized to master"

The one-to-one cam profile is active and synchronized to the master input, if the Lock On/Lock Off functionality is inactive. The Lock On/Lock Off functionality is activated by setting the enable input of the MB_CamLock function block to "High". The Lock Off input has to be set to "Low". In normal working conditions, this cam profile is active and follows the specifications of the master input.

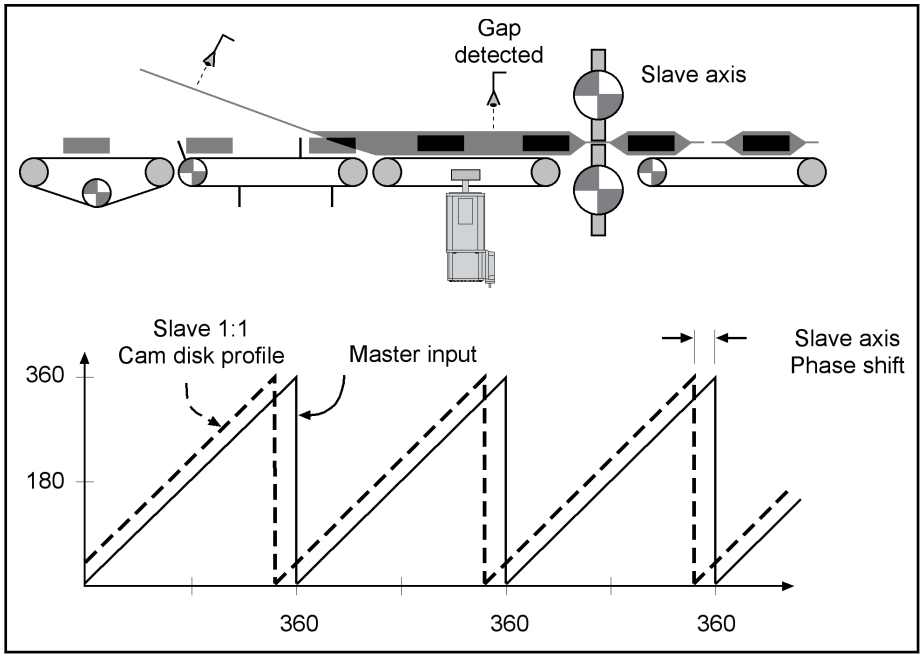


Fig.6-42: Processing cam active, normal operation of the packaging application

Lock Off cam profile

The Lock Off cam profile slows down the slave axis up to a complete halt via a cycle of the master. The Lock Off cam profile is active as long as both inputs "Enable" and "LockOff" of the MB_CamLock function block are set to "High". After this cycle the velocity of the slave axis is stopped and is only restarted when the Lock On cam profile is active.

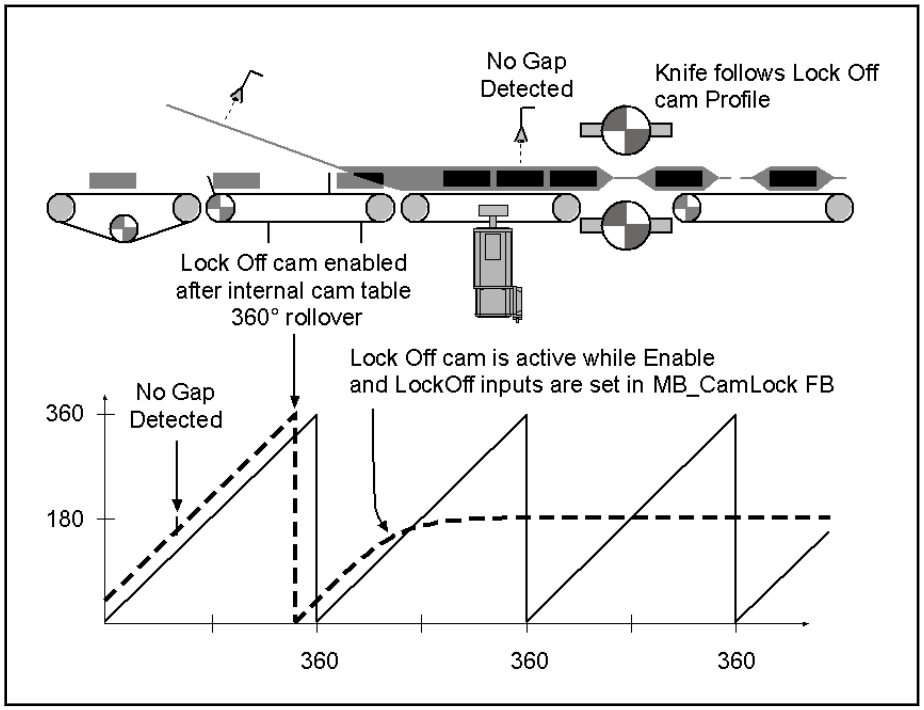


Fig.6-43: Lock Off cam active, No Product - No Seal

Lock On cam profile

The Lock On cam profile accelerates from a stop position until the velocity of the master input is reached during a cycle of the master (360°). After this cycle, the slave axis and the master have the same velocity. The Lock On cam is active until the slave axis is synchronized with the master. Afterwards, the slave axis follows the one-to-one cam during normal operation. The Lock On cam profile is active as long as the "Enable" input is set to "High" and the Lock Off input is set to "Low" at the MB_CamLock function block.

☞ Once activated, the Lock On cam profile is active upon zero passage of the master axis.

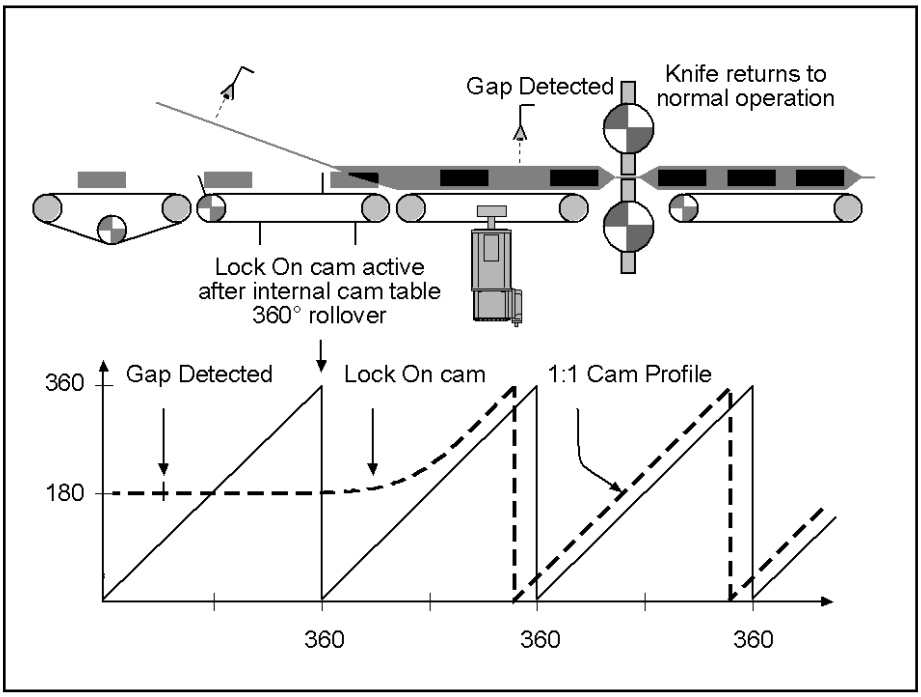


Fig.6-44: Lock On cam active, product present

6.5.3 MB_PrepareCams

Brief Description

The function block uses a polynomial 5th order to calculate and to generate 3 cam profiles and to load them into parameters specified by the inputs MC_CAM_ID. The resulting MotionProfiles contain boundary conditions for position and velocity.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-45: Reference table of the MB_PrepareCams function block

Interface Description

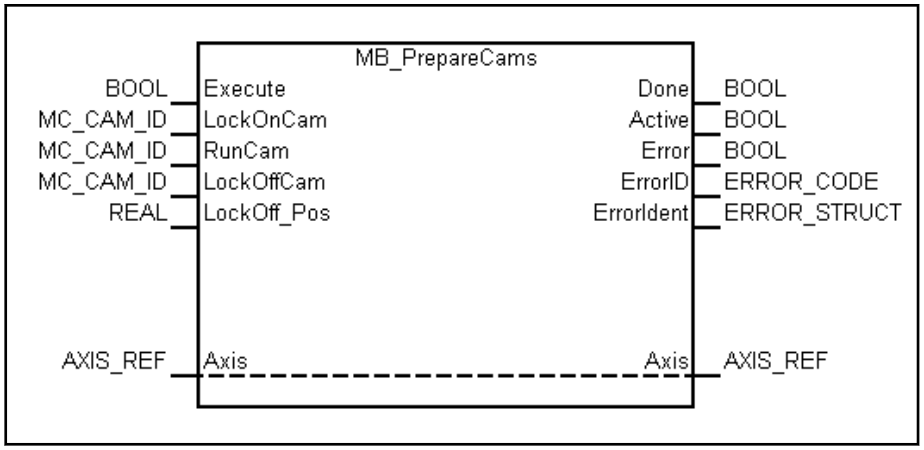


Fig.6-46: MB_PrepareCams function block

The MC_CAM_ID value only specifies which parameter is used to save the calculated cam profile. The current functionality (e.g., LockOnCam) is determined by the input in the function block.

For an IndraMotion MLC control system, the cam profile is saved in the control parameter C-0-2001 if the value is 1. However, in order to execute the function block MB_PrepareCams without any errors, the axis input AXIS VAR_IN_OUT is still required. The calculated cam profiles are stored in control parameters.

I/O type	Name	Data type	Description
VAR_IN_OUT	Axis	AXIS_REF	Reference to the slave axis for which cam profiles are saved
VAR_INPUT	Execute	BOOL	The positive edge starts the calculation of three cam profiles
	LockOnCam	MC_CAM_ID	Determines the target address of the Lock On cam table. Default value is 4
	RunCam	MC_CAM_ID	Determines the target address of the editing cam table. Default value is 3
	LockOffCam	MC_CAM_ID	Determines the target address of the Lock Off cam table. Default value is 2
	LockOff_Pos	REAL	This input is used to calculate the respective velocity profile
VAR_OUTPUT	Done	BOOL	Three cam profiles have been calculated, build and loaded
	Active	BOOL	Function block is active
	Error	BOOL	Indicates that an error has occurred
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

Fig.6-47: Interface variables of the MB_PrepareCams function block

IndraMotion MLC control system

An IndraMotion MLC control system supports a block of 98 control parameters to store CamLock cam profiles. Unlike drive parameters, the assignment between the MC_CAM_ID value and the control parameter is continuous. Starting with control parameter C-0-2001, an MC_CAM_ID value of 1 is

saved in the control parameter C-0-2001, an MC_CAM_ID value of 98 in C-0-2098. The control parameter C-0-2099 cannot be used to save a cam profile. This parameter is reserved to "stop the slave axis".

This function block has to run right after control startup to calculate the three cam profiles and download them into the relevant parameters. Use the "DONE" output to check that this procedure has been completed before the PLC program continues running.

☞ The MB_PrepareCams function block is executed only once at the start of the PLC program and before the start of the MB_CamLock function block.

Error Handling

The function block generates the following error messages in Additional1/Additional2 for the table "**F_RELATED_TABLE**":

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#0003	16#0000	Function block was aborted by another function block
RESOURCE_ERROR	16#0004	16#0000	This drive firmware version is not supported
INPUT_RANGE_ERROR	16#13A1	16#0001	The cam-related values were initialized improperly
INPUT_RANGE_ERROR	16#13A1	16#0002	Slave Axis_Ref; the AxisNo is not within the admissible range
INPUT_RANGE_ERROR	16#13A1	16#0003	LockOff_Pos has to be larger than 0 and smaller than 360; default = 180 degrees
CALCULATION_ERROR	16#13A2	16#0000	Calculation of the step width, result = 0
STATE_MACHINE_ERROR	16#0006	16#0000	Invalid state of the state machine

Fig.6-48: Error codes of the MB_PrepareCams function block

6.5.4 MB_CamLock

Brief Description

The The MB_CamLock function block is used to enable the Run, Lock On and Lock Off cam profiles calculated and stored by the MB_PrepareCams function block. In addition to the enabling of cam profiles, this function block also provides the following functionality:

- Electronic gear ratio
- Direction of synchronization (SyncMode)
- MC_CamIn and MC_CamOut functionality for drive cams
- Master fine adjustment

☞ The function block operation requires the cyclic call in the Motion task. Please note the parameterization in chapter 6.5.5 "MB_CamLock Parameterization" on page 201

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-49: Reference table

Interface Description

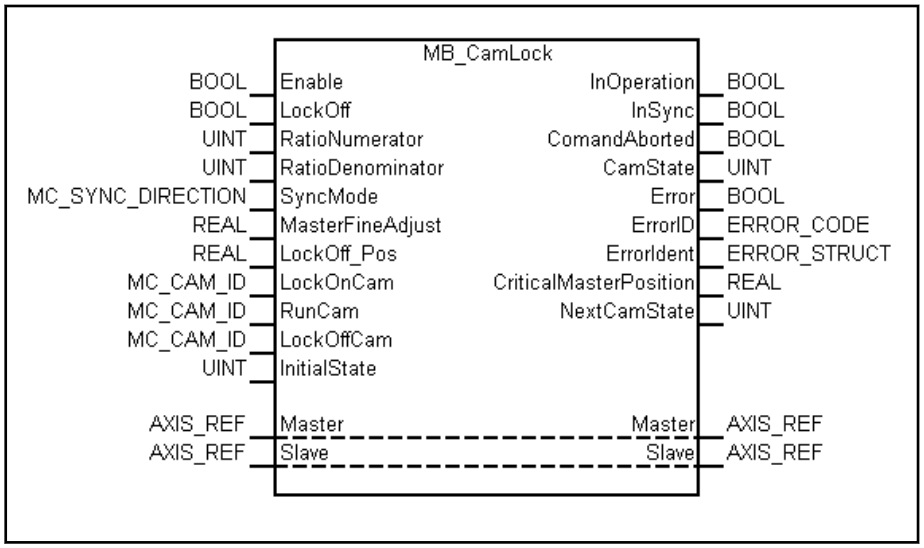


Fig.6-50: MB_CamLock function block

For an IndraMotion MLC control system, control cams can be used to assign a slave input as virtual axis.

The following table lists the different cam profiles controlled by the MB_CamLock function block:

Cam profile	"Enable" input	Lock Off entry	Description
Run	High	Low	Normal operation mode using a 1:1 cam profile
LockOff	High	Low to high	The slave axis switches from the Run cam to the LockOff cam profile and stops at the position specified in the LockOff_Pos input
LockOn	High	High to low	The slave axis accelerates from a stopped position using the LockOn cam profile, synchronizes with the master and switches to the Run cam

Fig.6-51: Normal CamLock operation

I/O type	Name	Data type	Description
VAR_IN_OUT	Slave	AXIS_REF	Real or virtual axis
	Master	AXIS_REF	Real or virtual axis
VAR_INPUT	Enable	BOOL	Activates the MB_CamLock function block. Due to the increasing edge, the slave is synchronized dynamically before entering the Run state. A falling edge executes a gear lock off

ML_TechMotion.library

I/O type	Name	Data type	Description
	LockOff	BOOL	TRUE: Locked off FALSE: Locked on If this input is TRUE before setting "Enable", the axis dynamically synchronizes, enters the Run state and and locks off (LockOff)
	RatioNumerator	UINT	Electronic gear ratio
	RatioDenumerator	UINT	Electronic gear ratio
	SyncMode	MC_SYNC_DIRECTION	0: = shortest distance 1: = positive direction 2: = negative direction
	MasterFineAdjust	REAL	Input required for the MB_MotionProfile function block
	LockOff_Pos	REAL	The position in degrees where the slave axis stops when locked off from the master. The default values is 180 degrees
	LockOnCam	MC_CAM_ID	Determines the source of the LockOn cam table. Default value is 4
	RunCam	MC_CAM_ID	Determines the source of the Run cam table. Default value is 3
	LockOffCam	MC_CAM_ID	Determines the source of the LockOff cam table. Default value is 2
	InitialState	UINT	Initial state of the state machine
VAR_OUTPUT	InOperation	BOOL	TRUE as long as the function block is enabled
	InSync	BOOL	The "InSync" output is TRUE when the "Run" cam is in synchronized state.
	CommandAborted	BOOL	Command aborted by another function block.
	CamState	UINT	Current state of the CamLock function block 1: in Dynamic Sync (dynamic synchronization) 2: in Run (running state) 3: in LockOff (locked off state) 4: at Standstill (stop state) 5: in LockOn (locked on state) 6: in Gear Out [gear lock out (continuous motion)]
	Error	BOOL	Indicates that an error has occurred
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	CriticalMasterPosition	REAL	Provides the critical master position required for the transition of the switching state
	NextCamState	UINT	Indicates the next output to be processed

Fig.6-52: Interface variables of the MB_CamLock function block

Signal-time diagram

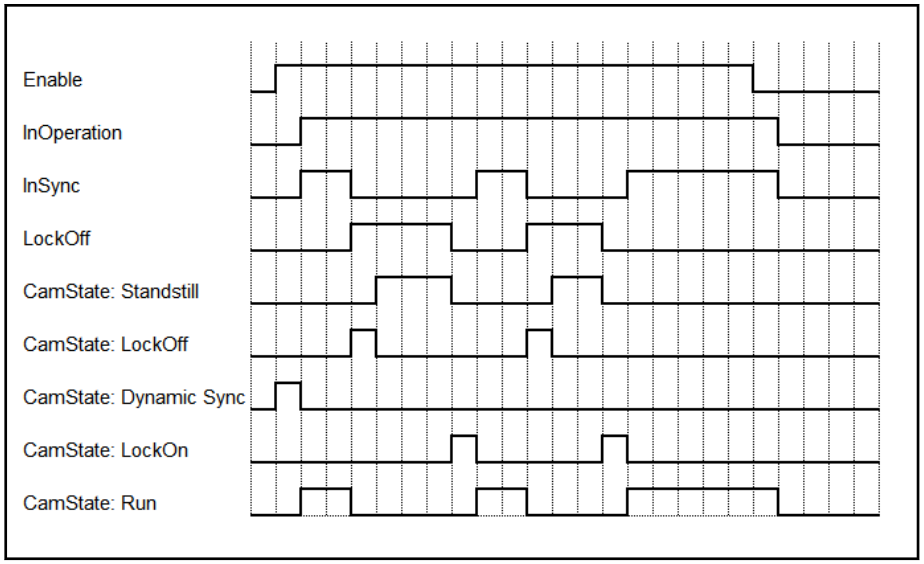


Fig.6-53: Signal-time diagram of the MB_CamLock

Functional Description

When the **Enable** input is set to "High" in combination with the **LockOff** input set to "Low", the slave axis dynamically synchronizes with the master and immediately switches to Run state. The **CamState** output transitions from Dynamic Sync state (1) to Run state (2). The function block remains in the Run state (editing a 1:1 cam) until the **LockOff** input is "High".

When the **LockOff** input goes "High" during Run mode, the function block executes a LockOff cam and stops at the position specified in the **LockOff_Pos** input. During the transition from Run to Stop, the **CamState** output transitions from Run (2) to LockOff (3) and then to Standstill (4). The default value for LockOff_Pos is 180 degrees, which is half distance from LockOn to Run and from Run to LockOff.

At this point, the function block remain in the stop state (standstill) until the LockOff input becomes "Low". This completes the cycle. To disable the CamLock function block, it is recommended to be in CamState=4 (stopping state) and set the Enable input to low.

If the **Enable** input is set to "Low" in the Run state, the slave desynchronizes from the LockOn/LockOff cam functionality and switches to continuous motion.

ML_TechMotion.library

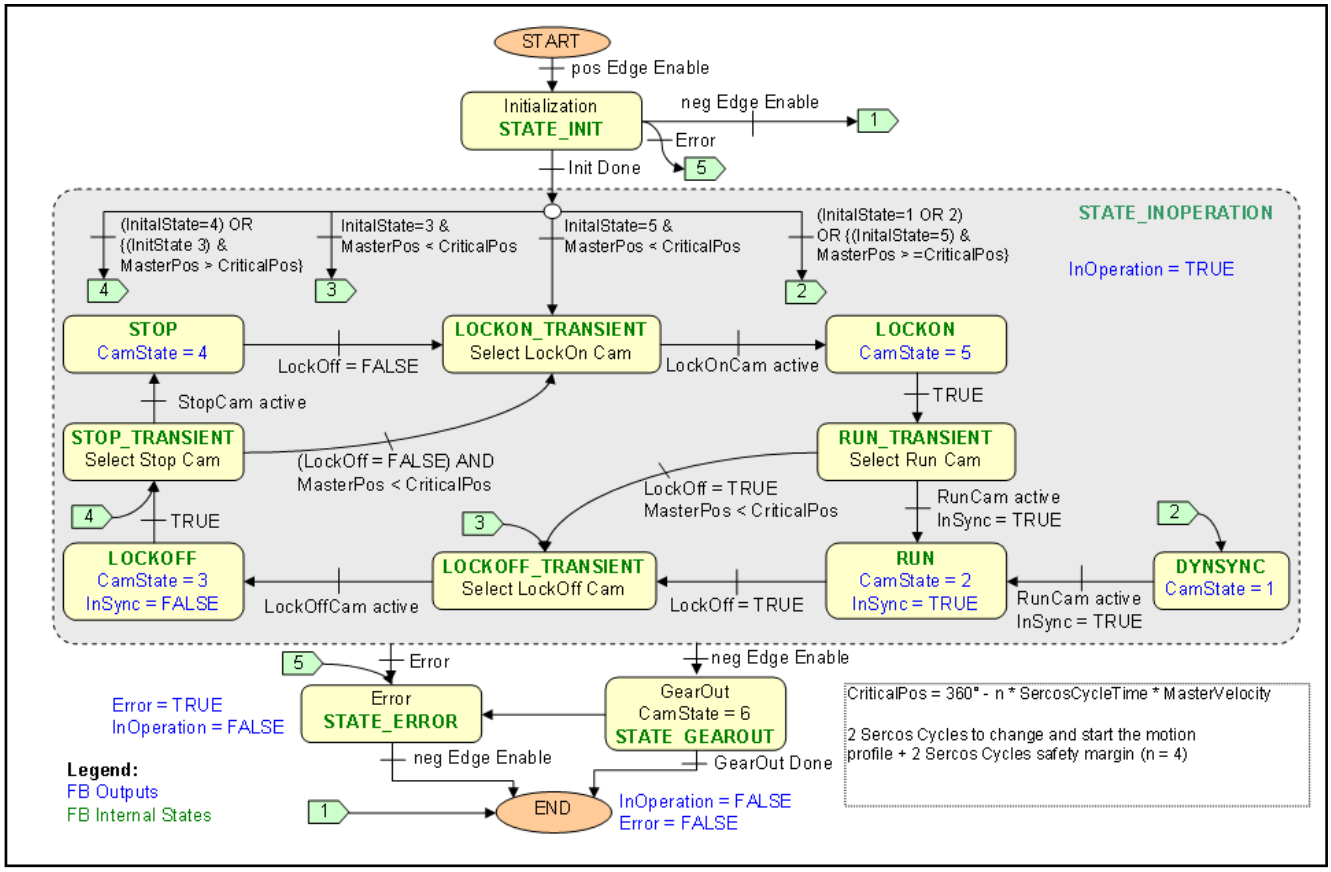
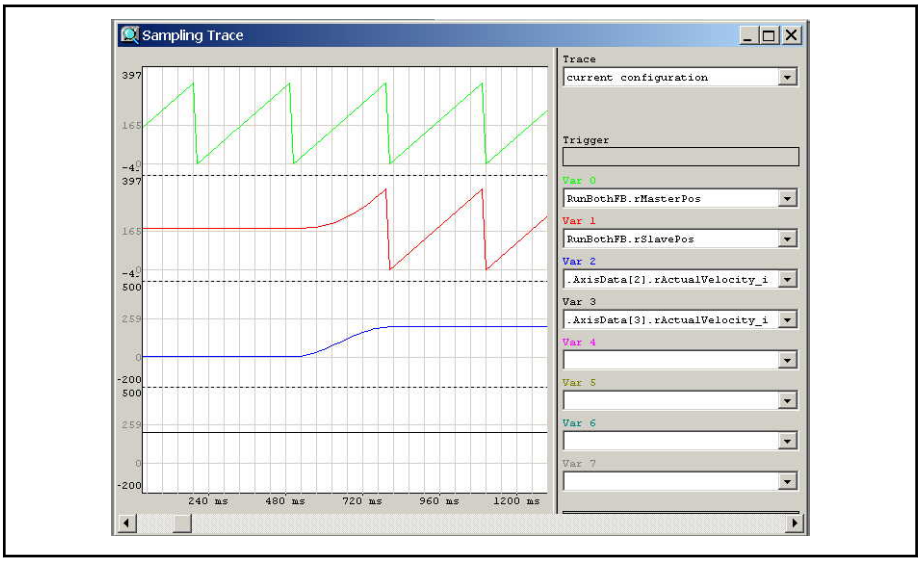


Fig.6-54: CamLock state machine

LockOn/LockOff sequence examples

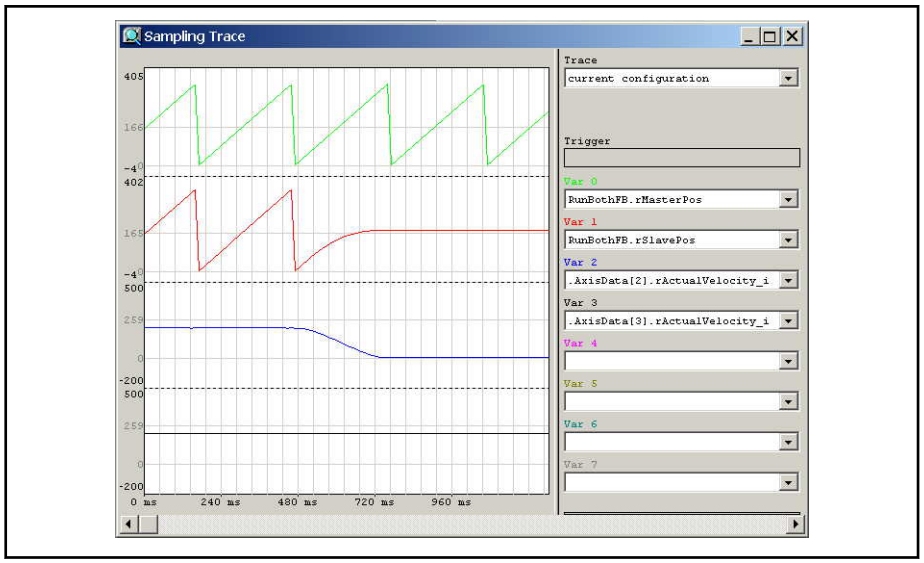


Channel 0 (green) = master position
Channel 1 (red) = slave position
Channel 2 (blue) = slave velocity
Master velocity = 200 rpm

Fig.6-55: LockOn profile with a LockOff_Pos of 180 degrees

The figure above shows the position and velocity profile during the LockOn profile. The LockOff_Pos position is 180 degrees (default). The figure above

shows the position and velocity profile during the LockOff profile without jumps and overshoots. The LockOff position is 180 degrees (default).



Channel 0 (green) = master position
Channel 1 (red) = slave position
Channel 2 (blue) = slave velocity
Master velocity = 200 rpm

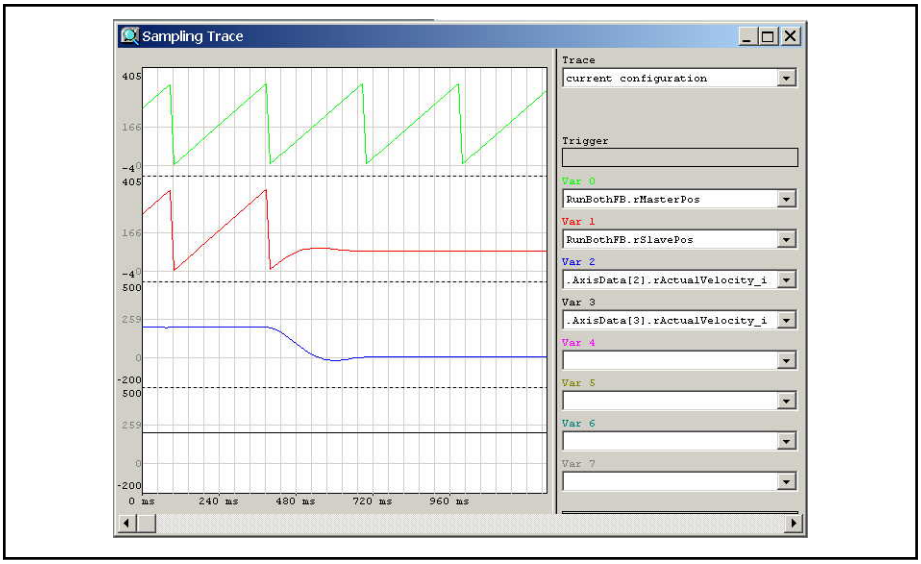
Fig.6-56: LockOff profile with a LockOff_Pos of 180 degrees

The figure above shows the position and velocity profile during the LockOff profile. The LockOff_Pos position is 180 degrees (default). The figure above shows the position and velocity profile during the LockOff profile without jumps and overshoots. The LockOff position is 180 degrees (default).

Channel 0 (green) = master position
Channel 1 (red) = slave position
Channel 2 (blue) = slave velocity
Master velocity = 200 rpm

Fig.6-57: LockOn profile with a LockOff_Pos of 90 degrees

The figure above shows the position and velocity profile during the LockOn profile if the LockOff_Pos is 90 degrees. It can be seen that the position and velocity profiles are still ok for a LockOff position of 90 degrees (there is a velocity overshoot, but no backward motion).



Channel 0 (green) = master position
Channel 1 (red) = slave position
Channel 2 (blue) = slave velocity
Master velocity = 200 rpm

Fig.6-58: LockOff profile with a LockOff_Pos of 90 degrees

The figure above shows the position and velocity profile during the LockOff profile if the LockOff position is 90 degrees. It can be seen that the position and velocity profiles are provided with an overrun for a LockOff position of 90 degrees. This position overshoot causes the a backward motion.

☞ It is recommended to use a value of 180 for the LockOff_Pos input (this input has to match for both the MB_PrepareCams and MB_CamLock function blocks). If a LockOff position other than 180 is used, then, the further away from 180 degrees, the more overruns in position and velocity occur.

Error Handling

The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#0001	16#0000	Power for the slave axis is not ON
RESOURCE_ERROR	16#0003	16#0000	Function block was aborted by another function block
STATE_MACHINE_ERROR	16#0006	16#0000	Invalid state of the state machine
ACCESS_ERROR	16#0011	16#0000	Control is not in phase BB (SERCOS P4)
INPUT_RANGE_ERROR	16#1301	16#0001	The LockOff_Pos has to be greater than 0 and lower than 360, default = 180 degrees
INPUT_RANGE_ERROR	16#1301	16#0002	InitialState is not within the range 1...5
INPUT_RANGE_ERROR	16#1301	16#0003	RatioNumerator =< 0
INPUT_RANGE_ERROR	16#1301	16#0004	RatioDenumerator =< 0
INPUT_RANGE_ERROR	16#1301	16#0005	SyncMode < 0 OR SyncMode > 2

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#1301	16#0006	LockOnCam < 1 OR LockOnCam > 98
INPUT_RANGE_ERROR	16#1301	16#0007	RunCam < 1 OR RunCam > 98
INPUT_RANGE_ERROR	16#1301	16#0008	LockOffCam < 1 OR LockOffCam > 98
INPUT_RANGE_ERROR	16#1301	16#0009	UserCam_Profile < 1 OR User-Cam_Profile > 98
INPUT_RANGE_ERROR	16#1301	16#0010	CamTable inputs are equal
ACCESS_ERROR	16#1302	16#0003	Slave axis modulo value is not 360°
ACCESS_ERROR	16#1302	16#0004	Master axis modulo value is not 360°

Fig.6-59: Error codes MB_CamLock function block

6.5.5 MB_CamLock Parameterization

The CamLock function blocks requires the following parametrization:

1. The "Interpolation in the drive" checkbox has to be deselected when adding a real axis in IndraWorks if this axis is used as slave axis of the MB_CamLock.
2. Master and slave axis have to be set to the modulo format 360°.
3. The drive has to be referenced before using this function block (absolute position preferred).
4. The MB_PrepareLockCams function block has to be executed once to calculate and store the three cams required.
   - LockOnCam, stored in CAM_TABLE_4 by default
   - RunCam, stored in CAM_TABLE_3 by default
   - LockOffCam, stored in CAM_TABLE_2 by default
   - User_Cam_Profile, stored in CAM_TABLE_1 by default)
5. Power on to the drive (drive in "Enabled" state)
6. The MB_CamLock contains time-critical Motion functionality and should thus be called in the MotionTask.
7. The function block only operates for one master axis moving in positive direction.
8. Note: The CamID 99 is always used as stop Cam.

6.6 Function Blocks for General Usage

6.6.1 MB_DistanceAccumulator

Brief Description

The MB_DistanceAccumulator function block sums up the relatively traveled distance via an encoder. With the rising edge of the **Enable** input, the distance traveled is summed up by the function block until the summed up distance is higher than or equal to the **TotalDistance** input value. The **DistanceReached** output is set to TRUE and the function block processing is completed. The current encoder position is specified using the **CurrentPosition** input.

ML_TechMotion.library

Interface description

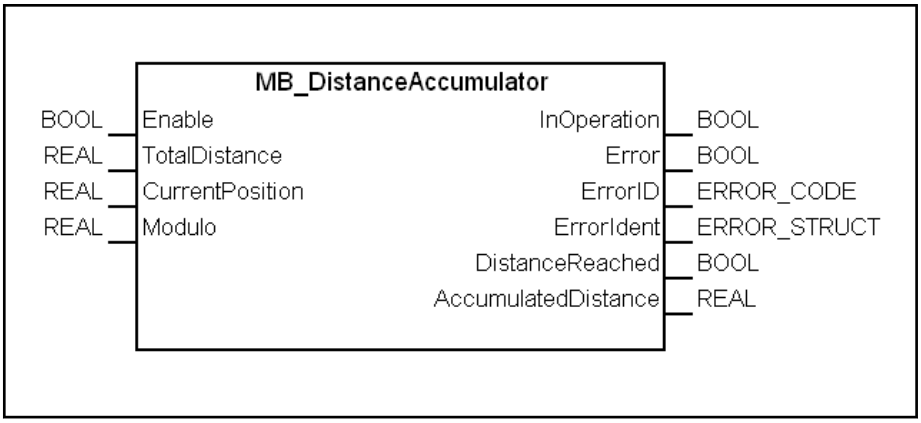


Fig.6-60: MB_DistanceAccumulator function block

I/O type	Name	Data type	Comment
VAR_INPUT	Enable	BOOL	Enables the function block with a rising edge
	TotalDistance	REAL	Total distance to be traveled before the process is completed
	CurrentPosition	REAL	Current encoder position
	Modulo	REAL	Modulo value: (360: rotary 0: linear) In case of a Servo axis, this value can be read from A-0-0045
VAR_OUTPUT	InOperation	BOOL	The function block is currently active
	Error	BOOL	Processing completed with errors
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	DistanceReached	BOOL	TotalDistance was reached and the processing of the function block is completed.
	AccumulatedDistance	REAL	Indicates the currently traveled distance

Fig.6-61: Interface variables of the MB_DistanceAccumulator function block

Error Handling

The function block generates the following error messages in Additional1/Additional2 for the **F_RELATED_TABLE**.

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#0003	16#0000	The function block aborted by another function block
STATE_MACHINE_ERROR	16#0006	16#0000	Invalid status of the state machine

Fig.6-62: Error codes of the MB_DistanceAccumulator function block

6.6.2 MB_AxisDistanceAccumulator

The MB_AxisDistanceAccumulator function block sums up the relatively traveled distance via an encoder. With the rising edge of the **Enable** input, the distance traveled is summed up by the encoder until the summed up distance is higher than or equal to the **TotalDistance** input value. As a result, the **DistanceReached** output is set to TRUE and the function block stops. This function block uses the axis reference input (AXIS_REF) to query the current encoder position automatically.

Interface Description

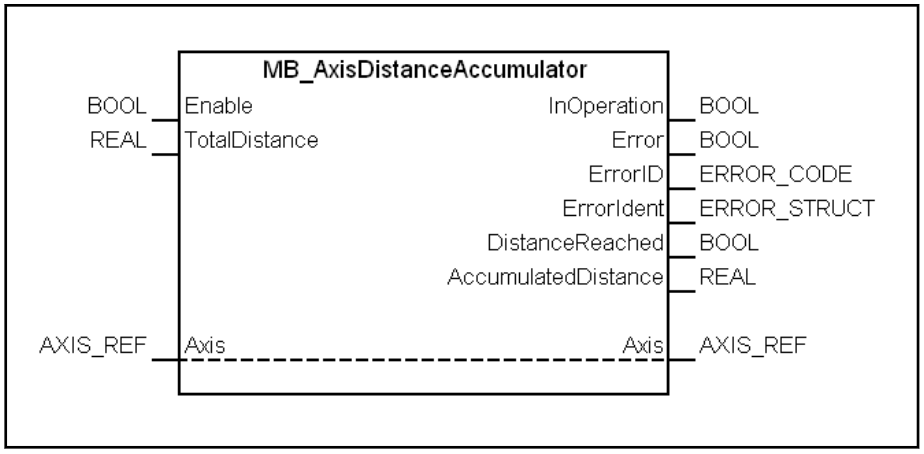


Fig.6-63: MB_AxisDistanceAccumulator function block

I/O type	Name	Data type	Comment
VAR_IN_OUT	Axis	AXIS_REF	Axis reference
VAR_INPUT	Enable	BOOL	Enables the function block with a rising edge
	TotalDistance	REAL	Total distance to be traveled before the process is completed
VAR_OUTPUT	InOperation	BOOL	The function block is currently active
	Error	BOOL	Processing completed with errors
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	DistanceReached	BOOL	TotalDistance was reached and the processing of the function block is completed
	AccumulatedDistance	REAL	Indicates the currently traveled distance

Fig.6-64: Interface variables of the MB_AxisDistanceAccumulator function block

Error Handling

The function block generates the following error messages in Additional1/Additional2 for the "**F_RELATED_TABLE**".

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#0003	16#0000	The function block aborted by another function block
STATE_MACHINE_ERROR	16#0006	16#0000	Invalid status of the state machine

Fig.6-65: Error codes of the MB_AxisDistanceAccumulator function block

6.6.3 MB_ReachVelocityType02

Brief Description

Calculates the respective jerk, acceleration and target velocity to travel along specified jerk-limited velocity characteristics (e.g. S-curve) and allows up to three changes in the target velocities

Therefore, a jerk limitation from 0% to 100 % can be selected. If the input value is 0, the velocity is linear. If it is 100%, it is shaped as S-curve. The command values for velocity, acceleration and jerk are calculated from the acceleration period, the current velocity, the target velocity and the JerkFactor. The acceleration phase can be stopped via "StopAcc".

ML_TechMotion.library

The target velocity can be changed during the acceleration. New command value for velocity, acceleration and jerk are calculated to reach the new target velocity. A new jerk is always effective in the remaining time. Thus, the created JerkFactor becomes invalid after a change in the target velocity. The complete acceleration period changes only if the new target velocity is very similar to the current velocity.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-66: Reference table

Interface description

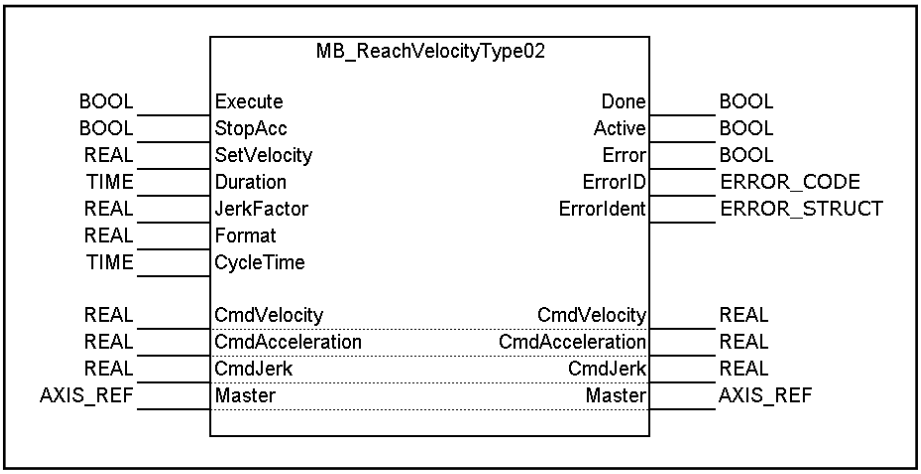


Fig.6-67: MB_ReachVelocityType02 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	CmdVelocity	REAL	Command velocity [rpm]
	CmdAcceleration	REAL	Command acceleration [rad/s2]
	CmdJerk	REAL	Command jerk [rad/s3]
	Master	AXIS_REF	Reference to the master axis It is recommended to create a virtual master axis
VAR_INPUT	Execute	BOOL	Processing of function block enabled
	StopAcc	BOOL	It stops further acceleration
	SetVelocity	REAL	Target velocity [m/min]
	Duration	TIME	Acceleration period
	CycleTime	TIME	Cycle time of the task in which the function block is called
	JerkFactor	REAL	Ramping the acceleration profile [%]
	Format	REAL	Format [mm]
VAR_OUTPUT	Done	BOOL	Processing completed. Data outputs are valid
	Active	BOOL	Function block is working and currently in an interim state
	Error	BOOL	Indicates an error

I/O type	Name	Data type	Description
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

Fig.6-68: Interface description of the MB_ReachVelocityType02

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
StopAcc	BOOL			FALSE	Continuous
SetVelocity	REAL	n.def.	n.def.	0	Continuous between a rising edge at "Execute" and Done = TRUE
Duration	TIME	> 0 s	n.def.	30 s	Rising edge at "Execute"
JerkFactor	REAL	0	100	50	Rising edge at "Execute"
Format	REAL	> 0	n.def.	1000	Rising edge at "Execute"
CycleTime	TIME	> 1 ms	n.def.	10 ms	Rising edge at "Execute"

Fig.6-69: Input behavior of the MB_ReachVelocityType02

Functional Description

The acceleration phase starts at a rising edge at "Execute". The acceleration period (duration), the web velocity to be set (SetVelocity) as well as the acceleration profile (JerkFactor) are applied. A special feature is is that "Active" is set at a rising edge at "Execute" but that the acceleration phase starts only after one second. That means that the following function blocks (e.g. the register control) can use "Active" to see the beginning of the acceleration phase. After the target velocity has been reached, "Active" is also present for one more second until the processing is completed with "Done". The velocity period (Duration), the path velocity to be traveled to (SetVelocity) and the acceleration profile (JerkFactor) are specified for a virtual master axis, since the axis to be referenced at the master (VAR_IN_OUT) has to be a virtual axis to ensure the desired velocity. To represent the curve characteristics at a real axis, it has to be defined as slave axis synchronously to the virtual master axis.

S-curve

The jerk is the temporal derivation of the acceleration. The JerkFactor indicates the temporal part of the acceleration phase in percent in which the jerk is not zero.

The following figure shows three acceleration phases:

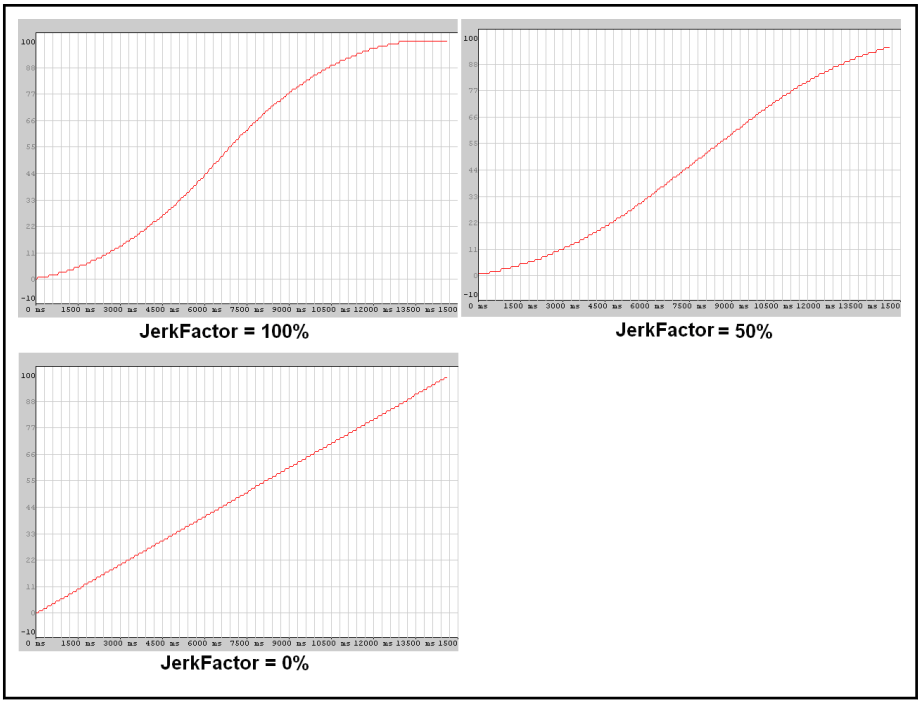


Fig.6-70: Acceleration phases with different JerkFactor

Changes at SetVelocity

After a target velocity change before reaching the end velocity, "CmdVelocity", "CmdAcceleration" and "CmdJerk" are calculated again by assuming the new target velocity and measuring the current time, velocity and acceleration. The function block allows the target velocity to be modified three times.

Figure fig. 6-71 "Effects of target velocity changes" on page 206 shows the master axis speed in rpm of an acceleration phase from 0 to 200 m/min (SetVelocity) in 30 seconds (Duration) and a format of 1 m with a JerkFactor of 30%. If the velocity is higher than 50 m/min, "SetVelocity" is changed to 300 m/min. After the change, the jerk is calculated again and the remaining acceleration phase shows an S-curve characteristic. The original JerkFactor is not considered anymore after the first change.

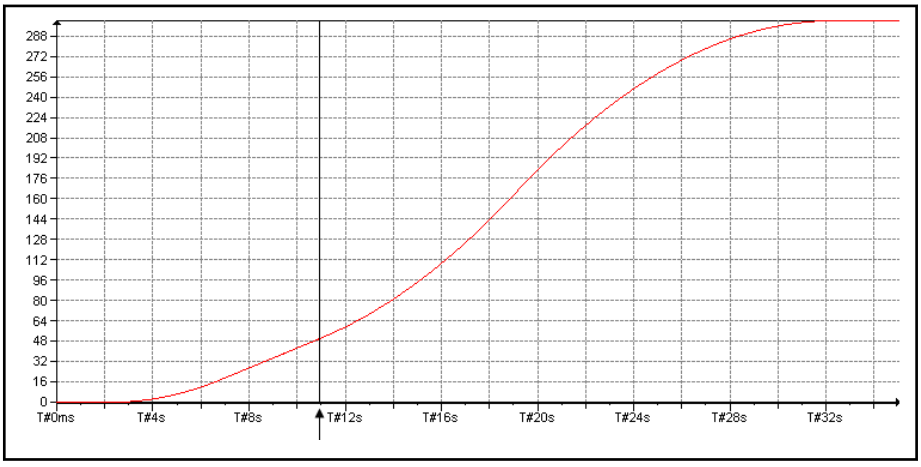


Fig.6-71: Effects of target velocity changes

Stopping the acceleration

The acceleration phase triggered with "Execute" can be immediately stopped with "StopAcc". The function block immediately stops further acceleration. The "Done" output is set. If the set velocity value "SetVelocity" should still be approached, "StopAcc" must be deactivated and another rising edge of

"Execute" has to be set. The reaching of the "SetVelocity" value is signaled by the "Done" output until a falling edge at "Execute".

The following figure shows the abort of the linear and s-curve characteristic acceleration phase via "StopAcc":

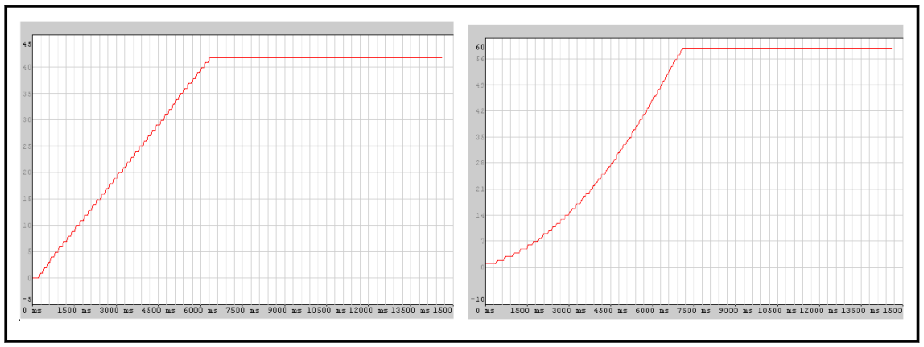


Fig.6-72: Aborting the acceleration phases (left an abort at 6300 ms, right an abort at 7000 ms)

Error handling

The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**, 16#0170:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#00000006	16#00000000	Error in state machine
INPUT_RANGE_ERROR(16#0006)	16#00001D11	16#00000000	Invalid printing cylinder format
INPUT_RANGE_ERROR(16#0006)	16#00001D11	16#00000023	Invalid cycle time
INPUT_RANGE_ERROR(16#0006)	16#00001D11	16#00000024	Invalid period
INPUT_RANGE_ERROR(16#0006)	16#00001D11	16#00000025	Invalid jerk factor
INPUT_INVALID_ERROR(16#0001)	16#0000000D	16#00000000	"AxisNo" for "Master" is not within the valid range

Fig.6-73: Error codes of MB_ReachVelocityType02

6.7 Function Blocks for Application "Position Monitoring"

6.7.1 Introduction and Overview

The position monitoring function blocks monitor the positions of two axes.

Several function block versions are available:

1. The MB_PositionMonitoring_Gantrycant to monitor the relative position of two axes with regard to tilting.
2. The MB_PositionMonitoring_PosCollision to monitor the absolute and relative positions of two axes with regard to indentation and collision.

6.7.2 MB_PositionMonitoring_GantryCant

Brief Description

The function block "MB_PositionMonitoring_GantryCant" monitors the relative position of two axes with regard to tilting. The retraction motion is possible with the help of the tolerance band functionality where the direction of the motion out of the tilting is continuously monitored.

For example, the function block can be used to monitor:

- Infeed axes of Flexo printing units
- Gantry axes with mechanically parallel travel distances, etc.

ML_TechMotion.library

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-74: Reference table of the MB_PositionMonitoring_GantryCant function block

Interface Description

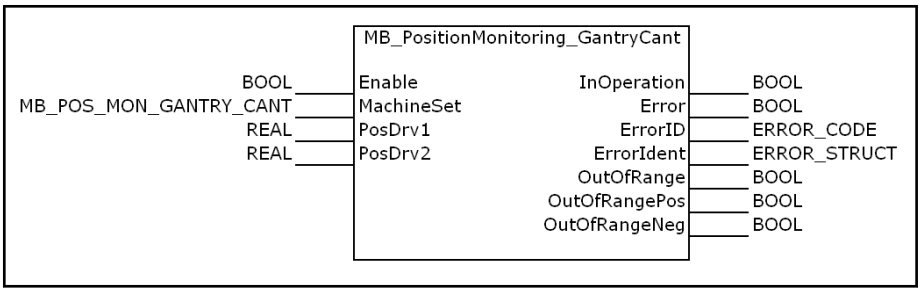


Fig.6-75: Function block MB_PositionMonitoring_GantryCant

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing enabled for function block (cyclic, state-controlled)
MachineSet	MB_POS_MON_ GAN-TRY_CANT	Machine-specific char-acteristics	
PosDrv1	REAL	Current position of axis 1	
PosDrv2	REAL	Current position of axis 2	
VAR_OUTPUT	InOperation	BOOL	Monitoring the axes completed without errors; output signals valid
Error	BOOL	Monitoring of the axes completed with errors, output variable "ErrorIdent" valid	
ErrorID	ERROR_CODE	Describing diagnostics in case of error	
ErrorIdent	ERROR_STRUCT	Describing diagnostics in case of error See "Error Handling" on page 210	
OutOfRange	BOOL	Permissible range is exceeded	

I/O type	Name	Data type	Description
OutOfRangePos	BOOL	Permissible range is exceeded. Axis position of axis 1 is higher than or equal to the axis position of axis 2	
OutOfRangeNeg	BOOL	Permissible range is exceeded. Axis position of axis 1 is smaller than the axis position of axis 2	

Fig.6-76: Interface variables of the MB_PositionMonitoring_GantryCant function block

Data Types

The data type MB_POS_MON_GANTRY_CANT represents the machine set of the function block "MB_PositionMonitoring_GantryCant". The structure contains the following data:

Structure element	Type	Min. value	Max. value	Default value	Description
Range	REAL	>= 2*Tolerance	n.def.	10.0	Maximum position difference allowed between the two axes
Tolerance	REAL	0.0	n.def.	5.0	Tolerance of the limits

Fig.6-77: Data structure of MB_POS_MON_GANTRY_CANT

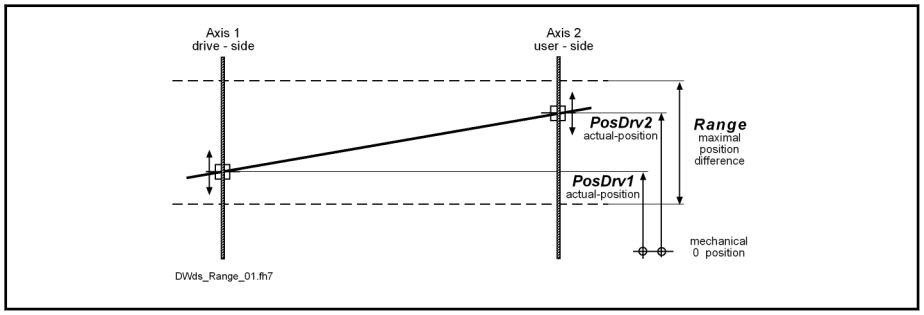


Fig.6-78: Printing unit top view – Maximum position difference allowed (range)

The specified tolerance has to fulfill the following condition:

$$Range > 2*Tolerance$$

Fig.6-79: Condition machine set type 01

Functional Description

The function block "MB_PositionMonitoring_GantryCant" cyclically monitors the positions of two axes with regard to tilting after enabling the process via "Enable". If the distance of the axes exceeds the maximum "range" allowed, this is signaled via the outputs "OutOfRange" and "OutOfRangePos" or "OutOfRangeNeg".

In the following cycle, the output "OutOfRange" is reset. The outputs "OutOfRangePos" or "OutOfRangeNeg" continue to signalize the requirement of a position adjustment until reaching the admissible distance minus the tolerance.

If the maximum distance reached plus the tolerance is exceeded again or even more during this position adjustment within this period, this is signalized by setting the output "OutOfRange" again.

If an error occurs during the processing of the function block, this is signalized via the output "Error" and, in this case, specified by the updated elements of the output structure "ErrorIdent". The other outputs are not updated in case of error.

Error codes In case of error, the "MB_PositionMonitoring_GantryCant" function block generates a comprehensive diagnostics which can be locally interpreted.

Error Handling The function block uses the error table **F_RELATED_TABLE**, 16#0170. It can generate the following error messages in Additional1/Additional2.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs are not within the valid range
STATE_MACHINE_ERROR (16#0005)	16#0006	16#????	Invalid function block state

Fig.6-80: Error codes of the MB_PositionMonitoring_GantryCant function block

6.7.3 MB_PositionMonitoring PosCollision

Brief Description The function block "MB_PositionMonitoring_PosCollision" monitors the absolute position and the relative position of two axes with regard to indentation and collision.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-81: Reference table of the MB_PositionMonitoring_PosCollision function block

Interface Description

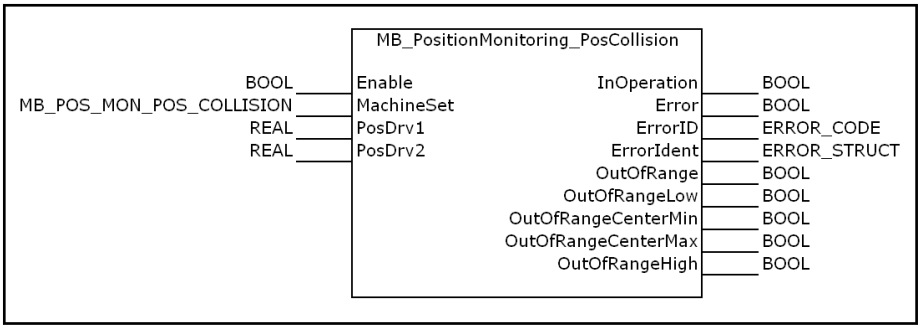


Fig.6-82: MB_PositionMonitoring_PosCollision function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing enabled for function block (cyclic, state-controlled)
	MachineSet	MB_POS_MON_POS_COLLISION	Machine-specific characteristics
	PosDrv1	REAL	Current position of axis 1
	PosDrv2	REAL	Current position of axis 2

I/O type	Name	Data type	Description
VAR_OUTPUT	InOperation	BOOL	Monitoring the axes completed without errors; output signals valid
	Error	BOOL	Monitoring of the axes completed with errors, output variable "ErrorIdent" valid
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Describing diagnostics in case of error See "Error codes" on page 212
	OutOfRange	BOOL	Permissible range is exceeded
	OutOfRangeLow	BOOL	Permissible range is exceeded ("DeltaLow")
	OutOfRangeCenterMin	BOOL	Permissible range is exceeded ("DeltaCenterMin")
	OutOfRangeCenter-Max	BOOL	Permissible range is exceeded ("DeltaCenterMax")
	OutOfRangeHigh	BOOL	Permissible range is exceeded ("DeltaHigh")

Fig.6-83: Interface variables of the MB_PositionMonitoring_PosCollision function block

Data Types

The data type MB_POS_MON_POS_COLLISION represents the machine set of the function block "MB_POS_MON_POS_COLLISION". The structure contains the following data:

Name	Type	Min. value	Max. value	Default value	Description
Position1	REAL	n.def.	n.def.	50.0	Command position of axis 1
Diameter1	REAL	0.0	n.def.	20.0	Diameter of axis 1
Diameter2	REAL	0.0	n.def.	20.0	Diameter of axis 2
DeltaLow	REAL	0.0	n.def.	5.0	Limitation of the position offset of axis 2
DeltaCenter-Min	REAL	0.0	<= DeltaCenter-Max - 2*Tolerance	2.0	Minimum distance between the axes 1 and 2
DeltaCenter-Max	REAL	>= 2*Tolerance	n.def.	5.0	Maximum distance between the axes 1 and 2
DeltaHigh	REAL	>= 2*Tolerance	n.def.	5.0	Limitation of axis position 2
Tolerance	REAL	0.0	n.def.	1.0	Tolerance of the limits

Fig.6-84: Data structure of the MB_POS_MON_POS_COLLISION

The specified tolerance has to fulfill the following condition:

$$Parameters > 2*Tolerance$$

Fig.6-85: Conditions for the machine set PosCollision

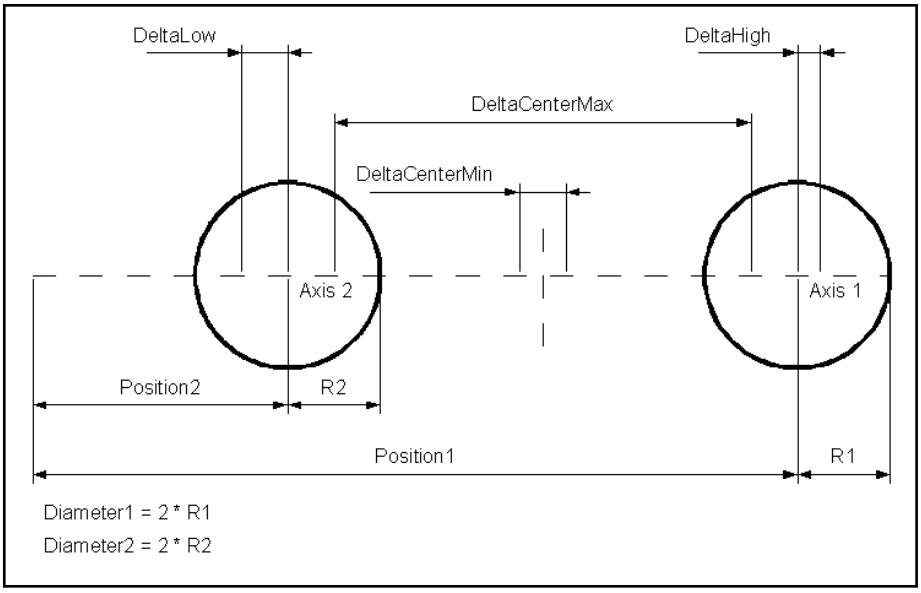


Fig.6-86: Machine set of the MB_PositionMonitoring_PosCollision function block

☞ The following information applies to the "MB_PositionMonitoring_PosCollision" function block:

- The command position of axis 2 results from the parameters of machine set 2
- The current position of axis 1 always has to be higher than the sum of the current position of axis 2 and the radii of both axes
- The function block can be cascaded for three or more axes

Functional Description

The function block "MB_PositionMonitoring_PosCollision" cyclically monitors the positions of two axes with regard to indentation and collision after having enabled processing with the help of "Enable". If the distance of the axes exceeds a maximum admissible range, this is signaled via the outputs "OutOfRange" and the corresponding ranges "OutOfRangeLow", "OutOfRangeCenterMin", "OutOfRangeCenterMax" and/or "OutOfRangeHigh".

In the following cycle, the output "OutOfRange" is reset. The other outputs continue to signal the requirement of a position adjustment until reaching the respective admissible distance minus the tolerance.

If the maximum distance reached plus the tolerance is exceeded again or even more during this position adjustment within this period, this is signalized by setting the output "OutOfRange" again.

If an error occurs during the processing of the function block, this is signalized via the output "Error" and, in this case, specified by the updated elements of the output structure "ErrorIdent". The other outputs are not updated in case of error.

Error codes

In case of error, the function block MB_PositionMonitoring_PosCollision generates a comprehensive analysis which can be locally interpreted.

The function block uses the error table "F-RELATED-TABLE", 16#0170.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0002	16#0000	Inputs are not within the valid range
STATE_MACHINE_ERROR	16#0006	16#????	Inputs are not within the valid range
CALCULATION _ERROR	16#000C	16#0000	Calculation error, wrong axis position

Fig.6-87: Error codes of the MB_PositionMonitoring_PosCollision function block

6.8 Function Block for the MagicBelt Application

6.8.1 Introduction and Overview

In the "MagicBelt" application, products are grouped in a nearly continuous stream and typically delivered to secondary packaging (e.g. cardboard box). "MagicBelt" applications are also known as double drag chains, DualBelt or MagicChain. In this application, each chain is controlled with its own individual instance of the MB_MagicBeltType01 function block described below. The function blocks communicate with each other using the applied data structures.

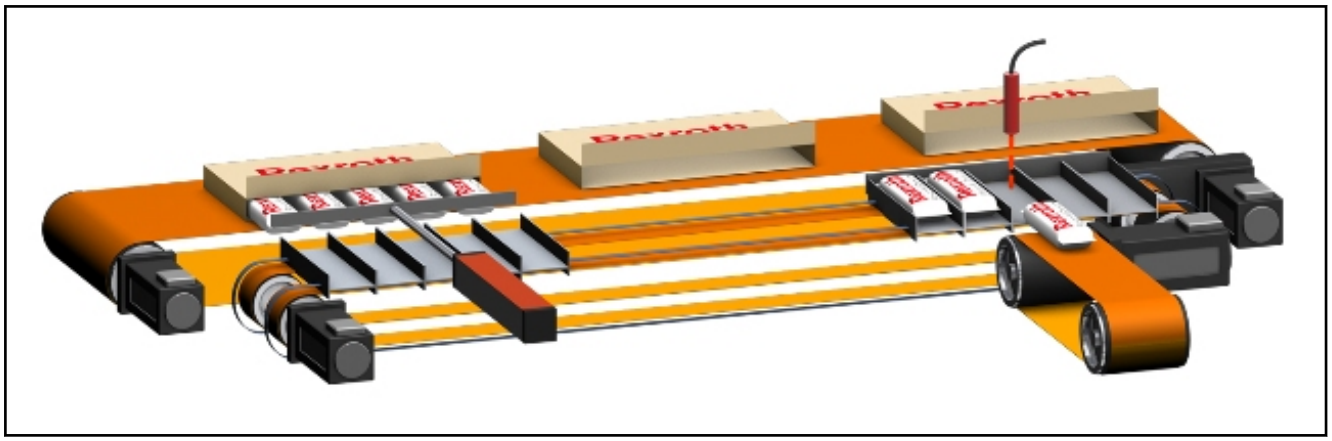


Fig.6-88: MagicBelt application

Definition of terms

The terms are defined as follows:

- Chain: the chain is driven by servo motors. A chain can contain up to 4 trains
- Train: the train is used to group products and contains several pockets.
- Pocket: a pocket can contain one or more products and is a part of a train.
- InfeedMaster: drive axis (or encoder) of the synchronizing wheel for the synchronizing wheel infeed
- InfeedAxis: virtual axis used internally to execute Infeed motion

Definitions and boundary conditions

In practice, several trains are often used per chain because they enable shorter travel motions and lower velocities. Thus, a "smaller" motor can be used which reduces mechanical wear on chain and motor.

If several trains are mounted on a chain, the following is assumed:

- The distance from one train begin to the next train begin is the same (symmetrical train distribution).
- All trains on one chain have the same pocket design (e.g. number and size of pockets).

- All chains are equipped with the same number of trains.

Thus, the modulo value of a chain drive is defined by the distance between the leading edge of one train and the leading edge of the next train (translatory unit). If there is only one train on the chain, the modulo value is equal to the chain length.

The modulo value is determined using the following equation:

$$ModuloValue = \frac{ChainLength}{TrainsPerChain}$$

Fig.6-89: Calculating the modulo value

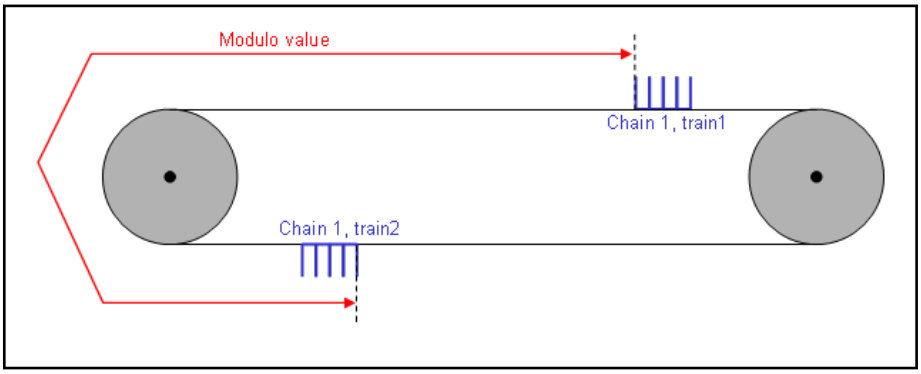


Fig.6-90: MagicBelt definitions: modulo value

6.8.2 MB_MagicBeltType01

Brief Description

The function block contains functionalities to control a chain in a MagicBelt application. The following functional description contains details regarding the function. Depending on the operation mode selected, the respective chain is controlled accordingly:

- **Jog mode:**

  The trains can be jogged individually or synchronously, optionally continuous or incremental.

- **Tune mode:**

  To determine dynamic limit values (e.g. maximum acceleration), the trains are synchronized and continuously travel over a set distance.

- **Automatic mode:**

  The trains travel alternately to the loading and unloading position(s).

The function block operation requires special boundary conditions (e.g. modulo parameterization, cyclic function block call in the Motion task...).

For this reason, please note the commissioning instructions at the end of this section.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-91: Reference table of the MB_MagicBeltType01 function block

Performance characteristics

- A maximum of 3 chains each can be managed (3 function block instances for each chain A,B,C).
- A maximum of 4 trains with equal distances between them can be arranged on one chain.

- A maximum of 40 pockets with **different** pocket widths can be used per train. For several pockets **of the same length** ,the maximum number of pockets increases accordingly; also refer to the structure element MB_MBELT_INFEED.InfeedStepData
- Indexed and synchronous loading are supported.
- A maximum of 10 different unloading positions can be approached.

Interface Description

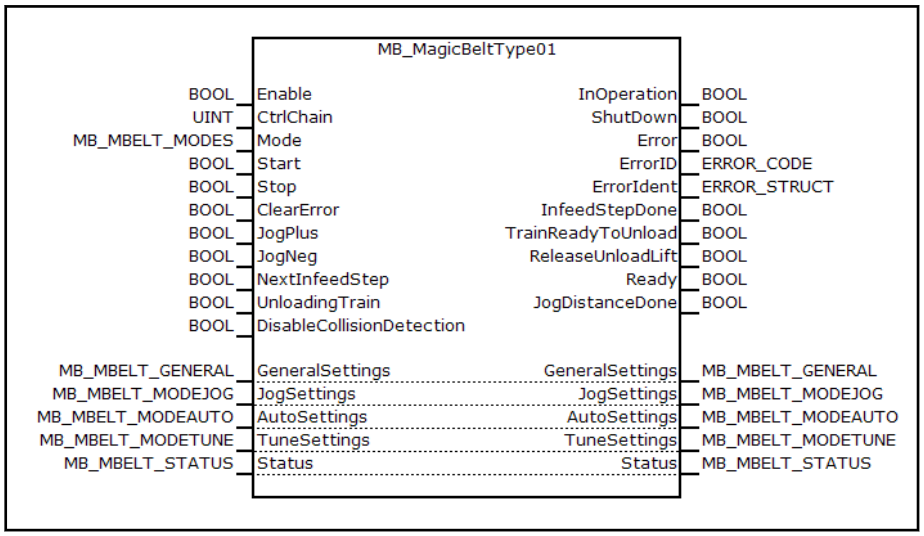


Fig.6-92: MB_MagicBeltType01 function block

The user program controls the chains using the function block inputs (e.g. start, stop...). The configuration data for operation is applied using VAR_IN_OUT data structures (e.g. AutoSettings...). Additional diagnostic information is available via another "Status" VAR_IN_OUT data structure.

I/O type	Name	Data type	Description
VAR_IN_OUT	GeneralSettings	MB_MBELT_GENERAL	General configuration of the MagicBelt application
	JogSettings	MB_MBELT_MODEJOG	Settings for the Jog mode
	AutoSettings	MB_MBELT_MODEAUTO	Settings for the Auto mode
	TuneSettings	MB_MBELT_MODETUNE	Settings for the Tune mode
	Status	MB_MBELT_STATUS	Status information on the chains
VAR_INPUT	Enable	BOOL	Processing enabled for function block (state-controlled)
	CtrlChain	UINT	Selects the chain number to be triggered: 1: Chain A, 2: Chain B, 3: Chain C
	Mode	MB_MBELT_MODES	Operation mode selection Switching between the operation modes is only executed in the "Stopped" state.
	Start	BOOL	Positive edge of "Start" starts the axis motion (also refer to StartType)
	Stop	BOOL	Positive edge of "Stop" stops the axis motion (also refer to StopType)

ML_TechMotion.library

I/O type	Name	Data type	Description
	ClearError	BOOL	Positive edge clears pending error
	JogPlus	BOOL	Jogging the axes in positive direction (only in active Jog mode)
	JogNeg	BOOL	Jogging the axes in negative direction (only in active Jog mode)
	NextInfeedStep	BOOL	For Infeed type "FEED_INDEXED": After a positive edge and after the "StartDelayTime" elapsed, the train is indexed to the next pocket. For Infeed type "FEED_SYNCHRONIZED": A positive edge within the expectation window ("ProductInExpectationWindow" = TRUE) signals a product in the pocket of the "InfeedMaster" Infeed wheel.
	UnloadingTrain	BOOL	As the train is unloaded, the input has to remain set until the unloading is completed. The train remains at the unloading position as long as this signal is set. A negative edge results in the train indexed to the next unloading or loading position.
	DisableCollisionDetection	BOOL	Switches off the collision monitoring, e.g. after a collision in order to move the chains away from each other
VAR_OUTPUT	InOperation	BOOL	Processing without errors, output data is valid.
	ShutDown	BOOL	Defined shutdown after a negative edge at "Enable"
	Error	BOOL	Signals error state
	ErrorID	ERROR_CODE	Error diagnosis
	ErrorIdent	ERROR_STRUCT	Detailed error diagnostics
	InfeedStepDone	BOOL	Output only for the function block of chain A (CtrlChain = 1 ChainA). Output is only relevant for Infeed variant "FEED_INDEXED". The output is set if the InfeedStep was executed. The output is reset with a positive edge of NextInfeedStep
	TrainReadyToUnload	BOOL	Train is at unloading position and ready to be unloaded.
	ReleaseUnloadLift	BOOL	Unloader enabled
	Ready	BOOL	Only output for function block with "CtrlChain" = 1 (ChainA): The output is set if the incoming product stream from MagicBelt can be absorbed, see also product jam monitoring
	JogDistanceDone	BOOL	The output is set if the specified distance was covered during incremental jogging. The output is reset if "JogPlus"/"JogNeg" is reset

Fig.6-93: Interface variables of the MB_MagicBeltType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
CtrlChain	UINT	1	3	1	Pos. edge at "Enable"
Mode	MB_MBELT_MODES	MODE_AUTO	MODE_JOG	MODE_AUTO	Continuous
Start	BOOL			FALSE	Continuous in the Auto mode and in the Tune mode
Stop	BOOL			FALSE	Continuous in the Auto mode and in the Tune mode
ClearError	BOOL			FALSE	Continuous in the Error mode
JogPlus	BOOL			FALSE	Continuous in the Jog mode
JogNeg	BOOL			FALSE	Continuous in the Jog mode
NextInfeedStep	BOOL			FALSE	Continuous in the Auto mode
UnloadingTrain	BOOL			FALSE	Continuous in the Auto mode
DisableCollision-Detection	BOOL			FALSE	Continuous

Fig.6-94: Min./max. and default values of the MB_MagicBeltType01 function block

MB_MBELT_MODES

The "MB_MBELT_MODES" enumeration contains the operation modes that can be selected for the MagicBelt function block.

Element	Value	Description
MODE_AUTO	16#100	Auto mode
MODE_TUNE	16#300	Tune mode
MODE_JOG	16#D00	Jog mode

Fig.6-95: MB_MBELT_MODES enumeration type elements

Used data structures/overview

The MB_MagicBeltType01 function block uses the following data structures. The data structures are shown below.

ML_TechMotion.library

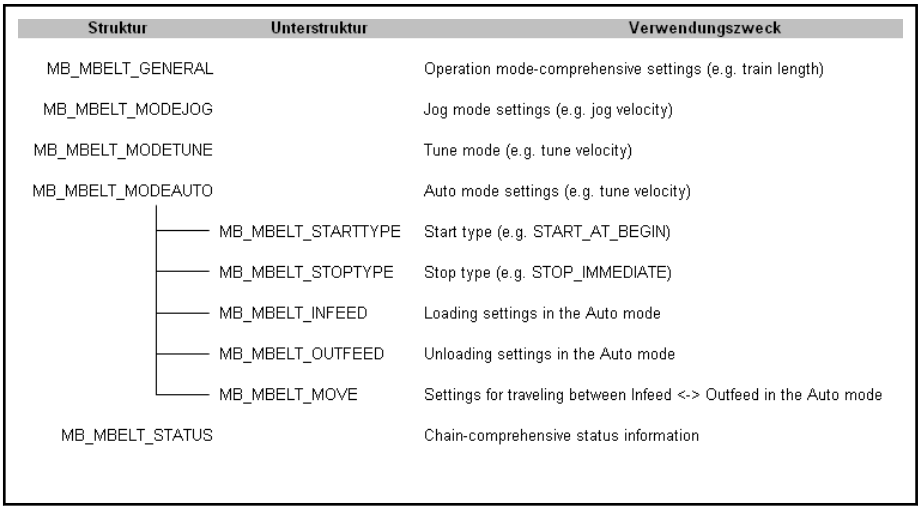


Fig.6-96: Data structures of the MagicBelt function block

MB_MBELT_GENERAL The "MB_MBELT_GENERAL" contains chain-comprehensive and operation mode-comprehensive settings:

Structure element	Type	Default value	Effective	Description
ChainLength	REAL	4000.0	pos. edge "Enable"	Chain length [mm]; Minimum = 1 Max = undefined
NoOfChains	UINT	2	pos. edge "Enable"	Number of chains Minimum = 1 Maximum = 3
TrainsPerChain	UINT	1	pos. edge "Enable"	Trains per chain Minimum = 1 Maximum = 4
DischargeBackwards	BOOL	FALSE	Continuous	Determines the direction in which the trains are discharged. FALSE: Travels across 2/3 of the belt length in the positive direction TRUE: Travels across 2/3 of the belt length in the negative direction
AllowNegDirection	BOOL	FALSE	Continuous	Allows the motion of the trains in negative direction
TrainDistance	REAL	5.0	Continuous	A train that lags behind makes up this distance to catch up with a train that is ahead (application in Automatic mode) [mm]. The TrainDistance has to be specified greater than the SafetyDistance because otherwise the collision monitoring is triggered. Minimum = 0.0mm Maximum = Modulo value/4

Structure element	Type	Default value	Effective	Description
SafetyDistance	REAL	0.5	Continuous	Collision monitoring safety distance [mm] Minimum = -10mm Maximum = 100mm
StopRamp	REAL	10000.0	Continuous	Stop ramp deceleration in case of error reactions or stop with the stop variant "STOP_IMMEDIATE". The Stop ramp has to be higher than or equal to the maximum value of all other acceleration (MaxMoveAcceleration; MaxInfeedAcceleration; MaxOutfeedAcceleration). Minimum = see description above Maximum = undefined
StartAtLastStepWindow	REAL	100	Continuous	In the start option "StartAtLastStep", the current axis positions are compared with the positions saved after reaching standstill. Only if the position difference for all axes is less than the range specified here, the procedure continues according to the start option "StartAtLastStep". If at least one axis is located outside of this window, the procedure continues with the start option "Start_At_Begin". Minimum: 0mm Maximum: ModuloValue
Feedrate	UINT	100	Continuous	All velocities in the "Auto Mode" are reduced accordingly if "FeedrateOverride" < 100% is specified. Minimum: 10% Maximum: 100%
InfeedMaster	AXIS_REF	NO_OBJECT	pos. edge "Enable"	Defines the master axis for synchronous loading with wheel infeed. Only necessary if synchronous loading is selected.
InfeedAxis	AXIS_REF	NO_OBJECT	pos. edge "Enable"	For the virtual axis "Infeed axis" used for the Infeed motion
DummyMaster	AXIS_REF	NO_OBJECT	pos. edge "Enable"	For internal "FlexProfile" positioning, a dummy axis has to be applied here. The axis type and settings are not relevant. The axis is not actively controlled and can be used for other purposes on the application side. It is needed for a valid axis reference
ChainAxes	ARRAY [1.0.3] OF AXIS_REF		pos. edge "Enable"	Axis assignment for the chains ChainAxes[1] = Chain A ChainAxes[2] = Chain B ChainAxes[3] = Chain C (optional)

Fig.6-97: Interface description and default value of the MB_MBELT_GENERAL data structure

ML_TechMotion.library

MB_MBELT_MODEJOG The data structure "MB_MBELT_MODEJOG" contains the comprehensive chain data for the Jog mode.

Structure element	Type	Default value	Effective	Description
Incremental	BOOL	FALSE	Pos. edge JogPlus/JogNeg	TRUE: With a positive edge at "JogPlus"/"JogNeg", the "JogDistance" is traveled either in positive or in negative direction. FALSE: Travel in positive or negative direction as long as "JogPlus"/"JogNeg" is enabled.
SingleAxisJog	BOOL	FALSE	Pos. edge JogPlus/JogNeg	FALSE: All trains are jogged synchronously. TRUE: Only the train selected by "JogAxisSelection" is jogged.
JogAxisSelection	UINT	1	Pos. edge JogPlus/JogNeg	Axis selection for single axis jogging 1: Chain A, 2: Chain B, 3: Chain C
JogVelocity	REAL	100.0	Pos. edge JogPlus/JogNeg	Jog acceleration
JogAcceleration	REAL	1000.0	Pos. edge JogPlus/JogNeg	Jog acceleration
JogDistance	REAL	2.0	Pos. edge JogPlus/JogNeg	Jog distance for incremental jogging

Fig.6-98: Interface description and default value of the MB_MBELT_MODEJOG data structure

MB_MBELT_MODETUNE The data structure "MB_MBELT_MODETUNE" contains the comprehensive chain data for the Tune mode.

Structure element	Type	Default value	Effective	Description
PosDirectionOnly	BOOL	FALSE	At motion start	TRUE: It is traveled relatively in the positive direction with "TuneDistance" travel. FALSE: "TuneDistance" travels alternately forward and backward from the start position.
MotionLaw	MC_STEP_MODE	X_FIT_SINE_TRAPEZE_ALIM	At motion start	Specifies the motion law for a Tune positioning procedure. Available motion laws are: • X_FIT_VEL_TRAPEZE_ALIM • X_FIT_SINE_TRAPEZE_ALIM • X_FIT_ACC_TRAPEZE_JLIM • REST_IN_REST_INCLINEDSINE • REST_IN_REST_MOD_TRAPEZE • X_MOTION_IN_MOTION_POLY5_VLIM Detailed information on the motion laws and the MC_STEP_MODE data type are in the FlexProfile documentation.
TuneVelocity	REAL	1000.0	At motion start	Maximum positioning velocity
TuneAcceleration	REAL	100.0	At motion start	Maximum positioning acceleration

Structure element	Type	Default value	Effective	Description
TuneJerk	REAL	200.0	At motion start	Tune jerk limitation (only effective for motion laws with limited jerk)
TunePosTime	REAL	500	At motion start	Tune positioning time, effective for all motion laws except: • X_FIT_VEL_TRAPEZE_ALIM • X_FIT_SINE_TRAPEZE_ALIM • X_FIT_ACC_TRAPEZE_JLIM
TuneDistance	REAL	10.0	At motion start	Traveling distance per positioning procedure

Fig.6-99: Interface description and default value of the MB_MBELT_MODETUNE data structure

MB_MBELT_MODEAUTO

The data structure "MB_MBELT_MODEAUTO" contains the comprehensive chain data for the Automatic mode.

Structure element	Type	Default value	Description
StartType	MB_BELT_START-TYPE	START_AT_BEGIN	Determines the start behavior for the Automatic mode. Refer to the MB_MBELT_STARTTYPE description.
StopType	MB_BELT_STOPTYPE	STOP_CYCLE_COMPLETE	Determines the stop behavior for the Automatic mode. Refer to the "MB_MBELT_STOPTYPE" description. Application at positive stop or negative edge at "Enable"
Infeed	MB_BELT_INFEED		Contains data to load the trains
Outfeed	MB_BELT_OUTFEED		Contains data to unload the trains
MoveTo	MB_BELT_MOVE		Contains data for the motion between loading and unloading

Fig.6-100: Interface description and default value of the MB_MBELT_MODEAUTO data structure

MB_MBELT_STARTTYPE

The "MB_MBELT_STARTTYPE" enumeration contains the possible start variants for the Automatic mode:

Element	Value	Description
START_AT_BEGIN	1	Defined start at reset position
START_WITH_TRAIN_UNLOAD	2	Defined start from reset position with upstream train unloading
START_AT_LAST_STEP	3	Start with the last (interrupted) state (e.g. after error or stop)

Fig.6-101: Element of the MB_MBELT_STARTTYPE enumeration type

MB_MBELT_STOPTYPE

The "MB_MBELT_STOPTYPE" enumeration contains the possible stop variants for the Automatic mode:

ML_TechMotion.library

Element	Value	Description
STOP_IMMEDIATE	1	Immediate stop with the MB_MBELT_GENERAL.StopRamp ramp
STOP_CYCLE_COMPLETE	2	Active product cycle (Infeed Step) is still executed. Afterwards, it stops (defined standstill). All trains not synchronized with the InfeedAxis (e.g. for movements between Outfeed positions) are stopped immediately
STOP_IMMEDIATE_SYNC	3	This stop variant is only useful for applications with synchronized infeed. This stop variants acts like STOP_IMMEDIATE except that the axes that are in a synchronous relation with the InfeedMaster remained synchronized. When the external InfeedMaster decelerates, these axes decelerate as well. The deceleration of the InfeedMaster has to be ensured outside of the function block

Fig.6-102: Element of the MB_MBELT_STOPTYPE enumeration type

MB_MBELT_INFEED

The "MB_MBELT_INFEED" data structure contains the comprehensive chain data for loading in the Automatic mode:

Structure element	Type	Default value	Effective	Description
InfeedType	MB_MBELT_FEEDTYPE	FEED_INDEXED	pos. edge "Enable"	Loading procedure (indexed, synchronized, with velocity, see "MB_MBELT_FEEDTYPE" enumeration)
StartDelayTime	TIME	T#50ms	Pos. edge at "NextInfeedStep"	Only relevant for indexed loading: Delay time (pos. edge "NextInfeedStep"--> start motion) for indexing Minimum = T#0ms Maximum = T#1h
InfeedLoadTime	TIME	T#10ms	Pos. edge at "NextInfeedStep"	Only relevant for indexed loading: Time required for train loading. Within this period, the train remains at standstill so that the product is not damaged during loading. Input is not effective for "MotionLaw": X_FIT_ACC_TRAPEZE_JLIM X_FIT_SINE_TRAPEZE_ALIM X_FIT_VEL_TRAPEZE_ALIM Since in this case the specified a, v, j maximum values are used
MaxProductRate	UINT	200	Pos. edge at "NextInfeedStep"	Only relevant for indexed loading: Maximum number of products per minute that are to be indexed in [1/min] (product stream). Input is not effective for "MotionLaw": X_FIT_ACC_TRAPEZE_JLIM X_FIT_SINE_TRAPEZE_ALIM X_FIT_VEL_TRAPEZE_ALIM Since in this case the specified a, v, j maximum values are used

Structure element	Type	Default value	Effective	Description
MotionLaw	MC_STEP_MODE	VELOCITY_IN_VELOCITY_MOD_SINE	Pos. edge at "NextInfeedStep"	Only relevant for indexed loading: Specifies the motion law for an Infeed step (for a detailed description of the motion laws, see the FlexProfile documentation). The following motion laws are recommended for MagicBelt applications: • VELOCITY_IN_VELOCITY_MOD_SINE: Modified sinusoid/low jerk, but high acceleration peak • X_VELOCITY_IN_VELOCITY_TRAPEZE_ALIM: Modified acceleration trapezoid/strongly affected by jerk but has the lowest acceleration peaks • X_FIT_ACC_TRAPEZE_JLIM: Jerk-limited motion. The "MaxProductRate" input is not effective for this motion law because the a, v, j maximum values are used. Attention: For "REST_IN_REST_XXX" motion laws, the previous Infeed step has to be completed before a new Infeed step is started. The error Add1 = 16#1A0F is output if this condition is not met
MinDistanceToOutfeed	REAL	400.0	Continuous	Minimum distance between a train in the "Infeed " state and another train in the "Outfeed " state. If the distance does not meet the lower limit, the "Ready" output is reset to prevent a product jam; for details, see product jam monitoring.
MaxInfeedVelocity	REAL	300000.0	Continuous	Infeed step maximum velocity: Only effective for "MotionLaw": X_FIT_ACC_TRAPEZE_JLIM X_FIT_SINE_TRAPEZE_ALIM X_FIT_VEL_TRAPEZE_ALIM X_MOTION_IN_MOTION_POLY5_VLIM X_VELOCITY_IN_VELOCITY_TRAPEZE_ALIM
MaxInfeedAcceleration	REAL	5000.0	Continuous	Infeed step maximum acceleration: Only effective for "MotionLaw": X_FIT_ACC_TRAPEZE_JLIM X_FIT_SINE_TRAPEZE_ALIM X_FIT_VEL_TRAPEZE_ALIM X_VELOCITY_IN_VELOCITY_TRAPEZE_ALIM
MaxInfeedJerk	REAL	1000000.0	Continuous	Infeed step maximum jerk: Only effective for "MotionLaw": X_FIT_ACC_TRAPEZE_JLIM

ML_TechMotion.library

Structure element	Type	Default value	Effective	Description
ProductBuffer	UINT	3	pos. edge "Enable"	Only for synchronized loading: Number of products between product detection and product ejection Minimum: 0 Maximum: 64
ProdExpectationWindowStart	REAL	30.0	Continuous	Only for synchronized loading: Start of the product expectation window. The "NextInfeedStep" is evaluated within the expectation window. Minimum = 0° Maximum = ProdExpectationWindowEnd
ProdExpectationWindowEnd	REAL	180.0	Continuous	Only for synchronized loading: End of the product expectation window. The "NextInfeedStep" is evaluated within the expectation window. Minimum = ProdExpectationWindowStart Maximum = 360°
FirstInfeedPosition	REAL	500.0	Continuous	First (absolute) loading position Minimum = 0 Maximum = ModuloValue
AdditionalSteps	UINT	0	Continuous	With "AdditionalSteps", a train that is already completely loaded can be moved one or more steps ahead of the next train before it travels to the Outfeed position (e.g. until hold-down is reached).
InfeedStepData	ARRAY [1..3, 1..40] OF MB_BELT_INFEEDSTEPS		Continuous	List with the size of the pockets and product number per pocket for all chains. "InfeedStepData"[**1**,x] = chain **A** "InfeedStepData"[**2**,x] = chain **B** "InfeedStepData"[**3**,x] = chain **C** (optional) For synchronized loading it is always assumed that there is one product per pocket

Fig.6-103: Interface description and default value of the MB_MBELT_INFEED data structure

MB_MBELT_FEEDTYPE

The "MB_MBELT_FEEDTYPE" enumeration contains the possible train loading and unloading procedures:

Element	Value	Description
FEED_INDEXED	1	Indexed loading
FEED_SYNCHRONIZED	2	Synchronized loading

Fig.6-104: Elements of the MB_MBELT_FEEDTYPE enumeration type

MB_MBELT_INFEEDSTEPS

The "MB_MBELT_INFEEDSTEPS" data structure contains the specific chain data for loading in Automatic mode:

Structure element	Type	Default value	Effective	Description
PocketWidth	REAL	50.0	Continuous	Distance from the leading edge of the current pocket to the leading edge of the following pocket. If several pockets are the same size, the number of pockets of the same size can be specified using the "PocketQuantity" element
PocketQuantity	UINT	1	Continuous	Number of pockets of the same size -1 = identifier for group formation (see example) 0 = identifier for train end (see example)
ProductsPer-Pocket	UINT	0	Continuous	Number of products per pocket (only relevant for indexed loading)

Fig.6-105: Interface description and default value of the MB_MBELT_INFEED-STEPS data structure

Example:

Example 1: Same size pockets (also refer to the figure below):

Train consists of 10 same size pockets 50 mm wide with one product each

ChainA_InfeedData[1].PocketWidth = 50mm

ChainA_InfeedData[1].PocketQuantity = 10

ChainA_InfeedData[1].ProductsPerPocket = 1

ChainA_InfeedData[2].PocketWidth = don’t care

ChainA_InfeedData[2].PocketQuantity = 0 <- Train end ID

ChainA_InfeedData[2].ProductsPerPocket = don’t care

Example:

Example 2: Different sized pockets (also refer to the figure below):

The train consists of 3 different pockets (40 mm, 80 mm, 120 mm) with (1,2,3) products per pocket

ChainA_InfeedData[1].PocketWidth = 40mm

ChainA_InfeedData[1].PocketQuantity = 1

ChainA_InfeedData[1].ProductsPerPocket = 1

ChainA_InfeedData[2].PocketWidth = 80mm

ChainA_InfeedData[2].PocketQuantity = 1

ChainA_InfeedData[2].ProductsPerPocket = 2

ChainA_InfeedData[3].PocketWidth = 120mm

ChainA_InfeedData[3].PocketQuantity = 1

ChainA_InfeedData[3].ProductsPerPocket = 3

ChainA_InfeedData[4].PocketWidth = don’t care

ChainA_InfeedData[4].PocketQuantity = 0 <- Train end ID

ChainA_InfeedData[4].ProductsPerPocket = don’t care

Example:

Example 3: Pocket group (also refer to the figure below):

The train consists of a pocket group with 5 * 3 different pockets (40mm, 80mm, 120mm) with (1,2,3) products per pocket

ChainA_InfeedData[1].PocketWidth = 40mm

ChainA_InfeedData[1].PocketQuantity = -1 <- group identification

ChainA_InfeedData[1].ProductsPerPocket = 1

ChainA_InfeedData[2].PocketWidth = 80mm

ChainA_InfeedData[2].PocketQuantity = -1 <- group identification

ChainA_InfeedData[2].ProductsPerPocket = 2

ChainA_InfeedData[3].PocketWidth = 120mm

ChainA_InfeedData[3].PocketQuantity = 5 <-group end identification (value > 0) with the number of groups

ChainA_InfeedData[3].ProductsPerPocket = 3

ChainA_InfeedData[4].PocketWidth = don’t care

ChainA_InfeedData[4]. PocketQuantity = 0 <- train end identification

ChainA_InfeedData[4]. ProductsPerPocket = don’t care

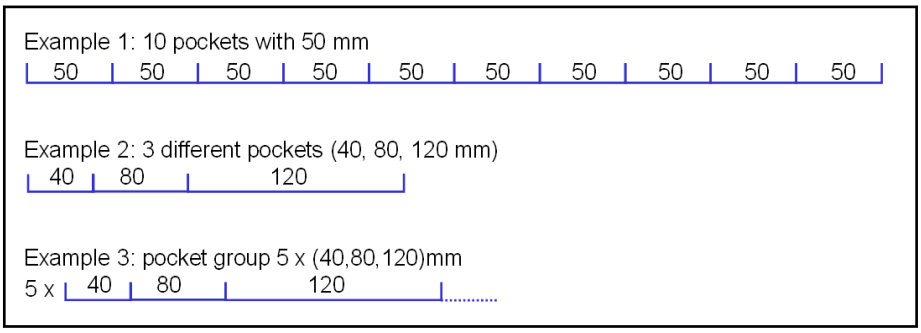


Fig.6-106: Train configurations of the examples shown above

MB_MBELT_OUTFEED

The "MB_MBELT_OUTFEED" data structure contains the comprehensive chain data for unloading in the Automatic mode:

Structure element	Type	Default value	Effective	Description
MotionLaw	MC_STEP_MODE	X_FIT_SINE_TRAPEZE_ALIM	Pos. edge at "UnloadingTrain"	Specifies the motion law for the motion to the next Outfeed position (for a detailed description of the motion laws refer to the FlexProfile documentation) Acceleration-limited motion (X_FIT_VEL_TRAPEZE_ALIM) Jerk-limited motion (X_FIT_ACC_TRAPEZE_JLIM) Acceleration-limited sinoide (X_FIT_SINE_TRAPEZE_ALIM)
MaxOutfeedVelocity	REAL	60000.0	Continuous	Maximum velocity to travel to the next unloading position
MaxOutfeedAcceleration	REAL	5000.0	Continuous	Maximum acceleration to travel to the next unloading position
MaxOutfeedJerk	REAL	0.0	Continuous	Maximum jerk to travel to the next unloading position Only effective for "MotionLaw" = X_FIT_ACC_TRAPEZE_JLIM

Structure element	Type	Default value	Effective	Description
FirstOutfeedPosition	REAL	2000.0	Continuous	First (absolute) unloading position
EnableUnloadHandshake	BOOL	FALSE	Continuous	FALSE: Unloading handshake disabled. For applications with no collision risk between the unloading device and the train (e.g. unloading robots) TRUE: Unloading handshake enabled. For applications with collision risk between the unloading device and the train. "UnloadLiftLowPos" and "UnloadLiftHighPos" have to be set accordingly (e.g. unloaders)
OutfeedDistances	ARRAY [1..3, 2..10] OF REAL	0.0	Continuous	This list is only necessary if more than one unloading position is to be approached. In this case, this list can be used to specify the distances from one unloading position to the following unloading position. Each negative edge of the "UnloadingTrain" input causes an indexing to the next unloading position. The end of the list is reached if the value 0 is entered. "OutfeedDistances"[**1**,x] = chain **A** "OutfeedDistances"[**2**,x] = chain **B** "OutfeedDistances"[**3**,x] = chain **C** (optional)
UnloadLiftLowPos	ARRAY [1...10] OF BOOL	TRUE	Continuous	Unloader position limit switch/TRUE = unloader retracted
UnloadLiftHighPos	ARRAY [1...10] OF BOOL	TRUE	Continuous	Unloader position limit switch/TRUE = unloader extended

Fig.6-107: Interface description and default value of the MB_MBELT_OUTFEED data structure

MB_MBELT_MOVE The "MB_MBELT_MOVE" data structure contains the comprehensive data for traveling between the loading and unloading positions.

Structure element	Type	Default value	Effective	Description
MaxMoveVelocity	REAL	300000.0	At the Move motion start	Maximum velocity for traveling between loading and unloading stations and for synchronization
MaxMoveAcceleration	REAL	10000.0	At the Move motion start	Maximum acceleration/deceleration for traveling between loading and unloading stations
MaxMoveJerk	REAL	0.0	At the Move motion start	Maximum jerk for traveling between loading and unloading stations
LoadTrainAcceleration	UINT	100	At the Move motion start	Percentage of the maximum acceleration (MaxMoveAcceleration) or deceleration used during the motion between the loading station and the first unloading station. Thus, a fully loaded train can travel with reduced acceleration

ML_TechMotion.library

Structure element	Type	Default value	Effective	Description
BeginReduced-Vel1	REAL	0.0	At the Move motion start	From this position onwards (the leading edge of the train) the train travels with reduced velocity. This can reduce the centrifugal forces generated by changing directions in curves
EndReduced-Vel1	REAL	0.0	At the Move motion start	From this position onwards (the leading edge of the train), the velocity limitation is cancelled.
BeginReduced-Vel2	REAL	0.0	At the Move motion start	From this position onwards (the leading edge of the train) the train travels with reduced velocity. This can reduce the centrifugal forces generated by changing directions in curves
EndReduced-Vel2	REAL	0.0	At the Move motion start	From this position onwards (the leading edge of the train), the velocity limitation is cancelled.
ReducedVelocity	REAL	100000.0	At the Move motion start	Velocity limitation for reducing centrifugal forces in curves
AdaptiveSyncRamp	BOOL	TRUE	At the Move motion start	Enables an adaptive synchronization ramp

Fig.6-108: Interface description and default value of the MB_MBELT_MOVE data structure

MB_MBELT_STATUS The " MB_MBELT_STATUS" data structure contains status information on the train:

Structure element	Type	Description
ChainStates	ARRAY [1..3] OF MB_MBELT_MODES	Current actual state of the chains
PocketsToBeLoaded	UINT	A train in the "Auto Infeed" state outputs the number of pockets still to be loaded in the train via this structure element. Thus, it can be calculated at a higher level how long the loading is likely to last
ActInfeedStep	UINT	A train in the "Auto Infeed" state outputs information on the "Infeed" step currently processed via this structure element.
ProductInExpectationWindow	BOOL	This output is used in the "FEED_SYNCHRONIZED" Infeed mode if the "InfeedMaster" axis is located in the expectation window. The expectation window is defined by the structure elements "MB_BELT_INFEED", "ProdExpectationWindowStart" and "MB_MBELT_INFEED" and "ProdExpectationWindowEnd"
AllChainsHomed	BOOL	All chains are homed.
Error	BOOL	Chain-comprehensive error flag
ErrorSourceChain	UINT	Error-causing chain number
MeasuredInfeedStepRate	REAL	Measured InfeedStepRate [min-1] (determined from the temporal gap of the "NextInfeedStep" edges)
MaxProdExpectationWindowEnd	REAL	Maximum possible end position of the expectation window "ProdExpectationWindowEnd" calculated from the current velocity of the InfeedMaster axis [°]
ChainInAutoInfeed	UINT	Chain number located in the "AUTO_INFEED" state

Structure element	Type	Description
SupressedInfeedSteps	UINT	Number of the suppressed InfeedSteps. Incremented in the loading variant "FEED_INDEXED" if a "NextInfeedStep" edge is applied and the Infeed train cannot execute the Infeed Step (for diagnostics purposes)
MaxNoOfInfeedSteps	UINT	Maximum number of Infeed steps for the current train at the Infeed position
FirstMasterChain	UINT	Number of chains that first reach the FirstInfeedPosition (for internal purposes only)
OutfeedPosOccupied	ARRAY [1...10] OF BOOL	Set for every Outfeed position if it is occupied by a train or the unloader
OccupiedChain	ARRAY [1...10] OF UINT	Chain number which occupies the Outfeed position
ChainCmdStates	ARRAY [1..3] OF MB_MBELT_STATES	Calculated target state (for internal purposes only)
RestoreData	ARRAY [1..3] OF MB_MBELT_RESTORE	Saved information at the point in time when stop occurs to be able to restore the most recent state at the "StartType" = " START_AT_LAST_STEP"

Fig.6-109: Interface description and default value of the MB_MBELT_STATUS data structure

Functional Description

With "Enable" = TRUE, the function block starts and the selected operation mode selected at "Mode" is enabled after initialization. The chain is controlled according to the selected operation mode. Switching between operation modes is only allowed in the secure "Stopped" state when the chains are at standstill. The operation modes and their functionality are described below.

The following inputs are independent of the operation modes and are always effective:

- Enable
- Mode
- CtrlChain
- ClearError
- DisableCollisionDetection

The following outputs are independent of the operation modes and are always effective:

- InOperation
- Error
- ErrorID
- ErrorIdent
- Shutdown

The following inputs/outputs are independent of the operation modes and are always effective:

- GeneralSettings
- Status

☞ All Chain axes have to be referenced (using Enable=TRUE) when the function block starts. If this condition is not met, the function block reports an error.

Jog operation mode

In the Jog operation mode, the following inputs/outputs are enabled:

Function block inputs:

- JogPlus
- JogNeg

Function block outputs:

- JogDistanceDone

Function block inputs/outputs:

- JogSettings

When setting up the machine, the trains can be jogged continuously or incrementally with the Jog operation mode. After enabling the Jog operation mode, the trains are at standstill (Stopped state). To prevent collisions, jogging the axes is locked with the optional unloader. For this reason, jogging is only possible if:

- The function block input "UnloadingTrain" is reset AND
- The optional unloaders are retracted, i.e. the unloader limit switch signals are: "MB_MBELT_OUTFEED. UnloadLiftLowPos[1..10]" = TRUE

If the above named condition is met, the trains can be jogged in positive or negative direction using the "JogPlus" or "JogNeg" inputs. The following options are possible:

- The "MB_MBELT_MODEJOG.Incremental" structure element can be used to switch between incremental and continuous jogging (default setting = continuous jogging).
- The "MB_MBELT_MODEJOG.SingleAxisJog" structure element can be used to enable single axis jogging. The train to be jogged is selected using the "MB_MBELT_MODEJOG.JogAxisSelection" structure element (default setting = synchronized jogging).

Tune operation mode

In the "Tune" operation mode, the following inputs/outputs are enabled:

Function block inputs:

- Start
- Stop

Function block inputs/outputs:

- TuneSettings

When setting up the machine, the optimum dynamic settings/motion laws can be determined with the Tune operation mode. In permanent start/stop operation, the trains are moved either in one direction or in a bipolar fashion according to the selected motion law. As in the Jog operation mode, the axis start is locked with the optional unloader. For a detailed description, see Jog operation mode.

After a positive edge of "Start", the chains are synchronized and travel continuously and cyclically over the distance "MB_MBELT_MODETUNE.TuneDistance". Depending on the setting of "MB_MBELT_MODETUNE.PosDirectionOnly", the trains travel either continuously in positive direction or alternately forward and backward. During the motion, the dynamic settings (e.g. "MB_MBELT_MODETUNE.TuneAcceleration" can be modified in order to determine the maximum acceleration values for example.

Auto operation mode

In the Auto operation mode, the following inputs/outputs are enabled:

Function block inputs:

- Start

- Stop
- NextInfeedStep
- UnloadingTrain

Function block outputs:

- InfeedStepDone
- TrainReadyToUnload
- ReleaseUnloadLift
- Ready

Function block inputs/outputs:

- AutoSettings
- Status

In Automatic mode, the trains are moved alternately between the loading and unloading positions in order to take and group the almost continuous product stream at the loading position. If the train is fully loaded, the unloading positions are approached in order to allow an unloader or an unloading robot to remove the product groups. After unloading, the respective train approaches the loading position again.

Start The positive "Start" edge enables train motion. Depending on the start option ("AutoSettings.StartType") these are the starting processes:

START_AT_BEGIN:

The train closest to the "MB_MBELT_INFEED.FirstInfeedPosition" is the leading train and therefore the master. It is also considered whether the trains may only move in positive direction or in a bipolar fashion (which can be set via "MB_MBELT_GENERAL.AllowNegDirection").

With the train distance "MB_MBELT_GENERAL.TrainDistance" all trains catch up to the leading train. The leading train travels to the "MB_MBELT_INFEED.FirstInfeedPosition" and all other trains follow synchronously. After the "FirstInfeedPosition" is reached, loading the train begins.

START_WITH_TRAIN_UNLOAD:

The procedure is similar to START_AT_BEGIN except that an additional train unloading motion is executed at about 2/3 *"MB_MBELT_GENERAL.ChainLength" before the movement to the "FirstInfeed" position in order to unload the trains safely. The direction of the unloading motion can be affected by the structure element "MB_MBELT_GENERAL.DischargeBackwards". The unloading motion is executed with the maximum velocity "MB_MBELT_MOVE.MaxMoveVelocity".

START_AT_LAST_STEP:

The most recent state after an error or stop can be restored with this start option. This start option first checks if all axis positions are still located within the adjustable window "MB_MBELT_GENERAL.StartAtLastStepWindow" (in comparison to the position saved after stop). If at least one of the axes is located outside of this window, the "START_AT_BEGIN" procedure is used. If all axes are located within this window, in the next step all axes are repositioned to the position most recently active before stop or an error. Then, the most recent state before stop or an error is restored. To complete the restoration, the axes are repositioned if necessary. For this reason, this start option can only be executed if the following boundary conditions are met:

- Position difference between the current actual position and the most recent position of the chain axes at standstill lies within the

"MB_MBELT_GENERAL.StartAtLastStepWindow". If this condition is not met, START_AT_BEGIN is executed

- • The last states saved before the interruption of all chain axes correspond with the Auto mode If this condition is not met, START_AT_BEGIN is executed
- If an unloading device is available (EnableUnloadHandshake=TRUE) and at least one unloader is extended (UnloadLiftHighPos[n] = TRUE) and a repositioning of more than 2 mm at at least one axis is required, "START_AT_LAST_STEP" cannot be executed because mechanical damage can occur. In this case an error is output

Repositioning is carried out as follows:

- Chains are synchronized (relative synchronization)
- Only if 3 chains are configured: Chain C is positioned at the last valid position saved. All other chains follow synchronously. Continues when chain C has reached the target position
- • Chain B is positioned at the last valid position saved Chain C (if present) remains at standstill. Chain A follows synchronously. Continues when chain B has reached the target position
- • Chain A is positioned at the last valid position saved. All other chains remain at standstill. Continues when chain A has reached the target position
- The most recent state and all necessary internal variables are restored in order to arrive at the state existing before the interruption

Loading

All settings for the loading procedures are specified using the structure element "MB_MBELT_MODEAUTO.Infeed".

In principle, indexed loading of the train is differentiated from synchronized loading. With indexed loading, the train moves along the loading position using the function block input "NextInfeedStep" until all pockets of the train are filled.

With synchronized loading, the train is synchronized with the product feed during the loading procedures (product feed with an Infeed wheel, for example). In general, this loading procedure treats the products more gently so higher index rates are possible. With this procedure, the pocket width is always assumed to be the same. If a product feed pocket is empty, during loading the train must synchronously skip a pocket width (compensation for empty section).

The train loading procedure is determined by the structure element "MB_MBELT_MODEAUTO.Infeed.InfeedType".

Indexed loading

With indexed loading, all elements in the data structure MB_MBELT_MODEAUTO.Infeed are in effect (exception: ProductBuffer and ProductExpectationWindow are not effective).

With indexed loading (AutoSettings.Infeed.InfeedType = FEED_INDEXED), the products arrive on conveyor belts with more or less the same distance between them although there is a minimum spacing distance. Even when the distances seem to be the same, one or more products can be missing. On the way into the pocket, the product triggers a sensor signal to indicate that the train can move on. This signal is applied at the function block input "NextInfeedStep".

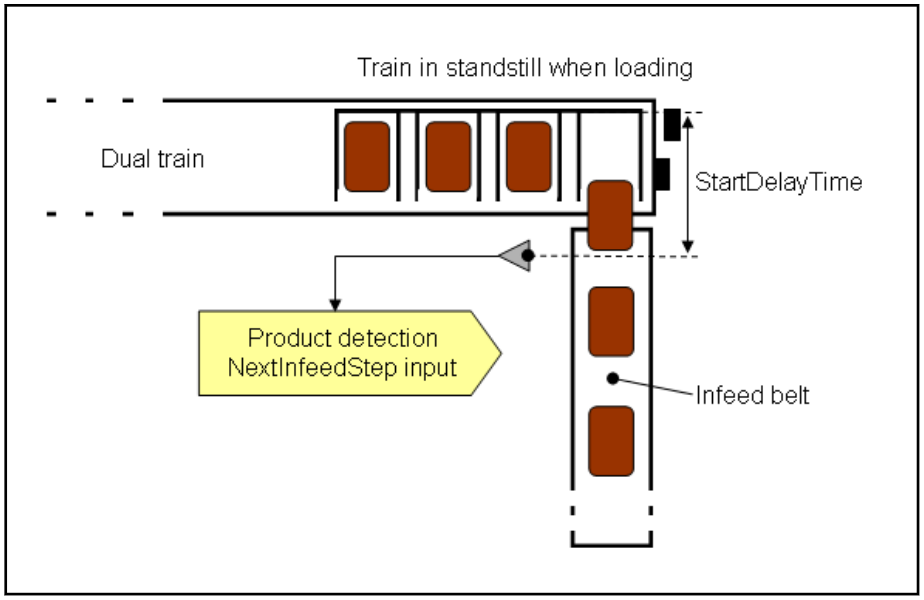


Fig.6-110: Indexed product feed/loading at standstill from the front

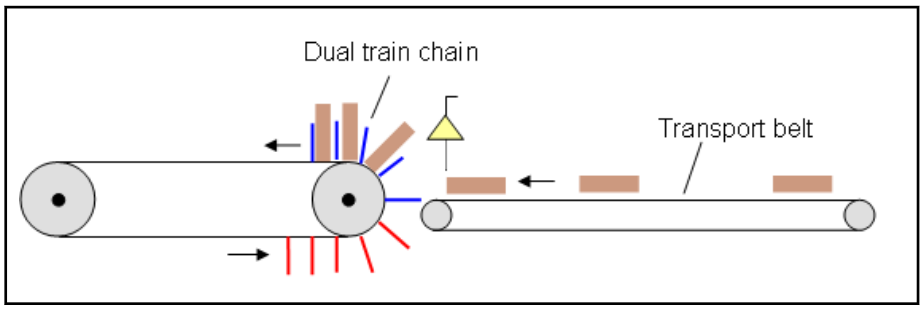


Fig.6-111: Indexed product feed/loading at standstill from the side

The transport distance can be set from the detection of the sensor signal up to the start of the indexing motion using the adjustable time "AutoSettings.Infeed.StartDelayTime". After the continuing motion is completed, the function block output "InfeedStepDone" is set for acknowledgement. The pocket width and the products per pocket can vary from pocket to pocket and is specified by the structure element "MB_MBELT_MODEAUTO.Infeed.InfeedStepData". The feed motion here is determined by the following structure elements (see also the following figure):

- "MB_MBELT_MODEAUTO.Infeed.MotionLaw" specifies the motion law used
- "MB_MBELT_MODEAUTO.Infeed.StartDelayTime" specifies the time delay until the Infeed motion starts
- "MB_MBELT_MODEAUTOInfeed.InfeedLoadTime"specifies the time that the train is to remain at standstill (while it is being loaded)
- "MB_MBELT_MODEAUTO.Infeed.MaxProductRate" specifies the maximum number of products per minute

The selected motion law "MB_MBELT_MODEAUTO.Infeed.MotionLaw" has great bearing on the Infeed motion executed. The following has to be observed:

- In principle, "VELOCITY_IN_VELOCITY..." MotionProfiles should be used if possible because they also allow the train to move forward with velocity (if the previous Infeed step is not yet fully completed, for exam-

ple). In special cases, all "Rest-in-Rest" FlexProfile motion laws can be used if it is certain that the train to be loaded reaches standstill after each Infeed step. The recommended Infeed motion laws are:

– Modified sinusoid (low jerk, but high acceleration peaks) ("VELOCITY_IN_VELOCITY_MOD_SINE")
– Modified acceleration trapezoid (strongly affected by jerk but has the lowest acceleration peaks) ("X_VELOCITY_IN_VELOCITY_TRAPEZE_ALIM")
– A jerk-limited motion is executed with the "X_FIT_ACC_TRAPEZE_JLIM" motion law. The "MB_MBELT_MODEAUTO.Infeed.MaxProductRate" input is not effective for this motion law because the a, v, j maximum values are used.

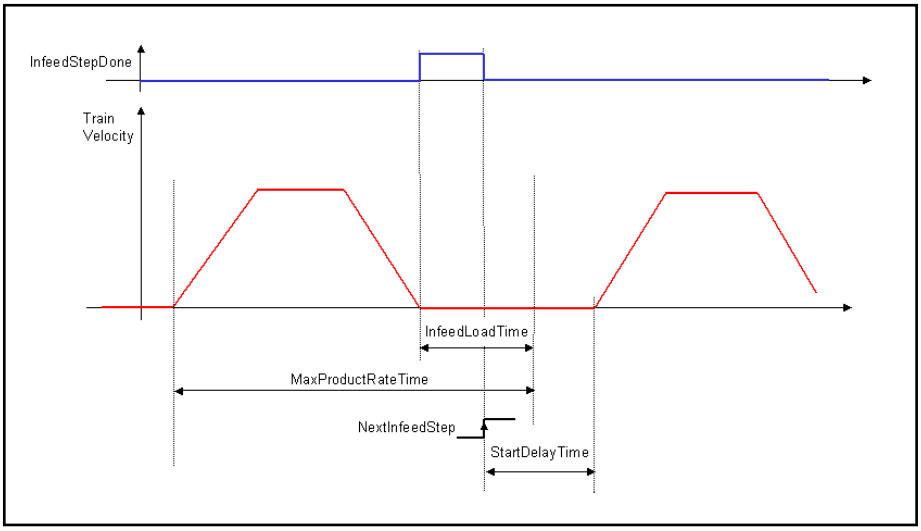


Fig.6-112: Overview and mode of operation of InfeedLoadTime, StartDelayTime and MaxProductRate

When values for "MB_MBELT_MODEAUTO.Infeed.StartDelayTime" are specified accordingly, the train can also move during the loading procedure. In this way applications in which the train is loaded while moving can be implemented (see following figure).

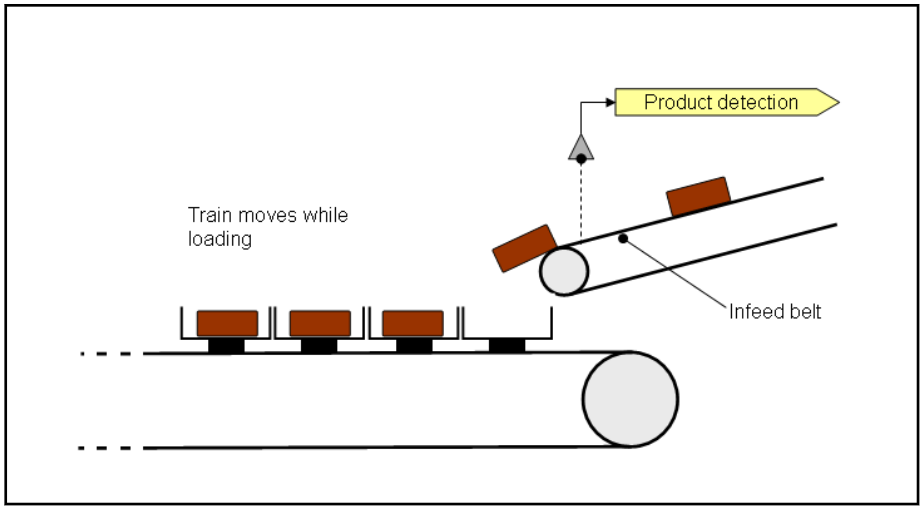


Fig.6-113: Indexed product feed/loading when the train is traveling

For the "FEED_INDEXED" loading variant, the product stream is typically specified for the system using the "NextInfeedStep" input. The train has to follow the specified product cycle. Typically for these applications, a train always has to be available at the Infeed to accept the products.

However, there are also applications in which the available products are collected and the train "requests" the products (example: carton transport applications). The MagicBelt function block specifies the cycle rate with the "InfeedStepDone" output. For these applications it is not mandatory that a train be at the loading position. These applications can also be implemented with the MagicBelt function block. The following procedures is recommended for these use cases:

If the AND operation "InfeedStepDone" AND "Ready" = TRUE, a train is at the Infeed and requests a product. The product (e.g. a cardboard box) should be moved into the train pocket. When the loading is completed, the train can move on with a positive edge at "NextInfeedStep". If the loading time is nearly constant, the InfeedLoadTime can be set accordingly and the NextInfeedStep input can be wired as follows:

NextInfeedStep := Ready AND InfeedStepDone AND ProductsAvailable

With this assignment, the train automatically moves farther as long as products are available (ProductsAvailable=TRUE). The "ProductsAvailable" signal is calculated in the application program.

Infeed with constant velocity

In principle, all "VELOCITY_IN_VELOCITY..." Infeed motion laws (such as X_VELOCITY_IN_VELOCITY_TRAPEZE_ALIM) allow continuing motion in the next "Infeed Step" if the train is still moving based on the previous "Infeed Step" (i.e. velocity > 0).

If the specified time intervals of the positive edges at the "NextInfeedStep" input are shorter than the specified maximum product rate "MB_MBELT_INFEED.MaxProductRate", the train no longer reaches standstill (prerequisite: MB_MBELT_INFEED.InfeedLoadTime = 0). When the "NextInfeedStep" time

intervals become progressively shorter, the velocity undershoots also become progressively smaller. If the positive edges at the "NextInfeedStep" input are applied at double the product rate "MB_MBELT_INFEED.MaxProductRate" (or higher), the train travels at a constant velocity.

Example:

MB_MBELT_MODEAUTO.Infeed.MaxProductRate = 60 products/min. --> 1 product is equivalent to 1 sec. --> The train at the Infeed moves at a constant velocity if a positive edge is applied at "NextInfeedStep" every 0.5 sec.

The structure element "MB_MBELT_STATUS.MeasuredInfeedStepRate" can be used to check the frequency of the positive edges at the "NextInfeedStep" input for diagnostic purposes. In the example mentioned above, NextInfeedStep = 120 is displayed. The train to be loaded moves then at a constant velocity that corresponds with 120 products/minute.

Synchronous loading

With synchronized loading, the following elements in the data structure MB_MBELT_MODEAUTO.Infeed are effective:

- ProductBuffer
- ProductExpectationWindowStart
- ProductExpectationWindowEnd
- AdditionalSteps
- MinDistanceToOutfeed

With synchronous loading, the train is synchronized with the product feed during the loading procedure. In this case, the product feed is carried out using a continuously rotating Infeed wheel ("MB_MBELT_GENERAL.InfeedMaster"), which moves farther along with each product in a master cycle (= 360°). A product sensor is wired to the "NextInfeedStep" input and it recognizes if a product is in the Infeed wheel compartment. The production recognition is enabled if the product is located within the expectation window range (ProdExpectationWindowStart ... ProdExpectationWindowEnd) relative to the position of the InfeedMaster. The detected product is entered in a shift register, which cycles with each master cycle. With each master cycle, the train travels one pocket length farther if a product is contained in the Infeed wheel compartment. If no product is in the compartment, the train skips it accordingly so no empty pocket results. In case of an empty pocket, the train cannot remain exactly at the "InfeedPosition" because the deceleration path still has to be travelled. Then the train remains in place for half of a compartment (synchronous skipping). Several products can be contained in the areas between product recognition and train loading. The number of these products is specified for the function block using "MB_MBELT_MODEAUTO.Infeed.ProductBuffer". With synchronous loading it is also possible to set different pocket sizes, but it is always assumed that there is one product per pocket. The optional adjustable train distance is added to the last Infeed step on the pocket width in order to compensate for this.

In the special case "MB_MBELT_MODEAUTO.INFEED.PRODUCTBUFFER" = 0 the product recognition is located directly in front of the train loading area. In this case, the product is included in the same master cycle and the motion law required for the next cycle is prepared. The preparation of the motion law for the next master cycle requires 3 SERCOS cycles. The end of the expectation window "MB_MBELT_INFEED.ProdExpectationWindowEnd" may not become greater than the calculated output value "MB_MBELT_STATUS.MaxProdExpectationWindowEnd". With the specified SERCOS cycle time and maximum velocity of the InfeedMaster, this value can also be calculated externally with the following formula:

$$\varphi_{MaxExpectWindowEnd}[°] = 360° - 3 \bullet t_{SercosCycle}[ms] \bullet v_{InfeedMaster}[\min^{-1}] \bullet 0,006$$

Fig.6-114: MB_MBELT_STATUS.MaxProdExpectationWindowEnd calculation

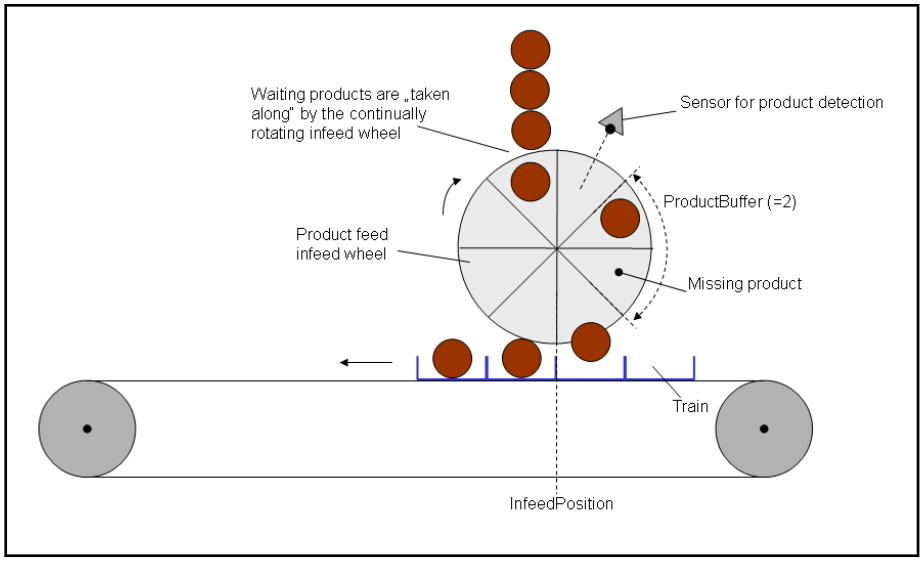


Fig.6-115: Synchronous product feed

Example for synchronous product feed (temporal sequence):

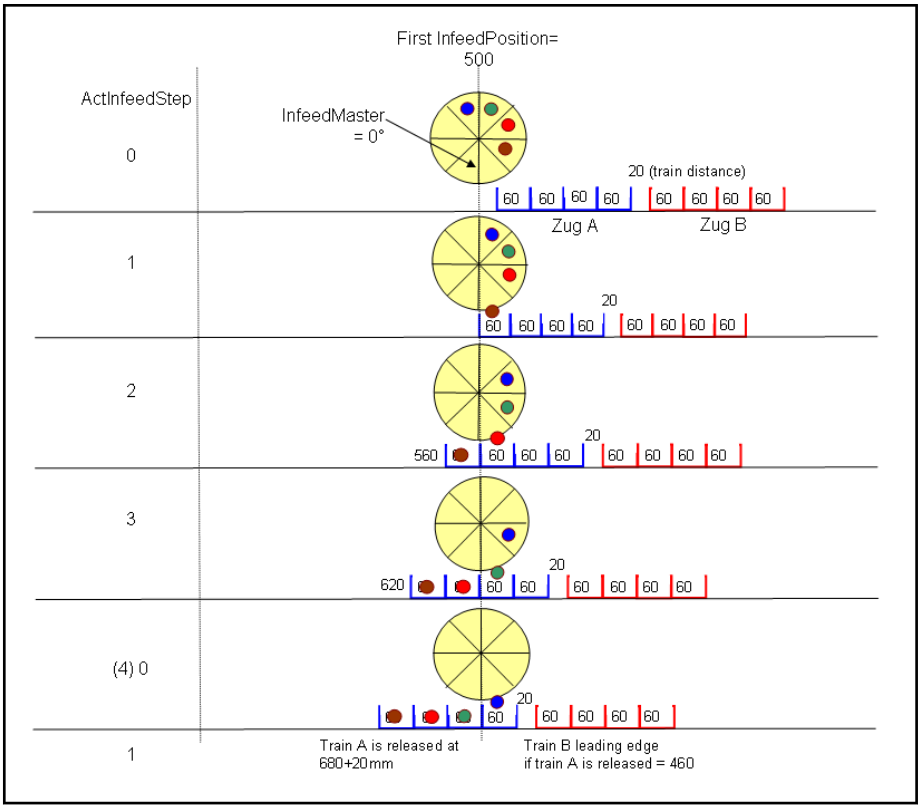


Fig.6-116: Synchronous product feed (temporal sequence)

Movements between the loading and unloading stations

After the train is loaded or unloaded it has to approach the next station as quickly as possible. The movements between the loading and unloading stations are specified using the "MB_MBELT_MOVE" data structure. Here the effective acceleration for a loaded train can be initiated using the structure element "MB_MBELT_MOVE.LoadTrainAcceleration". Furthermore, the velocity can be reduced in the areas with curves to reduce the centrifugal forces generated.

☞ The "MB_MBELT_MOVE" velocity/acceleration has to be set such that it is greater than or equal to the maximum Infeed velocity/acceleration. Otherwise, there is a risk of collision if a train is released from the loading motion. If the settings are wrong, the dynamic monitoring reports an error in order to avoid collisions.

Unloading

All settings for the unloading procedures are specified using the structure element "MB_MBELT_MODEAUTO.Outfeed".

To unload, first the train travels to the first absolute unloading position ("MB_MBELT_MODEAUTO.Outfeed.FirstOutfeedPosition"). The function block sets the output "TrainReadyToUnload" when the train reaches the "Outfeedposition".

A negative edge at the function block input "UnloadingTrain" results in the train moving to the next unloading or loading position. Further unloading positions can be specified using the structure element "MB_MBELT_MODEAUTO.Outfeed.OutfeedDistances". The distances specified here correspond to the distance from the most recent unloading position. If distance = 0 (default) is specified, following a negative edge at the function block input "UnloadingTrain" the "FirstInfeedPosition" is approached again.

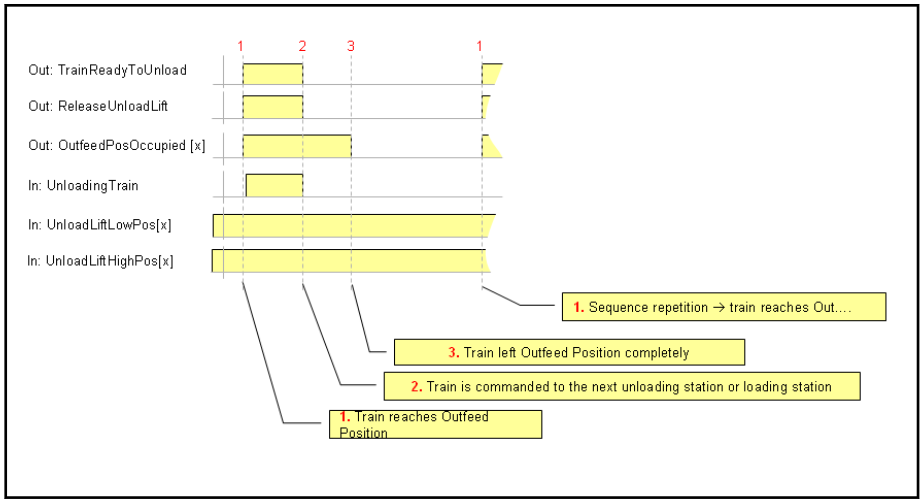


Fig.6-117: Signal-time diagram when unloading without unload handshake

If the train is unloaded by an unloader, the motion of the train and the unloader have to be locked to avoid collisions between train and unloader. The unloading handshake is enabled using the structure element "MB_MBELT_MODEAUTO.Outfeed.EnableUnloadHandshake".

In the simplest case the train does not start moving to the next unloading or loading position until the unloader has been retracted. The corresponding handshake and the signal-time diagram for this case are shown in the following figure.

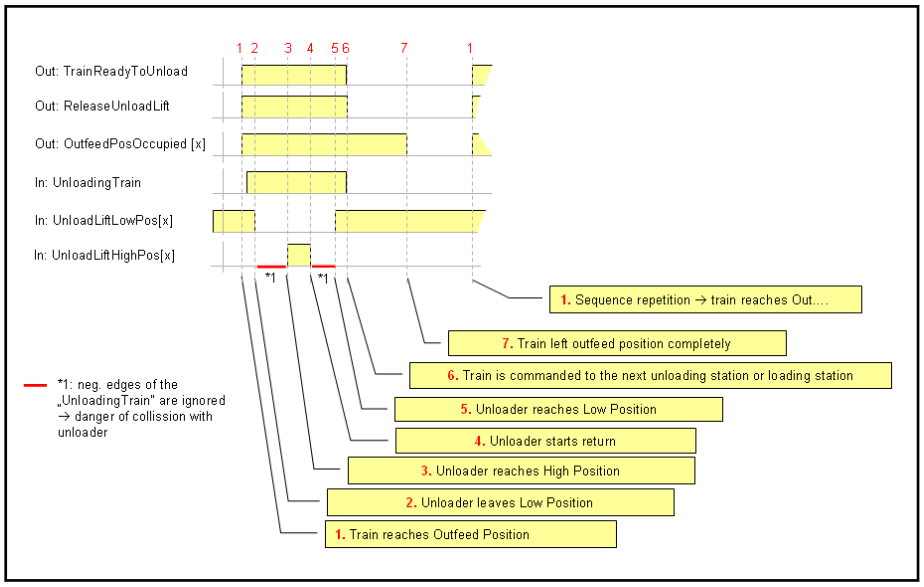


Fig.6-118: Signal-time diagram for unloading with handshake/continuing train travel when unloader is retracted

In another use case, the train moves already even though the unloader is fully extended. In this case the unloader may only be retracted when the train has travelled entirely beyond the unloader's penetration area. The unloading position remains occupied until the unloader is completely retracted. The corresponding handshake and the signal-time diagram for this case are shown in the following figure.

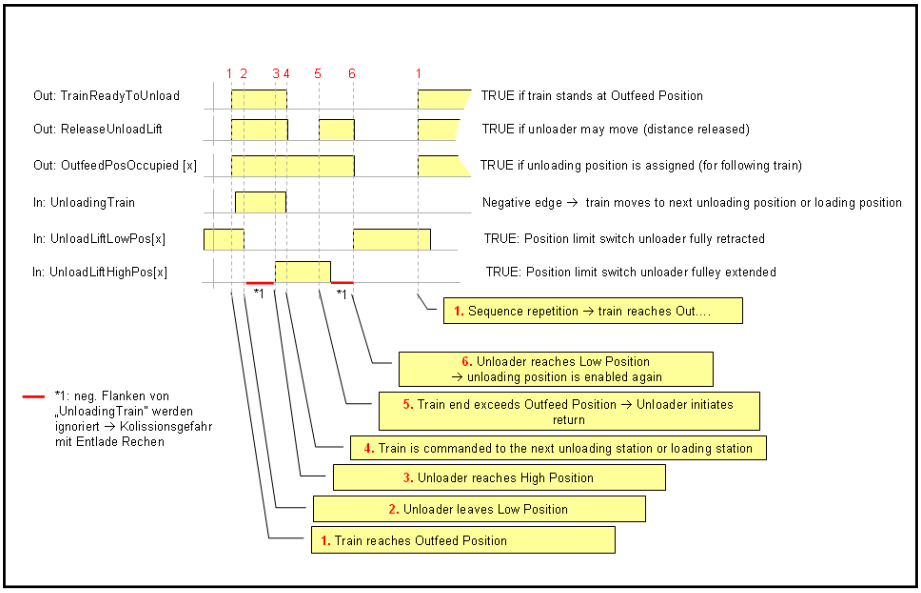


Fig.6-119: Signal-time diagram for unloading with handshake/continuing train travel when unloader is extended

StopBat

With a positive edge at the "Stop" input, the train is stopped according to the specified stop variant ("MB_MBELT_MODEAUTO. StopType"). With the synchronous Infeed variant, it is recommended that the chains be stopped by decelerating the InfeedMaster and then finally stopping the chains using StopType = STOP_IMMEDIATE.

STOP_IMMEDIATE:

In this stop variant, all trains and the InfeedAxis are decelerated immediately using the "MB_MBELT_GENERAL.StopRamp" ramp (e.g. for fatal disturbances).

STOP_CYCLE_COMPLETE:

In this stop variant, the active product cycle (Infeed step) is still executed. Afterwards, the stop is carried out (defined standstill). All trains not synchronized with the InfeedAxis (e.g. for movements between Outfeed positions) are stopped immediately.

STOP_IMMEDIATE_SYNC :

This stop variant is only useful for applications with synchronized infeed. This stop variants acts like STOP_IMMEDIATE except that the axes that are in a synchronous relation with the InfeedMaster remained synchronized. When the external InfeedMaster decelerates, these axes decelerate as well. The deceleration of the InfeedMaster has to be ensured outside of the function block.

Error reaction

In case of an error, all axes are stopped immediately using "MB_MBELT_GENERAL.StopRamp", as long as they can still follow the command values (procedure as with "STOP_IMMEDIATE"). Furthermore, the "Error" output is set and error codes are output to the corresponding function block outputs. The error acknowledgement can be carried out with a rising edge at the "ClearError" input. Depending on the StartType selected, after the error is cleared, the Automatic mode can resume at the place of interruption ("MB_MBELT_STARTTYPE" = START_AT_LAST_STEP) or from the defined reset position ("MB_MBELT_STARTTYPE" = START_AT_BEGIN).

Product jam monitoring

Unplanned deceleration (e.g. during unloading) can cause situations in which the continuous product stream can no longer be absorbed and a product jam

can occur. Such situations have to be recognized as early as possible in order to restrict or interrupt the product stream.

For this reason, product jam monitoring is enabled in the first train (train A), which sets the "Ready" output when the following condition occurs:

- There is a train at the loading station

AND

in instances **without** an unloading device ("MB_MBELT_INFEED.EnableUnloadHandshake" = FALSE):

- the distance between a train that was just loaded (train in "Infeed" state and other trains at the "Outfeed" positions is greater than "MB_MBELT_INFEED.MinDistanceToOutfeed"

in instances **with** an unloading device ("MB_MBELT_INFEED.EnableUnloadHandshake" = TRUE):

- the distance between a train that was just loaded (train in "Infeed" state and "Outfeed" positions that are marked as occupied (see "MB_MBELT_STATUS.OutfeedPosOccupied") is greater than "MB_MBELT_INFEED.MinDistanceToOutfeed". If an "Outfeed" position is occupied, a fixed limit is set at the ("Outfeed" position – train length) position. As long as the train is in the "Infeed" state and is at a distance "MB_MBELT_INFEED.MinDistanceToOutfeed" from this limit, the "Ready" output is set.

If the conditions are not met, the "Ready" output is reset. When the "Ready" output is reset, the upstream station should interrupt the product stream (e.g. discharging products if "Ready" = FALSE).

If the product stream persists even if "Ready"=FALSE, the monitoring feature checks if the train in the "Infeed" state would collide with with the train at the occupied "Outfeed" position(s). To do this, the "Stop" ramp is used with current train velocity and deceleration ("MB_MBELT_GENERAL.StopRamp"). If a collision is detected, the train affected is immediately stopped with the "Stop" ramp and an error is output. The instance with and the instance without unloading device are differentiated for determining train distances:

- **Without** unloading device ("MB_MBELT_INFEED.EnableUnloadHandshake" = FALSE) the distance between a train that was just loaded (train in "Infeed" state) and other trains at the "Outfeed" positions is monitored
- **With** unloading device the distance between a train that was just loaded (train in "Infeed" state) and "Outfeed" positions that are marked as occupied is monitored. If an "Outfeed" position is occupied, a fixed limit is set at the ("Outfeed" position – train length) position. If the Infeed train would travel beyond this limit with its deceleration ramp "MB_MBELT_GENERAL.StopRamp", a collision is detected and handled

Collision monitoring

Collision monitoring is enabled automatically after the function block start and runs cyclically on an ongoing basis. Collision monitoring consists of two steps:

In operating states with large distances to the train at the front (such as travelling in an empty section), the train distance has to be greater than the deceleration path to the train in front + the safety distance MB_MBELT_GENERAL.SafetyDistance. The collision monitoring signals if this condition is not met.

In operating states with very low distances to the train in front (such as catching up to the train in front), the train distance has to be greater than the safety

distance MB_MBELT_GENERAL.SafetyDistance. The collision monitoring signals if this condition is not met.

Due to system conditions, the collision monitoring is based on position command values. Train overruns caused by poorly parameterized position control can lead to collisions.

The collision monitoring can be disabled with the input **DisableCollisionDetection** = TRUE. However, this function should be used exclusively to move trains away from each other after a collision. As soon as the trains have travelled away from each other, the collision monitoring should be enabled again.

Dynamic monitoring

The "MagicBelt" application should be set such that the motion dynamics (e.g. acceleration, velocity) during travel between "In- and Outfeed" positions are set to greater or equal values relative to the "Infeed or Outfeed" motion. This is the only way to ensure that a train that executes an "Infeed or Outfeed" motion does not collide with a slower train in front. The function block generates an error and decelerates all axes with the "MB_MBELT_GENERAL.StopRamp" ramp if the dynamic data does not correspond with the rules mentioned above.

Monitoring unloading

When products are removed with a gripper, if an error occurs, products could remain in the train. Recognizing incomplete product removal must - if necessary - be done outside of the "MagicBelt" function block. If such a case is detected externally, the chains can be stopped using the "Stop" input. Following a positive "Start" edge, operation continues.

Error Handling

In case of an error, all trains are always stopped with ramps. Trains are not differentiated based on which train caused the error.

Restart

After the cause of error has been recovered, the error can be deleted with a positive edge at the **ClearError** input. With a positive edge at "Start" the trains start according to the specified start option. With the start option "START_AT_LAST_STEP", the state before the interruption can be restored and a train that is still loaded can travel farther to the unloading position, for example.

Error codes

The following error codes in the "F_RELATED_TABLE" are generated by the function block:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#0001	16#0000	No power connected to chain drive
RESOURCE_ERROR	16#0003	16#0000	The function block was aborted by another function block
STATE_MACHINE_ERROR	16#0006	16#0000	Invalid state of the internal state machine
RESOURCE_ERROR	16#0011	16#0000	Control not ready for operation
INPUT_INVALID_ERROR	16#1A00	16#0001	Infeed positioning time =< 0ms
INPUT_INVALID_ERROR	16#1A00	16#0002	Products per pocket = 0
INPUT_INVALID_ERROR	16#1A00	16#0003	Feedrate override <= 0 OR > 100
INPUT_INVALID_ERROR	16#1A00	16#0004	CtrlChain <= 0 OR > 3
INPUT_INVALID_ERROR	16#1A00	16#0005	ChainLength <= 0
INPUT_INVALID_ERROR	16#1A00	16#0006	NoOfChains <= 0 OR > 3

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#1A00	16#0007	TrainsPerChain <= 0 OR > 3
INPUT_INVALID_ERROR	16#1A00	16#0008	ProductBuffer <= 0
INPUT_INVALID_ERROR	16#1A00	16#0009	InfeedAxis AxisRef is invalid
INPUT_INVALID_ERROR	16#1A00	16#000A	Chain AxisRef is invalid
INPUT_INVALID_ERROR	16#1A00	16#000B	StopRamp too small. StopRamp has to be higher than or equal to all other given accelerations
INPUT_INVALID_ERROR	16#1A00	16#000C	InfeedType is invalid
INPUT_INVALID_ERROR	16#1A00	16#000D	FirstInfeedPosition is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#000E	BeginReducedVel1 is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#000F	EndReducedVel1 is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0010	BeginReducedVel2 is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0011	EndReducedVel2 is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0012	MaxProductRate is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0013	MinDistanceToOutfeed is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0014	MaxInfeedVelocity is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0015	MaxInfeedAcceleration is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0016	MaxInfeedJerk is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0017	StartDelayTime is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0018	InfeedLoadTime is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0019	Infeed motion law is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#001A	ProdExpectationWindowStart is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#001B	ProdExpectationWindowEnd is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#001C	AdditionalSteps is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#001D	InfeedStepData is not within the valid range

ML_TechMotion.library

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#1A00	16#001E	FirstOutfeedPosition is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#001F	MaxOutfeedVelocity is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0020	MaxOutfeedAcceleration is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0021	MaxOutfeedJerk is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0022	OutfeedDistances is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0023	MaxMoveVelocity is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0024	MaxMoveAcceleration is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0025	MaxMoveJerk is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0026	LoadTrainAcceleration is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0027	ReducedVelocity is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0028	JogAxisSelection is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#0029	JogVelocity is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#002A	JogAcceleration is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#002B	JogDistance is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#002C	TrainDistance is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#002D	SafetyDistance is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#002E	Mode input is not within the valid range
INPUT_INVALID_ERROR	16#1A00	16#002F	Expectation window end "ProdExpectationWindowEnd" is too big
RESOURCE_ERROR	16#1A01	16#0000	One or more chain drives are not referenced
RESOURCE_ERROR	16#1A02	16#0000	Outfeed position is locked
SYSTEM_ERROR	16#1A03	16#0000	Collision with train at the front detected
SYSTEM_ERROR	16#1A04	16#0000	Infeed acceleration is higher than Move acceleration

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#1A05	16#0000	Falling edge at TrainUnload but Unload Lift is not retracted or extended
RESOURCE_ERROR	16#1A06	16#0000	Modulo value of the chain drive or InfeedMaster is not set correctly
RESOURCE_ERROR	16#1A07	16#0000	Consecutive error caused by leading chain
SYSTEM_ERROR	16#1A08	16#xxxx	Unexpected state for another chain xxxx Chain number
SYSTEM_ERROR	16#1A09	16#xxxx	Invalid internal array index xxxx Array index
RESOURCE_ERROR	16#1A0A	16#xxxx	UnloadLiftLowPos AND UnloadLiftHighPos are set at the same point of time xxxx = Outfeed index
RESOURCE_ERROR	16#1A0B	16#xxxx	Resulting error caused by another chain xxxx = Chain number with error cause
RESOURCE_ERROR	16#1A0C	16#xxxx	Repositioning is required (START_AT_LAST_STEP) but unloading device is blocked xxxx = Chain number with error cause
SYSTEM_ERROR	16#1A0D	16#0000	Invalid internal step width
SYSTEM_ERROR	16#1A0E	16#0000	Internally calculated collision train is invalid
SYSTEM_ERROR	16#1A0F	16#0000	The next Infeed step was triggered, but the previous Infeed step was not yet completed and a REST_IN_REST Infeed motion law was selected
RESOURCE_ERROR	16#1A10	16#0001	Position scaling for the chain drive is invalid
RESOURCE_ERROR	16#1A10	16#0002	Velocity scaling for the chain drive is invalid
RESOURCE_ERROR	16#1A10	16#0003	Acceleration scaling for the chain drive is invalid
RESOURCE_ERROR	16#1A10	16#0004	Position scaling for the InfeedAxis is invalid
RESOURCE_ERROR	16#1A10	16#0005	Velocity scaling for the InfeedAxis is invalid
RESOURCE_ERROR	16#1A10	16#0006	Acceleration scaling for the InfeedAxis is invalid

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR	16#1A10	16#0007	Position scaling for the InfeedMaster is invalid
RESOURCE_ERROR	16#1A10	16#0008	Velocity scaling for the InfeedMaster is invalid
RESOURCE_ERROR	16#1A10	16#0009	Acceleration scaling for the InfeedMaster is invalid

Fig.6-120: Error codes of the MB_MagicBeltType01 function block

Commissioning Notes

The following points have to be taken into consideration when commissioning the "MB_MagicBeltType01" function block:

- One individual instance of the function block has to be called per chain.
- The function block "MB_MagicBeltType01" has to be called cyclically in a **SERCOS synchronous task**. The task watchdog has to be switched on (sensitivity = 1)
- The axes of the chain drive **Chain A, B, (C)** have to be created using IndraWorks. Note the following parameterization:
  - Axis type: real or virtual (for testing purposes)
  - When the drive is added, "Interpolation on drive" **has to be deselected** (command position value generation is carried out on the control)
  - Scaling type "**Translatory, Modulo, Unit [mm]**" has to be set.
  - The modulo value has to be set as follows: **Modulo value = ChainLength/TrainsPerChain**
- The **InfeedAxis** has to be created using IndraWorks. Note the following parameterization:
  - Axis type: **virtual**
  - Scaling type "**Translatory, Modulo, Unit [mm]**" has to be set.
  - The modulo value has to be set as follows: **Modulo value = ChainLength/TrainsPerChain**
- The **DummyMaster** axis has to be created in IndraWorks if no other one is available. All axis types are permitted here and no special parameterization is required because this axis is only used passively.
- With synchronous loading, the **InfeedMaster** axis also has to be created with IndraWorks. Note the following parameterization:
  - In principle, all axis types (real, virtual, encoder...) can be used.
  - Scaling type "**translatory, rotary**" has to be set.
  - Modulo value = **360°**
- The axes created in IndraWorks have to be assigned to the corresponding axes (e.g. "ChainAxes") using the "MB_MBELT_GENERAL" data structure (via AxisRef).

6.9 Function block for the SmartBelt Application

6.9.1 MB_SmartBeltType01

Brief Description

The MB_SmartBeltType01 function block measures the phase offset between the position of a product and the desired master position. If a phase offset is detected, a phase correction is applied to the position of the product, using a triangular velocity shape.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library
IndraMotion MLD 07VRS and higher with MA function package	MX_Packaging_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Packaging_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Packaging_17.lib (in preparation)

Fig.6-121: Reference table of the MB_SmartBeltType01 function block

Interface Description

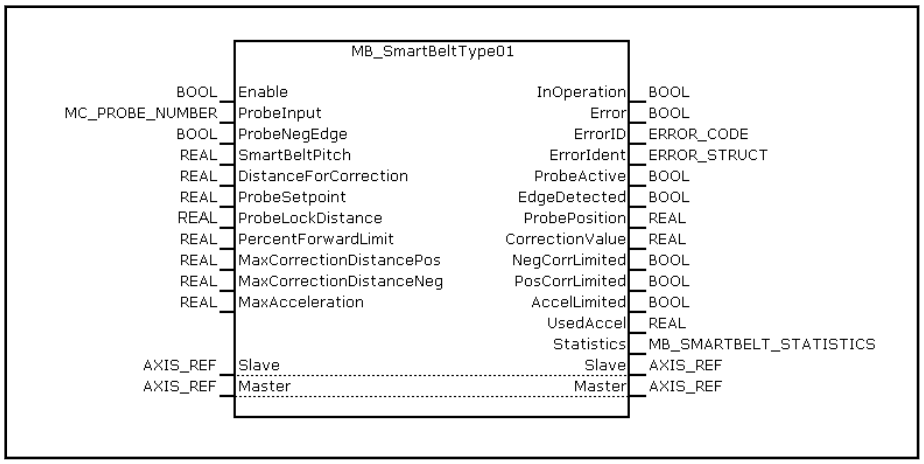


Fig.6-122: MB_SmartBeltType01 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	Slave	AXIS_REF	Address of SmartBelt axis
	Master	AXIS_REF	Address of master axis
VAR_INPUT	Enable	BOOL	Rising edge enables function block
	ProbeInput	MC_PROBE_NUM-BER	Selection of touch probe input on drive. Is read at the rising edge at the "Enable" input.
	ProbeNegEdge	BOOL	FALSE: Positive edge of touch probe is evaluated TRUE: Negative edge of touch probe is evaluated. Is read at the rising edge at the "Enable" input.
	SmartBeltPitch	REAL	Effective SmartBelt length (inch or mm). Is read at the rising edge at the "Enable" input.
	DistanceForCorrection	REAL	Distance left on the belt for correction. This is usually the distance from where the touch probe is mounted to the end of the belt. Is read at the rising edge at the "Enable" input.
	ProbeSetpoint	REAL	Touch probe command value (in or mm) Read cyclically

ML_TechMotion.library

I/O type	Name	Data type	Description
	ProbeLockDistance	REAL	Distance within the touch probe input is ignored (in or mm). If a touch probe event is detected, the next touch probe event is not considered until the belt has traveled the distance in "ProbeLockDistance". Read cyclically
	PercentForwardLimit	REAL	The percentage of the total product cycle that determines the distribution between forward and backward correction. Example: A value of 70.0 for a product cycle length of 100mm would correct up to 70mm in forward direction and up to 30mm in backward direction. Read cyclically
	MaxCorrectionDistPos	REAL	Maximum correction distance for positive correction (in or mm). Read cyclically
	MaxCorrectionDistNeg	REAL	Maximum correction distance for negative correction (in or mm). Read cyclically
	MaxAcceleration	REAL	Maximum allowed acceleration in units of drive (in/s², mm/s²). Read cyclically
VAR_OUTPUT	InOperation	BOOL	SmartBelt is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorStruct	ERROR_STRUCT	Detailed error description
	ProbeActive	BOOL	SmartBelt touch probe functionality active. Waiting for event
	EdgeDetected	BOOL	Touch probe event detected
	ProbePosition	REAL	Captured position in drive units based on "SmartBeltPitch"
	CorrectionValue	REAL	Correction value in drive units based on "SmartBeltPitch"
	NegCorrLimited	BOOL	Negative correction is limited to distance set in "MaxCorrectionDistanceNeg"
	PosCorrLimited	BOOL	Positive correction is limited to distance set in "MaxCorrectionDistancePos"
	AccelLimited	BOOL	Correction is limited to acceleration set in "MaxAcceleration"
	UsedAccel	REAL	Calculated acceleration used during phase correction (in/s² or mm/s²)
	Statistics	MB_SMART-BELT_STATISTICS	Structure containing different statistical data

Fig.6-123: Interface variables of the MB_SmartBeltType01 function block

MB_SMARTBELT_STATISTICS Structure

This structure contains statistical data that can be used for monitoring and assessing the application. The individual elements are listed in the following table:

Structure element	Type	Default value	Description
CurRate	REAL	0.0	Product rate based on distance between current and previous product
NbrOnBelt	UINT	0	Number of products currently on the belt. A product is considered "on the belt", when it has traveled less than a whole SmartBelt length since registration
MaxGap	REAL	0.0	Maximum gap between two consecutive products in units based on "SmartBeltPitch"
MinGap	REAL	"SmartBeltPitch"	Minimum gap between two consecutive products in units based on "SmartBeltPitch"
TotalNbr	UINT	0	Total number of products registered by the SmartBelt

Fig.6-124: MB_SMARTBELT_STATISTICS Structure

All elements in the MB_SMARTBELT_STATISTICS structure are set to their default values on the rising edge of the "Enable" input.

The values of the elements might not be accurate if products are not registered correctly.

The "CurRate" value is calculated based on time measurements. The resolution for the time is 1ms. For high product rates, the calculated rates may vary in the range of 1ms resolution. I.e, when a product rate of 400 products per minute is used, one product takes 1 / (400 / 60) = 150 ms. Assuming a resolution of 1ms, we could measure values in the range of 149ms, 150ms and 151ms, resulting in product rates of 403, 400 and 397 respectively.

The values for "MaxGap" and "MinGap" are calculated based on the distance traveled by the master. This is especially noticeable on the first belt when it is accelerated or decelerated. In this case, the SmartBelt cover a lot more/less distance than the master axis.

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value
Enable	BOOL			FALSE
ProbeInput	MC_PROBE_NUMBER	PROBE1	PROBE2	PROBE1
ProbeNegEdge	BOOL	-	-	FALSE
SmartBeltPitch	REAL	>0.0	10000.0	200.0
DistanceForCorrection	REAL	0.0	SmartBeltPitch	100.0
ProbeSetpoint	REAL	0.0	Modulo value of the master axis	0.0
ProbeLockDistance	REAL	0.0	SmartBeltPitch	0.0
PercentForwardLimit	REAL	0.0	100.0	50.0

ML_TechMotion.library

Name	Type	Min. value	Max. value	Default value
MaxCorrectionDistPos	REAL	0.0	SmartBeltPitch	100.0
MaxCorrectionDistNeg	REAL	-SmartBeltPitch	0.0	-100.0
MaxAcceleration	REAL	> 0.0	∞	100
Slave	AXIS_REF	AxisNo >= 1	<= MB_MAX_AXIS_NUMBER	-
Master	AXIS_REF	AxisNo >= 1	<= MB_MAX_AXIS_NUMBER	-

Fig.6-125: Overview on min./max. values and initial values of the inputs

Functional Description

Detailed description of some input and output variables

The measurement units have to be equal for all entries (inch or mm). If the "SmartBeltPitch" is for example specified in mm, the "DistanceForCorrection" also has to be in mm and the "MaxAcceleration" in mm/s².

"PercentForwardLimit" is used to determine the distribution of the forward and backward correction. If the product should only be corrected in forward direction for example, a value of 100.0 has to be entered. That compensates all possible position errors by a forward correction. This value indicates a percentage distribution to the front or to the back.

The maximum measured error for a product can range from (–0.5 * product cycle) to (+0.5 * product cycle). The term product cycle is identical to the term "CamShaftDistance" or corresponds to the product length plus the gap between the products. Refer to the following figure as an example:

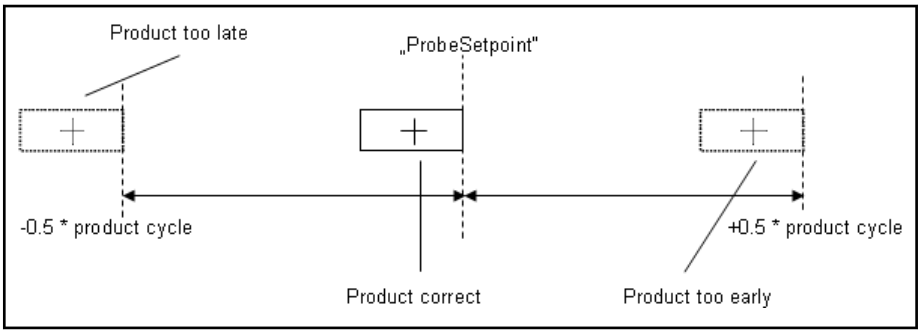


Fig.6-126: Meaning of the correction in positive or negative direction

The product can now be corrected in either the positive or negative direction. Thus, allowing the correction of the complete product cycle. "PercentForwardLimit" specifies how the measured error is corrected in relation to the product cycle length. If "PercentForwardLimit" is set to 50%, half of the error is corrected forward and half backward.

If "PercentForwardLimit" is set to 70%, corrections are made as follows:

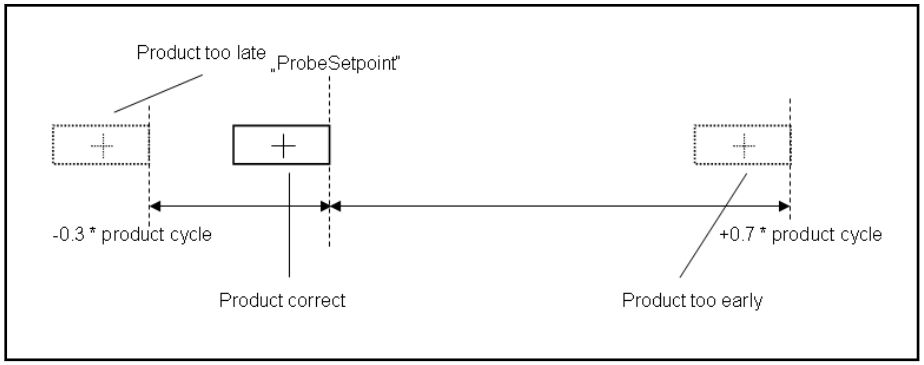


Fig.6-127: PercentForwardLimit

"MaxCorrectionDistancePos" and "MaxCorrectionDistanceNeg" limit the amount of absolute distance that can be compensated in either direction. Together with "PercentForwardLimit", combinations can be created to compensate for all errors or just fractional errors in either direction. For example, to correct 80% of all errors in the positive direction (correcting no more than 60mm in the positive direction and no more than 20mm in the negative direction of the length of a complete product cycle) configure the inputs as follows:

"PercentForwardLimit" = 80

"MaxCorrectionDistancePos" = 60

"MaxCorrectionDistanceNeg" = -20

Explanation: For a 200mm long product cycle, with "PercentForwardLimit" = 80, 80% (160 mm) is corrected forward and 20% (40mm) backward. "MaxCorrectionDistancePos" = 60 limits the available 160mm to only 60mm while "MaxCorrectionDistanceNeg" = -20 limits the allowed backward correction to -20 mm.

The overall scenario is described in the following:

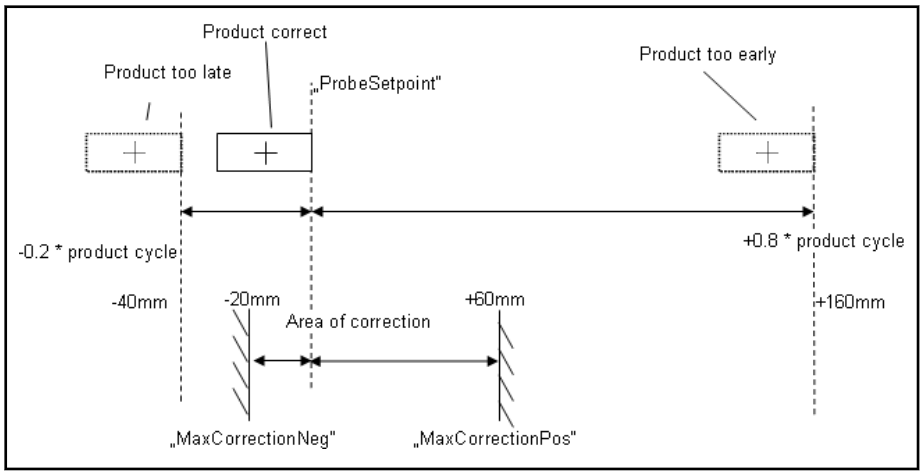


Fig.6-128: MaxCorrectionDistancePos and MaxCorrectionDistanceNeg

An error outside the limits of "MaxCorrectionDistancePos" and "MaxCorrectionDistanceNeg" is limited to "MaxCorrectionDistancePos" and "MaxCorrectionDistanceNeg". The remainder is corrected on the next belt. Thus, it is possible to distribute the corrections on all belts so that the first belt does not have the main correction load.

The higher the correction distance, the higher the used acceleration. Alternatively, the maximum acceleration can also be limited using "MaxAcceleration". Both maximum acceleration and maximum correction distance are active at the same time with the lower of the two restrictions being applied.

ML_TechMotion.library

Time requirements

The function block requires up to 400 PLC cycles until initialization is finished. The exact amount depends on the load on the control.

Error Handling

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0006)	16#0012	16#0000	Axis without power
RESOURCE_ERROR (16#0006)	16#0001	x	Axis (master or slave axis) is not located at the local control. "x" indicates the address of the incorrect axis
ACCESS_ERROR (16#0004)	16#0003	16#0000	The internally used function block MB_TouchProbeContinuous was interrupted by another function block
ACCESS_ERROR (16#0004)	16#0003	16#0001	The internally used function block MB_PhasingSlave was interrupted by another function block
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Error in state machine
INPUT_INVALID_ERROR(16#0001)	16#000D	x	Axis (master or slave axis) is not provided with a valid address. x indicates the address of the incorrect axis
INPUT_INVALID_ERROR(16#0001)	16#0F02	16#0010	"SmartBeltPitch" outside valid range (0..10000)
INPUT_INVALID_ERROR	16#0F02	16#0011	"ProbeInput" invalid (PROBE1, PROBE2)
INPUT_INVALID_ERROR	16#0F02	16#0012	"DistanceForCorrection" invalid (0.." SmartBeltPitch")
INPUT_INVALID_ERROR	16#0F02	16#0013	"MaxCorrectionDistPos" invalid (0.." SmartBeltPitch")
INPUT_INVALID_ERROR	16#0F02	16#0014	"MaxCorrectionDistNeg" invalid (-"SmartBeltPitch"..0)
INPUT_INVALID_ERROR	16#0F02	16#0015	Axis is not an IndraDrive axis (Pack profile not allowed)
INPUT_INVALID_ERROR	16#0F02	16#0017	Scaling of slave is not the preferred scaling
INPUT_INVALID_ERROR	16#0F02	16#0018	Scaling of master is not the preferred scaling
INPUT_INVALID_ERROR	16#0F02	16#0019	Acceleration value too small (>0)
INPUT_INVALID_ERROR	16#0F02	16#001A	Slave does not have modulo scaling
INPUT_INVALID_ERROR	16#0F02	16#001B	Master does not have modulo scaling

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR	16#0F02	16#001C	Settings in P-0-0154 (IndraMotion MLD)/A-0-2793 (IndraMotion MLC) has to be set to "Shortest Path"
INPUT_INVALID_ERROR	16#0F02	16#001D	Input "ProbeSetpoint" is out of limits (0..Master Modulo)
INPUT_INVALID_ERROR	16#0F02	16#001F	"ProbeLockDistance" input out of limits (0.."SmartBeltPitch")
INPUT_INVALID_ERROR	16#0F02	16#0020	"PercentForwardLimit" is out of limits (0..100.0)
INPUT_INVALID_ERROR	16#0F02	16#0021	Modulo value of slave has to be at least the sum of MaxCorrectionDistPos + ABS(MaxCorrectionDistNeg)
INPUT_INVALID_ERROR	16#0F02	16#0022	Modulo value of the slave axis has to be at least twice the size of the "SmartBeltPitch"
DEVICE_ERROR (16#0008)	16#0F06	16#0000	Scaling of velocity and scaling of position data is not the same

Fig.6-129: Error codes of the MB_SmartBeltType01 function block

In addition, the function block can issue all the following errors of sub function blocks:

- MB_GetCyclicParameterHandle
- MB_PhasingSlave
- MB_ReadParameter
- MB_ReadRealParameter
- MB_ReadStringParameter
- MC_TouchProbeInit
- MC_TouchProbeContinuous

Refer to the relevant documentation for error codes of the respective function block.

6.9.2 Introduction to SmartBelt Usage

SmartBelt Usage

SmartBelts are used to correctly space out products before they are processed. Products are provided by a feeding unit to the SmartBelts where the phase is adjusted. Then a processing unit further handles the products (e.g., a flow wrapper).

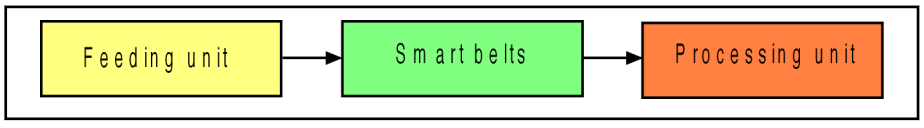


Fig.6-130: Example a flow wrapper

Depending on the application, different situations can arise:

- Feeding units provide products with the correct distance to each other but the processing unit cannot synchronize to the feeding unit when starting or stopping. In this situation, SmartBelts can be used to correct the gap between products

- Feeding units provide products from a storage on demand. No SmartBelts are necessary
- Feeding units provide products with a random, but small variation in the gap between the products. SmartBelts can be used
- Feeding units provide products with a random variation in the gap between the products. SmartBelts can be used together with following additional measures:
  - Velocity of the feeding unit can be controlled to adjust a constant product rate
  - Velocity of the processing unit can be controlled to adjust to the current product rate

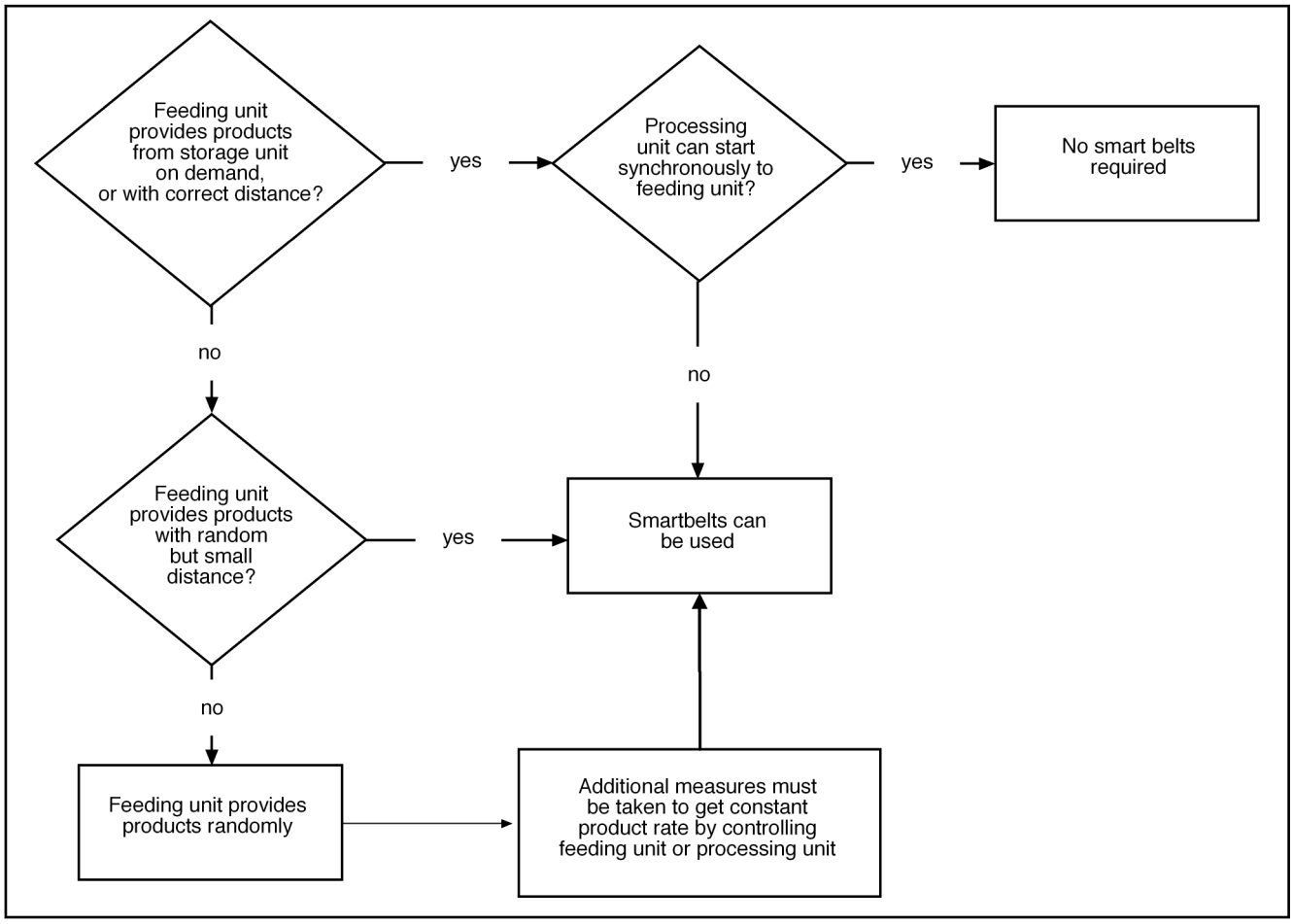


Fig.6-131: Usage of SmartBelts

Accumulating Conveyor

Using an accumulating conveyor increases the order of the products. In the following figure, belt 2 is an accumulating conveyor. Its length (L2) has to be longer than the length of the products on the previous belt (L1). Otherwise, the products from belt 1 push the products on belt 2.

The velocity on the accumulating conveyor has to be less than the velocity on belt 1 in order to accumulate products.

Belt 3 has a higher velocity compared to the accumulating conveyor in order to pull the products apart by a predefined distance.

Gaps between the products on the accumulating conveyor result in larger gaps on belt 3. However, they are not necessarily a multiple of the desired

distance. The velocity of the accumulating conveyor can be controlled in order to avoid this situation.

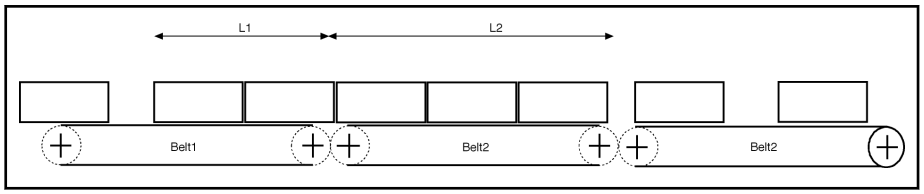


Fig.6-132: Accumulating Conveyor

Degree of Order

Assessment using Product Cycles and Gaps

The degree of order is critical to the number of SmartBelts necessary. The higher the order, the lower the number of SmartBelts. The degree of order directly affects the amount of correction for the SmartBelts. It is calculated with the following formula:

The following figure should explain the various parameters:

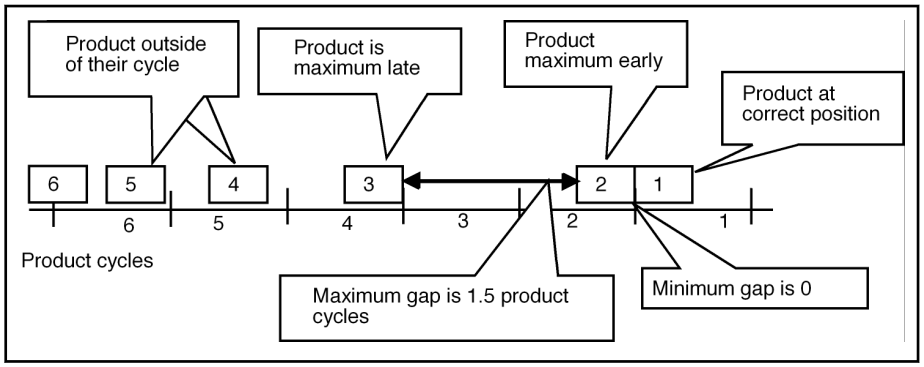


Fig.6-133: Assessment using Product Cycles and Gaps

The general assumption is that the leading edge of the product is point of reference. It does not matter if the leading, trailing edge or the center of the product is considered when making the following conclusions, as long as the same point of reference is used.

The the figure above, product 1 is exactly where it should be and would not need any further correction. Product 2 is too close to product 1, but is still within its dedicated product cycle. It has a minimum gap of 0 to its predecessor. Product 3 is too late, but also within its product cycle. Products 4 and 5 are outside their respective product cycles while product 6 is within its product cycle.

The amount of correction is calculated using the following variables:

- Product length (L_{Prod})
- Nominal product gap at the given velocity (L_{gap})
- Nominal product cycle at the given velocity (L_{abs})
- Minimum gap between products (L_{gapmin})
- Maximum gap between products (L_{gapmax})
- Consecutive products outside their product cycle (N_{gap})

The following example values apply to the above figure:

L_{Prod}: 40 mm (assumed)

L_{gap}: 50 mm (assumed)

L_{cycl}: 90mm (= L_{Prod} + L_{gap})

$L_{gap\ min}$: 0mm (gap between products 1 and 2)

$L_{gap\ max}$: 1.5 product cycle = 1.5 * L_{cycl} = 135mm (gap between products 2 and 3)

N_{gap}: 2 (gap between products 4 and 5)

The necessary amount of correction for the SmartBelts is calculated as follows:

$$Correction = \min\left(L_{gapmax}, \left(1.5 * L_{cycl} - L_{prod} + L_{gap} * \left(N_{gap} - 1\right) - \left(N_{gap} * L_{gap,min}\right)\right)\right)$$

For this example, the following values apply:

Correction = min (135 mm, (1.5 * 90 mm – 40 mm + 50 mm * (2 – 1) – (2 * 0 mm)))
= min (135 mm, 145 mm)
= 135 mm

This is example, 135mm would be the amount of distance the SmartBelts would have to travel in order to correct product position in both directions.

Assessment using Product Rate and Duration

A different method to estimate whether the correction capability is sufficient is to take a look at the product rate. The rate should be constant, but varies with time around an average value. The duration of that deviation and the size of the deviation can also be used to determine if a SmartBelt can support the application.

The formula to calculate the time of deviation is as follows:

$$t = \frac{1}{\frac{R}{60} * \frac{Deviation[\%]}{100}}$$

Where R is the intended product rate in [1/min] and "Deviation" is the percentage by which the product rate is not the intended one.

Example:

If the intended product rate is 200 products per minute, and deviations occur which are 10% in size. This would mean that there can be 180 to 220 products per minute. The above formula results in a time of :

$$t = \frac{1}{\frac{200}{60} * \frac{10\%}{100}} = 3s$$

In case the product rate deviates from the nominal rate for less than 3 seconds, the SmartBelt can support the application. Otherwise, either the products are not corrected on the belt or empty places occur.

Friction

The connection between friction and possible acceleration is explained as follows:

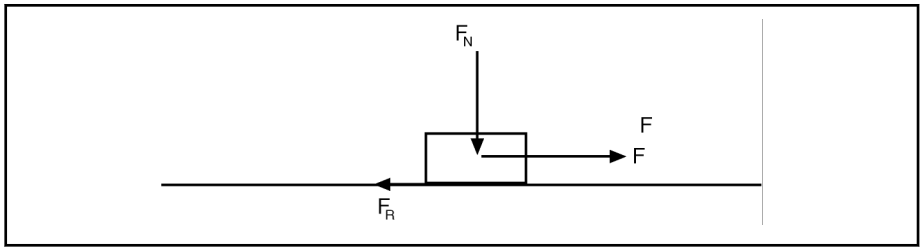


Fig.6-134: Friction

Static friction is defined as:

$F_R = \mu_H F_N$

with

$F_N = F_G = m * g$

and substituting the later by the first one

$F_R = \mu_H * m * g$

The necessary force for acceleration is:

$F = m * a$

Dynamic friction starts, when $F = F_R$

or substituted

$\mu_H * m * g = m * a_{max}$

solved for a_{max}:

$a_{max} = \mu_H * g$

a_{max} is the maximum acceleration, μ_H is the friction coefficient and g is the acceleration of gravity.

μ_H can also be visualized in a different way as the following figure shows. μ_H is the equivalent of tan φ. In other words, an object with a friction of 1 would start to slide on a plane at a 45° angle.

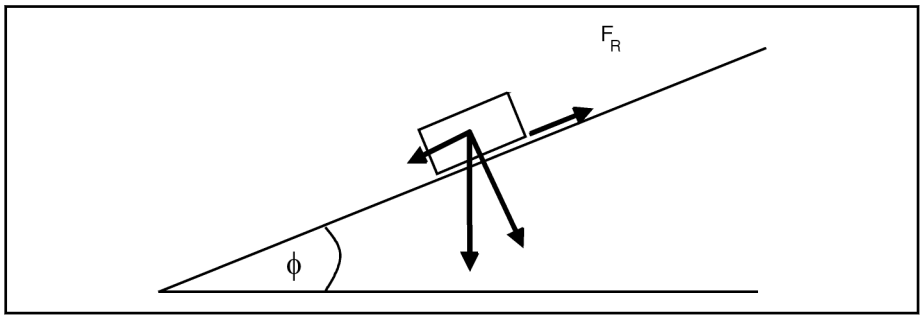


Fig.6-135: Relation between friction coefficient and angle

6.9.3 Project Planning

General

This section describes the settings and calculations that has to be performed when using the MB_SmartBeltType01 function block.

Mechanical Requirements for Belt and Machine

The belt should have a length that is approximately the size of the product plus the intended gap (product cycle). The belt length is the physical length of the whole belt, not the circumference. There is no benefit if the belt is significantly longer or shorter than the product cycle. In case of a longer belt, multiple products remain on a belt. Thus, causing the last product to wait for the

first one to leave the belt. In case of shorter belts, there is less time and higher accelerations to create gaps between the products.

The touch probe for each belt should be connected to the corresponding drive input. Note which touch probe input is being used. Mount the touch probe as close as possible to the start of the belt in case the positive edge is detected. However, the product has to have its center of mass already on the belt. For symmetric products this is half of the product length. The touch probe should be mounted at a location that is half the product length plus half the axis diameter of the belt (as shown in fig. 6-136 "Touch probe with detection of positive edge" on page 258).

If the negative edge of the product is used, similar logic applies. In this case, the touch probe should be located at a position half the product length minus half the axis diameter of the belt (before the belt) as shown in fig. 6-137 "Touch probe with detection of negative edge" on page 258.

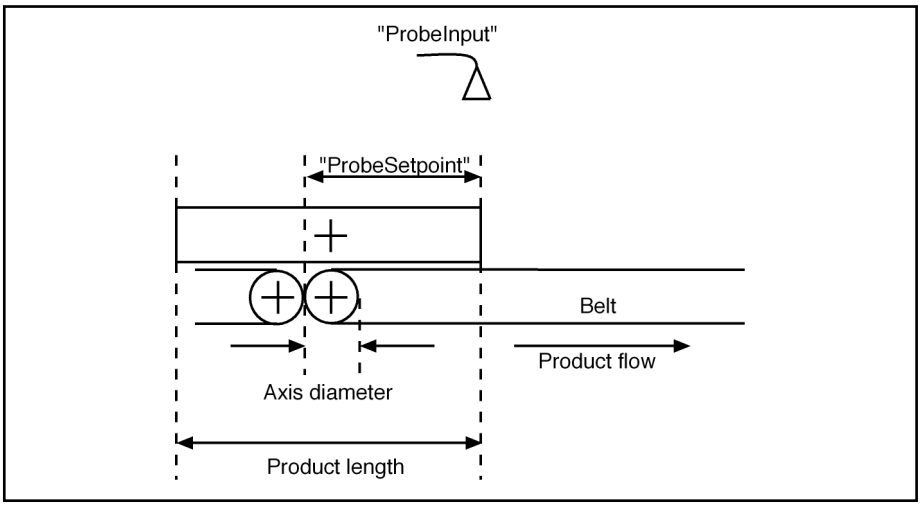


Fig.6-136: Touch probe with detection of positive edge

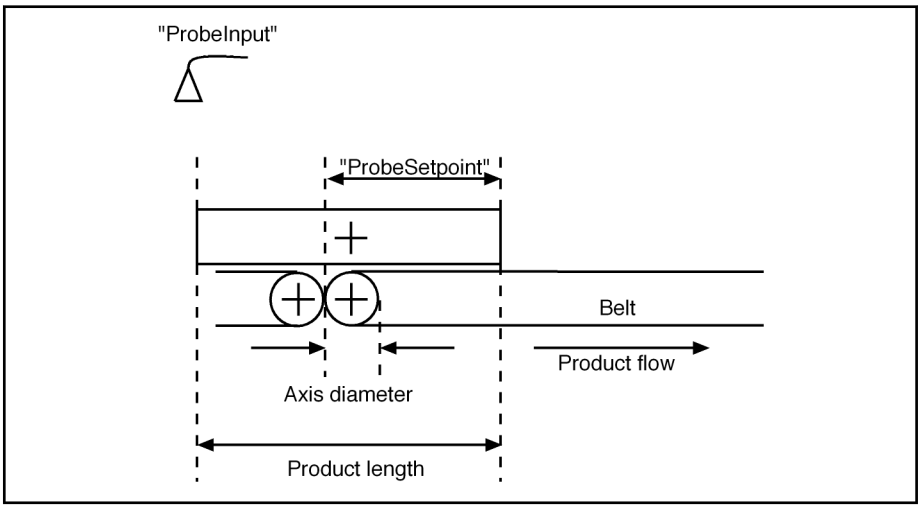


Fig.6-137: Touch probe with detection of negative edge

For multiple products with different formats running on the same SmartBelt, the touch probe should be mounted for registration with the negative edge. In this case, the smallest product determines the location of the touch probe. In this case, the compensation capability of larger products is reduced.

Drive Requirements

The drive must have a function package that supports synchronous operation modes and touch probe inputs.

For the IndraMotion MLC the following requirements apply:

- Only IndraDrives with interpolation in the drive can be used. The drive firmware must be at least MPx04V22 or higher. MPx03 is not supported and does not work.

For IndraMotion MLD the following requirements apply:

- The drive firmware must be at least MPx05VRS or higher. MPx03VRS or MPx04VRS is not supported and does not work.
- The MA package has to be activated.

Drive Parameter Settings

It is recommended to load base parameters before configuring the drive. In this section, the differences to the default parameter are listed.

Before using the function block, some additional settings must be made. Since the function block uses the touch probe functionality, the touch probe configuration must be set properly. Depending on the touch probe input that is used and the edge for triggering, the following settings are necessary for the optional cyclic channel. In order to have different options at runtime, multiple settings can be configured. For example, place S-0-0130 and S-0-0131 as well as S-0-0409 and S-0-0410 in the cyclic read channel. This allows the usage of either positive or negative edge at runtime.

The following parameters are set in the control to allow cyclic parameters to be transmitted to the drive. Additional parameters, set in the drive, are required for the touch probe functionality.

IndraWorks' drive parameter editor can be used to make parameter settings.

IndraMotion MLC parameter settings

Desired configuration	Cyclic read access	Signal control word
Touch probe 1, positive edge	S-0-0130, S-0-0409	S-0-0405, Bit 0
Touch probe 1, negative edge	S-0-0131, S-0-0410	S-0-0405, Bit 0
Touch probe 2, positive edge	S-0-0132, S-0-0411	S-0-0406, Bit 0
Touch probe 2, negative edge	S-0-0133, S-0-0412	S-0-0406, Bit 0

IndraMotion MLD parameter settings

Desired configuration	Actual values	Command values
Touch probe 1, positive edge	S-0-0130 S-0-0409	S-0-0405, Bit 0
Touch probe 1, negative edge	S-0-0131 S-0-0410	S-0-0405, Bit 0

ML_TechMotion.library

Desired configuration	Actual values	Command values
Touch probe 2, positive edge	S-0-0132 S-0-0411	S-0-0406, Bit 0
Touch probe 2, negative edge	S-0-0133 S-0-0412	S-0-0406, Bit 0

Touch probe functionality in the drive

The following parameters are set in the drive to enable the drive touch probe functionality:

Parameters	Value	Description
P-0-0300	S-0-0401 (touch probe 1) or S-0-0402 (touch probe 2)	Digital input assignment of touch probe functionality
P-0-0301	0 (Bit 0)	Bit assignment to touch probe functionality
S-0-0426	P-0-0775	Master position for touch probe measurement
S-0-0169	Bit 0 through 3	Activation of positive or negative edge detection for touch probe 1 and 2
P-0-0226	Bit 9 = TRUE	Touch probe function is automatically enabled upon entering phase 4 in the drive

In addition to application-specific settings, the following parameters must be set.

Hardware Independent Settings

Parameters	Value	Description
P-0-0072 P-0-0092 P-0-0780 P-0-0781 P-0-0782 P-0-0783 P-0-0784 P-0-0785	Linear Cam with 1:1 relationship between master and slave. Only one parameter (Cam) must be set. CamBuilder Software can be used to create and download a cam	The corresponding cam number (CamTableID) must be referenced in the MC_CamIn function block, when establishing synchronous operating mode
S-0-0121	Input revolutions of load gear	Mechanical gear, input revolutions
S-0-0122	Output revolutions of load gear	Mechanical gear, output revolutions
S-0-0123	Linear motion per revolution	Feed constant for the drive

IndraMotion MLC axis parameter settings

Parameters	Value	Description
A-0-0045	Twice the belt length	Modulo value of the axis
A-0-0059	Bit 0 = TRUE, bit 1 and 2 = FALSE, Bit 3 = FALSE, bit 7 = TRUE	Select linear, modulo and preferred scaling
A-0-0059	Bit 4 = FALSE for units in mm, = TRUE for units in inches	Selecting units of belt
A-0-2790	High enough for the application	Synchronization velocity

Parameters	Value	Description
A-0-2791	High enough for the application	Synchronization acceleration
A-0-0032	High enough for the application	Positive velocity limit
A-0-0033	High enough for the application	Negative velocity limit
A-0-0034	High enough for the application	Acceleration limit

The master axis used for the belt must also have a modulo and preferred scaling (A-0-0059: Bit 3 = FALSE, bit 7 = TRUE) as well as rotary or linear weighing. One master revolution converts to one product cycle (product length plus gap between products) for a rotary master. In case of a linear master, the modulo value must correspond to the product cycle. If one master cycle corresponds to multiple/fractions of the product cycle, the gear ratio must be set accordingly. For example, if one master revolution corresponds to three products, then the inputs for the gear should as follows:

- "RatioNumerator"/"InputRevolution" should be set to 1
- "RatioDenominator"/"OutputRevolution" should be set to 3

IndraMotion MLD parameter settings

Parameters	Value	Description
S-0-0103	Twice the belt length	Modulo value of the axis
S-0-0076	Bit 0 = TRUE, bit 1 and 2 = FALSE, Bit 3 = FALSE, bit 7 = TRUE	Select linear, modulo and preferred scaling
S-0-0076	Bit 4 = FALSE for units in mm, = TRUE for units in inches	Selecting units of belt
P-0-0143	High enough for the application	Synchronization velocity
P-0-0142	High enough for the application	Synchronization acceleration
S-0-0038	High enough for the application	Positive velocity limit
S-0-0039	High enough for the application	Negative velocity limit
S-0-0138	High enough for the application	Acceleration limit

The master for the belt must also have a modulo and preferred scaling (P-0-0756 in case of an axis generator) as well as rotary or linear weighing. One master revolution converts to one product cycle (product length plus gap between products) for a rotary master. In case of a linear master, the modulo value must correspond to the product cycle. If one master cycle corresponds to multiple/fractions of the product cycle, the gear ratio must be set accordingly in the PLC program. For example, if one master revolution corresponds to three products, then the inputs for the gear should be "RatioNumerator"/"InputRevolution" should be set to 1 and "RatioDenominator"/"OutputRevolution" should be set to 3.

PLC Programming

The axis to be operated with the SmartBelt has to be in a synchronous operation mode. The following code fragments show how to set it.

Structured text:

Program:

```
myMC_CamIn(Execute:= TRUE,
           RatioNumerator:= 1,
```

```
        RatioDenominator:= 1,
        MasterFineadjust:= 0,
        CamShaftDistance:= myProductCycle,
        SyncMode:= SHORTEST_WAY,
        StartMode:= RELATIVE,
        CamTableID:= myCamTableID,
        Master:= mySmartBeltMaster,
        Slave:= myBelt,
        InSync=> ,
        Active=> ,
        CommandAborted=> ,
        Error=> ,
        ErrorID=> ,
        ErrorIdent=> ,
        CamActiveID=>
);
```

Function block diagram:

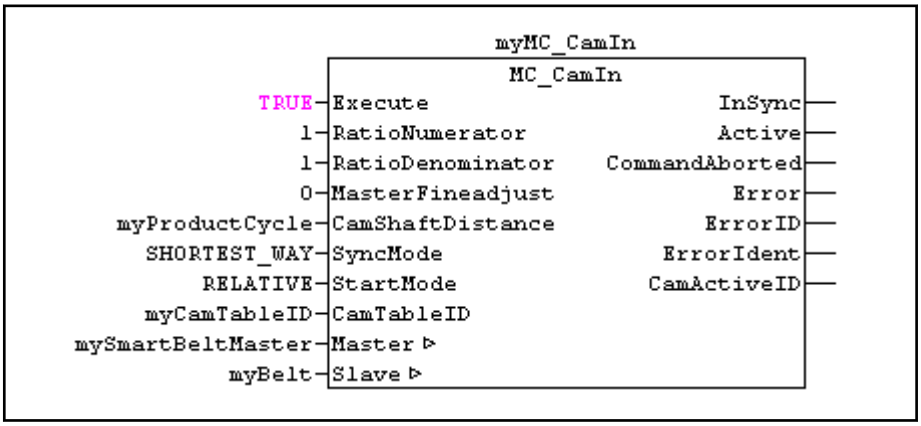


Fig.6-138: MC_CamIn settings for usage with SmartBelt

When programming the **AxisInterface**, the operating mode for the drive must also be set to "CamIn". Refer to the following window for an example:

Structured text:

Program:

```
arAxisCtrl[myBelt.AxisNo].SyncMode.Master := mySmartBeltMaster;
arAxisCtrl[myBelt.AxisNo].SyncMode.SyncDirection:= SHORTEST_WAY;
arAxisCtrl[myBelt.AxisNo].SyncMode.StartMode:= RELATIVE;
arAxisCtrl[myBelt.AxisNo].SyncMode.InputRevolution:= 1;
arAxisCtrl[myBelt.AxisNo].SyncMode.OutputRevolution:= 1;
arAxisCtrl[myBelt.AxisNo].SyncMode.FineAdjust:= 0;
arAxisCtrl[myBelt.AxisNo].SyncMode.CamShaftDistance := myProductCycle;
arAxisCtrl[myBelt.AxisNo].SyncMode.CamTableID := CAM_TABLE_1;
arAxisCtrl[myBelt.AxisNo].Admin._OpMode:= ModeSyncCam;
```

The synchronization distance has to be set to SHORTEST_WAY to guarantee that the SmartBelt moves forward and backward.

The starting mode must be set to RELATIVE. This allows the user to synchronize the SmartBelt to the master in standstill mode without the belt moving.

The "CamShaftDistance" input must be set to the product cycle of the application. This is the length of the product plus the gap between the products. Given this length, the SmartBelt determines whether to correct forward or backward. Even when no 1:1 relationship between master and SmartBelt ex-

ists, this value must be set for the product cycle. In this case, use the gear to choose the relationship between master axis and SmartBelt.

The cam ID used in the drive can be chosen freely but has to set accordingly in the "CamTableID" element. The cam must be downloaded to the drive before using it. The cam has to be in the ratio of 1:1. It can be created with the CamBuilder software. To reduce the download time, the number of cam points can be reduced to the minimum possible points.

The gear ratio used is 1:1 which follows a 1:1 ratio between the master cycle and the product cycle. If this ratio is different, the gear ratio must be set accordingly.

Specification of the "SmartBeltPitch"

The "SmartBeltPitch" indicates the physically effective length of the SmartBelt. The SmartBelt length should approximately be equal to the product length + gap between the products (= product cycle). The touch probe is mounted at the beginning of the SmartBelt and detects either the beginning or the end of the product. The SmartBelt function block assumes that the product has to move further at least by the "SmartBeltPitch" after detected by the touch probe and before leaving the belt. Only then, the following product can be corrected. This can be seen in the following. The product cycle is a bit smaller than the length of the SmartBelt. In the exemplary case, product 1 moves up to the dashed line before the following product (2) can be corrected.

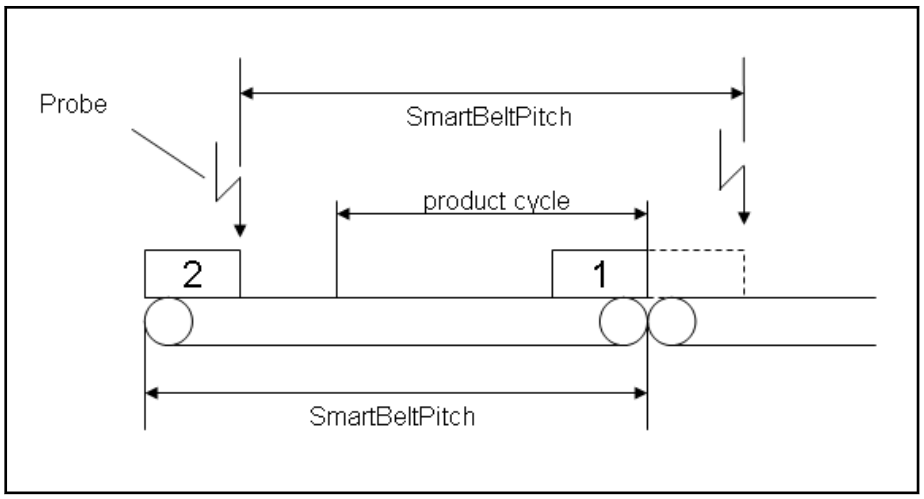


Fig.6-139: SmartBelt with physical length approximately corresponds to the product cycle

These assumptions are correct if the touch probe is mounted at the beginning of the SmartBelt in the area that does not exceed the product length (see following figure). In this case, the SmartBelt is slightly longer than two product cycles. Starting from the second belt, a product cycle A is also drawn, but backwards seen from the touch probe of the second belt. This shows that the following product is always still on the first belt when the first product had traveled the "SmartBeltPitch" length. Then, the second product can be corrected.

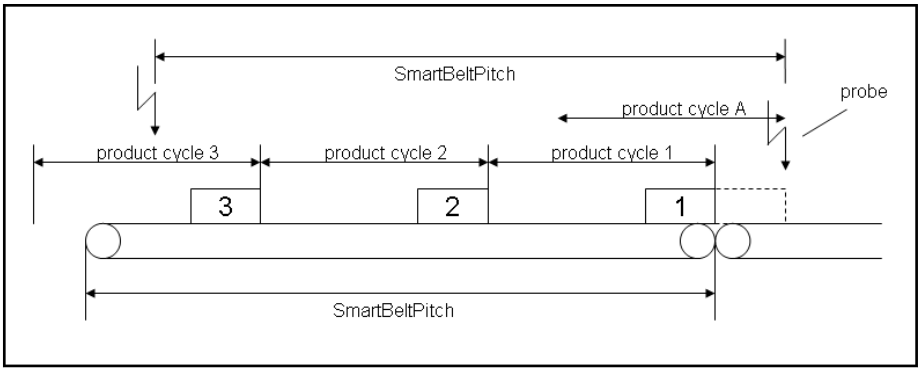


Fig.6-140: SmartBelt with physical length corresponding to a multiple of the product cycle

If the touch probe is not mounted at the beginning of the SmartBelt, the SmartBelt does not operate correctly since it wants to keep on moving a complete "SmartBeltPitch" length after the product detection. In this case, the product left the belt long before the following product is corrected. In the case drawn, the following product would also have left the belt since the product cycle A is completely on the following belt. The following product cannot be corrected anymore.

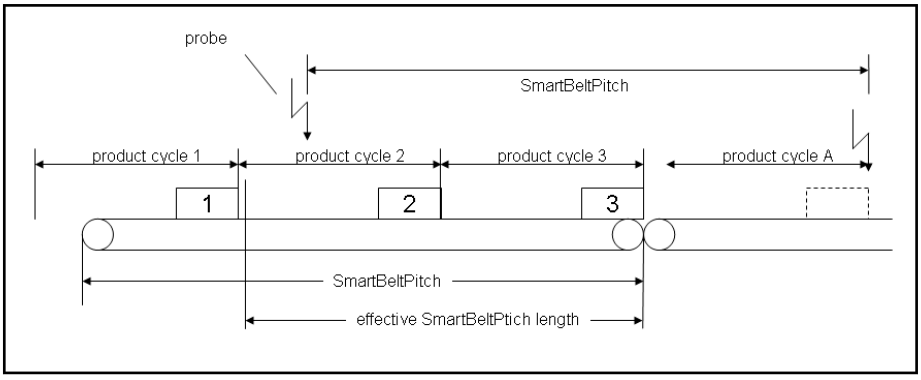


Fig.6-141: SmartBelt with touch probe not mounted at the beginning of the belt

To ensure this, the physical length may not be specified but the effective length of the SmartBelt has to be specified. This corresponds to the distance from the end of the belt to the touch probe + one product length. This distance is also drawn in the figure. The difference between the effective length and the physical length is the area at the beginning of the SmartBelt in which the product is on the belt but not yet detected by the touch probe. Thus, it is not corrected in this area and is thus irrelevant for the function block.

☞ To calculate the "ProbeSetpoint" in the following, the physical length of the SmartBelt and not the effective length has to be used!

Calculating the "ProbeSetpoint"

The "ProbeSetpoint" input identifies the desired master position at the moment the product passes the touch probe of a SmartBelt. This point can be calculated by using the distance between the touch probe location and the end of all SmartBelts, i.e. where the product must be aligned. This is illustrated in the following figure:

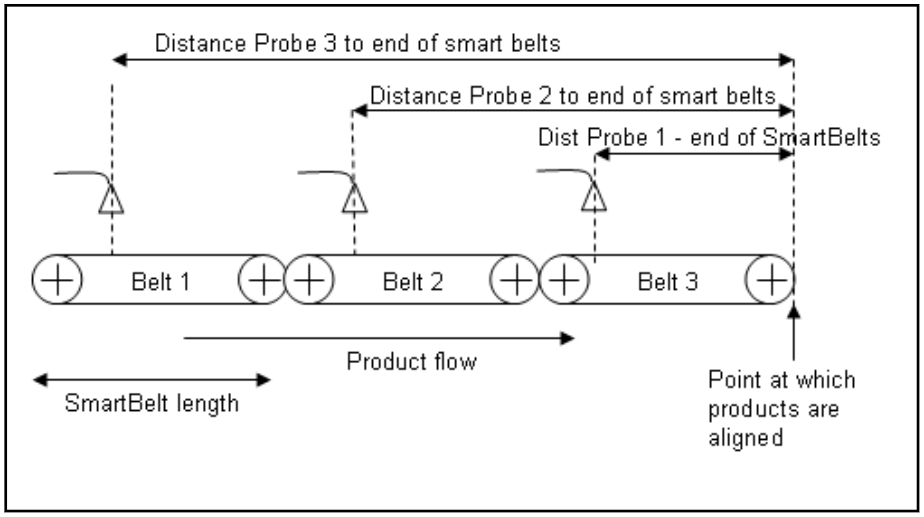


Fig.6-142: Measurements to calculate "ProbeSetpoint"

First, measure all the distances from the touch probes to the end of all SmartBelts, i.e where the products are aligned. Then the following formula can be used to calculate the value for the **positive edge detection** on the touch probe input:

$$\text{ProbeSetpoint}_k = L_{cycl} - \left(\left(\left(L_{SmartBelt_n} - L_{ProbePos_n} \right) + \sum_{n=1}^{n=k-1} L_{SmartBelt_n} \right) MOD \left(L_{cycl} \right) \right)$$

Fig.6-143: Formula for the positive edge detection

The following formula applies to the **negative edge detection** of the touch probe input:

$$\text{ProbeSetpoint}_k = L_{cycl} - \left(\left(\left(L_{SmartBelt_n} - L_{ProbePos_n} - L_{Prod} \right) + \sum_{n=1}^{n=k-1} L_{SmartBelt_n} \right) MOD \left(L_{cycl} \right) \right)$$

Fig.6-144: Formula for the negative edge detection

The following variables are used in the calculations:

- L_{cycl}: length of the product cycle (length of product (L_{prod}) plus length of gap)
- k: number of SmartBelts used
- $L_{ProbePos}$: touch probe position at a specified SmartBelt (measured from the beginning of the respective SmartBelt)
- $L_{SmartBelt}$: length of the SmartBelt

Assuming that three SmartBelts are used, each 150mm long, where the touch probe is mounted 30mm from the beginning of the SmartBelt, the product cycle is 120mm and the positive edge of the product is detected. The following "ProbeSetpoint" values can be calculated as follows:

- SmartBelt 1: k=1; ProbeSetpoint_1 = 120 - (((150-120) + 0) MOD (120)) = 120 - (30 MOD 120) = 90
- SmartBelt 2: k=2; ProbeSetpoint_2 = 120 - (((150-120) + 150) MOD (120)) = 120 - (180 MOD 120) = 60
- SmartBelt 3: k=3; ProbeSetpoint_3 = 120 - (((150-120) + 300) MOD (120)) = 120 - (330 MOD 120) = 30

6.9.4 SmartBelt Limitations and Behavior

Limitations

The function block is intended for phase corrections of products on one belt. The products must come onto the belt with a gap in between to identify each product properly. If the gap is too narrow, the second product is not detected that leads to a false correction.

The limit values for acceleration and/or maximum and minimum distances must be set to values that can be achieved. Otherwise, the products start to slide on the belt. This results in incorrect corrections.

Ideally, only one product should be on the belt at any time. If two or more products are on one belt, the second one is not corrected until the first one leaves the belt. The remaining distance for the correction is thus lower. In addition, the second product participates in the correction of the first product. Both values are taken into account when calculating the correction amount for the second product. However, it is not possible that the second product is corrected in the same way as it would if alone on a belt.

Two or More Products on One Belt

When two products are on a belt, the second product is not corrected, until the first one is off the belt. In this case, "off the belt" means the first product has traveled one "SmartBeltPitch" from the registration point. In the following figure, the product described with a dotted line "left the belt", as it has traveled a complete "SmartBeltPitch" since the registration on belt 1 (looking at: the positive edge of the product). This value is specified.

The second product has less time/distance to make its corrections. The remaining distance consists of the "DistanceForCorrection" minus the already traveled distance. Having this value, all variables (velocity, acceleration and time) can be calculated for the second product.

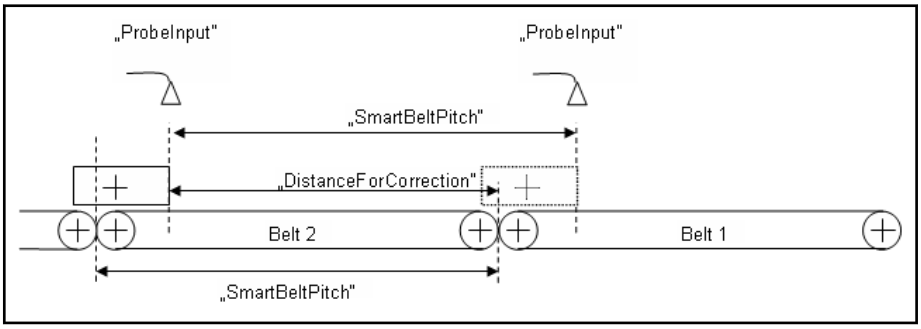


Fig.6-145: Two or More Products on One Belt

An additional problem is that the second product must always be corrected backwards. The correction is calculated based on the settings in "PercentForwardLimit", "MaxCorrectionDistancePos" and "MaxCorrectionDistanceNeg". Now, if the second product is on the belt as the first product leaves the belt, the algorithm calculates a new correction value for the second product. This could push the second product forward. However, this is not possible since the first product is there. In this case, a complete product cycle is subtracted in order to move the product to the next empty spot. Any backward motion is limited to "MaxCorrectionDistanceNeg".

Multiple Products per Master Cycle

If multiple or fractional products correspond to one master cycle, additional settings must be made. The gear ratio must be used to define this relationship.

For example, if one master revolution corresponds to three products, the inputs "RatioNumerator"/"InputRevolution" have to be set to 1 and "RatioDenominator"/"OutputRevolution" have to be set to 3. Refer to chapter "PLC Programming" on page 261 for details.

It is important to note that the "CamShaftDistance" input cannot be used to change the ratio between the master axis and the SmartBelt. If "CamShaftDistance" is not set to the product cycle, false corrections result!

6.9.5 Tips and Tricks

Correction Distance

The previous sections in this chapter provided details on the configuration of the SmartBelt functionality. This section provides additional information, so dependencies can be seen and improvements to an already running system can be made.

The SmartBelt uses the data of the input "DistanceForCorrection" to calculate the distance for correcting the gap between products. This is normally the distance between the touch probe position and the belt end. This is shown for product A in the figure below. This is safer, since the product can only be corrected if it is completely on the belt.

Product B has a longer distance for the correction. The center of mass is at the same position as the center of the SmartBelt axis. In this case, the chances of slippage are still low.

At product C is the center of mass between the two adjacent SmartBelts. In this case, slippage already occurs.

The value of "DistanceToCorrect" can be increased from case A to C in order to increase the amount of correction while monitoring slippage. It depends on the application, mainly friction and diameter of the SmartBelt axis, how much the value of "DistanceToCorrect" can be increased. Using this information, the amount of correction can be optimized.

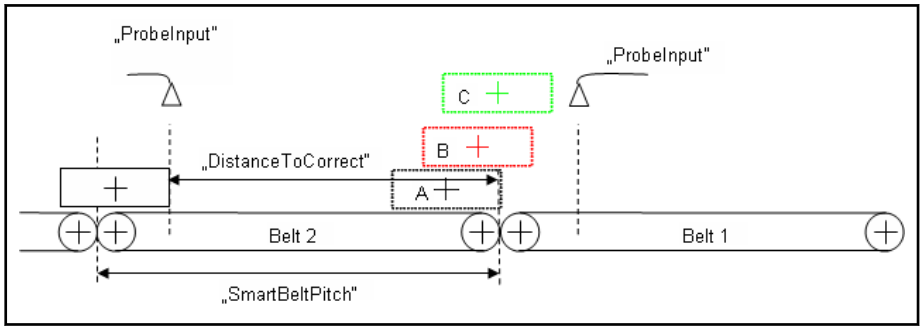


Fig.6-146: Different Distances for Correction

Slippage

Slippage occurs when the product cannot accelerate as quickly as the SmartBelt. In this case, an error in the amount of correction necessary is caused. The SmartBelt function block requires the maximum acceleration as an input to determine how much correction is possible and to limit the acceleration accordingly. However, often the possible acceleration is not known. The basic relationship between possible acceleration and friction is given in chapter "Friction" on page 256 and allows for a estimation of a possible value.

A method does exists to determine if slippage occurs. Each SmartBelt is provided with the output "CorrectionValue" that indicates the correction amount

of the SmartBelt. It also has output "ProbePosition" that identifies the measured position of the product. Now, the SmartBelt should be configured in a way that the correction is done in a forward direction ("PercentForwardLimit" set equal to 100% and "MaxCorrectionDistPos" equal to the SmartBelt length).

Afterwards, multiple products are feed to the SmartBelt in such a way that their position has the maximum offset compared to the master. In this case, the output "AccelLimited" is set to TRUE to indicate that the maximum acceleration had been used (if the output is not set to TRUE, the input "MaxAcceleration" for the maximum acceleration has to be reduced till the output becomes TRUE).

Once the maximum acceleration had been reached, output "ProbePosition" should be noted as well as "CorrectionValue". Even the output "ProbePosition" of the SmartBelt following the first SmartBelt should be noted. Now, the simple rule is that the "ProbePosition" value of the first SmartBelt minus "CorrectionValue" of the first SmartBelt should match the value of the "ProbePosition" value of the second SmartBelt. If the values differ significantly, the assumption is that slippage occurred. In this case, input "MaxAcceleration" must be reduced until no slippage occurs.

In case the values match, the acceleration can even be increased to get more correction capability. This can lead up to a point, where the maximum acceleration is never reached. In other words: The complete correction can be done on one belt.

6.9.6 Common Mistakes and Misunderstandings

Question	Answer:
My drive does not synchronize or takes very long to synchronize	When the drive is switched to linear mode, the units change, but the default values are kept the same. The acceleration and velocity for synchronizing are usually very small, slowing synchronization down. Increase the values for A-0-2790 and A-0-2791 (IndraMotion MLC) or P-0-0142 and P-0-0143 (IndraMotion MLD)
Error F2002028 (IndraMotion MLC) or F2028 (IndraMotion MLD)"Excessive control deviation" occurs permanently.	Set the parameters A-0-0032, A-0-0033 and A-0-0034 (IndraMotion MLC) or S-0-0038, S-0-0039 and S-0-0138 (IndraMotion MLD) to proper values. The default values for those parameters fit rotary scaling, but with linear scaling, they are usually too small and need adjustment
The function block is configure to correct 100%, but the product is not in the correct position after it left the belt	There can be multiple reasons: First, the phase difference might have been too large to be corrected within one belt length. In this case, the outputs "NegCorrLimited", "PosCorrLimited" or "AccelLimited" are set to TRUE. Second, if the values "MaxCorrectionDistancePos", "MaxCorrectionDistanceNeg" or "MaxAcceleration" are set too high, the products might not be able to physically follow the belt and start sliding. This condition is not indicated by the status outputs. Measures can be taken to increase the friction (vacuum belts) or more belts must be used to compensate for the errors on multiple belts. Third, if two or more products are on the belt, refer to the next question

Question	Answer:
When there are two or more products on the belt, the second or third product is not corrected properly	The second product has to wait until the first product has left the belt. Then, it is corrected. If the remaining distance is too small, the required correction cannot be achieved
If two products are located end-to-end on the belt, the second product is not corrected	If the products are located end-to-end, the touch probe function cannot detect the second product. The SmartBelt only sees one product. It has to be ensured that the process preceding the SmartBelts separates the products with a minimum distance which enables the touch probe functionality to detect the next product
Everything is configured, but the SmartBelt does not perform any correction and no error is present	Set bit 9 of parameter P-0-0226 in the drive and switch to phase 2 and then to phase 4 again. This automatically enables the touch probe functionality. Otherwise, the touch probe functionality is configured but not enabled. This leads to no touch probe measurements and hence, the SmartBelt does not recognize any product

6.10 Function blocks to create cam tables for technology functions

6.10.1 Introduction

The function blocks create cam tables. The MB_PilgrimStepCalcType01 calculates a cam table with up to 1,024 data points as well as the distances of the forward and backward transport as well as stop positions for the step-back operation. The PrintLengthCorrCalcType01 calculates a cam table with up to 1,024 data points for the technology function "print length correction" to change the length of a print on the printing material (web or shed).

6.10.2 MB_PilgrimStepCalcType01

Brief Description

The function block calculates a cam table with up to 1,024 data point elements as well as the distances of the forward and backward transport as well as stop positions for the step-back operation. It considers the possibility that the synchronous range of the printing cylinder can be bigger than the format length to be printed. In case of offset print works, the plate length for example is specified due to the mechanical fixture and therefore always constant. The printing format on the printing plate can be shorter.

The cam calculated by the function block operates the infeed roll and the outfeed roll of the printing machine.

Assignment: Target system/library

IndraMotion MLC/IndraLogic XLC 12VRS	ML_TechMotion.compiled-library

Fig.6-147: Reference table of the MB_PilgrimStepCalcType01

ML_TechMotion.library

Interface Description

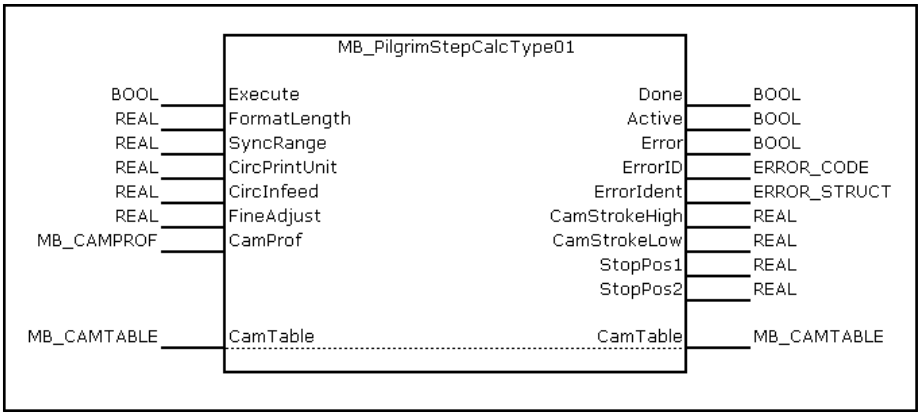


Fig.6-148: MB_PilgrimStepCalcType01 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	CamTable	MB_CAMTABLE	Structure with the calculated cam table elements and information on the number of elements
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	FormatLength	REAL	Length of the format to be printed [mm]
	SyncRange	REAL	Length of the synchronous range [mm]
	CircPrintUnit	REAL	Circumference of the printing cylinder [mm]
	CircInfeed	REAL	Circumference of the pull rolls [mm]
	FineAdjust	REAL	Fine adjustment [%] – defines the desired path tension
	CamProf	MB_CAMPROF	Cam profile selection

I/O type	Name	Type	Comment
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid, error outputs valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, it contains a detailed error description (see "Error Handling" on page 278)
	CamStrokeHigh	REAL	Distance P-0-0073 outfeed/P-0-0093 infeed P-0-0093 is specified via the function block MC_CamIn / input "CamShaftDistance" P-0-0073 is specified via the function block MB_WriteRealParameter
	CamStrokeLow	REAL	Distance P-0-0073 infeed/P-0-0093 outfeed P-0-0093 is specified via the function block MC_CamIn / input "CamShaftDistance" P-0-0073 is specified via the function block MB_WriteRealParameter Acceleration used related to the input "MaxAcc" in [%]
	StopPos1	REAL	Point of reversal 1 [°] with velocity = 0
	StopPos2	REAL	Point of reversal 2 [°] with velocity = 0

Fig.6-149: Interface variables of the MB_PilgrimStepCalcType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
FormatLength	REAL	>0.0	n.def.	100.0	Rising edge at "Execute"
SyncRange	REAL	>0.0	n.def.	100.0	Rising edge at "Execute"
CircPrintUnit	REAL	>0.0	n.def.	100.0	Rising edge at "Execute"
CircInfeed	REAL	>0.0	n.def.	100.0	Rising edge at "Execute"
FineAdjust	REAL	0.0	100.0	1.0	Rising edge at "Execute"
CamProf	MB_CAMPROF			CAM_PROF_STANDARD	Rising edge at "Execute"

Fig.6-150: Min./max. and default values of the MB_PilgrimStepCalcType01 function block

The enumeration chapter 5.14.3 "MB_CAMPROF" on page 156 selects between different cam profiles.

When selecting the cam profile "CAMPROF_STANDARD", the linear path has to be disabled since it is a profile without gear reduction. The linear path for the profiles "CAMPROF_ADDITIVE" and "CAMPROF_ADDITIVE_SEGMENTABLE" has to be ticked and the reduction has to be set to "1", since they are profiles with gear reduction.

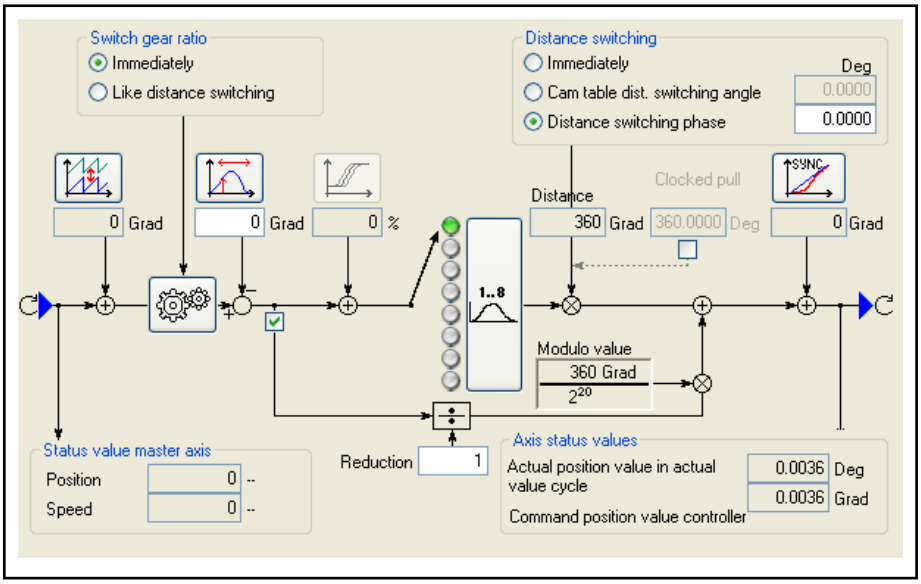


Fig.6-151: Enabling the linear path

The structure chapter 5.14.4 "MB_CAMTABLE" on page 157 is provided with the "CamTable" array containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

Functional Description

If not the complete circumference of a printing cylinder is covered with a printing plate or printing block, printed and non-printed parts alternate in continuous operation mode. To avoid these printing gaps, the web is decelerated by feed rolls, retracted by a defined distance and accelerated again to synchronous velocity so that the new print immediately follows the old one.

The compensation motion calculated by the function block is characterized by a continuity up to the acceleration.

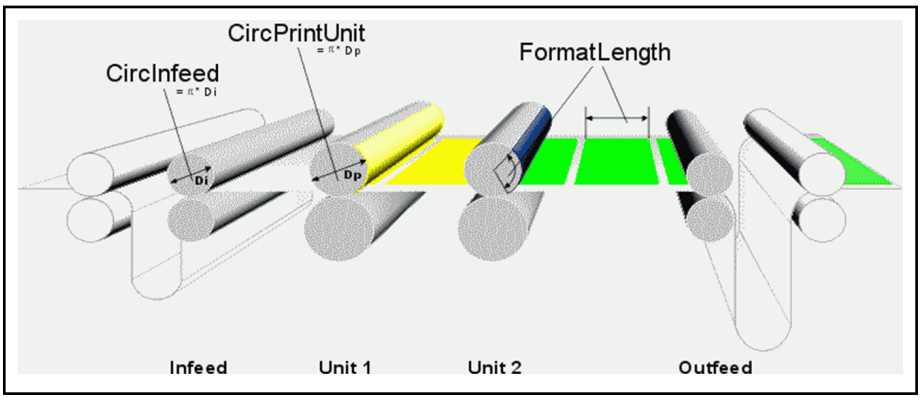


Fig.6-152: Schematic diagram of the step-back

Prerequisites to use the function block

- The function block can only be used for axes with interpolation in the drive
- The drive function "Clocked pull roll“ (see drive documentation P-0-0086) has to be switched on by the user

The MB_PilgrimStepCalcType02 calculates a cam table with up to 1,024 data point elements and both the distance factors "CamStrokeHigh" and "CamStrokeLow" and the stop positions "StopPos1" and "StopPos2" only once after the function block was enabled for processing using "Execute". The function block uses the mechanical specification of the circumference of the infeed or outfeed roll "CircInfeed", the circumference of the printing cylinder "CircPrintUnit", the printing plate length "FormatLength" on the printing cylinder and the total length of the synchronous range "SyncRange". The number of data point elements of a cam table is set by the "NumberOfElements" element of the "CamTable" structure.

It is assumed that the circumference of the infeed roll and the circumference of the outfeed roll are identical (CircInfeed).

According to the advancing specified by the fine adjustment "FineAdjust", two distances are calculated to keep the web tension constant. The higher distance is for the roll located in the front with regard to the transportation direction. To calculate the distances, the function block uses the following formulae:

$$CamStrokeLow = 360° * \frac{FormatLength}{CircInfeed}$$

The "CamStrokeLow" values applies to the parameter P-0-0073 of the infeed roll and for the parameter P-0-0093 of the outfeed roll

Fig.6-153: CamStrokeLow formula

$$CamStrokeHigh = 360° * \frac{FormatLength}{CircInfeed} * \left(1 + \frac{FineAdjust}{100}\right)$$

The "CamStrokeHigh " values applies to the parameter P-0-0073 of the outfeed roll and for the parameter P-0-0093 of the infeed roll

Fig.6-154: CamStrokeHigh formula

The outputs "StopPos1" and "StopPos2" contain master axis positions where cams are provided with a point of reversal to change the format at one of these positions during a control or to change the distance with the register controller.

The cam table to be calculated is stored in the "CamTable" structure with scaling in per cent. The validity of the cam elements is signaled by the "Done" output. The cam table can be loaded to the control or drive using the chapter 5.14.2 "MB_LoadCamData" on page 154 function block.

If an error occurs while processing the function block, this is signaled by the "Error" output. The elements of the "CamTable" structure as well as the outputs "CamStrokeHigh" and "CamStrokeLow", "StopPos1" and "StopPos2" are not updated in case of error.

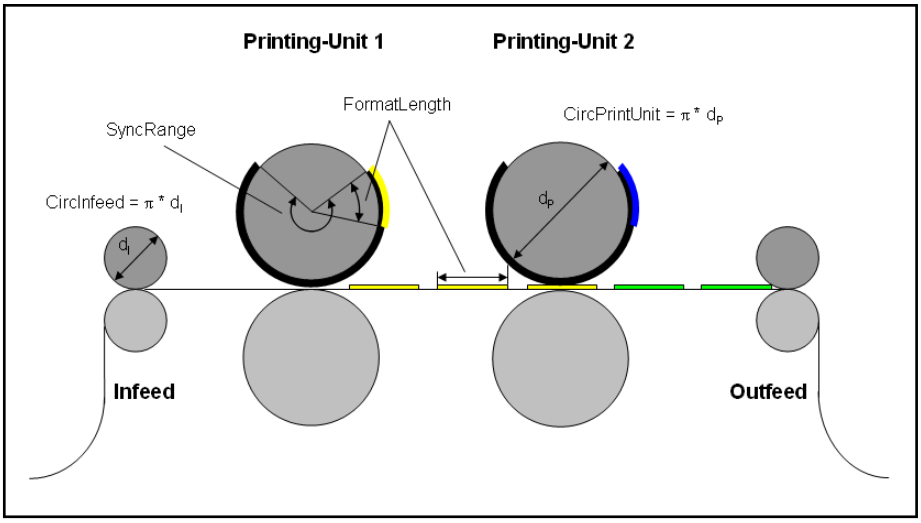


Fig.6-155: Process diagram (FormatLength < SyncRange)

Cam characteristics

The cam table of the MB_PilgrimStepCalcTyp01 function block provides the following characteristics:

- The function block is based on the MSV_CamTableType04 function block and also applies to those cases with a synchronous range of the printing cylinder bigger than the printing range
- The compensation motion for the back-transport of the web characterized by the cam table is constant up to the acceleration (s, v, a) and shows a jump in the jerk (j)
- The compensation motion begins with the first table element, then, a linear inclination which is phase-synchronous to the printing cylinder
- The algorithm to calculate the cam data points is divided into several cycles, i.e. calls the instance to avoid a considerable effect on the PLC cycle time
- The calculated distances of the function block provide the following characteristics:
  - The distances depend on the printing format and the circumference of the infeed roll
  - The distance "CamStrokeHigh" is provided for the parameter P-0-0093 of the infeed roll and the parameter P-0-0073 of the outfeed roll, the distance "CamStrokeLow" is provided for the parameter P-0-0073 of the infeed roll and the parameter P-0-0093 of the outfeed roll

To generate the web tension, the distance for the forward transport of the outfeed roll and the distance for the back-transport of the infeed roll exceed the distance for the forward transport of the infeed roll and the distance for the back-transport of the outfeed roll by the fine adjustment.

There are three different types of cams:

- **FormatLength = SyncRange = CircPrintUnit**

  The printing plate or printing block covers the complete synchronous zone of the cylinder and the synchronous zone covers the complete circumference of the printing cylinder. In this case, there is no compensation motion. If "SyncRange" is equal to "CircPrintUnit", "FormatLength" must be provided with the same value, since no compensation motion

can be executed. If "FormatLength" is lower than "SyncRange", the function block cancels the processing outputting an error.

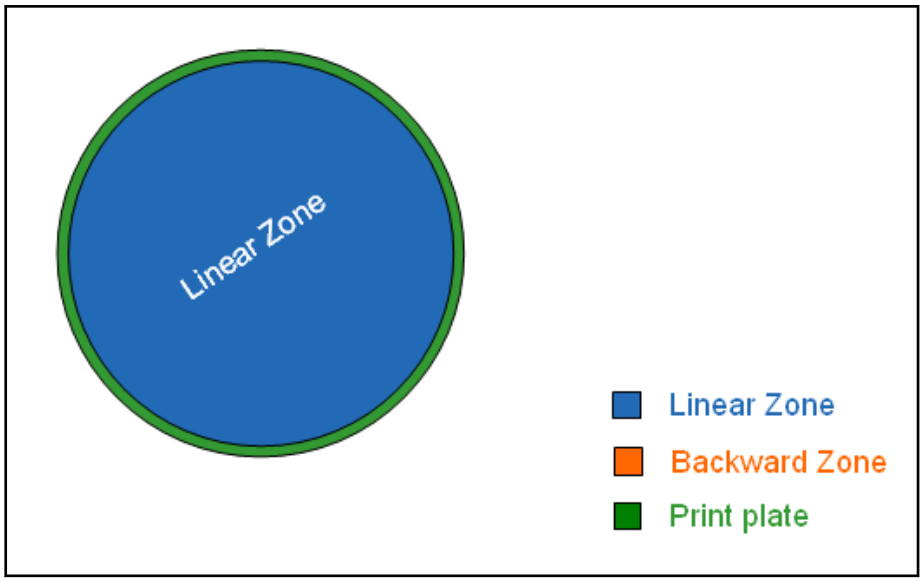
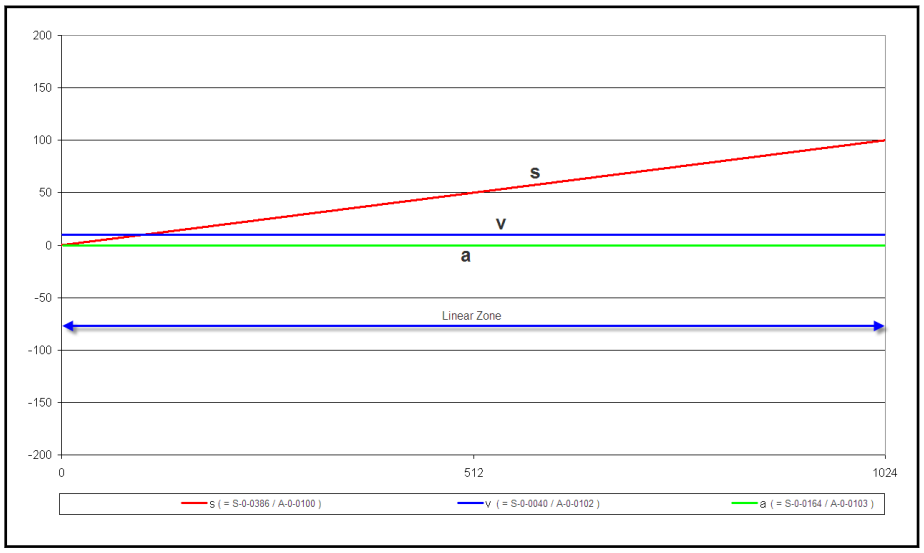


Fig.6-156: MB_PilgrimStepCalcType01 with FormatLength = SyncRange = CircPrintUnit



FormatLength = 2000.0; SyncRange = 2000.0; CircPrintUnit = 2000.0; CircInFeed = 2000.0

Fig.6-157: MB_PilgrimStepCalcType01 MotionProfile

- **FormatLength = SyncRange**

  The printing plate or printing block covers the complete synchronous zone of the cylinder. The complete remaining cylinder area can be used for compensation motion.

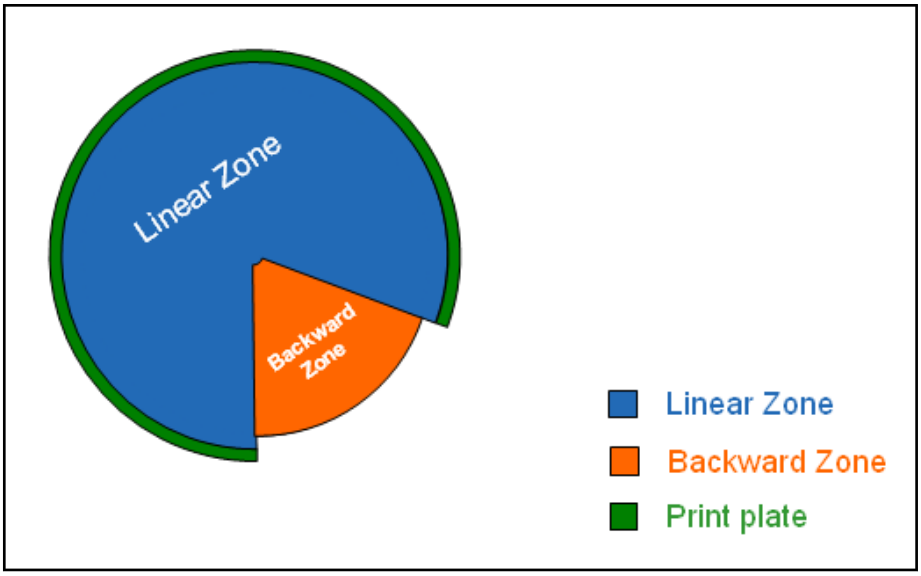
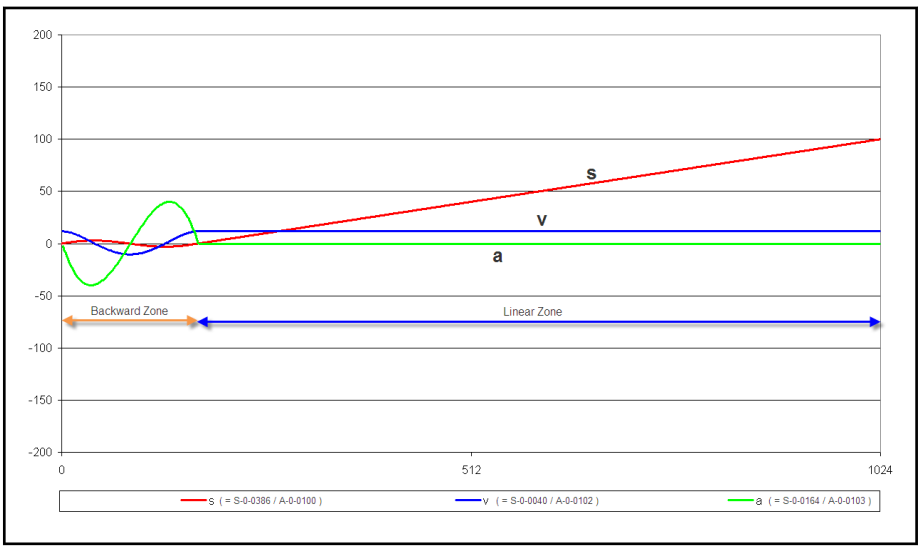
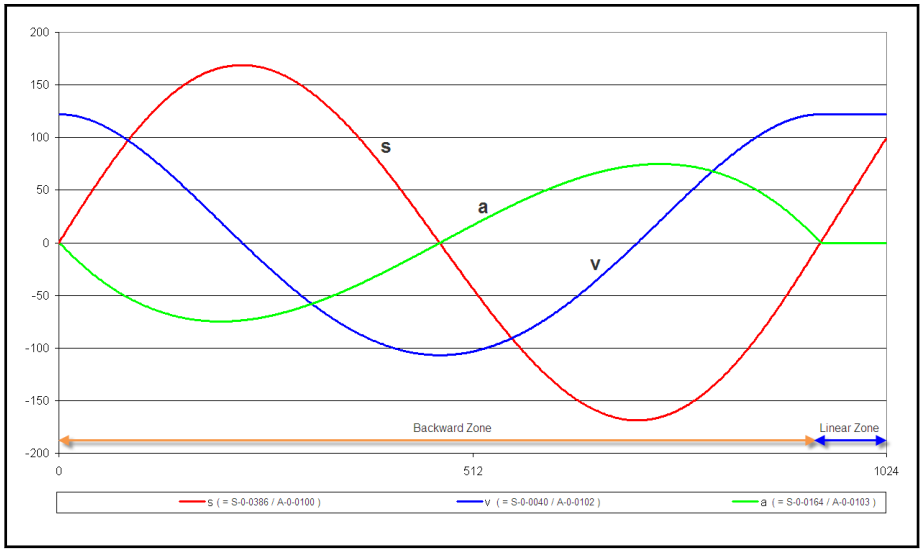


Fig.6-158: MB_PilgrimStepCalcType01 with FormatLength = SyncRange



FormatLength = 500.0; SyncRange = 500.0; CircPrintUnit = 600.0; CircInFeed = 600.0

Fig.6-159: MB_PilgrimStepCalcType01 MotionProfile



FormatLength = 200.0; SyncRange = 200.0; CircPrintUnit = 2500.0; CircInFeed = 2500.0

Fig.6-160: MB_PilgrimStepCalcType01 MotionProfile

- **FormatLength < SyncRange**

  The printing plate or printing block only covers partially the synchronous zone of the cylinder. This has to move up to the end of the synchronous range with synchronous velocity. Only if the cylinder has no contact with the web anymore, the material web can be decelerated, retracted and again accelerated. Here, the additional distance of the non-printing synchronous range has to be retracted.

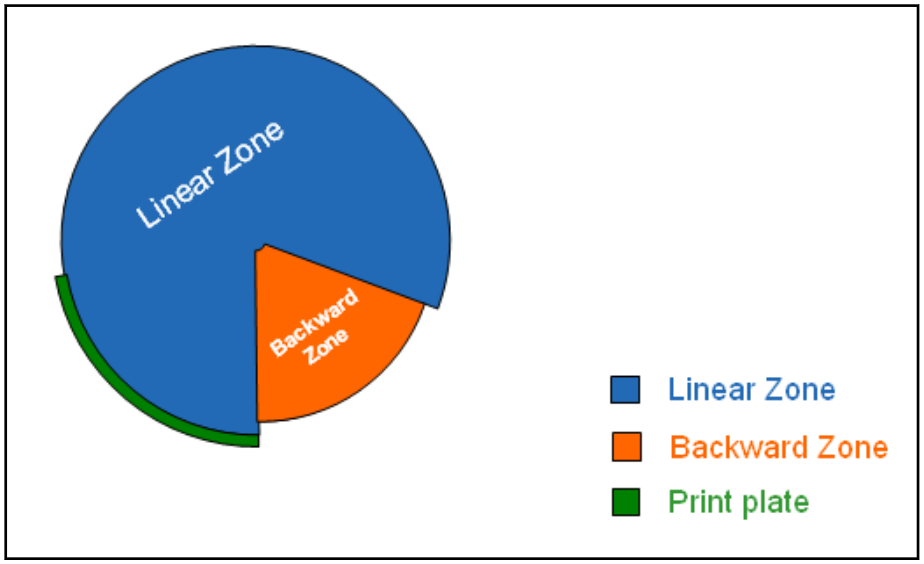
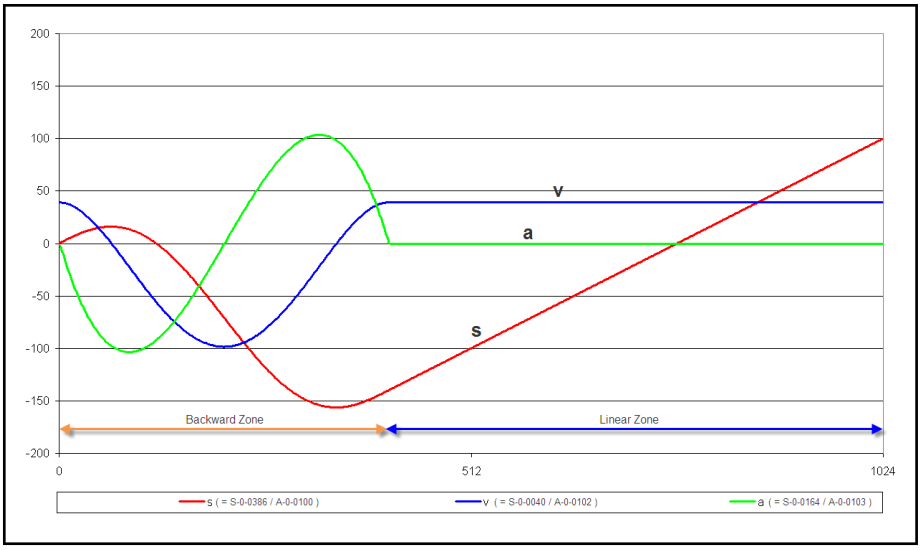


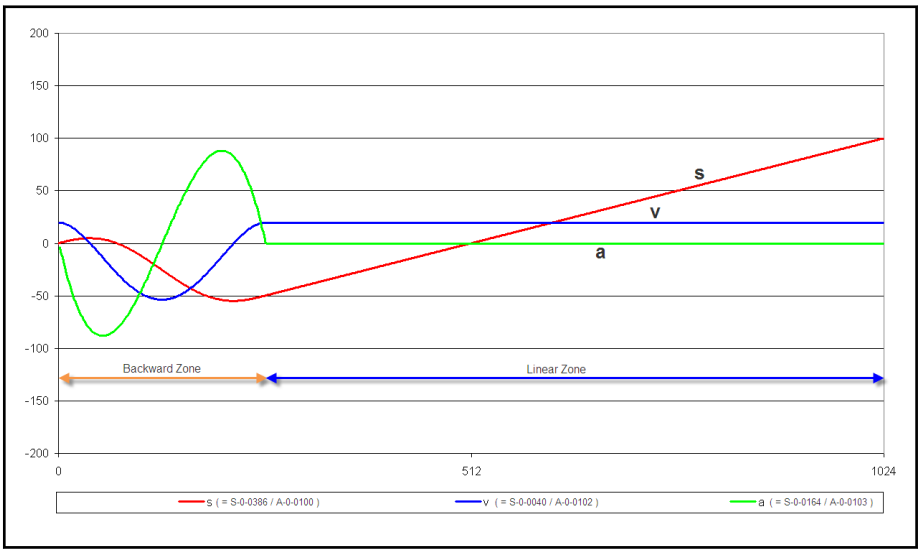
Fig.6-161: MB_PilgrimStepCalcType01 with FormatLength < SyncRange

ML_TechMotion.library



FormatLength = 500.0; SyncRange = 1200.0; CircPrintUnit = 2000.0; CircInFeed = 3000.0

Fig.6-162: MB_PilgrimStepCalcType01 MotionProfile



FormatLength = 1000.0; SyncRange = 1500.0; CircPrintUnit = 2000.0; CircInFeed = 3000.0

Fig.6-163: MB_PilgrimStepCalcType01 MotionProfile

Error Handling

The function block uses the F_RELATED_TABLE, 16#0170x error table. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#00002200	16#00000001	The "NumberOfElements" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002260	16#00000001	The input "FormatLength" is lower than the minimum value
INPUT_RANGE_ERROR	16#00002260	16#00000002	The input "SyncRange" is lower than the minimum value

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#00002260	16#00000003	The input "CircPrintUnit" is lower than the minimum value.
INPUT_RANGE_ERROR	16#00002260	16#00000004	The input "CircInfeed " is lower than the minimum value
INPUT_RANGE_ERROR	16#00002260	16#00000005	The "FineAdjust" input is lower than the minimum or higher than the maximum
INPUT_INVALID_ERROR	16#00002261	16#00000001	Address of the "CamTable" array is zero
INPUT_INVALID_ERROR	16#00002261	16#00000001	The "FormatLength" input is higher than "CircPrintUnit"
INPUT_INVALID_ERROR	16#00002261	16#00000002	The "SyncRange" input is higher than "CircPrintUnit"
INPUT_INVALID_ERROR	16#00002261	16#00000003	Input "FormatLength" is higher than "SyncRange"
INPUT_INVALID_ERROR	16#00002261	16#00000004	SyncRange = CircPrintUnit but FormatLength != SyncRange (no space for compensation motions)

Fig.6-164: Error codes of the MB_PilgrimStepCalcType01 function block

6.10.3 MB_PrintLengthCorrCalcType01

Brief Description

The function block calculates a cam table with up to 1,024 data point elements for the technology function printing length correction.

Assignment: Target system/library

IndraMotion MLC 12VRS	ML_TechMotion.compiled-library

Fig.6-165: Reference table of the MB_PrintLengthCorrCalcType01

Interface Description

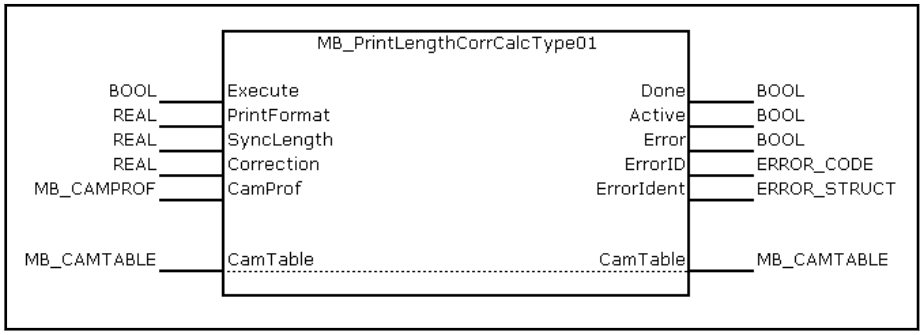


Fig.6-166: MB_PrintLengthCorrCalcType01 function block

ML_TechMotion.library

I/O type	Name	Type	Comment
VAR_IN_OUT	CamTable	MB_CAMTABLE	Structure with the calculated cam table elements and information on the number of elements
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	PrintFormat	REAL	Expected nominal printing format [mm]
	SyncLength	REAL	Synchronous format length, i.e. circumference of the printing cylinder [mm]
	Correction	REAL	Printing correction to compensate the deviation [mm]
	CamProf	MB_CAMPROF	Cam profile selection
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid, error outputs valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, it contains a detailed error description (see "Error Handling" on page 284)

Fig.6-167: Interface variables of the MB_PrintLengthCorrCalcType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
PrintFormat	REAL	>0.0	n.def.	1000.0	Rising edge at "Execute"
SyncLength	REAL	>0.0	n.def.	1500.0	Rising edge at "Execute"
Correction	REAL	-9999.9	9999.9	100.0	Rising edge at "Execute"
CamProf	MB_CAMPROF			CAM_PROF_STANDARD	Rising edge at "Execute"

Fig.6-168: Min./max. and default values of the MB_PrintLengthCorrCalcType01

The enumeration chapter 5.14.3 "MB_CAMPROF" on page 156 selects between different cam profiles.

When selecting the cam profile "CAMPROF_STANDARD", the linear path has to be disabled since it is a profile without gear reduction. The linear path for the profiles "CAMPROF_ADDITIVE" and "CAMPROF_ADDITIVE_SEGMENTABLE" has to be ticked and the reduction has to be set to "1", since they are profiles with gear reduction.

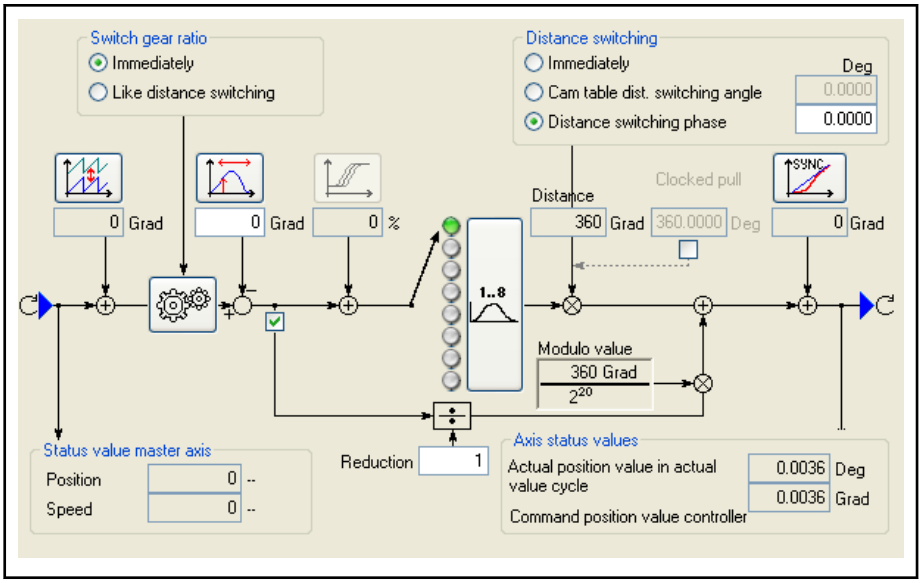


Fig.6-169: Enabling the linear path

The structure chapter 5.14.4 "MB_CAMTABLE" on page 157 is provided with the "CamTable" array containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

Functional Description

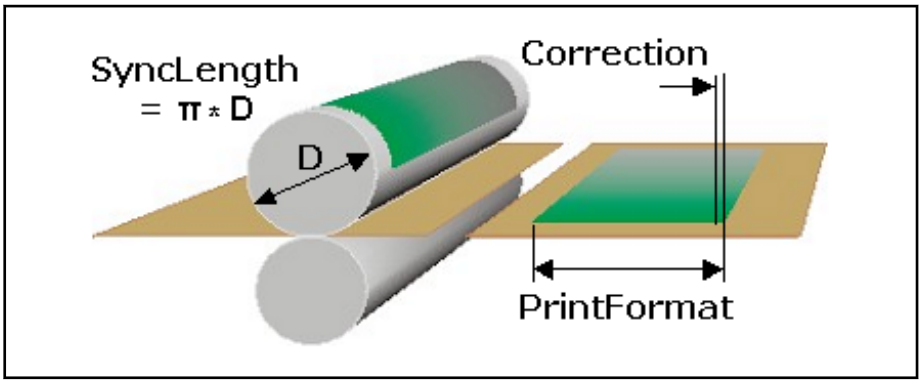


Fig.6-170: Schematic diagram of the printing length correction

The printing length correction changes the length of a printing on the printing material (material web or shed). Therefore, the velocity of the printing roll to the printing material is increased or decreased during the printing contact of the printing material. Within the following printing-free area, the position of the printing cylinder is corrected is a way that the next print can be started again in register.

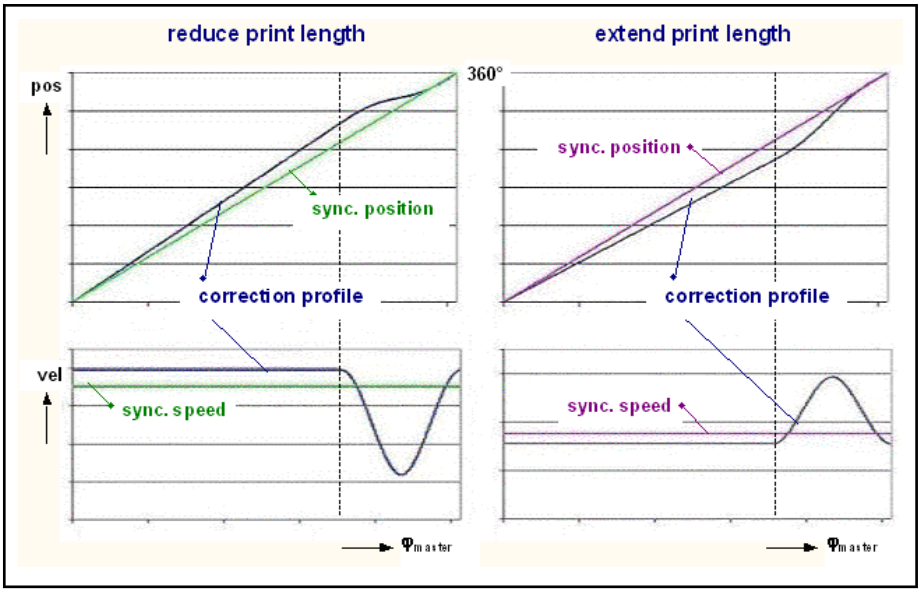


Fig.6-171: Profile of the printing length correction

With the specification of the formats "PrintFormat" and "SyncLength" and the correction "Correction", the “MB_PrintLengthCorrCalcType01“ function block non-recurrently calculates a cam table with 1,024 data point elements after enabling the processing with "Execute" based on the input values. The number of data point elements of a cam table is set by the "NumberOfElements" element of the "CamTable" structure.

The sum of printing format and printing correction may not exceed the circumference of the printing roll. The necessary positive and negative acceleration values increase with regard to the decreasing print-free zone.

- A positive correction value increases the distance traveled by the printing cylinder with the same master axis position and thus reduces the printing format

The cam table to be calculated is stored in the "CamTable" structure with scaling in per cent. The validity of the cam elements is signaled by the "Done" output. The cam table can be loaded to the control or drive using the chapter 5.14.2 "MB_LoadCamData" on page 154 function block.

If an error occurs while processing the function block, this is signaled by the "Error" output. The elements of the "CamTable" structure are not updated in case of error.

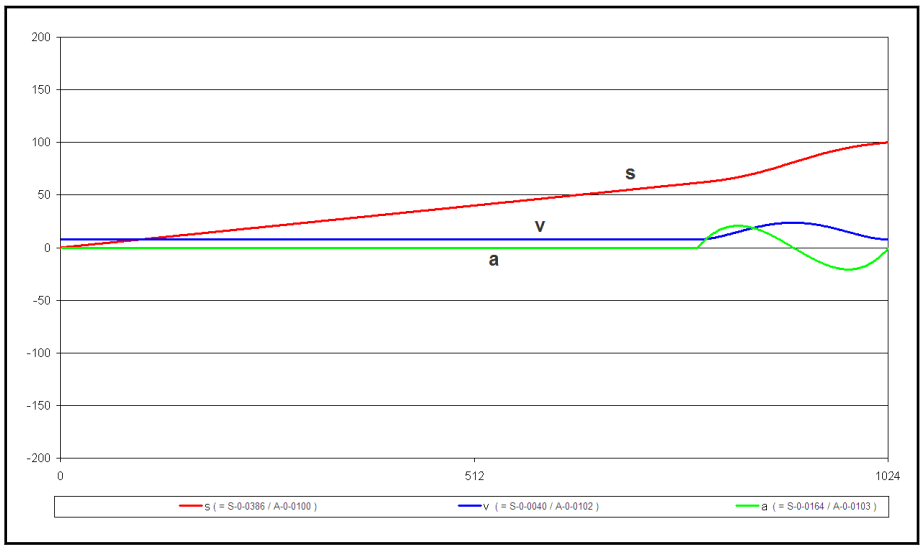
Cam characteristics

The cam table of the ML_PrintLengthCorrCalcType01 provides the following characteristics:

- The MotionProfile characterized by the cam table is constant up to the acceleration (s,v,) and shows a jump in the jerk (j)
- The correction of the printing length, i.e. the lengthening or shortening of the printing length format is based on an increase or reduction of the velocity of the printing cylinder during printing and the concluding compensation motion
- The printing format, i.e. the printing plate, may only correspond to a part of the synchronous format length, i.e. between two prints, there must be a print-free zone which is used for the compensation motion

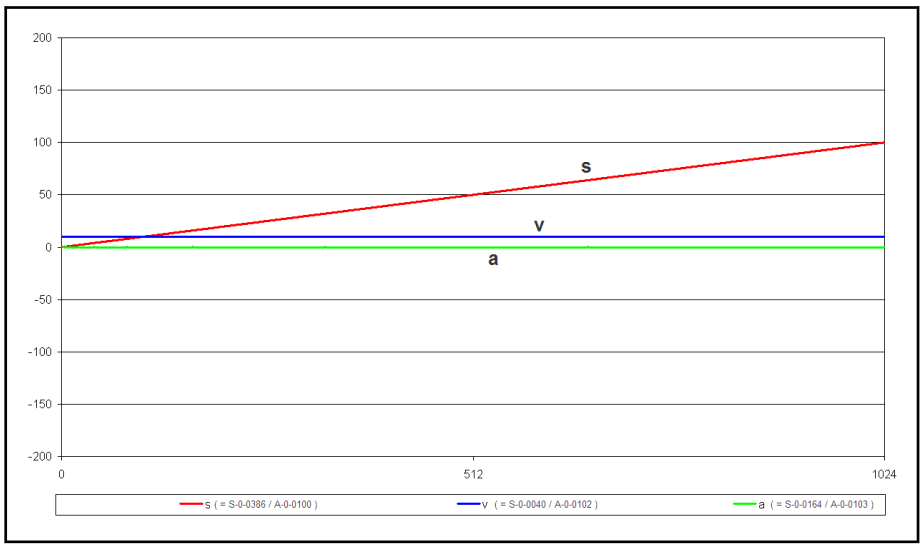
- The algorithm to calculate the cam data points is divided into several cycles, i.e. calls the function block instance to avoid a considerable effect on the PLC cycle time

MotionProfiles with different assignments of the "Correction" input



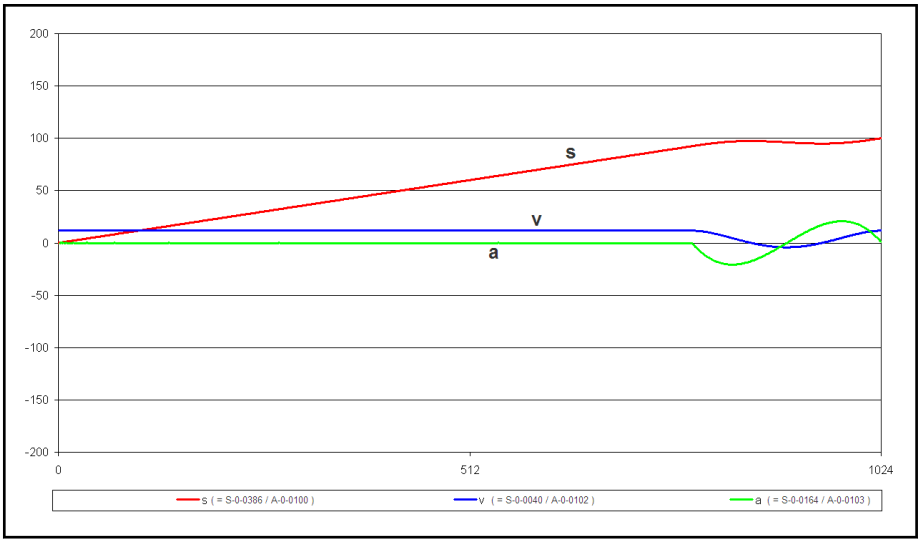
SyncLength: 1.3 * PrintLength; Correction: - 200

Fig.6-172: MB_PrintLengthCorrCalcType01 MotionProfile



SyncLength: 1.3 * PrintLength; Correction: 0

Fig.6-173: MB_PrintLengthCorrCalcType01 MotionProfile



SyncLength: 1.3 * PrintLength; Correction: 200

Fig.6-174: MB_PrintLengthCorrCalcType01 MotionProfile

Error Handling

The function block uses the error table F_RELATED_TABLE, 16x0170x. It can generate the following error messages in Additional1/Additional2:

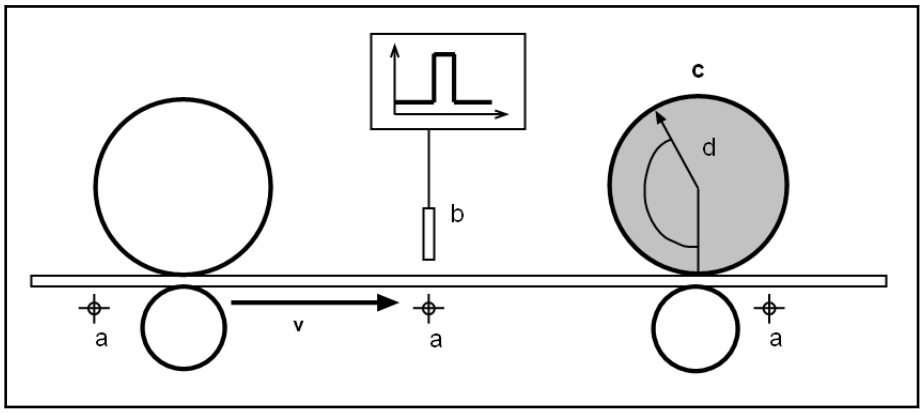
ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#00002200	16#00000001	The "NumberOfElements" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002240	16#00000001	The input "PrintFormat" is lower than the minimum value
INPUT_RANGE_ERROR	16#00002240	16#00000002	The input "SyncLength" is lower than the minimum value
INPUT_RANGE_ERROR	16#00002240	16#00000003	The "Correction" input is lower than the minimum or higher than the maximum
INPUT_INVALID_ERROR	16#00002201	16#00000001	Address of the "CamTable" array is zero
INPUT_INVALID_ERROR	16#00002241	16#00000001	Invalid calculation: (PrintFormat + Correction) > SyncLength or SyncLength <= PrintFormat

Fig.6-175: Error codes of the MB_PrintLengthCorrCalcType01 function block

7 ML_TechRegi.library/MX_TechRegi.lib

7.1 Introduction and Overview

Registration marks printed on a paper web or a paper sheet are detected using a digital input and then read in by the register controller as a position measuring value (position measurement). The position offset, velocity of the drive or the master axis drive parameters are adjusted by the register controller function blocks.



a: Registration mark
b: Register touch probe sensor
c: Printing cylinders
d: φ print

Fig.7-1: Measuring principle with web printing

The register controller controls the position of material in terms of the position of a reference axis. This can be the position of a sheet related to the printing cylinder. The closed loop control is conducted with the help of an electrical input signal at the reference axis. The electrical input signal is provided by the optically probing of a registration mark on the material. The following function blocks are supported:

Function block	ML_TechRegi for IndraMotion MLC	MX_TechRegi for IndraMotion MLD
MB_RegisterControllerType02	X	X
MB_RegisterControllerType04	X	X
MB_RegisterControllerSideType01	X	X
MB_RegiMeasuringType01	X	X
MB_RegiMeasuringType02	X	X
MB_RegiMeasuringType03	X	X
MB_RegiRegulateType01	X	X
MB_RegiRegulateType02	X	X
MB_RegiRegulateType03	X	X
MB_RegiRegulateType04	X	

ML_TechRegi.library/MX_TechRegi.lib

Function block	ML_TechRegi for IndraMotion MLC	MX_TechRegi for IndraMotion MLD
MB_RegiSetpointLockType01	X	X
IL_RegiQualityType01	X	X

Fig.7-2: Overview of function block support by the library

7.2 Register Controller Function Block

7.2.1 Overview

This section describes the register controller function block for the libraries ML_TechRegi.compiled-library (IndraMotion MLC) and MX_TechRegi.lib (IndraMotion MLD).

7.2.2 MB_RegisterControllerType02

Brief Description

The MB_RegisterControllerType02 register controller function block supports the following functionality:

- Calculate a correction value based on both measured and command value using a P- or PI-control loop
- Preset feature
- Pause feature
- Min./max limiting of the calculated control value
- Correction window feature
- Lock setpoint on demand

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-3: Reference table of the MB_RegisterControllerType02

ML_TechRegi.library/MX_TechRegi.lib

Interface Description

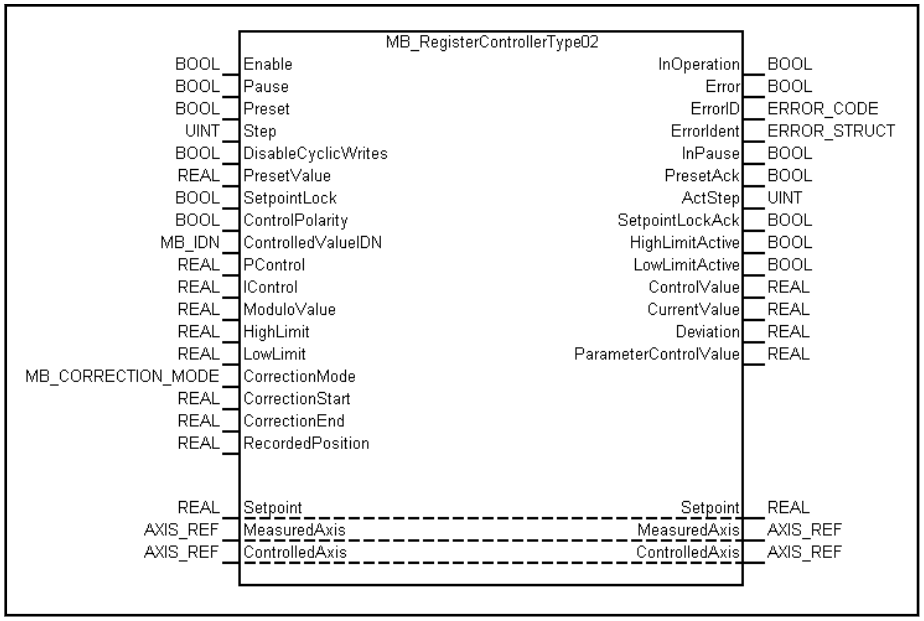


Fig.7-4: MB_RegisterControllerType02 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	Setpoint	REAL	Command value. This value is compared to "RecordedPosition"
	MeasuredAxis	AXIS_REF	Reference to the measured axis
	ControlledAxis	AXIS_REF	Reference to the controlled axis
VAR_INPUT	Enable	BOOL	Enables the Register Controller
	Pause	BOOL	The "Pause" input is only evaluated while the "Enable" input is active. The "ControlValue" and "ParameterControlValue" outputs are constant while "Pause" is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces the control loop to "PresetValue". In addition, the "Preset" input sets the "ParameterControlValue" to the value of "PresetValue"
	Step	UINT	Processing step
	DisableCyclicWrites	BOOL	Disables cyclic writing to the selected parameter in "ControlledValueIDN" when TRUE
	PresetValue	REAL	Value passed to the "ControlValue" and "ParameterControlValue" outputs on the rising edge of the "Preset" input
	SetpointLock	BOOL	Enables set point lock. If TRUE, the next measured mark will be latched
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	ControlledValueIDN	MB_IDN	SERCOS IDN of the controlled parameter (e.g. P-0-0691). The following IDNs are supported: **IndraMotion MLD**: P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695 **IndraMotion MLC**: A-0-2730, P-0-0690, A-0-2615, P-0-0691, A-0-2610, P-0-0692, A-0-2600, P-0-0694, A-0-2605, P-0-0695, A-0-2620
	PControl	REAL	Proportional gain of the PI-controller
	IControl	REAL	Integral time T_n of the PI-controller.
	ModuloValue	REAL	Modulo value from measured axis
	HighLimit	REAL	Maximum value for "ControlValue"
	LowLimit	REAL	Minimum value for "ControlValue"
	CorrectionMode	MB_CORREC-TION_MODE	CORR_ABS = absolute correction window [CorrectionStart .. CorrectionEnd]. CORR_REL = relative correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_REL_ABS = relative start .. absolute end correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "CorrectionEnd"] CORR_ABS_REL = absolute start .. relative end correction window [Start = "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_WIN_DISABLED = Correction window is disabled
	CorrectionStart	REAL	Start of the correction window
	CorrectionEnd	REAL	End of the correction window
	RecordedPosition	REAL	Measured value of axis position
VAR_OUTPUT	InOperation	BOOL	Function block is in operation
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "ControlValue" and "ParameterControlValue" outputs maintain their values
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ControlValue" and "ParameterControlValue" outputs
	ActStep	UINT	Step Value, counting up with every new value at the output "ControlValue"
	SetpointLockAck	BOOL	"Setpoint" is locked. Current value in "RecordedPosition" is copied to "Setpoint"
	HighLimitActive	BOOL	High limit is active. "ControlValue" would be higher, but is limited to "HighLimit"

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	LowLimitActive	BOOL	Low limit is active. "ControlValue" would be lower, but is limited to "LowLimit"
	ControlValue	REAL	Control value calculated by the register controller
	CurrentValue	REAL	The current recorded position used in the register controller
	Deviation	REAL	"Deviation" = ("Setpoint" - "CurrentValue") of the register controller
	ParameterControlValue	REAL	The register controller calculates a value based on the selected parameter in input "ControlledValueIDN" and stores the result in output "ParameterControlValue". This value is written to the selected parameter input "ControlledValueIDN" when input "DisableCyclicWrites" = FALSE

Fig.7-5: Interface variables of the MB_RegisterControllerType02 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	"Enable" TRUE (continuous)
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
DisableCyclicWrites	BOOL			FALSE	Continuous
SetpointLock	BOOL			FALSE	Continuous
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
ControlPolarity	BOOL			FALSE	Continuous
ControlledValueIDN	MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
Setpoint	REAL	n.def.	n.def.	0.0	Change of "Step"
PControl	REAL	0.0	n.def.	1.0	Continuous
IControl	REAL	0.0	n.def.	0.0	Continuous
ModuloValue	REAL	0.0	n.def.	0.0	Rising edge at "Enable"
HighLimit	REAL	LowLimit	n.def.	0.0	Continuous
LowLimit	REAL	n.def.	HighLimit	0.0	Continuous
CorrectionMode	MB_CORRECTION_MODE			CORR_REL	Continuous
CorrectionStart	REAL	0.0	ModuloValue	0.0	Continuous

ML_TechRegi.library/MX_TechRegi.lib

Name	Type	Min. value	Max. value	Default value	Effective
CorrectionEnd	REAL	0.0	ModuloValue	0.0	Continuous
RecordedPosition	REAL	n.def.	n.def.	0.0	Change of "Step"

Fig.7-6: Overview of min./max. values and default values of the inputs

Time diagram

The following diagram shows the signal-time diagram of MB_RegisterControllerType02 including "Pause" and "Preset" functions.

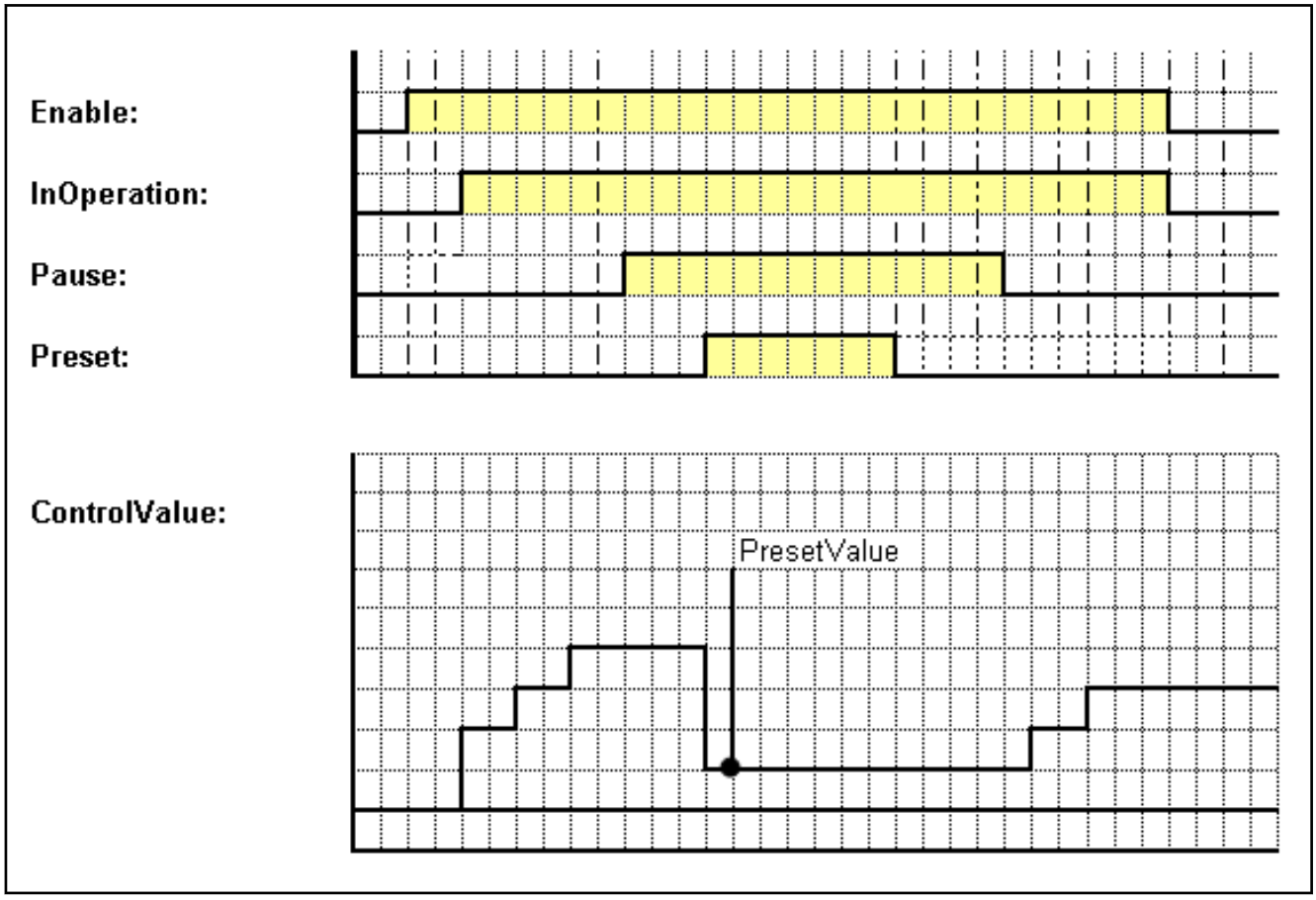


Fig.7-7: Time diagram of the MB_RegisterControllerType02 function block with the functions "Pause" and "Preset"

Functional Description

A sensor detects marks, perforations, cuts or pasted joints on the material and provides a (binary) signal to the drive.

The touch probe feature of the drive determines the edge of the sensor signal and latches the following positions, depending on the signal selection of the touch probe:

Actual drive position parameters	S-0-0051, S-0-0053, P-0-0753
Master axis position parameters	P-0-0052, P-0-0227, P-0-0775, P-0-0776, P-0-0778

The touch probe function records positional data with a resolution of 0.5 µs. The sensor must provide a 24V signal with a rise time in the µs range. The sensor specific delay time can be compensated using the "DeadTimePosEdge" or "DeadTimeNegEdge" inputs of the MB_InitTouchProbe function block.

The register controller calculates the control deviation between the measured and setpoint positions and determines the required correction value once a new edge of the sensor signal is detected.

Changes of the correction value become active via the trapezoidal profile, velocity ramp, PT1-filter or instantaneously, depending on the selected control parameter ("ControlledValueIDN") input of the function block. The following table shows the available control parameters ("ControlledValueIDN") as well the resulting MotionProfile of all available drive operation modes with synchronization.

			Available in the following Operation Modes			
Parameter Name	**Controlled-ValueIDN**	**Motion Behavior in case the parameter value is changed**	**Velocity-Synchronization**	**Phase-Synchronization**	**Cam Shaft**	**Electronic Motion Profile**
Angle offset begin of profile	P-0-0061 A-0-2730	Changes of P-0-0061 become active via ramp using P-0-0158 (velocity). If P-0-0158=0, changes of P-0-0061 become active instantaneously.	No	No	Yes	Yes
Additive velocity command value, process loop	P-0-0690 A-0-2615	Changes of P-0-0694 become active instantaneously	Yes	No	No	No
Additive position command value, process loop	P-0-0691 A-0-2610	If P-0-0155 Bit0 = 0: Changes of P-0-0691 become active via PT1 filter using P-0-0060 (filter time constant). If P-0-0060=0, changes of P-0-0691 become active instantaneously. If P-0-0155 Bit0 = 1: Changes of P-0-0691 become active instantaneously	Yes	Yes	Yes	Yes
Additive master axis position, process loop	P-0-0692 A-0-2600	Changes of P-0-0692 is smoothed with a 1st order filter using P-0-0693(filter time constant) and then added to the sum of P-0-0054, Additive master axis position and P-0-0053, Master axis position or P-0-0052, Actual position value of measuring encoder.	Yes	Yes	Yes	Yes
Gear ratio fine adjust, process loop	P-0-0694 A-0-2605	Changes of P-0-0694 become active instantaneously	Yes	Yes	Yes	Yes
Angle offset begin of profile, process loop	P-0-0695 A-0-2620	Changes of P-0-0695 become active via PT1 filter P-0-0696 (filter time constant). If P-0-0696=0, changes of P-0-0695 become active instantaneously.	No	No	Yes	Yes

Fig.7-8: Supported Control Parameter and Behavior

The following "ControlledValueIDN" values are integrated:

- P-0-0061, A-0-2730
- P-0-0691, A-0-2610
- P-0-0692, A-0-2600
- P-0-0695, A-0-2620

Initialization

During initialization, the "Setpoint" and "RecordedPosition" inputs must have valid data.

PresetValue

For a trouble-free start of the controller outputs "ControlValue" and "ParameterControlValue", set the "Preset" input before activating the "Enable" input. During this period, the "Setpoint" and "RecordedPosition" inputs must have valid data. In addition, the upper limits and lower limits are monitored to ensure a limitation of the "ControlValue" output if the current value for "PresetValue" is outside the limits of "HighLimit" or "LowLimit".

The "PresetAck" output is active if the value in the "PresetValue" input is copied to the "ControlValue" and "ParameterControlValue" outputs. The "Preset" input has priority over the "Pause" input. The value in "PresetValue" is copied to the respective outputs on a rising edge of the "Preset" input if "Enable" is active. If successful, the "PresetAck" output is set to TRUE.

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. In this case, the deviation ("Setpoint" – "RecordedPosition") is set to 0. Therefore, the outputs "ControlValue" and "ParameterControlValue" do not change anymore. This can be used to maintain the values of the "ControlValue" and "ParameterControlValue" outputs. The "ActStep" output is updated even if the "Pause" input is active. As long as the "Pause" input is active, the "InPause" output is TRUE.

Lock Setpoint

A manual to Automatic mode change is achieved by latching to the next register mark that is detected. For example, when a new web is pulled into position and the register mark is detected, the "SetpointLock" input is set to TRUE. The value in "Setpoint" is overwritten with the value in "RecordedPosition" the next time the "Step" input changes.

Correction Window

The correction window is an angular range relating to the measuring axis. The register controller controls the control deviations within this window. The position and size of the correction window are set with the help of the following inputs:

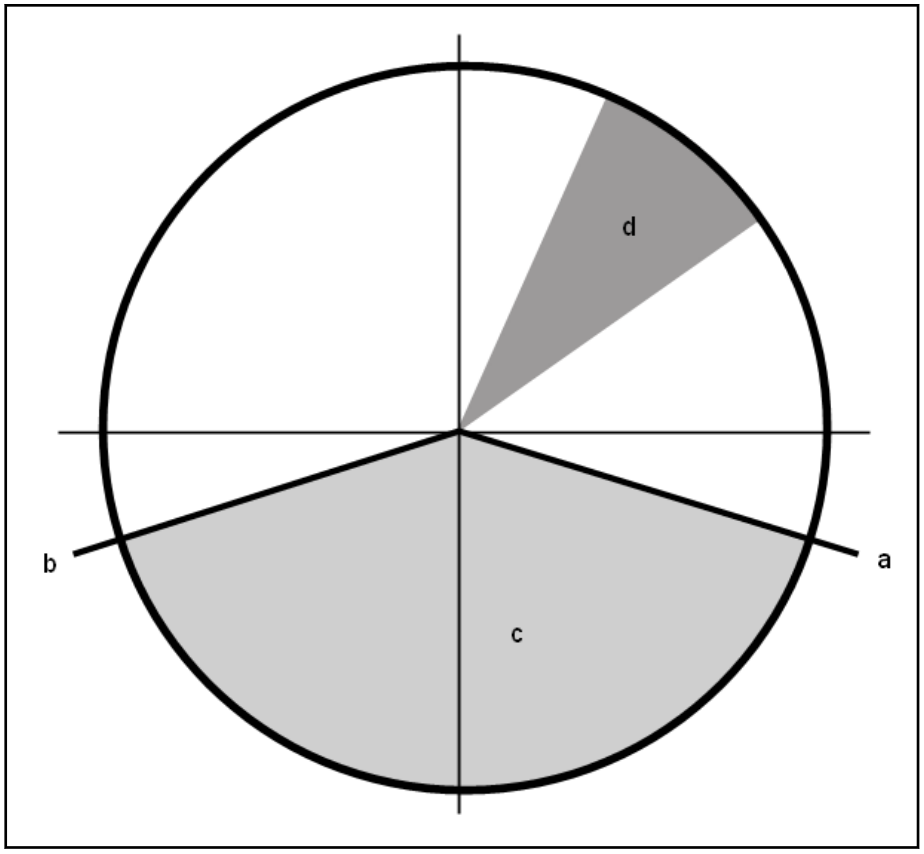
The "CorrectionMode" input sets the correction window either absolute or relative. If the "CorrectionMode" is set to CORR_WIN_DISABLED, the correction window is inactive.

The "CorrectionStart" input defines the start of the correction window.

The "CorrectionEnd" input defines the end of the correction window.

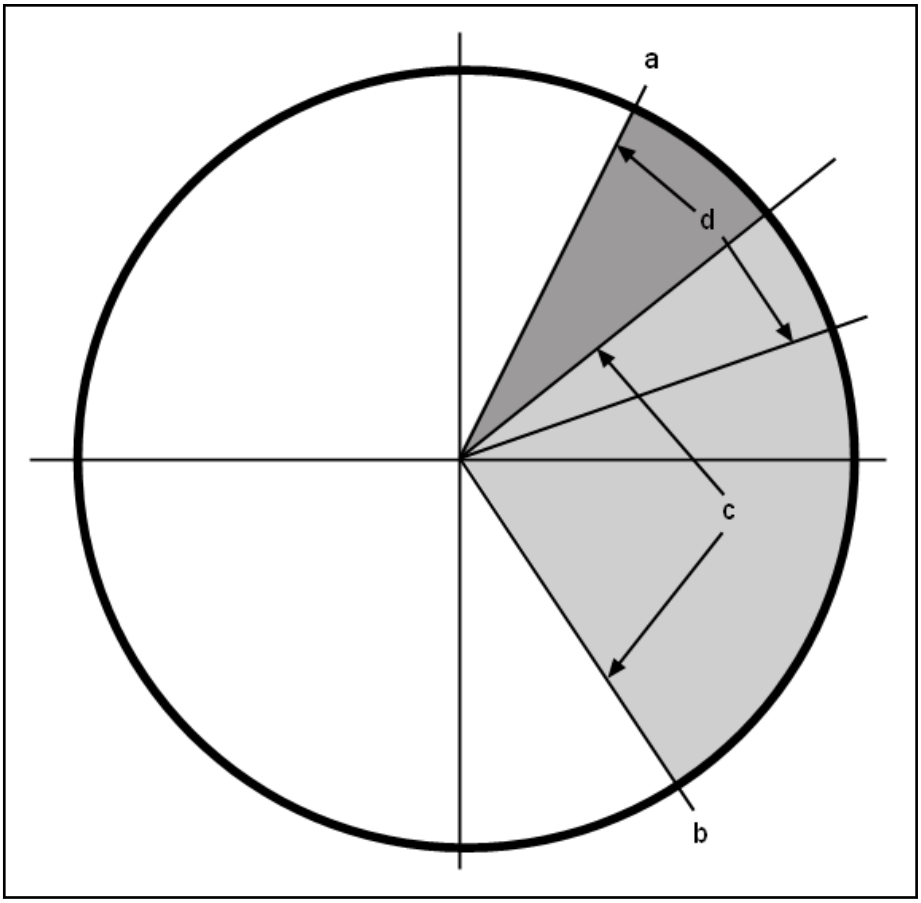
The correction window supports only a positive modulo range (e.g., 0...+360 degrees) of the measuring axis.

ML_TechRegi.library/MX_TechRegi.lib



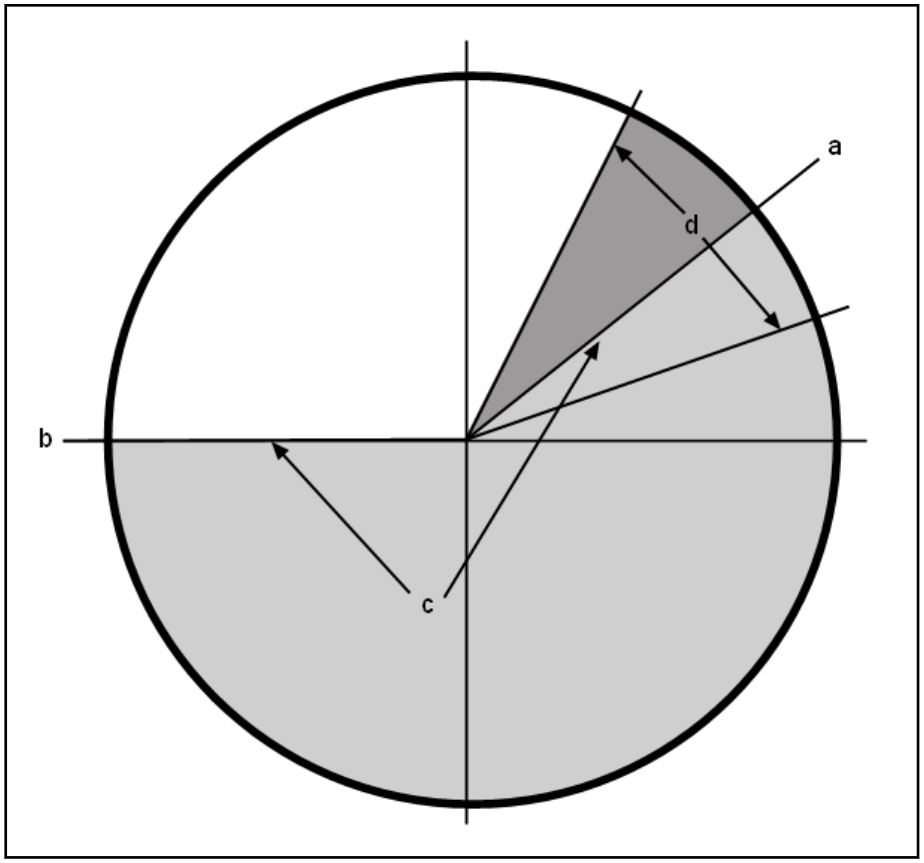
a: Start correction window
b: End correction window
c: Correction Window
d: Expectation window
Fig.7-9: Correction Window Absolute

ML_TechRegi.library/MX_TechRegi.lib



a: Start correction window = measuring end
b: Relative end of correction window to measuring end
c: Correction Window
d: Expectation window

Fig.7-10: Correction Window Relative



a: Start correction window = measuring end
b: End of correction window = perm. angle
c: Correction Window
d: Expectation window

Fig.7-11: Correction Window Relative Start Absolute End

Normal mode

The RegisterController only calculates a new value if the "Step" input changes. The "PControl" and "IControl" controller parameters can be changed continuously.

If the value in the "ControlValue" output exceeds or falls below the values of outputs "HighLimit" or "LowLimit", then "ControlValue" is limited to one of the following limit values:

- If the value in "ControlValue" is positive, then the value in "HighLimit" is used.
- If the value in "ControlValue" is negative, then the value in "LowLimit" is used.

If the register controller has an integral action, then an anti-reset wind-up is carried out and the integral action is no longer summed. In this case the "HighLimitActive" and "LowLimitActive" outputs are set to TRUE.

Required Parameterization

IndraMotion MLC parameterization

If a real axis is used as the controlled axis ("ControlledAxis" input), then the parameter used in the "ControlledValueIDN" input must be one of the following supported P-Parameters:

- P-0-0690, P-0-0691, P-0-0692, P-0-0694, and P-0-0695

Additionally, this P-Parameter must be configured in the MDT of the cyclic data channel in the drive.

If a virtual axis is used as the controlled axis, then the parameter used in the "ControlledValueIDN" input must be one of the following supported A-Parameters:

- A-0-2730, A-0-2615, A-0-2610, A-0-2600, A-0-2605, A-0-2620

IndraMotion MLD parameterization

The parameter used as controlled parameter for the "ControlledValueIDN" (P-Parameter) input must be configured in the "AxisData" structure. Thus, the axis data structure has to be activated. The following P-Parameters are supported:

- P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695

Error Handling

The function block outputs the error messages of the following function blocks used.

- MB_RegiMeasuringType01 (fig. 7-51 " Error codes of the MB_RegiMeasuringType01 function block" on page 330)
- IL_PIType02 (library RIL_LoopControl, refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries")
- MB_RegiRegulateType02 (fig. 7-81 " Error codes of the MB_RegiRegulateType02 function block" on page 349)
- MB_RegiSetpointLockType01 (fig. 7-104 " Error codes of the MB_RegiSetpointLockType01 function block" on page 367)

The function block generates the following error messages in Additional1/Additional2 using the table "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"Axis_Ref" is outside valid range(measured/controlled)
INPUT_RANGE_ERROR(16#0006)	16#0813	16#0001	"HighLimit" < "LowLimit"
INPUT_RANGE_ERROR(16#0006)	16#0813	16#0002	"PControl" < 0

Fig.7-12: Error codes of the MB_RegisterControllerType02 function blocks

7.2.3 MB_RegisterControllerType04

Brief Description

The MB_RegisterControllerType04 register controller function block supports the following functionality:

- Calculate a correction value based on both measured and command value using a P- or PI-control loop
- Preset feature
- Pause feature
- Min./max limiting of the calculated control value
- Correction window feature
- Shift register
- Dead band Kp characteristic curve
- PT1 element
- Lock setpoint on demand

ML_TechRegi.library/MX_TechRegi.lib

- Mark gating function

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-13: Reference table of the MB_RegisterControllerType04

Interface Description

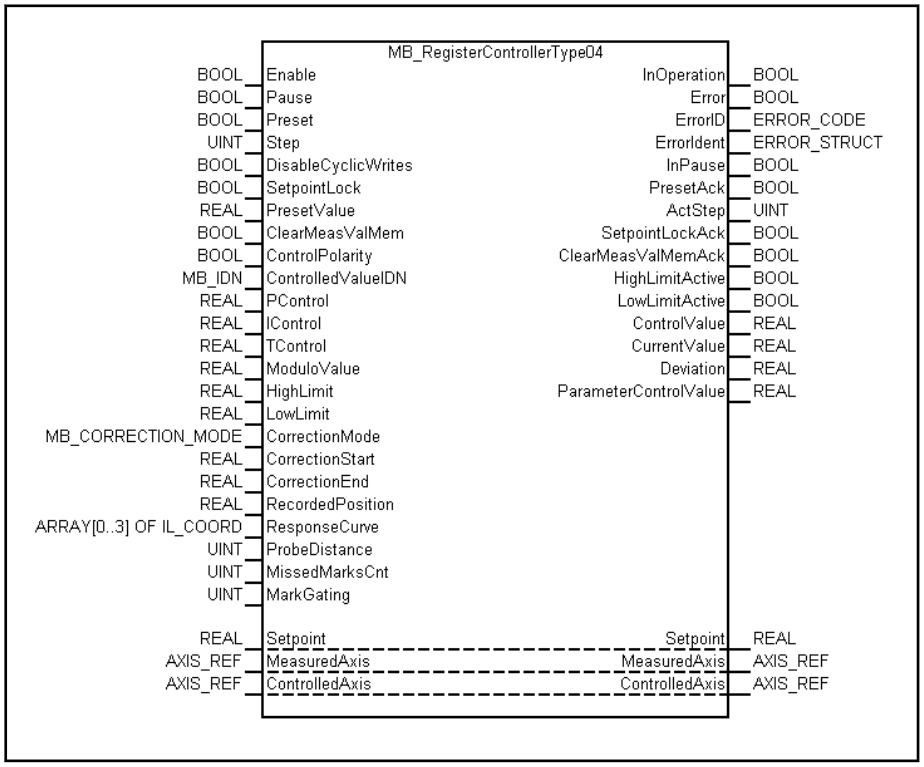


Fig.7-14: MB_RegisterControllerType04 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Setpoint	REAL	Command value. This value is compared to "RecordedPosition"
	MeasuredAxis	AXIS_REF	Reference to the measured axis
	ControlledAxis	AXIS_REF	Reference to the controlled axis
VAR_INPUT	Enable	BOOL	Enables the Register Controller
	Pause	BOOL	The "Pause" input is only evaluated while the "Enable" input is active. The "ControlValue" and "ParameterControlValue" outputs are constant while "Pause" is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces the control loop to "PresetValue". In addition, the "Preset" input sets the "ParameterControlValue" to the value of "PresetValue"
	Step	UINT	Processing step

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	DisableCyclicWrites	BOOL	Disables cyclic writing to the selected parameter in "ControlledValueIDN" when TRUE
	PresetValue	REAL	Value passed to the "ControlValue" and "ParameterControlValue" outputs on the rising edge of the "Preset" input
	SetpointLock	BOOL	Enables set point lock. If TRUE, the next measured mark will be latched
	ClearMeasValMem	BOOL	Clear the memory for the measured values which is used by the touch probe distance function
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE
	ControlledValueIDN	MB_IDN	SERCOS IDN of the controlled parameter (e.g. P-0-0691). The following IDNs are supported: **IndraMotion MLD**: P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695 **IndraMotion MLC**: A-0-2730, P-0-0690, A-0-2615, P-0-0691, A-0-2610, P-0-0692, A-0-2600, P-0-0694, A-0-2605, P-0-0695, A-0-2620
	PControl	REAL	Proportional gain of the PI-controller
	IControl	REAL	Integral action time T_n of the PI-controller.
	TControl	REAL	Filter constant for the input "RecordedPosition"
	ModuloValue	REAL	Modulo value from measured axis
	HighLimit	REAL	Maximum value for "ControlValue"
	LowLimit	REAL	Minimum value for "ControlValue"
	CorrectionMode	MB_CORRECTION_MODE	CORR_ABS = absolute correction window [CorrectionStart .. CorrectionEnd]. CORR_REL = relative correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_REL_ABS = relative start .. absolute end correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "CorrectionEnd"] CORR_ABS_REL = absolute start .. relative end correction window [Start = "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_WIN_DISABLED = Correction window is disabled
	CorrectionStart	REAL	Start of the correction window
	CorrectionEnd	REAL	End of the correction window
	RecordedPosition	REAL	Position where edge occurred (in technical units)
	ResponseCurve	ARRAY[0...3] OF IL_COORD	Characteristic curve

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	ProbeDistance	UINT	The distance between marks sensor and controlled axis in format lengths (master axis revolutions, formats),
	MissedMarksCnt	UNIT	Missed mark counter if using expectation window
	MarkGating	UINT	Every n-th mark is used for the register
VAR_OUTPUT	InOperation	BOOL	Register controller in operation
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "ControlValue" and "ParameterControlValue" outputs maintain their values
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ControlValue" and "ParameterControlValue" outputs
	ActStep	UINT	Step Value, counting up with every new value at the output "ControlValue"
	SetpointLockAck	BOOL	"Setpoint" is locked. Current value in "RecordedPosition" is copied to "Setpoint"
	ClearMeasValMemAck	BOOL	Clearing the memory of the measured values is done
	HighLimitActive	BOOL	High limit is active. "ControlValue" would be higher, but is limited to "HighLimit"
	LowLimitActive	BOOL	Low limit is active. "ControlValue" would be lower, but is limited to "LowLimit"
	ControlValue	REAL	Control value calculated by the register controller
	CurrentValue	REAL	Current value used in the register controller. It is equal to Input "RecordedPosition" if "ProbeDistance" and "MarkGating" are disabled
	Deviation	REAL	"Deviation" = ("Setpoint" - "CurrentValue") of the register controller
	ParameterControlValue	REAL	The register controller calculates a value based on the selected parameter in input "ControlledValueIDN" and stores the result in output "ParameterControlValue". This value is written to the selected parameter input "ControlledValueIDN" when input "DisableCyclicWrites" = FALSE

Fig.7-15: Interface variables of the MB_RegisterControllerType04 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	"Enable" TRUE (continuous)

ML_TechRegi.library/MX_TechRegi.lib

Name	Type	Min. value	Max. value	Default value	Effective
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
DisableCyclic-Writes	BOOL			FALSE	Continuous
SetpointLock	BOOL			FALSE	Continuous
ClearMeasVal-Mem	BOOL			FALSE	Continuous
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
ControlPolarity	BOOL			FALSE	Continuous
ControlledVa-lueIDN	MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
Setpoint	REAL	n.def.	n.def.	0.0	Change of "Step"
PControl	REAL	0.0	n.def.	1.0	Continuous
IControl	REAL	0.0	n.def.	0.0	Continuous
TControl	REAL	0.0	n.def.	0.0	Continuous
ModuloValue	REAL	0.0	n.def.	0.0	Rising edge at "Enable"
HighLimit	REAL	LowLimit	n.def.	0.0	Continuous
LowLimit	REAL	n.def.	HighLimit	0.0	Continuous
CorrectionMode	MB_CORREC-TION_MODE			CORR_REL	Continuous
CorrectionStart	REAL	0.0	ModuloValue	0.0	Continuous
CorrectionEnd	REAL	0.0	ModuloValue	0.0	Continuous
RecordedPosition	REAL	n.def.	n.def.	0.0	Change of "Step"
ResponseCurve	ARRAY[0...3] OF IL_COORD	n.def.	n.def.	0.0	Continuous
ProbeDistance	UINT	0	30	0	Continuous
MissedMarksCnt	UINT	0	65.535	0	Continuous. Initializa-tion at rising edge of "Enable" and falling edge of "Pause"
MarkGating	UINT	0	65.535	1	Rising edge at "Enable". A value of "0" disables mark gat-ing (identical to a value of "1")

Fig.7-16: Min./max. and default values of the inputs

Time diagram

The following diagram shows the signal-time diagram of MB_RegisterControl-lerType04 including "Pause" and "Preset" functions.

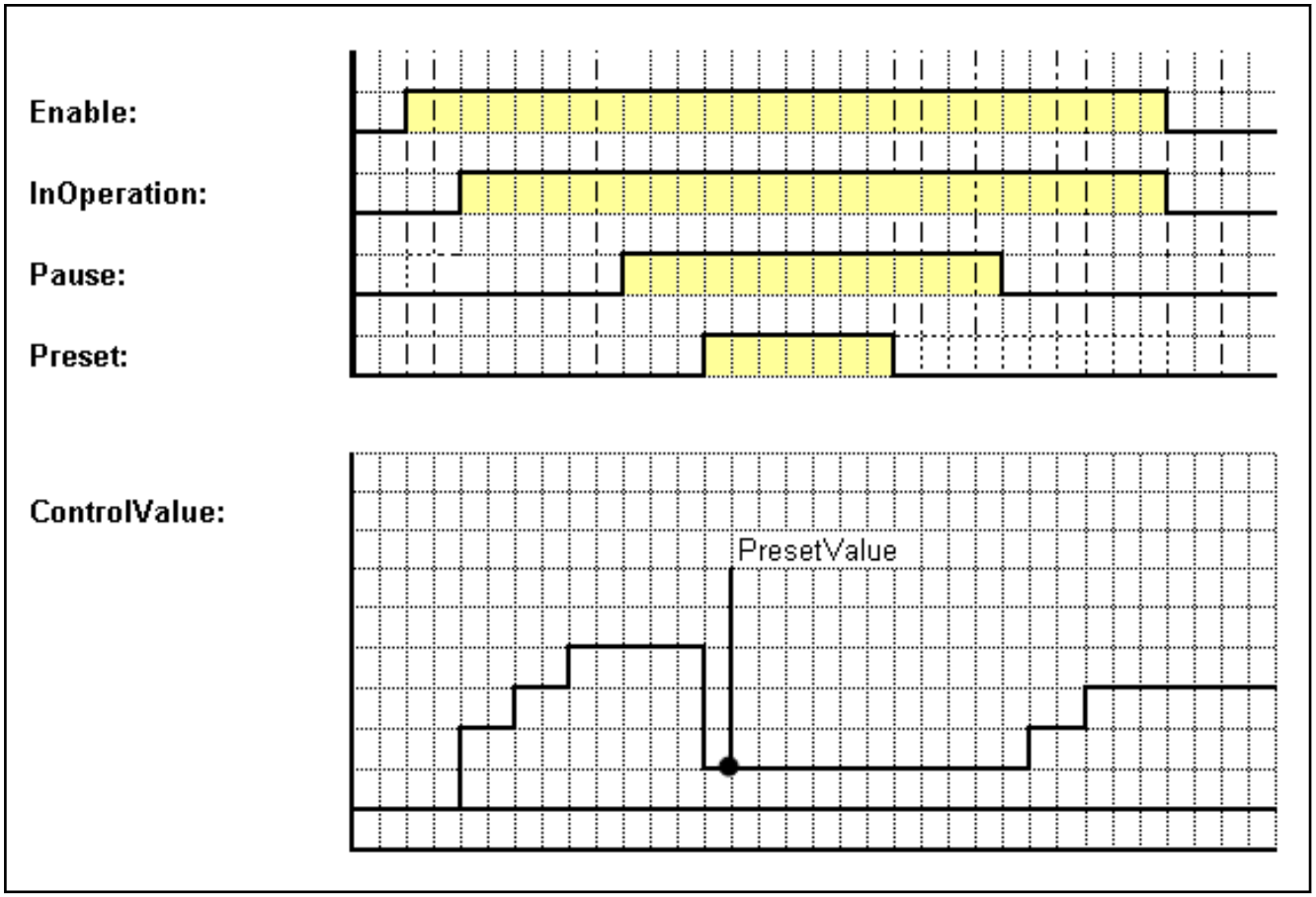


Fig.7-17: Time diagram of the MB_RegisterControllerType04 function block with the functions "Pause" and "Preset"

Functional Description

A sensor detects marks, perforations, cuts or pasted joints on the material and provides a (binary) signal to the drive.

The touch probe feature of the drive determines the edge of the sensor signal and latches the following positions, depending on the signal selection of the touch probe:

Actual drive position parameters	S-0-0051, S-0-0053, P-0-0753
Master axis position parameters	P-0-0052, P-0-0227, P-0-0775, P-0-0776, P-0-0778

The touch probe function records positional data with a resolution of 0.5 µs. The sensor must provide a 24V signal with a rise time in the µs range. The sensor specific delay time can be compensated using the "DeadTimePosEdge" or "DeadTimeNegEdge" inputs of the MB_InitTouchProbe function block.

The register controller calculates the control deviation between the measured and setpoint positions and determines the required correction value once a new edge of the sensor signal is detected.

Changes of the correction value become active via the trapezoidal profile, velocity ramp, PT1-filter or instantaneously, depending on the selected control parameter ("ControlledValueIDN") input of the function block. The following table shows the available control parameters ("ControlledValueIDN") as well the resulting MotionProfile of all available drive operation modes with synchronization.

ML_TechRegi.library/MX_TechRegi.lib

			Available in the following Operation Modes			
Parameter Name	**Controlled-ValueIDN**	**Motion Behavior in case the parameter value is changed**	**Velocity-Synchronization**	**Phase-Synchronization**	**Cam Shaft**	**Electronic Motion Profile**
Angle offset begin of profile	P-0-0061 A-0-2730	Changes of P-0-0061 become active via ramp using P-0-0158 (velocity). If P-0-0158=0, changes of P-0-0061 become active instantaneously.	No	No	Yes	Yes
Additive velocity command value, process loop	P-0-0690 A-0-2615	Changes of P-0-0694 become active instantaneously	Yes	No	No	No
Additive position command value, process loop	P-0-0691 A-0-2610	If P-0-0155 Bit0 = 0: Changes of P-0-0691 become active via PT1 filter using P-0-0060 (filter time constant). If P-0-0060=0, changes of P-0-0691 become active instantaneously. If P-0-0155 Bit0 = 1: Changes of P-0-0691 become active instantaneously	Yes	Yes	Yes	Yes
Additive master axis position, process loop	P-0-0692 A-0-2600	Changes of P-0-0692 is smoothed with a 1st order filter using P-0-0693(filter time constant) and then added to the sum of P-0-0054, Additive master axis position and P-0-0053, Master axis position or P-0-0052, Actual position value of measuring encoder.	Yes	Yes	Yes	Yes
Gear ratio fine adjust, process loop	P-0-0694 A-0-2605	Changes of P-0-0694 become active instantaneously	Yes	Yes	Yes	Yes
Angle offset begin of profile, process loop	P-0-0695 A-0-2620	Changes of P-0-0695 become active via PT1 filter P-0-0696 (filter time constant). If P-0-0696=0, changes of P-0-0695 become active instantaneously.	No	No	Yes	Yes

Fig.7-18: Supported Control Parameter and Behavior

The following "ControlledValueIDN" values are integrated:

- P-0-0061, A-0-2730
- P-0-0691, A-0-2610
- P-0-0692, A-0-2600
- P-0-0695, A-0-2620

Initialization During initialization, the “Setpoint” and “RecordedPosition” inputs must have valid data.

PresetValue For a trouble-free start of the controller outputs "ControlValue" and "ParameterControlValue", set the "Preset" input before activating the "Enable" input.

During this period, the "Setpoint" and "RecordedPosition" inputs must have valid data. In addition, the upper limits and lower limits are monitored to ensure a limitation of the "ControlValue" output if the current value for "PresetValue" is outside the limits of "HighLimit" or "LowLimit".

The "PresetAck" output is active if the value in the "PresetValue" input is copied to the "ControlValue" and "ParameterControlValue" outputs. The "Preset" input has priority over the "Pause" input. The value in "PresetValue" is copied to the respective outputs on a rising edge of the "Preset" input if "Enable" is active. If successful, the "PresetAck" output is set to TRUE.

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. In this case, the deviation ("Setpoint" – "RecordedPosition") is set to 0. Therefore, the outputs "ControlValue" and "ParameterControlValue" do not change anymore. This can be used to maintain the values of the "ControlValue" and "ParameterControlValue" outputs. The "ActStep" output is updated even if the "Pause" input is active. As long as the "Pause" input is active, the "InPause" output is TRUE.

Lock Setpoint

A manual to Automatic mode change is achieved by latching to the next register mark that is detected. For example, when a new web is pulled into position and the register mark is detected, the "SetpointLock" input is set to TRUE. The value in "Setpoint" is overwritten with the value in "RecordedPosition" the next time the "Step" input changes.

Correction Window

The correction window is an angular range relating to the measuring axis. The register controller controls the control deviations within this window. The position and size of the correction window are set with the help of the following inputs:

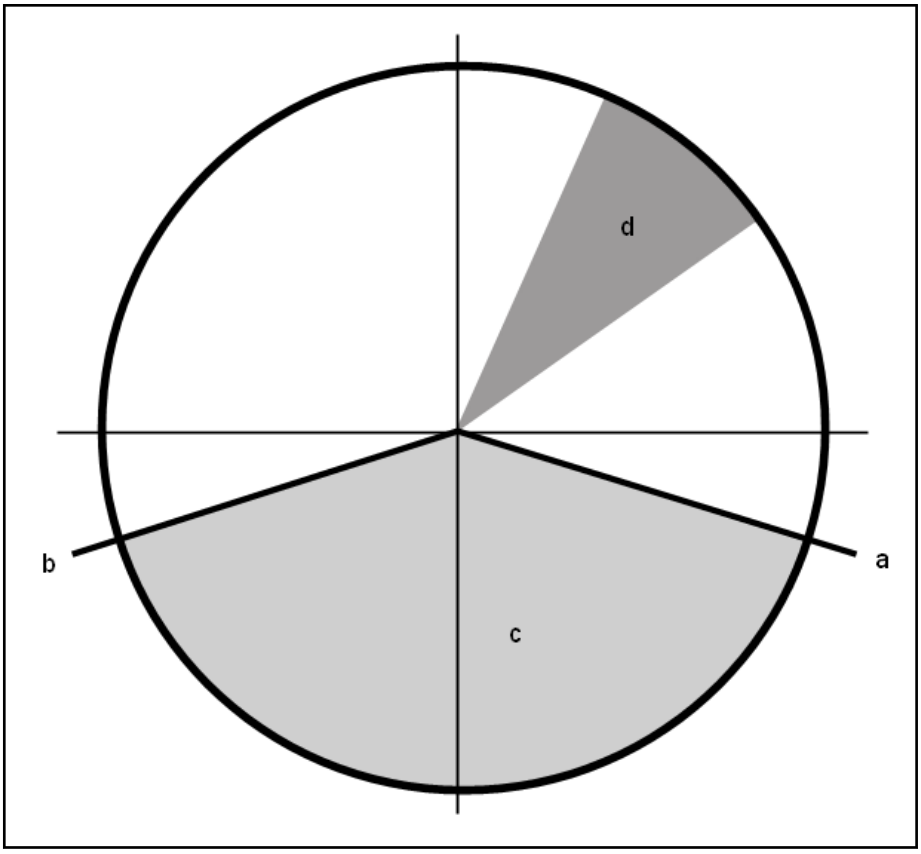
The "CorrectionMode" input sets the correction window either absolute or relative. If the "CorrectionMode" is set to CORR_WIN_DISABLED, the correction window is inactive.

The "CorrectionStart" input defines the start of the correction window.

The "CorrectionEnd" input defines the end of the correction window.

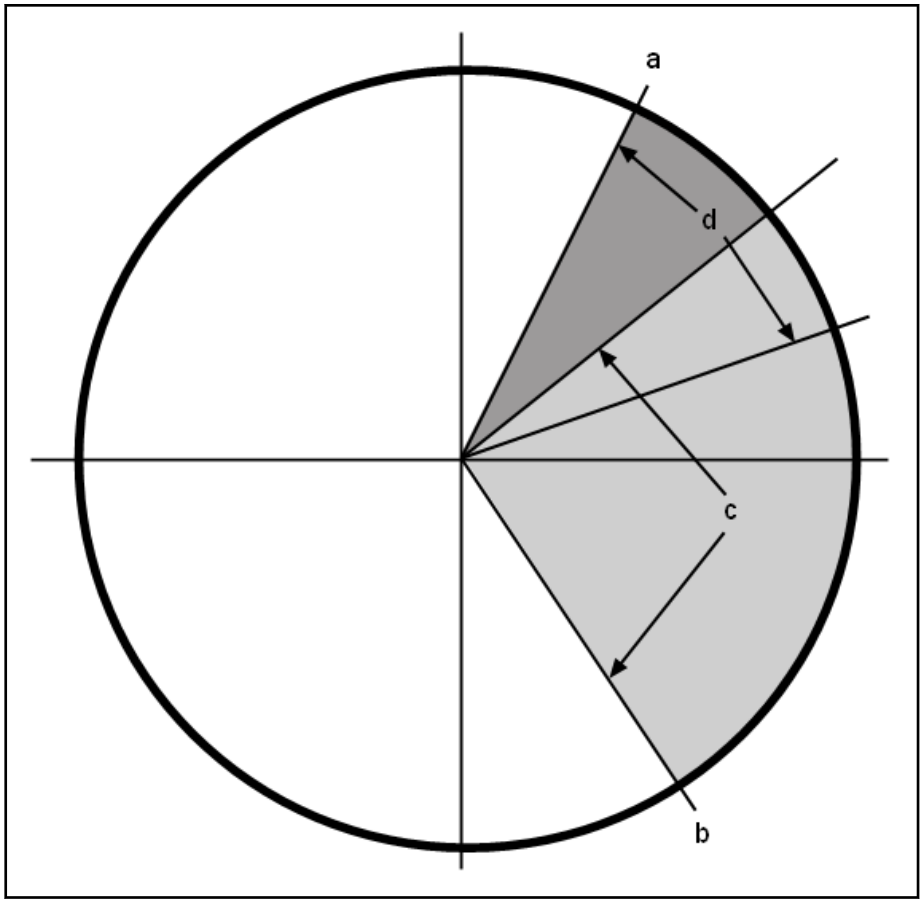
The correction window supports only a positive modulo range (e.g., 0...+360 degrees) of the measuring axis.

ML_TechRegi.library/MX_TechRegi.lib



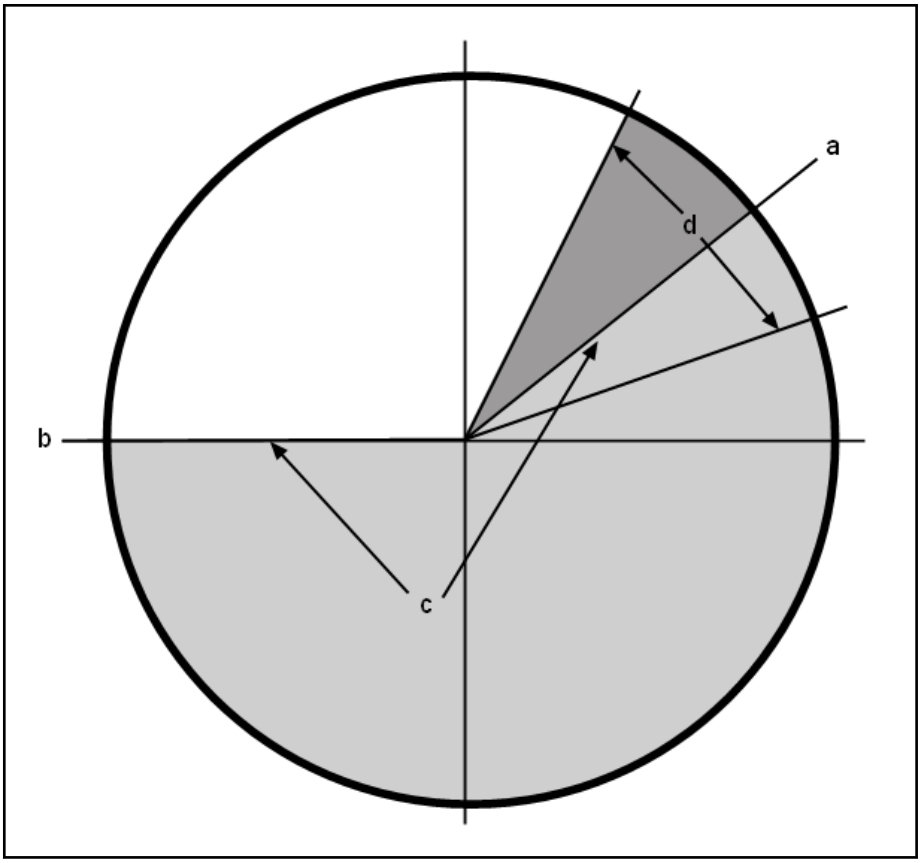
a: Start correction window
b: End correction window
c: Correction Window
d: Expectation window

Fig.7-19: Correction Window Absolute



a: Start correction window = measuring end
b: Relative end of correction window to measuring end
c: Correction Window
d: Expectation window
Fig.7-20: Correction Window Relative

ML_TechRegi.library/MX_TechRegi.lib



a: Start correction window = measuring end
b: End of correction window = perm. angle
c: Correction Window
d: Expectation window
Fig.7-21: Correction Window Relative Start Absolute End

Smoothing The "TControl" input is the attenuation constant for the "RecordedPosition" input. If set to 0, smoothing is inactive.

ProbeDistance In the simplest case, the measuring axis is a format cylinder (= controlled axis) and the distance of the sensor to the measuring (= controlled) axis is less than the format length. Measurement and correction relate to the same product. Many times it is necessary to mount the mark reader (sensor) further away from the measuring axis, e.g., hazardous areas in printing facilities. If the distance is greater than the format length, then the measured register error cannot be corrected until the product has reached the format cylinder. The correction must be delayed by as many formats as there are formats between the controlled axis and the sensor. The "ProbeDistance" input is the distance between the mark reader (sensor) and the controlled axis in format lengths (master axis revolutions, formats). If set to 0, the distance of the sensor to the measuring axis is less than the format length.

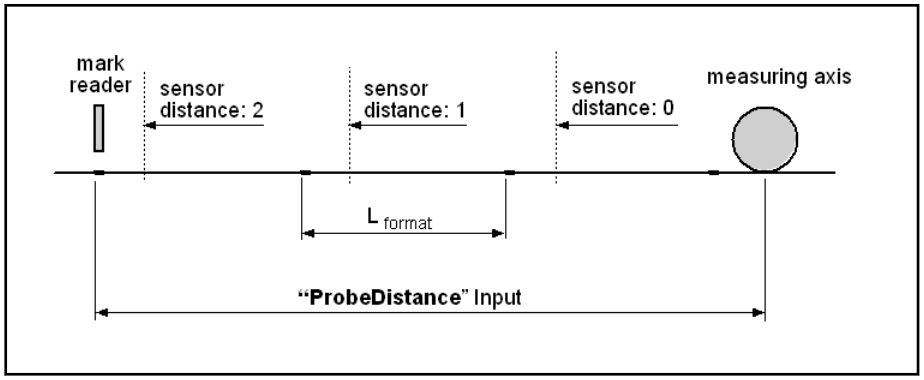


Fig.7-22: Probe Distance

Clearing the Internal Memory of Measured Values

The touch probe distance feature contains an internal memory that stores measured values (like a shift register). This internal memory can be cleared (set to "0") using the "ClearMeasValMem" input.

DeadBand The differential signal deviation from the filtered "RecordedPosition" input and the "Setpoint" input is scaled via the "ResponseCurve" input. A characteristic curve is used in order to pause the register controller within small register errors.

MarkGating This feature allows the user to control the registration only every N-th mark by gating invalid marks. This can be used e.g. if the marks on the web were printed with a cliche where the printed image appears several times. In this situation, the mark distances between consecutive marks are not constant. For example, the mark distances between every N-th consecutive mark is constant. The register controller then only uses every N-th mark for the closed loop register control.

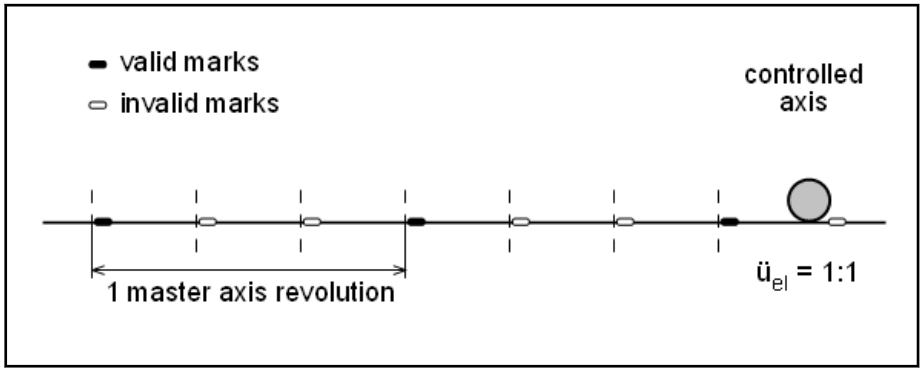


Fig.7-23: Mark Gating

Normal mode The RegisterController only calculates a new value if the "Step" input changes. The "PControl" and "IControl" controller parameters can be changed continuously.

If the value in the "ControlValue" output exceeds or falls below the values of outputs "HighLimit" or "LowLimit", then "ControlValue" is limited to one of the following limit values:

- If the value in "ControlValue" is positive, then the value in "HighLimit" is used.
- If the value in "ControlValue" is negative, then the value in "LowLimit" is used.

ML_TechRegi.library/MX_TechRegi.lib

If the register controller has an integral action, then an anti-reset wind-up is carried out and the integral action is no longer summed. In this case the "HighLimitActive" and "LowLimitActive" outputs are set to TRUE.

Required Parameterization

IndraMotion MLC parameterization

If a real axis is used as the controlled axis ("ControlledAxis" input), then the parameter used in the "ControlledValueIDN" input must be one of the following supported P-Parameters:

- P-0-0690, P-0-0691, P-0-0692, P-0-0694, and P-0-0695

Additionally, this P-Parameter must be configured in the MDT of the cyclic data channel in the drive.

If a virtual axis is used as the controlled axis, then the parameter used in the "ControlledValueIDN" input must be one of the following supported A-Parameters:

- A-0-2730, A-0-2615, A-0-2610, A-0-2600, A-0-2605, A-0-2620

IndraMotion MLD parameterization

The parameter used as controlled parameter for the "ControlledValueIDN" (P-Parameter) input must be configured in the "AxisData" structure. Thus, the axis data structure has to be activated. The following P-Parameters are supported:

- P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695

Error Handling

The function block outputs the error messages of the following function blocks used.

- MB_RegiMeasuringType03 (fig. 7-66 " Error codes of the MB_RegiMeasuringType03 function block" on page 337)
- IL_PIType02 (library RIL_LoopControl, refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries")
- MB_RegiRegulateType02 (fig. 7-81 " Error codes of the MB_RegiRegulateType02 function block" on page 349)
- MB_RegiSetpointLockType01 (fig. 7-104 " Error codes of the MB_RegiSetpointLockType01 function block" on page 367)

The function block generates the following error messages in Additional1/Additional2 using the table "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"Axis_Ref" is not within the valid range (measured)
INPUT_RANGE_ERROR(16#0006)	16#0813	16#0001	"HighLimit" < "LowLimit"
INPUT_RANGE_ERROR(16#0006)	16#0813	16#0002	"PControl" < 0

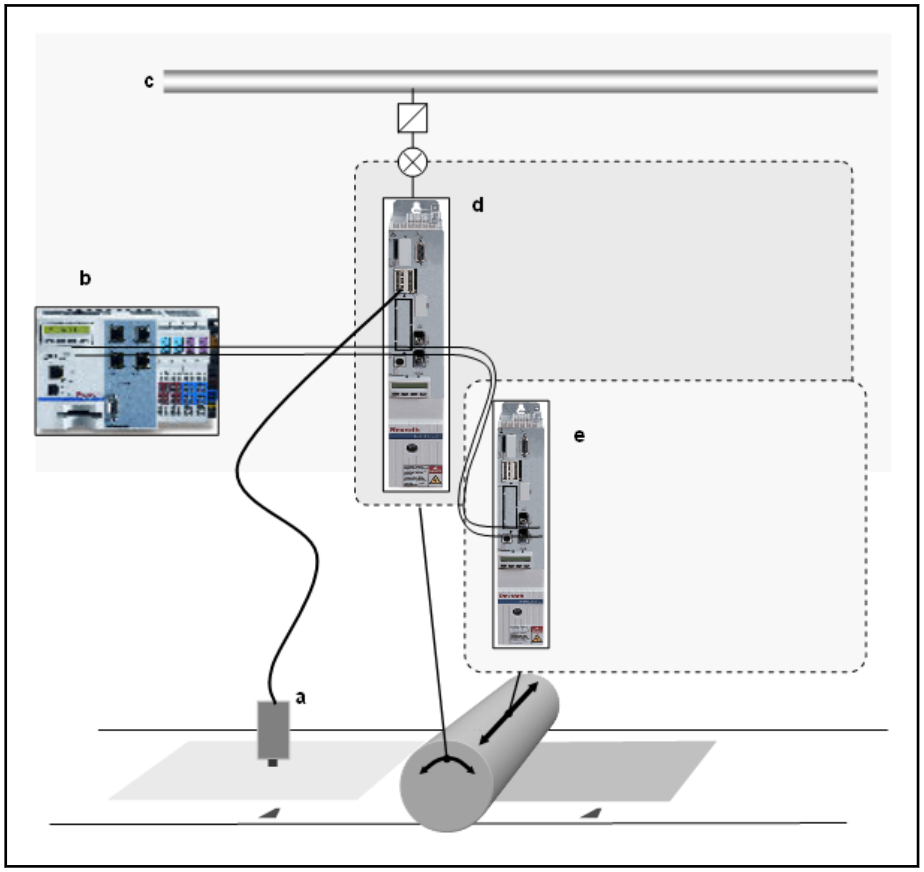
Fig.7-24: Error codes of the MB_RegisterControllerType04 function block

7.2.4 Register Controller - Configuration

Overview

The MB_RegisterControllerType04 function block is used for paper, printing, packaging and film applications, which use synchronized drives (e.g. Phase-, Cam-Synchronization) to feed material through a machine.

In these types of applications, variations in the characteristics of the material, material slippage and the production process affect the accuracy of the material position. A sensor determines the current position of registration marks, mounted or printed on the material and the register controller measures the deviation to the set point value. It calculates the required correction value using a control loop. Using this method, the registration marks stay precisely on their set point and do not drift. The following figure shows an example of a register controller implementation:



a	Sensor
b:	IndraMotion MLC Control
c	Master Axis (Virtual Axis)
d	Register Control, Probe Function, Phase Synchronization

Fig.7-25: Example Register Controller on an IndraMotion MLC-based System

Using the Register Control

Overview

The fastest way to use the register control functionality in an application is to combine the register controller function block with touch probe function block. The touch probe function block provides the actual value to the register controller. The register controller calculates the correction value based on the actual deviation between the set point value and the actual value.

The example in this section demonstrates the MB_RegisterControllerType04 register controller function block with the MB_TouchProbeContinuous function block.

Configuring the Touch Probe Function Block

The following touch probe signals must be configured using the MB_InitTouchProbe function block:

1. Drive touch probe 1
2. Positive touch probe 1
3. Marker failure enabled
4. Expected window enabled

For a description of the touch probe function blocks, please refer to chapter 5.7.3 "MB_InitTouchProbe" on page 98.

Configuring the Touch Probe and RegisterControllerType04 Function Blocks

The following figure shows how the MB_TouchProbeContinuous and MB_RegisterControllerType04 function blocks would be connected when triggering the touch probe function using a positive edge. As part of the MB_InitTouchProbe function block configuration, the marker failure and expected window are enabled. For this reason, the counter for marker failures must also be connected.

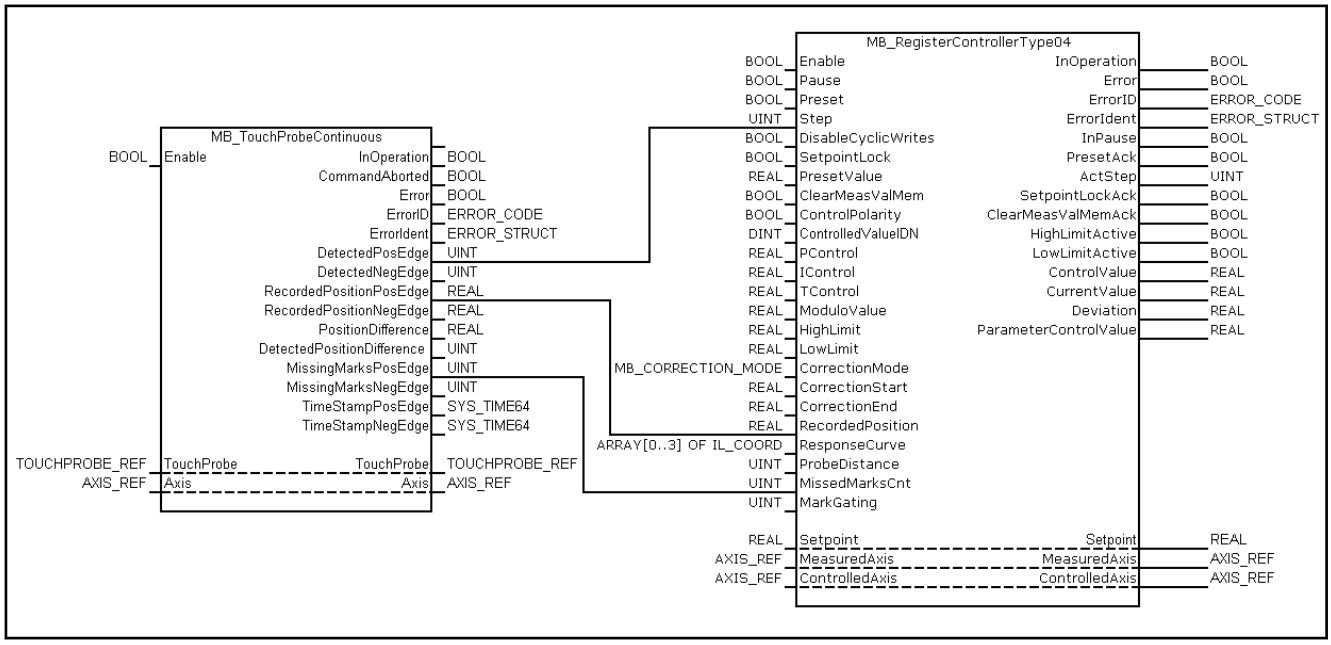


Fig.7-26: MB_TouchProbeContinuous and MB_RegisterControllerType04 Interface

Refer to the relevant function block descriptions for a complete description of the inputs and outputs.

The following signal connections are outlined in the figure above:

1. "DetectedPosEdge" to "Step"
2. "RecordedPositionPosEdge" to "RecordedPosition"
3. "MissingMarksPosEdge" to "MissedMarksCnt"

   This connection is only necessary if the expectation window and the counter for marker failures are used.

Parameterizing the Register Controller

Axis Scaling

In order to use the register control functionality, both axes (measured axis, controlled axis) must be set to rotary modulo format. The IndraWorks menu selection is as follows:

ML_TechRegi.library/MX_TechRegi.lib

- For IndraMotion MLD, select **IndraWorks ▸ Axis ▸ Scaling/Mechanical System ▸ Scaling/Units**.
- For IndraMotion MLC, select **IndraWorks ▸ Motion ▸ Virtual Axes/Real Axes ▸ Axis ▸ Scaling/Mechanical System ▸ Scaling/Units**.

Refer to the following figure for an example:

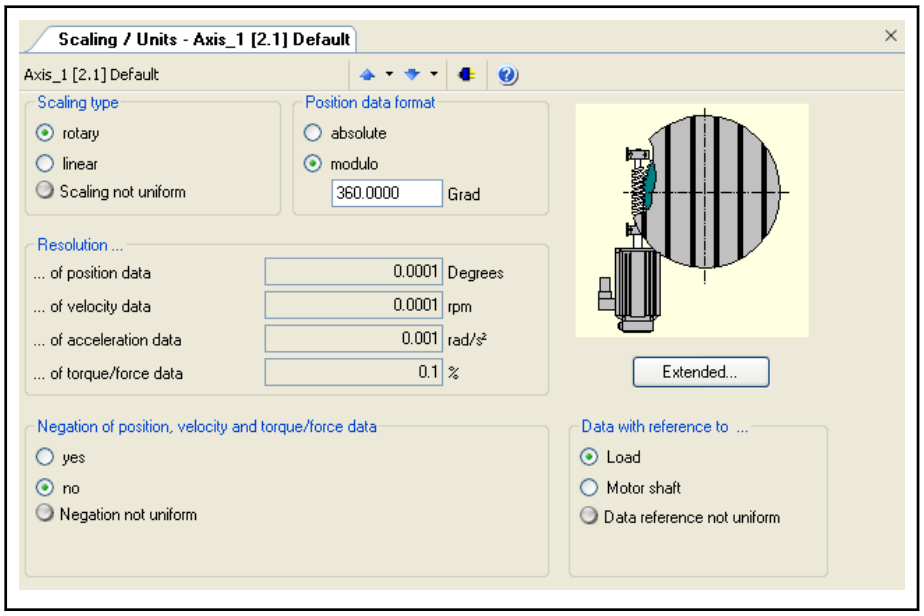


Fig.7-27: Setting the Axis Scaling

Configuring the IndraMotion MLC Master Data Telegram (MDT)

The parameter used for the "ControlledValueIDN" input must be configured in the MDT of the controlled axis. From the relevant IndraWorks project select **IndraWorks ▸ Motion** , right click over the controlled axis and select **Communication ▸ Cyclical SERCOS Channel**.

Refer to the following figure for an example:

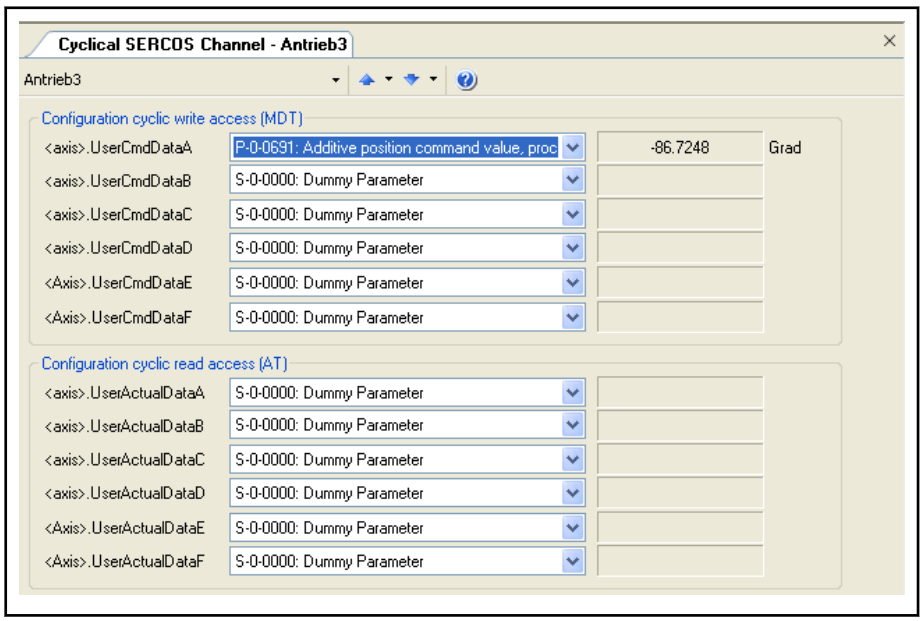


Fig.7-28: MDT configuration of the controlled axis (IndraMotion MLC)

IndraMotion MLD Axis Data Configuration

The parameter used for the "ControlledValueIDN" input must be configured in the AxisData structure of axis. From the relevant IndraWorks project select **IndraWorks ▸ MLD ▸ Configuration** and check the **AxisData structure support** option.

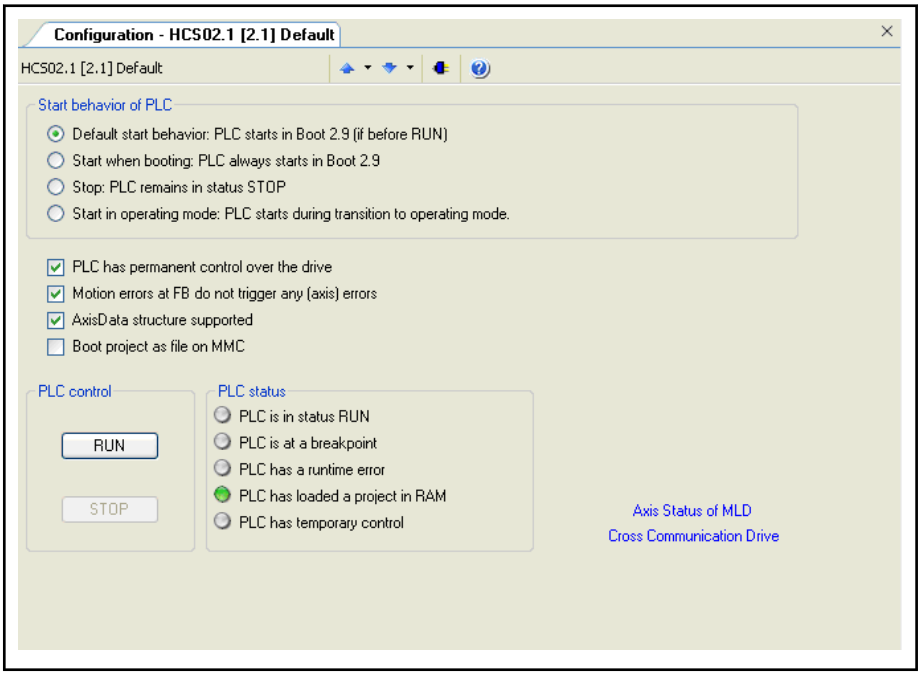


Fig.7-29: IndraMotion MLD configuration

Next, select **IndraWorks ▸ MLD ▸ Configuration ▸ AxisData** and select the parameter used for the "ControlledValueIDN" input.

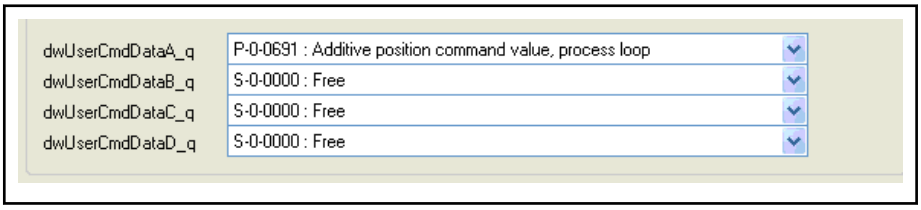


Fig.7-30: AxisData Configuration

7.2.5 Register Controller – Examples for Applications

Punching in Label Printing

The following example describes a punching process on a label printing machine:

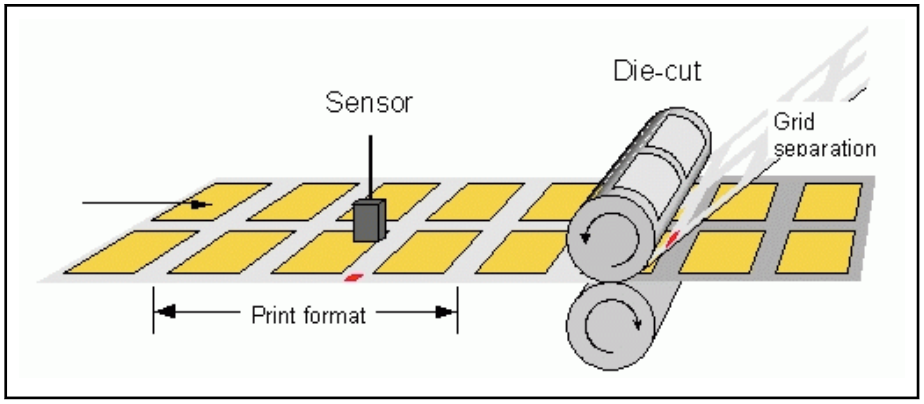


Fig.7-31: Closed loop register control of a punching device

Labels are printed on a composite material including a backing foil and a self-adhesive top layer. The unprinted grid is stamped and rewound.

The task of the register controller is to adjust the punching cylinder on the printed web.

The punching cylinder runs in phase-synchronized mode. The adjustment with regard to the printed web is accomplished by adjusting the angular offset. The sensor is connected with the drive of the punching cylinder. The angular position (the actual position value) of the punching cylinder is determined via the sensor signal.

The positioning command for the register controller corresponds to the cylinder position which is exactly aligned to the product. The correction value is calculated based on the difference between the reference value and the actual value and by adding this value to the current angular offset.

The most important settings for the touch probe, the register controller and the drive operation mode of this application are:

- Measured value (touch probe function) = Actual position value encoder 1 (S-0-0051)
- Controlled value (ControlledValueIDN) = additive position command value (P-0-0691)
- Drive operation mode = phase synchronization

Closed Loop Insetter Control

The following figure shows a closed loop insetter control in a material infeed application:

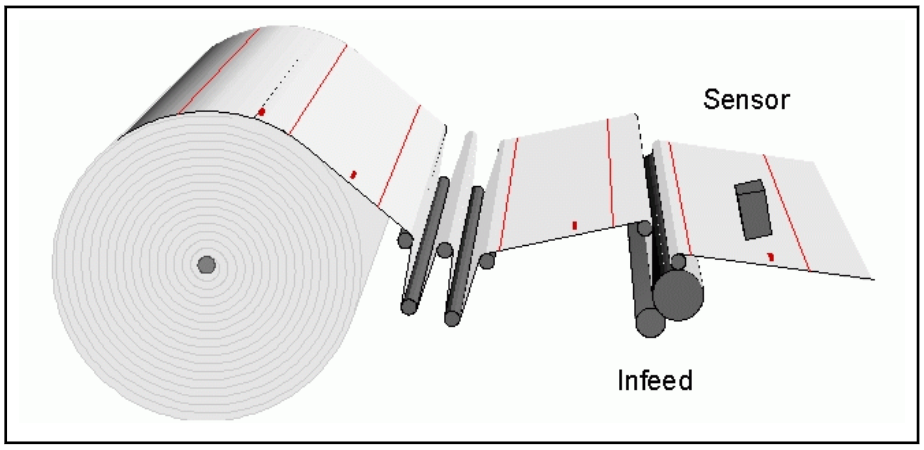


Fig.7-32: Closed loop insetter control of the material infeed

ML_TechRegi.library/MX_TechRegi.lib

The material infeed unit transports preprinted material to a printing machine. After the preprinting and drying process, the web has shrunk, i. e. the format length does not exactly correspond to the nominal format. The web has to be elongated again for further processing.

The task of the register controller is to control the velocity of the material infeed rollers so that the preprinted web is elongated to the nominal paper format. This implies that one paper format is transported during each revolution of the master axis.

The positioning command of the register controller is related to the master axis.

The material infeed roller runs in the velocity synchronization mode. The velocity of the infeed roller is controlled via the gear ratio fine adjustment. The elongation of the web is correct if the printed mark is always detected in the same master axis position by the mark reader. The master axis position is detected by the sensor signal. The mark reader is connected with the touch probe input of the material infeed roller drive.

The most important settings for the touch probe, the register controller and the drive operation mode of this application are:

- Measured value (touch probe function) = resulting master axis position (P-0-0775)
- Controlled value (ControlledValueIDN) = gear ratio fine adjustment (P-0-0694)
- Drive operation mode = speed synchronization

Flow Wrapper

The following figure shows a sideseal axis (foil infeed axis) of a flow wrapper application:

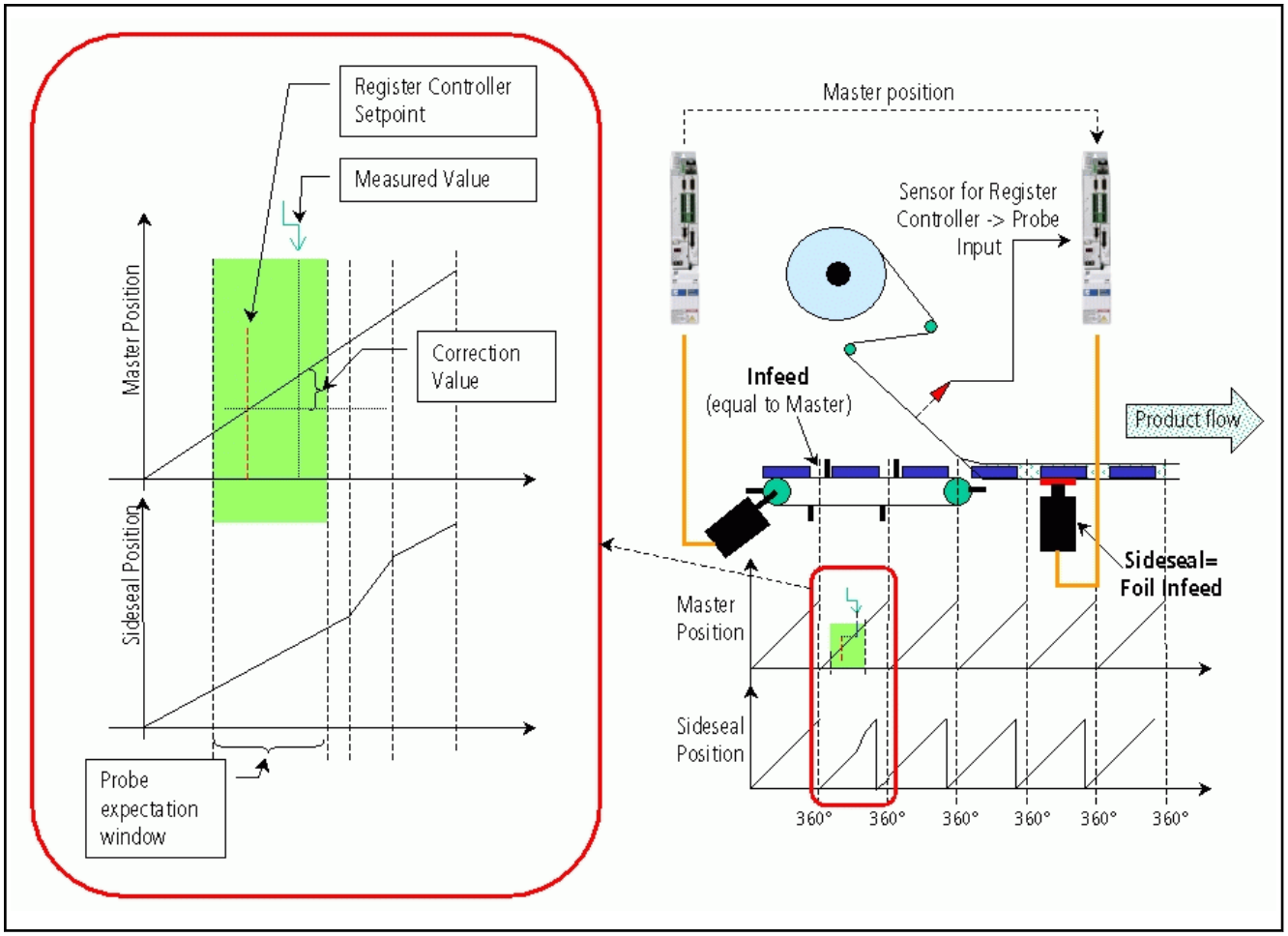


Fig.7-33: Foil infeed axis of a flow wrapper application

In this application, the product is fed into the packaging machine with the help of an infeed belt. The foil infeed axis (sideseal axis) runs synchronously with the infeed belt and pulls in the foil. During this process, the correct alignment of the foil to the product as well as an equal product length has to be ensured.

Slippage between the material and the infeed rollers as well as inaccuracies of the material can result in positioning errors between the servo motor and the foil. Thus, register marks are printed on the material allowing a correction of positioning errors.

Prints, pictures and dirt on the product can result in unwanted signals at the print mark sensor input. Thus, an expectation window for the printed mark signal is required. The drive touch probe function is only active within the expectation window to measure the printed mark position. Furthermore, it is required to detect missing marks within the expectation window as well.

The correction motion is proportional to the difference between the measured value and the command value. Due to mechanical limits of the machine, the correction motion has to be limited to a user-defined value.

The register controller has to adjust the foil infeed axis to the printed foil to adjust the foil to the product position.

The foil infeed axis runs in the phase synchronization mode. The register controller calculates an additive position command value for the foil infeed axis by which the foil is correctly readjusted.

The most important settings for the touch probe, the register controller and the drive operation mode of this application are:

- Measured value (touch probe function) = resulting master axis position (P-0-0775)
- Controlled value (ControlledValueIDN) = additive position command value (P-0-0691)
- Drive operation mode = phase synchronization

7.3 Side Register Controller Function Block

7.3.1 Overview

This section describes the side register controller function block that is part of the ML(X)_TechRegi library.

7.3.2 MB_RegisterControllerSideType01

Brief Description

The MB_RegisterControllerSideType01 register controller function block supports the following functionality:

- Calculate a correction value based on both measured and setpoint values using a P- or PI-control loop
- Preset feature
- Pause feature
- Min./max limiting of the calculated control value
- Setpoint lock feature
- Dead band K_P characteristic curve
- PT1 element

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-34: Reference table of the MB_RegisterControllerSideType01

ML_TechRegi.library/MX_TechRegi.lib

Interface Description

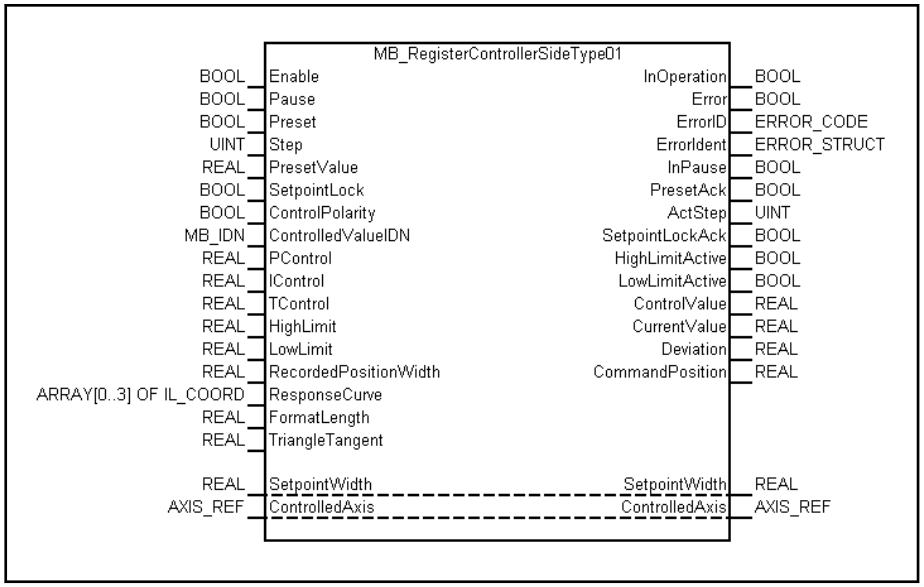


Fig.7-35: MB_RegisterControllerSideType01 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	SetpointWidth	REAL	Setpoint width value (in technical units)
	ControlledAxis	AXIS_REF	Reference to the controlled axis
VAR_INPUT	Enable	BOOL	The function block is enabled as long as the input is TRUE
	Pause	BOOL	The "Pause" input is only evaluated while the "Enable" input is active. The "ControlValue" and "CommandPosition" outputs are constant while "Pause" is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces the control loop to "PresetValue"
	Step	UINT	Processing step
	PresetValue	REAL	Value passed to the "ControlValue" output on the rising edge of the "Preset" input
	SetpointLock	BOOL	Enables set point lock. If TRUE, the next measured mark will be latched
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE
	ControlledValueIDN	MB_IDN	The following control IDN is supported: S-0-0258
	PControl	REAL	Proportional gain of the PI-controller. If set to 0, the proportional part of the controller is disabled and the proportional gain is set internally to 1 in order to work as an I-controller
	IControl	REAL	Integral action time T_n of the PI-controller. If set to 0, the integral part of the PI-controller is disabled
	TControl	REAL	Filter constant for the input "RecordedPositionWidth"
	HighLimit	REAL	Maximum value for "ControlValue"
	LowLimit	REAL	Minimum value for "ControlValue"

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	RecordedPosition-Width	REAL	Width of measured mark (in technical units)
	ResponseCurve	ARRAY[0...3] OF IL_COORD	Characteristic curve
	FormatLength	REAL	Length of the format in mm
	TriangleTangent	REAL	Value of tan α = (opposite leg length / adjacent leg length)
VAR_OUTPUT	InOperation	BOOL	Side register controller is in operation
	Error	BOOL	It indicates an error Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "ControlValue" and "Command" outputs maintain their values
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ControlValue" output
	ActStep	UINT	Step Value, counting up with every new value at the output "ControlValue". Note: The process controller is paused and output "ActStep" is still counting with every new value on input "Step".
	SetpointLockAck	BOOL	Set point is locked. The current value in "RecordedPositionWidth" is copied to "SetpointWidth"
	HighLimitActive	BOOL	High limit is active. "ControlValue" would be higher, but is limited to "HighLimit"
	LowLimitActive	BOOL	Low limit is active. ControlValue would be lower, but is limited to "LowLimit"
	ControlValue	REAL	Control value calculated by the register controller
	CurrentValue	REAL	Recorded position which is currently used in the register controller side. Note: Transformed "RecordedPositionWidth" value.
	Deviation	REAL	Difference Set point side (transformed "Setpoint-Width") - "CurrentValue" of the side register controller (mm)
	CommandPosition	REAL	Calculated command position for the drive

Fig.7-36: Interface variables of the MB_RegisterControllerSideType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous

ML_TechRegi.library/MX_TechRegi.lib

Name	Type	Min. value	Max. value	Default value	Effective
Preset	BOOL			FALSE	"Enable" TRUE (continuous)
Step	UINT	0	65335	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
SetpointLock	BOOL	n.def.	n.def.	FALSE	Continuous
ControlPolarity	BOOL			FALSE	Continuous
ControlledValueIDN	MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
PControl	REAL	0.0	n.def.	1.0	Continuous
IControl	REAL	0.0	n.def.	0.0	Continuous
TControl	REAL	0	n.def.	0.0	Continuous
HighLimit	REAL	LowLimit	n.def.	0.0	Continuous
LowLimit	REAL	n.def.	HighLimit	0.0	Continuous
RecordedPositionWidth	REAL	n.def.	n.def.	0.0	Change of "Step"
ResponseCurve	ARRAY[0...3] OF IL_COORD	n.def.	n.def.	0.0	Continuous
FormatLength	REAL	0.0	n.def.	0.0	Rising edge at "Enable"
TriangleTangent	REAL	0.0	n.def.	1.0	Rising edge at "Enable"

Fig.7-37: Min./max. and default values of the inputs

Time diagram

The following diagram shows the signal-time diagram of the MB_RegisterControllerType04 function block including the "Pause" and "Preset" functions.

ML_TechRegi.library/MX_TechRegi.lib

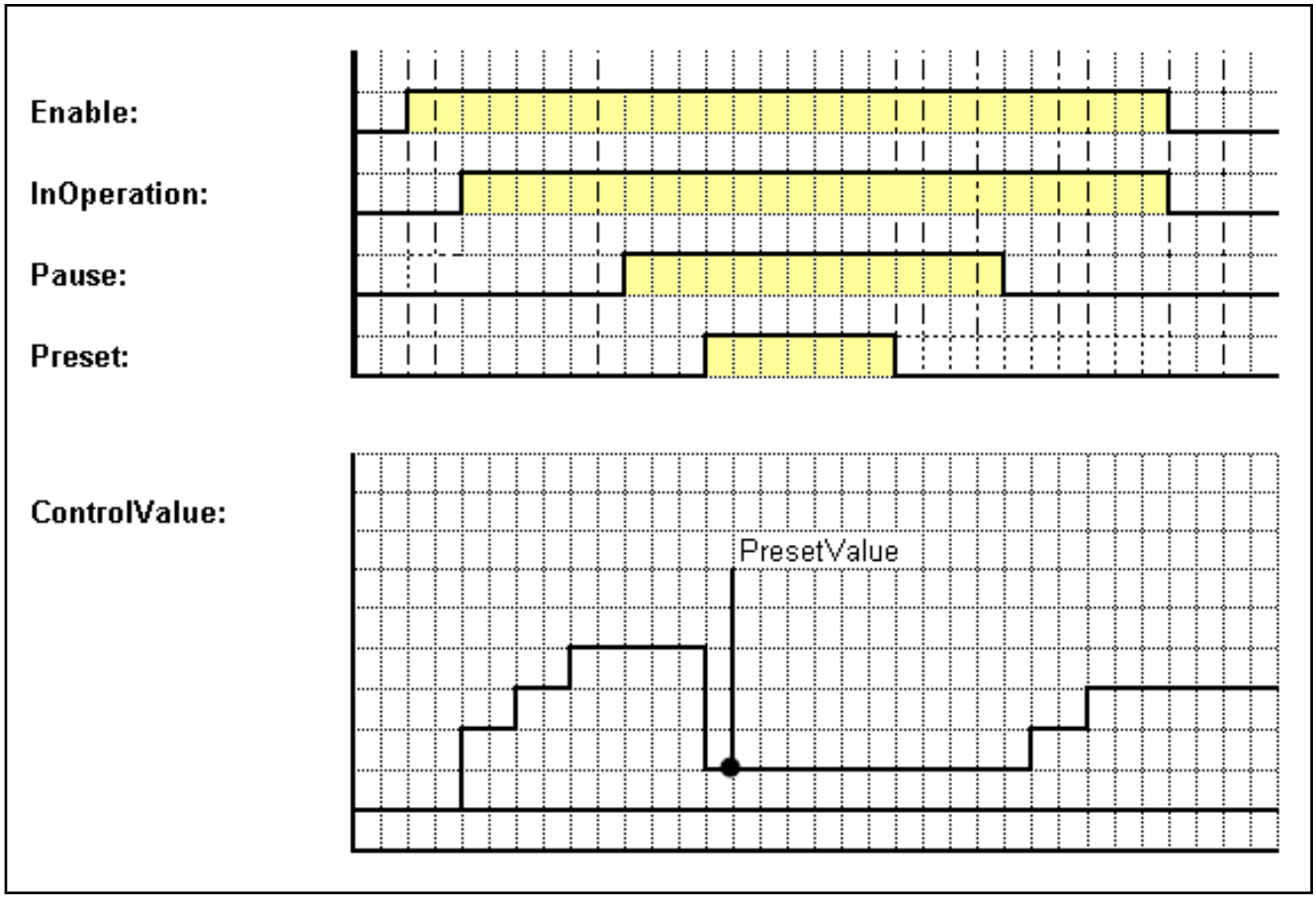


Fig.7-38: Time Diagram of the MB_RegisterControllerSideType01 with the Functions "Pause" and "Preset"

Functional Description

Initialization During initialization, the "SetpointWidth" and "RecordedPositionWidth" inputs must have valid data.

PresetValue For a trouble-free start of the controller outputs "ControlValue" and "CommandPosition", set the "Preset" input before activating the "Enable" input. During this period, the "SetpointWidth" and "RecordedPositionWidth" inputs must have valid data. In addition, the upper limits and lower limits are monitored to ensure a limitation of the "ControlValue" output if the current value for "PresetValue" is outside the limits of "HighLimit" or "LowLimit".

The "PresetAck" output is active if the value in the "PresetValue" input is copied to the "ControlValue" output. The "Preset" input has priority over the "Pause" input. The value in "PresetValue" is copied to the respective output on a rising edge of the "Preset" input if "Enable" is active. If successful, the "PresetAck" output is set to TRUE.

Pause The "Pause" input is only evaluated as long as the "Enable" input is active. In this case, the deviation ("SetpointWidth" –" RecordedPositionWidth") is set to 0. Therefore, the outputs "ControlValue" and "CommandPosition" do not change anymore. This can be used to maintain the values of the "ControlValue" and "CommandPosition" outputs. The "ActStep" output is updated even if the "Pause" input is active. As long as the "Pause" input is active, the "InPause" output is TRUE.

Smoothing The "TControl" input is the attenuation constant for the "RecordedPositionWidth" input. If set to 0, smoothing is inactive.

DeadBand The differential signal deviation from the filtered"RecordedPositionWidth" input and the "SetpointWidth" input is scaled via the "ResponseCurve" input. A characteristic curve is used in order to pause the register controller within small register errors.

FormatLength The length of the format (master axis revolutions, formats) in mm.

TriangleTangent The mark length between opposite leg and adjacent leg (tan α).

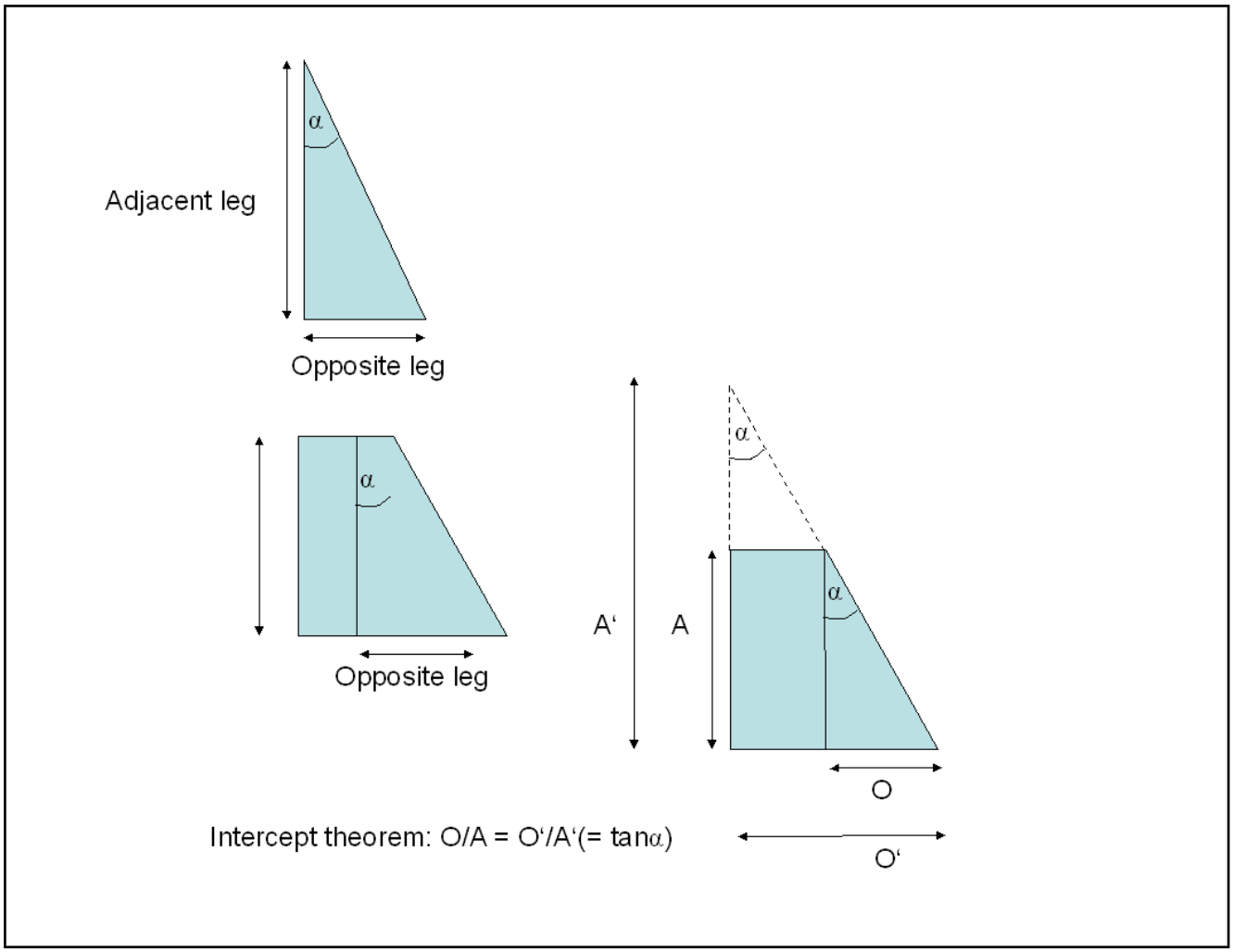


Fig.7-39: Right triangle mark and conical frustum mark

Normal mode The register controller side only calculates a new value if the "Step" input changes. The "PControl" and "IControl" and "TControl" controller parameters can be changed continuously.

If the value in the "ControlValue" output exceeds or falls below the values of outputs "HighLimit" or "LowLimit", then "ControlValue" is limited to one of the following limit values:

- If the value in "ControlValue" is positive, then the value in "HighLimit" is used
- If the value in "ControlValue" is negative, then the value in "LowLimit" is used

If the register controller side has an integral action, then an anti-reset wind-up is carried out and the integral action is no longer summed. In this case the "HighLimitActive" and "LowLimitActive" outputs are set to TRUE.

ML_TechRegi.library/MX_TechRegi.lib

Error Handling

The function block outputs the error messages of the following function blocks used

- MB_RegiMeasuringType01 (fig. 7-51 " Error codes of the MB_RegiMeasuringType01 function block" on page 330)
- IL_PIType02 (library RIL_LoopControl, refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries")
- MB_RegiRegulateType01 (fig. 7-72 " Error codes of the MB_RegiRegulateType01 function block" on page 341)
- MB_RegiSetpointLockType01 (fig. 7-104 " Error codes of the MB_RegiSetpointLockType01 function block" on page 367)

The function block generates the following error messages in Additional1/Additional2 using the table "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	AxisNo of "ControlledAxis" is not within the valid range
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Axis has no power "ControlledAxis"
RESOURCE_ERROR (16#0003)	16#0009	16#0000	"Axis_Ref" has changed "ControlledAxis"
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0001	Selected "ControlledValueIDN" is not supported
INPUT_RANGE_ERROR(16#0006)	16#0813	16#0001	"HighLimit" < "LowLimit"
INPUT_RANGE_ERROR(16#0006)	16#0813	16#0002	"PControl" < 0
INPUT_RANGE_ERROR(16#0006)	16#0816	16#0001	Input "SetpointWidth" < 0
INPUT_RANGE_ERROR(16#0006)	16#0816	16#0002	Input "RecordedPositionWidth" < 0
INPUT_RANGE_ERROR(16#0006)	16#0816	16#0003	Input "TriangleTangent" is not in valid range
INPUT_RANGE_ERROR(16#0006)	16#0816	16#0004	Input "FormatLength" is not in valid range
INPUT_INVALID_ERROR(16#0001)	16#0C01	16#0001	"LowLimit" > "HighLimit"
INPUT_INVALID_ERROR(16#0001)	16#0C01	16#0002	Selection of controller type is not supported. Check Inputs "IControl" and "PControl"
INPUT_RANGE_ERROR(16#0006)	16#0C02	16#0001	"TControl" input is invalid
INPUT_RANGE_ERROR(16#0006)	16#0C02	16#0003	"IControl" input is invalid
INPUT_RANGE_ERROR(16#0006)	16#0C02	16#0004	"PControl" input is invalid
INPUT_INVALID_ERROR(16#0001)	16#0C05	16#0001	X-value in "ResponseCurve" not in ascending order
INPUT_INVALID_ERROR(16#0001)	16#0C05	16#0002	More than one value for one X-Position in "ResponseCurve"

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR(16#0001)	16#0C05	16#0003	Negative X-Value in "Response-Curve"
ACCESS_ERROR (16#0004)	16#0815	16#0008	The controlled axis is an IndraMotion MLD-M slave axis, but Parameter P-0-1367, Bit 6 is not TRUE

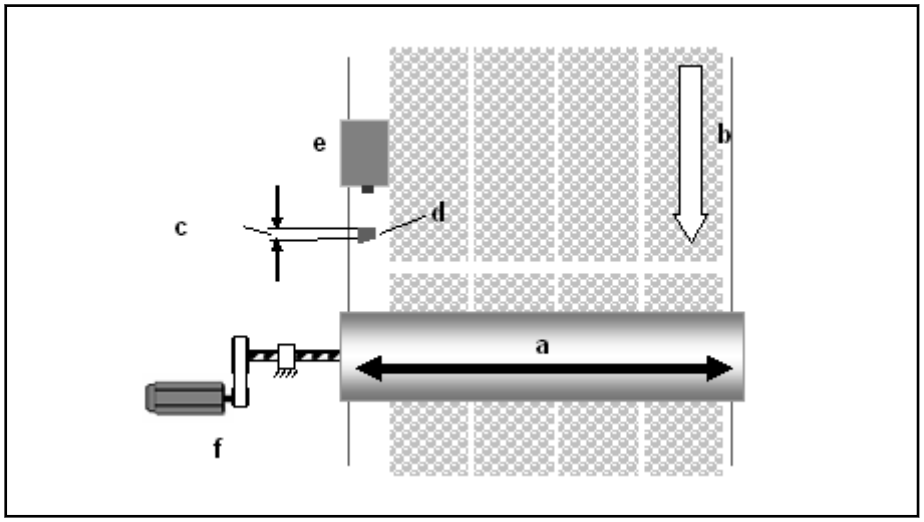
Fig.7-40: Error codes of MB_RegisterControllerSideType01 function block

7.3.3 Configuring the Side Register

Overview

The side register is a function used for the axial adjustment of rotating cylinders. A typical example is the adjustment of a printing cylinder for a pre-printed web.

The triangle mark is the axial reference on the printed image. A optical sensor registers the incoming and outgoing edge of the mark. The difference between these two edges is the width of the mark. This value is the actual value of the side register. The difference between the actual width of the mark and setpoint is converted into the target position of the servo motor.



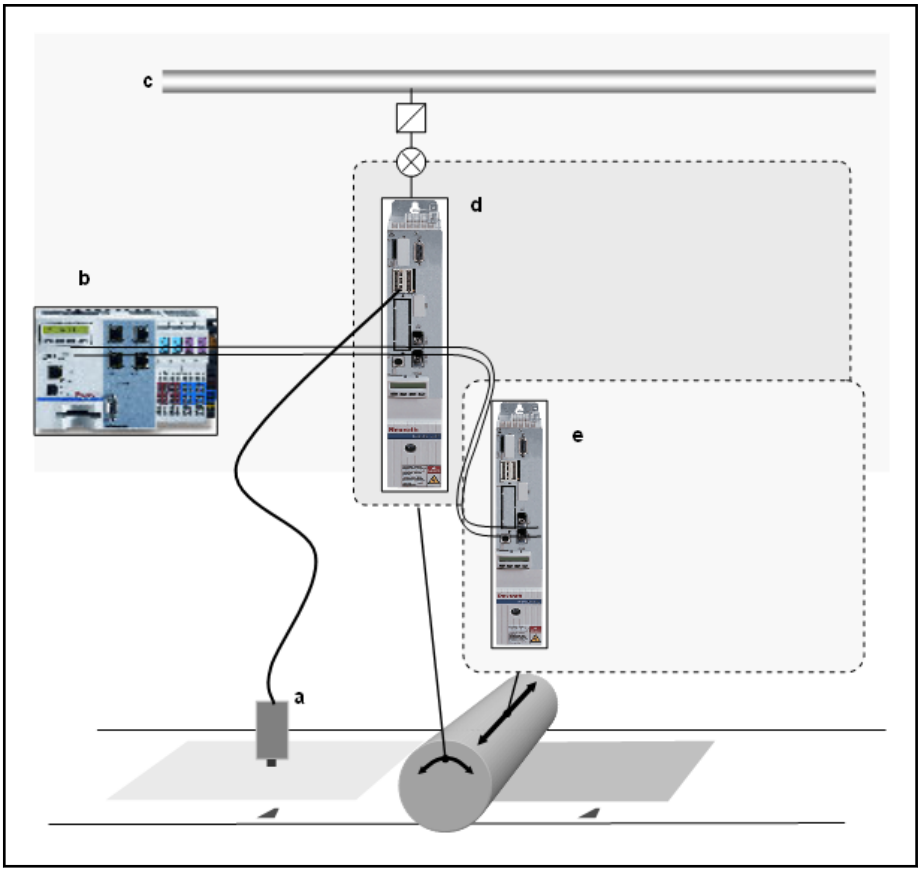
a: Side register
b: Web direction
c: Measured value
d: Registration mark
e: Sensor
f: Servo motor

Fig.7-41: Side Register Configuration

Using Side Register Control

The following picture shows a possible configuration of a side register controller for an IndraMotion MLC system.

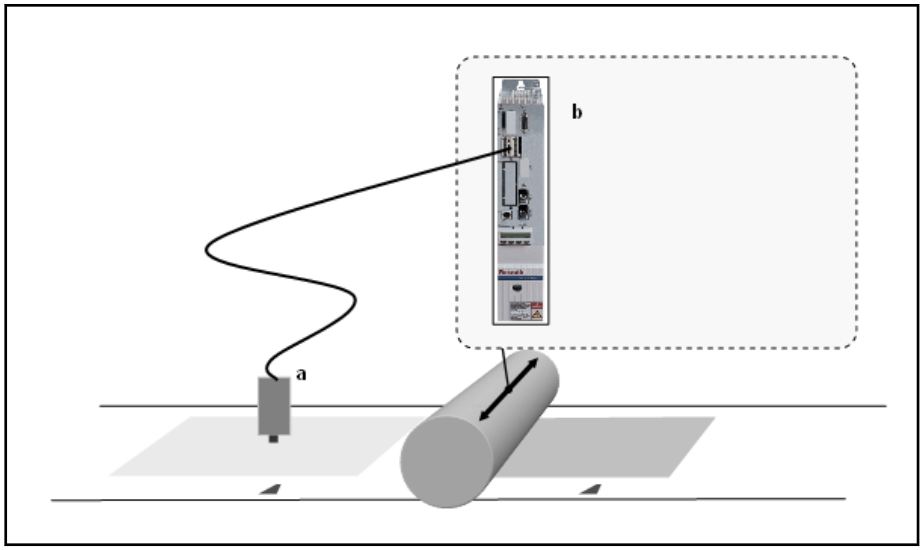
ML_TechRegi.library/MX_TechRegi.lib



a: Sensor
b: IndraMotion MLC
c: Master axis
d: Register controller, touch probe function, phase synchronization
e: Side register controller, absolute positioning

Fig.7-42: Register controller and side register controller overview on the IndraMotion MLC

The following figure shows the configuration of a side register controller for an IndraMotion MLD system.



a: Sensor
b: IndraMotion MLD: probe function, side register controller, absolute positioning

Fig.7-43: Side register controller overview on IndraMotion MLD

The axial axis must be an absolute linear axis.

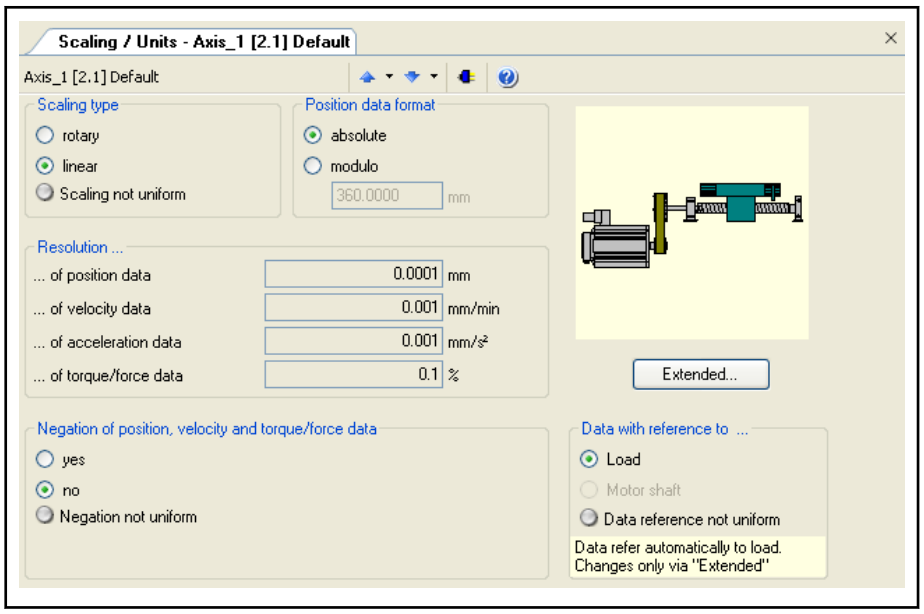


Fig.7-44: Scaling of the Controlled Axis of the Side Register Controller

Example configuration

The fastest way to use the side register controller functionality in an application is to combine the touch probe function blocks and the AxisInterface. The touch probe function block provides the actual value to side register. The side register controller calculates the target position for the axis. The side register has no internal functionality to write the position to a position command parameter of the axis. Therefore, the AxisInterface is used to write the command position to respective parameter of the axis.

This example demonstrates the MB_RegisterControllerSideType01 side register controller with the MB_TouchProbeContinuous function block and the AxisInterface function blocks.

ML_TechRegi.library/MX_TechRegi.lib

1. Output "PositionDifference" to input the "RecordedPositionWidth"
2. Output "DetectedPositionDifference" to input "Step"

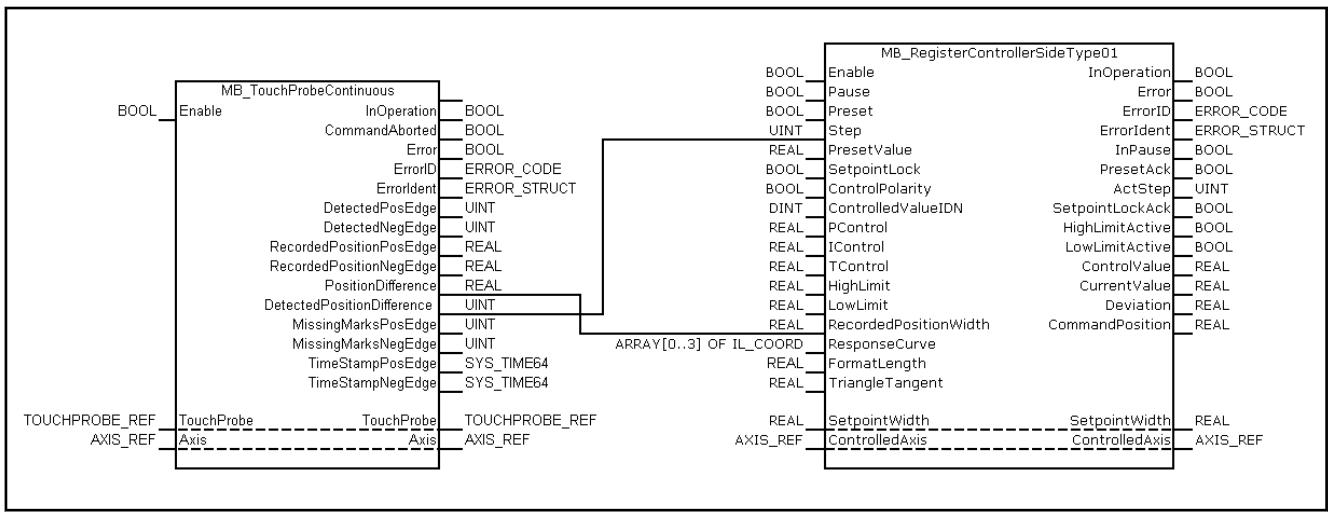


Fig.7-45: Schematic representation of the function blocks MB_TouchProbeContinuous and MB_RegisterControllerSideType01

3. Assign "CommandPosition" to the array arAxisCtrl[ControlledAxis].PosMode.Position

---

AxisInterface Settings

- Operating Mode : Absolute positioning
- The following command activates the operating mode

  arAxisCtrl[ ].Admin._OpMode := ModePosAbs; or arAxisCtrl[].Admin.MODE_POS_ABS := TRUE;
- EnableCyclicScanning must be activated

For more information, refer to the manual "Rexroth IndraMotion xxVRS Generic Application Template" in the chapter "ML_TechInterface.library".

xx - Version

Program:

```
fbMB_RegisterControllerSideType01(
    Enable:= bEnableRegi,
    Pause:= bPauseRegi,
    Preset:= bPresetRegi,
    Step:= uiDetectedPositionDifference,
    PresetValue:= bPresetValueRegi,
    SetpointLock:= bSetpointLock,
    ControlPolarity:= bControlPolarity,
    ControlledValueIDN:= diControlledValueIDN,
    PControl:= rPControl,
    IControl:= rIControl,
    TControl:= rTControl,
```

```
    HighLimit:= rHighLimit,
    LowLimit:= rLowLimit,
    RecordedPositionWidth:= rPositionDifference,
    ResponseCurve:= stResponseCurve,
    FormatLength:= rForamtLength,
    TriangleTangent:= rTriangleTangent,
    SetpointWidth:= rSetpointWidth,
    ControlledAxis:= Drive5,
    InOperation=> bInOperationRegi,
    Error=> bErrorRegi,
    ErrorID=> stErrorIDRegi,
    ErrorIdent=> stErrorIdentRegi,
    InPause=> bInPauseRegi,
    PresetAck=> bPresetAckRegi,
    ActStep=> uiActStepRegi,
    SetpointLockAck=> bSetpointLockAck,
    HighLimitActive=> bHighLimitActive,
    LowLimitActive=> bLowLimitActive,
    ControlValue=> rControlValue,
    CurrentValue=> rCurrentValue,
    Deviation=> rDeviation,
    CommandPosition=>CommandPosition );

fbMB_AxisInitType01(
    Execute:= bExecuteAxisInit,
    ReadModuloValue:= FALSE,
    EnableCyclicScanning:= TRUE,
    AdminCtrl:= arAxisCtrl[Drive5.AxisNo].Admin,
    PosModeCtrl:= arAxisCtrl[Drive5.AxisNo].PosMode,
    VelModeCtrl:= arAxisCtrl[Drive5.AxisNo].VelMode,
    SyncModeCtrl:= arAxisCtrl[Drive5.AxisNo].SyncMode,
    AxisDataPtrStatus:= arAxisStatus[Drive5.AxisNo].PtrAxisData,
    AdminStatus:= arAxisStatus[Drive5.AxisNo].Admin,
    DiagStatus:= arAxisStatus[Drive5.AxisNo].Diag,
    Axis:= Drive5,
    Done=> bDoneInit,
    Active=> bActiveInit,
    Error=> bErrorInit,
    ErrorID=> stErrorIDInit,
    ErrorIdent=> stErrorIdent);
(* Position command to operate the side register control *)
```

ML_TechRegi.library/MX_TechRegi.lib

```
arAxisCtrl[Drive5.AxisNo].PosMode.Position := CommandPosition;
fbMB_AxisInterfaceType01(
    AdminCtrl:= arAxisCtrl[Drive5.AxisNo].Admin,
    PosModeCtrl:= arAxisCtrl[Drive5.AxisNo].PosMode,
    VelModeCtrl:= arAxisCtrl[Drive5.AxisNo].VelMode,
    SyncModeCtrl:= arAxisCtrl[Drive5.AxisNo].SyncMode,
    AdminStatus:= arAxisStatus[Drive5.AxisNo].Admin,
    DiagStatus:= arAxisStatus[Drive5.AxisNo].Diag,
    PosStatus:= arAxisStatus[Drive5.AxisNo].PosMode,
    VelStatus:= arAxisStatus[Drive5.AxisNo].VelMode,
    SyncModeStatus:= arAxisStatus[Drive5.AxisNo].SyncMode );
```

7.4 Register Control, Measuring and Regulating Function Blocks

7.4.1 Overview

This section describes the measuring, regulating and locking function blocks which can be used to build custom made register controller function blocks that are part of the ML(X)_TechRegi library.

7.4.2 MB_RegiMeasuringType01

Brief Description

The MB_RegiMeasuringType01 function block is used to process the measured touch probe signal.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-46: Reference table of the MB_RegiMeasuringType01

Interface Description

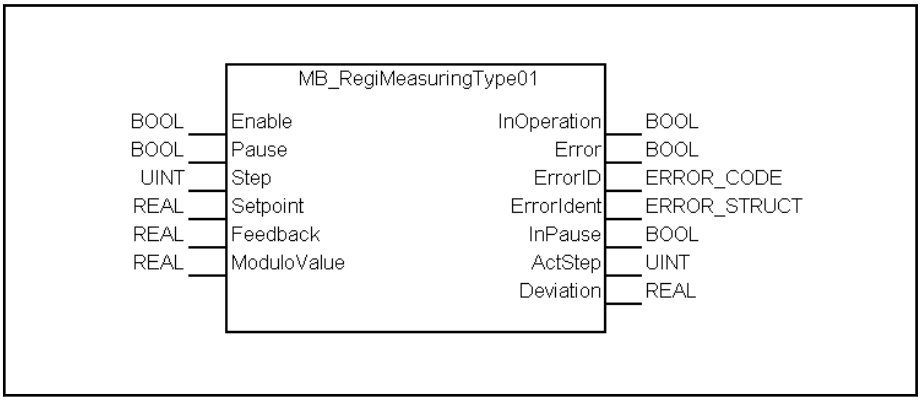


Fig.7-47: MB_RegiMeasuringType01 Function Block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active

I/O type	Name	Data type	Description
	Step	UINT	New values are calculated when the "Step" input changes
	Setpoint	REAL	Command value
	Feedback	REAL	Measured value
	ModuloValue	REAL	Modulo value from measured axis
VAR_OUTPUT	InOperation	BOOL	Measuring function block is in operation
	Error	BOOL	It indicates an error occurred during function block operation
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "Deviation" output maintains its value
	ActStep	UINT	Step Value, counting up with every new value at the output "Deviation"
	Deviation	REAL	"Setpoint" - "Feedback"

Fig.7-48: Interface variables of the MB_RegiMeasuringType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
Setpoint	REAL	0	ModuloValue	0.0	Change of "Step"
Feedback	REAL	0	ModuloValue	0.0	Change of "Step"
ModuloValue	REAL	0	n.def.	0.0	Rising edge at "Enable"

Fig.7-49: Min./max. and default values of the inputs

Time diagram

Time diagram according to the PLCopen specification.

ML_TechRegi.library/MX_TechRegi.lib

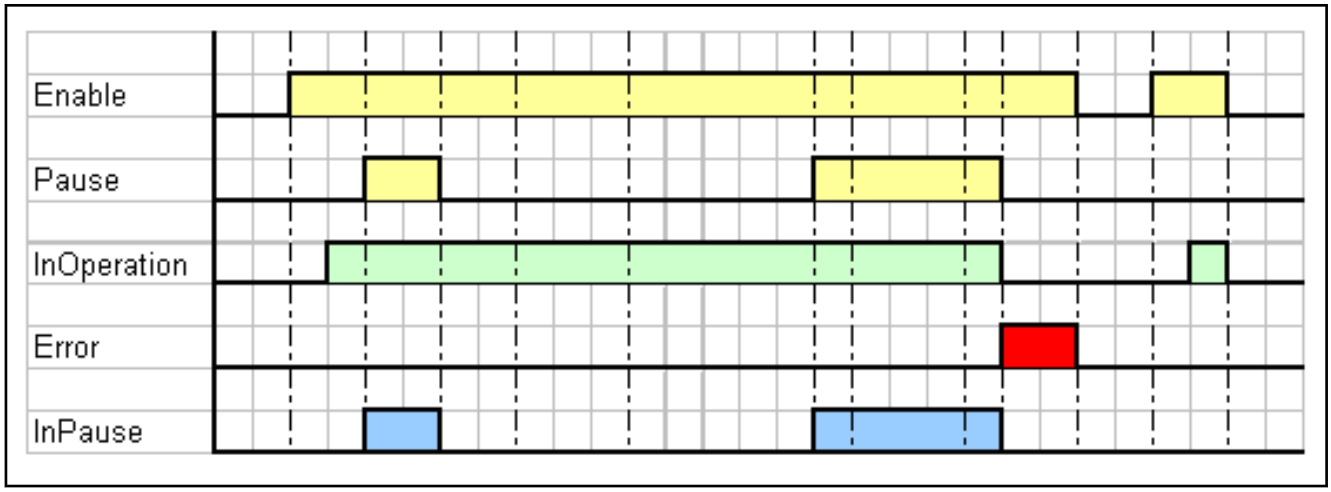


Fig.7-50: Time Diagram of MB_RegiMeasuringType01

Functional Description The MB_RegiMeasuringType01 function block subtracts the "Setpoint" and "Feedback" inputs ("Setpoint" - "Feedback"). The result of the subtraction is the value of the "Deviation" output.

Pause The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. While "Pause" is active, the "InPause" output is true and the "Deviation" output maintains its value.

Error Handling The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected Axis ("Axis_Ref") was changed while function block is in operation
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0002	Input "Setpoint" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0003	Input "Setpoint" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0004	Input "Feedback" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0005	Input "Feedback" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0006	Input "ModuloValue" < 0

Fig.7-51: Error codes of the MB_RegiMeasuringType01 function block

7.4.3 MB_RegiMeasuringType02

Brief Description The MB_RegiMeasuringType02 function block is used to process the measured touch probe signal.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-52: Reference table of the MB_RegiMeasuringType02

ML_TechRegi.library/MX_TechRegi.lib

Interface Description

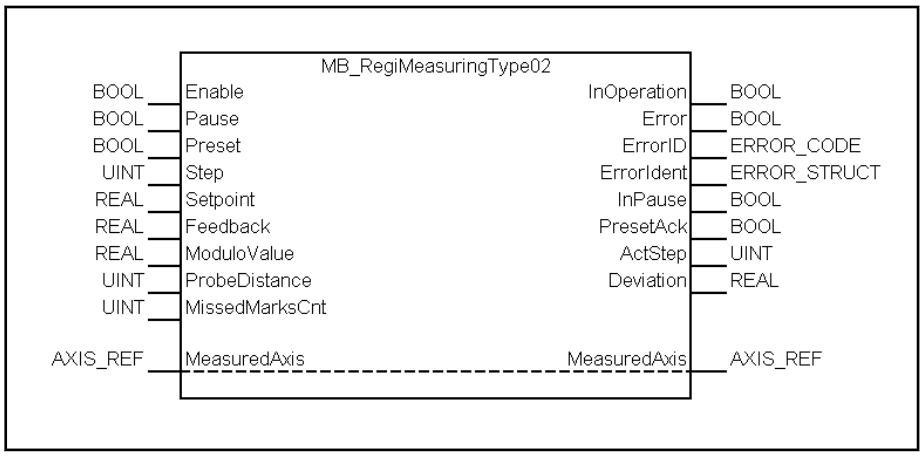


Fig.7-53: MB_RegiMeasuringType02 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	MeasuredAxis	AXIS_REF	Reference to the measured axis
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces the shift register memory to "0".
	Step	UINT	New values are calculated when the "Step" input changes
	Setpoint	REAL	Command value
	Feedback	REAL	Measured value
	ModuloValue	REAL	Modulo value from measured axis
	ProbeDistance	UINT	The distance between mark sensor and measuring (= controlled) axis in master axis revolutions (format lengths)
	MissedMarksCnt	UINT	Counter for marker failures if an expectation window is used
VAR_OUTPUT	InOperation	BOOL	Measuring function block is in operation
	Error	BOOL	It indicates an error occurred during function block operation
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "Deviation" output maintains its value
	PresetAck	BOOL	Preset function is done

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	ActStep	UINT	Step Value, counting with every new value on the output "Deviation" up
	Deviation	REAL	"Setpoint" - "Feedback" delayed if shift register is active

Fig.7-54: Interface variables of the MB_RegiMeasuringType02 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	"Enable" TRUE (continuous)
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
Setpoint	REAL	0	ModuloValue	0.0	Change of "Step"
Feedback	REAL	0	ModuloValue	0.0	Change of "Step"
ModuloValue	REAL	0	n.def.	0.0	Rising edge at "Enable"
ProbeDistance	UINT	0	30	0	Continuous
MissedMarksCnt	UINT	0	65.535	0	Continuous

Fig.7-55: Min./max. and default values of the inputs

Time diagram

Time diagram according to the PLCopen specification.

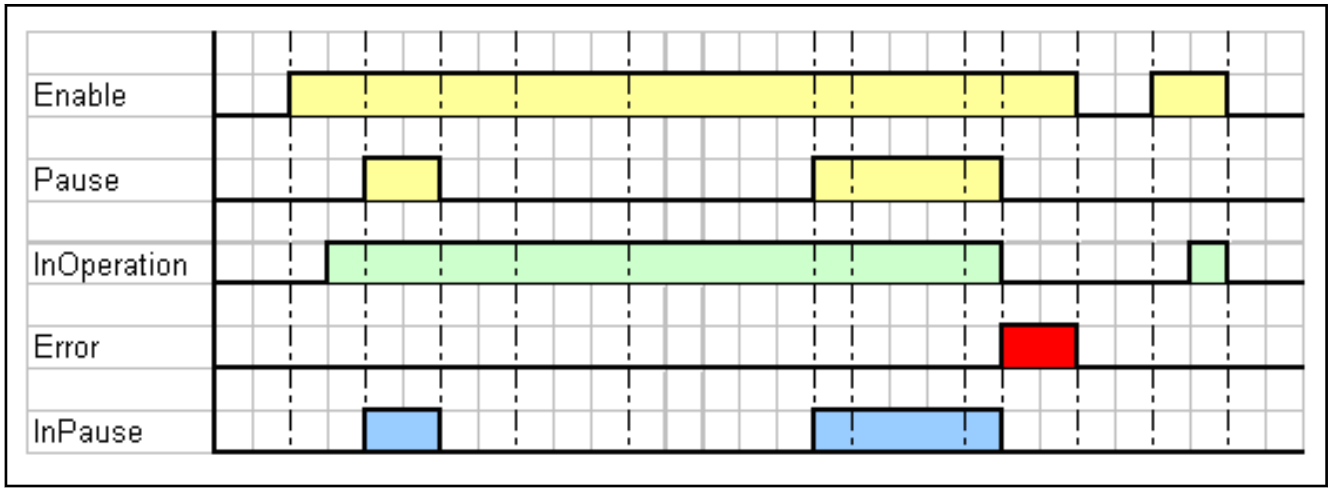


Fig.7-56: Time Diagram of MB_RegiMeasuringType02

Functional Description

The MB_RegiMeasuringType02 function block subtracts the "Setpoint" and "Feedback" inputs ("Setpoint" - "Feedback"). The result of the subtraction is the value of the "Deviation" output when the shift register is deactivated. If the shift register is activated, it is delayed by the number of "ProbeDistance".

Pause The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. While "Pause" is active, the "InPause" output is true and the "Deviation" output maintains its value.

Preset The shift register can be pre written with the "Preset" input. In order to pre write the shift register memory, "Enable" must be active. A zero value is written to the memory on a rising edge of the "Preset" input if "Enable" is active. If successful, the "PresetAck" output is set to TRUE.

ProbeDistance The "ProbeDistance" input is the distance between the mark sensor and the measuring (= controlled) axis, in master axis revolutions (format lengths). The maximum number of format lengths is 30. If "ProbeDistance" is 0, the shifting register is deactivated. The "Deviation" output is set to "Setpoint" "Feedback".

If "ProbeDistance" is not 0, then the shift register is activated. This means that the values at the "Deviation" output are delayed by the number of master axis revolutions (format lengths) on input "ProbeDistance".

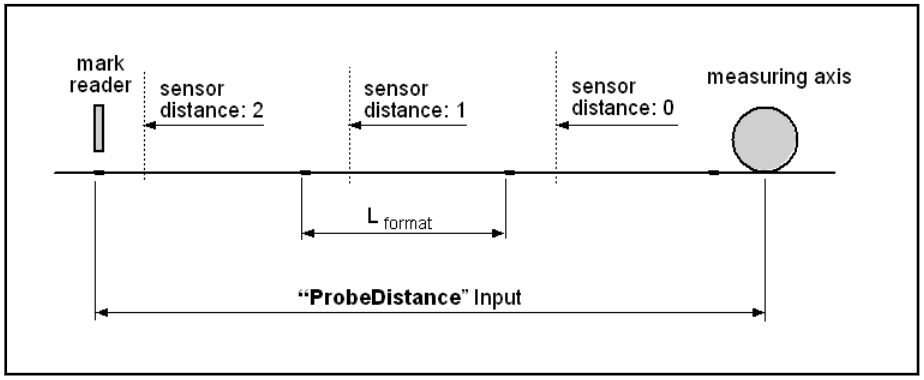


Fig.7-57: Setting "ProbeDistance" Input

MissedMarkCnt This is the marker failure monitoring counter from the touch probe function when the expectation window is enabled. If a mark is missed, a zero is written to the shift register if activated. In the event of a deactivated shifting register, the zero is written to the "Deviation" output.

Error Handling The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected Axis ("Axis_Ref") was changed while function block is in operation
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0001	Input "ProbeDistance" is not within the valid range (0 <= "ProbeDistance"<= 30)
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0002	Input "Setpoint" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0003	Input "Setpoint" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0004	Input "Feedback" > "ModuloValue"

ML_TechRegi.library/MX_TechRegi.lib

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0005	Input "Feedback" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0006	Input "ModuloValue" < 0

Fig.7-58: Error codes of the MB_RegiMeasuringType02 function block

7.4.4 MB_RegiMeasuringType03

Brief Description

The MB_RegiMeasuringType03 function block is used to process the measured touch probe signal.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-59: Reference table of the MB_RegiMeasuringType03

Interface Description

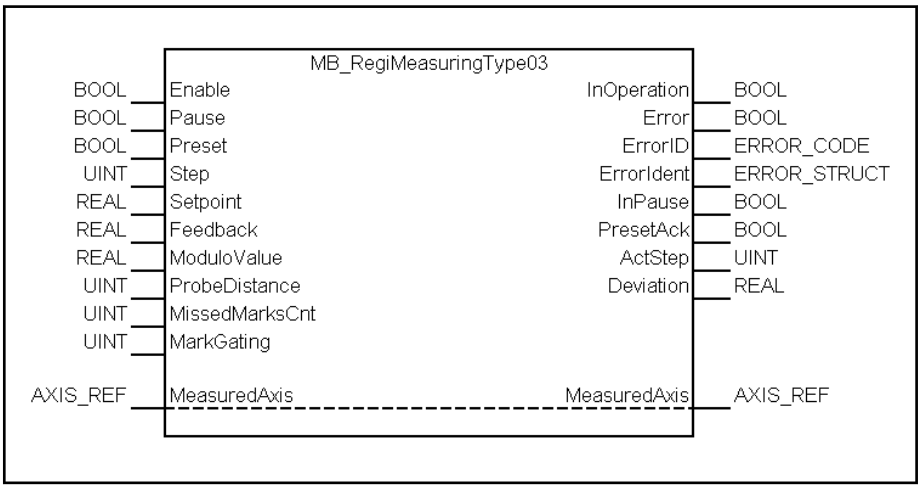


Fig.7-60: MB_RegiMeasuringType03 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	MeasuredAxis	AXIS_REF	Reference to the measured axis
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces the shift register memory to "0".
	Step	UINT	New values are calculated when the "Step" input changes
	Setpoint	REAL	Command value
	Feedback	REAL	Measured value
	ModuloValue	REAL	Modulo value from measured axis
	ProbeDistance	UINT	The distance between mark sensor and measuring axis in master axis revolutions (format lengths)

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	MissedMarksCnt	UINT	Missed mark counter if using a expectation window
	MarkGating	UINT	Every n-th mark is used for registration
VAR_OUTPUT	InOperation	BOOL	Measuring function block is in operation
	Error	BOOL	It indicates an error occurred during function block operation
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "Deviation" output maintains its value
	PresetAck	BOOL	Preset function is done
	ActStep	UINT	Step Value, counting up with every new value at the output "Deviation"
	Deviation	REAL	"Setpoint" - "Feedback" delayed if shift register is active

Fig.7-61: Interface variables of the MB_RegiMeasuringType03 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	"Enable" TRUE (continuous)
Step	UINT	0	65.535	0	Continuous Starts at rising edge of "Enable" and falling edge of "Pause"
Setpoint	REAL	0	ModuloValue	0.0	Change of "Step"
Feedback	REAL	0	ModuloValue	0.0	Continuous
ModuloValue	REAL	0	n.def.	0.0	Rising edge at "Enable"
ProbeDistance	UINT	0	60	0	Continuous
MissedMarksCnt	UINT	0	65.535	0	Continuous
MarkGating	UNIT	0	65.535	1	Rising edge at "Enable". A value of "0" disables mark gating (identical to a value of "1")

Fig.7-62: Min./max. and default values of the inputs

Time diagram

Time diagram according to PLCopen specification.

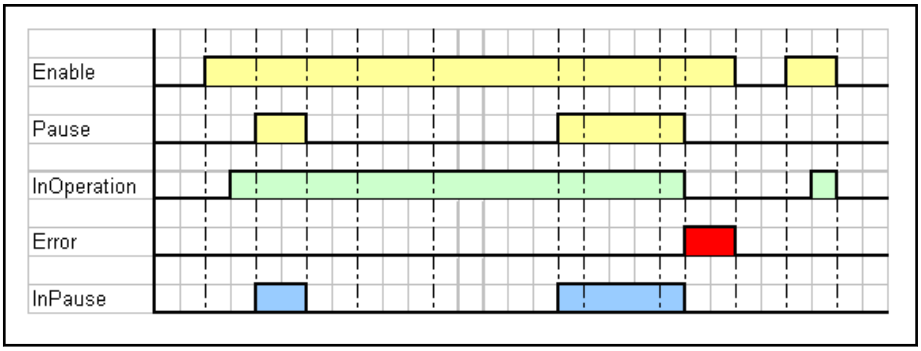


Fig.7-63: Time Diagram of MB_RegiMeasuringType03

Functional Description

The MB_RegiMeasuringType03 function block subtracts the "Setpoint" and "Feedback" inputs ("Setpoint" - "Feedback"). The result of the subtraction is the value of the "Deviation" output when the shift register is deactivated. If the shift register is activated, it is delayed by the number of "ProbeDistance". Also, only every n-th mark will can used for registration with the mark gating feature of the function block.

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. While "Pause" is active, the "InPause" output is true and the "Deviation" output maintains its value.

Preset

The shift register can be pre written with the "Preset" input. In order to pre write the shift register memory, "Enable" must be active. A zero value is written to the memory on a rising edge of the "Preset" input if "Enable" is active. If successful, the "PresetAck" output is set to TRUE.

ProbeDistance

The "ProbeDistance" input is the distance between the mark sensor and the measuring (= controlled) axis, in master axis revolutions (format lengths). The maximum number of format lengths is 60. If "ProbeDistance" is 0, the shift register is deactivated. The "Deviation" output is set to "Setpoint" – "Feedback".

If "ProbeDistance" is not 0, then the shift register is activated. This means that the values at the "Deviation" output are delayed by the number of master axis revolutions (format lengths) on input "ProbeDistance".

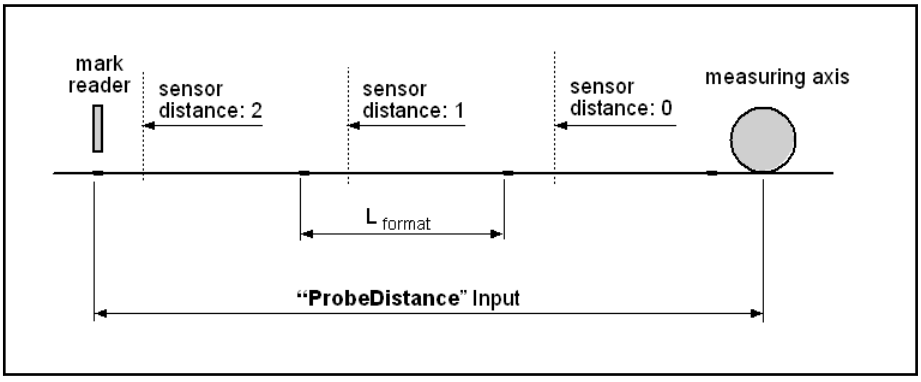


Fig.7-64: Setting "ProbeDistance" Input

MissedMarkCnt

This is the marker failure monitoring counter from the touch probe function when the expectation window is enabled. If a mark is missed, a zero is written to the shift register if activated. In the event of a deactivated shifting register, the zero is written to the "Deviation" output.

MarkGating

This feature allows the user to control the registration only every N-th mark by gating invalid marks. This can be used e.g. if the marks on the web were printed with a printing plate where the printed image appears several times.

In this situation, the mark distances between consecutive marks are not constant. For example, the mark distances between every N-th consecutive mark is constant. The RegiMeasuring function block only uses every N-th mark for the register control.

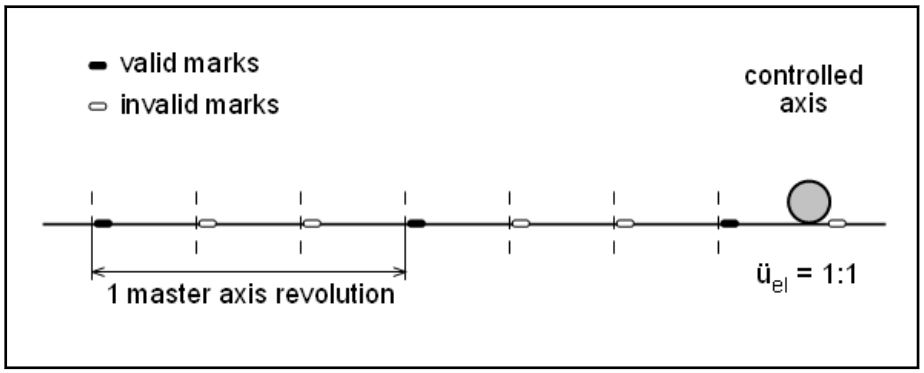


Fig.7-65: Mark gating

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected Axis ("Axis_Ref") was changed while function block is in operation
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0001	Input "ProbeDistance" is not within the valid range (0 <= "ProbeDistance" <= 30)
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0002	Input "Setpoint" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0003	Input "Setpoint" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0004	Input "Feedback" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0005	Input "Feedback" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0006	Input "ModuloValue" < 0
INPUT_RANGE_ERROR(16#0006)	16#0814	16#0007	Input "MarkGating" < 0

Fig.7-66: Error codes of the MB_RegiMeasuringType03 function block

7.4.5 MB_RegiRegulateType01

Brief Description

The MB_RegiRegulateType01 function block is used to write a control value to the drive.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-67: Reference table of the MB_RegiRegulateType01

Interface Description

ML_TechRegi.library/MX_TechRegi.lib

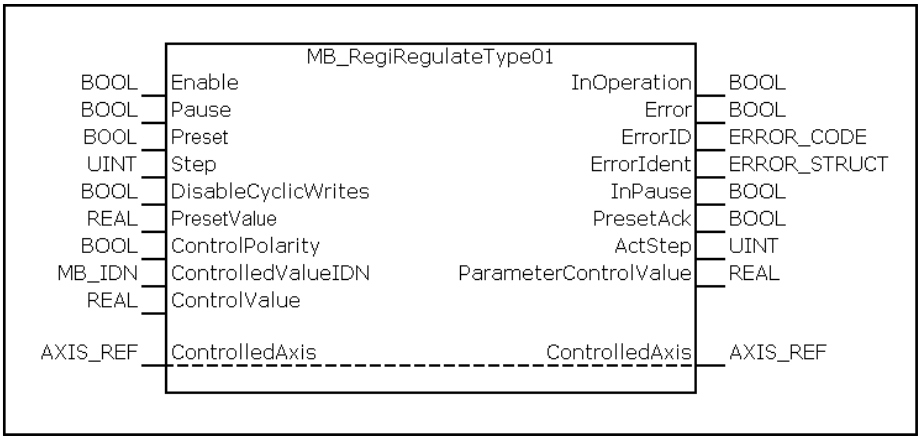


Fig.7-68: MB_RegiRegulateType01 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	ControlledAxis	AXIS_REF	Reference to the controlled axis
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces "ParameterControlValue" to "PresetValue"
	Step	UINT	Processing step
	DisableCyclicWrites	BOOL	Disables cyclic writing to the selected parameter in "ControlledValueIDN" when TRUE
	PresetValue	REAL	Value passed to the "ParameterControlValue" output on the rising edge of the "Preset" input
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE
	ControlledValueIDN	MB_IDN	SERCOS IDN of the controlled parameter (e.g. P-0-0691). The following IDNs are supported: **IndraMotion MLD**: P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695 **IndraMotion MLC**: A-0-2730, P-0-0690, A-0-2615, P-0-0691, A-0-2610, P-0-0692, A-0-2600, P-0-0694, A-0-2605, P-0-0695, A-0-2620
	ControlValue	REAL	Calculated control value from, for example, IL_PIType02
VAR_OUTPUT	InOperation	BOOL	Function block is in operation
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "ParameterControlValue" output maintains its value

I/O type	Name	Data type	Description
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ParameterControlValue" output
	ActStep	UINT	Step Value, counting with every new value on the output "ParameterControlValue" up
	ParameterControlValue	REAL	The register controller function block calculates a value based on the selected parameter in input "ControlledValueIDN" and stores the result in output "ParameterControlValue". This value is written to the selected parameter input "ControlledValueIDN" when input "DisableCyclicWrites" = FALSE

Fig.7-69: Interface variables of the MB_RegiRegulateType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
ControlPolarity	BOOL			FALSE	Continuous
ControlledValueIDN	MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
ControlValue	REAL	n.def.	n.def.	0.0	Continuous
ModuloValue	REAL	0.0	n.def.	0.0	Rising edge at "Enable"

Fig.7-70: Overview of min./max. values and default values of the inputs

Time diagram

Time diagram according to the PLCopen specification.

ML_TechRegi.library/MX_TechRegi.lib

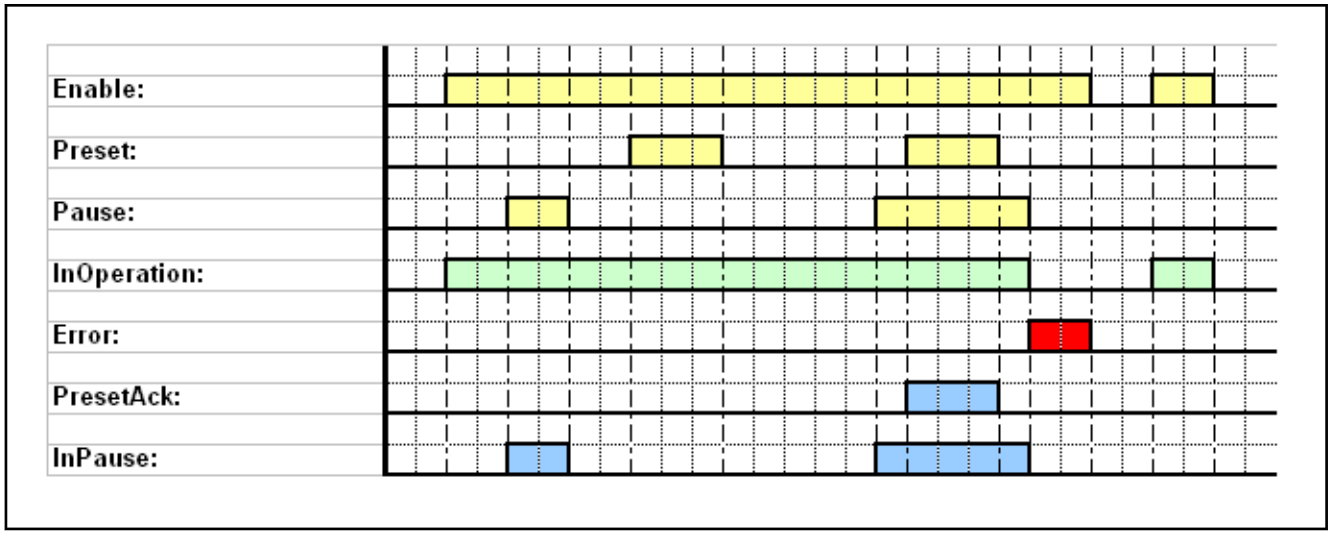


Fig.7-71: Time diagram of MB_RegiRegulateType01

Functional Description

The function block is used to write control values to a desired parameter in an axis. Depending of the desired target parameter, "ControlledValueIDN", the function block includes an integration of the "ParameterControlValue" input. The following "ControlledValueIDN" values are integrated:

- P-0-0061, A-0-2730
- P-0-0691, A-0-2610
- P-0-0692, A-0-2600
- P-0-0695, A-0-2620

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. As long as "Pause" is active, the "InPause" output is TRUE and the "ParameterControlValue" output maintains its value.

Preset

The "PresetAck" output is active if the value in "PresetValue" is taken into account. The "Preset" input has priority over "Pause". The value in "PresetValue" is taken into account on a rising edge of "Preset" if the "Enable" input is active. If successful, the "PresetAck" output is set TRUE and the value in "PresetValue" is written to the controlled parameter.

DisableCyclicWrites

Disables cyclic writing to the controlled parameter (Input "ControlledValueIDN") from the function block.

Normal mode

The RegiRegulateType only calculates a new value when the "Step" input changes. The controlled axis must be a modulo axis.

Required Parameterization

IndraMotion MLC parameterization

If a real axis is used as the controlled axis ("ControlledAxis" input), then the parameter used in the "ControlledValueIDN" input must be one of the following supported P-Parameters:

- P-0-0690, P-0-0691, P-0-0692, P-0-0694, and P-0-0695

Additionally, this P-Parameter must be configured in the MDT of the cyclic data channel in the drive.

If a virtual axis is used as the controlled axis, then the parameter used in the "ControlledValueIDN" input must be one of the following supported A-Parameters:

- A-0-2730, A-0-2615, A-0-2610, A-0-2600, A-0-2605, A-0-2620

ML_TechRegi.library/MX_TechRegi.lib

IndraMotion MLD parameterization

The parameter used as controlled parameter for the "ControlledValueIDN" (P-Parameter) input must be configured in the "AxisData" structure. Thus, the axis data structure has to be activated. The following P-Parameters are supported:

- P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

INPUT_RANGE_ERROR(16#0006)

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis ("AXIS_REF") was changed while function block is in operation
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"AXIS_REF" is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000A	S-0-0103 or A-0-0045 of the controlled axis is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000B	S-0-0091 or A-0-0032 of the controlled axis is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000C	P-0-0750 of the controlled axis is not within the valid range
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0001	Selected "ControlledValueIDN" is not supported
INPUT_INVALID_ERROR(16#0001)	16#0812	16#00A0	The type of the parameter "ControlledValueIDN" to be controlled does not match the controlled axis (A-parameter for the axis with interpolation used in the drive)
INPUT_INVALID_ERROR(16#0001)	16#0812	16#00B0	The type of the parameter "ControlledValueIDN" to be controlled does not match the controlled axis (P-parameter for the axis with interpolation used in the control)
ACCESS_ERROR (16#0003)	16#0815	16#0002	Required parameter P-0-0061 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0003	Required parameter P-0-0690 is not configured in the optional cyclic MDT data of the controlled axis

ML_TechRegi.library/MX_TechRegi.lib

ErrorID	Additional1	Additional2	Description
ACCESS_ERROR (16#0003)	16#0815	16#0004	Required parameter P-0-0691 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0005	Required parameter P-0-0692 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0006	Required parameter P-0-0694 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0007	Required parameter P-0-0695 is not configured in the optional cyclic MDT data of the controlled axis

Fig.7-72: Error codes of the MB_RegiRegulateType01 function block

7.4.6 MB_RegiRegulateType02

Brief Description

The MB_RegiRegulateType02 function block is used to write a control value to the drive. If the correction window is active, the control deviations are corrected within one parameterized correction window.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-73: Reference table of the MB_RegiRegulateType02

Interface Description

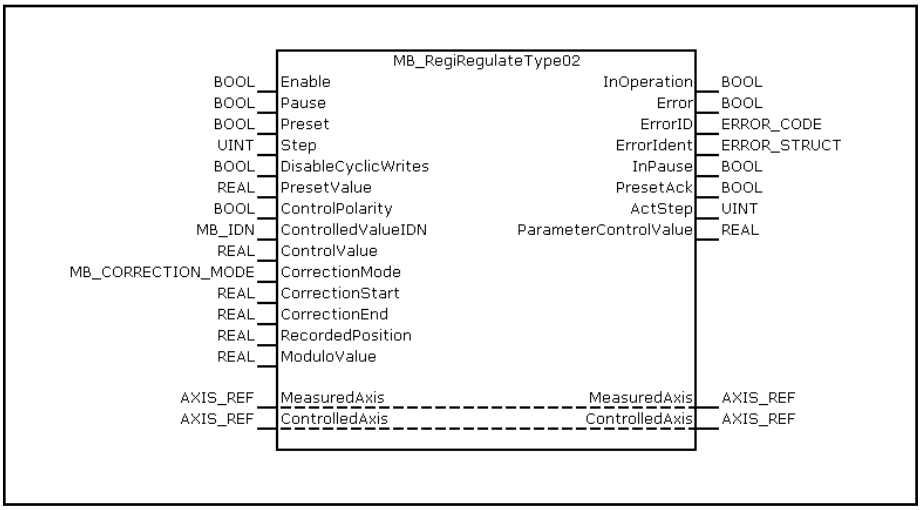


Fig.7-74: MB_RegiRegulateType02 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	MeasuredAxis	AXIS_REF	Reference to the measured axis
	ControlledAxis	AXIS_REF	Reference to the controlled axis
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces "ParameterControlValue" to "PresetValue"
	Step	UINT	Processing step
	DisableCyclicWrites	BOOL	Disables cyclic writing to the selected parameter in "ControlledValueIDN" when TRUE
	PresetValue	REAL	Value passed to the "ParameterControlValue" output on the rising edge of the "Preset" input
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE
	ControlledValueIDN	MB_IDN	SERCOS IDN of the controlled parameter (e.g. P-0-0691). The following IDNs are supported: **IndraMotion MLD**: P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695 **IndraMotion MLC**: A-0-2730, P-0-0690, A-0-2615, P-0-0691, A-0-2610, P-0-0692, A-0-2600, P-0-0694, A-0-2605, P-0-0695, A-0-2620
	ControlValue	REAL	Calculated control value from, for example, IL_PIType02
	CorrectionMode	MB_CORRECTION_MODE	CORR_ABS = absolute correction window [CorrectionStart .. CorrectionEnd]. CORR_REL = relative correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_REL_ABS = relative start .. absolute end correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "CorrectionEnd"] CORR_ABS_REL = absolute start .. relative end correction window [Start = "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_WIN_DISABLED = Correction window is disabled
	CorrectionStart	REAL	Start of the correction window
	CorrectionEnd	REAL	End of the correction window
	RecordedPosition	REAL	Measured value
	ModuloValue	REAL	Modulo value from measured axis
VAR_OUTPUT	InOperation	BOOL	Function block is in operation
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	InPause	BOOL	"Pause" is active. "ParameterControlValue" output maintains its value
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ParameterControlValue" output
	ActStep	UINT	Step Value, counting up with every new value at the output "ParameterControlValue"
	ParameterControlValue	REAL	The register controller function block calculates a value based on the selected parameter in input "ControlledValueIDN" and stores the result in output "ParameterControlValue". This value is written to the selected parameter input "ControlledValueIDN" when input "DisableCyclicWrites" = FALSE

Fig.7-75: Interface variables of the MB_RegiRegulateType02 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
ControlPolarity	BOOL			FALSE	Continuous
ControlledValueIDN	MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
ControlValue	REAL	n.def.	n.def.	0.0	Continuous
CorrectionMode	MB_CORRECTION_MODE	0	4	CORR_REL	Continuous
CorrectionStart	REAL	0.0	ModuloValue	0.0	Continuous
CorrectionEnd	REAL	0.0	ModuloValue	0.0	Continuous
RecordedPosition	REAL	0.0	ModuloValue	0.0	Change of Input "Step"
ModuloValue	REAL	0.0	n.def.	0.0	Rising edge at "Enable"

Fig.7-76: Min./max. and default values of the inputs

Time diagram

Time diagram according to the PLCopen specification.

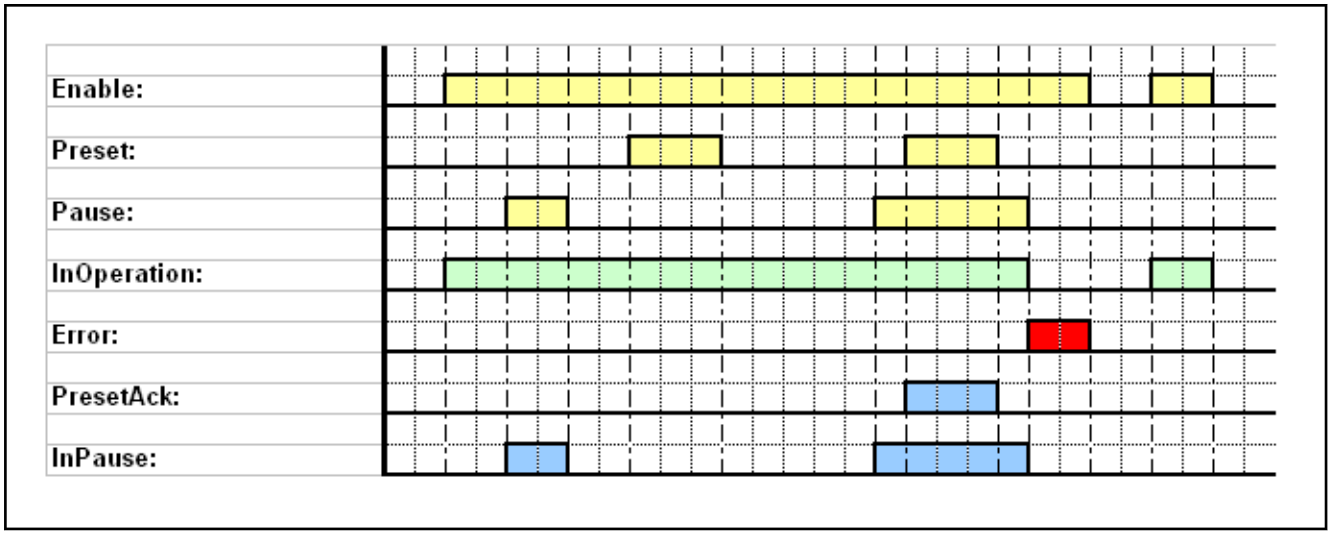


Fig.7-77: Time diagram of MB_RegiRegulateType02

Functional Description

The function block is used to write control values to a desired parameter in an axis. Depending of the desired target parameter, "ControlledValueIDN", the function block includes an integration of the "ParameterControlValue" input. The following "ControlledValueIDN" values are integrated:

- P-0-0061, A-0-2730
- P-0-0691, A-0-2610
- P-0-0692, A-0-2600
- P-0-0695, A-0-2620

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. As long as "Pause" is active, the "InPause" output is TRUE and the "ParameterControlValue" output maintains its value.

Preset

The "PresetAck" output is active if the value in "PresetValue" is taken into account. The "Preset" input has priority over "Pause". The value in "PresetValue" is taken into account on a rising edge of "Preset" if the "Enable" input is active. If successful, the "PresetAck" output is set TRUE and the value in "PresetValue" is written to the controlled parameter.

DisableCyclicWrites

Disables cyclic writing to the controlled parameter (Input "ControlledValueIDN") from the function block.

Correction Window

The correction window is an angular range relating to the measuring axis. The register controller controls the control deviations within this window. The position and size of the correction window are set with the help of the following inputs:

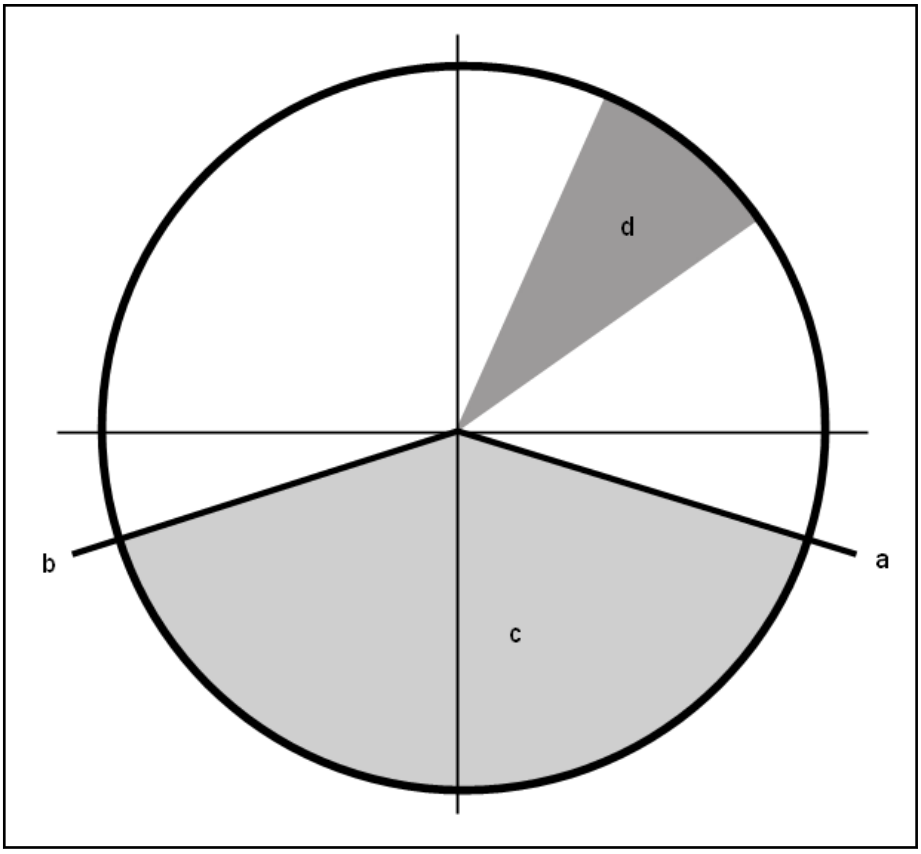
The "CorrectionMode" input sets the correction window either absolute or relative. If the "CorrectionMode" is set to CORR_WIN_DISABLED, the correction window is inactive.

The "CorrectionStart" input defines the start of the correction window.

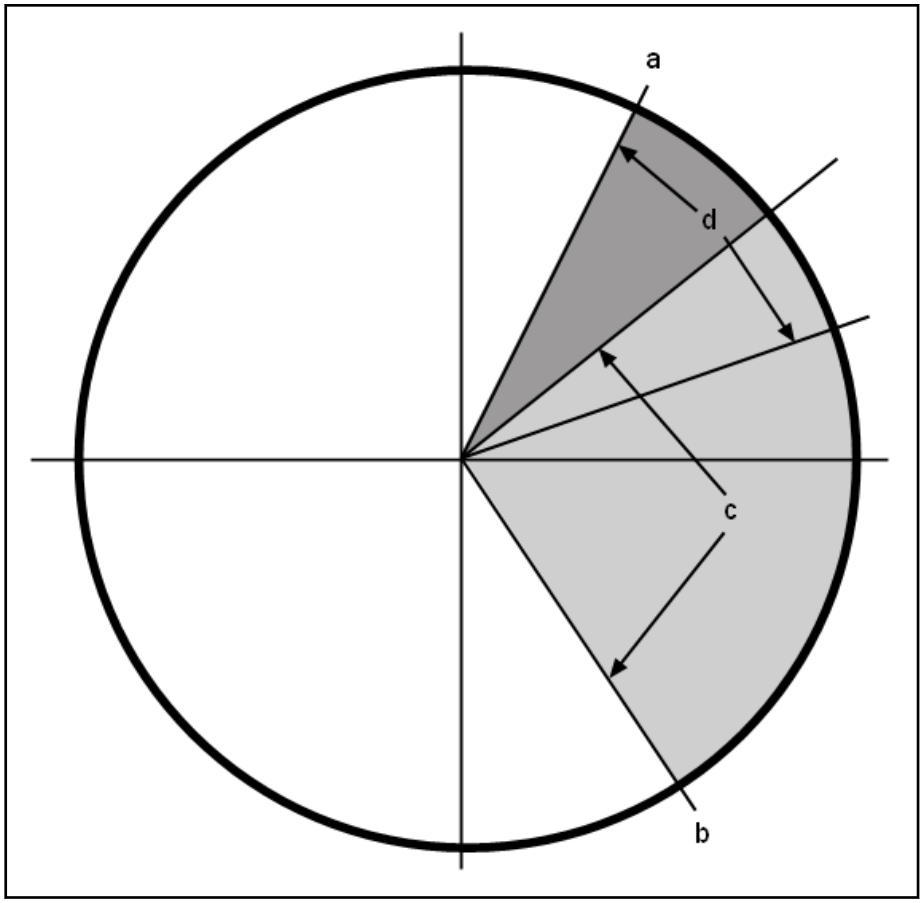
The "CorrectionEnd" input defines the end of the correction window.

The correction window supports only a positive modulo range (e.g., 0...+360 degrees) of the measuring axis.

ML_TechRegi.library/MX_TechRegi.lib



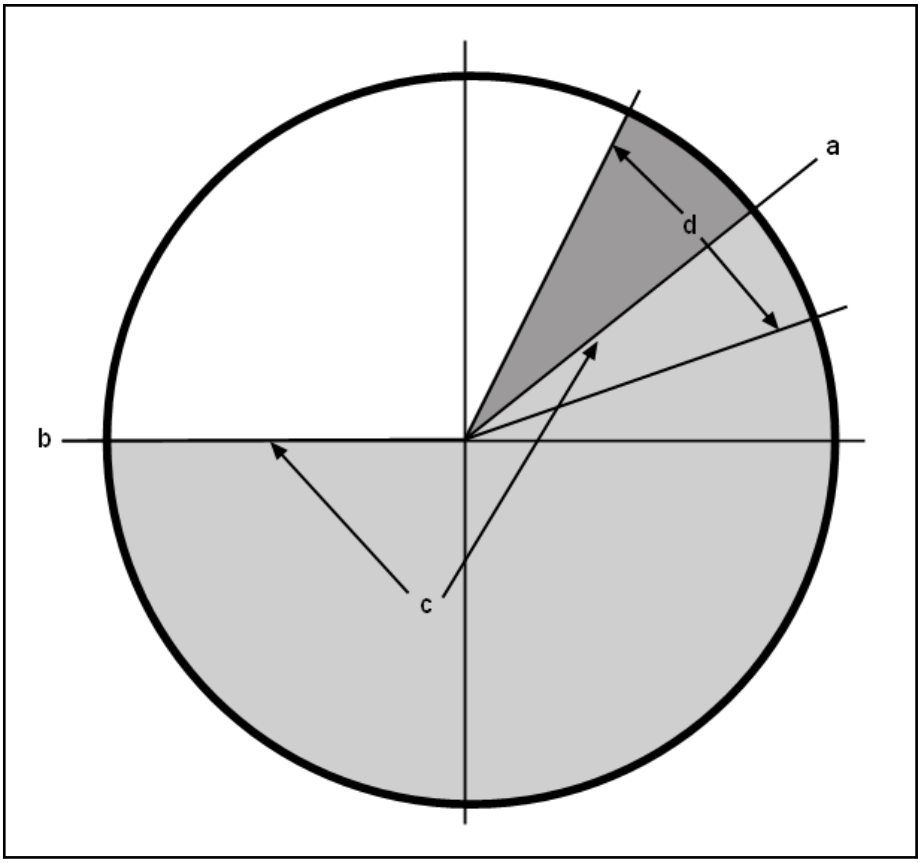
a: Start correction window
b: End correction window
c: Correction Window
d: Expectation window
Fig.7-78: Correction Window Absolute



a: Start correction window = measuring end
b: Relative end of correction window to measuring end
c: Correction Window
d: Expectation window

Fig.7-79: Correction Window Relative

ML_TechRegi.library/MX_TechRegi.lib



a: Start correction window = measuring end
b: End of correction window = perm. angle
c: Correction Window
d: Expectation window
Fig.7-80: Correction Window Relative Start Absolute End

Normal mode

The RegiRegulateType function block only calculates a new value if the "Step" Input changes. The controlled and measured axis must be modulo axes.

Required Parameterization

IndraMotion MLC parameterization

If a real axis is used as the controlled axis ("ControlledAxis" input), then the parameter used in the "ControlledValueIDN" input must be one of the following supported P-Parameters:

- P-0-0690, P-0-0691, P-0-0692, P-0-0694, and P-0-0695

☞ Additionally, this P-Parameter must be configured in the MDT of the cyclic data channel in the drive.

If a virtual axis is used as the controlled axis, then the parameter used in the "ControlledValueIDN" input must be one of the following supported A-Parameters:

- A-0-2730, A-0-2615, A-0-2610, A-0-2600, A-0-2605, A-0-2620

IndraMotion MLD parameterization

The parameter used as controlled parameter for the "ControlledValueIDN" (P-Parameter) input must be configured in the "AxisData" structure. Thus, the axis data structure has to be activated. The following P-Parameters are supported:

- P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695

Data type

MB_CORRECTION_MODE

Modes of the correction window.

Program:

```
TYPE MB_CORRECTION_MODE:
   (CORR_WIN_DISABLED :=0, (* Correction window is disabled *)
    CORR_ABS          :=1, (* Correction window is absolute *)
    CORR_REL          :=2, (* Correction window is relative *)
    CORR_REL_ABS      :=3, (* Correction window is relative start
                              and absolute end *)
    CORR_ABS_REL      :=4);(* Correction window is absolute start
                              and relative end *)
END_TYPE
```

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

INPUT_RANGE_ERROR(16#0006)

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis ("AXIS_REF") was changed while function block is in operation
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	"AXIS_REF" is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0002	Input "CorrectionStart" < 0
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0003	Input "CorrectionEnd" < 0
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0004	Value < 0 Input "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0005	Value < 0 Input "RecordedPosition"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0006	Input "CorrectionStart" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0007	Input "CorrectionEnd" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0008	Input "RecordedPosition" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0009	Correction path > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000A	S-0-0103 or A-0-0045 of the controlled axis is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000B	S-0-0091 or A-0-0032 of the controlled axis is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000C	P-0-0750 of the controlled axis is not within the valid range
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0001	Selected "ControlledValueIDN" is not supported

ML_TechRegi.library/MX_TechRegi.lib

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0002	Input invalid "CorrectionMode"
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0003	"CorrectionStart" = "CorrectionEnd"
INPUT_INVALID_ERROR(16#0001)	16#0812	16#00A0	The type of the parameter "ControlledValueIDN" to be controlled does not match the controlled axis (A-parameter for the axis with interpolation used in the drive)
INPUT_INVALID_ERROR(16#0001)	16#0812	16#00B0	The type of the parameter "ControlledValueIDN" to be controlled does not match the controlled axis (P-parameter for the axis with interpolation used in the control)
ACCESS_ERROR (16#0003)	16#0815	16#0002	Required parameter P-0-0061 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0003	Required parameter P-0-0690 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0004	Required parameter P-0-0691 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0005	Required parameter P-0-0692 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0006	Required parameter P-0-0694 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0007	Required parameter P-0-0695 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0003)	16#0815	16#0009	The required parameter P-0-0753 is not configured in the optional cyclic AT data of the axis measured.

Fig.7-81: Error codes of the MB_RegiRegulateType02 function block

7.4.7 MB_RegiRegulateType03

Brief Description

The MB_RegiRegulateType03 function block is used to write a control value to the drive using a parameterized correction ramp.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-82: Reference table of the MB_RegiRegulateType03

Interface Description

ML_TechRegi.library/MX_TechRegi.lib

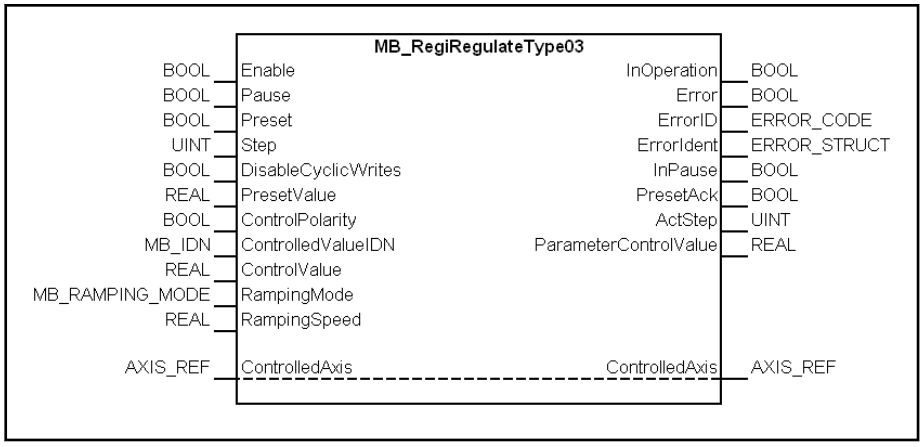


Fig.7-83: MB_RegiRegulateType03 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	ControlledAxis	AXIS_REF	Reference to the controlled axis
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces "ParameterControlValue" to "PresetValue"
	Step	UINT	Processing step
	DisableCyclicWrites	BOOL	Disables cyclic writing to the selected parameter in "ControlledValueIDN" when TRUE
	PresetValue	REAL	Value passed to the "ParameterControlValue" output on the rising edge of the "Preset" input
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE
	ControlledValueIDN	MB_IDN	SERCOS IDN of the controlled parameter (e.g. P-0-0691). The following IDNs are supported: **IndraMotion MLD**: P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695 **IndraMotion MLC**: A-0-2730, P-0-0690, A-0-2615, P-0-0691, A-0-2610, P-0-0692, A-0-2600, P-0-0694, A-0-2605, P-0-0695, A-0-2620
	ControlValue	REAL	Calculated control value from, for example, IL_PIType02
	RampingMode	MB_RAMPING_MODE	The following ramping modes are supported: • 0 = RAMPING_CONSTANT ramping velocity is constant • 1 = RAMPING_ADAPTIVE ramping velocity is calculated based on the time difference in the "Step" input
	RampingSpeed	REAL	This value uses the gradient of the ramping

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
VAR_OUTPUT	InOperation	BOOL	Function block is in operation
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "ParameterControlValue" output maintains its value
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ParameterControlValue" output
	ActStep	UINT	Step Value, counting up with every new value at the output "ParameterControlValue"
	ParameterControlValue	REAL	The register controller function block calculates a value based on the selected parameter in input "ControlledValueIDN" and stores the result in output "ParameterControlValue". This value is written to the selected parameter input "ControlledValueIDN" when input "DisableCyclicWrites" = FALSE

Fig.7-84: Interface variables of the MB_RegiRegulateType03 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
ControlPolarity	BOOL			FALSE	Continuous
ControlledValueIDN	MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
ControlValue	REAL	n.def.	n.def.	0.0	Continuous
RampingMode	MB_RAMPING_MODE	0.0	1	RMPING_CONS	Rising edge at "Enable"
RampingSpeed	REAL	0.0	n.def.	1.0	Continuous

Fig.7-85: Min./max. and default values of the inputs

Time diagram

Time diagram according to the PLCopen specification.

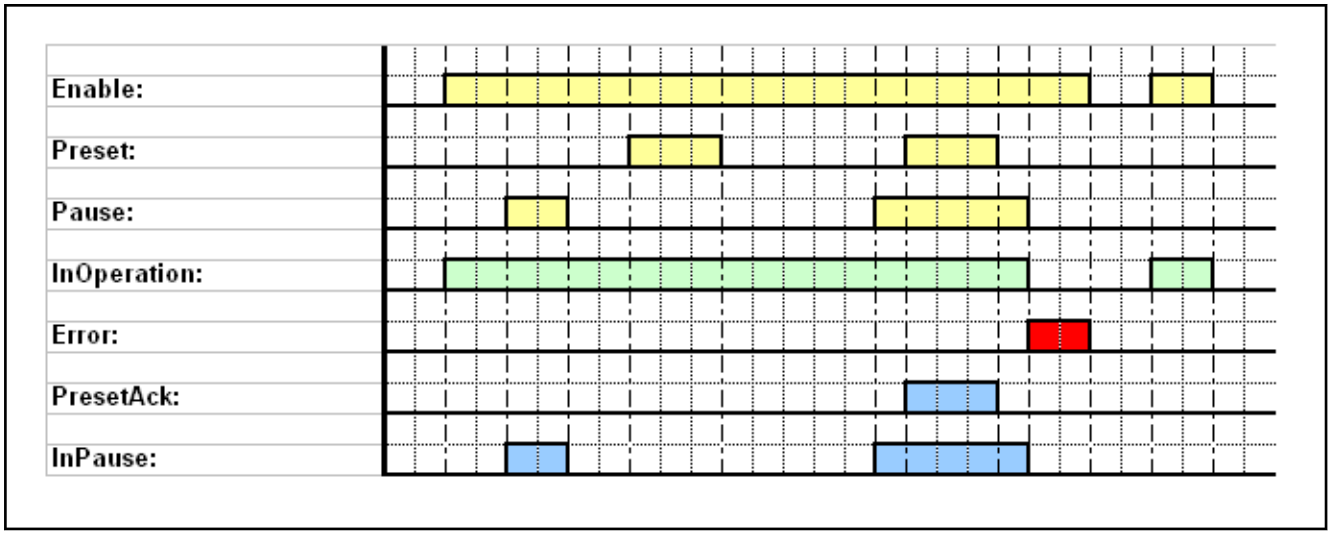


Fig.7-86: Time diagram of MB_RegiRegulateType03

Functional Description

The function block is used to write control values to a desired parameter in an axis. Depending of the desired parameter, "ControlledValueIDN", the function block includes the integration of the "ParameterControlValue" input. The following "ControlledValueIDN" values are integrated:

- P-0-0061, A-0-2730
- P-0-0691, A-0-2610
- P-0-0692, A-0-2600
- P-0-0695, A-0-2620

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. As long as "Pause" is active, the "InPause" output is TRUE and the "ParameterControlValue" output maintains its value.

Preset

The "PresetAck" output is active if the value in "PresetValue" is taken into account. The "Preset" input has priority over "Pause". The value in "PresetValue" is taken into account on a rising edge of "Preset" if the "Enable" input is active. If successful, the "PresetAck" output is set TRUE and the value in "PresetValue" is written to the controlled parameter.

DisableCyclicWrites

Disables cyclic writing to the controlled parameter (Input "ControlledValueIDN") from the function block.

Ramping Mode

There are two different ramping modes. One mode is constant ramping with the gradient from the "RampingSpeed" ("RAMPING_CONSTANT") input . The other mode depends on the change of the "Step" input. The faster the input "Step" is changing; the steeper the ramping of "ParameterControlValue" ("RAMPING_ ADAPTIVE").

- RAMPING_CONSTANT:

  In the mode, the change of output "ParameterControlValue" depends on input "RampingSpeed". For example, if input "RampingSpeed" is set to 1, then ramping of output "ParameterControlValue" is 1 degree/second for parameter P-0-0691.

ML_TechRegi.library/MX_TechRegi.lib

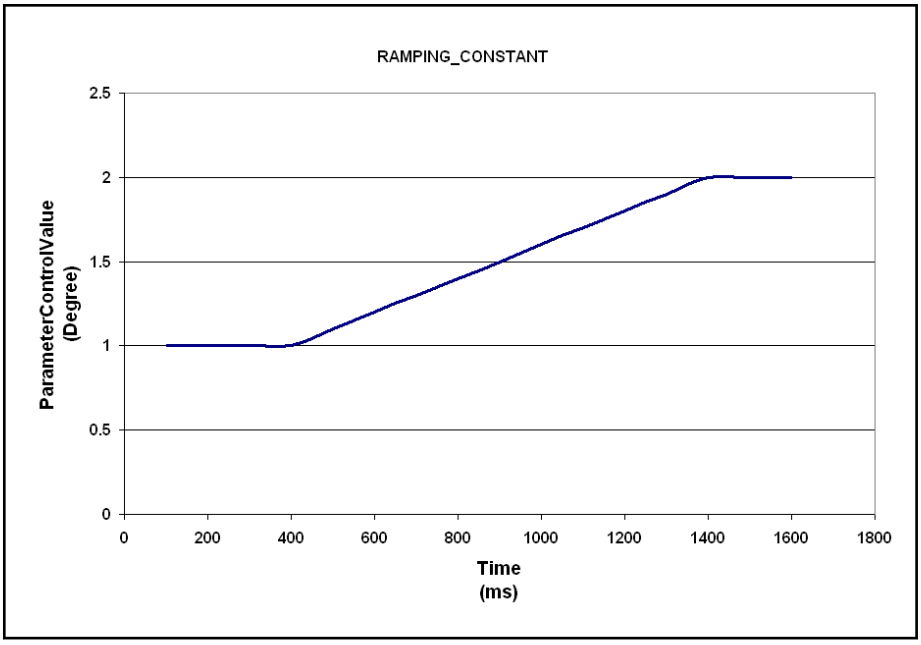


Fig.7-87: Ramping Constant

- RAMPING_ADAPTIVE:

  In the mode, the change of "ParameterControlValue" depends on the time difference between changes of input "Step". For example, a change every 0.6(0.3) seconds on input "Step" result in ramping of output "ParameterControlValue" 1(2) degree/second for parameter P-0-0691.

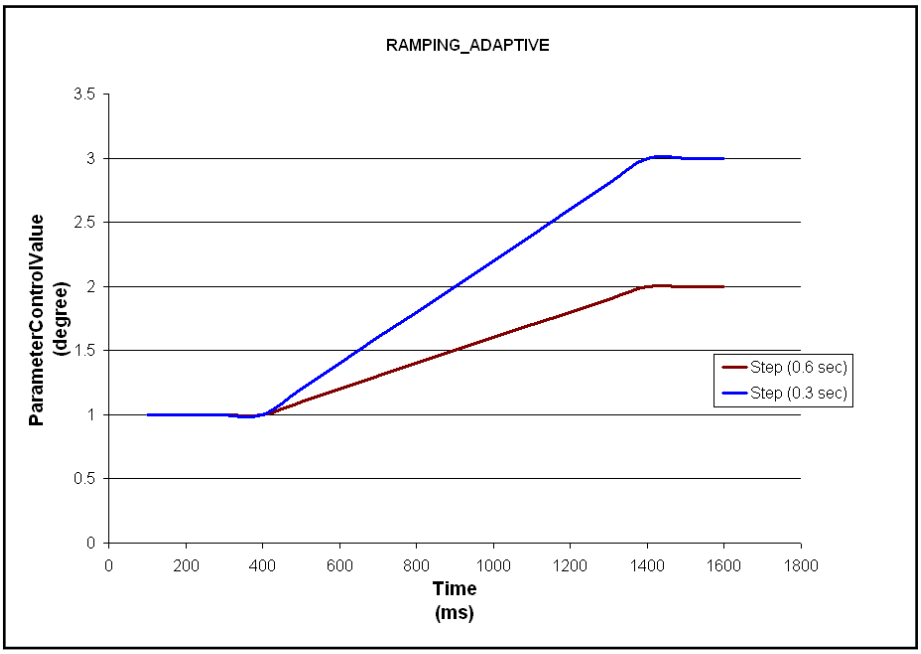


Fig.7-88: Ramping Adaptive

Ramping velocity The ramping velocity is only valid in connection with ramping mode RAMPING_CONSTANT.

Normal mode The RegiRegulateType only calculates a new value when the "Step" input changes. The controlled axis must be a modulo axis.

ML_TechRegi.library/MX_TechRegi.lib

Required Parameterization

IndraMotion MLC parameterization

If a real axis is used as the controlled axis ("ControlledAxis" input), then the parameter used in the "ControlledValueIDN" input must be one of the following supported P-Parameters:

- P-0-0690, P-0-0691, P-0-0692, P-0-0694, and P-0-0695

Additionally, this P-Parameter must be configured in the MDT of the cyclic data channel in the drive.

If a virtual axis is used as the controlled axis, then the parameter used in the "ControlledValueIDN" input must be one of the following supported A-Parameters:

- A-0-2730, A-0-2615, A-0-2610, A-0-2600, A-0-2605, A-0-2620

IndraMotion MLD parameterization

The parameter used as controlled parameter for the "ControlledValueIDN" (P-Parameter) input must be configured in the "AxisData" structure. Thus, the axis data structure has to be activated. The following P-Parameters are supported:

- P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

INPUT_RANGE_ERROR(16#0006)

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected axis ("AXIS_REF") was changed while function block is in operation
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000A	S-0-0103 or A-0-0045 of the controlled axis is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000B	S-0-0091 or A-0-0032 of the controlled axis is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#000C	P-0-0750 of the controlled axis is not within the valid range
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0001	Selected "ControlledValueIDN" is not supported
INPUT_INVALID_ERROR(16#0001)	16#0812	16#000C	Value < 0 for "RampingSpeed" input
INPUT_INVALID_ERROR(16#0001)	16#0812	16#000D	Modulo value of the controlled axis is 0 or could not be read

ML_TechRegi.library/MX_TechRegi.lib

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR(16#0001)	16#0812	16#00A0	The type of the parameter "ControlledValueIDN" to be controlled does not match the controlled axis (A-parameter for the axis with interpolation used in the drive)
INPUT_INVALID_ERROR(16#0001)	16#0812	16#00B0	The type of the parameter "ControlledValueIDN" to be controlled does not match the controlled axis (P-parameter for the axis with interpolation used in the control)
ACCESS_ERROR (16#0004)	16#0815	16#0002	Required parameter P-0-0061 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0004)	16#0815	16#0003	Required parameter P-0-0690 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0004)	16#0815	16#0004	Required parameter P-0-0691 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0004)	16#0815	16#0005	Required parameter P-0-0692 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0004)	16#0815	16#0006	Required parameter P-0-0694 is not configured in the optional cyclic MDT data of the controlled axis
ACCESS_ERROR (16#0004)	16#0815	16#0007	Required parameter P-0-0695 is not configured in the optional cyclic MDT data of the controlled axis

Fig.7-89: Error codes of the MB_RegiRegulateType03 function block

7.4.8 MB_RegiRegulateType04

Brief Description

The MB_RegiRegulateType04 function block is used to write a control value to up to eight drives. If the correction window is active, the control deviations are corrected within one parameterized correction window.

Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library

Fig.7-90: Reference table of the MB_RegiRegulateType04

Interface Description

ML_TechRegi.library/MX_TechRegi.lib

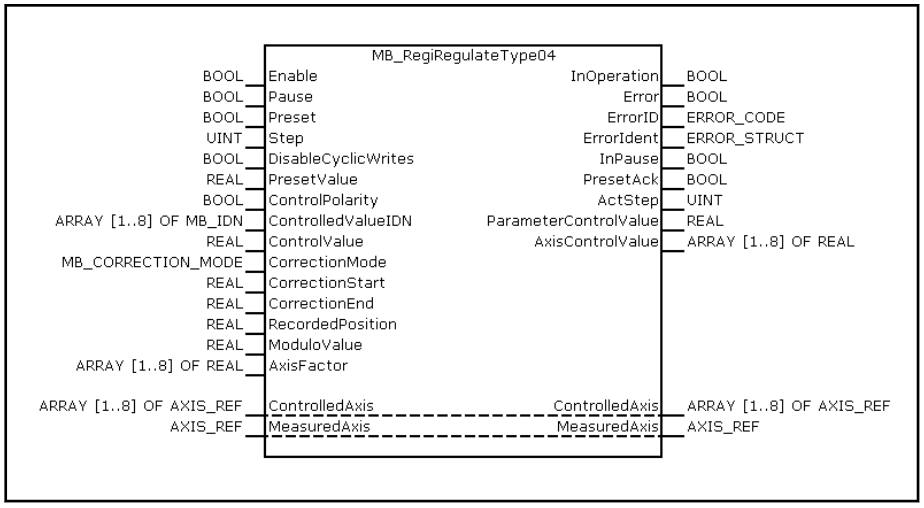


Fig.7-91: MB_RegiRegulateType04 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	MeasuredAxis	AXIS_REF	Reference to the measured axis
	ControlledAxis	ARRAY [1..8] OF AXIS_REF	Reference to the controlled axes
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pause	BOOL	The "Pause" input will only be evaluated while the "Enable" input is active
	Preset	BOOL	The "Preset" input is only evaluated when "Enable" is TRUE. A positive edge of this input forces "ParameterControlValue" to "PresetValue"
	Step	UINT	Processing step
	DisableCyclicWrites	BOOL	Disables cyclic writing to the selected parameter in "ControlledValueIDN" when TRUE
	PresetValue	REAL	Value passed to the "ParameterControlValue" output on the rising edge of the "Preset" input
	ControlPolarity	BOOL	The polarity of the control value is inverted while this input is TRUE
	ControlledValueIDN	ARRAY [1...8] OF MB_IDN	SERCOS IDN of the controlled parameter (e.g. P-0-0691). The following IDNs are supported: **IndraMotion MLD**: P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695 **IndraMotion MLC**: A-0-2730, P-0-0690, A-0-2615, P-0-0691, A-0-2610, P-0-0692, A-0-2600, P-0-0694, A-0-2605, P-0-0695, A-0-2620
	ControlValue	REAL	Calculated control value from, for example, IL_PIType02

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
	CorrectionMode	MB_CORRECTION_MODE	CORR_ABS = absolute correction window [CorrectionStart .. CorrectionEnd]. CORR_REL = relative correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_REL_ABS = relative start .. absolute end correction window [Start = "RecordedPosition" + "CorrectionStart" .. End = "CorrectionEnd"] CORR_ABS_REL = absolute start .. relative end correction window [Start = "CorrectionStart" .. End = "RecordedPosition" + "CorrectionEnd"] CORR_WIN_DISABLED = Correction window is disabled
	CorrectionStart	REAL	Start of the correction window
	CorrectionEnd	REAL	End of the correction window
	RecordedPosition	REAL	Measured value
	ModuloValue	REAL	Modulo value from measured axis
	AxisFactor	ARRAY [1..8] OF REAL	Factor to archive an axis dependent scaling. The value of "ParameterControlValue" is multiplied with this axis-dependent factor (ARRAY) and is distributed to every controlled axis
VAR_OUTPUT	InOperation	BOOL	Function block is in operation
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	InPause	BOOL	"Pause" is active. "ParameterControlValue" and "AxisControlValue" outputs maintain their values.
	PresetAck	BOOL	"Preset" is done. The preset value is copied to the "ParameterControlValue" output
	ActStep	UINT	Step Value, counting up with every new value at the output "ParameterControlValue"
	ParameterControlValue	REAL	The register controller function block calculates a value based on the selected parameter in input "ControlledValueIDN" and stores the result in output "ParameterControlValue". This value is written to the selected parameter input "ControlledValueIDN" when input "DisableCyclicWrites" = FALSE
	AxisControlValue	ARRAY [1..8] OF REAL	Calculated value of the register controller function block written to the "ControlledValueIDN" input. The "AxisControlValue" is the scaled "ParameterControlValue". Used for diagnostic purposes or if "DisableCyclicWrites" = TRUE for further operation

Fig.7-92: Interface variables of the MB_RegiRegulateType04 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
Step	UINT	0	65.535	0	Continuous. Starts at rising edge of "Enable" and falling edge of "Pause"
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
ControlPolarity	BOOL			FALSE	Continuous
ControlledVa-lueIDN	ARRAY [1...8] OF MB_IDN	n.def.	n.def.	0	Rising edge at "Enable"
ControlValue	REAL	n.def.	n.def.	0.0	Continuous
CorrectionMode	MB_CORREC-TION_MODE	0	4	CORR_REL	Continuous
CorrectionStart	REAL	0.0	ModuloValue	0.0	Continuous
CorrectionEnd	REAL	0.0	ModuloValue	0.0	Continuous
RecordedPosition	REAL	0.0	ModuloValue	0.0	Change of Input "Step"
ModuloValue	REAL	0.0	n.def.	0.0	Rising edge at "Enable"
AxisFactor	ARRAY [1..8] OF REAL	n.def.	n.def.	1.0	Continuous

Fig.7-93: Min./max. and default values of the inputs

Time diagram

Time diagram according to the PLCopen specification.

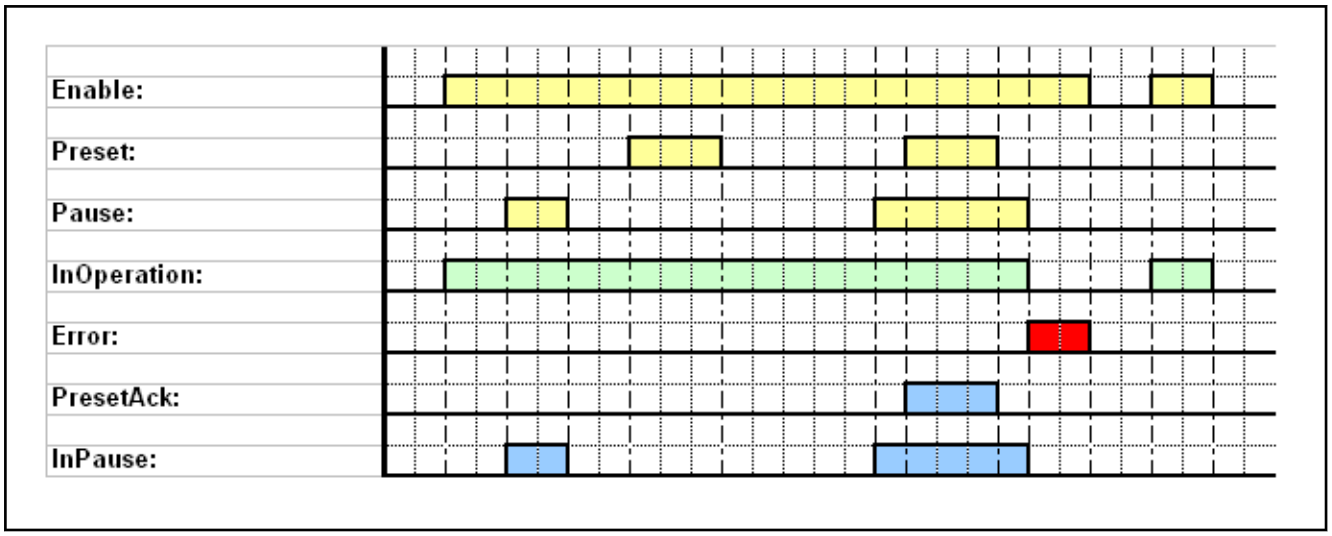


Fig.7-94: Time diagram of MB_RegiRegulateType04

ML_TechRegi.library/MX_TechRegi.lib

Functional Description

The function block is used to write control values to a desired parameter in an axis. Depending of the desired parameter, "ControlledValueIDN", the function block includes the integration of the "ParameterControlValue" input. An integration is done for the following ControlledValueIDN values:

- P-0-0061, A-0-2730
- P-0-0691, A-0-2610
- P-0-0692, A-0-2600
- P-0-0695, A-0-2620

Pause

The "Pause" input is only evaluated as long as the "Enable" input is active. The "ActStep" output is updated even if the "Pause" input is active. As long as "Pause" is active, the "InPause" output is TRUE and the "ParameterControlValue" and "AxisControlValue" outputs maintain their values.

Preset

The "PresetAck" output is active if the value in "PresetValue" is taken into account. The "Preset" input has priority over "Pause". The value in "PresetValue" is taken into account on a rising edge of "Preset" if the "Enable" input is active. If successful, the "PresetAck" output is set TRUE and the value in "PresetValue" is written to the controlled parameter.

DisableCyclicWrites

Disables cyclic writing to the controlled parameter (Input "ControlledValueIDN") from the function block.

Correction Window

The correction window is an angular range relating to the measuring axis. The register controller controls the control deviations within this window. The position and size of the correction window are set with the help of the following inputs:

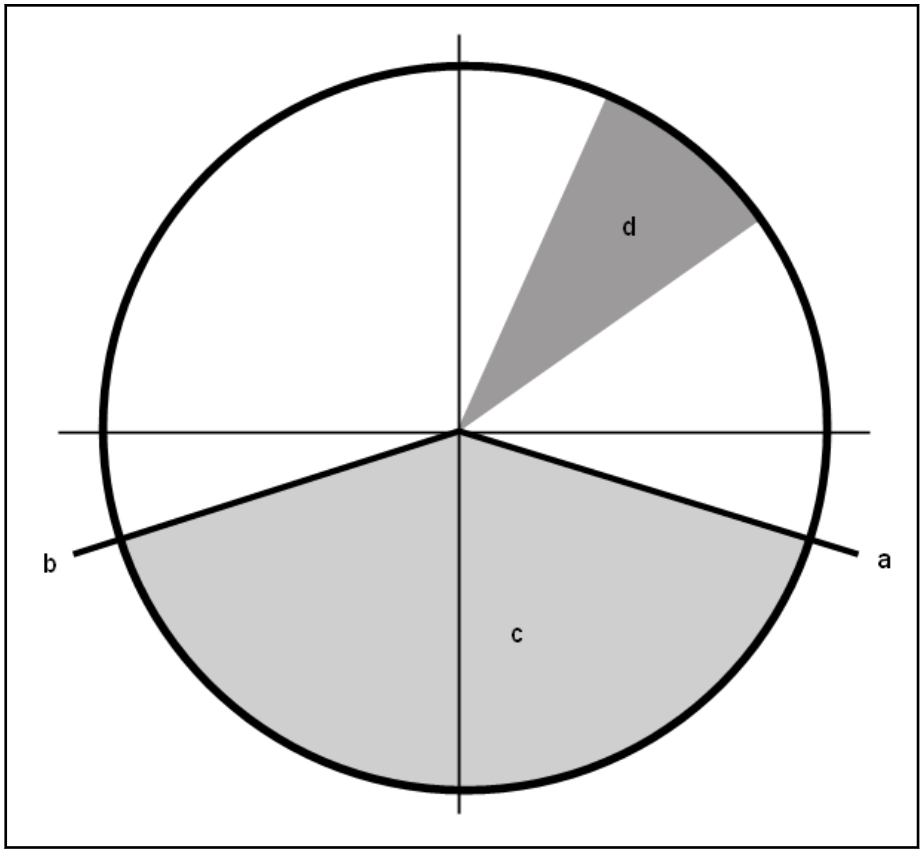
The "CorrectionMode" input sets the correction window either absolute or relative. If the "CorrectionMode" is set to CORR_WIN_DISABLED, the correction window is inactive.

The "CorrectionStart" input defines the start of the correction window.

The "CorrectionEnd" input defines the end of the correction window.

The correction window supports only a positive modulo range (e.g., 0...+360 degrees) of the measuring axis.

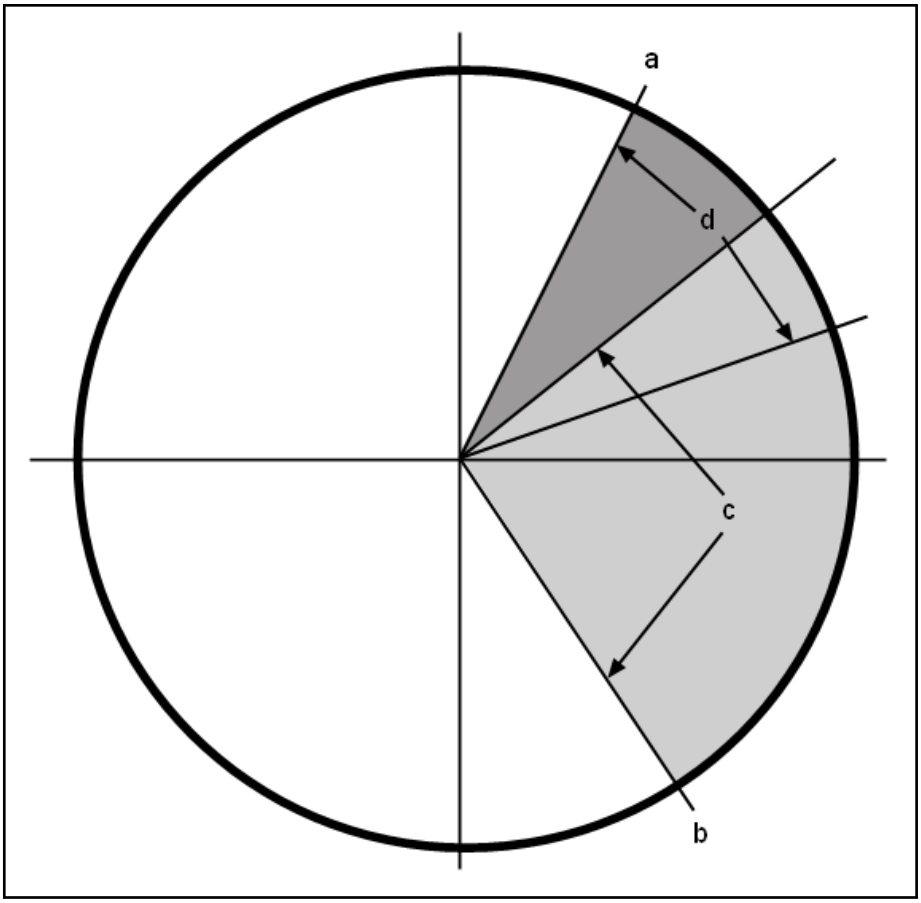
ML_TechRegi.library/MX_TechRegi.lib



a: Start correction window
b: End correction window
c: Correction Window
d: Expectation window

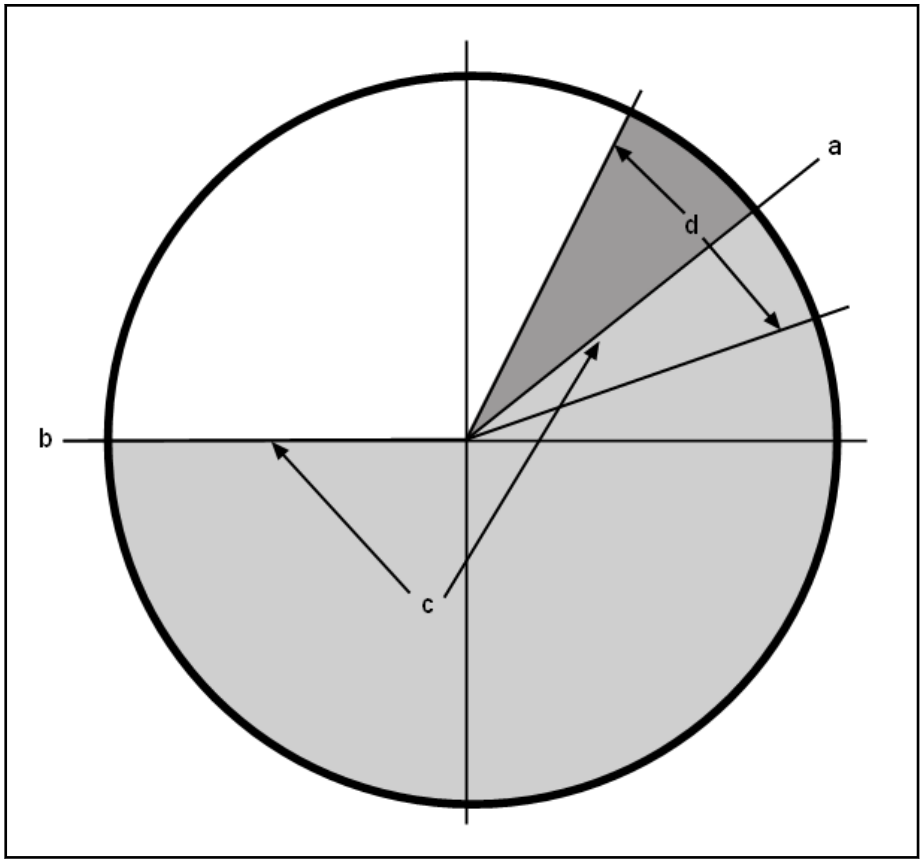
Fig.7-95: Correction Window Absolute

ML_TechRegi.library/MX_TechRegi.lib



a: Start correction window = measuring end
b: Relative end of correction window to measuring end
c: Correction Window
d: Expectation window

Fig.7-96: Correction Window Relative



a: Start correction window = measuring end
b: End of correction window = perm. angle
c: Correction Window
d: Expectation window
Fig.7-97: Correction Window Relative Start Absolute End

Multiple axes

The calculated control values are written to the respective parameters ("ControlValueIDN") of up to 8 controlled axes. The axes are referenced with the "ControlledAxis[x]" input.

Individual Control Value Scaling

The control value of each individual axis can be scaled with the "AxisFactor[x]" input.

Normal mode

The RegiRegulateType function block only calculates a new value if the "Step" Input changes. The controlled and measured axis must be modulo axes.

Required Parameterization

IndraMotion MLC parameterization

If a real axis is used as the controlled axis ("ControlledAxis" input), then the parameter used in the "ControlledValueIDN" input must be one of the following supported P-Parameters:

- P-0-0690, P-0-0691, P-0-0692, P-0-0694, and P-0-0695

☞ Additionally, this P-Parameter must be configured in the MDT of the cyclic data channel in the drive.

If a virtual axis is used as the controlled axis, then the parameter used in the "ControlledValueIDN" input must be one of the following supported A-Parameters:

- A-0-2730, A-0-2615, A-0-2610, A-0-2600, A-0-2605, A-0-2620

ML_TechRegi.library/MX_TechRegi.lib

IndraMotion MLD parameterization

The parameter used as controlled parameter for the "ControlledValueIDN" (P-Parameter) input must be configured in the "AxisData" structure. Thus, the axis data structure has to be activated. The following P-Parameters are supported:

- P-0-0061, P-0-0690, P-0-0691, P-0-0692, P-0-0694, P-0-0695

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

INPUT_RANGE_ERROR(16#0006)

ErrorID	Additional1	Additional2	Description
RESOURCE_ERROR (16#0003)	16#0001	16#0000	Drive is not enabled or drive error
RESOURCE_ERROR (16#0003)	16#0004	16#0000	Drive firmware is not supported
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
RESOURCE_ERROR (16#0003)	16#0009	16#0000	Selected Axis (AXIS_REF) was changed while the function block is operating
INPUT_RANGE_ERROR(16#0006)	16#000D	16#0000	AXIS_REF is not within the valid range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0002	Input "CorrectionStart" < 0
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0003	Input "CorrectionEnd" < 0
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0004	Value < 0 Input "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0005	Value < 0 Input "RecordedPosition"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0006	Input "CorrectionStart" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0007	Input "CorrectionEnd" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0008	Input "RecordedPosition" > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0009	Correction path > "ModuloValue"
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0**x**10	S-0-0103 or A-0-0045 of the controlled axis [**x**] is outside the validity range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0**x**20	S-0-0091 or A-0-0032 of the controlled axis [**x**] is outside the validity range
INPUT_RANGE_ERROR(16#0006)	16#0811	16#0**x**30	P-0-0750 of the controlled axis [**x**] is not within the valid range
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0**x**01	Selected "ControlledValueIDN[**x**]" is not supported
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0002	Input invalid "CorrectionMode"
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0003	"CorrectionStart" = "CorrectionEnd"

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0**x**A0	The type of the parameter to be controlled "ControlledValueIDN[**x**]" does not match the controlled axis "Controlled Axis[**x**]" (A-parameter for the axis with interpolation used in the drive)
INPUT_INVALID_ERROR(16#0001)	16#0812	16#0**x**B0	The type of the parameter to be controlled "ControlledValueIDN[**x**]" does not match the controlled axis "Controlled Axis[**x**]" (P-parameter for the axis with interpolation used in the control)
ACCESS_ERROR (16#0003)	16#0815	16#0**x**02	Required parameter P-0-0061 is not configured in the optional cyclic MDT data of the controlled axis[**x**]
ACCESS_ERROR (16#0003)	16#0815	16#0**x**03	Required parameter P-0-0690 is not configured in the optional cyclic MDT data of the controlled axis[**x**]
ACCESS_ERROR (16#0003)	16#0815	16#0**x**04	Required parameter P-0-0691 is not configured in the optional cyclic MDT data of the controlled axis[**x**]
ACCESS_ERROR (16#0003)	16#0815	16#0**x**05	Required parameter P-0-0692 is not configured in the optional cyclic MDT data of the controlled axis[**x**]
ACCESS_ERROR (16#0003)	16#0815	16#0**x**06	Required parameter P-0-0694 is not configured in the optional cyclic MDT data of the controlled axis[**x**]
ACCESS_ERROR (16#0003)	16#0815	16#0**x**07	Required parameter P-0-0695 is not configured in the optional cyclic MDT data of the controlled axis[**x**]
ACCESS_ERROR (16#0003)	16#0815	16#0009	The required parameter P-0-0753 is not configured in the optional cyclic AT data of the axis measured.

Fig.7-98: Error codes of the MB_RegiRegulateType04 function block

7.5 Register Setpoint Lock Function Block

7.5.1 Overview

This section describes the register setpoint lock function block that is part of the libraries (IndraMotion MLC) and MX_TechRegi (IndraMotion MLD).

7.5.2 MB_SetpointLockType01

Brief Description The MB_RegiSetpointLockType01 function block is used to latch a value for a register controller.

ML_TechRegi.library/MX_TechRegi.lib

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-99: Reference table of the MB_SetpointLockType01

Interface Description

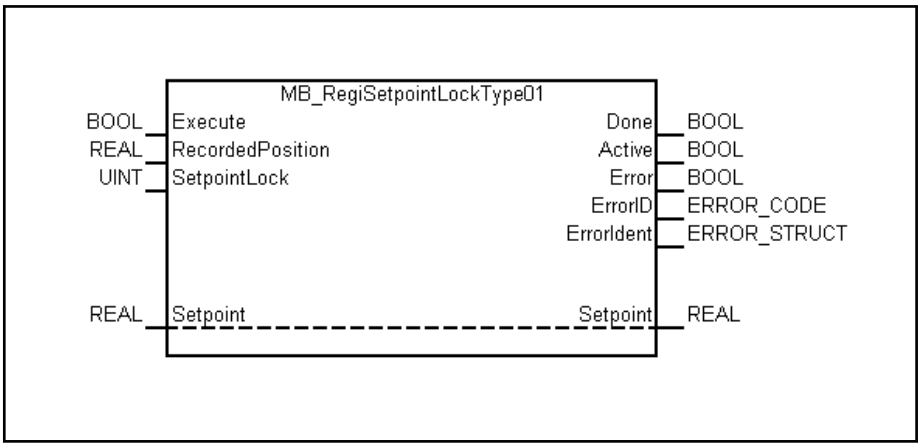


Fig.7-100: MB_SetpointLockType01 Function Block

I/O type	Name	Data type	Description
VAR_IN_OUT	Setpoint	REAL	Setpoint
VAR_IN	Execute	BOOL	The setpoint lock function block becomes active on a rising edge
	RecordedPosition	REAL	The recorded position is copied to input "Setpoint" when "SetpointLock" changes
	SetpointLock	UINT	Value must increase if the value in "RecordedPosition" is locked. At 65.535 it rolls over to 0 (modulo type value). Typically an "actStep" output from a tech function block is used as an input
VAR_OUT	Done	BOOL	Function block is done and the value in input "RecordPosition" is copied to "Setpoint"
	Active	BOOL	Function block is active
	Error	BOOL	It indicates an error occurred during function block operation
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description

Fig.7-101: Interface variables of the MB_RegiSetpointLockType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

ML_TechRegi.library/MX_TechRegi.lib

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
RecordedPosition	REAL			0.0	Change of "Setpoint-Lock"
SetpointLock	UINT	0	65.535	0	Continuous. Initialized at rising edge of "Execute"

Fig.7-102: Min./max. and default values of the inputs

Time diagram

The following diagram shows the signal-time diagram of the MB_RegiSetpointLockType01 function block.

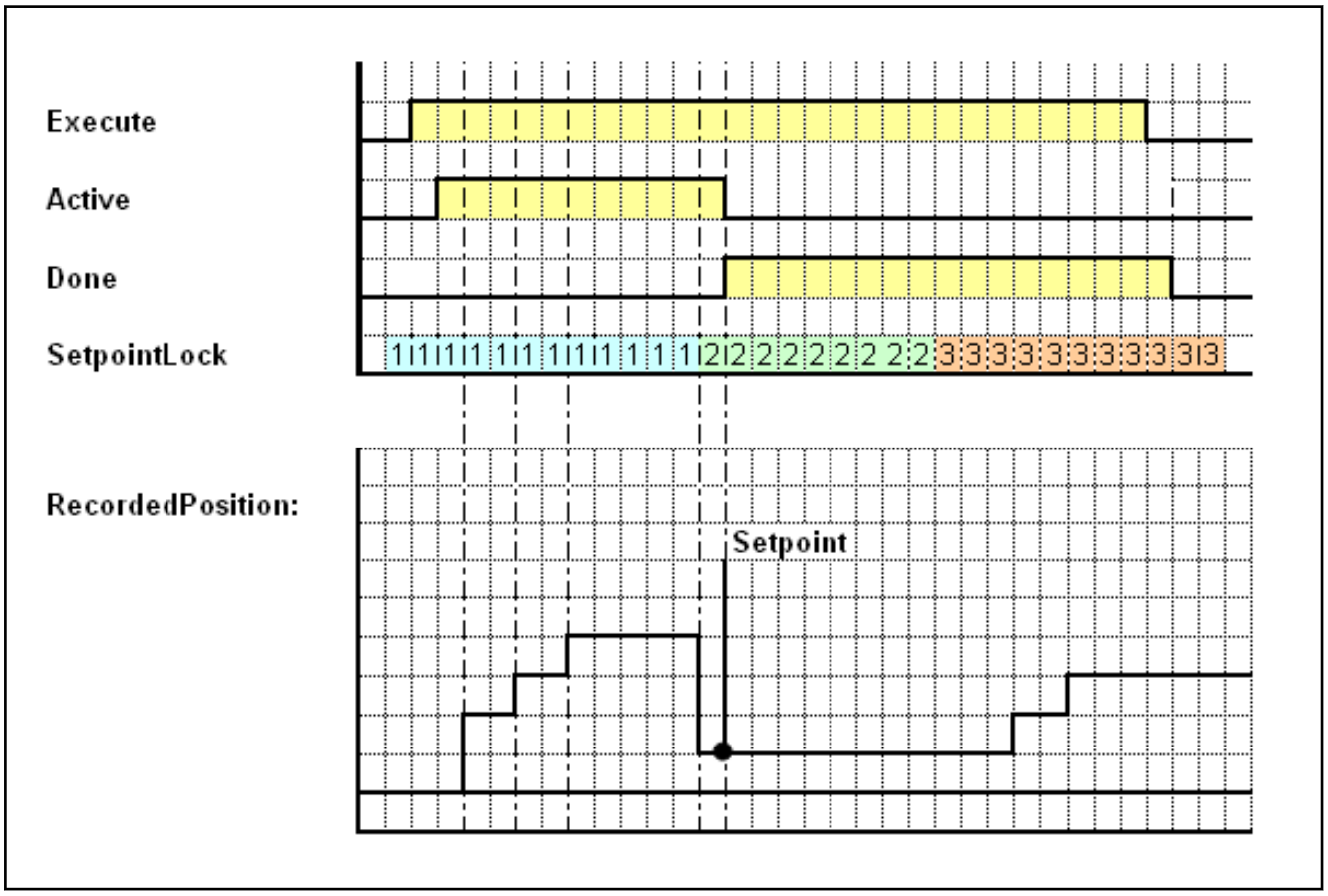


Fig.7-103: Time Diagram of MB_RegiSetpointLockType01

Functional Description

The function block sets "Setpoint" to the next occurring new value at the "RecordedPosition" input. The next occurrence is detected when the "SetpointLock" input changes. This can be used to set a new setpoint, e.g. for a register controller.

Error Handling

The function block generates the following error messages in Additional1/Additional2 using the "**F_RELATED_TABLE**", 16#0170:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block

Fig.7-104: Error codes of the MB_RegiSetpointLockType01 function block

ML_TechRegi.library/MX_TechRegi.lib

7.6 Register Quality Function Block

7.6.1 Overview

This section describes the register quality function block that is part of the libraries (IndraMotion MLC) and MX_TechRegi (IndraMotion MLD).

7.6.2 IL_RegiQualityType01

Brief Description

This function block is event-triggered and evaluates up to 16 register deviations. With the help of two limit values, the printed images are divided into "good, medium and bad" quality and summed up. The quality of the printed image depends on the color mark.

Target system	Library
IndraMotion MLC 12VRS	ML_TechRegi.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechRegi.lib

Fig.7-105: Reference table of the IL_RegiQualityType01

Interface Description

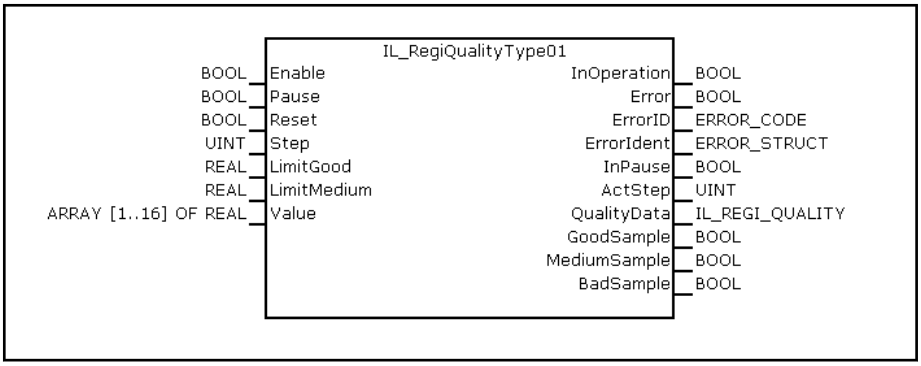


Fig.7-106: IL_RegiQualityType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Calculation of data in each cycle while Enable = TRUE
	Pause	BOOL	The output data is not changed, while Pause = TRUE
	Reset	BOOL	Function block reset
	Step	UINT	Current calculation step
	LimitGood	REAL	Max. absolute value [mm] of the register deviation to classify the quality of the printed image still as "good"
	LimitedMedium	REAL	If the absolute value of the register deviation [mm] is between "LimitGood" and "LimitMedium", the printed image is classified as "medium" quality. A register deviation greater than "LimitMedium" would be considered as "bad" quality
	Value	ARRAY [1.0.16] OF REAL	It contains register deviations of a printed images (up to 16 register marks)

ML_TechRegi.library/MX_TechRegi.lib

I/O type	Name	Data type	Description
VAR_OUTPUT	InOperation	BOOL	Processing without errors, output data is valid
	Error	BOOL	Error (see ErrorID and ErrorIdent)
	ErrorID	ERROR_CODE	Error description
	ErrorIdent	ERROR_STRUCT	For a detailed error description, see Error codes, page 370
	InPause	BOOL	It indicates that the "Pause" input is enabled and the values do not change at the outputs
	ActStep	UINT	Last processed step
	QualityData	IL_REGI_QUALITY	It contains the calculated quality data for the printing process
	GoodSample	BOOL	Is TRUE if the current printed image is of "good quality"
	MediumSample	BOOL	Is TRUE if the current printed image is of "medium quality"
	BadSample	BOOL	Is TRUE if the current printed image is of "bad quality"

Fig.7-107: Interface variables of the IL_RegiQualityType01 function block

IL_REGI_QUALITY		
GoodSampleCount	UDINT	Number of printed images of "good" quality
MediumSampleCount	UDINT	Number of printed images of "medium" quality
BadSampleCount	UDINT	Number of printed images of "bad" quality
GoodSampleRatio	REAL	Percentage of printed images with "good" quality
MediumSampleRatio	REAL	Percentage of printed images with "medium" quality
BadSampleRatio	REAL	Percentage of printed images with "bad" quality

Fig.7-108: Definition of structures used for IL_RegiQualityType01

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Reset	BOOL			FALSE	Continuous
Step	UINT	0	65535	0	Continuous
LimitGood	REAL	0.0	LimitMedium	1.0	Rising edge at "Enable"
LimitMedium	REAL	LimitGood	n.def	2.0	Rising edge at "Enable"
Value	ARRAY [1.0.16] OF REAL	n.def	n.def	0.0	If "Step" is changing

Fig.7-109: Min./max. and default values of the IL_RegiQualityType01 inputs

ML_TechRegi.library/MX_TechRegi.lib

Functional Description

The register errors of the printed image are an important characteristic for the quality of the printing process. The largest register error of a color mark is most important criteria for the quality of a printed image.

Based on the register deviation the printed image is subdivided into three quality categories ("Good","Medium","Bad"). The quality assessment is done with the inputs "LimitGood" and "LimitMedium". If the Inputs "LimitGood" and "LimitMedium" are equal, the printed image is only subdivided into two quality categories ("Good","Bad"). If the same value is specified for "LimitGood" and "LimitMedium", there is only a classification in "good" and "bad" (refer to the following figure).

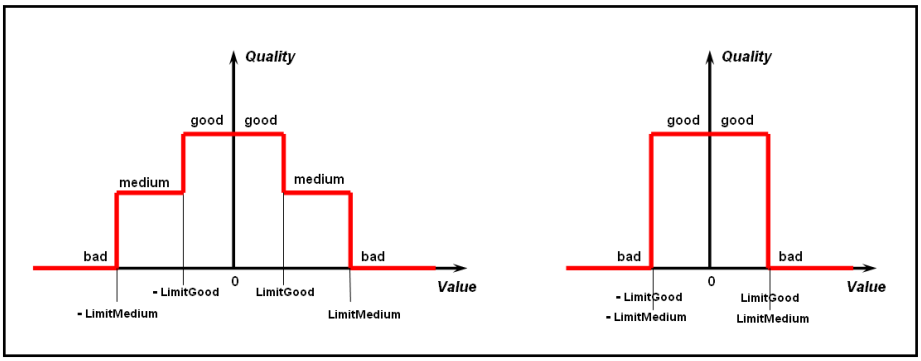


Fig.7-110: Quality assessment principle for the IL_RegiQualityType01

The output data structure "QualityData" sums up the printed images into one of the three quality categories. The structure also contains the statistical percentages for each quality category. All values are set to "0" at a rising edge of the "Reset" input.

Processing is carried out continuously when the **Enable** input is set. The "ActStep" output indicates that a new data package with measured values was received and evaluated. The Boolean outputs "Good/Medium/BadSample" indicate additionally the quality of the current printed image. If the "Pause" input is set, the current state is frozen, the "InPause" output is set and no further evaluation is carried out.

Error codes

The function block uses the error table **F_RELATED_TABLE**. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid state of the function block
INPUT_INVALID_ERROR	16#0817	16#0001	"LimitGood" value < 0.0
INPUT_INVALID_ERROR	16#0817	16#0002	"LimitMedium" value < "LimitGood"

Fig.7-111: Error codes of the IL_RegiQualityType01 function block

8 ML_TechCrank.library

8.1 Introduction and Overview

General

Technology function blocks (Tech FB) extend the basic functionality of the target system IndraMotion MLC and provide application-specific functionalities. The library described contains functionalities for crank and bell-crank kinematics.

The function blocks described are provided for the target system IndraMotion MLC via the "ML_TechCrank.compiled-library" library.

This documentation describes the functionality as well as the inputs and outputs of the technology function blocks.

Prerequisites

Technology function blocks of the library "ML_TechCrank.compiled-library" require the firmware support of the target system IndraMotion MLC. Specific requirements of the technology function blocks are documented in the chapter of the respective function blocks.

Overview on Function Blocks

The following table provides an overview on the available function blocks.

Function block	Description
Crank kinematics	
MB_CamTableCrank	Calculates the cam for transforming a translatory virtual master axis position into rotary values (crank angle)
MB_CamTableCrankSuperimposed	Superimposes a user cam with the transformation can for a superimposed cam.
MB_PhiToXvirt	Converts crank angle (Phi) into virtual translatory position values (Xvirt) to be used by the PLC program
MB_MasterToPhi	It displays crank angles calculated on the master axis position and the superimposed cam
MB_XvirtToXmech	It converts virtual translatory position values (Xvirt) into mechanic translatory position values (Xmech)
Bell-crank kinematics	
MB_BellCrankData	It calculates the Y-positions and crank angle Phi at the top and bottom pole
MB_CamTableBellCrank	It calculates a cam to transform a bell-crank drive
MB_CamTableBellCrankSuperimposed	Superimposes a user cam with the transformation can for a superimposed cam.
MB_PhiToYvirt	It converts the crank angle Phi into a mechanical and virtual translatory position
MB_YvirtToPhi	It converts the virtual translatory position into the crank angle Phi
MB_YvirtToPhiPoly5	It converts the virtual translatory position into the crank angle Phi using a substitute polynomial 4th order
MB_YvirtToYmech	It converts the virtual translatory position into the mechanical position
MB_YVirtToWorkRange	It returns the working range of crank 1

Fig.8-1: ML_TechCrank.compiled-library

8.2 Functions and Function Blocks for the Crank Kinematics

8.2.1 Introduction and Overview

The functions and function blocks described in this section convert translatory into rotary motions (crank angle) and vice versa, and are designed for crank kinematics applications.

Crank kinematics often are used to drive cross sealing shoes in bagging machines or forming and punching tools in thermoforming machines. The translatory carriage in this kinematics is moved by the rotary motion of a crank. he axis of the servo motor driving the crank is located opposite and displaced of the axis of the translatory slide.

The crank kinematics shown below outputs command and actual values (position, velocity etc.) in translatory units, even though the measurement system outputs in rotary units. Therefore, command and actual values have to be converted from translatory to rotary units and vice versa.

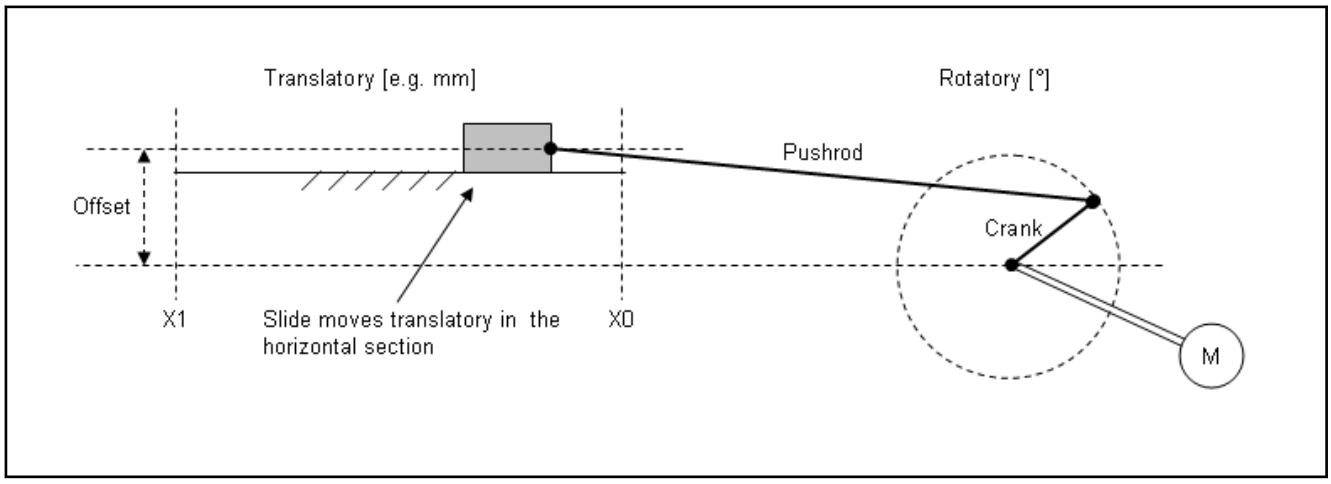


Fig.8-2: Single-axis crank kinematics with offset crank axis

The following functions and function blocks are supported:

Function	Description
MB_MasterToPhi	It displays crank angles calculated on the master axis position and the superimposed cam

Fig.8-3: Crank kinematics function

Function block	Description
MB_CamTableCrank	Calculates the cam for transforming a translatory virtual master axis position into rotary values (crank angle)
MB_CamTableCrankSuperimposed	Superimposes a user cam with the transformation can for a superimposed cam.

Function block	Description
MB_PhiToXvirt	Converts crank angle (Phi) into virtual translatory position values (Xvirt) to be used by the PLC program
MB_XvirtToXmech	It converts virtual translatory position values (Xvirt) into mechanic translatory position values (Xmech)

Fig.8-4: Crank kinematics function blocks

Typical use cases are:

Use case 1: Operation Mode Single Axis

The term single axis mode relates to the position or velocity-controlled operation of the translatory axis. In the present application, a virtual master axis (with translatory scaling) is moved by position or velocity control function blocks (e.g. MC_MoveAbsolute, MC_MoveRelative or MC_MoveVelocity).

The virtual master axis position is converted from translatory position values into crank angle values by the transformation cam. The transformation cam is calculated by the MB_CamTableCrank function block and activated with the MC_CamIn function block. Then, the crank axis follows the virtual master axis (issuing translatory units), at which the transformation cam converts the translatory position values into crank angle values. For detailed information, see chapter 8.2.3 "MB_CamTableCrank" on page 377.

Use case 2: Synchronous Operation with Cam

In this application, a translatory axis with a user-defined cam profile follows a master axis. The user-defined cam exclusively relates to the translatory axis. Thus, the transfer behavior of the crank does not have to be taken into account in the user-defined cam.

In contrast to that, the user defined cam is superimposed with a transformation cam by the MB_CamTableSuperimposed function block, and the resulting cam is transferred to the drive by the PLC program. For detailed information, see chapter 8.2.4 "MB_CamTableCrankSuperimposed" on page 380.

Reversed Transformation

In both the use cases above, translatory units are converted into rotary units (command value generation). Rotatory units have to be converted into translatory units to ensure that the PLC can subsequently process position and velocity commands. Therefore, the MB_PhiToXvirt function block is used. For detailed information, see chapter 8.2.5 "MB_PhiToXvirt" on page 383.

ML_TechCrank.library

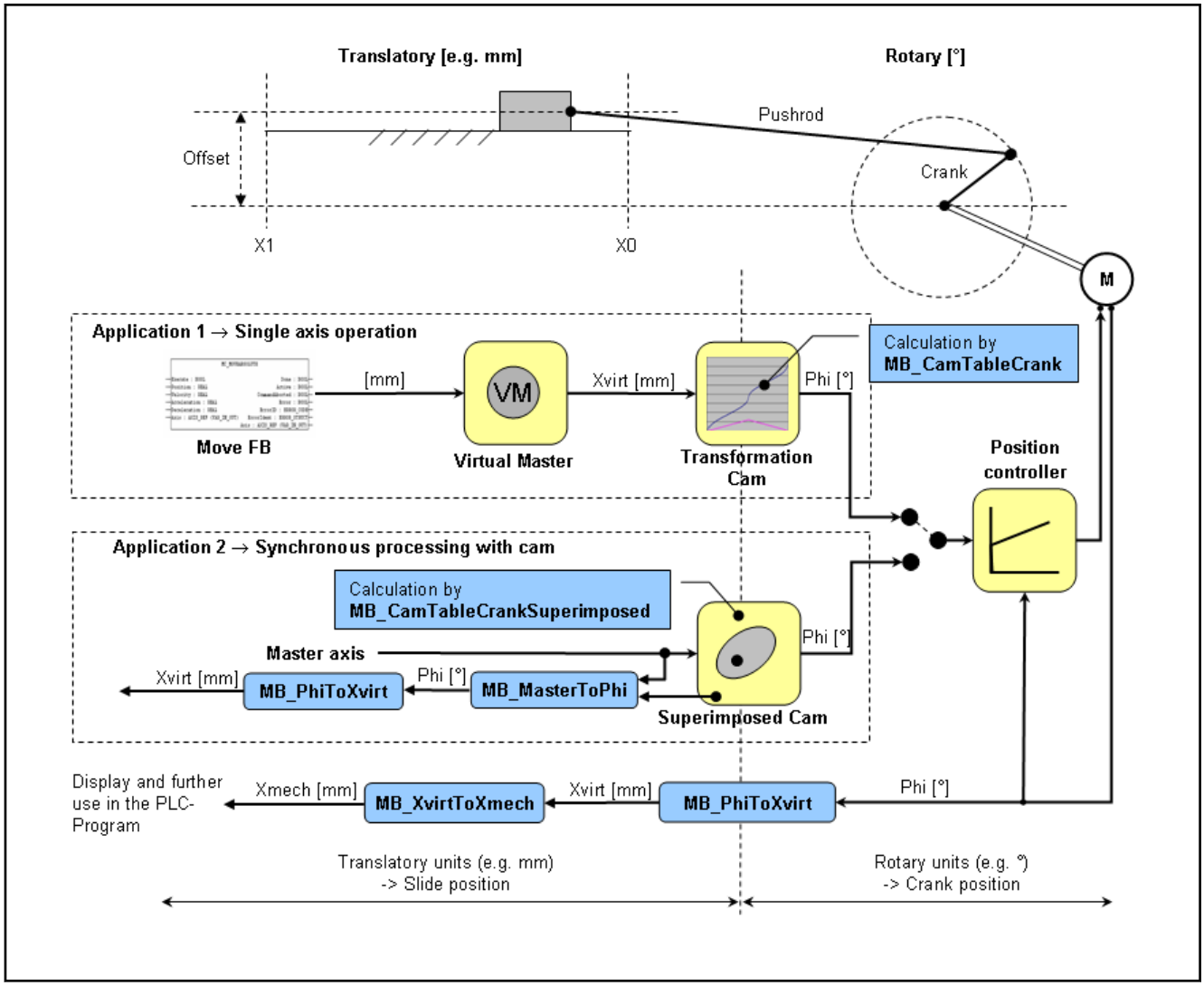


Fig.8-5: Schematic representation of the crank kinematics technology function blocks

8.2.2 General Definitions

Definition of the basic variables in the crank kinematics

The following illustrations depict the relations between crank and translatory coordinates, as well as the importance of different mechanical parameters.

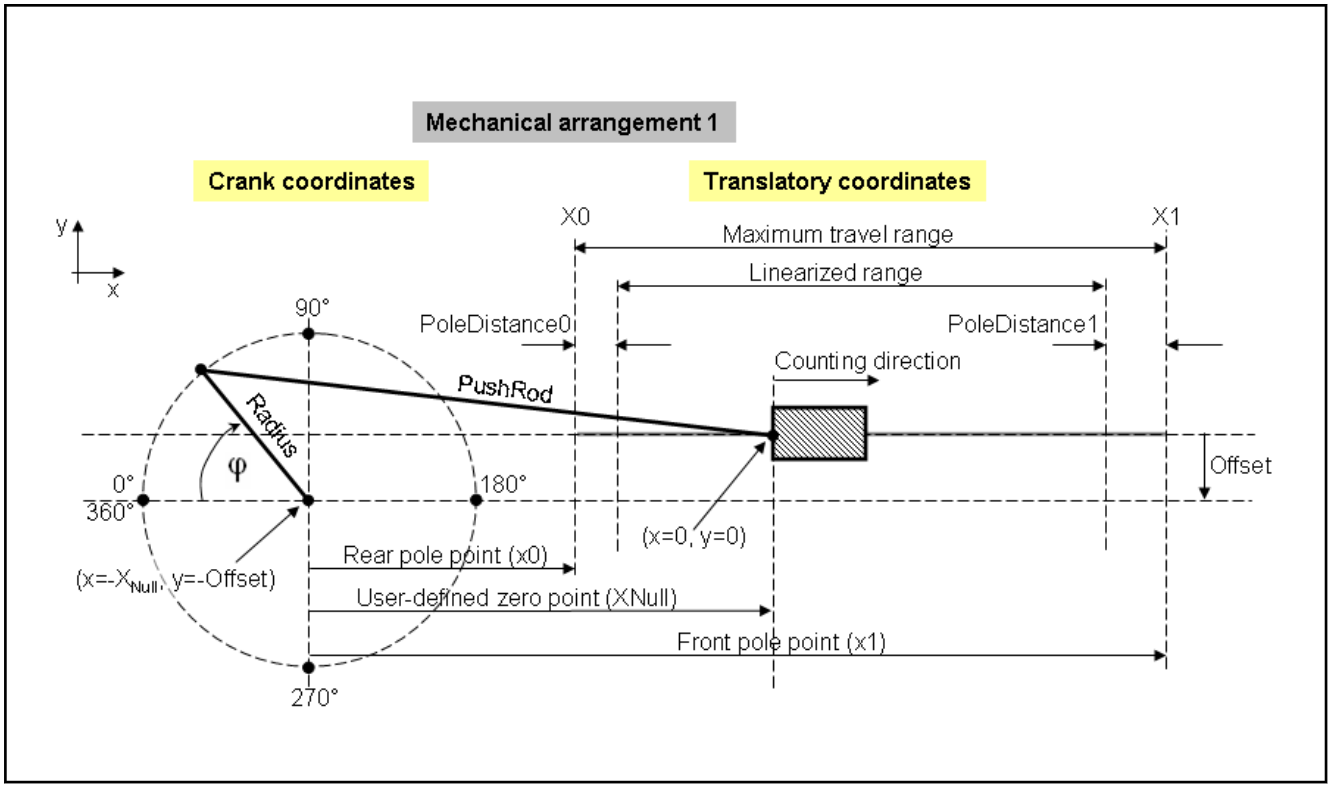


Fig.8-6: Mechanical arrangement 1: Relation of crank and translatory coordinates in case of crank kinematics

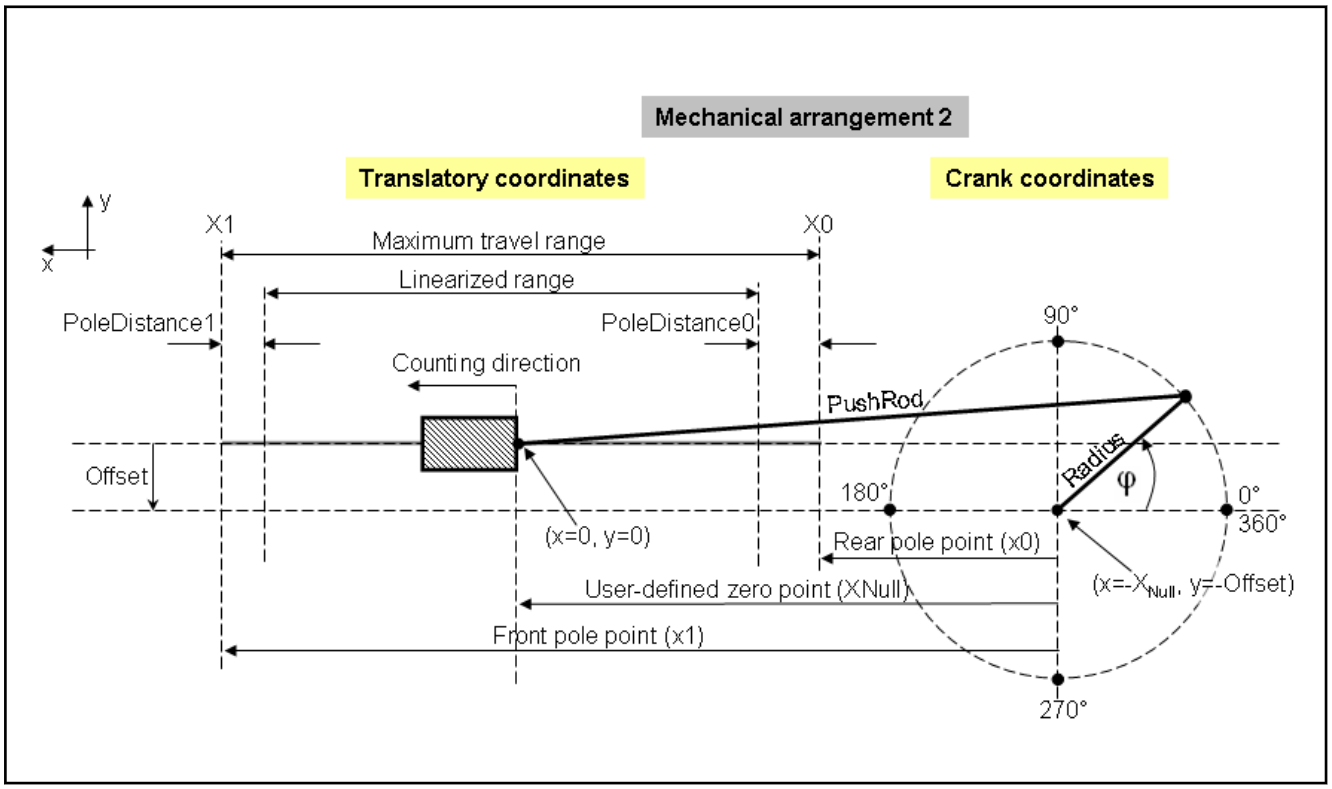


Fig.8-7: Mechanical arrangement 2: Relation of crank and translatory coordinates in case of crank kinematics

ML_TechCrank.library

Variable/term	Definition
Xzero	Distance between crank axis and slide zero point [mm]
PoleDistance0 PoleDistance1	Distance from the pole X0 / X1 (a compensation function is used outside the linearized area)
Maximum traveling range	Traveling range of the translatory coordinate system = |X1-X0|.
Linearized range	In the linearized range, no approximation polynom is effective
Transformation cam	Converts translatory (mm) coordinates into crank angles (degrees).

Fig.8-8: Variable and terms

Counting Direction

To the mechanical arrangements shown in fig. 8-6 " Mechanical arrangement 1: Relation of crank and translatory coordinates in case of crank kinematics" on page 375 and fig. 8-7 " Mechanical arrangement 2: Relation of crank and translatory coordinates in case of crank kinematics" on page 375, the following counting directions apply in general:

- Counting direction of the crank angle:
  - Mechanical arrangement 1: Clockwise, starting on the left with 0° and ending with 360°
  - Mechanical arrangement 2: Counterclockwise, starting on the right with 0° and ending with 360°
- Counting direction of the translatory slide:
  - Mechanical arrangement 1: Increasing numerical values to the right
  - Mechanical arrangement 2: Increasing numerical values to the left

Mechanical (Xmech) and Virtual (Xvirt) Translatory Position

Mechanically, the slide (translatory axis) can move between the rear (X0) and the front (X1) pole. Between these two limits, the "maximum traveling range" is situated. Every mechanical translatory position (Xmech) corresponds to two different crank angles. Thus, the virtual translatory position (Xvirt) assigns a unique translatory position to every crank angle. The virtual translatory position thus moves to positive direction, even though the mechanical position (Xmech) reverses the direction by exceeding the pole. The modulo overflow of Xvirt is defined by the user via the zero point (Xzero). Furthermore, the modulo value of Xvirt corresponds to the double traveling range of the crank kinematics. The traveling range is determined as follows:

$$\text{Travelrange} = (\text{FrontPolePoint} - \text{RearPolePoint})$$

$$\text{Travelrange} = \sqrt{(\text{Radius} + \text{PushRod})^2 - \mathit{Offset}^2} - \sqrt{(\mathit{Radius} - \mathit{PushRod})^2 - \mathit{Offset}^2}$$

Fig.8-9: Traveling range equation for the crank kinematic

The following illustration depicts Xvirt and Xmech in an example:

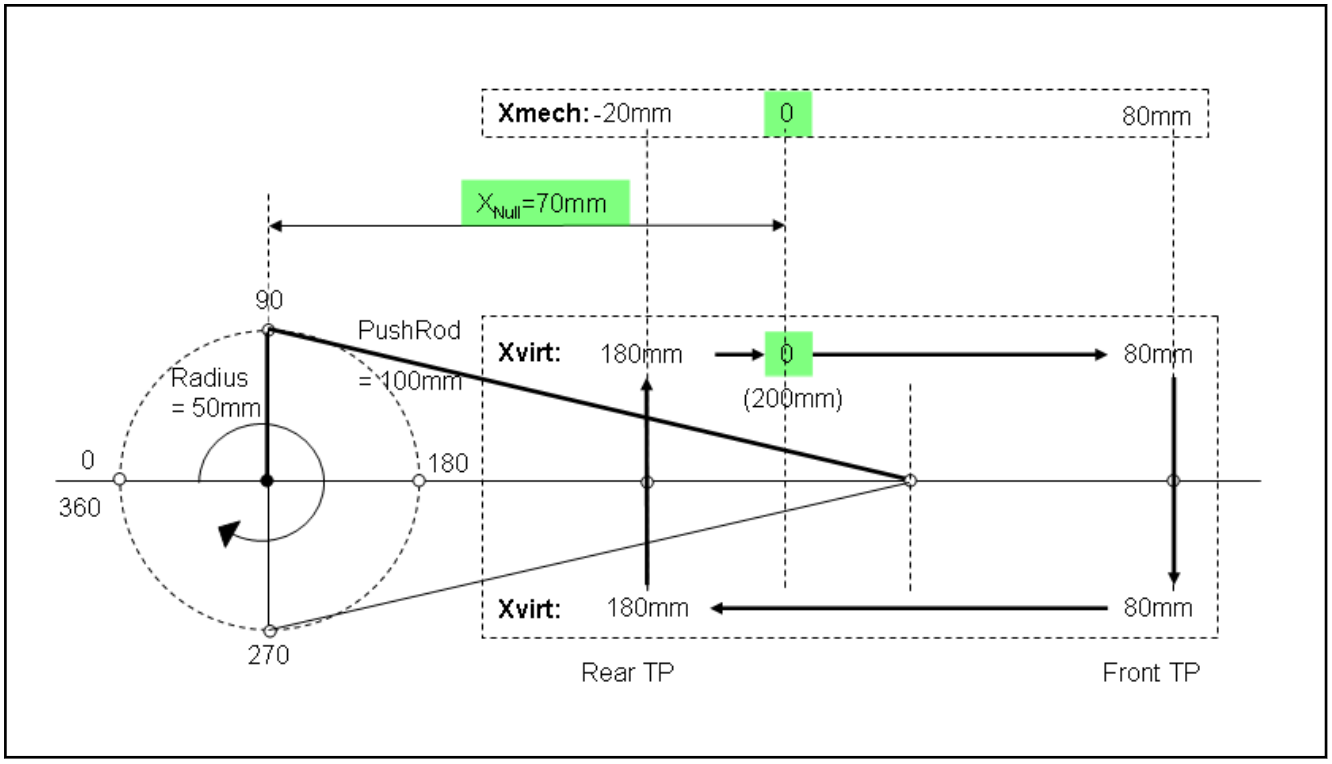


Fig.8-10: Counting direction of the slide position (Xvirt and Xmech)

8.2.3 MB_CamTableCrank

Brief Description

The function block MB_CamTableCrank calculates a transformation cam (with 1,024 data points) based on crank-specific input values. This transformation cam converts translatory position values into rotary position values (crank angle) to be used in a crank kinematics. It allows for coupling a translatory virtual master axis with a rotary crank drive. For detailed information, see chapter 8.2.1 "Introduction and Overview" on page 372.

The virtual axis in translatory units moves in the single axis mode while the drive of the transformation cam follows.

A motion extending the linear area can be executed by specifying a value for "PoleDistance0" and "PoleDistance1". Within the "PoleDistance", a compensation function (approximation polynomial 5th grade) approximating the crank drive and limiting the resulting drive dynamics. During transition in and from the linearized range, position, velocity and acceleration are constant.

The function block provides the calculated transformation cam using VAR_IN_OUT "CamTable". The PLC program has to load the generated cam table to the drive (e.g. via MB_WriteListParameter) to be used (e.g. by the function block MC_CamIn).

ML_TechCrank.library

Interface Description

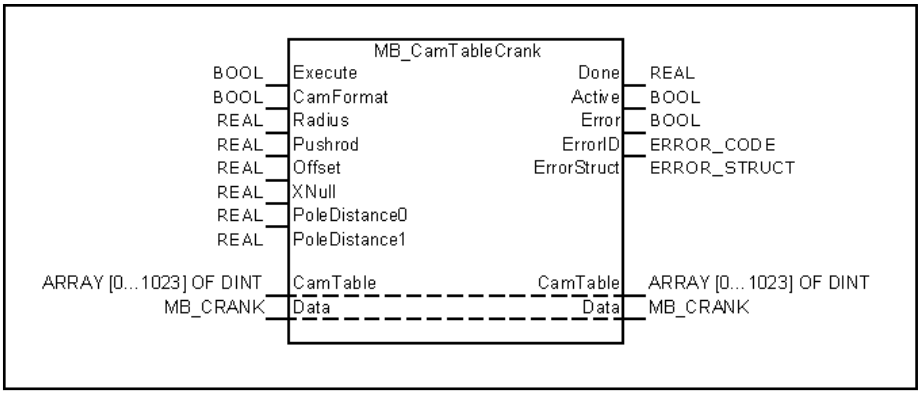


Fig.8-11: MB_CamTableCrank function block

I/O type	Name	Data type	Explanation
VAR_IN_OUT	CamTable	ARRAY [0...1023] OF DINT	Array with the data of the calculated transformation cam.
	Data	MB_CRANK	Structure with internally calculated data of the crank kinematics. The data is calculated in this function block and transferred to other function blocks
VAR_INPUT	Execute	BOOL	Calculation of the transformation cam is triggered by a positive edge.
	CamFormat	BOOL	TRUE = new format (last point = first point = 360°) FALSE = old format (last point = 360° - d)
	Radius	REAL	Crank length in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Offset	REAL	Offset of the slide level regarding the crank center [mm]
	Xzero	REAL	Distance between crank center and slide zero point [mm]
	PoleDistance0	REAL	Distance between the rear pole X0 [mm] and the linearized range, in which the movement is influenced by the limited drive dynamics
	PoleDistance1	REAL	Distance between the front pole X1 [mm] and the linearized range, in which the movement is influenced by the limited drive dynamics
VAR_OUTPUT	Done	BOOL	Calculation finished, cam table data and output data is valid
	Active	BOOL	Function block is being executed.
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description

Fig.8-12: Interface variables of the MB_CamTableCrank function block

Data Structure MB_Crank

The data structure MB_Crank is responsible for internal data exchange between function blocks and functions. The content of the data structure MB_CRANK is calculated by the function blocks MB_CamTableCrank and

MB_CamTableCrankSuperimposed and is applicable to all remaining relevant function blocks and functions.

Boundary condition

The following boundary conditions have to be met to prevent PLC error messages:

Boundary condition	Reason
Pushrod ≥ Radius + |Offset|	If this boundary condition is not met, the translatory slide cannot travel mechanically on a constant y-coordinate
(Xzero ≥ rear TP) & (Xzero ≤ rear TP + 2 x traveling range)	The zero point has to be in the traveling range

Fig.8-13: Boundary conditions

In order to be able to travel the translatory slide in single axis mode, the following configuration has to be conducted in IndraWorks:

- Translatory scaling of the virtual axis with a modulo value of (2 x traveling range).
- Rotary scaling of the crank drive (real axis) with a modulo value of 360°.

☞ In order to configure the scaling factors, IndraWorks has to be in online mode and the drive has to be in parameterizing mode.

Use case for MB_CamTableCrank

The following sequential chart applies to the single-axis mode:

1. The MB_CamTableCrank function block calculates the transformation cam.
2. The calculated transformation cam is written to the crank drive by the MB_WriteListParameter function block.
3. Using the MC_Power function block, the crank drive is activated.
4. The crank drive is referenced with the MC_Home function block. This step is necessary, if there is no absolute sensor at the crank drive engine.
5. Input of the actual crank drive position in the function block MB_PhiToXvirt. The actual crank drive position is converted from crank angle degrees to a virtual translatory value in mm.
6. To move the virtual master axis to the actual position of the crank drive, the output value Xvirt of the function block MB_PhiToXvirt has to be transferred to the MC_MoveAbsolute function block for the virtual axis. After this sequence, there is a virtual axis in the actual position of the crank drive and the crank drive is synchronized with the master axis without having executed a synchronization motion.
7. Switching the crank drive to the cam operating mode via MC_CamIn. Setting the CamShaftDistance = 360, gear ratio to 1:1, selecting the transformation cam and the virtual master axis as master axis. The crank drive does not move for synchronization with the virtual master axis, as the virtual master axis has been set to the crank position in step 6. If step 6 has not been conducted, the crank drive conducts a dynamic synchronization motion.
8. Now, the virtual axis can be moved in single axis mode using the function blocks MC_MoveAbsolute and MC_MoveVelocity (the crank axis now follows the master axis and in doing so uses the transformation cam).

ML_TechCrank.library

8.2.4 MB_CamTableCrankSuperimposed

Brief Description

The function block MB_CamTableSuperimposed superimposes the user-defined cam (CamInput) on the transformation cam and return the resulting superimposed cam via the "CamOutput" output. The transformation cam is calculated within the function block, comparable to the MB_CamTableCrank function block. The principle of superimposition is shown in fig. 8-14 " Principle of cam superimposition" on page 381 . The user defined cam has to contain 1,024 data points and has to define the movement of the translatory axis in relation to the master axis (irrespective of the crank kinematics). The 100 percent value of the table corresponds to the movement (2 x traveling range = modulo value of the virtual translatory axis).

The calculated superimposed cam has to be written on the crank drive using the MB_WriteListParameter function block and enabled by the PLC program via the function block MC_CamIn.

Notes for users

Depending on the corresponding end point of the user-defined cam, the following cases are distinguished:

- If the end point of the user-defined cam is near 100%, the crank does not reverse the direction (but rotations continually in the same direction). Energetically, this is the most economic procedure, as the natural movement of the crank is used.
- If the end point of the user-defined cam is 0%, a see-saw movement of the translatory axis takes place, during which the crank changes direction continually (see figure below).

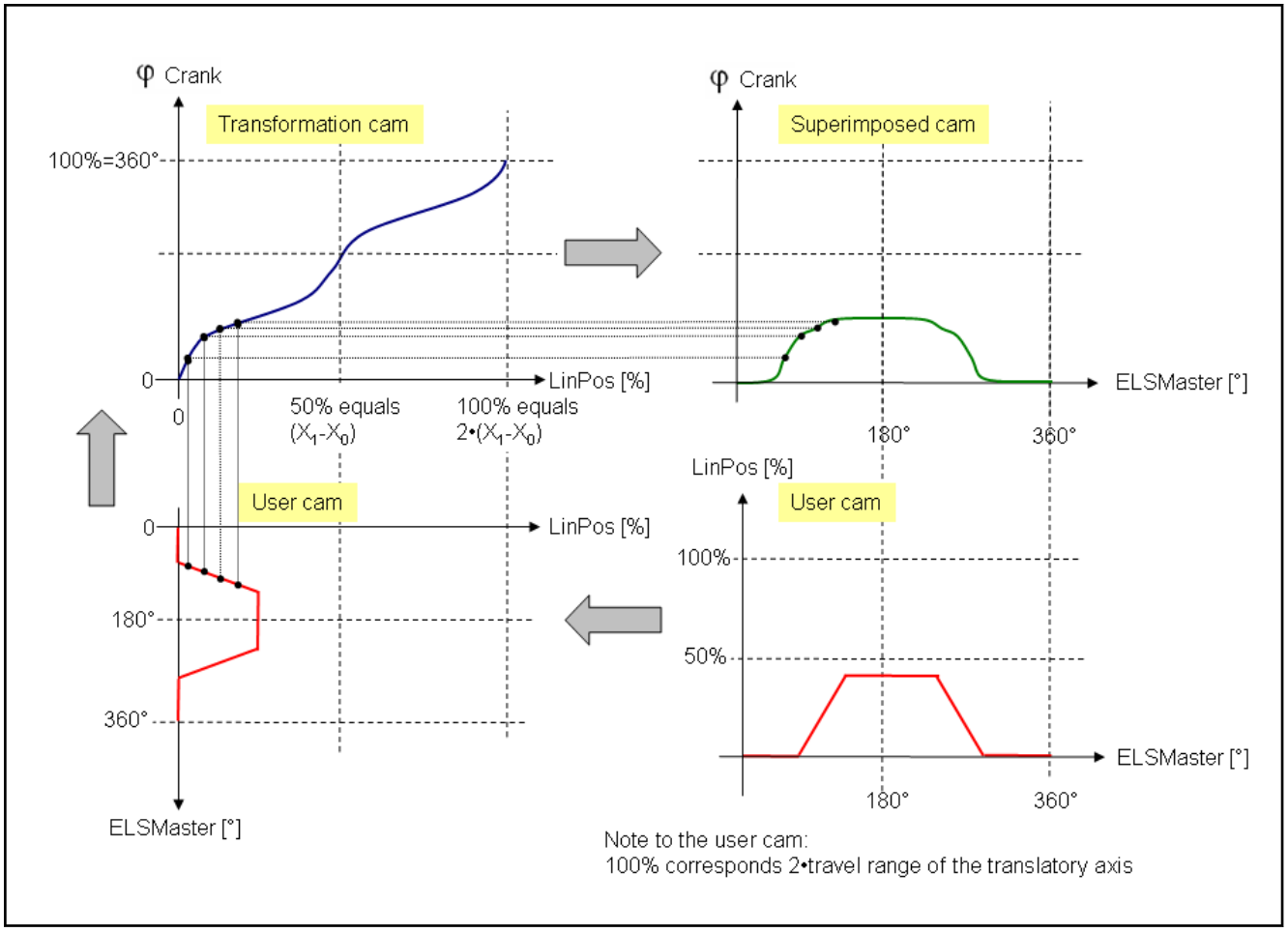


Fig.8-14: Principle of cam superimposition

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-15: Reference table of the MB_CamTableCrankSuperimposed

ML_TechCrank.library

Interface Description

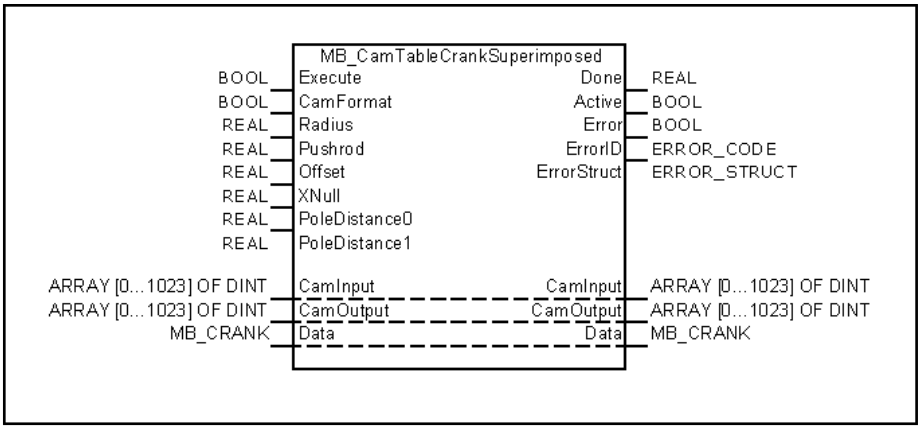


Fig.8-16: MB_CamTable_Superimposed function block

I/O type	Name	Data type	Description
VAR_IN_OUT	CamInput	ARRAY [0...1023] OF DINT	Array with data of the user-defined cam to preset the translatory motion profile
	CamOutput	ARRAY [0...1023] OF DINT	Array with data of the calculated superimposed cam to activate the crank.
	Data	MB_CRANK	Structure with internally calculated data of the crank kinematics. The data is calculated in this function block and then used by other function blocks .
VAR_INPUT	Execute	BOOL	Calculation of the superimposed cam disk is triggered by a positive pitch.
	CamFormat	BOOL	TRUE = new format (last point = first point = 360°) FALSE = old format (last point = 360° - d)
	Radius	REAL	Crank length in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Offset	REAL	Offset of the slide level regarding the crank center [mm]
	Xzero	REAL	Distance between crank center and slide zero point [mm]
	PoleDistance0	REAL	Distance between the rear pole X0 [mm] and the linearized range, in which the movement is influenced by the limited drive dynamics
	PoleDistance1	REAL	Distance between the back pole X1 [mm] and the linearized range, in which the movement is influenced by the limited drive dynamics
VAR_OUTPUT	Done	BOOL	Calculation finished, cam table data and output data is valid
	Active	BOOL	Function block is being executed.
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description

Fig.8-17: Interface variables of the ML_CamTableSuperimposed function block

Boundary condition

For detailed information, see "Boundary condition" on page 379.

In order to be able to travel the translatory slide in synchronous mode with a specified, user-defined cam, the following configuration has to be conducted in IndraWorks:

- Rotary scaling of the crank drive (real axis) with a modulo value of 360°.

In order to configure the scaling factors, IndraWorks has to be in online mode and the drive has to be in parameterizing mode.

Use Case Superimposed Cam

The following sequential chart applies to a cam-synchronous application:

1. Input of a user-defined cam using the input CamInput of the function block (e.g. by reading a drive cam or reading out from a file, etc.)
2. Output of a superimposed cam by the function block MB_CamTableCrankSuperimposed. It was calculated with the superimposition of a user-defined cam in a transformation cam (see fig. 8-14 " Principle of cam superimposition" on page 381).
3. The calculated superimposed cam is written to the crank drive by the MB_WriteListParameter function block.
4. The crank drive is activated using the MC_Power function block.
5. The crank drive is referenced with the MC_Home function block. This step is required if no absolute sensor is used.
6. The position command value of the superimposed cam and the current crank drive position can differ. There are two option to add the cam operation mode (via MC_CamIn, with the setting values CamShaftDistance=360, gear ratio 1:1 and the selection of the superimposed cam):
   - Adding the cam operation mode at the crank drive without previous position calibration. Then, the crank drive executes a dynamic synchronization.
   - Crank drive synchronization with the master axis by entering the master axis position and the superimposed cam in the MB_MasterToPhi function block. Before switching to synchronous mode, the crank drive has to be positioned with the MC_MoveAbsolute function block in a way that it corresponds to the output position of the MB_MasterToPhi function block.

8.2.5 MB_PhiToXvirt

Brief Description

The MB_PhiToXvirt function block converts crank angle values (Phi) to virtual translatory position values (Xvirt).

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-18: Reference table of the MB_PhiToXvirt

ML_TechCrank.library

Interface Description

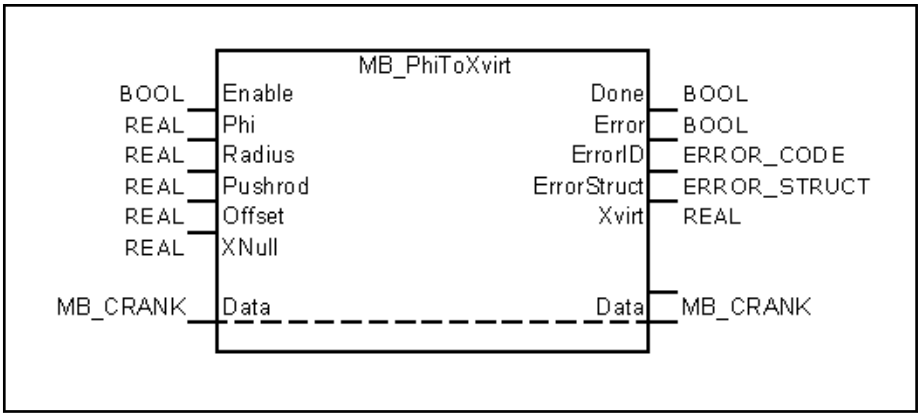


Fig.8-19: MB_PhiToXvirt function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Data	MB_CRANK	Structure with the internal data of the crank kinematics (to be calculated before MB_CamTableCrank or MB_CamTableCrankSuperimposed)
VAR_INPUT	Enable	BOOL	Calculation of Xvirt in every cycle, while Enable=TRUE
	Phi	REAL	Crank position in [°]
	Radius	REAL	Crank length in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Offset	REAL	Offset of the slide level regarding the crank center [mm]
	Xzero	REAL	Distance between crank center and slide zero point [mm]
VAR_OUTPUT	Done	BOOL	Calculation complete, Xvirt valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description
	Xvirt	REAL	Virtual translatory position (Xvirt)

Fig.8-20: Interface variables of the MB_PhiToXvirt function block

8.2.6 MB_MasterToPhi

Brief Description

The function MB_MasterToPhi provides the crank angle (Phi) calculated based on the master axis position (master value) and the (superimposition) cam. The superimposed cam has to be calculated before retrieving this function (by MB_CamTableCrankSuperimposed). The output value of the MB_MasterToPhi function is used to position the crank drive in such a way before switching to synchronous mode that the position corresponds to the position of the master axis.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib

IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-21: Reference table of the MB_MasterToPhi

Interface Description

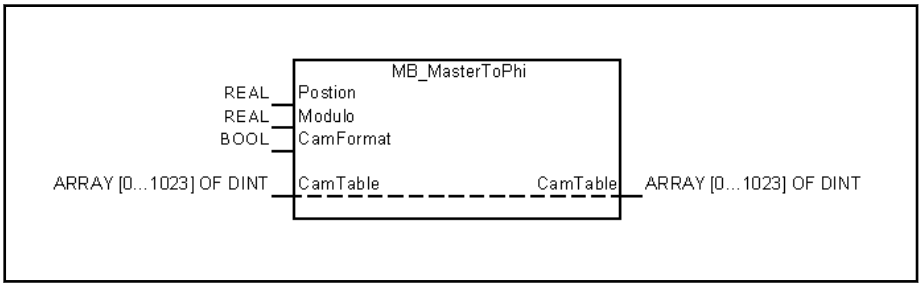


Fig.8-22: Function MB_MasterToPhi

I/O type	Name	Data type	Description
VAR_IN_OUT	CamTable	ARRAY [0...1023] OF DINT	Array with the data of the superimposed cam.
VAR_INPUT	Position	REAL	Master axis position
	Modulo	REAL	Modulo value of the master axis
	CamFormat	BOOL	TRUE = new format (last point = first point = 360°) FALSE = old format (last point = 360° - d)

Fig.8-23: Interface variables of the MB_MasterToPhi function

8.2.7 MB_XvirtToXmech

Brief Description

The MB_XvirtToXmech function block converts the virtual translatory position (Yvirt) to the mechanical translatory position (Ymech) of the carriage. Xmech values can be used for display purposes (e.g. on an HMI device).

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-24: Reference table of the MB_XvirtToXmech

ML_TechCrank.library

Interface Description

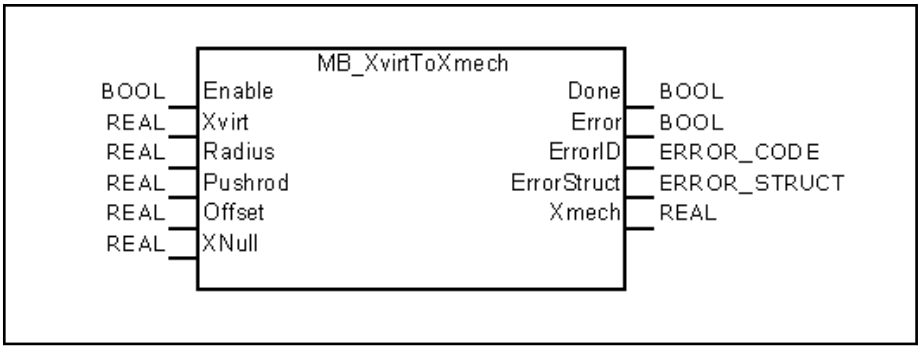


Fig.8-25: MB_XvirtToXmech function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	The function block calculates as long as "Enable" is TRUE
	Xvirt	REAL	Virtual translatory position in [mm]
	Radius	REAL	Crank length in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Offset	REAL	Offset of the slide level regarding the crank center [mm]
	Xzero	REAL	Distance between crank center and slide zero point [mm]
VAR_OUTPUT	Done	BOOL	Output value "Phi" valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description
	Xmech	REAL	Mechanical translatory position (Ymech) of the carriage

Fig.8-26: Interface variables of the MB_XvirtToXmech function block

8.3 Functions and Function Blocks for Bell-Crank Kinematics

8.3.1 Introduction and Overview

Bell-crank lever kinematics are used to drive forming tools for thermoforming machines or pressing machines for example. The following figure shows the mechanical structure of the bell-crank lever kinematics used below.

In case of this kinematics, turning crank 1 results in a translational movement in y direction. The crank is driven by the actuator. The measurement system is attached to the actuator and thus works in rotational units.

For the bell-crank lever kinematics shown, set points and actual values (such as position, velocity) are specified respectively displayed in translational units, although the measurement system works in a rotational manner. The Tech function modules described below conduct the corresponding transformations.

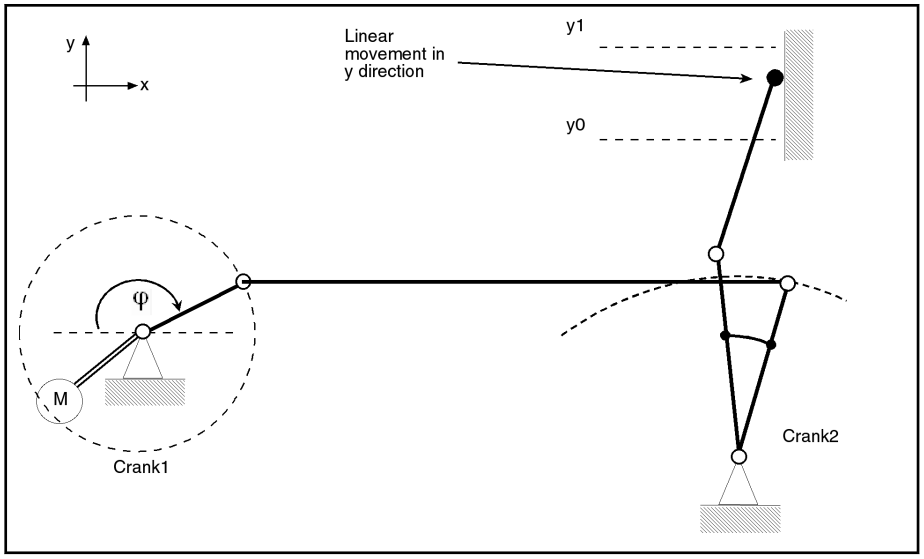


Fig.8-27: Bell-crank lever kinematics

8.3.2 General Definitions

Basic Variables

The following illustration shows the relation between crank and translatory coordinates, as well as the importance of the individual mechanical parameters.

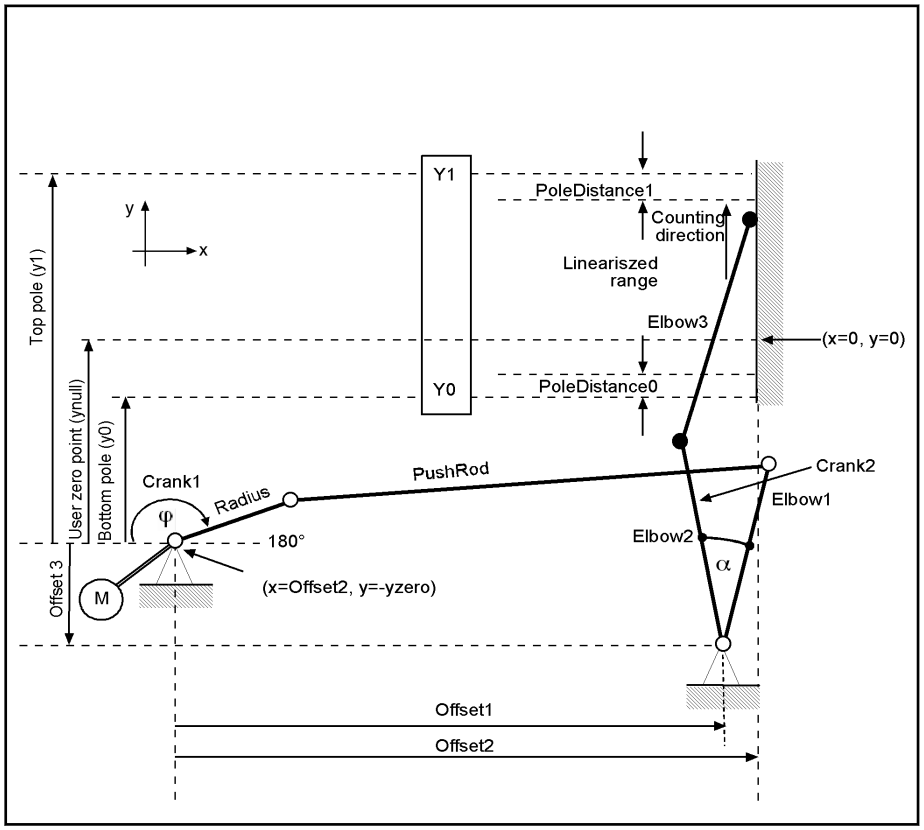


Fig.8-28: Relation between crank and translatory coordinates at the bell-crank kinematics

Counting Direction

Basically, the following counting directions are applicable to the arrangement shown in fig. 8-28 "Relation between crank and translatory coordinates at the bell-crank kinematics" on page 387:

- Counting direction of the crank angle:

  Clockwise, starting on the left with 0° and ending with 360°
- Counting mode of the translatory y position:

  Increasing numerical values upwards

Mechanical (Ymech) and Virtual (Yvirt) Translatory Position

A mechanical translatory position (Ymech) typically can be achieved via two different angular positions of crank 1. Thus, the mechanical translatory position (Ymech) cannot be assigned clearly to one crank angle. In order to solve this ambiguity, the virtual translatory position (Yvirt) is introduced, which continues counting in positive direction even if the direction of the translatory axis is reversed.

Using the zero point (Yzero) the user can specify where the mechanical zero point and thus the modulo overflow as well is located in the traveling range. In doing so, the modulo value of Yvirt has to correspond to the double traveling range of the translatory axis. For this, the traveling range (Sv) is defined as distance between top (Y1) and bottom (Y0) pole.

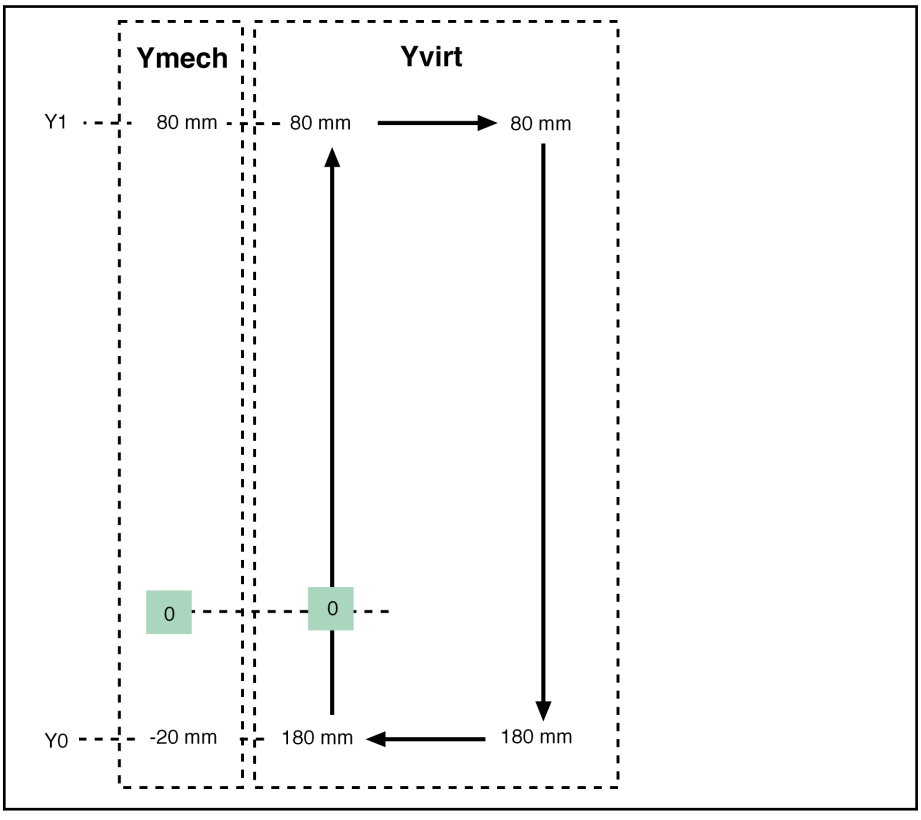


Fig.8-29: Relation between Yvirt and Ymech

Basic Principle of the Cam Transformation

The cams calculated by the function blocks of the bell-crank kinematics transform a master axis position into a crank angle of the bell-crank drive. Basically, the following two use case are distinguished:

- In the first case the real crank axis is coupled to a translationally scaled virtual axis using a transformation cam.

  In doing so, the virtual axis can be operated in all available operation modes in user-friendly translatory units (e.g. mm, mm/s) and the crank follows according to the transformation cam. In doing so, the necessary transformation cam is calculated by the "MB_CamTableBellCrank" function block. For the target system IndraMotion MLD, the virtual axis does not support any synchronous operation modes. That is why this use case is only reasonable to a limited extent.

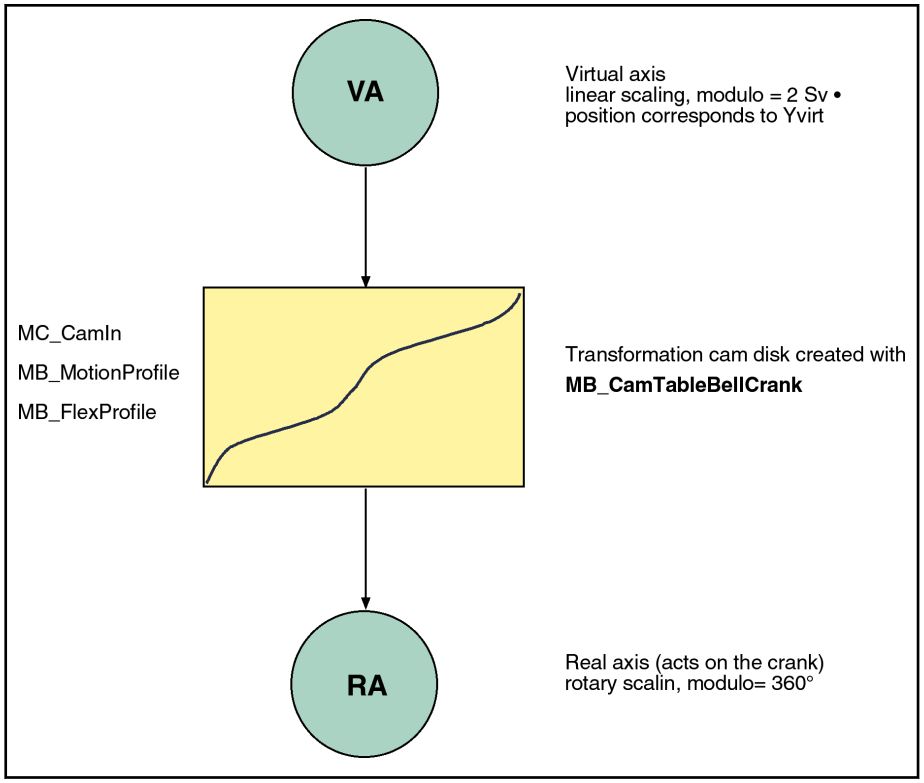


Fig.8-30: Mode of operation of the transformation cam with a translatory virtual axis

- In the second case, the real crank axis is coupled to any master axis with freely selectable modulo scaling using a superimposed user cam.

  The user cam describes the movement of the translatory axis in relation to the master axis, without taking the non-linear transmission behavior of the kinematics Thus, the "MB_CamTableBellCrankSuperimposed" function block superimposes the user cam with the transformation equations of the kinematics and calculates the superimposed cam. With this, the bell-crank drive with a user cam can be coupled to any master axis with freely selectable modulo value. This use case is particularly reasonable for the target system IndraMotion MLD, as the bell-crank drive can be coupled to a master axis. For the target system IndraMotion MLC this use case example has a lower significance. However, an additional virtual axis is not required.

ML_TechCrank.library

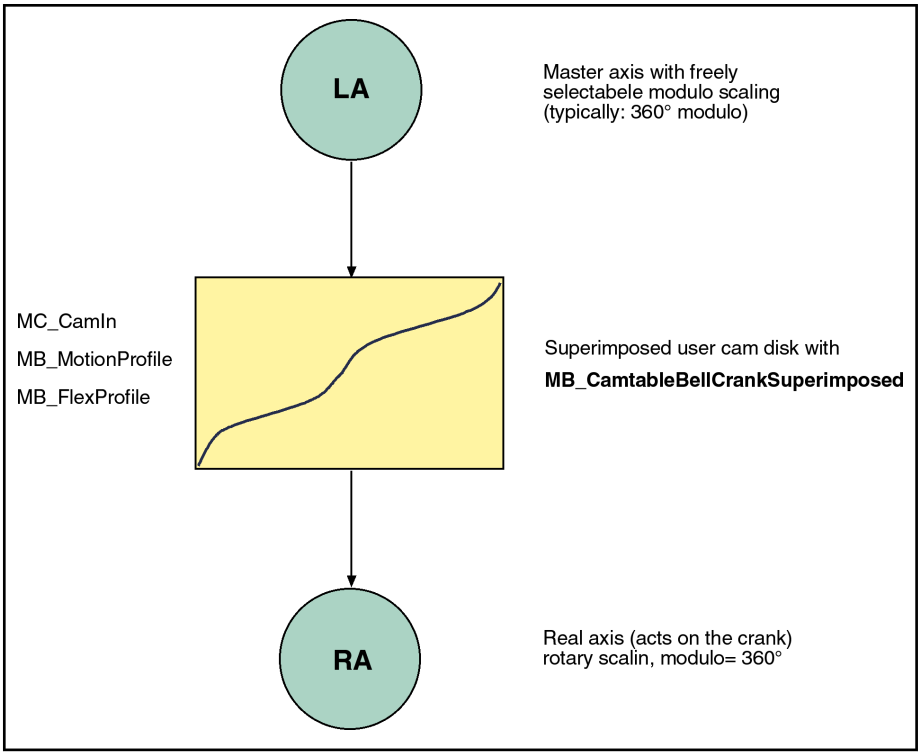


Fig.8-31: Mode of operation of the superimposed to any master axis

Simplifications and Basic Conditions

The following simplifications are specified and have to be met. The mechanical kinematic data (such as radius, pushrod) have to be selected accordingly.

- Both cranks reach their dead center simultaneously at the top pole (Y1). Thus, radius and pushrod as well as elbow2 and elbow3 form a straight line at the top pole
- The second crank moves only in working range 1. In working range 1, the mechanical position of the y-axis moves in positive direction if the second crank turns clockwise. However, the first crank can move in both working ranges and turn endlessly in one direction

8.3.3 MB_BellCrankData

Functional Description

The "MB_BellCrankData" function block calculates the y positions (Y0, Y1) and the crank angles Phi (Phi0, Phi1) at the bottom and top poles as well as the maximum travel range (Sv) of the y-axis.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-32: Reference table of the MB_BellCrankData

Interface Description

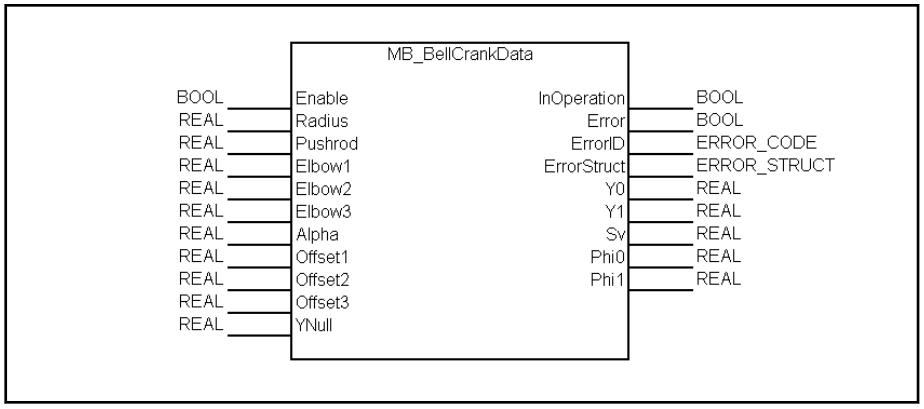


Fig.8-33: MB_BellCrankData function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Calculation of data in each cycle while Enable = TRUE
	Radius	REAL	Length of crank 1 in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
VAR_OUTPUT	InOperation	BOOL	Processing without errors, output data is valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description
	Y0	REAL	Mechanical y position of the bottom pole [mm]
	Y1	REAL	Mechanical y position of the top pole [mm]
	Sv	REAL	Maximum traveling range of the y-axis = ABS(Y1-Y0) [mm]
	Phi0	REAL	Angle Phi of crank 1 at the bottom pole [°]
	Phi1	REAL	Angle Phi of crank 1 at the top pole [°]

Fig.8-34: Interface variables of the MB_BellCrankData function block

Error Handling

The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-35: Error codes of the MB_BellCrankData function block

8.3.4 MB_CamTableBellCrank

Functional Description

The MB_CamTableBellCrank function block calculates the necessary transformation cam (with 1,024 points) for a bell-crank drive. The transformation cam serves for converting translatory units into the crank angle Phi. Thus, the rotary crank drive can be coupled to a translatory scaled virtual axis using the transformation cam.

The virtual axis can be moved in translatory units and the crank drive correspondingly follows the transformation cam

In order to travel through the poles as well, the distance to the poles (PoleDistance0 und PoleDistance1) can be defined. Within the PoleDistance, a compensation function (approximation polynom of 5th grade) approximating the bell-crank drive and limiting the resulting drive dynamics. During transition in and from the linearized range, position, velocity and acceleration are constant.

The function block provides the PLC program with the calculated cam using the input/output "CamTable".

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS without MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS without MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS without MA function package	MX_Technology_17.lib (in preparation)

Fig.8-36: Reference table of the MB_CamTableBellCrank

Interface Description

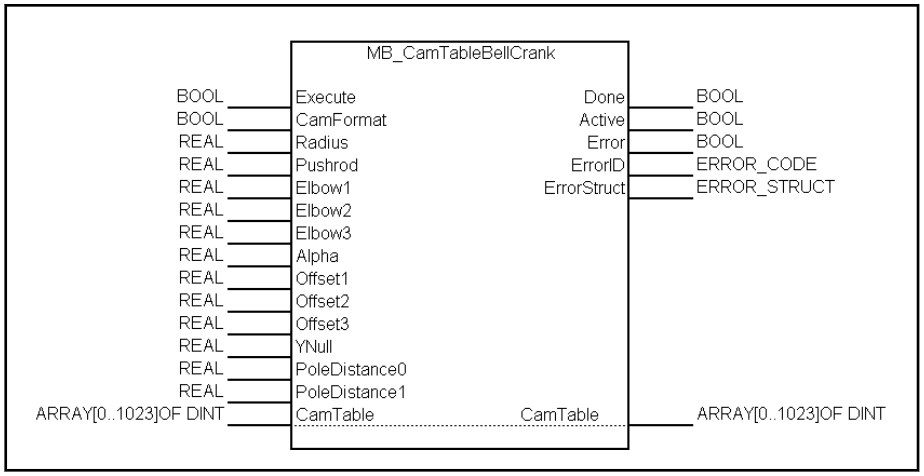


Fig.8-37: MB_CamTableBellCrank function block

.

ML_TechCrank.library

I/O type	Name	Data type	Description
VAR_IN_OUT	CamTable	ARRAY [0...1023] OF DINT	Array with the data of the calculated transformation cams
VAR_INPUT	Execute	BOOL	Positive edge starts calculation of the transformation cam
	CamFormat	BOOL	TRUE: New cam format -> last table point corresponds to first table point
	Radius	REAL	Length of crank 1 in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
	PoleDistance0	REAL	Distance from the bottom pole Y0 (in [mm]) to travel through the pole with limited drive dynamics. Within this area, a polynomial 5th order is used approximating the bell-crank kinematics and limiting the necessary drive dynamics
	PoleDistance1	REAL	Distance from the top pole Y1 (in [mm]) to travel through the pole with limited drive dynamics. Within this area, a polynomial 5th order is used approximating the bell-crank kinematics and limiting the necessary drive dynamics
VAR_OUTPUT	Done	BOOL	Calculation complete -> cam table valid
	Active	BOOL	Function block is processed
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description

Fig.8-38: Interface variables of the MB_CamTableBellCrank function block

Necessary boundary conditions and requirements

- Both cranks reach their dead center simultaneously at the top pole (Y1). Thus, radius and pushrod as well as elbow2 and elbow3 form a straight line at the top pole
- The second crank moves only in working range 1. In working range 1, the mechanical position of the y-axis moves in positive direction if the second crank turns clockwise. However, the first crank can move in both working ranges and turn endlessly in one direction

- The axis used for the cam (as master) has to be translatory and modulo-scaled.

  The modulo value is to be set as follows: Modulo value = 2 • traveling range (Sv). For this, the traveling range (Sv) is defined as distance between top (Y1) and bottom (Y0) pole.

  The travel range (Sv) can be determined using the MB_BellCrankData function block
- The crank drive has to be scaled rotationally and modulo 360°

Use case of MB_CamTableBellCrank

Using the following sequence, the translatory axis can be traveled in single axis mode:

- Calculation of the transformation cam using the MB_CamTableBellCrank function block
- Load calculated transformation cam to the crank drive (e.g. with MB_WriteListParameter if not already loaded)
- Activation of Power (MC_Power)
- Establishing the absolute dimensional reference, if no absolute sensor is used. Therefore, the drive command "Drive-controlled referencing" C0600 is used
- Positioning the virtual axis on the translatory position resulting from the crank angle. The resulting translatory position "Yvirt" can be determined from the crank angle (Phi) using the MB_PhiToYvirt function block. Afterwards, the virtual axis is set to the determined position
- Activating the cam operation mode at the crank drive (e.g. using MC_CamIn, with CamShaftDistance=360 ,1:1 gear and selection of the transformation cam calculated and loaded before). The crank drive does not perform any synchronizing motion, as the virtual axis already is in "Synchronous position"
- Now, the virtual axis can be traveled in single axis mode using the function blocks C_MoveAbsolute and MC_MoveVelocity or AxisInterface (the crank axis then follows the transformation cam). The virtual axis can be operated in a synchronous operation mode as well

The crank drive performs a dynamic synchronizing motion when the operation mode is activated, unless the virtual axis was set to the crank position before.

Error Handling

The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0006	PoleDistance0 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0007	PoleDistance1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0008	PoleDistance0+PoleDistance1 > traveling range
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-39: Error codes of the MB_CamTableBellCrank function block

8.3.5 MB_CamTableBellCrankSuperimposed

Functional Description

This function block superimposes the specified user cam (CamInput) with the transformation cam and provides the result via the input/output "CamOutput". The transformation cam is calculated within the function block according to the same rules as in the function block "MB_CamTableBellCrank". The principle of superimposition is shown in fig. 8-40 "Principle of cam superimposition" on page 397. The user cam has to contain 1,024 data points and defines the motion of the translatory axis related to the master axis (without taking the bell-crank kinematics into account). Thus, the table value 100% corresponds to movement 2 • traveling range = modulo value of the virtual translatory axis.

The calculated superimposed cam has to be loaded on the drive and used via the cam operation mode to be used by the PLC program.

Depending on the specified end point of the user cam, the following cases are distinguished:

- If the end point of the user cam is near 100%, crank 1 does not reverse the direction (crank 1 "rotates") -> energy-saving procedure, as natural motion of the crank is used.
- If the end point of the user cam is 0%, the translatory axis moves forwards and backwards with reversal of direction of crank 1 (as shown in fig. 8-40 "Principle of cam superimposition" on page 397)

Advanced Note

From MLC03VRS onwards, the virtual axis used supports synchronous operation modes. Thus, superimposing the user cam with the transformation cam typical for the MLC03 applications is only required if an additional virtual axis should not be used.

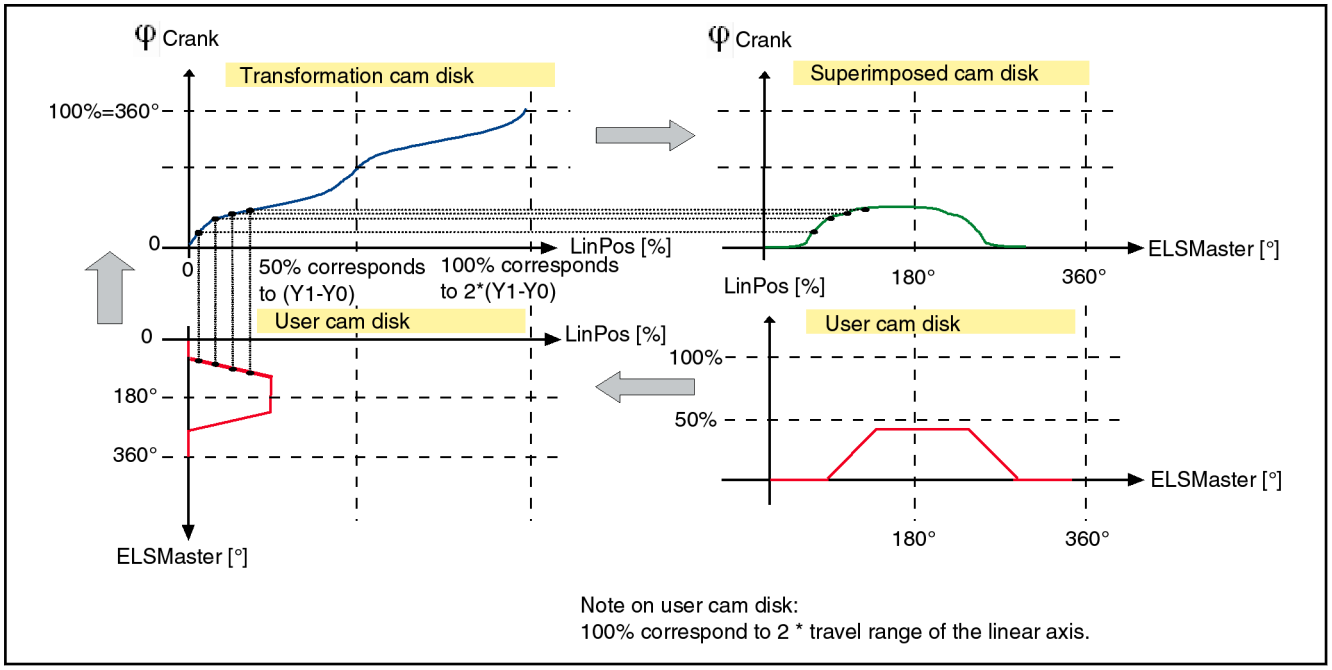


Fig.8-40: Principle of cam superimposition

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-41: Reference table of the MB_CamTableBellCrankSuperimposed

Interface Description

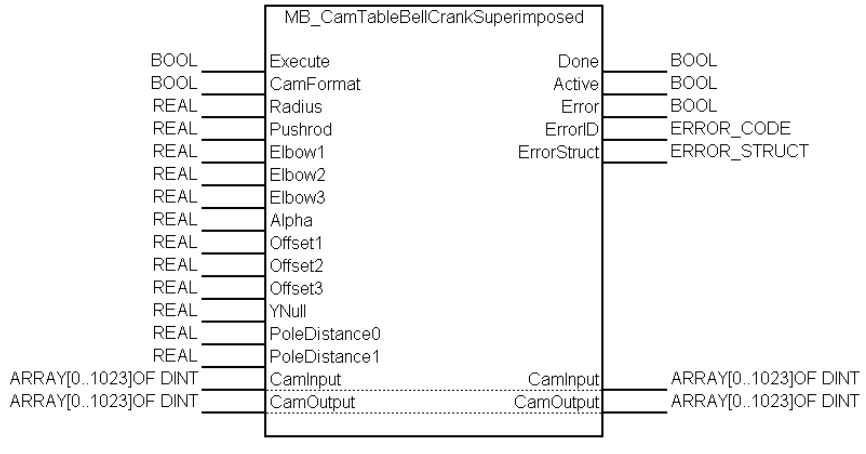


Fig.8-42: Function block MB_CamTableBellCrankSuperimposed

ML_TechCrank.library

I/O type	Name	Type	Comment
VAR_IN_OUT	CamInput	ARRAY [0...1023] OF DINT	Array with the data of the user cam to specify the translatory motion
	CamOutput	ARRAY [0...1023] OF DINT	Array with data of the calculated superimposed cam to actuate the crank axis
VAR_INPUT	Execute	BOOL	Positive edge starts calculation of the superimposed cam
	CamFormat	BOOL	TRUE: New cam format -> last table point corresponds to first table point
	Radius	REAL	Length of crank 1 in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
	PoleDistance0	REAL	Distance from the bottom pole Y0 (in [mm]) to travel through the pole with limited drive dynamics. Within this area, a polynomial 5th order is used approximating the bell-crank kinematics and limiting the necessary drive dynamics
	PoleDistance1	REAL	Distance from the top pole Y1 (in [mm]) to travel through the pole with limited drive dynamics. Within this area, a polynomial 5th order is used approximating the bell-crank kinematics and limiting the necessary drive dynamics
VAR_OUTPUT	Done	BOOL	Calculation complete -> output cam table valid
	Active	BOOL	Function block is processed
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorStruct	ERROR_STRUCT	Detailed error description

Fig.8-43: Interface variables of the MB_CamTableBellCrankSuperimposed function block

to use the superimposed cam

If the translatory axis is to be synchronized to the superimposed cam, the procedure is:

- The function block is provided with the user cam via input/output "CamInput" (e.g. by reading a drive cam, a file...)
- The superimposed cam is calculated by the MB_CamTableBellCrankSuperimposed function block
- Upon superimposition of the cam, the user program loads the superimposed cam "CamOutput" to the drive or the control system (MB_WriteListParameter)
- Activation of Power (MC_Power) at the crank drive
- Establishing the absolute dimensional reference of the crank axis if no absolute sensor is used. Therefore, the drive command C0600 "Drive-controlled referencing" has to be used
- The current crank position and the position command value from the superimposed cam can be different. Thus, the cam operation mode (e.g. via MC_CamIn, with CamShaftDistance= 360, 1:1 gearbox and selection of the superimposed cam) can be activated using the following 2 options:
  - Activation of the cam operation mode at the crank drive without prior position comparison -> The crank drive synchronizes dynamically
  - Crank axis is set to "Synchronous position" before activating the cam operation mode. Therefore, the crank angle from the superimposed cam can be determined using the function "MB_MasterToPhi". Before activating the synchronous operation mode, the crank drive can be set to this position. Afterwards, the superimposed cam can be activated without synchronizing motion

to scale the axes

- The master axis used has to be modulo-scaled (translatory or rotary).
- The axis for driving the crank has to be scaled rotationally and modulo 360°.

Necessary boundary conditions

See "Necessary boundary conditions and requirements" on page 394 of MB_CamTableBellCrank.

Error Handling

The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0006	PoleDistance0 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0007	PoleDistance1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0008	PoleDistance0+PoleDistance1 > traveling range
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-44: Error codes of the MB_CamTableBellCrankSuperimposed function block

8.3.6 MB_PhiToXvirt

Functional Description

The function block MB_PhiToYvirt converts the crank angle (Phi) to the mechanical and virtual translatory position (Ymech, Yvirt).

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-45: Reference table of the MB_PhiToYvirt

Interface Description

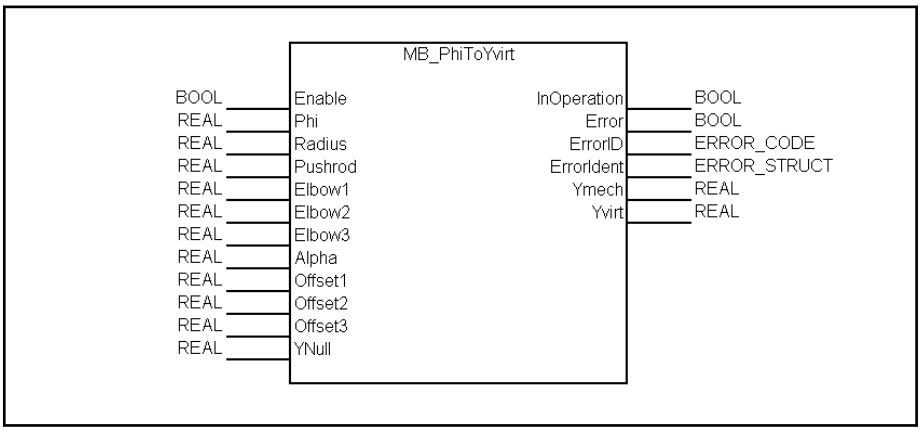


Fig.8-46: Function block MB_PhiToYvirt

I/O type	Name	Type	Comment
VAR_INPUT	Enable	BOOL	Calculation of Yvirt in every cycle as long as Enable=TRUE
	Phi	REAL	Crank position in [°]
	Radius	REAL	Length of crank in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
VAR_OUTPUT	InOperation	BOOL	Calculation complete -> Yvirt valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	Ymech	REAL	Mechanical translatory position [mm]
	Yvirt	REAL	Virtual translatory position [mm]

Fig.8-47: Interface variables of the MB_PhiToYvirt function block

Error Handling The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-48: Error codes of the MB_PhiToYvirt function block

8.3.7 MB_YvirtToPhi

Functional Description

The MB_YvirtToPhi function block converts the virtual translatory position (Yvirt) in the crank angle (Phi).

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-49: Reference table of the MB_YvirtToPhi

Interface Description

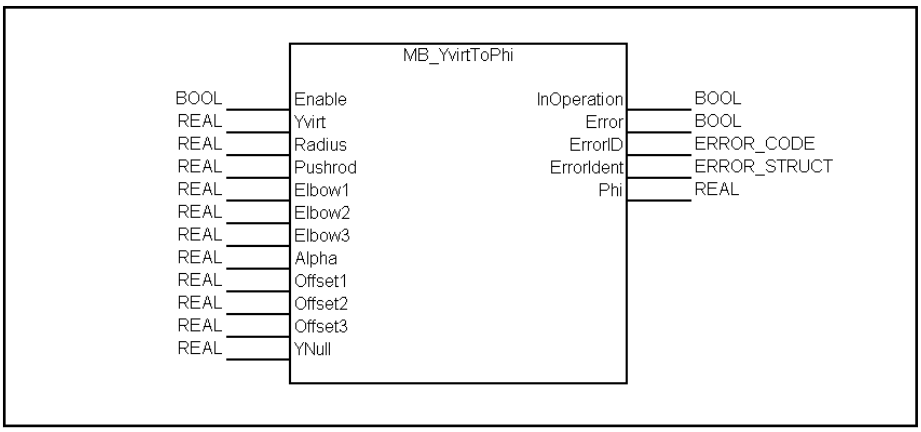


Fig.8-50: MB_YvirtToPhi function block

I/O type	Name	Type	Description
VAR_INPUT	Enable	BOOL	Calculation of Phi in every cycle as long as Enable=TRUE
	Yvirt	REAL	Virtual translatory position [mm]
	Radius	REAL	Length of crank in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
VAR_OUTPUT	InOperation	BOOL	Calculation complete -> Phi valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	Phi	REAL	Crank angle in [°]

Fig.8-51: Interface variables of the MB_YvirtToPhi function block

Error Handling The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-52: Error codes of the MB_YvirtToPhi function block

8.3.8 MB_YvirtToPhiPoly5

Functional Description

The MB_YvirtToPhiPoly5 function block converts the virtual translatory position (Yvirt) to the crank angle (Phi). Within the PoleDistance, a compensation function (approximation polynom of 5th grade) approximating the bell-crank drive and limiting the resulting drive dynamics. During transition in and from the linearized range, position, velocity and acceleration at crank 1 are constant.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-53: Reference table of the MB_YvirtToPhiPoly5

Interface Description

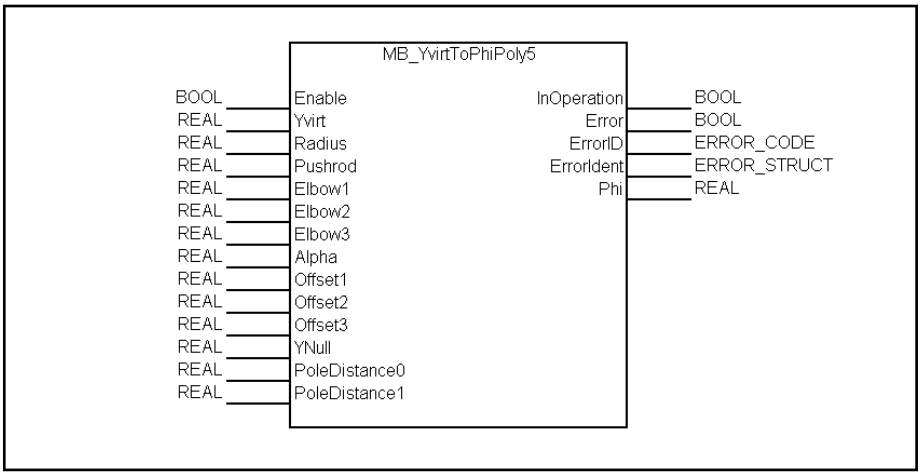


Fig.8-54: MB_YvirtToPhiPoly5 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Calculation of Phi in every cycle as long as Enable=TRUE
	Yvirt	REAL	Virtual translatory position [mm]
	Radius	REAL	Length of crank in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
	PoleDistance0	REAL	Distance from the bottom pole Y0 (in [mm]), in order to pass the pole with limited drive dynamics. Within this area, a polynomial 5th order is used approximating the bell-crank kinematics and limiting the necessary drive dynamics
	PoleDistance1	REAL	Distance from the top pole Y1 (in [mm]) to travel through the pole with limited drive dynamics. Within this area, a polynomial 5th order is used approximating the bell-crank kinematics and limiting the necessary drive dynamics
VAR_OUTPUT	InOperation	BOOL	Calculation complete -> Phi valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	Phi	REAL	Crank angle in [°]

Fig.8-55: Interface variables of the MB_YvirtToPhiPoly5 function block

Error Handling The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0006	PoleDistance0 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0007	PoleDistance1 <= 0

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0008	PoleDistance0+PoleDistance1 > traveling range
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-56: Error codes of the MB_YvirtToPhiPoly5 function block

8.3.9 MB_YvirtToYmech

Functional Description

The MB_YvirtToYmech function block converts the virtual translatory position (Yvirt) to the mechanical translatory position (Ymech). Ymech can be used for display purposes for example.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-57: Reference table of the MB_YvirtToYmech

Interface Description

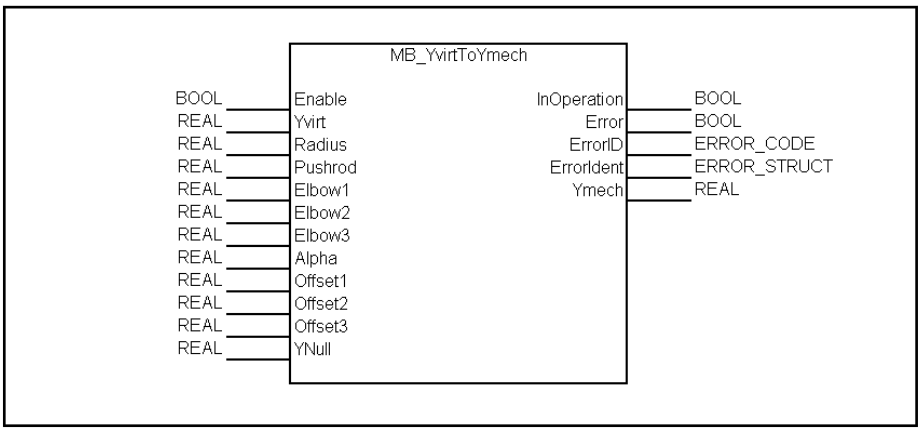


Fig.8-58: MB_YvirtToYmech function block

I/O type	Name	Type	Comment
VAR_INPUT	Enable	BOOL	The function block calculates as long as Enable=TRUE
	Yvirt	REAL	Virtual translatory position in [mm]
	Radius	REAL	Length of crank in [mm]
	Pushrod	REAL	Length of the pushrod in [mm]
	Elbow1	REAL	Length of the bell crank 1 [mm]
	Elbow2	REAL	Length of the bell crank 2 [mm]
	Elbow3	REAL	Length of the bell crank 3 [mm]
	Alpha	REAL	Angle between bell-cranks 1 and 2 [°]
	Offset1	REAL	X axis offset between crank and bell-crank pivot point [mm].
	Offset2	REAL	X-axis offset between crank pivot point and translatory motion [mm]
	Offset3	REAL	Y axis offset between crank and bell-crank pivot point [mm]
	Yzero	REAL	Distance crank pivot point to zero position of the translatory motion [mm].
VAR_OUTPUT	InOperation	BOOL	Output value (Ymech) valid
	Error	BOOL	Error (see ErrorID and ErrorStruct)
	ErrorID	ERROR_CODE	Error description
	ErrorIdent	ERROR_STRUCT	Detailed error description
	Ymech	REAL	Mechanical translatory position (Ymech) [mm]

Fig.8-59: Interface variables of the MB_YvirtToYmech

Error Handling The function block generates the following error messages in Additional1/Additional2 of the table "F_RELATED_TABLE", 16#0170:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#1000	16#0004	Radius <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0005	Pushrod <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#0009	Elbow1 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000A	Elbow2 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000B	Elbow3 <= 0
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000C	Alpha > 360° OR Alpha < -360°
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000D	Yzero < (Y0+Yzero)

ML_TechCrank.library

ErrorID	Additional1	Additional2	Description
INPUT_ RANGE _ERROR (16#0006)	16#1000	16#000E	Yzero > ((Y0+Yzero)+2*traveling range)
CALCULATION_ERROR (16#0007)	16x1002	16#xxyy	Internal calculation error, see Additional 2: xx = 16#01: Error caused by MB_BellCrankData xx= 16#02: Error caused by MB_PhiToYvirt xx = 16#03: Error caused by MB_YvirtToPhi xx = 16#04: Error caused by MB_YvirtToPhiPoly5

Fig.8-60: Error codes of the MB_YvirtToYmech function block

8.3.10 MB_YVirtToWorkRange

Functional Description

The function MB_YVirtToWorkRange provides the working range (1 or 2) of crank 1 as return value resulting from the virtual translatory position. In working range 1 the mechanical position of the y-axis moves in positive direction if crank 1 rotates clockwise. In working range 2 the mechanical position of the y-axis moves in negative direction if crank 1 rotates clockwise. Internally, this function is used by other function blocks.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechCrank.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology_07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.8-61: Reference table of the MB_YVirtToWorkRange

Interface Description

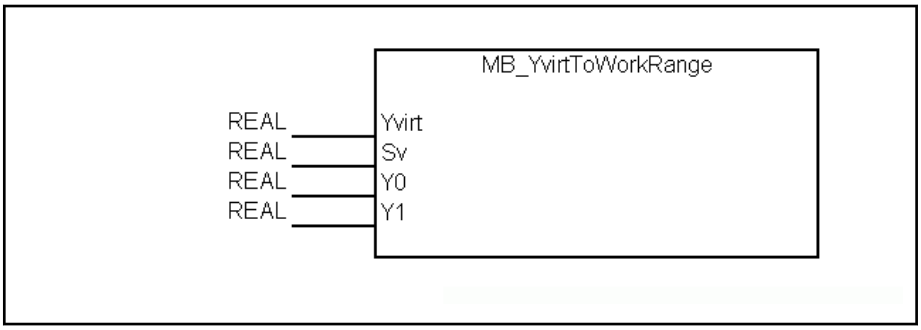


Fig.8-62: MB_YVirtToWorkRange function

I/O type	Name	Data type	Description
VAR_INPUT	Yvirt	REAL	Virtual translatory position [mm]
	Sv	REAL	Maximum traveling range of the y-axis = ABS(Y1-Y0) [mm]
	Y0	REAL	Mechanical y position of the bottom pole [mm]
	Y1	REAL	Mechanical y position of the top pole [mm]

Fig.8-63: Interface variables of the MB_YVirtToWorkRange function

Error Handling

The function does not generate any errors.

9 RMB_TechWinder.library/MX_TechWinder.lib

9.1 Introduction and Overview

Different function blocks are combined in the RMB_TechWinder.compiled-library libraries and MX_TechWinder.lib (IndraMotion MLD) summarizing different functionalities that provide functionalities for winding and unwinding fabric webs (winders).

With these function blocks, the functions of a

- dancer position controller
- diameter calculator with dancer
- diameter calculator with closed-loop tensile stress control and
- a diameter calculator with open-loop tensile stress control

can be implemented.

The function blocks required to implement these functionalities are described in the following.

The following table provides an overview on all the documented function blocks and their respective application.

Already connected complete winder function blocks

Function block	Description
MB_WinderDancerCtrlType01	Function block for winders with dancer contains among others: MB_DancerControlType03 and MB_DiameterCalculatorType03 MB_DiameterMeasurementType01
MB_WinderTensionCtrlType02	The function block for winders with closed-loop tensile stress control or open-loop tensile stress control (without sensor) contains among others: MB_DiameterCalculatorType03 MB_DiameterMeasurementType01
MB_WinderTensionCtrlSpeed-Type01	The function block for winders with closed-loop tensile stress control and speed correction contains among others: MB_DiameterCalculatorType03 MB_DiameterMeasurementType01

Fig.9-1: Already connected winder function blocks

The preconnected function blocks combine all functions of the individual function blocks and are responsible for the correct connection of the individual function blocks.

Function blocks for individual functionalities

RMB_TechWinder.library/MX_TechWinder.lib

Function block	Description
MB_DancerControlType03	Function block to calculate a closed-loop dancer position control for a tensile axis with dancer or winding axis
MB_DiameterCalculatorType03	Function block to calculate the current diameter for winding axes with velocity adaptation
MB_DiameterMeasurementType01	Function block to read a measured diameter for winding axes with velocity adaptation.

Fig.9-2: Single function blocks

Function blocks for additional winding functions

Function block	Description
MB_CalcInertiaLimitType02	Function block to calculate the maximum moment of inertia of the winding material on a winder
MB_UnwindMaterialType02	Function block to calculate the remaining length, the remaining time...of the fabric web for an unwinder
MB_WindTaperProfileType01	Function block to calculate a winding tightness profile for a winder with closed-loop or open-loop tension control
MB_WindSpeedControlAdaption-Type01	Function block for an automatic adaptation of the K_p-gain of the speed controller

Fig.9-3: Function blocks for additional functionalities

Library dependencies

IndraMotion MLC control - RMB_TechWinder

The required libraries are automatically included.

IndraMotion MLD control - MX_TechWinder

The following libraries are required:

- MX_Technology0x (MPX06 drive firmware) or
- RIL_LoopControl

Function packages for IndraMotion MLD

For IndraMotion MLD, the drive function package MA "IndraMotion MLD Advanced" is required for the function blocks of the "MX_TechWinder library" .

This function package is to be ordered separately with the respective drive firmware.

9.2 Function Block for Closed-Loop Dancer Position Control

9.2.1 Introduction and Overview

The dancer position controller is used for example on tensile axes or winders with dancers. The dancer position controller controls the position of a dancer. The tension is specified by a force acting upon the freely movable part of the dancer. This force can be applied by a weight, a spring or a pneumatic element. The actual value of the dancer position is recorded via an analog channel, applied to the control and processed in the process controller. The following figure shows the principle of closed-loop dancer position control.

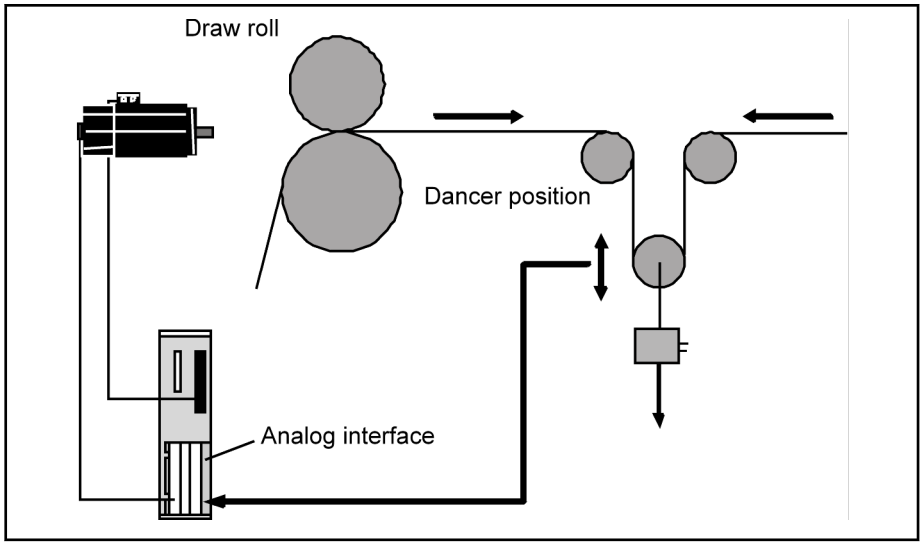


Fig.9-4: Principle of a closed-loop dancer position control

The dancer position controller is implemented in the function block MB_DancerControlType03 and designed as PID controller. By setting the controller parameters, the individual controller functions (P-value: proportional value, I-value: integral value, D-value: differential value) can be controlled. If a parameter is set to zero, the respective controller value is also not effective.

The following table shows the possible and permitted controller configurations.

No.	P-value	I-value	D-value	Permitted	Remark
1	0	0	0	Yes	Controller ineffective, output = 0.0
2	0	0	1	No	D-controller not defined, error
3	0	1	0	Yes	Exclusive I-controller
4	0	1	1	No	DI-controller not defined, error
5	1	0	0	Yes	Exclusive P-controller
6	1	0	1	Yes	PD-controller
7	1	1	0	Yes	PI-controller
8	1	1	1	Yes	PID-controller

Fig.9-5: Permitted and prohibited controller configurations (1 corresponds to the value set (value ≥ 0.0), 0 corresponds to the value not set (value = 0.0)

Functional diagram of the dancer position controller

The following functional diagram schematically illustrates the mode of operation of the dancer position controller.

RMB_TechWinder.library/MX_TechWinder.lib

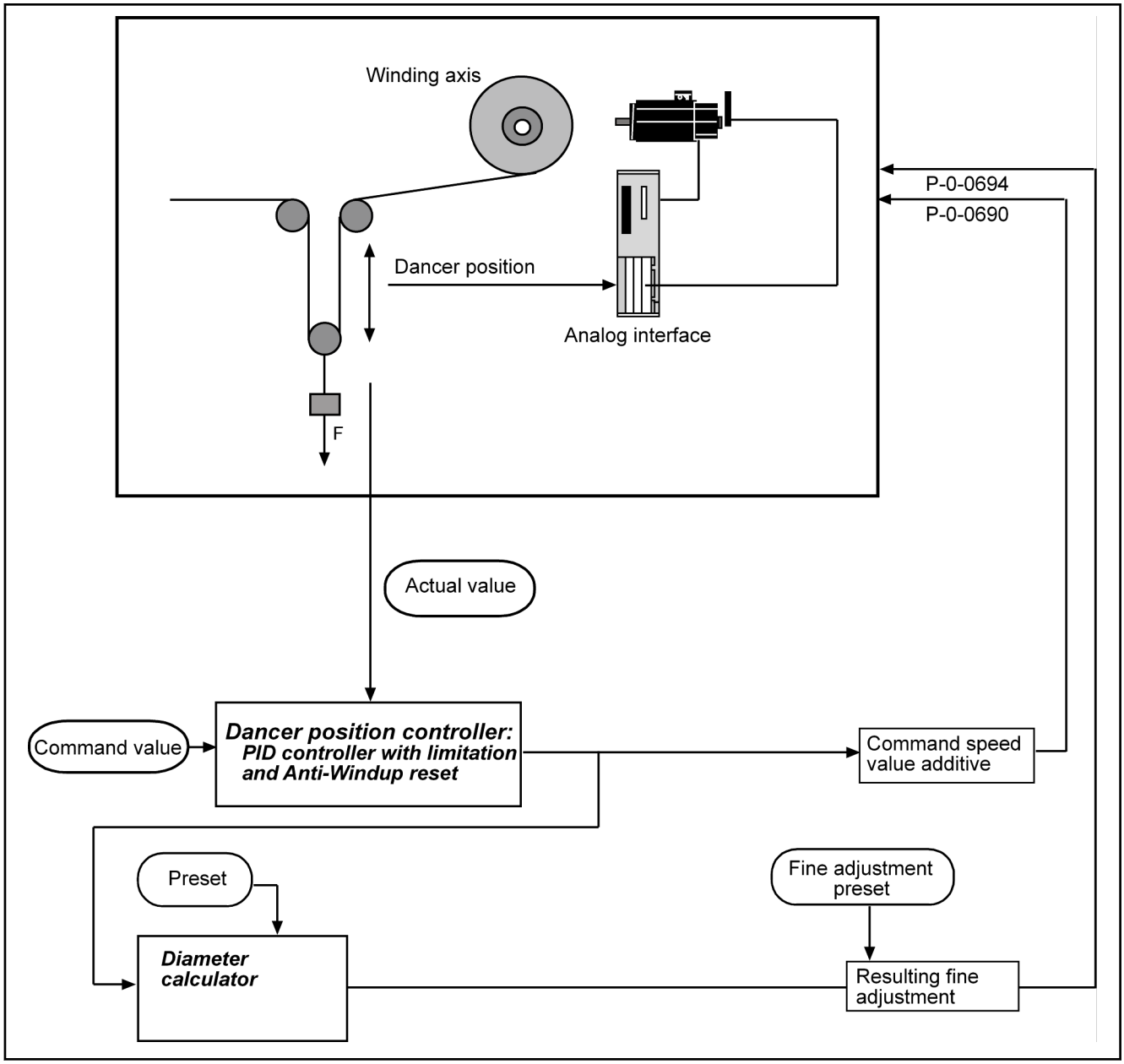


Fig.9-6: Functional diagram of the dancer position controller

9.2.2 MB_DancerControlType03

Brief Description

The function block calculates a closed-loop dancer position control for a pull roll with a dancer or winding axis with a dancer on a cyclic basis. The cyclic output variable of the dancer position controller is the speed offset of the respective drive of a pull roll or winding axis. The current control deviation of the dancer position controller is output.

The output "AddVelocityDancer" of the function block acts on the parameter P-0-0690 "Velocity command value, additive, process controller". If this is not desired, it can be prevented using the input "DisableCyclicWrites". It is only written to the "AddVelocityDancer" output.

RMB_TechWinder.library/MX_TechWinder.lib

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechWinder.lib

Fig.9-7: Reference table of the MB_DancerControlType03

Interface Description

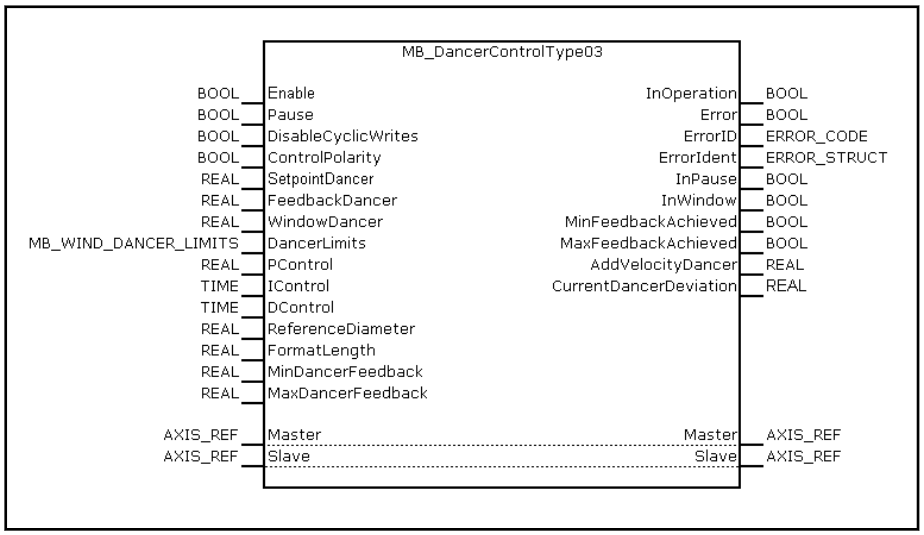


Fig.9-8: Function block MB_DancerControlType03

I/O type	Name	Type	Comment
VAR_IN_OUT	Master	AXIS_REF	Reference to master axis
	Slave	AXIS_REF	Reference to slave axis (tensile axis)
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	Pause	BOOL	Dancer controller paused, no velocity adaptation (level-controlled)
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output
	ControlPolarity	BOOL	Polarity of the closed-loop dancer control is inverted (applied in case of pos. edge at "Enable")
	SetpointDancer	REAL	Command value (setpoint) for closed-loop dancer control
	FeedbackDancer	REAL	Actual value (feedback) for dancer position
	WindowDancer	REAL	Windows for monitoring the actual dancer position value [%] with regard to the command dancer position value. If the dancer position deviates from the command value by more than the given value, the function block reports this deviation by resetting the "InWindow" output
	DancerLimits	MB_WIND_DANCER_LIMITS	Structure for the automatic control variable limitation of the dancer
	PControl	REAL	Value for the Kp-gain of the closed-loop dancer control

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	IControl	TIME	Value for the integral action time of the closed-loop dancer control
	DControl	TIME	Value for the derivative time of the closed-loop dancer control
	ReferenceDiameter	REAL	Reference diameter (value to standardize PControl and the control variable limitation "DancerLimits". That means that the values are internally adapted) For an exclusive dancer position controller: Pull roll diameter [mm] For winders with dancer: Assign current winding diameter [mm] from the diameter calculator function block
	FormatLength	REAL	Format length [mm] (value to standardize PControl) (1 revolution of the reference axis unwound)
	MinDancerFeedback	REAL	Lower threshold for the actual dancer value (only for display with a binary output).
	MaxDancerFeedback	REAL	Upper threshold for the actual dancer value (only for display with a binary output).
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	InPause	BOOL	Dancer controller pausing
	InWindow	BOOL	Actual dancer value within dancer window
	MinFeedbackAchieved	BOOL	Actual value of the dancer position is below the input "MinDancerFeedback"
	MaxFeedbackAchieved	BOOL	Actual value of the dancer position is above the input "MaxDancerFeedback"
	AddVelocityDancer	REAL	Speed offset for the winding drive or draw roll drive [rpm]
	CurrentDancerDeviation	REAL	Current control deviation of the dancer controller

Fig.9-9: Interface variables of the MB_DancerControlType03 function block

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
ControlPolarity	BOOL			FALSE	Rising edge at "Enable"

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective
SetpointDancer	REAL	0.0	n.def	0.0	Continuous
FeedbackDancer	REAL	n.def	n.def	0.0	Continuous
WindowDancer	REAL	0.0	n.def	0.0	Continuous
DancerLimits	STRUCT				
PControl	REAL	0.0	n.def	0.0	Continuous
IControl	TIME	0 s	n.def	0 s	Continuous
DControl	TIME	0 s	n.def	0 s	Continuous
ReferenceDiameter	REAL	0.0	n.def	100	Continuous
FormatLength	REAL	0.0	n.def	0.0	Rising edge at "Enable"
MinDancerFeedback	REAL	0.0	MaxDancerFeedback	0.0	Continuous
MaxDancerFeedback	REAL	MinDancerFeedback	n.def	0.0	Continuous

Fig.9-10: Min./max. and default values of the MB_DancerControlType03 inputs

Name	Type	Min. value	Max. value	Default value	Effective	Description
LowLimitControl	REAL	0.0	n.def	0.0	Continuous	Lower limitation of the controller output (at a speed of 0) bipolar [rpm] (the value is internally standardized with the diameter ratio of "ReferenceDiameter" and "RepeatLength")
HighLimitControl	REAL	0.0	n.def	0.0	Continuous	Upper limitation of the controller output (at a speed of VelocityHighLimit) bipolar [rpm] (the value is internally standardized with the diameter ratio of "ReferenceDiameter" and "RepeatLength")
VelocityHighLimit	REAL	0.0	n.def	0.0	Continuous	Speed of the master for HighLimitControl [rpm]

Fig.9-11: Interface description of the data structure of the MB_WIND_DANCER_LIMITS

The data structure MB_WIND_DANCER_LIMITS summarized in the table describes the limiters of the controller control variable shown in the diagram.

RMB_TechWinder.library/MX_TechWinder.lib

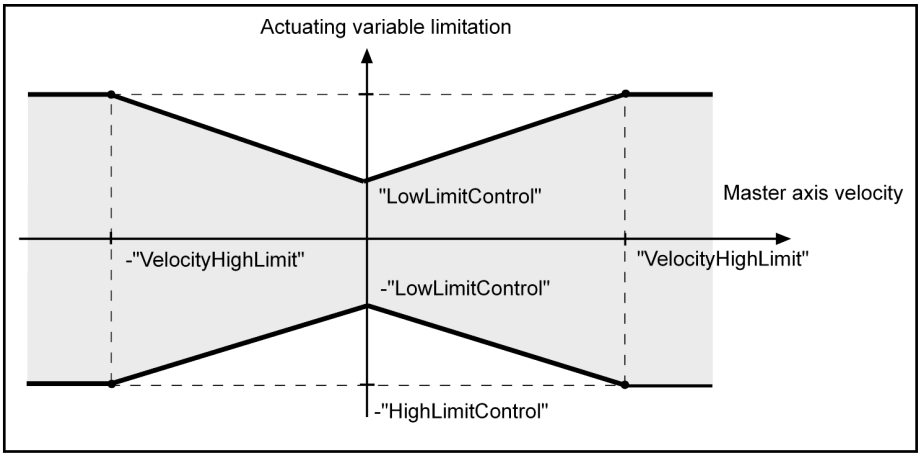


Fig.9-12: Control variable limitation of the dancer position controller

Control variable limitation

The control variable limitation of the dancer position controller is dynamically adjusted according to the reference axis velocity and the diameter ratio ("format length" / Pi * "ReferenceDiameter"). This means that at higher velocity, the limitation of the additive velocity of the dancer position controller is automatically higher than at lower velocity. Consideration of the diameter ratio results in the limitation for winding applications automatically being adapted to a changing "ReferenceDiameter".

Signal-time diagram

The following signal-time diagram represents the selected signals of the function block. The function block is activated by the "Enable" input, which is signaled at the "InOperation" output. Operational readiness can be interrupted by an error signaled at the "Error" output. The internal controller process can be paused by the "Pause" input. "Pause" is signaled at the "InPause" output.

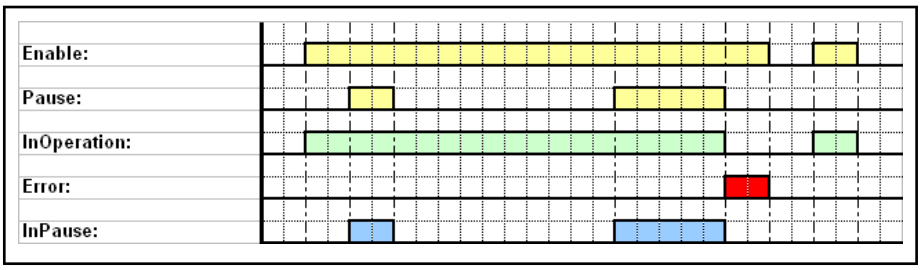


Fig.9-13: Signal-time diagram for selected inputs and outputs of the MB_DancerControlType02 function block

Functional Description

The function block cyclically calculates closed-loop dancer position control for a pull roll with dancer or for a winding axis with dancer. The IL_PIDType01 function block from the RIL_LoopControl library is used as controller. The controller is a PID-controller. The cyclic output variables of the dancer position controller are the speed offset of the respective drive for the pull roll or the winding axis, as well as the current control deviation of the dancer position controller. The "AddVelocityDancer" output of the function block should act on the parameter P-0-0690 "Additive velocity command value, process controller" (external connection) for a closed-loop dancer position control. The current control deviation "CurrentDancerDeviation" is available for diagnostic purposes. The proportional gain (factor K_P) and the low and high values for control variable limitation "DancerLimits" are automatically adjusted with the "ReferenceDiameter" input based on the format length "FormatLength". The diameter of the pull roll "ReferenceDiameter" has to be specified for closed-loop dancer position control for a pull roll. "ReferenceDiameter" specifies the current winding diameter with a winder with closed-loop dancer position control. If the values for the integral part (integral action time

T_n, "IControl") and the derivative element (derivative time T_D, "DControl") are zero, the internal memory parameters of the PID-controller are deleted. There has to be a value proportional to the dancer roll adjustment (e.g. an analog tension value) at the "FeedBackDancer" input. The command value and the actual value must have the same unit.

The function block only processes positive command values and actual values.

Negative command values result in an error.

Negative actual values are limited to zero.

Closed-loop dancer position control

To determine the dancer position, a measuring device is required that converts the dancer position into a proportional signal. The actual position of the dancer is subtracted from a given command value and specified as a control deviation on an internally integrated PID-controller. The control variable of the controller acts upon the speed command value by addition. Thus, the winding axis rotates faster or slower compared to the master axis. The position of the dancer is thereby controlled.

Under normal conditions (e. g. no high-speed command value jumps), the circumferential velocity of a winding axis is approximately adjusted by means of the fine adjustment of the gear. The tensile axis is approximately adjusted by means of the electronic gear. Thus, the dancer position controller should only control small deviations.

Direction of rotation of the winding axis, control direction of the closed-loop dancer control

The direction of rotation of the tensile axis or winding axis can be set using the drive parameter P-0-0108 "Master drive polarity" or the axis parameter A-0-2798 "Polarity of master axis position". If the winding axis does not rotate in the desired direction, the parameter P-0-0108 or the axis parameter A-0-2798 has to be reset accordingly. When inverting the drive direction of rotation, the control direction (ControlPolarity) is also to be inverted.

Error Handling

The following table lists and describes the error numbers of the function block.

The error codes from the "F_RELATED table (16#0170)" are used. Additionally, the error codes of the function block IL_PIDType01 from the RIL_LoopControl library (refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries") as well as the MB_GetCyclicParameterHandle (see chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128) can be output.

RMB_TechWinder.library/MX_TechWinder.lib

ErrorID	Additional1	Additional2	Text
STATE_MACHINE_ERROR	16#1160	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#1161	16#0002	HighLimit < LowLimit
		16#0003	FormatLength < 0.0
		16#0004	ReferenceDiameter < 0.0
		16#0005	ProcessWindow < 0.0
		16#0006	HighLimit < 0.0
		16#0007	LowLimit < 0.0
		16#0008	VelocityHighLimit < 0.0
		16#0009	P-Gain < 0.0
		16#000A	Setpoint < 0.0
		16#000B	MinFeedback < 0.0
		16#000C	MaxFeedback < 0.0
		16#000D	MaxFeedback < MinFeedback
		16#000E	Number of slave axis incorrect
		16#000F	Number of master axis incorrect
DEVICE_ERROR	16#1162	16#0001	Axis not in operating mode
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated

Fig.9-14: Error codes of the MB_DancerControlType03 function block

Required hardware
- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware
- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS

  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software
- IndraWorks12VRS for IndraMotion MLC or IndraMotion MLD

Special features for the IndraMotion MLC

Interpolation in the drive has to be activated for the dancer controller axis.

(Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **parameter P-0-0690 ("Additive velocity command value, process controller")** has to be entered in the cyclic SERCOS channel of the corresponding drive "UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

Special features for the IndraMotion MLD

"Interpolation in the drive" has to be activated in the drive for the dancer controller axis when using the IndraMotion MLD-S control with a superordinate IndraMotion MLC control (Real axes → Corresp. axis → Properties). This setting does not exist when using the IndraMotion MLD-M control.

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **parameter P-0-0690 ("Additive velocity command value, process controller")** has to be entered in the cyclic channel between the control and drive of the respective drive ("UserCmdDataX").

The real axes are parameterized in IndraWorks in parameterization mode. (The dialog is in the project tree below the MLD node. → AxisData)

The "AxisData structure" of the drive must be parameterized (P-0-1367 "PLC configuration" Bit 6 is set or dialog PLC configuration)

9.2.3 Commissioning of a Closed-Loop Dancer Position Control

Drive control loop

It is assumed that the speed control loop of the draw roll drive was set. That means that the parameters S-0-0100 "Velocity controller proportional gain" and S-0-0101 "Velocity controller integral time" have to be set according to the application.

Commanding dancer controller axis

The correct operation mode of the drive and the drive enable of the drive have to be set for the dancer controller axis. This can be carried out in two different ways as follows:

- Set the "ModeSyncVel" operation mode by means of the AxisInterface (recommended)
- Use the PLCopen "MC_Power", "MC_GearIn", "MC_Stop" function blocks to command the axis

Basic settings

First of all, the control direction of the dancer position controller is defined depending on the direction of rotation of the tension axis. The controller acts in opposite directions, depending on this setting. The control direction is defined by setting the "ControlPolarity" input.

Another important basic setting is the setting of the reference diameter ("ReferenceDiameter" input). For the winder with dancer (see chapter 9 Function blocks for winder functionality, page 409) the currently calculated winder diameter is applied. The diameter of the tensile axis is entered for the dancer position controller. Using the "MinDancerFeedback" and "MaxDancerFeedback" inputs, values for the adjustment range of the dancer can be entered, which will be displayed by the "MinFeedbackAchieved" and "MaxFeedbackAchieved" outputs. These values should be within the limit positions of the dancer.

The process variable window ("WindowDancer") is defined. The monitoring process of the dancer command position is parameterized in this window. If the actual dancer position value differs from the position command value ("SetpointDancer"), related to the command value, by more than the specified percentage, the "InWindow" binary output is reset ("InWindow": = FALSE). This allows the superordinate PLC to react to the process accordingly. Exiting the parameterized window has initially no effect on the function of the dancer position controller. If, for example, the dancer command value "SetpointDancer" is 500.0 and the window size has been set to "WindowDancer" = 10%, the limits of the window are 450.0 and 550.0. Within this value range, the "InWindow" output = TRUE.

Finally, the format length has to be specified. The format length corresponds to an unwound revolution of the reference axis ("FormatLength" input).

Parameterization of the dancer position controller

The controller parameters are set in the next step.

These parameters are the

- proportional gain ("PControl")
- Integral action time ("IControl")
- derivative time ("DControl")

To set the parameters, the method according to Ziegler and Nichols can be used. The control loop is closed and the proportional gain ("PControl") is increased until the output of the control loop performs a constant oscillation with the period T_{crit} at proportional gain $K_{p,crit}$ at a constant input. The setting rules for the gain K_p ("PControl"), the integral action time T_n ("IControl") and the derivative time T_v ("DControl") are as given in the following table.

Controllers	PControl	IControl	DControl
P-controller	$0.5 \cdot K_{P,crit}$		
PI-controller	$0.45 \cdot K_{P,crit}$	$0.85 \cdot T_{crit}$	
PD-controller	$0.55 \cdot K_{P,crit}$		$0.15 \cdot T_{crit}$
PID-controller	$0,6 \cdot K_{P,crit}$	$0.5 \cdot T_{crit}$	$0,12 \cdot T_{crit}$

Fig.9-15: Controller parameterization according to Ziegler and Nichols

Finally, the inputs of the MB_WIND_DANCER_LIMITS structure are specified. The operating range of the dancer position controller is determined by specifying the upper and lower limit ("LowLimitControl", "HighLimitControl") of the internal limiter.

Initial startup of the dancer position controller

First, a value is specified as command value at the "SetpointDancer" input. Generally, it is helpful if this value corresponds to the centered position of the dancer. In this case, the dancer has a comparatively long web distance.

After having enabled the dancer position controller ("Enable" = TRUE), the dancer moves to the defined command value position. The web velocity fluctuations cause fluctuations in the web tension that act on the dancer positions as thus on the control deviation of the dancer position controller. The dancer tries to compensate this using the dancer position controller by adapting the drive velocity of the tensile axis or of the winding axis. These variables are displayed at the "CurrentDancerDeviation" and "AddVelocityDancer" outputs and can thus be monitored.

9.3 Function Block for Diameter Calculation

9.3.1 Introduction and Overview

The MB_DiameterCalculatorType03 function block provides a versatile function block to carry out diameter calculations. It is used in all diameter calculators described in this manual. Therefore, it is described first. The following explains the function block in detail.

9.3.2 MB_DiameterCalculatorType03

Brief Description

The function block cyclically calculates the current diameter, the velocity adaptation for the winding axis and the current moment of inertia of the winding material on a winder. The function block is called up for each winding axis. Cyclic output variables are the current winding diameter "CurrentDiameter", the resulting gear fine adjustment for the associated winding drive "GearRatioFineAdjust" and the current moment of inertia of the winding material "CurrentInertia".

The fine adjustment of the gear "GearRatioFineAdjust" affects the parameter P-0-0694 ("Gear ratio fine adjust, process controller"). This parameter is entered in the respective cyclic SERCOS channel. Parameterization is carried out in the parameterization mode.

RMB_TechWinder.library/MX_TechWinder.lib

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechWinder.lib

Fig.9-16: Reference table of the MB_DiameterCalculatorType03

Interface Description

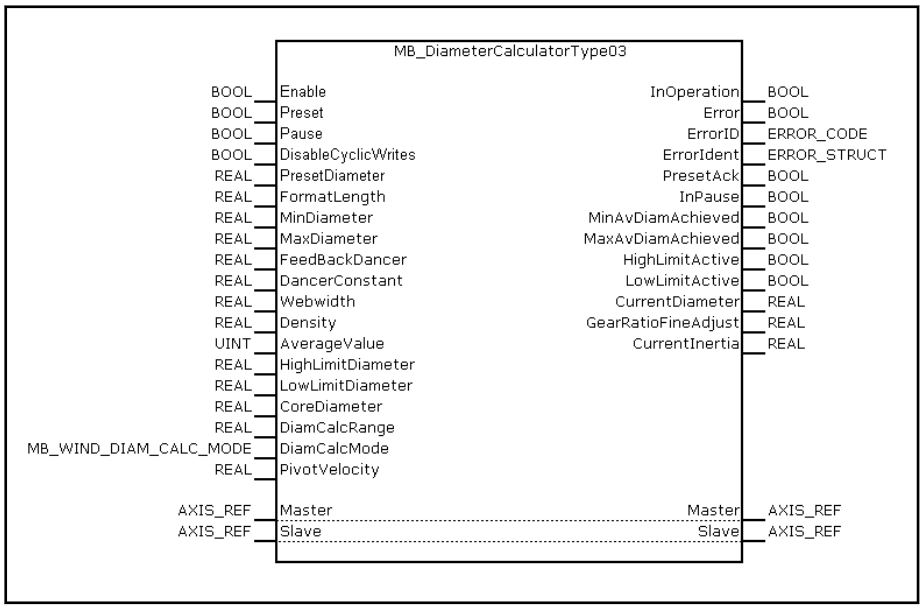


Fig.9-17: Function block MB_DiameterCalculatorType03

I/O type	Name	Type	Comment
VAR_IN_OUT	Master	AXIS_REF	Reference to master axis
	Slave	AXIS_REF	Reference to slave axis (tensile axis)
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	Preset	BOOL	Preset of the current winding diameter
	Pause	BOOL	Diameter calculator paused (level-controlled)
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output
	PresetDiameter	REAL	CurrentDiameter is set to PresetVal, GearRatioFineAdjust is adjusted accordingly (edge-controlled)
	FormatLength	REAL	Format length [mm] (corresponds to an unwound revolution of the reference axis)
	MinDiameter	REAL	Minimum diameter [mm] Is only used for display purposes by the "MinAvDiamAchieved" output
	MaxDiameter	REAL	Maximum diameter [mm] Is only used for display purposes by the "MaxAvDiamAchieved" output

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	FeedbackDancer	REAL	Actual dancer position value (only for the "Winder with dancer" functionality)!
	DancerConstant	REAL	Web distance when adjusting the dancer roll related to the unit of the dancer adjustment [mm] Sign has to be set depending on the structure of the dancer (only for the "Winder with dancer" functionality)!
	Webwidth	REAL	Web width [mm]
	Density	REAL	Material density [kg/dm^3]
	AverageValue	UINT	Number of diameter values averaged. The input is limited to a maximum of 30
	HighLimitDiameter	REAL	Upper limit of the diameter [mm]. The diameter is limited to this value
	LowLimitDiameter	REAL	Lower limit of the diameter [mm]. The diameter does not fall below this value
	CoreDiameter	REAL	Core diameter of the winder [mm]
	DiamCalcRange	REAL	Range for the diameter calculation [degrees]. It specifies the number of degrees after which the diameter should be calculated on a regular basis. Relates to the winding axis (load-sided)
	DiamCalcMode	MB_WIND_DIAM_CALC_MODE	Diameter calculation mode 0x0000: CALC_DIAM_MANUAL Diameter calculation depending on CalcRange 0x0001: FIRST_REV_AUTO_CALC Diameter calculation is executed after every 5 degrees for the first revolution. Thereafter, it depends on the CalcRange
	PivotVelocity	REAL	Pivot velocity of the winder (relative velocity to the winder motion) [mm/s] This corrects the diameter calculation if there is a relative motion of the winding axis
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	PresetAck	BOOL	Preset diameter was applied (is set as long as "Preset" is pending)
	InPause	BOOL	Diameter calculation paused
	MinAvDiamAchieved	BOOL	Minimum diameter reached (based on the averaged diameter)
	MaxAvDiamAchieved	BOOL	Maximum diameter reached (based on the averaged diameter)

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	HighLimitActive	BOOL	Currently calculated diameter has reached the upper limit. (Diameter value is discarded)
	LowLimitActive	BOOL	Currently calculated diameter has reached the lower limit. (Diameter value is discarded)
	CurrentDiameter	REAL	Current winding diameter
	GearRatioFineAdjust	REAL	Fine gear adjustment for speed adaptation of the drive [%]
	CurrentInertia	REAL	Current total moment of inertia (based on the motor side [kg/m^2]

Fig.9-18: Interface variables of the MB_DiameterCalculatorType03 function block

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
PresetDiameter	REAL	> 0.0	n.def	100.0	Continuous
FormatLength	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"
MinDiameter	REAL	> 0.0	n.def	100.0	Continuous
MaxDiameter	REAL	> 0.0	n.def	2000.0	Continuous
FeedbackDancer	REAL	0.0	n.def	0.0	Continuous
DancerConstant	REAL	n.def	n.def	0.0	Continuous
Webwidth	REAL	0.0	n.def	0.0	Rising edge at "Enable"
Density	REAL	0.0	n.def	1.0	Rising edge at "Enable"
AverageValue	UINT	1	30	1	Continuous
HighLimitDiame-ter	REAL	0.0	n.def	2000.0	Continuous
LowLimitDiameter	REAL	0.0	n.def	100.0	Continuous
CoreDiameter	REAL	0.0	n.def	100.0	Rising edge at "Enable"
DiamCalcRange	REAL	0.0	n.def	360.0	Continuous

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective
DiamCalcMode	ENUM			0	Rising edge at "Enable"
PivotVelocity	REAL	0.0	n.def	0.0	Continuous

Fig.9-19: Min./max. and default values of the MB_DiameterCalculatorType03

Name	Value	Description
CALC_DIAM_MANUAL	0	Diameter calculation depending on CalcRange
FIRST_REV_AUTO_CALC	1	Diameter calculation is executed after every 3 degrees for the first revolution. Thereafter, it depends on the CalcRange

Fig.9-20: Elements of the MB_WIND_DIAM_CALC_MODE enumeration type

Signal-time diagram

The following signal-time diagram describes the inputs and outputs of the function block affecting its operation. Operation of the function block is not possible without setting the "Enable" input. The operational readiness is signaled by the "InOperation" output and only interrupted by an error displayed by the "Error" output. If there is a positive edge at the "Preset" output, the value applied at the "PresetDiameter" input is written as output of the diameter calculator. This is possible irrespective of the "Pause" input. Setting "Preset" is acknowledged by the "PresetAck" output. Setting the "Pause" input, which freezes the calculation of the diameter, is signaled at the "InPause" output.

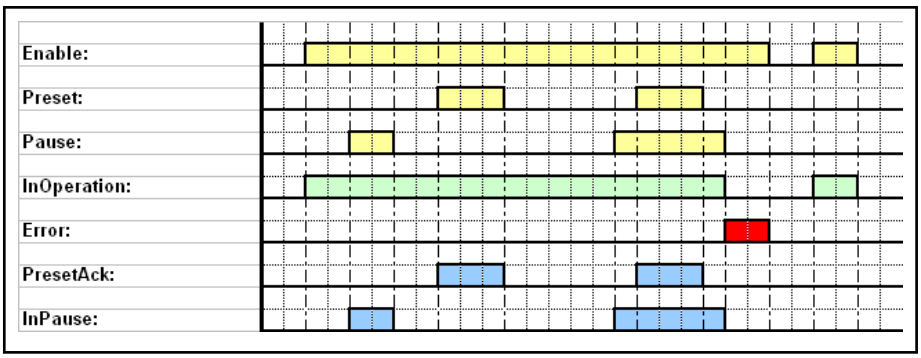


Fig.9-21: Signal-time diagram for selected inputs and outputs of the MB_DiameterCalculatorType03 function block

Functional Description

Preset of the diameter

Since the calculated diameter directly affects the winding axis velocity via the resulting fine adjustment, a meaningful value has to be preset for the diameter at the start of the calculation when activating the diameter calculator. Thus, a value for the diameter, which is close to the actual diameter, should be specified at the input "PresetDiameter" and a preset should be performed. For a rewinder, the value should be slightly lower and for an unwinder, it should be slightly higher for an unwinder. This allows an easier startup in both cases.

With each new roll, a preset of the winding diameter should be performed once at the start. This is executed via the "Preset" input. The diameter is set to the value of the "PresetDiameter" input. Setting the diameter with the help of "Preset" is always possible without setting the "Pause" input.

After having switched on the control, the winding diameter, the gear fine adjustment and the moment of inertia are all equal zero. This means that a preset should be performed each time the control is switched on in order to preset the respective variables with meaningful values. If this is not done, no meaningful value can be calculated for the current winding diameter. This can

result in a malfunction of the diameter calculator and thus lead to a winder stop.

If it not possible to specify a correct preset value for the initial winding, we recommend the use of "auto-calc mode" to calculate the diameter (FIRST_REV_AUTO_CALC). In this mode, the range of diameter calculation (DiamCalcRange) is automatically adjusted for the first winding axis revolution after the system has been switched on in order to obtain a meaningful starting diameter more quickly.

☞ If the current diameter is to be kept after the control has been switched off, the "CurrentDiameter" output has to be copied to a remanent variable. When the control is switched on again, a preset can be performed again with this value.

Diameter calculation

The calculation of the diameter is proportional to the quotient from the reference axis velocity n_{Ref} and the winding axis velocity n_{Winder} and to the format length F (corresponding to an unwound revolution of the reference axis).

$$d \sim \frac{n_{Ref}}{n_{Winkel}} * F$$

Fig.9-22: Proportionality in diameter calculation

☞ The ratio "Format length"/"CoreDiameter" may not be greater than 25.

Otherwise, the gear adjustment via the gear fine adjustment is insufficient!

The calculated winding diameter directly affects the resulting fine adjustment of the gear. Thus, the velocity of the winder is adjusted according to the diameter.

Average value generation

The calculated current diameter "CurrentDiameter" can be averaged by setting the "AverageValue" input.

The number of the diameter values to be averaged is applied at "AverageValue". Only the current, internally calculated, diameters within the limits of "HighLimitDiameter" and "LowLimitDiameter" contribute to generation of the average value. Values outside these limits are discarded. Values between 1 and 10 are appropriate for "AverageValue". Therefore, "AverageValue" is limited to a maximum of 30. If a higher value is entered, "AverageValue" is limited to 30. The higher the number selected, the more precise is the calculated diameter. However, the accuracy of the diameter is reduced by increasing values for "AverageValue", as this diameter is adjusted due to the "filtering". Thus, the value for "AverageValue" should be selected appropriately. With smaller winding diameters, the variation in the diameter values calculated are higher. Therefore, the value of the filter should not be too high. Otherwise, the delay of the diameter calculated, caused by the averaging, would result in a distinct "lag" of the diameter calculated. For a winder with dancer, for example, this can result in an undesired displacement of the dancer.

Calculation of the moment of inertia

With the help of the inputs "Webwidth" and "Density", the moment of inertia of the winder is calculated internally in the diameter calculator and displayed at

the "CurrentInertia" output. With the help of the moment of inertia, which depends on the diameter, an adaptation of the K_P gain of the speed loop (S-0-0100) can be implemented in the PLC project in the IndraLogic if required.

Preparation for splice

Finally, the inputs "MinDiameter" and "MaxDiameter" can be set. The variables specified here affect the outputs "MinAvDiamAchieved" and "MaxAvDiamAchieved" and display a message if the currently calculated winding diameter exceeds the specified limits. The respective values should be within the admissible diameter range defined by the inputs "HighLimitDiameter" and "LowLimitDiameter". Thus, the time for specific process steps, for example, for a splice or for the adjustment of the "DiamCalcRange" input, can be detected and displayed in the PLC based on the specification of the minimum/maximum diameter.

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used. Additionally, the error codes of the MB_GetCyclicParameterHandle function block (see chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128) can be output.

ErrorID	Additional1	Additional2	Text
STATE_MACHINE_ERROR	16#1100	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#1101	16#0001	HighLimitDiameter < LowLimitDiameter
		16#0002	FormatLength < 0.0
		16#0003	WebWidth < 0.0
		16#0004	Density < 0.0
		16#0005	Core diameter < 0.0
		16#0006	DiamCalcRange < 0.0
		16#0007	Preset diameter < 0.0
		16#0008	Minimum diameter < 0.0
		16#0009	Maximum diameter < 0.0
		16#000A	HighLimitDiameter < 0.0
		16#000B	LowLimitDiameter < 0.0
		16#000C	DiamCalcMode invalid
		16#000D	Number of slave axis incorrect
		16#000E	Number of master axis incorrect
DEVICE_ERROR	16#1102	16#0001	No operating mode (no SERCOS phase 4)
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated

Fig.9-23: Error codes of the MB_DiameterCalculatorType03 function block

Required hardware

- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware

- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS

XX - Device variant, e.g. CML65

- IndraDrive drive firmware

Required software

- IndraWorks 12VRS for IndraMotion MLC or IndraMotion MLD

Special features for the IndraMotion MLC

The interpolation in the drive has to be enabled for the winding axis.

(Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **Parameter P-0-0694 ("Gear ratio fine adjustment, process controller")** has to be entered in the cyclic SERCOS channel of the corresponding drive "UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

The standstill window of the reference axis should be set as low as possible (e.g. 0.1 rpm). When there is a standstill message of the reference axis, the diameter calculator does not calculate!

Special features for the IndraMotion MLD

Interpolation in the drive has to be activated in the drive for the winding axis when using the IndraMotion MLD-S control with a superordinate IndraMotion MLC control (Real axes → Corresp. axis → Properties). This setting does not exist when using the IndraMotion MLD-M control.

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **Parameter P-0-0694 ("Fine adjustment gear ratio, process controller")** has to be entered in the cyclic channel between the control and the drive of the corresponding drive ("UserCmdDataX").

The real axes are parameterized in the parameterization mode in IndraWorks (the dialog is located in the project tree below the MLD node → AxisData).

The "AxisData structure" of the drive must be parameterized (P-0-1367 "PLC configuration" Bit 6 is set or dialog PLC configuration).

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

The standstill window of the reference axis should be set as low as possible (e.g. 0.1 rpm). When there is a standstill message of the reference axis, the diameter calculator does not calculate!

9.3.3 Commissioning the Diameter Calculator

Drive control loop

It is assumed that the speed control loop of the winding drive was set. That means that the parameters S-0-0100 "Velocity controller proportional gain" and S-0-0101 "Velocity controller integral time" have to be set according to the application.

Proceed with the following to commission the diameter calculator:

- Specify the fabric web variables
- Specify the winding variables
- Specify the cyclically changeable inputs
- Specify the inputs for manipulation of the diameter calculation and
- further specifications

Specifying the fabric web variables

The fabric web variables are the inputs and values related to the dimensions of the fabric web. They can normally be considered as constant values during operation. Exceptions are webs of fabric with variable materials or variable dimensions.

The following inputs describe fabric web variables:

- FormatLength
- Webwidth
- Density

Since these variables are not read on a cyclical basis, they have to be applied as input values at the inputs prior to activation of the function block.

Specifying the winding variables

Winding variables describe the spatial and general dimensions of the winder. These are the core diameter "CoreDiameter" and the maximum and minimum permissible winding diameter "LowLimitDiameter" and "HighLimitDiameter". With the help of the ratio of reference axis speed and winding axis speed, the current diameter can be determined with the core diameter. The current diameter "CurrentDiameter" is limited to these defined limits.

Specifying the cyclically changeable inputs

At the beginning and during the diameter calculation, it can be necessary to manually set the value of the diameter. Therefore, the "PresetDiameter" input is described. It is read on a cyclical basis and thus becomes effective within a cycle.

Specifying the inputs to manipulate the diameter calculation

The method for calculation of the diameter is implemented in such a way that the result of the calculation can be influenced with the aid of two inputs of the function block. "AverageValue" specifies the number of diameter values to be used for averaging. "DiamCalcRange" allows the frequency of the calculation to be influenced, which, in turn, influences the accuracy of the diameter value. Both variables can be manipulated and adjusted during operation since they are read and processed on a cyclical basis.

Further specifications

Depending on the respective application, the inputs "DancerConstant", "FeedBackDancer" (both only for a winder with dancer), "MinDiameter" and "MaxDiameter" are connected. If these inputs are not required for the functions to be implemented in the PLC project, it is recommended to apply the same values to the inputs "MinDiameter" and "MaxDiameter" which have been applied to the inputs "LowLimitDiameter" and "HighLimitDiameter". That avoids a respective error message.

Commanding winding axis

The correct operation mode of the drive and the drive enable of the drive have to be set for the winding axis. This can be carried out in two different ways as follows:

- Set the "ModeSyncVel" operation mode by means of the AxisInterface (recommended)
- Use the PLCopen "MC_Power", "MC_GearIn", "MC_Stop" function blocks to command the axis

9.4 Function Block to Record an Externally Measured Diameter

9.4.1 Introduction and Overview

The MB_DiameterMeasurementType01 function block reads a measured diameter and adapts the velocity of the winding drive accordingly. In this case,

it replaces the function block MB_DiameterCalculatorType03. It is used in the function blocks MB_WinderDancerCtrlType01 and MB_WinderTensionCtrl-Type02.

9.4.2 MB_DiameterMeasurementType01

Brief Description

The function block calculates and reads a measured diameter and calculates the velocity adaptation for the winding axis and the current moment of inertia of the winding material on a winder. The function block is called up for every winding axis, where the diameter is measured. Cyclic output variables are the resulting gear fine adjustment for the associated winding drive "GearRatioFineAdjust" and the current moment of inertia of the winding material "CurrentInertia".

The fine adjustment of the gear "GearRatioFineAdjust" influences the parameter P-0-0694 ("Gear ratio fine adjustment, process controller". This parameter is entered in the respective cyclic SERCOS channel. Parameterization is carried out in the parameterization mode.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechWinder.lib

Fig.9-24: Reference table of the MB_DiameterMeasurementType01

Interface Description

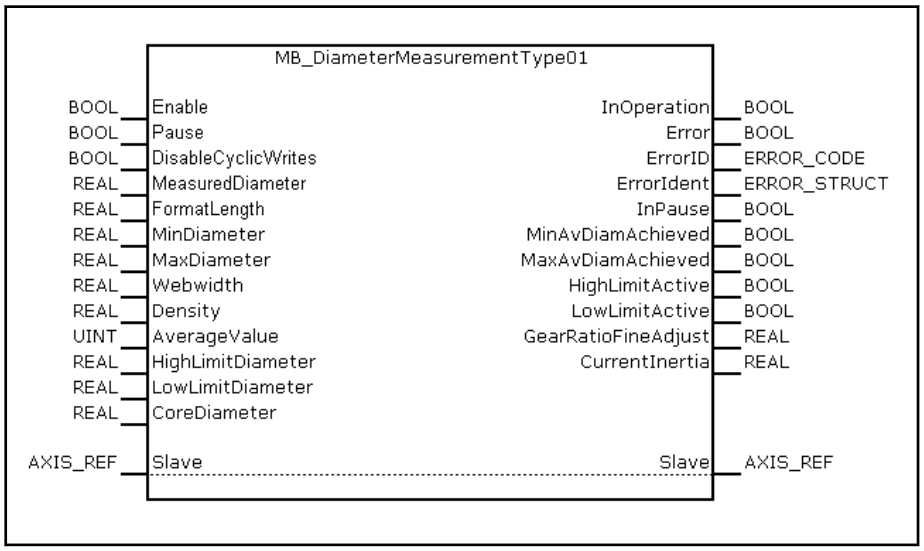


Fig.9-25: MB_DiameterMeasurementType01 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	Slave	AXIS_REF	Reference to the slave axis
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	Pause	BOOL	Function block pauses, gear fine adjustment remains "frozen"
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output
	MeasuredDiameter	REAL	Measured diameter [mm]

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	FormatLength	REAL	Format length [mm] (1 revolution of the master axis unwound)
	MinDiameter	REAL	Minimum diameter [mm], Is only used for display purposes by the "MinAvDiamAchieved" output
	MaxDiameter	REAL	Maximum diameter [mm], is only used for display purposes by the "MaxAvDiamAchieved" output
	WebWidth	REAL	Web width [mm]
	Density	REAL	Material density [kg/dm^3]
	AverageValue	UINT	Number of diameter values averaged. The input is limited to a maximum of 30
	HighLimitDiameter	REAL	Upper limit for the diameter [mm] The diameter is limited to this value
	LowLimitDiameter	REAL	Lower limit for the diameter [mm] The diameter is limited to this value
	CoreDiameter	REAL	Core diameter of the winder [mm]
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	InPause	BOOL	Dancer controller pausing
	MinAvDiamAchieved	BOOL	Minimum diameter reached (based on the averaged diameter)
	MaxAvDiamAchieved	BOOL	Maximum diameter reached (based on the averaged diameter)
	HighLimitActive	BOOL	Currently calculated diameter has reached the upper limit (Diameter value is discarded)
	LowLimitActive	BOOL	Currently calculated diameter has reached the lower limit (Diameter value is discarded)
	GearRatioFineAdjust	REAL	Fine gear adjustment for speed adaptation of the drive [%]
	CurrentInertia	REAL	Current total moment of inertia (based on the motor side) [kg/m^2]

Fig.9-26: Interface variables of the MB_DiameterMeasurementType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
MeasuredDiame-ter	REAL	> 0.0	n.def	100.0	Continuous
FormatLength	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"
MinDiameter	REAL	> 0.0	n.def	100.0	Continuous
MaxDiameter	REAL	> 0.0	n.def	2000.0	Continuous
Webwidth	REAL	0.0	n.def	0.0	Rising edge at "Enable"
Density	REAL	0.0	n.def	1.0	Rising edge at "Enable"
AverageValue	UINT	0	30	1	Continuous
HighLimitDiame-ter	REAL	0.0	n.def	2000.0	Continuous
LowLimitDiameter	REAL	0.0	n.def	100.0	Continuous
CoreDiameter	REAL	0.0	n.def	100.0	Rising edge at "Enable"

Fig.9-27: Min./max. and default values of the MB_DiameterMeasurement-Type01 inputs

Signal-time diagram

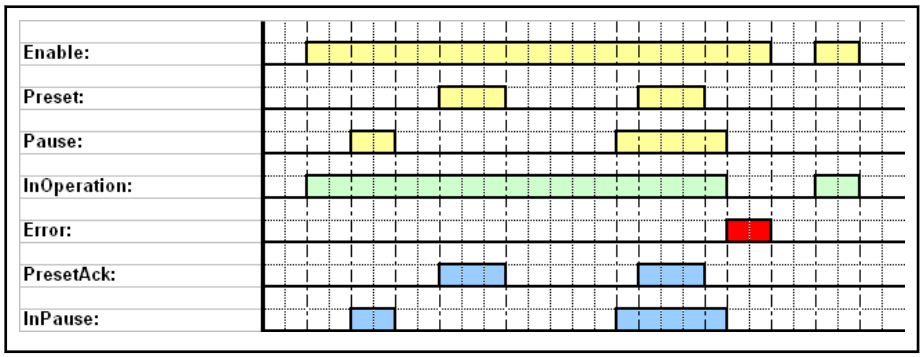


Fig.9-28: Signal-time diagram for selected inputs and outputs of the function block MB_DiameterMeasurementType01

Functional Description

The function block MB_DiameterMeasurementType01 reads a measured diameter value (e.g. by means of an ultra-sound sensor) and ensures the correct velocity adjustment of the respective winding axis based on the diameter measured.

The function block is used in the function blocks MB_WinderDancerCtrl-Type01 and MB_WinderTensionCtrlType02.

☞ The ratio "Format length"/"CoreDiameter" may not be greater than 25.

Otherwise, the gear adjustment via the gear fine adjustment is insufficient!

RMB_TechWinder.library/MX_TechWinder.lib

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used. Additionally, the error codes of the MB_GetCyclicParameterHandle function block (see chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128) can be output.

ErrorID	Additional1	Additional2	Text
STATE_MACHINE_ERROR	16#1180	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#1181	16#0001	HighLimit < LowLimit
INPUT_RANGE_ERROR	16#1181	16#0002	FormatLength < 0.0
INPUT_RANGE_ERROR	16#1181	16#0003	WebWidth < 0.0
INPUT_RANGE_ERROR	16#1181	16#0004	Density < 0.0
INPUT_RANGE_ERROR	16#1181	16#0005	CoreDiameter < 0.0
INPUT_RANGE_ERROR	16#1181	16#0008	MinDiameter < 0.0
INPUT_RANGE_ERROR	16#1181	16#0009	MaxDiameter < 0.0
INPUT_RANGE_ERROR	16#1181	16#000A	HighLimit < 0.0
INPUT_RANGE_ERROR	16#1181	16#000B	Lowlimit < 0.0
INPUT_RANGE_ERROR	16#1181	16#000C	Incorrect slave axis number
DEVICE_ERROR	16#1182	16#0001	No operating mode (no SERCOS phase 4)
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated

Fig.9-29: Error codes of the MB_DiameterMeasurementType01 function block

Required hardware

- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware

- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software

- IndraWorks 12VRS for IndraMotion MLC or IndraMotion MLD

Special features for the IndraMotion MLC

The interpolation in the drive has to be enabled for the winding axis.

(Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **Parameter P-0-0694 ("Gear ratio fine adjustment, process controller")** has to be entered in the cyclic SERCOS channel of the corresponding drive "UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

Special features for the IndraMotion MLD

Interpolation in the drive has to be activated in the drive for the winding axis when using the IndraMotion MLD-S control with a superordinate IndraMotion MLC control (Real axes → Corresp. axis → Properties). This setting does not exist when using the IndraMotion MLD-M control.

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **Parameter P-0-0694 ("Fine adjustment gear ratio, process controller")** has to be entered in the cyclic channel between the control and drive of the corresponding drive ("UserCmdDataX").

The real axes are parameterized in IndraWorks in parameterization mode. (The dialog is in the project tree below the MLD node → AxisData.)

The "AxisData structure" of the drive must be parameterized (P-0-1367 "PLC configuration" Bit 6 is set or dialog PLC configuration)

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

9.5 Diameter Calculator with Dancer

9.5.1 Introduction and Overview

The diameter calculator with dancer is used on center winders. The tensile stress is specified by a force which acts upon the movable part of the dancer. This force can be applied by a weight, a spring or a pneumatic element. The closed loop control of the dancer position is performed in the dancer position controller. The actual value of the dancer position is recorded, for example via an analog channel, and is transferred to the control.

The current winding diameter is calculated by an internally integrated diameter calculator. The velocity of the winding axis is adjusted with the help of the calculated winding diameter. Thus, the calculated winding diameter replaces an actual diameter recorded via a sensor system.

The diameter calculator with dancer thus represents a combination of a winder (represented by the winding axis and the diameter calculator) and a dancer position controller. The diameter calculator influences the fine adjustment of the gear and the dancer position controller influences the additive speed command value.

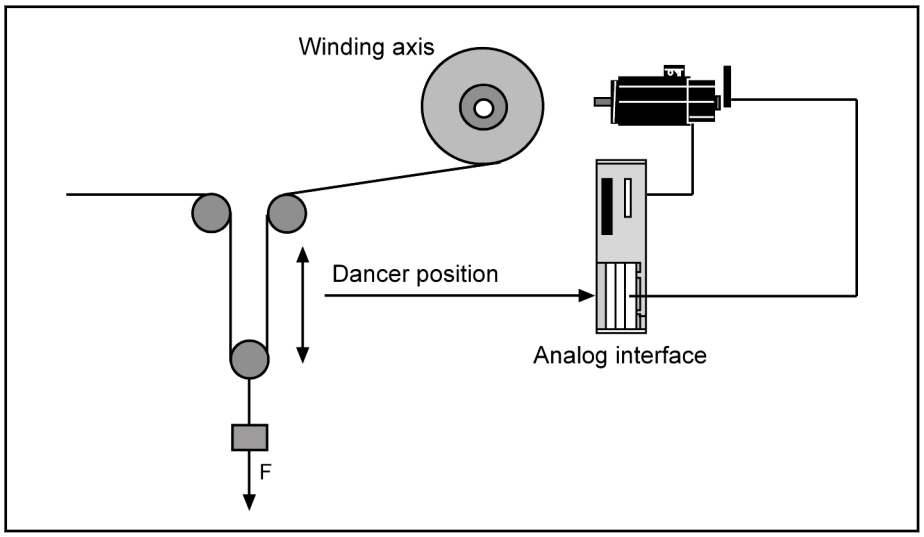


Fig.9-30: Principle of the diameter controller with dancer position controller

RMB_TechWinder.library/MX_TechWinder.lib

9.5.2 MB_WinderDancerCtrlType01

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechWinder.lib

Fig.9-31: Reference table of the MB_WinderDancerCtrlType01

Interface Description

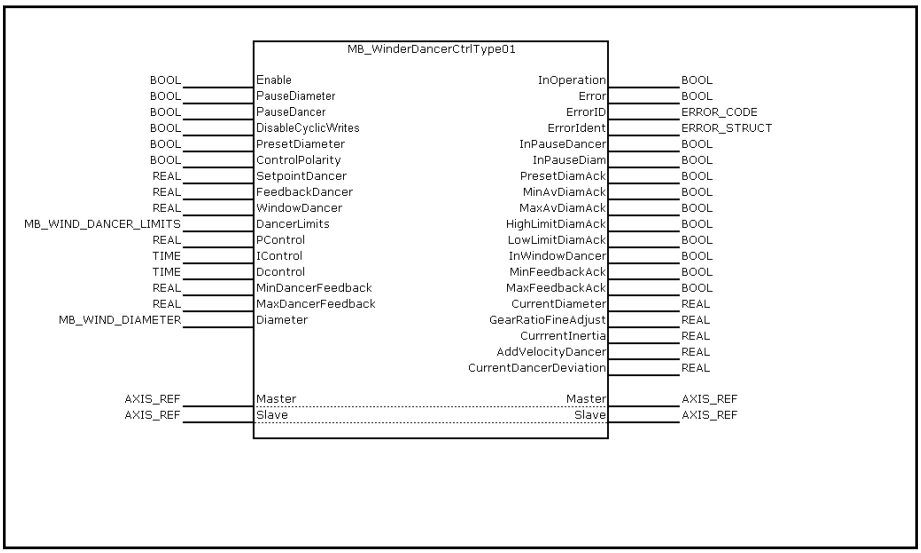


Fig.9-32: MB_WinderDancerCtrlType01 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	Master	AXIS_REF	Reference to master axis
	Slave	AXIS_REF	Reference to the slave axis (winding axis)
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	PauseDiameter	BOOL	Diameter calculator paused; no further measurement of diameter and velocity adjustment (level-controlled)
	PauseDancer	BOOL	Dancer controller paused (level-controlled)
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output
	PresetDiameter	BOOL	CurrentDiameter is set to PresetVal, GearRatioFineAdj is adjusted accordingly (edge-controlled)
	ControlPolarity	BOOL	Polarity of the closed-loop dancer control is inverted (applied in case of pos. edge at "Enable")
	SetpointDancer	REAL	Command value (setpoint) for closed-loop dancer control
	FeedbackDancer	REAL	Actual value (feedback) for dancer position

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	WindowDancer	REAL	Window for monitoring the actual dancer position value [%] with regard to the command dancer position value. If the dancer position deviates from the command value by more than the given value, the function block reports this deviation by resetting the "InWindow" output
	DancerLimits	MB_WIND_DANCER_LIMITS	Structure for the automatic control variable limitation of the dancer
	PControl	REAL	Value for the Kp-gain of the closed-loop dancer control
	IControl	TIME	Value for the integral action time of the closed-loop dancer control
	DControl	TIME	Value for the derivative time of the closed-loop dancer control
	MinDancerFeedback	REAL	Lower threshold for the actual dancer value (only for display with a binary output).
	MaxDancerFeedback	REAL	Upper threshold for the actual dancer value (only for display with a binary output).
	Diameter	MB_WIND_DIAMETER	Structure for the diameter calculator
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	InPauseDancer	BOOL	Dancer controller pausing
	InPauseDiam	BOOL	Diameter calculation paused
	PresetDiamAck	BOOL	Preset diameter is applied (is set providing "Preset" is pending)
	MinAvDiamAchieved	BOOL	Minimum diameter reached (based on the averaged diameter)
	MaxAvDiamAchieved	BOOL	Maximum diameter reached (based on the averaged diameter)
	HighLimitDiamActive	BOOL	Currently calculated diameter has reached the upper limit (Diameter value is discarded)
	LowLimitDiamActive	BOOL	Currently calculated diameter has reached the lower limit (diameter value is discarded)
	InWindowDancer	BOOL	Actual dancer value within dancer window
	MinFeedbackAchieved	BOOL	Actual value of the dancer position is below the input "MinDancerFeedback"
	MaxFeedbackAchieved	BOOL	Actual value of the dancer position is above the input "MaxDancerFeedback"

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	CurrentDiameter	REAL	Current winding diameter
	GearRatioFineAdjust	REAL	Fine gear adjustment for speed adaptation of the drive [%]
	CurrentInertia	REAL	Current moment of inertia of the winding material + mechanics (based on the motor side) [kg m^2]
	AddVelocityDancer	REAL	Speed offset for the winding drive [rpm]
	CurrentDancerDeviation	REAL	Current control deviation of the dancer controller

Fig.9-33: Interface variables of the MB_WinderDancerCtrlType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
PauseDiameter	BOOL			FALSE	Continuous
PauseDancer	BOOL			FALSE	Continuous
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
PresetDiameter	BOOL			FALSE	Continuous
ControlPolarity	BOOL			FALSE	Rising edge at "Enable"
SetpointDancer	REAL	0.0	n.def	0.0	Continuous
FeedbackDancer	REAL	n.def	n.def	0.0	Continuous
WindowDancer	REAL	0.0	n.def	0.0	Continuous
DancerLimits	STRUCT				
PControl	REAL	0.0	n.def	0.0	Continuous
IControl	TIME	0 s	n.def	0 s	Continuous
DControl	TIME	0 s	n.def	0 s	Continuous
MinDancerFeedback	REAL	0.0	MaxDancerFeedback	0.0	Continuous
MaxDancerFeedback	REAL	MinDancerFeedback	n.def	0.0	Continuous
Diameter	STRUCT				depending on the structure element

Fig.9-34: Min./max. and default values of the MB_WinderDancerCtrlType01 inputs

MB_WIND_DIAMETER

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective	Description
PresetVal	REAL	> 0.0	n.def	100.0	Continuous	Default value for the preset diameter [mm]
CoreVal	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"	Diameter of the core [mm]
CalcRange	REAL	0.0	n.def	360.0	Continuous	Range for diameter calculation [degrees] (load-sided)
RepeatLength	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"	Format length [mm] (1 revolution of the reference axis unwound)
Webwidth	REAL	0.0	n.def	0.0	Rising edge at "Enable"	Web width [mm] Is required for the moment of inertia
Density	REAL	0.0	n.def	1.0	Rising edge at "Enable"	Material density [kg/dm^3] Is required for the moment of inertia
AverageValue	UINT	0	20	1	Continuous	Number of diameter values to be averaged [-]
HighLimit	REAL	LowLimit	n.def	2000.0	Continuous	Upper limit for the diameter [mm]. The diameter is limited to this value
LowLimit	REAL	> 0.0	HighLimit	100.0	Continuous	Lower limit for the diameter [mm] The diameter does not fall below this value
MinDiameter	REAL	> 0.0	n.def	100.0	Continuous	Minimum diameter [mm]
MaxDiameter	REAL	> 0.0	n.def	2000.0	Continuous	Maximum diameter [mm]
MechanicInertia	REAL	0.0	n.def	0.0	Continuous	Moment of inertia of the mechanics [kg m^2] based on the motor side Is required for the resulting moment of inertia
DancerConstant	REAL	0.0	n.def	0.0	Continuous	Displacement of the dancer, based on the unit of dancer adjustment [mm/-] Is only required with a winder with dancer
MeasuredVal	REAL	0.0	n.def	0.0	Continuous	Measured diameter [mm]
PivotVelocity	REAL	0.0	n.def	0.0	Continuous	Pivot velocity (for relative motion of the winding axis) [mm/s]

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective	Description
DiamCalcMode	MB_WIND_DIAM_CALC_MODE				Rising edge at "Enable"	Diameter calculation mode CALC_DIAM_MANUAL diameter calculation depending on CalcRange FIRST_REV_AUTO_CALC Diameter calculation is performed every 5 degrees for the first revolution and thereafter depending on the CalcRange
DiamSource	BOOL				Rising edge at "Enable"	Diameter determination: FALSE: Diameter is calculated TRUE: Diameter is measured (MeasuredVal)

Fig.9-35: Interface description of the MB_WIND_DIAMETER data structure

MB_WIND_DANCER_LIMITS

Name	Type	Min. value	Max. value	Default value	Effective	Description
LowLimitControl	REAL	0.0	n.def	0.0	Continuous	Lower limit of the controller output (at zero speed), bipolar [rpm]
HighLimitControl	REAL	0.0	n.def	0.0	Continuous	Upper limit of the controller output (at VelocityHighLimit speed), bipolar [rpm]
VelocityHighLimit	REAL	0.0	n.def	0.0	Continuous	Speed of the master for HighLimitControl [rpm]

Fig.9-36: Interface description of the data structure of the MB_WIND_DANCER_LIMITS

Functional Description

Diameter calculation

The diameter calculation provides the currently calculated diameter over the function block MB_DiameterCalculatorType03 retrieved on a cyclical basis. The additive fine adjustment of the winding axis is set according to the diameter. The axis is moved faster or slower compared to the reference axis to achieve a constant tension of the web of fabric.

A relative motion of the dancer roll is taken into account in the diameter calculation with the help of the "DancerConstant" input. This dancer constant has to be determined once for this purpose.

The following procedure is recommended:

1. Where there is deflection of the dancer from one end position to the other end position, determine the web distance (in mm).
2. Determine the difference between both the actual dancer values of the respective end position.
3. Calculate the quotient of both values.

4. Determine the sign of the dancer constant by conducting a practical test. For this purpose, the diameter calculation is observed at a slowly rotating winding axis and reference axis. Command value jumps to the dancer command value should not have a significant influence on the calculation of the diameter if the sign is correct.
5. "Subsequently optimize" the calculated value for the dancer constant,as described in step 4 if required.

☞ If the current diameter is to be kept after the control has been switched off, the "CurrentDiameter" output has to be copied to a remanent variable. When the control is switched on again, a preset can be performed again with this value.

Splice

After a splice, the winding diameter generally changes abruptly. The last calculated internal value for the winding diameter does no longer correspond to the current diameter. For a correct diameter calculation, a winding diameter preset is reasonable. The input value for the Preset should correspond approximately to the actual value of the winding diameter in order to achieve a resulting fine adjustment as accurate as possible.

☞ If the preset is not performed, malfunctions can occur. Thus, the preset should be performed before activating the diameter calculator.

With a flying splice, it must be ensured that the circumferential velocity of the new roll corresponds to the current one. This has to be ensured with an appropriate PLC project in the IndraLogic. The "Pause" input can be used when activating the diameter calculator for the new roll, for example "Pause" can be disabled after having removed the old roll.

Closed-loop dancer position control

To determine the dancer position, a measuring device is required that converts the dancer position into a proportional signal. The actual position of the dancer is subtracted from a given command value and specified as a control deviation on an internal PID-controller. The control variable of the controller acts upon the speed command value by addition. Thus, the winding axis rotates faster or slower compared to the master axis. This controls the tensile stress of the dancer (see chapter chapter 9.6 "Function Block for the Diameter Calculator with Open-Loop Tensile Stress Control or Closed-Loop Tensile Stress Control" on page 443).

Under normal conditions (e. g. no high variations of the speed command value or abrupt changes in the diameter of the winder), the approximate adjustment of the circumferential velocity of the winding axis is performed by the fine adjustment of the gear. Therefore, the diameter calculator described previously is used. Thus, the dancer position controller has to control only small deviations.

Direction of rotation of the winding axis, control direction of the closed-loop dancer control

The direction of rotation of the tensile axis or winding axis can be set using the drive parameter P-0-0108 "Master drive polarity" or the axis parameter A-0-2798 "Polarity of master axis position". If the winding axis does not rotate in the desired direction, the parameter P-0-0108 or the axis parameter A-0-2798 has to be reset accordingly. When inverting the drive direction of rotation, the control direction (ControlPolarity) is also to be inverted.

RMB_TechWinder.library/MX_TechWinder.lib

Functional diagram of the dancer position controller

The following functional diagram schematically explains the functions of the diameter calculator with dancer.

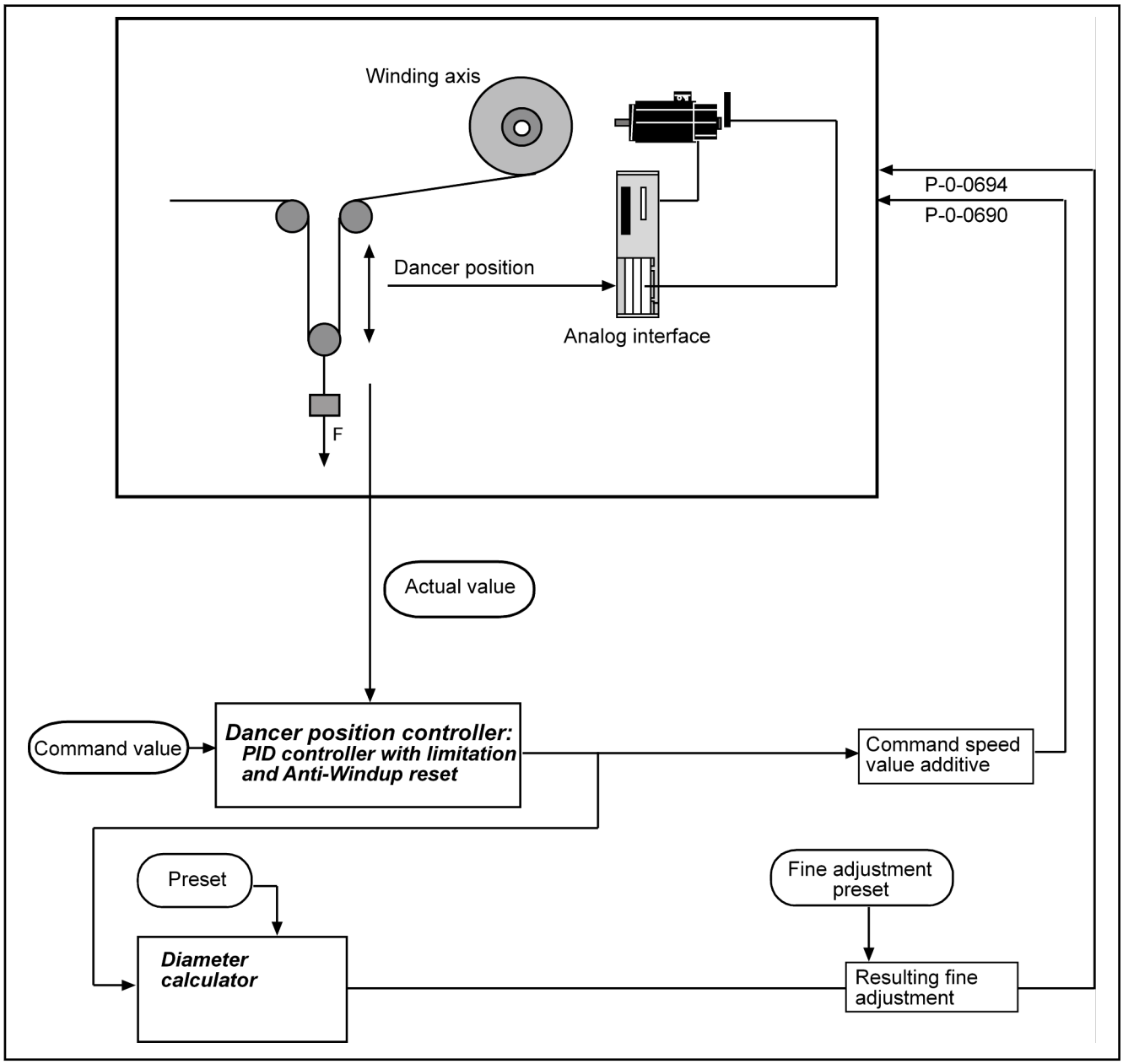


Fig.9-37: Functional diagram of the diameter calculator with dancer

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used. Additionally, the error codes of the function blocks MB_DancerControlType03 (see chapter 9.2.2 "MB_DancerControlType03 " on page 412), MB_DiameterCalculatorType03 (see chapter 9.3.2 "MB_DiameterCalculatorType03 " on page 420), MB_DiameterMeasurementType01 (see chapter 9.4.2 "MB_DiameterMeasurementType01 " on page 429), IL_PIDType01 from the RIL_LoopControl library (refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries") as well as MB_GetCyclicParameterHandle (see chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128) can be output.

RMB_TechWinder.library/MX_TechWinder.lib

ErrorID	Additional1	Additional2	Text
STATE_MACHINE_ERROR	16#1190	16#0020	Error in state machine

Fig.9-38: Error codes of the MB_WinderDancerCtrlType01 function block

Required hardware

- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware

- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software

- IndraWorks 12VRS for IndraMotion MLC or IndraMotion MLD

Required parameterization for the IndraMotion MLC control

The interpolation in the drive has to be enabled for the winding axis.

(Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **parameter P-0-0694 ("Gear ratio fine adjustment, process controller")** and **parameter P-0-0690 ("Additive velocity command value, process controller")** have to be entered in the cyclic SERCOS channel of the corresponding drive ("UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

The standstill window of the reference axis should be set as low as possible (e.g. 0.1 rpm). When there is a standstill message of the reference axis, the diameter calculator does not calculate!

Required parameterization for the IndraMotion MLD control

Interpolation in the drive has to be activated in the drive for the winding axis when using the IndraMotion MLD-S control with a superordinate IndraMotion MLC control (Real axes → Corresp. axis → Properties). This setting does not exist when using the IndraMotion MLD-M control.

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **parameter P-0-0694 ("Gear ratio fine adjustment, process controller")** and **parameter P-0-0690 ("Additive velocity command value, process controller")** have to be entered in the cyclic SERCOS channel of the corresponding drive ("UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

The real axes are parameterized in the parameterization mode in IndraWorks (the dialog is located in the project tree below the MLD node → AxisData).

The "AxisData structure" of the drive must be parameterized (P-0-1367 "PLC configuration" Bit 6 is set or dialog PLC configuration)

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

The standstill window of the reference axis should be set as low as possible (e.g. 0.1 rpm). When there is a standstill message of the reference axis, the diameter calculator does not calculate!

RMB_TechWinder.library/MX_TechWinder.lib

9.5.3 Commissioning the Diameter Calculator with Dancer

Drive control loop

It is assumed that the speed control loop of the winding drive was set. That means that the parameters S-0-0100 "Velocity controller proportional gain" and S-0-0101 "Velocity controller integral time" have to be set according to the application.

This section describes the commissioning of a diameter calculator with dancer. For this purpose, the following steps are reasonable:

- Parameterization of the dancer position controller
- Parameterization for the diameter calculator
- Initial startup of the winding axis

Commanding winding axis

The correct operation mode of the drive and the drive enable of the drive have to be set for the winding axis. This can be carried out in two different ways as follows:

- Set the "ModeSyncVel" operation mode by means of the AxisInterface (recommended)
- Use the PLCopen "MC_Power", "MC_GearIn", "MC_Stop" function blocks to command the axis

Parameterizing the dancer position controller

Since the function of the diameter calculator with dancer is built up on the function of the dancer position controller, the parameterization of the dancer position controller can be performed as described under in the chapter 9.2 Function Block for Closed-Loop Dancer Position Control, page 410.

Parameterizing the diameter calculator

The individual inputs of the diameter calculator MB_DiameterCalculator-Type03 are enabled as described before. A number of special aspects have to be taken into account when commissioning the diameter calculator with dancer.

Due to the parameterization of the dancer position controller, the inputs "FormatLength" and "FeedBackDancer" of the diameter calculator have been preset. Another variable, which affects the dancer, is the dancer constant "Dancer constant". This specifies the web distance with the full displacement of the dancer takes from negative to positive end, based on the unit of dancer adjustment. The "ReferenceDiameter" input should be linked to the currently calculated diameter "CurrentDiameter".

Since the calculated diameter directly affects the velocity of the winding axis via the resulting fine adjustment, a meaningful value has to be preset for the diameter prior to the start of the calculation when activating the diameter calculator. Thus, a diameter value, which is close to the actual diameter, has to be specified at the "PresetDiameter" input. With a rewinder, the value can be slightly lower and slightly higher with an unwinder.

Finally, the inputs "MinDiameter" and "MaxDiameter" can be set. The variables specified act on the outputs "MinAvDiamAchieved" and "MaxAvDiamAchieved". The respective values should be within the admissible diameter range defined by the inputs "HighLimitDiameter" and "LowLimitDiameter". Thus, the time for specific process steps, for example, for a splice or adjustment of the "DiamCalcRange", can be detected and displayed in the PLC, based on the specification of the minimum/maximum diameter.

☞ The ratio "Format length"/"CoreDiameter" may not be greater than 25.

Otherwise, the gear adjustment via the gear fine adjustment is insufficient!

Initial startup of the winding axis

The dancer position controller and the diameter calculator are subsequently activated. After having activated the closed-loop dancer position control, the dancer moves to its command position. As described before, this position should correspond to the central position of the dancer.

The phase positions of the winding axis and the reference axis are summed up and, depending on the value of the "DiamCalcRange" input, the diameter is regularly calculated and displayed as current diameter value at the "CurrentDiameter" output. After having activated the diameter calculator, "CurrentDiameter" initially displays the zero value until a diameter has been calculated for the first time. A preset has to be executed if another value, apart from zero, is used. The value "PresetDiameter" is applied as the current diameter value at a positive edge at "Preset".

The period between two calculations depends on the winding axis speed and the value at the "DiamCalcRange" input.

The diameter calculator affects the fine adjustment of the gear. The respective value is displayed at the "GearRatioFineAdjust" output.

9.6 Function Block for the Diameter Calculator with Open-Loop Tensile Stress Control or Closed-Loop Tensile Stress Control

9.6.1 Introduction and Overview

The function block provides two different diameter calculators:

- Diameter calculator with tensile stress control (open-loop)
  - corresponds to a diameter calculator without sensor
- Diameter calculator with tensile stress control (closed-loop)
  - corresponds to a diameter calculator with load cell

The diameter calculators are implemented in the MB_WinderTensionCtrlType02 function block.

The function block cyclically calculates the current winding diameter, the current moment of inertia of the winder, the corresponding velocity adaptation via the fine adjustment of the gear, an additive speed offset for tensioning the web of fabric and the required torque limit.

☞ When using this winding concept, the mechanical friction (motor coupling, gear...) should be kept as low as possible. The mechanical friction should always be lower than the smallest tensile torque that results at the minimum tension and the minimum winding diameter.

Diameter calculator with open-loop tensile stress control

The diameter calculator with open-loop tensile stress control is used on center winders. The drive of the winding axis generally operates at its torque limit. The torque limit depends on the current winding diameter and the tension set.

The tensile stress is generated by adjusting the torque of the winding axis. Additional measuring elements are not required.

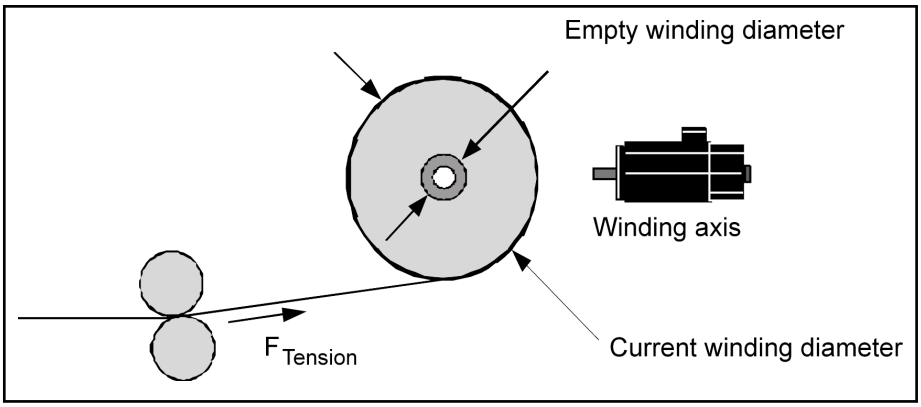


Fig.9-39: Diameter calculator with tensile stress control (open-loop)

Diameter calculator with closed- loop tensile stress control

The diameter calculator with closed-loop tensile stress control is used on center-driven winders. The drive of the winding axis generally operates at its torque limit. The torque limit depends on the current winding diameter and the tension set. The torque limit corresponds to the value of the motor torque required to rotate the winder at the defined speed. It consists of load torque, friction torque and acceleration torque.

The tension controller is provided as PI-controller. It contains the function block "IL_PIDType01" from the "RIL_LoopControl" library. The control variable of the controller and the current winding diameter determine the torque limit value of the drive. The command value of the tension should also be scaled and transferred to the controller output ("PreControlWeighting" not equal to zero). When the scaling is not zero, the tension for the torque creation consists of the controller output variable added to the scaled command value. Scaling to zero means that only the tension controller is active. Thus, the tension controller should only be operated additively (e.g. with a scaling of 1.0).

The actual tension value is measured directly with the help of a load cell and transferred to the tension controller as actual value via an analog channel or field bus. With the web tensioned, the diameter calculator can calculate the current winding diameter.

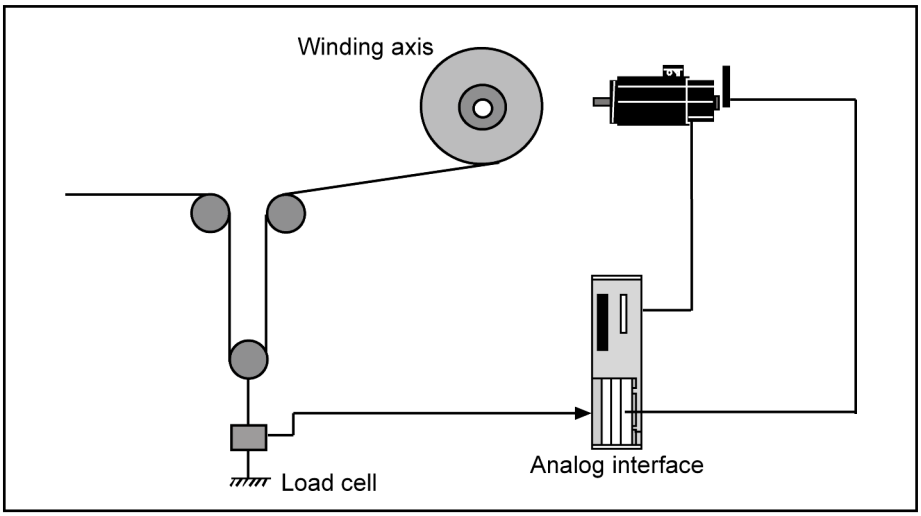


Fig.9-40: Diameter calculator with tensile stress control (closed-loop)

RMB_TechWinder.library/MX_TechWinder.lib

9.6.2 MB_WinderTensionCtrlType02

Brief Description

The function block cyclically calculates the current winding diameter, the current winder moment of inertia, the corresponding velocity adaptation via the fine adjustment of the gear, an additive speed offset for tensioning the web of fabric and the required torque limit.

Target variables of the function block are the parameters P-0-0694 "Gear ratio fine adjust, process controller" P-0-0690 "Additive velocity command value, process controller" and S-0-0092 "Bipolar torque/force limit value". These parameters have to be implemented in the cyclic IndraMotion MLC data containers.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechWinder.lib

Fig.9-41: Reference table of the MB_WinderTensionCtrlType02

Interface Description

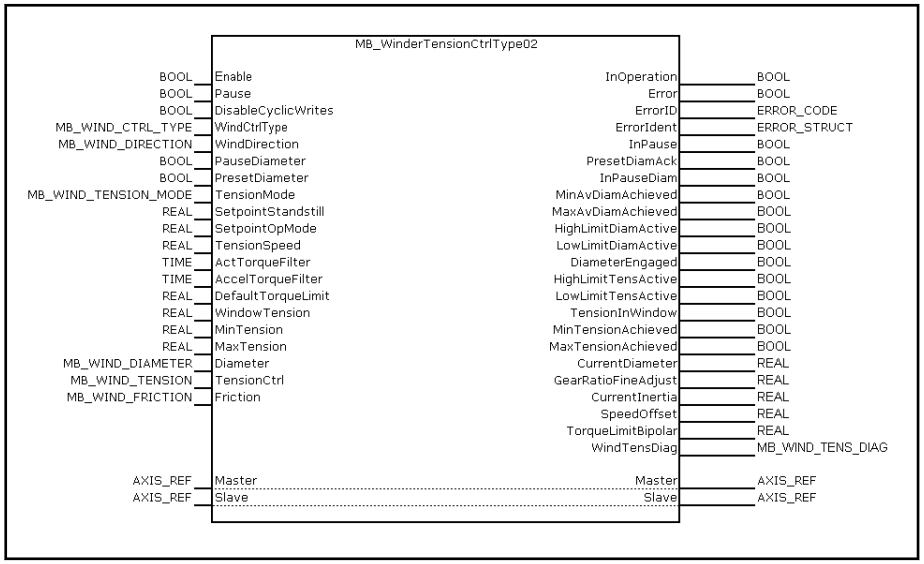


Fig.9-42: MB_WinderTensionCtrlType02 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	Master	AXIS_REF	Reference to master axis
	Slave	AXIS_REF	Reference to the slave axis (winding axis)
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	Pause	BOOL	Pausing the tension controller and the diameter calculator (values are frozen)
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	WindCtrlType	MB_WIND_CTRL_TYPE	Winding structure Open-loop tension control (without sensor) OPEN_LOOP (16#0000) Closed-loop tension control (load cell) CLOSED_LOOP (16#0001)
	WindDirection	MB_WIND_DIRECTION	Winding direction Rewinder REWIND (16#0000) Unwinder UNWIND (16#0001)
	PauseDiameter	BOOL	Pausing the diameter calculation Closed-loop/open-loop control of the torque limit remains active
	PresetDiameter	BOOL	CurrentDiameter is set to PresetVal, GearRatioFineAdj is adjusted accordingly (edge-controlled)
	TensionMode	MB_WIND_TENSION_MODE	Mode for tensioning the material web Tensioning with constant speed CONST_SPEED Tensioning at constant web velocity CONST_WEB_VEL
	SetpointStandstill	REAL	Specification of the standstill tension [N] (Changeover between standstill tension and operating tension by means of the standstill message from the drive).
	SetpointOpMode	REAL	Specifying the operating tension [N] (Changeover between standstill tension and operating tension by means of the standstill message from the drive).
	TensionSpeed	REAL	Tensioning speed in the case of an empty winder [min^{-1}]
	ActTorqueFilter	TIME	Filter time constant for the actual torque S-0-0084 "Actual torque/force value"
	AccelTorqueFilter	TIME	Filter time constant for acceleration torque
	DefaultTorqueLimit	REAL	Value for the bipolar torque limit after deactivation of "Enable" [%]
	WindowTension	REAL	Window for the actual tension value [%] in relation to the command tensile stress value (output "TensionInWindow")
	MinTension	REAL	Lower threshold for the actual tension value (only for a display with the binary output "MinTensionAck")
	MaxTension	REAL	Upper threshold for the actual tension value (only for a display with binary output "MaxTensionAck")
	Diameter	MB_WIND_DIAMETER	Structure for the diameter calculator
	TensionCtrl	MB_WIND_TENSION	Structure for tension controller
	Friction	MB_WIND_FRICTION	Structure for the friction torque

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	InPause	BOOL	Tension controller and diameter calculator paused
	PresetDiamAck	BOOL	Preset diameter was applied (is set as long as "Preset" is pending)
	InPauseDiam	BOOL	Diameter calculator paused
	MinAvDiamAchieved	BOOL	Minimum diameter reached (based on the averaged diameter)
	MaxAvDiamAchieved	BOOL	Maximum diameter reached (based on the averaged diameter)
	HighLimitDiamActive	BOOL	Currently calculated diameter has reached the upper limit (diameter value is discarded)
	LowLimitDiamActive	BOOL	Currently calculated diameter has reached the lower limit (diameter value is discarded)
	DiameterEngaged	BOOL	Diameter "engaged" TRUE: engaged FALSE: disengaged
	HighLimitTensActive	BOOL	Output of the tension controller has achieved the upper limit
	LowLimitTensActive	BOOL	Output of the tension controller has achieved the lower limit
	TensionInWindow	BOOL	Actual tension control value is within the tension control window
	MinTensionAchieved	BOOL	Tensile stress is below the "MinTension" input
	MaxTensionAchieved	BOOL	Tensile stress is above the input "MaxTension"
	CurrentDiameter	REAL	Currently calculated diameter [mm]
	GearRatioFineAdjust	REAL	Resulting fine adjustment of the gear for speed adaptation of the drive [%]
	CurrentInertia	REAL	Current moment of inertia of the winding material + mechanics (based on the motor side [kg m^2]
	SpeedOffset	REAL	Tensioning speed [rpm]
	TorqueLimitBiploar	REAL	Bipolar torque limit [%]
	WindTensDiag	MB_WIND_TENS_DIAG	Current control deviation of the dancer controller

Fig.9-43: Interface variables of the MB_WinderTensionCtrlType02 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
WindCtrlType	MB_WIND_CTRL_TYPE				Rising edge at "Enable"
WindDirection	MB_WIND_DI-RECTION				Rising edge at "Enable"
PauseDiameter	BOOL			FALSE	Continuous
PresetDiameter	REAL			FALSE	Continuous
TensionMode	MB_WIND_TEN-SION_MODE				Rising edge at "Enable"
SetpointStandstill	REAL	0.0	n.def	0.0	Continuous
SetpointOpMode	REAL	n.def	n.def	0.0	Continuous
TensionSpeed	REAL	0.0	n.def	0.0	Continuous
ActTorqueFilter	TIME	0 s	n.def	0 s	Continuous
AccelTorqueFilter	TIME	0 s	n.def	0 s	Continuous
DefaultTorqueLi-mit	REAL	0.0	n.def	100.0	Continuous
WindowTension	REAL	0.0	n.def	0.0	Continuous
MinTension	REAL	0.0	n.def	0.0	Continuous
MaxTension	REAL	0.0	n.def	0.0	Continuous
Diameter	MB_WIND_DIAM-ETER				depending on the structure element
TensionCtrl	MB_WIND_TEN-SION				Continuous
Friction	MB_WIND_FRIC-TION				Continuous

Fig.9-44: Min./max. and default values of the MB_WinderTensionCtrlType02 inputs

MB_WIND_CTRL_TYPE

Name	Value	Description
OPEN_LOOP	0	Winding type "open-loop", without sensor
CLOSED_LOOP	1	Winding type "closed-loop", with load cell

Fig.9-45: Elements of the MB_WIND_CTRL_TYPE enumeration type

MB_WIND_DIRECTION

RMB_TechWinder.library/MX_TechWinder.lib

Name	Value	Description
REWIND	0	Rewinder
UNWIND	1	Unwinder

Fig.9-46: Elements of the MB_WIND_DIRECTION enumeration type

MB_WIND_TENSION_MODE

Name	Value	Description
CONST_SPEED	0	Tensioning method: Constant tensioning speed
CONST_WEB_VEL	1	Tensioning method: Constant web velocity

Fig.9-47: Elements of the MB_WIND_TENSION_MODE enumeration type

MB_WIND_DIAMETER

Name	Type	Min. value	Max. value	Default value	Effective	Description
PresetVal	REAL	> 0.0	n.def	100.0	Continuous	Default value for the preset diameter [mm]
CoreVal	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"	Diameter of the core [mm]
CalcRange	REAL	0.0	n.def	360.0	Continuous	Range for diameter calculation [degrees] (load-sided)
RepeatLength	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"	Format length [mm] (1 revolution of the reference axis unwound)
Webwidth	REAL	0.0	n.def	0.0	Rising edge at "Enable"	Web width [mm] (is required for the moment of inertia)
Density	REAL	0.0	n.def	1.0	Rising edge at "Enable"	Material density [kg/dm^3] (is required for the moment of inertia)
AverageValue	UINT	0	20	1	Continuous	Number of diameter values to be averaged [-]
HighLimit	REAL	LowLimit	n.def	2000.0	Continuous	Upper limit for the diameter [mm] The diameter is limited to this value
LowLimit	REAL	> 0.0	HighLimit	100.0	Continuous	Lower limit for the diameter [mm] The diameter does not fall below this value
MinDiameter	REAL	> 0.0	n.def	100.0	Continuous	Minimum diameter [mm]
MaxDiameter	REAL	> 0.0	n.def	2000.0	Continuous	Maximum diameter [mm]

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective	Description
MechanicInertia	REAL	0.0	n.def	0.0	Continuous	Moment of inertia of the mechanics [kg m^2] based on the motor side (is required for the resulting moment of inertia)
DancerConstant (is only required for a winder with dancer)	REAL	0.0	n.def	0.0	Continuous	Displacement of the dancer, based on the unit of dancer adjustment [mm/-] (is only required for a winder with dancer)
MeasuredVal	REAL	0.0	n.def	0.0	Continuous	Measured diameter [mm]
PivotVelocity	REAL	0.0	n.def	0.0	Continuous	Pivot velocity (for relative motion of the winding axis) [mm/s]
DiamCalcMode	MB_WIND_DIAM_CALC_MODE				Rising edge at "Enable"	Diameter calculation mode CALC_DIAM_MANUAL Diameter calculation depending on CalcRange FIRST_REV_AUTO_CALC Diameter calculation is executed after every 5 degrees for the first revolution. Thereafter, it depends on the CalcRange
DiamSource	BOOL				Rising edge at "Enable"	Diameter determination: FALSE: Diameter is calculated TRUE: Diameter is measured (MeasuredVal)

Fig.9-48: Interface description of the MB_WIND_DIAMETER data structure

MB_WIND_TENSION

Name	Type	Min. value	Max. value	Default value	Effective	Description
PControl	REAL	0.0	n.def	0.0	Continuous	Kp-gain of the tension controller
IControl	REAL	0.0	n.def	0.0	Continuous	Tn integral action time of the tension controller
FeedbackVal	REAL	0.0	n.def	0.0	Continuous	Actual value of the tension
PreControl-Weighting	REAL	0.0	n.def	1.0	Continuous	Scaling of the command value addition

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective	Description
HighLimit	REAL	0.0	n.def	0.0	Continuous	Upper limit of the tension controller
LowLimit	REAL	0.0	n.def	0.0	Continuous	Lower limit of the tension controller

Fig.9-49: Interface description of the MB_WIND_TENSION data structure

MB_WIND_FRICTION

Name	Type	Min. value	Max. value	Default value	Effective	Description
StandstillTorque	REAL	0.0	n.def	0.0	Continuous	Friction torque at standstill
MaxTorque	REAL	0.0	n.def	0.0	Continuous	Friction torque at S-0-0091 "Velocity limit value, bipolar"
MinTorque	REAL	0.0	n.def	0.0	Continuous	Minimum friction torque
MinTorqueVelocity	REAL	0.0	n.def	0.0	Continuous	Velocity at minimum friction torque

Fig.9-50: Interface description of the MB_WIND_FRICTION data structure

MB_WIND_TENS_DIAG

Name	Type	Min. value	Max. value	Default value	Effective	Description
TensionCtrlOutput	REAL	n.def	n.def	0.0	Continuous	Output of the tension controller [N]
TorqueLimit	REAL	n.def	n.def	0.0	Continuous	Signed torque limit [%]
AccelerationTorque	REAL	n.def	n.def	0.0	Continuous	Current acceleration torque [%]
FrictionTorque	REAL	n.def	n.def	0.0	Continuous	Current friction torque [%]

Fig.9-51: Interface description of the MB_WIND_TENS_DIAG data structure

Signal-time diagram

The following signal-time diagram represents the selected signals of the function block. With a positive edge at the "Enable" input, which is signaled at the "InOperation" output, the function block is operated. Operational readiness can be interrupted by an error acknowledged at the "Error" output. The internal controller process can be paused by the "Pause" input. "Pause" is signaled at the "InPause" output. The "CurrentDiameter" output is configured with a preset value ("PresetVal" input in the structure MB_WIND_DIAMETER) via a positive edge at the "PresetDiameter" input, irrespective of the "Pause" input. However, it is recommended to switch the function block to "Pause". By doing so, the diameter preset can be performed in a more controlled way. The setting of the diameter is displayed at the output "PresetDiamAck".

RMB_TechWinder.library/MX_TechWinder.lib

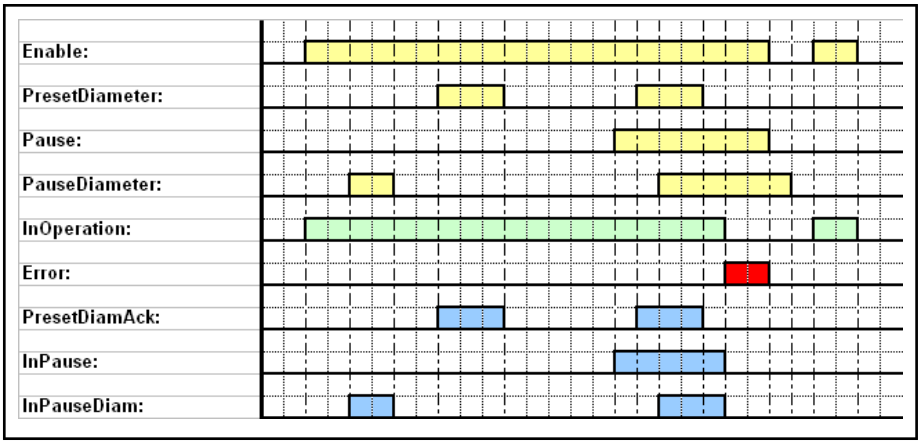


Fig.9-52: Signal-time diagram for selected inputs and outputs of the function block MB_WinderTensionCtrlType01

Functional Description

Diameter calculator with open-loop tensile stress control

The functional scope of the diameter calculator with open-loop tensile stress control includes:

- Tensioning the untensioned winding material
- Speed-dependent friction compensation
- Compensation of acceleration torque
- Separate parameterization of the operating tensile stress and standstill tensile stress and
- Calculation of diameter-dependent variables

  (using the MB_DiameterCalculatorType03 function block)

Functional diagram of the diameter calculator with open-loop tensile stress control

The following functional diagram schematically explains the functions of the diameter calculator with open-loop tensile stress control.

RMB_TechWinder.library/MX_TechWinder.lib

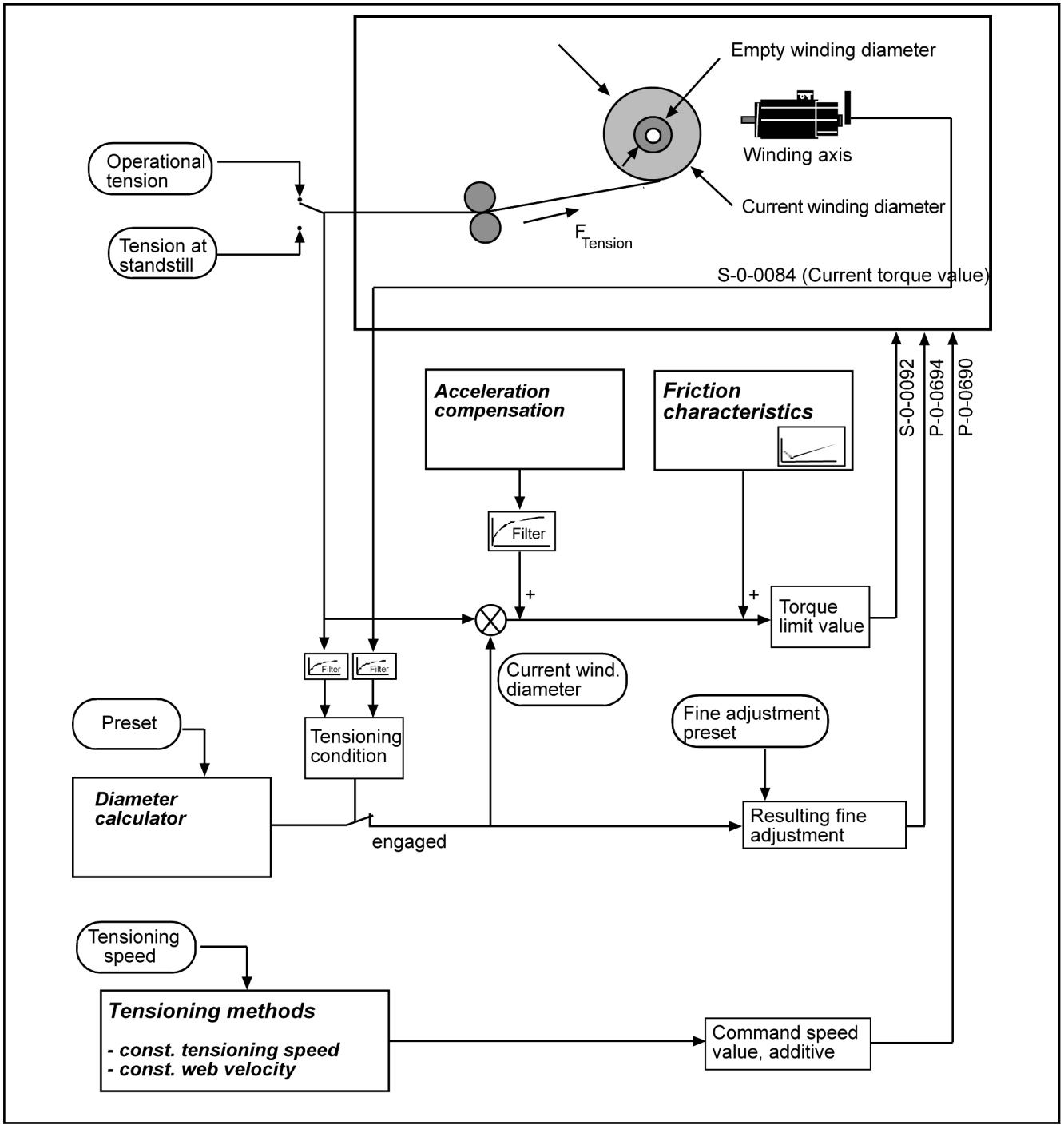


Fig.9-53: Functional diagram of the winder with tensile stress control (open-loop)

Diameter calculator with closed-loop tensile stress control

The functional scope of the diameter calculator with closed-loop tensile stress control includes:

- Tensioning of the untensioned winding material
- Speed-dependent friction compensation
- Acceleration torque compensation,

RMB_TechWinder.library/MX_TechWinder.lib

- Separate parameterization of the operating tensile stress and standstill tensile stress
- PI-controller for tension (via IL_PIDType01) and
- Calculating diameter-dependent variables (via MB_DiameterCalculator-Type02)

Functional diagram of the diameter calculator with closed-loop tensile stress control

The following functional diagram schematically explains the functions of the diameter calculator with closed-loop tensile stress control.

RMB_TechWinder.library/MX_TechWinder.lib

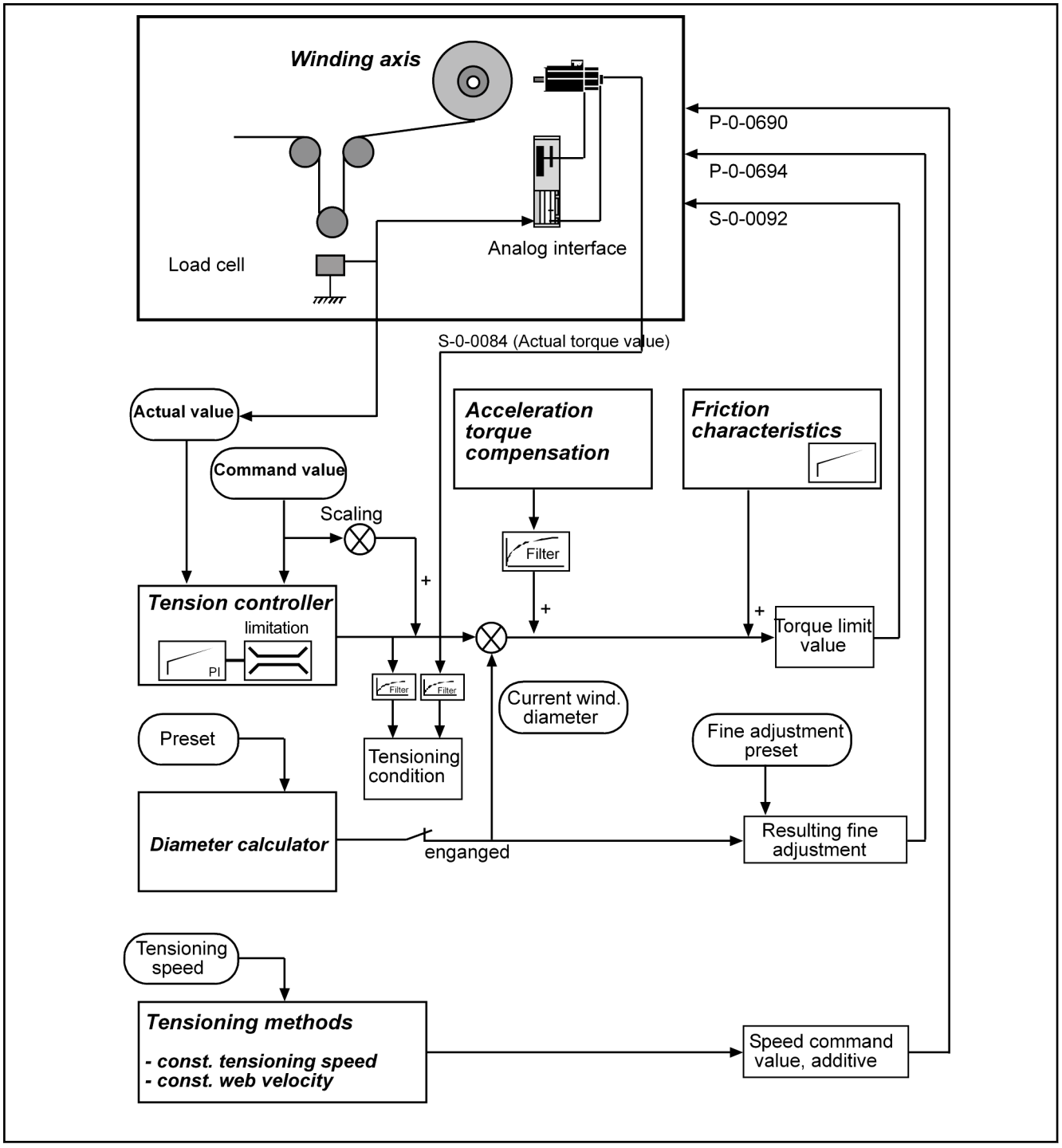


Fig.9-54: Functional diagram of the winder with tensile stress control (closed-loop)

Direction of rotation of the winding drive, rewinder/unwinder

With the input of the function block, the winder type is set to a rewinder or unwinder type. The direction of rotation of the drive has to be set either by the scaling parameters or by the polarity of the master drive.

☞ The direction of rotation of the master axis of the winding drive (that is the master axis of the function block MC_GearIn or the master axis of the AxisInterface) must be positive which means a positive speed command value. The winding drive must be inverted using the polarity parameters.

Diameter calculation

The torque for the winding drive results from the product of the tensile stress and the winding radius. Thus, the radius or the diameter of the winder has to be continuously determined. An internal diameter calculator (function block MB_DiameterCalculatorType03, see chapter 9.3.2 "MB_DiameterCalculatorType03 " on page 420) is always incorporated in the function block.

The diameter calculation will only become active when the winding material is tensioned and no standstill message of the reference axis is active. The calculation will be performed cyclically depending on the speed or the position of the winding axis. For diagnostic purposes, the current diameter can be read at the function block.

The calculated winding diameter directly influences the resulting fine adjustment of the gear. If the web tension is too low or negative (formation of loops), the diameter calculator is interrupted since no useful diameter calculation can be performed any longer under these circumstances. If the actual torque of the drive is more than 25% below the command value, the diameter calculator stops calculating (engaging condition).

☞ If the current diameter is to be kept after the control has been switched off, the "CurrentDiameter" output has to be copied to a remanent variable. When the control is switched on again, a preset can be performed again with this value.

☞ The ratio "Format length"/"CoreDiameter" may not be greater than 25.

Otherwise, the gear adjustment via the gear fine adjustment is insufficient!

Tensioning the winding material

When enabling the winding calculator ("Enable" = TRUE), the winding material is initially tensioned with the tensioning speed. To avoid the formation of loops in the winding material, the master axis should not be rotating. The winding material is tensioned if the torque command value of the drive exceeds a predefined limit and if the relative deviation between the torque command value and the actual torque value is lower than a defined limit.

Two different procedures can be used for tensioning the winding material:

- Tensioning at constant speed and
- Tensioning at constant web velocity.

When tensioning the winding material at a constant speed, the tensioning speed is maintained on a constant level. Thus, the web velocity increases with an increasing winding diameter.

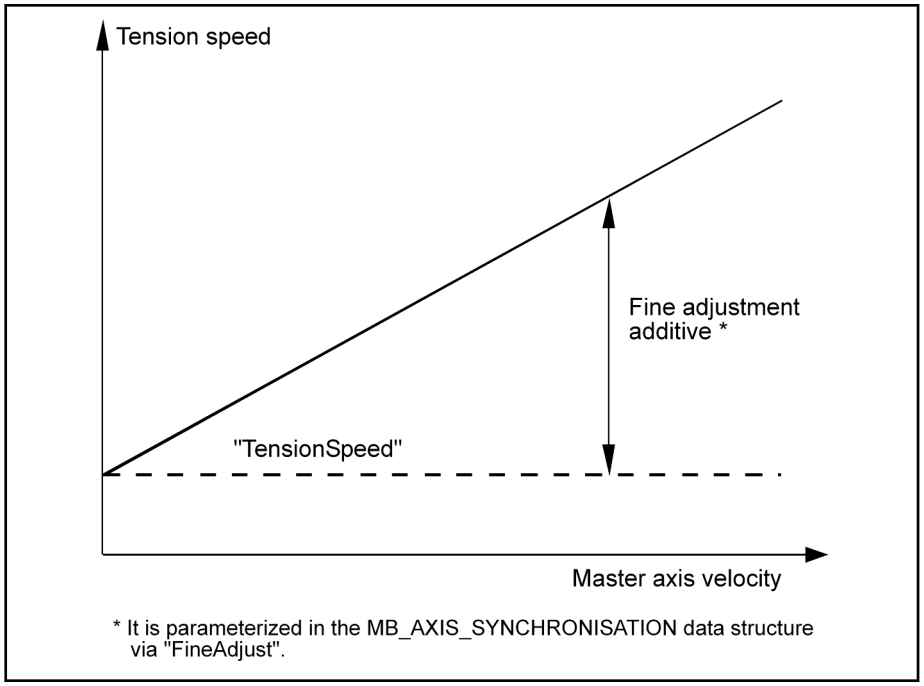


Fig.9-55: Tensioning at constant tensioning speed

When tensioning the winding material at a constant web velocity, the tensioning speed is reduced reciprocally to the winding diameter. The calculation of the winding axis speed is based on the tensioning speed of an empty winder. This has to be transferred according to the function block.

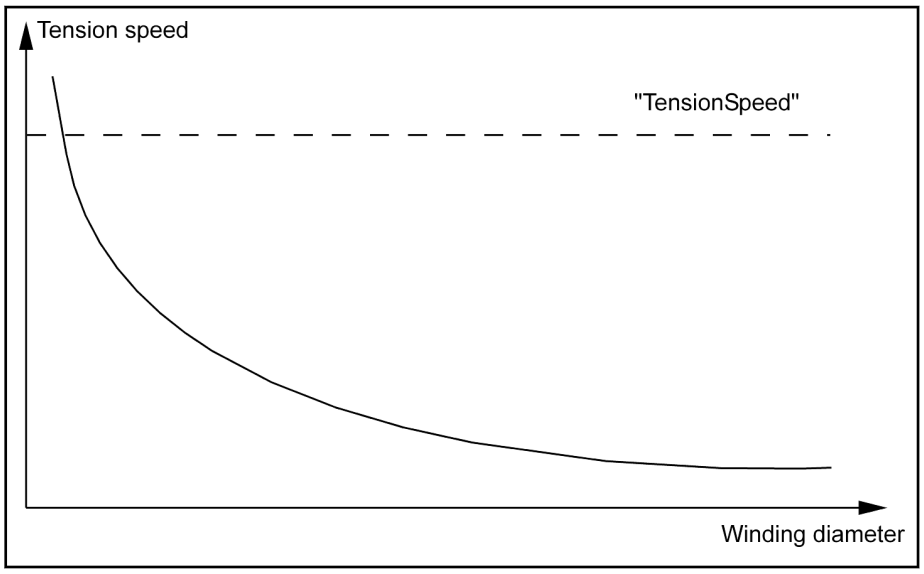


Fig.9-56: Tensioning with constant web velocity.

Friction compensation

Friction torque compensation is performed using different inputs. Approaching the friction behavior is defined as follows:

The friction is initially reduced from the standstill friction torque up to a specific speed. The friction torque increases linearly with the winding axis speed up to the maximum speed (S-0-0091 "Bipolar velocity limit value" or A-0-0032 "Velocity limit value, positive" or A-0-0033 "Velocity limit value, negative"). This behavior also applies to negative speeds. The curve is mirrored at the y-

axis (friction torque). With this friction torque behavior, the effect of the friction can be counteracted by a compensation.

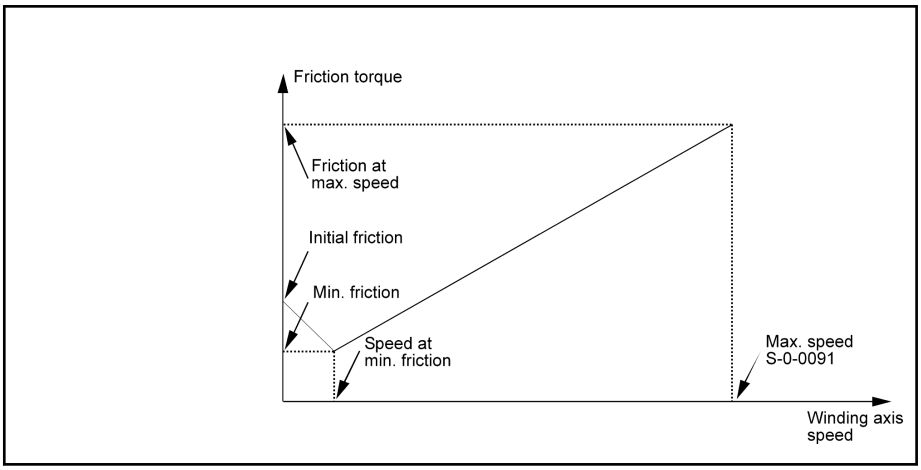


Fig.9-57: Friction torque compensation

Acceleration torque compensation

An additional acceleration torque is required for the acceleration of the winding axis. This torque depends on the winding diameter. By specifying certain material and fabric web data, the inertia of the winder is calculated according to the diameter. The calculated acceleration torque can also be filtered.

Splice

After a splice, the winding diameter generally changes abruptly. The last calculated internal value for the winding diameter does no longer correspond to the current diameter. A preset of the winding diameter is required to calculate the diameter correctly. The input value for the preset should correspond approximately to the actual value of the winding diameter.

☞ If the preset is not performed, malfunctions can occur. The preset should therefore be completed before the diameter calculator becomes active again.

The value for the preset should be selected as follows:

- With a rewinder, it should be slightly lower than or equal to the actual winding diameter and
- with an unwinder, it has to be slightly higher or equal to the actual winding diameter

With a flying splice, it has to be ensured that the circumferential velocity of the new roll corresponds to the current roll. This should be implemented respectively in the PLC project of the IndraLogic.

Filtering the actual torque value

At low tensions, an unfiltered actual torque value can result in a disengaging diameter calculator. A filtering process should therefore be performed and the corresponding input should be activated. Therefore, the actual torque and the command torque of the drive are compared (e.g. 20ms).

Additional leading and lagging of the winding axis

Additional leading or lagging via the fine adjustment of the gear(presetting fine adjustment "FineAdjust") can be set via the structure variable "qrAxisCtrl[AxisNumber].Admin.SyncMode.FineAdjust" of the MB_AXIS_SYNCHRONISATION data structure of the ML_TechInterface library. For a re-

winder, a low positive value (e. g. 5 %) should be set. A low negative value should be set for an unwinder. This results in an improved startup behavior of the respective winding direction.

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used. Additionally, the error codes of the function blocks MB_DiameterCalculatorType03 (see chapter 9.3.2 "MB_DiameterCalculatorType03 " on page 420) and IL_PIDType01 from the RIL_LoopControl library (refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries") as well as MB_GetCyclicParameterHandle (see chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128) can be output.

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#11B0	16#20	Error in state machine
INPUT_RANGE_ERROR	16#11B1	16#01	Diameter.MechanicInertia < 0.0
INPUT_RANGE_ERROR	16#11B1	16#02	TensionCtrl.PreControlWeighting < 0.0
INPUT_RANGE_ERROR	16#11B1	16#03	MaxTension < 0.0
INPUT_RANGE_ERROR	16#11B1	16#04	MaxTension < MinTension
INPUT_RANGE_ERROR	16#11B1	16#05	Friction.StandstillTorque < 0.0
INPUT_RANGE_ERROR	16#11B1	16#06	Friction.MaxTorque < 0.0
INPUT_RANGE_ERROR	16#11B1	16#07	Friction.MinTorque < 0.0
INPUT_RANGE_ERROR	16#11B1	16#08	Friction.MinVelocity < 0.0
INPUT_RANGE_ERROR	16#11B1	16#09	WindowTension < 0.0
INPUT_RANGE_ERROR	16#11B1	16#0A	MinTension < 0.0
INPUT_RANGE_ERROR	16#11B1	16#0B	MaxTension < 0.0
INPUT_RANGE_ERROR	16#11B1	16#0C	Winding axis not in "BB"
INPUT_RANGE_ERROR	16#11B1	16#0D	Incorrect torque scaling (A-0-0050) Scaling for the motor torque has to be based on the motor side
INPUT_RANGE_ERROR	16#11B1	16#0E	Axis number of the slave axis incorrect
INPUT_RANGE_ERROR	16#11B1	16#0F	Axis number of the master axis incorrect
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated

Fig.9-58: Error codes of the MB_WinderTensionCtrlType01 function block

Required hardware

- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware

- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software

- IndraWorks 12VRS for IndraMotion MLC or IndraMotion MLD

Special features for the MLC

The interpolation in the drive has to be enabled for the winding axis.
(Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **parameter P-0-0694 ("Gear ratio fine adjustment, process controller")** and the **parameter P-0-0690 ("Additive velocity command value, process controller" and S-0-0092 "Bipolar torque/force limit value"** have to be entered in the cyclic SERCOS channel of the corresponding drive ("UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear"). The parameters A-0-0032 "Velocity limit value, positive" and A-0-0033 "Velocity limit value, negative" should be set according to the maximum speed of the winding drive (this value has to be in the parameter S-0-0091 "Velocity limit value, bipolar").

The standstill window of the reference axis should be set as low as possible (e.g. 0.1 rpm). When there is a standstill message of the reference axis, the diameter calculator does not calculate!

Special features for the MLD

Interpolation in the drive has to be activated in the drive for the winding axis when using the IndraMotion MLD-S control with a superordinate IndraMotion MLC control (Real axes → Corresp. axis → Properties). This setting does not exist when using the IndraMotion MLD-M control.

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The **parameter P-0-0694 ("Gear ratio fine adjustment, process controller")** and the **parameter P-0-0690 ("Additive velocity command value, process controller" and S-0-0092 "Bipolar torque/force limit value"** have to be entered in the cyclic SERCOS channel of the corresponding drive ("UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

The real axes are parameterized in the parameterization mode in IndraWorks (the dialog is located in the project tree below the MLD node → AxisData).

The "AxisData structure" of the drive must be parameterized (P-0-1367 "PLC configuration" Bit 6 is set or dialog PLC configuration).

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear"). The parameter S-0-0091 "Bipolar velocity limit value" should be set in accordance with the maximum speed of the winding drive.

The standstill window of the reference axis should be set as low as possible (e.g. 0.1 rpm). When there is a standstill message of the reference axis, the diameter calculator does not calculate!

9.6.3 Commissioning the Diameter Calculator with Open-Loop Tensile Stress Control and Closed-Loop Tensile Stress Control

Drive control loop

It is assumed that the speed control loop of the winding drive was set. That means that the parameters S-0-0100 "Velocity controller proportional gain" and S-0-0101 "Velocity controller integral time" have to be set according to the application.

The commissioning of the diameter calculator is described in the following. Commissioning can be performed in seven steps:

1. Determining the winder type
2. Determining the direction of rotation of the winding axis
3. Determining of the friction torque at standstill
4. Determining of the friction torque at maximum speed

5. Determining of the minimum friction torque
6. final settings and
7. initial startup of the winding axis

Commanding winding axis

The correct operation mode of the drive and the drive enable of the drive have to be set for the winding axis. This can be carried out in two different ways as follows:

- Set the "ModeSyncVel" operation mode by means of the AxisInterface (recommended)
- Use the PLCopen "MC_Power", "MC_GearIn", "MC_Stop" function blocks to command the axis

Determining winder type

First, it has to be determined which winder type is available or is to be used. Then, the winding direction is defined at the "WindDirection" input (TRUE: Unwind, FALSE: Rewind). Furthermore, the diameter calculator is selected with the "WindCtrlType" input (TRUE: Diameter calculator with closed-loop tensile stress control, FALSE: Diameter calculator with open-loop tensile stress control).

Determining direction of rotation of winding axis

The direction of rotation of the winder can be set with the help of the drive parameter P-0-0-108 "Master drive polarity" or the axis parameter A-0-2798 "Polarity of master axis position". If the winding axis does not rotate in the desired direction, the parameter P-0-0108 or the axis parameter A-0-2798 has to be reset accordingly. The polarity of the master axis position is only checked when enabling the function block ("Enable" = TRUE). Thus, modifying the mentioned parameters during the operation of the function block does not have any effect.

Alternatively, the direction of rotation can be changed via the scaling type of the winding drive. This can be done by switching the sign of the position/acceleration data and the torque/force data (parameters S-0-0-055, A-0-0029, S-0-0043, S-0-0085).

The following figure shows the general relations between rewinders and unwinders.

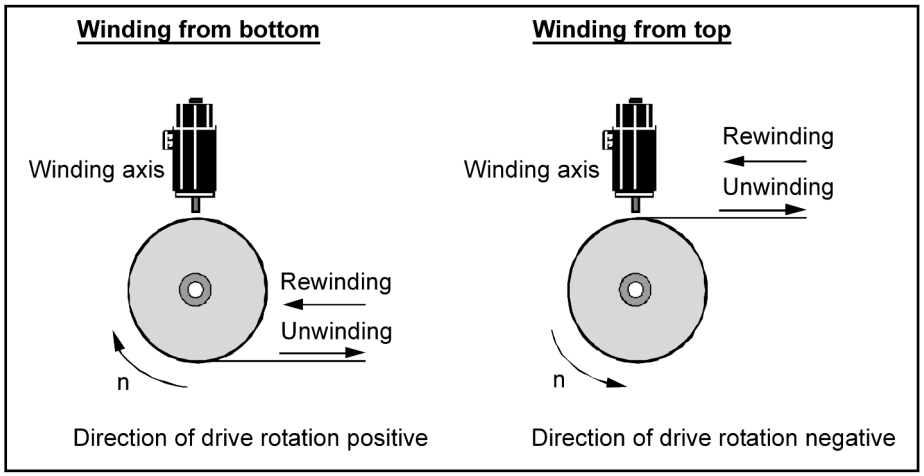


Fig.9-59: Principles of rewinders and unwinders

Determining friction torque at standstill

1. The "standstill friction torque" ("StandstillTorque" variable in the data structure MB_WIND_TENSION) is determined when the master axis is at standstill and the winder is empty (core only). The winder function

block is disabled. For the winding axis (slave axis), select the operation mode "Velocity control" (in the AxisInterface of the ML_TechInterface "MODE_VEL" library). The axis can also be controlled using the PLCopen function block MC_MoveVelocity. 10 rpm can be set as command value.

2. The value of parameter P-0-0109 "Peak torque/force limit" is set to zero. The axis has to stop now. The value for this parameter is incremented until the drive starts to rotate. The last value corresponds to the friction torque at standstill.
3. The axis enable "MODE_VEL" is canceled (also by using the PLCopen function block "MC_Stop").

   The parameter P-0-0109 "Peak torque/force limit" is reset to its previous value (e.g. 400%).

   The friction torque setting at standstill is thus completed.

Determining friction torque at maximum speed

1. The "friction torque at maximum speed" ("MaxTorque" variable in the data structure MB_WIND_TENSION) is determined when the master axis is at standstill and with an empty winder (core only). The function block is disabled.
2. The operating mode "Velocity control" has to be selected for the winding axis (slave axis) (in the AxisInterface of the ML_TechInterface "MODE_VEL", optionally PLCopen function block).

   The command value for the selected operation mode has to be increased incrementally up to the value of the parameter S-0-0091 "Bipolar velocity limit value". The value for the parameter S-0-0091 "Bipolar velocity limit value" has to be entered according to application and motor data.

3. In the next step, the value of the parameter P-0-0109 "Peak torque/force limit" is decreased incrementally (e.g. from 100% towards 0%) until the drive speed also starts to decrease. The speed of the drive can be traced using the drive parameter S-0-0040 "Actual velocity value" or the axis parameter A-0-0102 "Actual velocity value". The last value of P-0-0109"Peak torque/force limit" corresponds to the maximum friction torque during maximum speed.
4. The axis enable "MODE_VEL" is canceled (also by using the PLCopen function block "MC_Stop").

   The parameter P-0-0109 "Peak torque/force limit" is reset to its previous value (e.g. 400%).

   Setting the friction torque maximum speed "MaxTorque" is completed.

Determining minimum friction torque

The minimum friction torque is determined using the deceleration test with an empty winder. The winding drive is decelerated from a constant velocity (e.g. 50 min^{-1}) to standstill. At the same time, the drive parameters S-0-0080 "Torque/force command value" (axis parameter A-0-0038 "Torque/force limit value, bipolar") and S-0-0040 "Actual velocity value" (axis parameter A-0-0102 "Actual velocity value") are recorded with an oscilloscope or with the trace function in IndraWorks.

The minimum value is read from the torque curve and can be transferred to the structure MB_WIND_FRICTION as "MinTorque". The corresponding speed is entered as "MinTorqueVelocity" in MB_WIND_FRICTION.

Final settings

Finally, the remaining input variables are specified in the following table.

Input variable	Note/setting
SetpointOpMode	Setting the desired operating tension
SetpointStandstill	Setting the standstill tension
TensionMode	Selecting the tensioning speed characteristics: • TRUE: Tensioning with constant web velocity • FALSE: Tensioning with constant web velocity
Offset in fine adjustment	In the data structure MB_AXIS_SYNCHRONISATION of the ML_TechInterface, the fine adjustment can be preset via the variable "FineAdjust" with a value depending on the winding direction. This variable is cyclically processed if "EnableCyclicScanning" is set to TRUE. As described before, the winding direction is determined with "WindDirection". For an unwinder, a positive value is used for "FineAdjust". For a rewinder, a negative value is used. These values should be ≤ 5 %
ActTorqueFilter	If the actual torque is to be filtered (as measured value), a value not equal to zero has to be applied at this input of the function block as a time constant. This value acts on the default parameter S-0-0084 "Actual torque/force value"
AccelTorqueFilter	If the acceleration torque is to be filtered, a value unequal to zero has to be applied at this input of the function block as time constant

Fig.9-60: Final determination of the remaining input variables

Initial startup of the winding axis

The winding material is placed in an untensioned state (slightly sagging). The winding axis has to be commanded either via the AxisInterface or via the PLCopen function blocks. The diameter calculator is activated by setting the "Enable" input. The winding material should now be tensioned with the predefined tensioning speed ("TensionSpeed" input). The "SpeedOffset" output provides the tensioning speed.

The process variable window "WindowTension" can be set to a value from 10 - 20%.

The master axis is enabled with a comparatively low speed (e. g. 10 min^{-1}). After a number of revolutions, the "DiameterEngaged" output has to be set. It is thus displayed that the diameter calculation is active ("engaged"). This depends on the web tension or the actual torque at the winding axis and the torque limit (drive parameter S-0-0092 "Bipolar torque/force limit value" or axis parameter A-0-0038 "Torque/force limit value, bipolar"). The torque/force limit is displayed at the "TorqueLimitBipolar" output.

When the diameter calculator is engaged, the standstill tension and the operating tension can be checked. Therefore, the value "CurrentTension" from the data structure ML_WIND_TENS_DIAG is monitored. In this data structure, the current torque limit value "TorqueLimit", the current acceleration torque

"Acceleration torque" and the currently applied friction torque "FrictionTorque" can also be monitored.

The function block provides different process variables at its outputs. The calculated current diameter of the winder ("CurrentDiameter" output) should correspond to the actual diameter of the winder. Slight deviations can be disregarded. Furthermore, the fine adjustment of the gear for the speed adaptation of the drive axis of the winder ("GearRatioFineAdjust" output), the current moment of inertia of the winder ("CurrentInertia" output) and the tensioning speed ("SpeedOffset" output) are displayed.

Monitoring the torque limit value

During the initial control startup, a drive warning "E2056 Torque limit value = 0" can occur if the winder function block is not yet enabled.

In order to prevent this, the value for the parameter S-0-0092 "Bipolar torque/force" can be written with a value unequal to zero such as 100% in the start-up phase of the PLC program (e.g. warm start) of the corresponding cyclic data containers.

If the warning "E8260 Torque - force command value limit active" occurs when the function block is enabled, this warning can be eliminated by setting bit 4 of the parameter P-0-0556, "Axis controller configuration".

In rare cases (e.g. multiple F8078 "Error in the speed control loop"), it can be necessary to disable the velocity controller monitoring (set bit 1 of the parameter P-0-0556 "Axis controller configuration").

9.7 Function Block for the Winder Diameter with Closed-Loop Tensile Stress Control and Speed Correction

9.7.1 Introduction and Overview

The diameter calculator with closed-loop tensile stress control and speed correction is used on center-driven winders. The drive of the winding axis generally does NOT operate at its torque limit. Due to safety reasons, a torque limit is calculated that can additionally be set (the limit should be above the required motor torque and should not limit the drive in normal operation).

This winder is mainly used for elastic, expandable material.

The actual tension value is measured directly with the help of a load cell and transferred to the tension controller as actual value via an analog channel or field bus. With the web tensioned, the diameter calculator can calculate the current winding diameter.

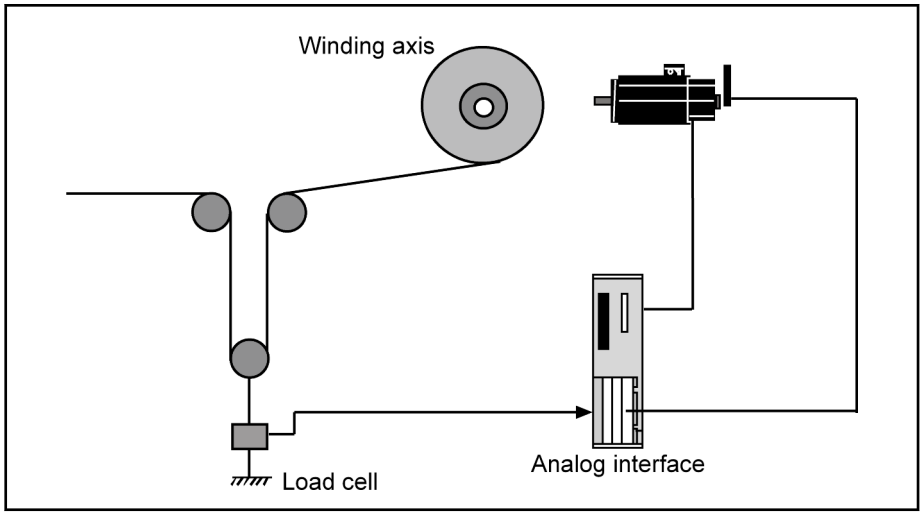


Fig.9-61: Diameter calculator with tensile stress control (closed-loop)

9.7.2 MB_WinderTensionCtrlSpeedType01

Brief Description

The function block cyclically calculates the current winding diameter, the current winder moment of inertia, the corresponding velocity adaptation via fine adjustment of the gear, an additive speed offset for tensioning the web and a parameterizable torque limit.

Target variables of the function block are the parameters P-0-0694 "Gear ratio fine adjustment, process controller" P-0-0690 "Additive velocity command value, process controller", S-0-0081 "Additive torque/force command value" and S-0-0092 "Bipolar torque/force limit value".

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library

Fig.9-62: Reference table of the MB_WinderTensionCtrlSpeedType01

Interface Description

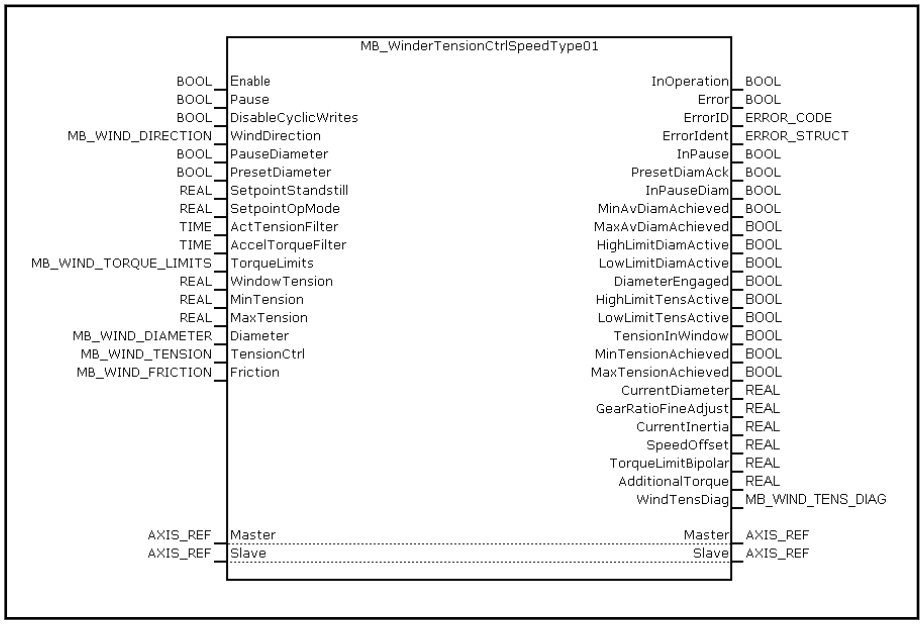


Fig.9-63: MB_WinderTensionCtrlSpeedType01 function block

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
VAR_IN_OUT	Master	AXIS_REF	Reference to master axis
	Slave	AXIS_REF	Reference to the slave axis (winding axis)
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	Pause	BOOL	Pausing the tension controller and the diameter calculator (values are frozen)
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output
	WindDirection	MB_WIND_DIRECTION	Winding direction Rewinder REWIND (16#0000) Unwinder UNWIND (16#0001)
	PauseDiameter	BOOL	Pausing the diameter calculation
	PresetDiameter	BOOL	CurrentDiameter is set to PresetVal, GearRatioFineAdjust is adjusted accordingly (edge-controlled)
	SetpointStandstill	REAL	Specification of the standstill tension [N] (Switching between standstill tension and operating tension by means of the standstill message of the drive)
	SetpointOpMode	REAL	Specifying the operating tension [N] (Changeover between standstill tension and operating tension by means of the standstill message from the drive).
	ActTensionFilter	TIME	Filter time constant for actual tension control
	AccelTorqueFilter	TIME	Filter time constant for acceleration torque
	TorqueLimits	MB_WIND_TORQUE_LIMITS	Structure for bipolar torque limit (S-0-0092)
	WindowTension	REAL	Window for the actual tension control value [%] in relation to the command tensile stress value ("TensionInWindow")
	MinTension	REAL	Lower threshold for the actual tension value (only for a display with the binary output "MinTensionAck")
	MaxTension	REAL	Upper threshold for the actual tension control value (only for a display with binary output "MaxTensionAck")
	Diameter	MB_WIND_DIAMETER	Structure for the diameter calculator
	TensionCtrl	MB_WIND_TENSION	Structure for tension controller
	Friction	MB_WIND_FRICTION	Structure for the friction torque
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Type	Comment
	InPause	BOOL	Tension controller and diameter calculator paused
	PresetDiamAck	BOOL	Preset diameter was applied (is set as long as "Preset" is pending)
	InPauseDiam	BOOL	Diameter calculator paused
	MinAvDiamAck	BOOL	Minimum diameter achieved (relates to the averaged diameter)
	MaxAvDiamAck	BOOL	Maximum diameter achieved (relates to the averaged diameter)
	HighLimitDiamAck	BOOL	Currently calculated diameter has reached the upper limit (diameter value is discarded)
	LowLimitDiamAck	BOOL	Currently calculated diameter has reached the lower limit (diameter value is discarded)
	DiameterEngaged	BOOL	Diameter "engaged" TRUE: engaged FALSE: disengaged
	HighLimitTensActive	BOOL	Output of the tension controller has achieved the upper limit
	LowLimitTensActive	BOOL	Output of the tension controller has achieved the lower limit
	TensionInWindow	BOOL	Actual tension control value is within the tension control window
	MinTensionAchieved	BOOL	Tensile stress is below the "MinTension" input
	MaxTensionAchieved	BOOL	Tensile stress is above the input "MaxTension"
	CurrentDiameter	REAL	Currently calculated diameter [mm]
	GearRatioFineAdjust	REAL	Resulting fine adjustment of the gear for speed adaptation of the drive [%]
	CurrentInertia	REAL	Current total moment of inertia (based on the motor side [kg/m^2]
	SpeedOffset	REAL	Tensioning speed [rpm]
	TorqueLimitBipolar	REAL	Bipolar torque limit [%]
	AdditionalTorque	REAL	Additive torque [%]
	WindTensDiag	MB_WIND_TENS_DIAG	Diagnostic structure

Fig.9-64: Interface variables of the MB_WinderTensionCtrlSpeedType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
WindDirection	MB_WIND_DI-RECTION			FALSE	Rising edge at "Enable"
PauseDiameter	BOOL			FALSE	Continuous
PresetDiameter	BOOL			FALSE	Continuous
SetpointStandstill	REAL	0.0	n.def.	0.0	Continuous
SetpointOpMode	REAL	0.0	n.def.	0.0	Continuous
ActTensionFilter	TIME	0 s	n.def.	0 s	Continuous
AccelTorqueFilter	TIME	0 s	n.def.	0 s	Continuous
TorqueLimits	MB_WIND_TOR-QUE_LIMITS				Continuous
WindowTension	REAL	0.0	n.def.	0.0	Continuous
MinTension	REAL	0.0	n.def.	0.0	Continuous
MaxTension	REAL	0.0	n.def	0.0	Continuous
Diameter	MB_WIND_DIAM-ETER				Depending on the structure element
TensionCtrl	MB_WIND_TEN-SION				Continuous
Friction	MB_WIND_FRIC-TION				Continuous

Fig.9-65: Min./max. and default values of the MB_WinderTensionCtrlSpeed-Type01 inputs

MB_WIND_DIRECTION

Element	Value	Description
REWIND	0	Rewinder
UNWIND	1	Unwinder

Fig.9-66: Elements of the MB_WIND_DIRECTION enumeration type

MB_WIND_TORQUE_LIMITS

Name	Type	Min. value	Max. value	Default value	Effective	Description
DefaultTorqueLi-mit	REAL	0.0	400.0	100.0	Continuous	Value for the bipolar torque limit after deactivation of "Enable" [%]
TorqueLimitFactor	REAL	0.0	400.0	50.0	Continuous	Factor by which the calcula-ted TorqueLimit is in-creased [%]
MinTorqueLimit	REAL	0.0	400.0	10.0	Continuous	Minimum TorqueLimit by which the calculated Tor-queLimit is increased [%]

Fig.9-67: Interface description of the data structure of the MB_WIND_TOR-QUE_LIMITS

RMB_TechWinder.library/MX_TechWinder.lib

MB_WIND_DIAMETER

Name	Type	Min. value	Max. value	Default value	Effective	Description
PresetVal	REAL	> 0.0	n.def	100.0	Continuous	Default value for the preset diameter [mm]
CoreVal	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"	Diameter of the core [mm]
CalcRange	REAL	0.0	n.def	360.0	Continuous	Range for diameter calculation [degrees]
RepeatLength	REAL	> 0.0	n.def	100.0	Rising edge at "Enable"	Format length [mm] (1 revolution of the reference axis unwound)
Webwidth	REAL	0.0	n.def	0.0	Rising edge at "Enable"	Web width [mm] Is required for the moment of inertia
Density	REAL	0.0	n.def	1.0	Rising edge at "Enable"	Material density [kg/dm^3] Is required for the moment of inertia
AverageValue	UINT	0	20	1	Continuous	Number of diameter values to be averaged [-]
HighLimit	REAL	LowLimit	n.def	2000.0	Continuous	Upper limit for the diameter [mm]
LowLimit	REAL	> 0.0	HighLimit	100.0	Continuous	Lower limit for the diameter [mm]
MinDiameter	REAL	> 0.0	n.def	100.0	Continuous	Minimum diameter [mm]
MaxDiameter	REAL	> 0.0	n.def	2000.0	Continuous	Maximum diameter [mm]
MechanicInertia	REAL	0.0	n.def	0.0	Continuous	Moment of inertia of the mechanics [kg/m^2] based on the motor side Is required for the resulting moment of inertia
DancerConstant	REAL	0.0	n.def	0.0	Continuous	Displacement of the dancer, based on the unit of dancer adjustment [mm/-] Is only required with a winder with dancer
MeasuredVal	REAL	0.0	n.def	0.0	Continuous	Measured diameter [mm]
PivotVelocity	REAL	0.0	n.def	0.0	Continuous	Pivot velocity (for relative motion of the winding axis) [mm/s]

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective	Description
DiamCalcMode	MB_WIND_DIAM_CALC_MODE				Rising edge at "Enable"	CALC_DIAM_MANUAL diameter calculation depending on CalcRange FIRST_REV_AUTO_CALC Diameter calculation is performed every 5 degrees for the first revolution and thereafter depending on the CalcRange
DiamSource	BOOL			False	Rising edge at "Enable"	Diameter determination: FALSE: Diameter is calculated TRUE: Diameter is measured (MeasuredVal)

Fig.9-68: Interface description of the MB_WIND_DIAMETER data structure

MB_WIND_DIAM_CALC_MODE

Element	Value	Description
CALC_DIAM_MANUAL	0	Diameter calculation depending on CalcRange
FIRST_REV_AUTO_CALC	1	Diameter calculation is executed after every 3 degrees for the first revolution. Thereafter, it depends on the CalcRange

Fig.9-69: Elements of the MB_WIND_DIAM_CALC_MODE enumeration type

MB_WIND_TENSION

Name	Type	Min. value	Max. value	Default value	Effective	Description
PControl	REAL	0.0	n.def	0.0	Continuous	Kp-gain of the tension controller
IControl	REAL	0.0	n.def	0.0	Continuous	Tn integral action time of the tension controller
FeedbackVal	REAL	0.0	n.def	0.0	Continuous	Actual value of the tension
PreControl-Weighting	REAL	0.0	n.def	1.0	Continuous	Scaling of the command value addition
HighLimit	REAL	0.0	n.def	0.0	Continuous	Upper limit of the tension controller
LowLimit	REAL	0.0	n.def	0.0	Continuous	Lower limit of the tension controller

Fig.9-70: Interface description of the MB_WIND_TENSION data structure

MB_WIND_FRICTION

Name	Type	Min. value	Max. value	Default value	Effective	Description
StandstillTorque	REAL	0.0	n.def	0.0	Continuous	Friction torque at standstill
MaxTorque	REAL	0.0	n.def	0.0	Continuous	Friction torque at S-0-0091 Velocity limit value, bipolar

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective	Description
MinTorque	REAL	0.0	n.def	0.0	Continuous	Minimum friction torque
MinTorqueVelocity	REAL	0.0	n.def	0.0	Continuous	Velocity at minimum friction torque

Fig.9-71: Interface description of the MB_WIND_FRICTION data structure

MB_WIND_TENS_DIAG

Structure element	Type	Min. value	Max. value	Default value	Effective	Description
TensionCtrlOutput	REAL	n.def	n.def	0.0	Continuous	Output of the tension controller [N]
TorqueLimit	REAL	n.def.	n.def	0.0	Continuous	Signed torque limit [%]
AccelerationTorque	REAL	n.def	n.def	0.0	Continuous	Current acceleration torque [%]
FrictionTorque	REAL	n.def	n.def	0.0	Continuous	Current friction torque [%]
CurrentTension	REAL	0.0	n.def	0.0	Continuous	Current actual tension control value [N]

Fig.9-72: Interface description of the MB_WIND_TENS_DIAG data structure

Signal-time diagram

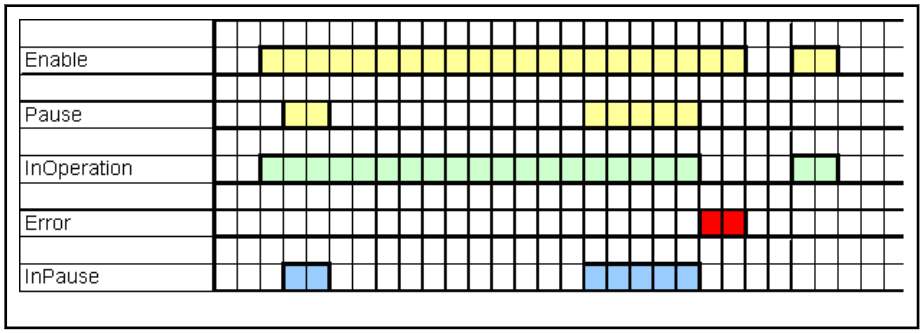


Fig.9-73: Signal-time diagram of MB_WinderTensionCtrlSpeedType01

Functional Description

The function block is a diameter calculator with open-loop tensile stress control that usually does not operate at the torque limit.

The function block cyclically calculates the current winding diameter, the current moment of inertia of the winder, the corresponding velocity adaptation via the fine adjustment of the gear, an additive speed offset for tensioning the web of fabric and the command torque limit value for precontrol.

The MB_DiameterCalculatorType03 and MB_DiameterMeasurementType01 function blocks are used internally.

Target variables of the function block: parameter P-0-0694 "Gear ratio fine adjustment, process controller" P-0-0690 "Additive velocity command value, process controller", S-0-0081 "Additive torque/force command value" and S-0-0092 "Bipolar torque/force limit value".

Due to a changing winder diameter, the velocity is changed using the fine adjustment of the gear, the tension controller has an additive effect on the command velocity value, the calculated torque (torque for tension, friction torque, acceleration torque) is precontrolled via the additive command torque value.

Due to safety reasons, the torque limit value on the currently calculated torque command value is multiplied with a scaling factor and is set with an additional additive value. In the normal winder mode, the torque limitation should not be effective and the winding drive should be operated in a speed-controlled way.

RMB_TechWinder.library/MX_TechWinder.lib

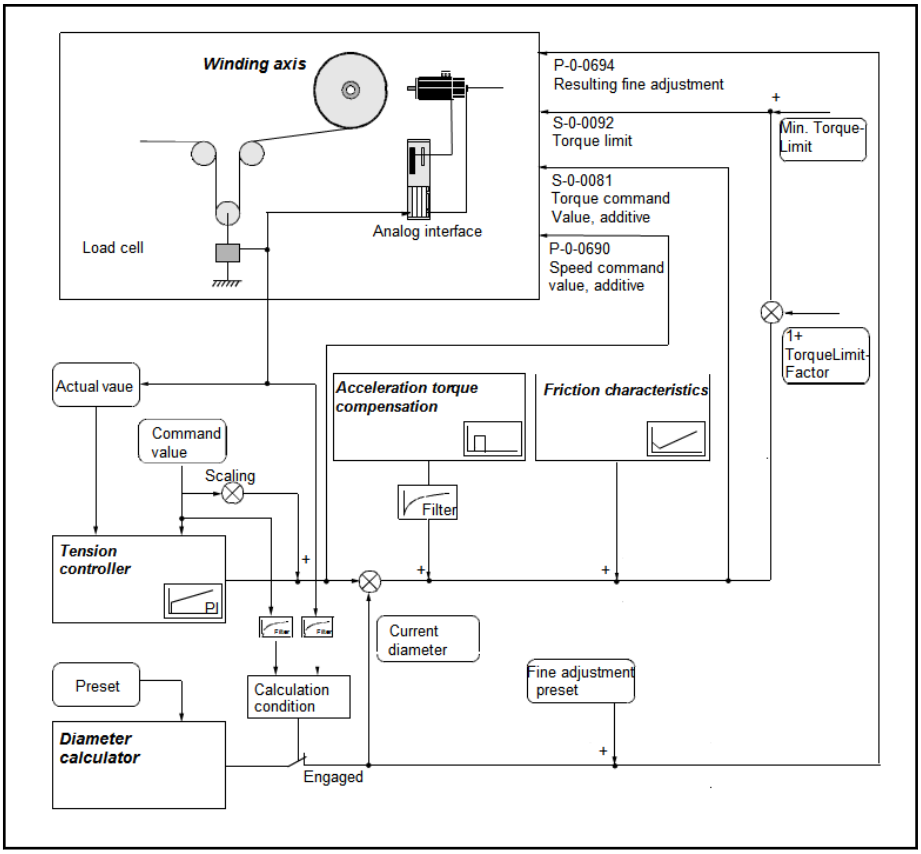


Fig.9-74: Schematic representation of the control for a winding axis

Direction of rotation of the winding drive, rewinder/unwinder

With the enumeration of the "MB_WindDirection" function block, the winder type is set to a rewinder or unwinder type. The direction of rotation of the drive has to be set either by the scaling parameters or by the polarity of the master drive.

The direction of rotation of the master axis of the winding drive (that is the master axis of the function block MC_GearIn or the master axis of the AxisInterface) must be positive which means a positive speed command value. The winding drive must be inverted using the polarity parameters.

Diameter calculation

The torque for the winding drive results from the product of the tensile stress and the winding radius. Thus, the radius or the diameter of the winder has to be continuously determined. An internal diameter calculator (function block MB_DiameterCalculatorType03, see chapter 9.3.2 "MB_DiameterCalculator-Type03 " on page 420) is always incorporated in the function block.

The diameter calculation will only become active when the winding material is tensioned and no standstill message of the reference axis is active. The calculation will be performed cyclically depending on the speed or the position of the winding axis. For diagnostic purposes, the current diameter can be read at the function block.

The calculated winding diameter directly influences the resulting fine adjustment of the gear. If the web tension is too low or negative (formation of loops), the diameter calculator is interrupted since no useful diameter calcula-

tion can be performed any longer under these circumstances. If the actual tension control of the drive is more than 25% below the command value, the diameter calculator stops calculating (engaging condition).

☞ If the current diameter is to be kept after the control has been switched off, the "CurrentDiameter" output has to be copied to a remanent variable. When the control is switched on again, a preset can be performed again with this value.

☞ The ratio "Format length"/"CoreDiameter" may not be greater than 25.

Otherwise, the gear adjustment via the gear fine adjustment is insufficient!

Tensioning the winding material

When enabling the winding calculator ("Enable" = TRUE), the winding material is initially tensioned with the additive speed offset. To avoid the formation of loops in the winding material, the master axis should not be rotating. The winding material is tensioned if the command tension value of the tension controller exceeds a defined minimum value and if the relative deviation between the command tensile stress value and the actual torque value is lower than a defined limit.

Friction compensation

Friction torque compensation is performed using different inputs. Approaching the friction behavior is defined as follows:

The friction is initially reduced from the standstill friction torque up to a specific speed. The friction torque increases linearly with the winding axis speed up to the maximum speed (S-0-0091 "Bipolar velocity limit value" or A-0-0032 "Velocity limit value, positive" or A-0-0033 "Velocity limit value, negative"). This behavior also applies to negative speeds. The curve is mirrored at the y-axis (friction torque). With this friction torque behavior, the effect of the friction can be counteracted by a compensation.

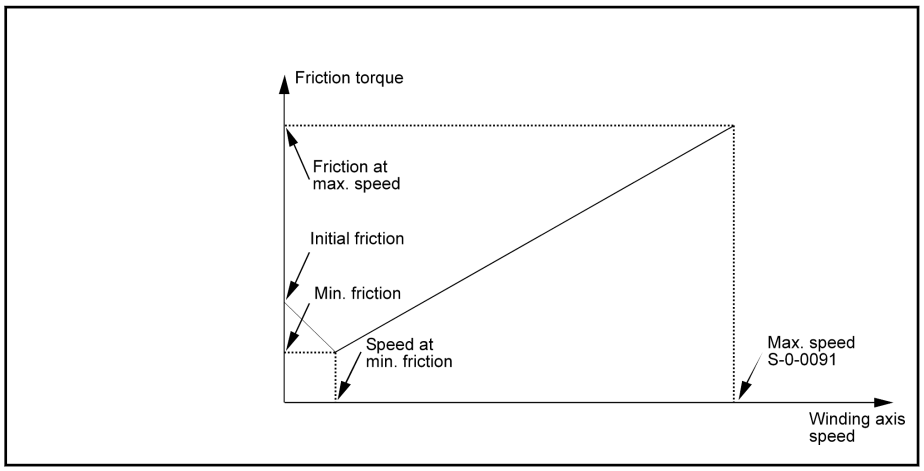


Fig.9-75: Friction torque compensation

Acceleration torque compensation

An additional acceleration torque is required for the acceleration of the winding axis. This torque depends on the winding diameter. By specifying certain material and fabric web data, the inertia of the winder is calculated according to the diameter. The calculated acceleration torque can also be filtered.

Torque limit and precontrol

The drive of the winding axis generally does NOT operate at its torque limit. Due to safety reasons, a calculated torque with a settable additive value is set for the torque limit. (this value should be above the required motor torque and not limit the drive in normal mode).

A torque limit proportional to the tension and the respective winder diameter is calculated cyclically.

$$M_{Limit} \sim F_{Tension} * r_{Winder} + M_{Accel.} + M_{Friction}$$

Fig.9-76: Torque limit

This cyclic value for the torque limit is precontrolled as additive torque command value.

Furthermore, this value can be increased by a proportional ratio (TorqueLimitFactor) and an additive ratio (MinTorqueLimit) so that the torque limit does not become active during normal winder operation.

$$M_{ResLimit} \sim M_{Limit} * (1 + TorqueLimitFactor) + MinTorqueLimit$$

Fig.9-77: Resulting torque limit

The resulting torque limit corresponds to the TorqueLimitbipolar output. By default, the calculated torque limit is increased by a TorqueLimitFactor of 50% and a minimum TorqueLimit of 10%.

During tensioning it can be practical to set the torque limit lower (e.g. TorqueLimitFactor 10%), if tensioning takes place with too much torque. The TorqueLimitFactor should be increased again after tensioning is completed.

Splice

After a splice, the winding diameter generally changes abruptly. The last calculated internal value for the winding diameter does no longer correspond to the current diameter. A preset of the winding diameter is required to calculate the diameter correctly. The input value for the preset should correspond approximately to the actual value of the winding diameter.

☞ If the preset is not performed, malfunctions can occur. The preset should therefore be completed before the diameter calculator becomes active again.

The value for the preset should be selected as follows:

- With a rewinder, it should be slightly lower than or equal to the actual winding diameter and
- with an unwinder, it has to be slightly higher or equal to the actual winding diameter

With a flying splice, it has to be ensured that the circumferential velocity of the new roll corresponds to the current roll. This should be implemented respectively in the PLC project of the IndraLogic.

Filtering the actual tension control value

At low tensions, an unfiltered actual tension control value can result in a disengaging diameter calculator. A filtering process should therefore be performed and the corresponding input should be activated (e.g 20ms).

Additional leading and lagging of the winding axis

RMB_TechWinder.library/MX_TechWinder.lib

Additional leading or lagging via the fine adjustment of the gear(presetting fine adjustment "FineAdjust") can be set via the structure variable "qrAxisCtrl[AxisNumber].Admin.SyncMode.FineAdjust" of the MB_AXIS_SYNCHRONISATION data structure of the ML_TechInterface library. For a rewinder, a low positive value (e. g. 5 %) should be set. A low negative value should be set for an unwinder. This results in an improved startup behavior of the respective winding direction.

Error Handling

The following table lists and describes the error codes of the function block. The error codes from the "F_RELATED table (16#0170)" are used. Additionally, the error codes of the function blocks MB_DancerControlType03 (see chapter 9.2.2 "MB_DancerControlType03 " on page 412), MB_DiameterCalculatorType03 (see chapter 9.3.2 "MB_DiameterCalculatorType03 " on page 420), MB_DiameterMeasurementType01 (see chapter 9.4.2 "MB_DiameterMeasurementType01 " on page 429), IL_PIDType01 from the RIL_LoopControl library (refer to the documentation "Rexroth IndraWorks 12VRS Basic Libraries") as well as MB_GetCyclicParameterHandle (see chapter 5.9.5 "MB_GetCyclicParameterHandle" on page 128) can be output.

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#1120	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#1121	16#0001	Diameter.MechanicInertia < 0.0
INPUT_RANGE_ERROR	16#1121	16#0002	TensionCtrl.PreControlWeighting < 0.0
INPUT_RANGE_ERROR	16#1121	16#0003	MaxTension < 0.0
INPUT_RANGE_ERROR	16#1121	16#0004	MaxTension < MinTension
INPUT_RANGE_ERROR	16#1121	16#0005	Friction.StandstillTorque < 0.0
INPUT_RANGE_ERROR	16#1121	16#0006	Friction.MaxTorque < 0.0
INPUT_RANGE_ERROR	16#1121	16#0007	Friction.MinTorque < 0.0
INPUT_RANGE_ERROR	16#1121	16#0008	Friction.MinVelocity < 0.0
INPUT_RANGE_ERROR	16#1121	16#0009	WinderTension < 0.0
INPUT_RANGE_ERROR	16#1121	16#000A	MinTension < 0.0
INPUT_RANGE_ERROR	16#1121	16#000B	MaxTension < 0.0
INPUT_RANGE_ERROR	16#1121	16#000C	Incorrect torque scaling (A-0-0050 or S-0-0086), scaling for motor torque has to be on the motor side
INPUT_RANGE_ERROR	16#1121	16#000E	Axis number of the slave axis incorrect
INPUT_RANGE_ERROR	16#1121	16#000F	Axis number of the master axis incorrect
INPUT_RANGE_ERROR	16#1121	16#0010	SetpointStandstill < 0.0
INPUT_RANGE_ERROR	16#1121	16#0011	SetpointOpMode < 0.0
INPUT_RANGE_ERROR	16#1121	16#0012	DefaultTorqueLimit < 0.0
INPUT_RANGE_ERROR	16#1121	16#0013	MinTorqueLimit < 0.0
INPUT_RANGE_ERROR	16#1121	16#0014	TorqueLimitFactor < 0.0
INPUT_RANGE_ERROR	16#1122	16#0001	Winding axis not in "BB"
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware

RMB_TechWinder.library/MX_TechWinder.lib

ErrorID	Additional1	Additional2	Description
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated
RESSOURCE_ERROR	16#0012	16#00xx	AxisRef.CntrlNo incorrect

Fig.9-78: Error codes of the MB_WinderTensionCtrlSpeedType01 function block

Special features of IndraMotion MLC

The interpolation in the drive has to be enabled for the winding axis. (Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

The parameter P-0-0694 "Gear ratio fine adjustment, process controller" and parameter P-0-0690 "Additive velocity command value, process controller" and S-0-0092 "Torque/force limit value bipolar" and S-0-0081 "Torque/force command value, additive" have to be entered in the cyclic SERCOS channel of the corresponding drive ("UserCmdDataX"). The real axes are parameterized in IndraWorks in parameterization mode.

An actual existing mechanical gear ratio from the drive axis to the winding axis has to be entered in the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 "Output revolutions of load gear". The parameters A-0-0032 "Velocity limit value, positive" and A-0-0033 "Velocity limit value, negative" should be set according to the maximum speed of the winding drive (this value can be read out in the parameter S-0-0091 "Velocity limit value, bipolar").

The standstill window of the reference axis has to be set to a value that is as low as possible (e.g. 0.1 rpm), as the calculated diameter is stopped in case of a standstill message of the reference axis.

9.7.3 Commissioning the Winder with Tension Control and Speed Correction

Drive control loop

It is assumed that the speed control loop of the winding drive was set. That means that the parameters S-0-0100 "Velocity controller proportional gain" and S-0-0101 "Velocity controller integral time" have to be set according to the application.

The commissioning of the diameter calculator is described in the following. Commissioning can be performed in seven steps:

1. Determining the winder type
2. Determining the direction of rotation of the winding axis
3. Determining of the friction torque at standstill
4. Determining of the friction torque at maximum speed
5. Determining of the minimum friction torque
6. final settings and
7. initial startup of the winding axis

Commanding winding axis

The correct operation mode of the drive and the drive enable of the drive have to be set for the winding axis. This can be carried out in two different ways as follows:

- Set the "ModeSyncVel" operation mode by means of the AxisInterface (recommended)
- Use the PLCopen "MC_Power", "MC_GearIn", "MC_Stop" function blocks to command the axis

Determining winder type

First, it has to be determined which winder type is available or is to be used. Then, the winding direction is defined at the "WindDirection" input (TRUE: Unwind, FALSE: Rewind).

Determining direction of rotation of winding axis

The direction of rotation of the winder can be set with the help of the drive parameter P-0-0-108 "Master drive polarity" or the axis parameter A-0-2798 "Polarity of master axis position". If the winding axis does not rotate in the desired direction, the parameter P-0-0108 or the axis parameter A-0-2798 has to be reset accordingly. The polarity of the master axis position is only checked when enabling the function block ("Enable" = TRUE). Thus, modifying the mentioned parameters during the operation of the function block does not have any effect.

Alternatively, the direction of rotation can be changed via the scaling type of the winding drive. This can be done by switching the sign of the position/acceleration data and the torque/force data (parameters S-0-0-055, A-0-0029, S-0-0043, S-0-0085).

The following figure shows the general relations between rewinders and unwinders.

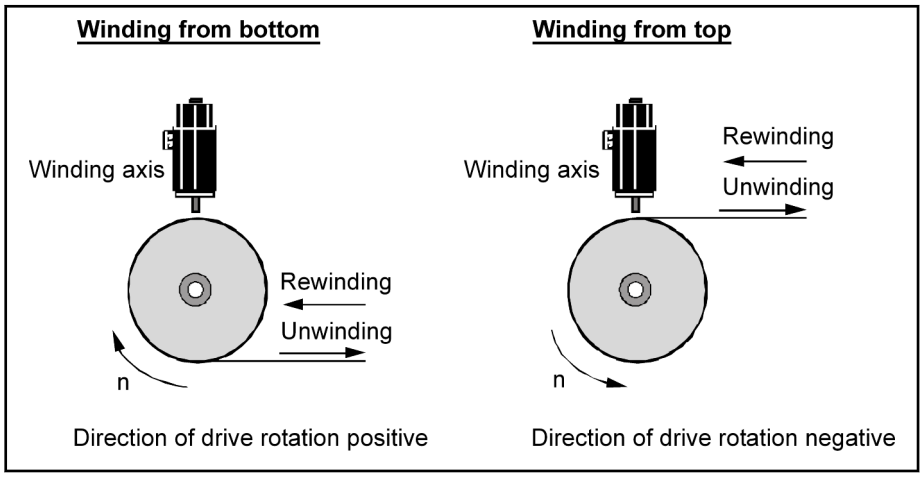


Fig.9-79: Principles of rewinders and unwinders

Determining friction torque at standstill

1. The "standstill friction torque" ("StandstillTorque" variable in the data structure MB_WIND_TENSION) is determined when the master axis is at standstill and the winder is empty (core only). The winder function block is disabled. For the winding axis (slave axis), select the operation mode "Velocity control" (in the AxisInterface of the ML_TechInterface "MODE_VEL" library). The axis can also be controlled using the PLCopen function block MC_MoveVelocity. 10 rpm can be set as command value.
2. The value of the parameter P-0-0109 "Peak torque/force limit" is set to zero. The axis has to stop now. The value for this parameter is incremented until the drive starts to rotate. The last value corresponds to the friction torque at standstill.
3. The axis enable "MODE_VEL" is canceled (also by using the PLCopen function block "MC_Stop").

   The parameter P-0-0109 "Peak torque/force limit" is reset to its previous value (e.g. 400%).

The friction torque setting at standstill is thus completed.

Determining friction torque at maximum speed

1. The "friction torque at maximum speed" ("MaxTorque" variable in the data structure MB_WIND_TENSION) is determined when the master axis is at standstill and with an empty winder (core only). The function block is disabled.
2. The operating mode "Velocity control" has to be selected for the winding axis (slave axis) (in the AxisInterface of the ML_TechInterface "MODE_VEL", optionally PLCopen function block).

   The command value for the selected operation mode has to be increased incrementally up to the value of the parameter S-0-0091 "Bipolar velocity limit value". The value for the parameter S-0-0091 "Peak torque/force limit" has to be entered according to the application and motor data.

3. In the next step, the value of the parameter P-0-0109 "Peak torque/force limit" is decreased incrementally (e.g. from 100% towards 0%) until the drive speed also starts to decrease. The speed of the drive can be traced using the drive parameter S-0-0040 "Actual velocity value" or the axis parameter A-0-0102 "Actual velocity value". The last value of P-0-0109"Peak torque/force limit" corresponds to the maximum friction torque during maximum speed.
4. The axis enable "MODE_VEL" is canceled (also by using the PLCopen function block "MC_Stop").

   The parameter P-0-0109 "Peak torque/force limit" is reset to its previous value (e.g. 400%).

   Setting the friction torque maximum speed "MaxTorque" is completed.

Determining minimum friction torque

The minimum friction torque is determined using the deceleration test with an empty winder. The winding drive is decelerated from a constant velocity (e.g. 50 min^{-1}) to standstill. At the same time, the drive parameters S-0-0080 "Torque/force command value" (axis parameter A-0-0038 "Torque/force limit value, bipolar") and S-0-0040 "Actual velocity value" (axis parameter A-0-0102 "Actual velocity value") are recorded with an oscilloscope or with the trace function in IndraWorks.

The minimum value is read from the torque curve and can be transferred to the structure MB_WIND_FRICTION as "MinTorque". The corresponding speed is entered as "MinTorqueVelocity" in MB_WIND_FRICTION.

Final settings

Finally, the remaining input variables can be determined.

The variables are listed in the table below.

Input variable	Note/setting
SetpointOpMode	Setting the desired operating tension
SetpointStandstill	Setting the standstill tension

Input variable	Note/setting
Offset in fine adjustment	In the data structure MB_AXIS_SYNCHRONISATION of the ML_TechInterface, the fine adjustment can be preset via the variable "FineAdjust" with a value depending on the winding direction. This variable is cyclically processed if "EnableCyclicScanning" is set to TRUE. As described before, the winding direction is determined with "WindDirection". For an unwinder, a positive value is used for "FineAdjust". For a rewinder, a negative value is used. These values should be ≤ 5 %
ActTorqueFilter	If the actual torque is to be filtered (as measured value), a value unequal to zero has to be applied at this input of the function block as a time constant
AccelTorqueFilter	If the acceleration torque is to be filtered, a value unequal to zero has to be applied at this input of the function block as time constant

Fig.9-80: Final determination of the remaining input variables

Initial startup of the winding axis

The winding material is placed in an untensioned state (slightly sagging). The winding axis has to be commanded either via the AxisInterface or via the PLCopen function blocks. The diameter calculator is activated by setting the "Enable" input. Now, the winding material has to be tensioned using the web tensioning control. The "SpeedOffset" output represents the velocity used for tensioning.

The process variable window "WindowTension" can be set to a value from 10 - 20% for example.

The master axis is enabled with a comparatively low speed (e. g. 10 min^{-1}). After a number of revolutions, the "DiameterEngaged" output has to be set. It is thus displayed that the diameter calculation is active ("engaged"). The calculation depends on the web tension or the actual torque at the winding axis.

When the diameter calculator is engaged, the standstill tension and the operating tension can be checked. Therefore, the value "CurrentTension" from the data structure MB_WIND_TENS_DIAG is monitored. In this data structure, the current torque limit value "TorqueLimit", the current acceleration torque "Acceleration torque" and the currently applied friction torque "FrictionTorque" can also be monitored.

The function block provides different process variables at its outputs. The calculated current diameter of the winder ("CurrentDiameter" output) should correspond to the actual diameter of the winder. Slight deviations can be disregarded. Furthermore, the fine adjustment of the gear for the speed adaptation of the drive axis of the winder ("GearRatioFineAdjust" output), the current moment of inertia of the winder ("CurrentInertia" output) and the speed offset ("SpeedOffset" output), the additive torque "AddTorque" and the torque limit value ("TorqueLimitBipolar" output) are displayed.

Monitoring the torque limit value

During the initial control startup, a drive warning "E2056 Torque limit value = 0" can occur if the winder function block is not yet enabled.

In order to prevent this, the value for the parameter S-0-0092 "Bipolar torque/force" can be written with a value unequal to zero such as 100% in the start-up phase of the PLC program (e.g. warm start) of the corresponding cyclic data containers.

If the warning "E8260 Torque - force command value limit active" occurs when the function block is enabled, this warning can be eliminated by setting bit 4 of the parameter P-0-0556, "Axis controller configuration".

In rare cases (e.g. multiple F8078 "Error in the speed control loop"), it can be necessary to disable the velocity controller monitoring (set bit 1 of the parameter P-0-0556 "Axis controller configuration").

9.8 Supplementary Function Blocks

9.8.1 Introduction and Overview

This section summarizes function blocks, which can supplement functionalities of the different diameter calculators and the dancer position controller. They can also be implemented into the PLC project of the IndraLogic.

The following function blocks are available for additional functionalities (e. g. remaining runtime for a splice):

Description	Type	Description
MB_CalcInertiaLimitType02	FB	Calculates the maximum moment of inertia of the winding material on the winder
MB_UnwindMaterialType02	FB	Calculates process variables for an unwinder like the unrolled and the remaining length of the web of fabric, the material thickness and the remaining runtime for a splice.
MB_WindTaperProfileType01	FB	It calculates a winding tightness characteristic curve for a winder
MB_WindSpeedControlAdaption-Type01	FB	It calculates for a winder a Kp adaptation for the speed controller

Fig.9-81: Brief description of additional function blocks for different winder types from the RMB_TechWinder library

9.8.2 MB_CalcInertiaLimitType02

Brief Description

The function block represents a non-cyclic function block in contrast to the other function blocks described in this documentation. It is systematically,for example, called for the initialization of another function block.

In this function block, the material properties and dimensions of the web of fabric are analyzed and the maximum moment of inertia at the maximum diameter "MaxDiameter" is calculated for the winding material on the winder. The calculated maximum moment of inertia "MaxInertia" can be used for a speed controller adaptation for example (adaptation of the parameter S-0-0100 "Velocity controller proportional gain"). The proportional gain between a minimum value and a maximum value can be adapted. For example, a linear increase of the proportional gain factor can be set.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_TechWinder.lib

Fig.9-82: Reference table of the MB_CalcInertiaLimitType02

RMB_TechWinder.library/MX_TechWinder.lib

Interface Description

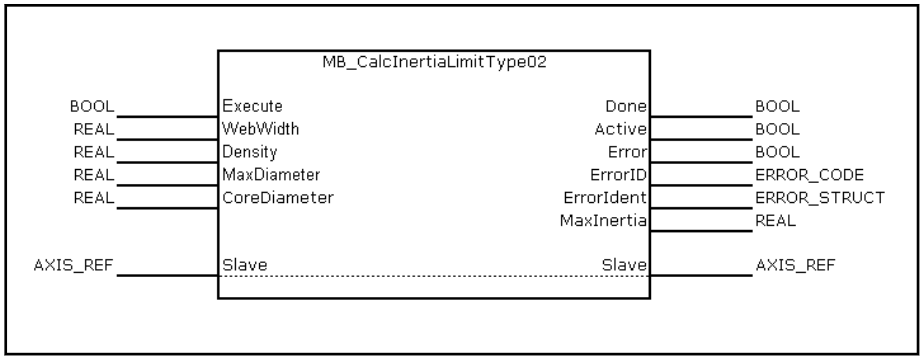


Fig.9-83: MB_CalcInertiaLimitType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Activating the function block (edge-controlled)
	WebWidth	REAL	Web width [mm]
	Density	REAL	Material density of the web [kg/dm^3]
	MaxDiameter	REAL	Maximum diameter of the winder [mm]
	CoreDiameter	REAL	Core diameter of the winder [mm]
VAR_OUTPUT	Done	BOOL	Maximum moment of inertia "MaxInertia" calculated
	Active	BOOL	Processing, function block enabled
	Error	BOOL	Error occurred
	ErrorID	ERROR_CODE	Error code
	ErrorIdent	ERROR_STRUCT	Advanced error display
	MaxInertia	REAL	Maximum moment of inertia [kg/dm^3]
VAR_IN_OUT	Slave	AXIS_REF	Axis structure of the winding axis

Fig.9-84: Interface variables of the MB_CalcInertiaLimitType02 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
Webwidth	REAL	0.0	n.def	0.0	Rising edge at "Execute"
Density	REAL	0.0	n.def	1.0	Rising edge at "Execute"
MaxDiameter	REAL	> 0.0	n.def	2000.0	Rising edge at "Execute"
CoreDiameter	REAL	> 0.0	n.def	100.0	Rising edge at "Execute"

Fig.9-85: Min./max. and default values of the MB_CalcInertiaLimitType02

Signal-time diagram

The following signal-time diagram for the operating inputs and outputs displays the edge-controlled behavior. If there is a positive edge at the "Execute" input, the function block becomes active. The processing which is, in this example, specified for a duration of two cycles (two boxes in the figure)

is displayed at the "Active" output. After the processing, the output is set to "Done". This state remains if "Execute" remains set and if no error ("Error") occurs. If "Execute" is not set, "Done" is deleted after one cycle. The "Error" output is set in case of error. "Error" remains set until "Execute" is reset. If "Execute" is not set, the error message at the "Error" output is deleted after one cycle.

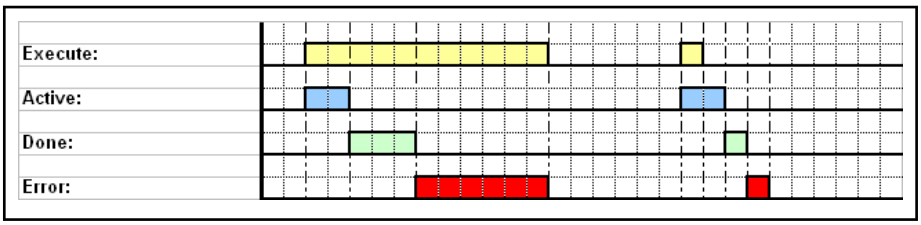


Fig.9-86: Signal-time diagram for selected inputs and outputs of the MB_CalcInertiaLimitType02 function block

Functional Description

The moment of inertia "MaxInertia" is calculated for the specified maximum diameter "MaxDiameter" using the core diameter of the winder "CoreDiameter", the web width "WebWidth" and the web density "Density".

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used.

ErrorID	Additional1	Additional2	Text
STATE_MACHINE_ERROR	16#1140	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#1141	16#0001	WebWidth < 0.0
INPUT_RANGE_ERROR	16#1141	16#0002	Density < 0.0
INPUT_RANGE_ERROR	16#1141	16#0003	MaxDiameter < 0.0
INPUT_RANGE_ERROR	16#1141	16#0004	CoreDiameter < 0.0
INPUT_RANGE_ERROR	16#1141	16#0005	MaxDiameter < CoreDiameter
INPUT_RANGE_ERROR	16#1141	16#0006	Number of slave axis incorrect
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated

Fig.9-87: Error codes of the MB_CalcInertiaLimitType02 function block

Required hardware

- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware

- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software

- IndraWorks 12VRS for IndraMotion MLC or IndraMotion MLD

Required parameterization

An actually existing mechanical gear ratio from the drive axis to the winding axis has to be entered into the parameters S-0-0121 ("Input revolutions of load gear") and S-0-0122 ("Output revolutions of load gear").

9.8.3 MB_UnwindMaterialType02

Brief Description

The function block acts as a material counter . For an unwinder, it calculates the following cyclic process variables: unrolled web length "UnrolledLength", remaining web length "RestLength", material thickness "MaterialThickness" and the remaining runtime "RemainRunTime".

RMB_TechWinder.library/MX_TechWinder.lib

This data can prepare a splice.

The current winding diameter "CurrentDiameter" and the remaining roll diameter "RestRollDiameter" are input variables.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library

Fig.9-88: Reference table of the MB_UnwindMaterialType02

Interface Description

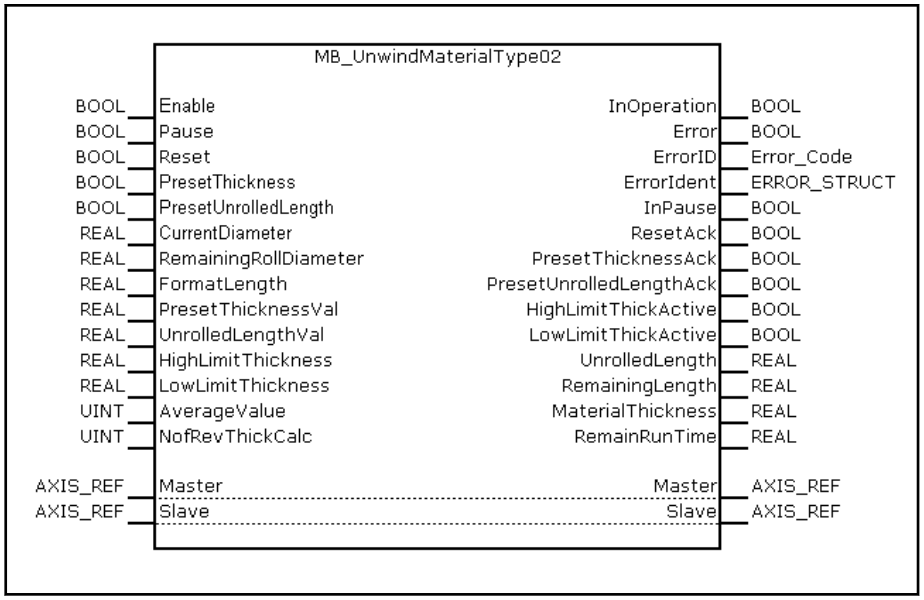


Fig.9-89: MB_UnwindMaterialType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enabling the function block (level-controlled)
	Pause	BOOL	Material count paused
	Reset	BOOL	Resetting the values for the unrolled web length, remaining web length
	PresetThickness	BOOL	Preset for the material thickness edge-controlled
	PresetUnrolledLength	BOOL	Preset for the unrolled web length is edge-controlled
	CurrentDiameter	REAL	Current winding diameter [mm] from the diameter calculator function block
	RemainingRollDiameter	REAL	Remaining roll diameter [mm]
	FormatLength	REAL	Format length [mm] (corresponds to an unwound revolution of the reference axis)
	PresetThicknessVal	REAL	Preset value for the material thickness [µm]
	UnrolledLengthVal	REAL	Preset value for the unwound web length [mm]
	HighLimitThickness	REAL	Upper limit of the material thickness [µm]
	LowLimitThickness	REAL	Lower limit of the material thickness [µm]
	AverageValue	UINT	Number of the material thickness values which is averaged. The input is limited to a maximum of 30

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Data type	Description
	NofRevThickCalc	UINT	Number of revolutions which is accomplished after the calculation of the material thickness
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Error occurred
	ErrorID	ERROR_CODE	Error code
	ErrorIdent	ERROR_STRUCT	Advanced error display
	InPause	BOOL	Material calculation paused
	ResetAck	BOOL	Reset completed
	PresetThicknessAck	BOOL	Material thickness preset
	PresetUnrolledLength-Ack	BOOL	Unrolled length preset
	HighLimitThickA-chieved	BOOL	The currently calculated material thickness has achieved the upper limit (value will be discarded)
	LowLimitThickA-chieved	BOOL	The currently calculated material thickness has reached the lower limit (value is discarded)
	RemainingLength	REAL	Remaining web length on the winder [mm]
	MaterialThickness	REAL	Material thickness of the web [µm]
	UnrolledLength	REAL	Unrolled web length [mm]
	RemainRunTime	REAL	Remaining runtime until splice (based on the remaining roll diameter) [s]
VAR_IN_OUT	Master	AXIS_REF	Axis structure of the reference axis
	Slave	AXIS_REF	Axis structure of the winding axis

Fig.9-90: Interface variables of the MB_UnwindMaterialType02 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Reset	BOOL			FALSE	Continuous
PresetThickness	BOOL			FALSE	Continuous
PresetUnrolled-Length	BOOL			FALSE	Continuous
CurrentDiameter	REAL	0.0	n.def	0.0	Continuous
RestRollDiameter	REAL	0.0	n.def	0.0	Rising edge at "Enable"
FormatLength	REAL	0.0	n.def	0.0	Rising edge at "Enable"
PresetThick-nessVal	REAL	> 0.0	n.def	0.0	Continuous

RMB_TechWinder.library/MX_TechWinder.lib

Name	Type	Min. value	Max. value	Default value	Effective
UnrolledLengthVal	REAL	0.0	n.def	0.0	Continuous
HighLimitThickness	REAL	0.0	n.def	500.0	Continuous
LowLimitThickness	REAL	0.0	n.def	10.0	Continuous
AverageValue	UINT	0	30	1.0	Continuous
NofRevThickCalc	UINT	0	n.def	50	Continuous

Fig.9-91: Min./max. and default values of the MB_UnwindMatrialType02 function block

Signal-time diagram

The following explains the signal-time diagram.

The function block is activated with the "Enable" input. This is displayed at the "InOperation" output. If an error occurs, the "Error" output is set and operational availability is interrupted. The execution of the function block can be interrupted with the "Pause" input. This is signaled at the "InPause" output. Resetting the function block ("Reset" input) or setting material parameters ("PresetThickness" and "PresetUnrolledLength" inputs) is performed irrespective of the "Pause" input. However, it is recommended that the function block is activated in "Pause". Thus, the preset and the reset can be performed in a more controlled way. Setting and resetting is displayed at the "ResetAck", "PresetThicknessAck" and "PresetUnrolledLengthAck" outputs.

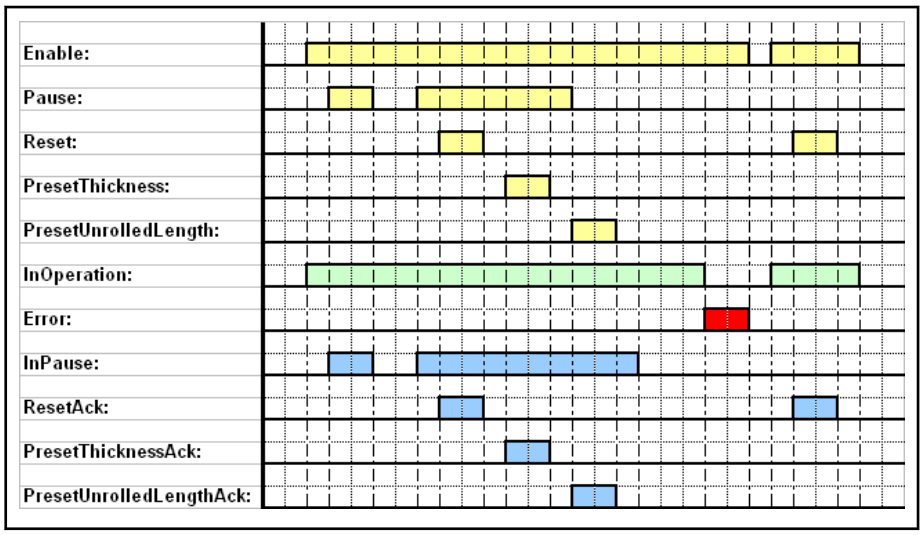


Fig.9-92: Signal-time diagram for selected inputs and outputs of the MB_UnwindMaterialType02 function block

Functional Description

Calculating the material variables

The remaining length of the "RestLength" web is calculated with the help of the current diameter. This means that variations in the current diameter also have to be observed while calculating the remaining length ("RestLength"). The same applies to the calculation of the material thickness ("MaterialThickness") which, in turn, works with the value of the remaining length as well as for the calculation of the remaining runtime ("RemainRunTime") which uses the material thickness and velocity of the reference axis. The more precisely the current diameter is determined, the more precise are the outputs: "UnrolledLength", "RestLength", "MaterialThickness" and "RemainRunTime" of the function block to be calculated.

RMB_TechWinder.library/MX_TechWinder.lib

Further input variables are used. In order to obtain physically useful values for the material thickness, this is limited by "HighLimitThickness" and "LowLimitThickness". This limitation is effective when averaging the material thickness. A fixed number of values "AverageValue" is averaged. A calculation of the material thickness calculates all "NofRevThickCalc" revolutions of the winding axis.

Preset of material variables

At the beginning of the material counter, the outputs are initialized. This is performed with the values "PresetThicknessVal" and "UnrolledLengthVal". These values can also be used as preset values at runtime. These are set by the inputs "PresetThickness" and "PresetUnrolledLength".

Reset the material calculator

Moreover, the material counter can be reset by the "Reset" input. Resetting ("Reset") and presetting (Preset) the outputs can be performed at any time if the function block is active.

Averaging the material thickness

The calculated material thickness can be averaged by setting the "AverageValue" input . The number of material thickness values used for averaging is entered at "AverageValue".

Only the current diameters internally calculated and within the limits of "HighLimitThickness" and "LowLimitThickness" contribute to the average value. Values outside these limits are discarded. Values between 1 and 10 are reasonable for "AverageValue". Therefore, "AverageValue" is limited to a maximum of 30. If a higher value is entered, "AverageValue" is set to 30. However, no error message is displayed as this does not represent a critical process. The higher the selected number, the more precise the calculated value of the material thickness. However, the accuracy is reduced with increasing values for "AverageValue" since the calculated value is lagging due to "filtering". The value for "AverageValue" should therefore be selected appropriately.

After the control has been switched off, the memories are deleted. The values for "UnrolledLength", "RestLength", "MaterialThickness" and "RemainRunTime" are therefore 0 when the control is switched on again.

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used.

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#11A0	16#20	Error in state machine
INPUT_RANGE_ERROR	16#11A1	16#01	CurrentDiameter < 0.0
INPUT_RANGE_ERROR	16#11A1	16#02	RestRollDiameter < 0.0
INPUT_RANGE_ERROR	16#11A1	16#03	FormatLength <= 0.0
INPUT_RANGE_ERROR	16#11A1	16#04	PresetThicknessVal < 0.0
INPUT_RANGE_ERROR	16#11A1	16#05	UnrolledLengthVal < 0.0
INPUT_RANGE_ERROR	16#11A1	16#06	HighLimitThickness < 0.0
INPUT_RANGE_ERROR	16#11A1	16#07	LowLimitThickness < 0.0
INPUT_RANGE_ERROR	16#11A1	16#08	HighLimitThickness < LowLimitThickness

RMB_TechWinder.library/MX_TechWinder.lib

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#11A1	16#09	AverageValue < 0
INPUT_RANGE_ERROR	16#11A1	16#0A	NofRevThickCalc < 0
INPUT_RANGE_ERROR	16#11A1	16#0B	Number of slave axis incorrect
INPUT_RANGE_ERROR	16#11A1	16#0C	Number of master axis incorrect
RESSOURCE_ERROR	16#0004	16#0000	Incorrect drive firmware
RESSOURCE_ERROR	16#000F	16#0003	Drive function package "MA" not activated

Fig.9-93: Error codes of the MB_UnwindMaterialType02 function block

Required hardware
- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware
- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software
- IndraWorks 12VRS for IndraMotion MLC or IndraMotion MLD

Required parameterization

The interpolation in the drive has to be enabled for the winding axis.

(Real axes → Corresp. axis → Properties)

The scaling type has to be identical for master axis and slave axis. Rotary scaling and preference scaling are recommended.

9.8.4 MB_WindTaperProfileType01

Brief Description

The function block realizes a diameter-dependent adaptation of the winding tightness for a winder.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library

Fig.9-94: Reference table of the MB_WindTaperProfileType01

Interface Description

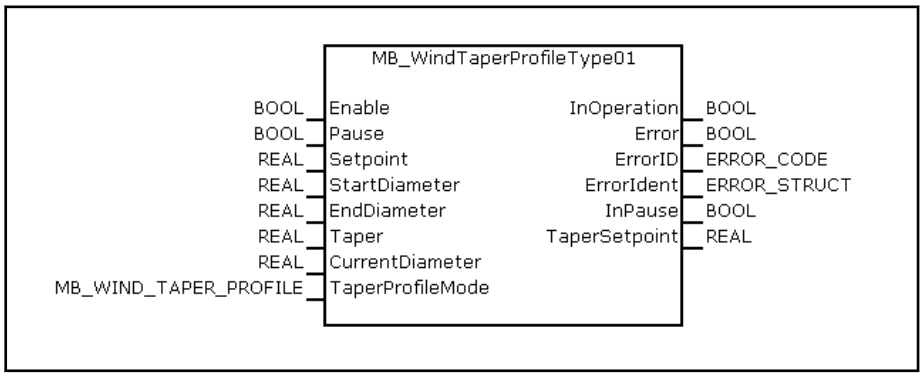


Fig.9-95: MB_WindTaperProfileType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing of function block enabled (level-controlled)
	Pause	BOOL	Function block pauses, TaperTensionSetpoint remains "frozen"

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Data type	Description
	Setpoint	REAL	Command value [-] Command tension value for a winder with torque limitation. Command value for pneumatic force for a winder with dancer
	StartDiameter	REAL	Starting diameter [mm] from which the tension should be adapted
	EndDiameter	REAL	Diameter [mm] from which the tension is not adapted anymore (remains constant). This applies only for the linear characteristic curve
	Taper	REAL	Winding tightness [%] specifies the winding tightness characteristic curve. If the value is between 0 and 100%, the tensile stress decreases with an increasing diameter. If the value is between 0 and -100%, the tensile stress increases with an increasing diameter
	CurrentDiameter	REAL	Current diameter [mm] of the rewinder. (e.g. from the MB_WinderTensionCtrlType02 function block)
	TaperProfileMode	MB_WIND_TAPER_PROFILE	Selecting the winding tightness profile
VAR_OUTPUT	InOperation	BOOL	Function block is working
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	InPause	BOOL	Winding tightness pauses (tension command value is "frozen")
	TaperSetpoint	REAL	Command value considering the winding tightness characteristic curve

Fig.9-96: Interface variables of the MB_WindTaperProfileType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Setpoint	REAL	0.0	n.def.	0.0	Continuous
StartDiameter	REAL	> 0.0	n.def.	100.0	Continuous
EndDiameter	REAL	StartDiameter	n.def.	1000.0	Continuous
Taper	REAL	-100.0	100.0	0.0	Continuous
CurrentDiameter	REAL	0.0	n.def.	0.0	Continuous
TaperProfileMode	MB_WIND_TAPER_PROFILE			NO_PROFILE	Rising edge at "Enable"

Fig.9-97: Min./max. and default values of the MB_WindTaperProfileType01 function block

RMB_TechWinder.library/MX_TechWinder.lib

Element	Value	Description
NO_PROFILE	0	No profile
PROFILE_LINEAR	1	Linear curve profile
PROFILE_HYPERBOLIC	2	Hyperbolic curve profile

Fig.9-98: Elements of the MB_WIND_TAPER_PROFILE enumeration type

Signal-time diagram

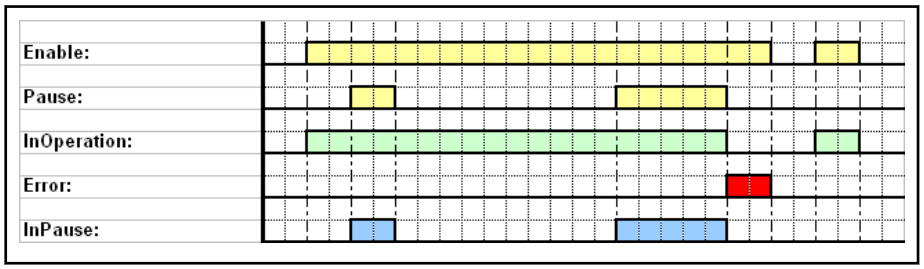


Fig.9-99: Signal-time diagram for selected inputs and outputs of the MB_WindTaperProfileType01 function block

Functional Description

When winding material, it is often desired that the tension used for winding is influenced depending on the winding diameter. While winding, the tension should be higher at the beginning and should be reduced with the winding diameter increasing.

The MB_WindTaperProfileType01 function block influences a determinable command value according to a selectable characteristic curve and a determinable winding diameter. The winding tightness characteristic curve can be influenced via the "Taper" input (winding tightness). If the value for the winding tightness is between 0% and 100%, the tension decreases with an increasing diameter. If the value is between 0% and -100%, the tensile stress increases with an increasing diameter. The higher the absolute value for the winding tightness, the higher the increase or decrease of the tension.

The function block can be used for a winder with speed limitation. It is connected in front of the function block MB_WinderTensionCtrlType02. That means that the "SetpointOpMode" input of the function block "MB_WinderTensionCtrlType02" is interconnected with the "TaperSetpoint" output.

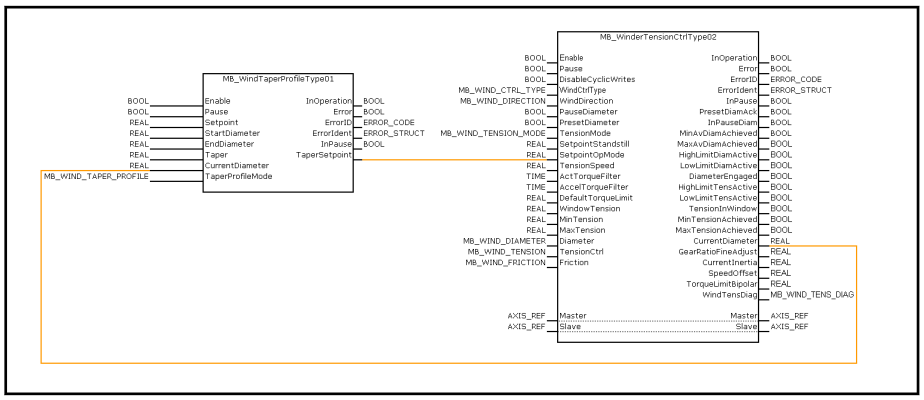


Fig.9-100: MB_WindTaperProfileType01, interconnection

If the function block for a winder with dancer is used, the "TaperSetpoint" output affects (e.g. analog tension value) the pneumatic force the added to the dancer). The "Setpoint" input then also corresponds to a value for the pneumatic force.

Linear winding tightness characteristic curve:

RMB_TechWinder.library/MX_TechWinder.lib

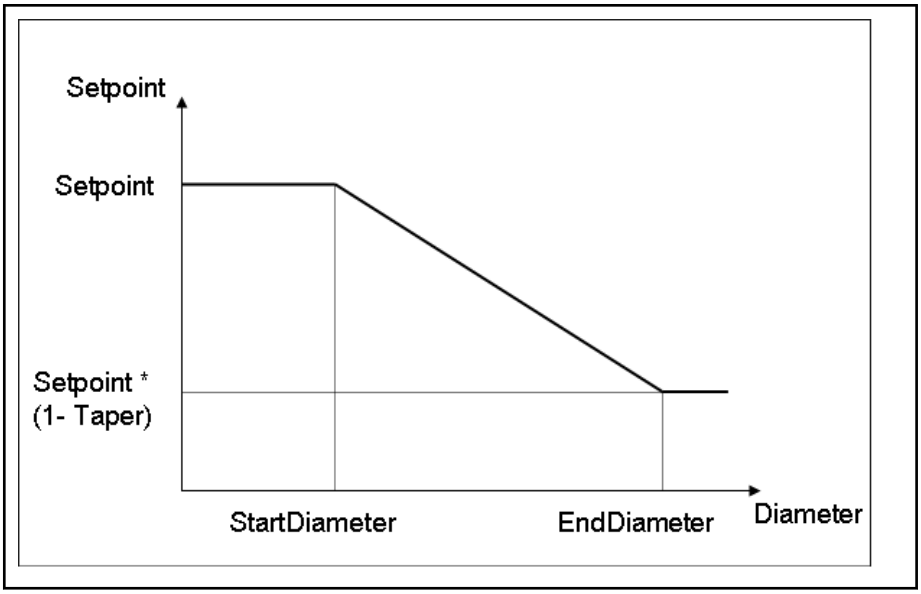


Fig.9-101: MB_WindTaperProfileType01, Linear winding tightness characteristic curve

Hyperbolic winding tightness characteristic curve:

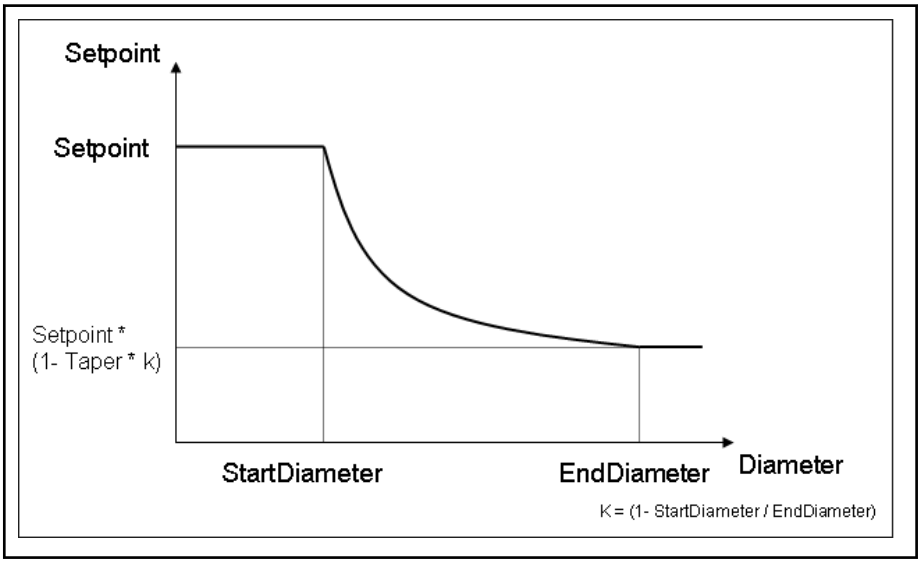


Fig.9-102: MB_WindTaperProfileType01, Hyperbolic winding tightness characteristic curve

Error Handling

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used.

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#1150	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#1151	16#0001	TensionSetpoint < 0.0
INPUT_RANGE_ERROR	16#1151	16#0002	StartDiameter <= 0.0
INPUT_RANGE_ERROR	16#1151	16#0003	EndDiameter <= 0.0
INPUT_RANGE_ERROR	16#1151	16#0004	EndDiameter <= StartDiameter
INPUT_RANGE_ERROR	16#1151	16#0005	Taper < -100.0 or Taper > 100.0

RMB_TechWinder.library/MX_TechWinder.lib

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#1151	16#0006	CurrentDiameter <= 0.0
INPUT_RANGE_ERROR	16#1151	16#0007	Incorrect TaperProfile mode

Fig.9-103: Error codes of the MB_WindTaperProfileType01 function block

Required hardware
- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware
- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS
  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software
- IndraWorks 12VRS for IndraMotion MLC

Special features for the IndraMotion MLC

Special features for the IndraMotion MLC control

No special parameterization required.

9.8.5 MB_WindSpeedControlAdaptionType01

Brief Description

The function block executes an adaptation of the Kp-gain of the speed controller of the respective winding drive that depends on the current moment of inertia of the winder and a minimum and maximum Kp-value.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechWinder.compiled-library

Fig.9-104: Reference table of the MB_WindSpeedControlAdaptionType01

Interface Description

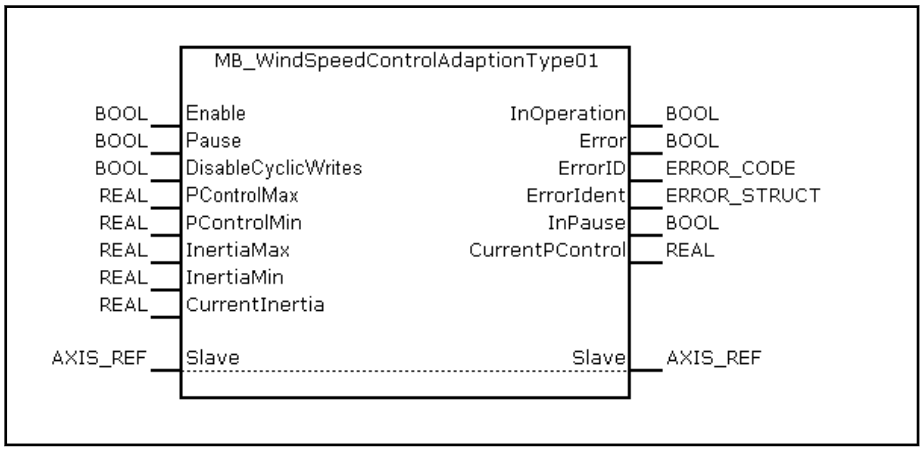


Fig.9-105: MB_WindSpeedControlAdaptionType01 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Slave	AXIS_REF	Axis structure of the winding axis
VAR_INPUT	Enable	BOOL	Processing of the function block enabled (once, edge-controlled)
	Pause	BOOL	Function block pauses, CurrentPControl remains "frozen"
	DisableCyclicWrites	BOOL	Writing on drive parameter S-0-0100 can be switched off
	PControlMax	REAL	Kp-gain for maximum diameter [Nm/(rad/s)]

RMB_TechWinder.library/MX_TechWinder.lib

I/O type	Name	Data type	Description
	PControlMin	REAL	Kp-gain for minimum diameter [Nm/(rad/s)]
	InertiaMax	REAL	Moment of inertia at maximum diameter or moment of inertia [kg*m^2] (related to the motor shaft)
	InertiaMin	REAL	Moment of inertia at minimum diameter (core) or moment of inertia [kg*m^2] (related to the motor shaft)
	CurrentInertia	REAL	Current moment of inertia of the winder [kg/m^2] (related to the motor shaft)
VAR_OUTPUT	InOperation	BOOL	Processing completed without error, output data valid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	InPause	BOOL	Function block pauses (CurrentPControl is "frozen")
	CurrentPControl	REAL	Current Kp-value (speed controller) [Nm/(rad/s)]

Fig.9-106: Interface variables of the MB_WindSpeedControlAdaptionType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
PControlMax	REAL	PControlMin	n.def.	0.0	Rising edge at "Enable"
PControlMin	REAL	0.0	n.def.	0.0	Rising edge at "Enable"
InertiaMax	REAL	InertiaMin	n.def.	0.0	Rising edge at "Enable"
InertiaMin	REAL	0.0.	n.def.	0.0	Continuous
CurrentInertia	REAL	0.0	n.def.	0.0	Continuous

Fig.9-107: Min./max. and default values of the MB_WindSpeedControlAdaption-Type01 function blocks

Signal-time diagram

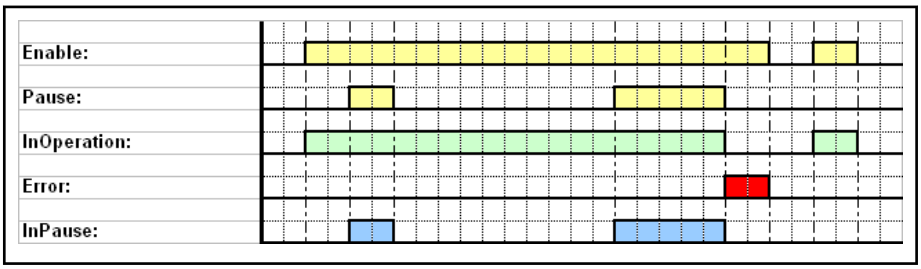


Fig.9-108: Signal-time diagram for selected inputs and outputs of the MB_Wind-SpeedControlAdaptionType01 function block

RMB_TechWinder.library/MX_TechWinder.lib

Functional Description

The function block linearly calculates the value for the Kp of the speed controller depending on the current moment of inertia. This value is calculated between a minimum and maximum value. The minimum and maximum Kp have to be determined at empty core and at full winding diameter. Furthermore, the values for the minimum and maximum moment of inertia (at the core or at full winding diameter) are required. These can be determined using the function block "MB_CalcInertiaLimitType02".

The current kp-value of the speed controller is calculated from these values with regard to the current moment of inertia of the winder.

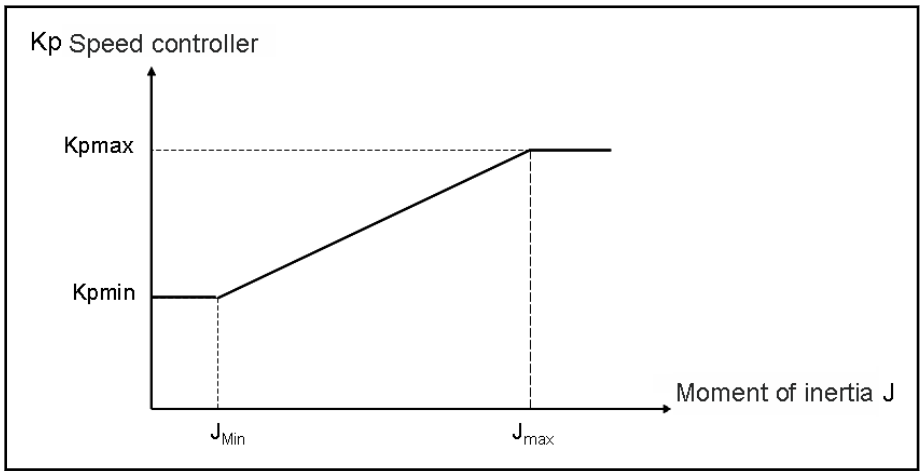


Fig.9-109: MB_WindSpeedControlAdaptionType01

The current Kp-value is cyclically written to the drive parameter S-0-0100 "Velocity controller, proportional gain". Therefore, the parameter S-0-0100 has to be configured in the cyclic data channels between control and drive. If the function block is switched off, the last value of "CurrentPControl" remains present.

This writing can be disabled with the "DisableCyclicWrites" input. The "CurrentPControl" output is then not written internally by the function block but has to be transferred from the PLC program to the drive parameters.

☞ When switching on the control, the cyclic data channels are always initialized with zero between control and drive. If the parameter S-0-0100 "Velocity controller, proportional gain" is configured cyclically, it has to be ensured that the respective cyclic channel is pre-initialized with a reasonable value after the control start. This applies until the function block was enabled.

The following table lists and describes the error numbers of the function block. The error codes from the "F_RELATED table (16#0170)" are used.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#11E0	16#0020	Error in state machine
INPUT_RANGE_ERROR	16#11E1	16#01	PControlMax < 0.0
INPUT_RANGE_ERROR	16#11E1	16#02	PControlMin < 0.0
INPUT_RANGE_ERROR	16#11E1	16#03	InertiaMax < 0.0
INPUT_RANGE_ERROR	16#11E!	16#04	CurrentInertia < 0.0
INPUT_RANGE_ERROR	16#11E1	16#05	InertiaMin < 0.0
INPUT_RANGE_ERROR	16#11E1	16#06	PControlMax <= PControlMin

RMB_TechWinder.library/MX_TechWinder.lib

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#11E1	16#07	InertiaMax <= InertiaMin
DEVICE_ERROR	16#11E2	16#01	Axis not in BB

Fig.9-110: Error codes of the MB_WindSpeedControlAdaptionType01 function block

Required hardware
- IndraMotion MLC hardware CML65, CML45, CML25
- IndraDrive

Required firmware
- IndraMotion MLC firmware FWA-CMLXX*-MLC-12VRS

  XX - Device variant, e.g. CML65
- IndraDrive drive firmware

Required software
- IndraWorks 12VRS for IndraMotion MLC

Special features for the IndraMotion MLC

The **Parameter S-0-0100 ("Velocity controller, proportional gain")** has to be entered in the cyclic SERCOS channel of the corresponding drive "UserCmd-DataX"). The real axes are parameterized in IndraWorks in parameterization mode.

☞ The initial values are always "0" for the parameter in the cyclic SERCOS channel. That means that when switching on control, the value for the parameter S-0-0100 "Velocity controller, proportional gain" is 0 if it is entered in the cyclic SERCOS channel and if the function block is not enabled. To avoid this, the respective data container can be once initialized with a value when activating the control, e.g.

AxisData[Antriebname.AxisNo].dwUserCmdDataX_q.REAL_ := 10.0;

9.9 Parameter Overview

This section lists the parameters used in the documentation.

There are axis parameters (A-parameters) for the IndraMotion MLC control, product-specific parameters (P-parameters) and default parameters (S-parameters) for the respective drive.

For the IndraMotion MLC control, there are only product-specific parameters (P-parameters) and default parameters (S-parameters).

Parameters	Description
A-0-0029	Position polarities
A-0-0038	Torque/force limit value, bipolar
A-0-0050	Scaling type for torque/force data
A-0-0102	Actual velocity value
A-0-0500	Configuration of user-defined command value A
A-0-0502	Configuration of user-defined command value B
A-0-0504	Configuration of user-defined command value C
A-0-0506	Configuration of user-defined command value D
A-0-0520	Configuration of actual user-defined value A
A-0-0522	Configuration of actual user-defined value B

RMB_TechWinder.library/MX_TechWinder.lib

Parameters	Description
A-0-0524	Configuration of actual user-defined value C
A-0-0526	Configuration of actual user-defined value D
A-0-2798	Polarity of master axis position

Fig.9-111: IndraMotion MLC axis parameters (A-parameters)

Parameters	Description
S-0-0040	Actual velocity value
S-0-0043	Velocity polarity parameter
S-0-0055	Position polarities
S-0-0080	Torque/force command value
S-0-0084	Actual torque/force value
S-0-0085	Torque/force polarity parameter
S-0-0091	Bipolar velocity limit value
S-0-0092	Torque/force limit value, bipolar
S-0-0100	Velocity controller, proportional gain
S-0-0121	Input revolutions of load gear
S-0-0122	Output revolutions of load gear

Fig.9-112: Default parameters (S-parameters)

Parameters	Description
P-0-0108	Master drive polarity
P-0-0665	Axis controller configuration
P-0-0690	Additive velocity command value
P-0-0694	Gear ratio fine adjustment, process controller

Fig.9-113: Product-specific parameters (P-parameters)

Parameters	Description
P-0-1623	CCD: Configuration master command values
P-0-1624	CCD: Configuration actual slave values
P-0-1625	CCD: Configuration list slave command values
P-0-1626	CCD: Configuration list actual slave values
P-0-1730	CCD: MDT real-time container 1, master
P-0-1731-39	CCD: MDT real-time container 1, slave n (n = 1-9)
P-0-1740	CCD: MDT real-time container 2, master
P-0-1741-49	CCD: MDT real-time container 2, slave n (n = 1-9)
P-0-1750	CCD: MDT real-time container 3, master
P-0-1751-59	CCD: MDT real-time container 3, slave n (n = 1-9)
P-0-1760	CCD: MDT real-time container 4, master

RMB_TechWinder.library/MX_TechWinder.lib

Parameters	Description
P-0-1761-69	CCD: MDT real-time container 4, slave n (n = 1-9)
P-0-1770	CCD: AT real-time container 1, master
P-0-1771-79	CCD: AT real-time container 1, slave n (n = 1-9)
P-0-1780	CCD: AT real-time container 2, master
P-0-1781-89	CCD: AT real-time container 2, slave n (n = 1-9)
P-0-1790	CCD: AT real-time container 3, master
P-0-1791-99	CCD: AT real-time container 3, slave n (n = 1-9)
P-0-1800	CCD: AT real-time container 4, master
P-0-1801-09	CCD: AT real-time container 4, slave n (n = 1-9)

Fig.9-114: Product-specific parameters (P-parameters)

10 RMB_TechCrossCutCrossSeal.library

10.1 MB_CrossCutSealType01

Brief Description

The MB_CrossCutSealType01 function block calculates, loads and activates a MotionProfile for CrossSealer or CrossCutter applications. The function block uses the MB_CrossCutterCalcType04, page 511 function block internally for the calculation.

With the calculated MotionProfile, the slave axis moves with product velocity within the sealing or cutting area. Other functions like OverSpeed, PushOut or cosine correction can be optionally added within this area.

Outside the sealing or cutting area, an almost optimum compensating motion (with regard to acceleration and energy loss) is performed if the specified format length does not correspond to the distance of the knife or the sealing jaw. Furthermore, a pendular motion of the CrossSealer or CrossCutter axis within freely selectable limits can be predefined to optimally use the acceleration capacity of the drive in case of large format lengths.

During operation, input variables like, for example, the format length, can be changed to automatically calculate a new MotionProfile in the background. The newly calculated MotionProfile is activated and becomes effective at the switching phase.

At a negative edge at "Enable", either the CrossSealer or the CrossCutter are deactivated. The axis is immediately stopped via the predefined deceleration ramp.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechCrossCutCrossSeal.compiled-library
IndraMotion MLD/MPx07 with MA function package	MX_Technology07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.10-1: Reference table of the MB_CrossCutSealType01

RMB_TechCrossCutCrossSeal.library

Interface Description

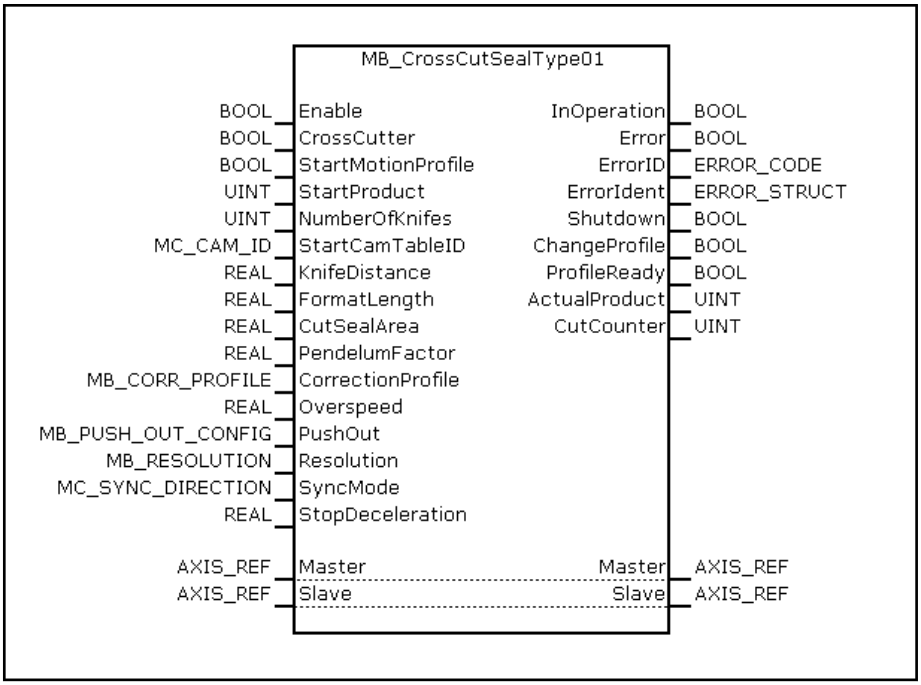


Fig.10-2: MB_CrossSealType01 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	Master	AXIS_REF	Reference to master axis
	Slave	AXIS_REF	Reference to the slave axis (= CrossCutter axis/CrossSealer axis) The modulo value is 360° electrical and is mechanically defined as the distance between two knives/sealing jaws
VAR_INPUT	Enable	BOOL	Processing of function block enabled

RMB_TechCrossCutCrossSeal.library

I/O type	Name	Type	Comment
	CrossCutter	BOOL	If CrossCutter = TRUE, the CrossCutter use case (see the following detailed description) is selected. In this use case, the master axis has to be standardized so that the material to be cut covers the distance that corresponds to the knife distance in one revolution. The electronic gear of the MotionProfile used is set to the gear ratio "FormatLength/KnifeDistance". Furthermore, the input "CutSealArea" functions as the cut angle of the CrossCutter whereas the input variables "ProductsPerCut" and "StartProduct" are not effective. If the CrossCutter = FALSE, the CrossSealer use case (see the following detailed description) is selected. In this use case, the master axis has to be standardized so that one revolution corresponds to a foil feed of one product length. The electronic gear of the MotionProfile used is set to the gear ratio "1/ProductsPerCut". Furthermore, the input "CutSealArea" corresponds to the sealing length and is thus defined in translatory units (e.g. mm or inch). The inputs "ProductsPerCut" and "StartProduct" are evaluated in this case
	StartMotionProfile	BOOL	The activation of a new MotionProfile is blocked as long as this input is set to FALSE. StartMotionProfile = TRUE activates the new MotionProfile if the output "ProfileReady" is TRUE
	ProductsPerCut	UINT	Input is only effective if CrossCutter = FALSE. Number of products per sealing process. Corresponds to the number of master revolutions per sealing or cutting process
	StartProduct	UINT	Input is only effective if CrossCutter = FALSE. In the case of more than one product per sealing or cutting process (ProductsPerCut > 1), the start product can be defined via this input when enabling the function block
	NumberOfKnives	UINT	Number of sealing jaws or knives available on the circumference of the sealing or cutting cylinder. The sealing jaws or knives are positioned with same distance from each other
	StartCamTableID	MC_CAM_ID	Starting number of the first cam point table. A block of a maximum of 6 point tables is required
	KnifeDistance	REAL	Distance of the knives or sealing jaws. Corresponds to the circumference divided by the number of knives or sealing jaws [mm or inch].
	FormatLength	REAL	Predefined cutting length or sealing distance/package length [mm or inch]

RMB_TechCrossCutCrossSeal.library

I/O type	Name	Type	Comment
	CutSealArea	REAL	Within this area, the knife or the sealing jaw moves with product velocity. The velocity within this window can be affected by additional inputs like, for example, OverSpeed, PushOut, cosine correction etc. If the input CrossCutter = FALSE, this input corresponds to the sealing length and is thus defined in translatory units (e.g. in mm or inch). If the input CrossCutter = TRUE, this input acts rotatorily as angle in [°]
	PendelumFactor	REAL	Maximum pendulum radius in [%] 0%: No pendular motion (standstill in case of large format length) 100%: Maximum pendular motion (pendular motion possible up to the edges of the sealing area or cutting area)
	CorrectionProfile	MB_CORR_PROFILE	Special profile selection. The inputs "OverSpeed" or "PushOut" have to be used for the parameterization of the special profile (see chapter 10.2.4 "MB_CORR_PROFILE" on page 521)
	Overspeed	REAL	Constant overspeed or reduction of the master axis velocity in the cutting area in [%] 0%: No overspeed 100%: Double master axis velocity -50%: Half the master axis velocity This input is only effective if the option CORR_PROFILE_OVERSPEED is selected at the input "CorrectionProfile"
	PushOut	MB_PUSH_OUT_CONFIG	Structure to define a PushOut motion during the cut or the sealing process (see chapter 10.2.6 "MB_PUSH_OUT_CONFIG" on page 523). In the CorrectionProfile, either the option CORR_PROFILE_PUSH_OUT or CORR_PROFILE_COS_COMP_AND_PUSH_OUT has to be selected. Otherwise, PushOut is deactivated and thus ineffective
	Resolution	MB_RESOLUTION	Specification of the resolution of the cam calculation by determining the number of data points in three stages (details can be found in the data structure MB_RESOLUTION (for details, refer to chapter 10.2.5 "MB_RESOLUTION" on page 522)
	SyncMode	MC_SYNC_DIRECTION	Synchronizes the direction (Shortest_Way, Catch_Up, Slow_Down)
	StopDeceleration	REAL	At a negative edge at "Enable", the slave axis is stopped with the deceleration defined. Units according to axis scaling in rotary units
VAR_OUTPUT	InOperation	BOOL	CrossCutter runs without any failures
	Error	BOOL	Processing completed with error

I/O type	Name	Type	Comment
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Error Handling, page 510
	Shutdown	BOOL	The CrossSealer is stopped with the deceleration ramp "StopDeceleration" after a negative edge at "Enable". During this time, the output "ShutDown" is activated to display the defined stop
	ChangeProfile	BOOL	If input variables like, for example, FormatLength are changed during operation, a new MotionProfile is calculated and loaded in the background. Subsequently, the new MotionProfile is used. Since this process requires several cycles, this output is activated for this period. If any other input variables are changed during this process (e g. FormatLength), the current process is interrupted and performed again with new data. The output is reset once the newly calculated MotionProfile becomes effective and the switching phase was traveled through
	ProfileReady	BOOL	This output is set as soon as the MotionProfile was calculated, loaded and activated. The output is deleted as soon as the MotionProfile became effective (after having passed the switching angle).
	ActualProduct	UINT	In the case of more than one product per sealing process (ProductsPerCut > 1), the currently processed product is displayed via this output. At a rising edge at "Enable", this output is reset to the input variable "StartProduct".
	CutCounter	UINT	The CutCounter increases with each modulo overrun (360° → 0°) of the slave axis. At a rising edge at "Enable", the CutCounter is reset to zero

Fig.10-3: Interface variables of the MB_CrossSealType01 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
CrossCutter	BOOL			FALSE	Rising edge at "Enable"
StartMotionProfile	BOOL			TRUE	Continuous
ProductsPerCut	UINT	1	32	1	Rising edge at "Enable"
StartProduct	UINT	1	32	1	Rising edge at "Enable"
NumberOfKnives	UINT	1	8	1	Rising edge at "Enable"
StartCamTableID	MC_CAM_ID	1	94 (for MLC) 3 (for MLD)	1	Rising edge at "Enable"

RMB_TechCrossCutCrossSeal.library

Name	Type	Min. value	Max. value	Default value	Effective
KnifeDistance	REAL	> 0.0mm/Inch	n.def	100.0mm/Inch	Rising edge at "Enable"
FormatLength	REAL	> 0.2* KnifeDistance	30* KnifeDistance	100.0mm/Inch	Continuous*
CutSealArea	REAL	0° 0mm	< 360° < KnifeDistance	10° 10mm/Inch	Continuous*
PendelumFactor	REAL	0.0%	100%	0.0%	Continuous*
CorrectionProfile	MB_CORR_PROFILE	0	4	0	Continuous*
Overspeed	REAL	- 50.0%	+100.0%	0.0%	Continuous*
PushOut	MB_PUSH_OUT_CONFIG				Continuous*
Resolution	MB_RESOLUTION	RESOLUTION_LOW	RESOLUTION_HIGH	RESOLUTION_MEDIUM	Rising edge at "Enable"
SyncMode	MC_SYNC_DIRECTION	0	2	Catch_Up	Rising edge at "Enable"
StopDeceleration	REAL	> 0.0	n.def	1000.0	Continuous

* The acceptance of the inputs requires several cycles and is displayed at the "ChangeProfile" output

Fig.10-4: Min./max. and default values of the MB_CrossSealType01 function block

Functional Description

After the processing was enabled via **Enable**, the function block calculates a MotionProfile as well as cam point tables for the CrossSealer/CrossCutter application according to the specified input data. The function block also supports multiple CrossSealers/CrossCutters with equally distributed knives or sealing jaws.

For multiple CrossCutters or CrossSealers:

In the case of multiple CrossCutter or CrossSealer applications, the parameterization of the mechanical gear has to be adjusted (see also Necessary boundary conditions and requirements, page 509).

Therefore, the calculated motion sequence is divided into the following two sections:

- The **cutting or sealing area** is defined by the input variable **CutSealArea**. Within this area, the knife or the sealing jaw moves with the product velocity. In addition, other functions like, for example, **OverSpeed** for overspeed or a reduction of the velocity, **PushOut** for a push out motion or the **cosine correction** can be optionally added.

  In this area, the following input variables become effective:

  – **CorrectionProfile**

    It provides special profiles in the cutting and sealing area (OverSpeed, PushOut, cosine correction, PushOut + cosine correction).

  – **OverSpeed**

    It specifies the increase or reduction of the velocity in **CutSealArea** [%]

  - **PushOut**

    Defines a special PushOut motion during the cutting or sealing process

- **Outside** the cutting or sealing area, an optimum compensating motion with regard to jerk and energy is performed if the specified format length **FormatLength** does not correspond to the distance of the knives or the sealing jaws **KnifeDistance**.. Furthermore, a **pendular motion** of the CrossSealer axis or CrossCutter axis can be specified within adjustable limits to optimally use the acceleration capacity of the drive for large format lengths.

  The following input variable becomes effective in this area:

  - **PendelumFactor** specifies the maximum admissible pendulum radius [%]

The MotionProfile and the cam point tables are automatically loaded and activated after the calculation. During this period, the output **ChangeProfile** is set. If the MotionProfile is effective and if the slave axis is synchronized, the output **InOperation** is activated. In this state, the slave axis moves synchronously to the master axis using the previously calculated MotionProfile. In this state, the cut or seal counter **CutCounter** is increased with each revolution of the CrossCutter axis or CrossSealer axis.

If input variables like, for example, **FormatLength** are changed during the operation, a new MotionProfile is calculated in the background. Since this process requires several cycles, the output **ChangeProfile** is activated during this period. The output **ProfileReady** is set if the new MotionProfile has been activated but if the switching phase has not yet been traveled through. If the switching phase is reached subsequently, the calculated MotionProfile becomes effective and the outputs **ChangeProfile** and **ProfileReady** are reset. If any input data is changed again during the current change of the MotionProfile, the changes are discarded and the calculation with the new input data is restarted.

The data points of the cam point tables used within the MotionProfile can be affected via the **Resolution** input. A high resolution results in a maximum accuracy but requires the most time during the calculation. The time effort can be reduced by a factor of 3 or 9 using the settings "Medium" and "Low". The reduction of the resolution is useful if different product lengths are continuously processed with high cycle rates (> 500 cycles/minute).

A negative edge at **Enable** results in a defined shutdown of the CrossCutter/ CrossSealer. The slave axis is decelerated until standstill with the adjustable ramp **StopDeceleration**. During the defined shutdown, the output **Shutdown** is activated. The output **Shutdown** is reset if the slave axis has reached its standstill position or if an error occurred.

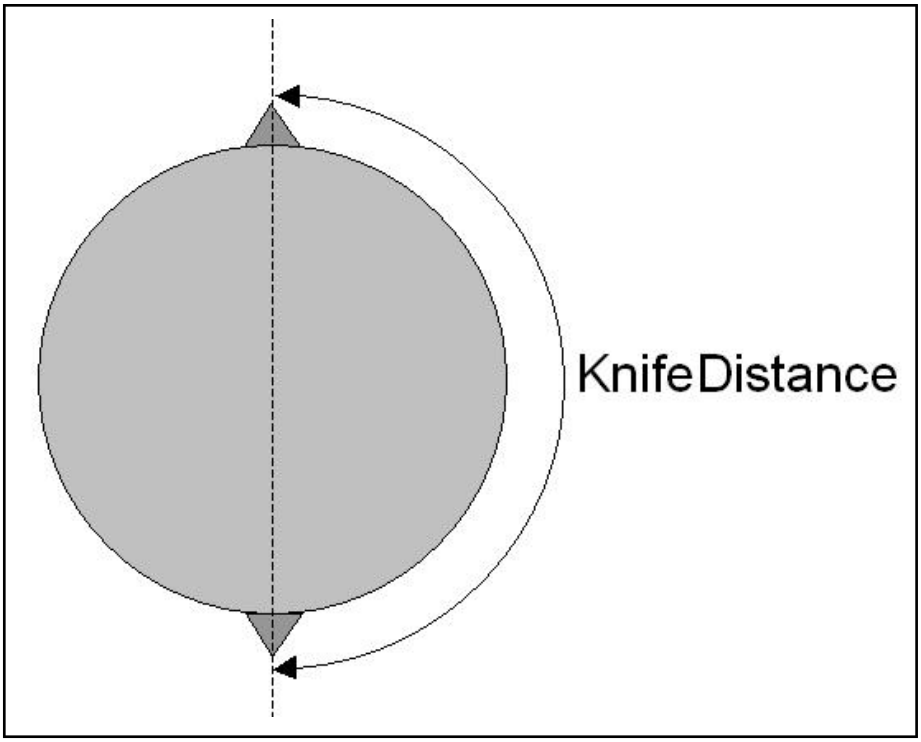


Fig.10-5: Definition of the input "KnifeDistance"

For scaling:

The translatory input variables of the function block (e. g. FormatLength) can be specified either in "mm" or in "inch". However, it must be taken into consideration that all translatory input variables are specified in the same unit (mm or inch).

CrossSealer use case

Via the **CrossCutter** input, the function block generally differentiates between the CrossCutter application and the CrossSealer use case:

If the input **CrossCutter** is FALSE, a cross sealer application is assumed. The triggering of a cross sealer cylinder of flow wrappers is a typical example for this use case. The sealing jaw should move with the product velocity and with an optimum correction profile during the sealing process. Outside the sealing range, a compensating motion is to be performed if the format length **FormatLength** does not correspond to the distance of the sealing jaws **KnifeDistance**. Furthermore, the CrossSealer can be optionally controlled by an upstream, cycle-synchronous lock on and lock off motion to realize the functions "NoGap/NoSeal" and "NoProduct/NoBag".

The CrossSealer use case provides the following characteristics:

- The **CutSealArea** input corresponds to the length of the sealing seam and is thus specified in translatory units (mm or inch)
- One master axis revolution (360°) corresponds to the foil feed of one product length
- One sealing process can include several products. The number of products per sealing process is specified via the **ProductsPerCut** input. This input directly influences the electronic gear of the MotionProfile used
- The first product is set via the **StartProduct** input after a positive edge at "Enable". This variable internally effects a position offset of the MotionProfile used. After a positive edge at "Enable", the output **ActualProduct** is set to the input **StartProduct**

- Optionally, the cycle-synchronous lock on and lock off can be connected upstream via another MB_CamLock function block

To start the function block:

If the master axis and the slave axis are at any position at the function block start, the slave axis performs a dynamic synchronization to synchronize with the master axis. If this is not desired, the master axis and the slave axis have to be set to the position 0° or 180° prior to the start of the function block.

The following figure shows the master axis structure and the effects of the function block for the CrossSealer use case with optional, cycle-synchronous lock on and lock off:

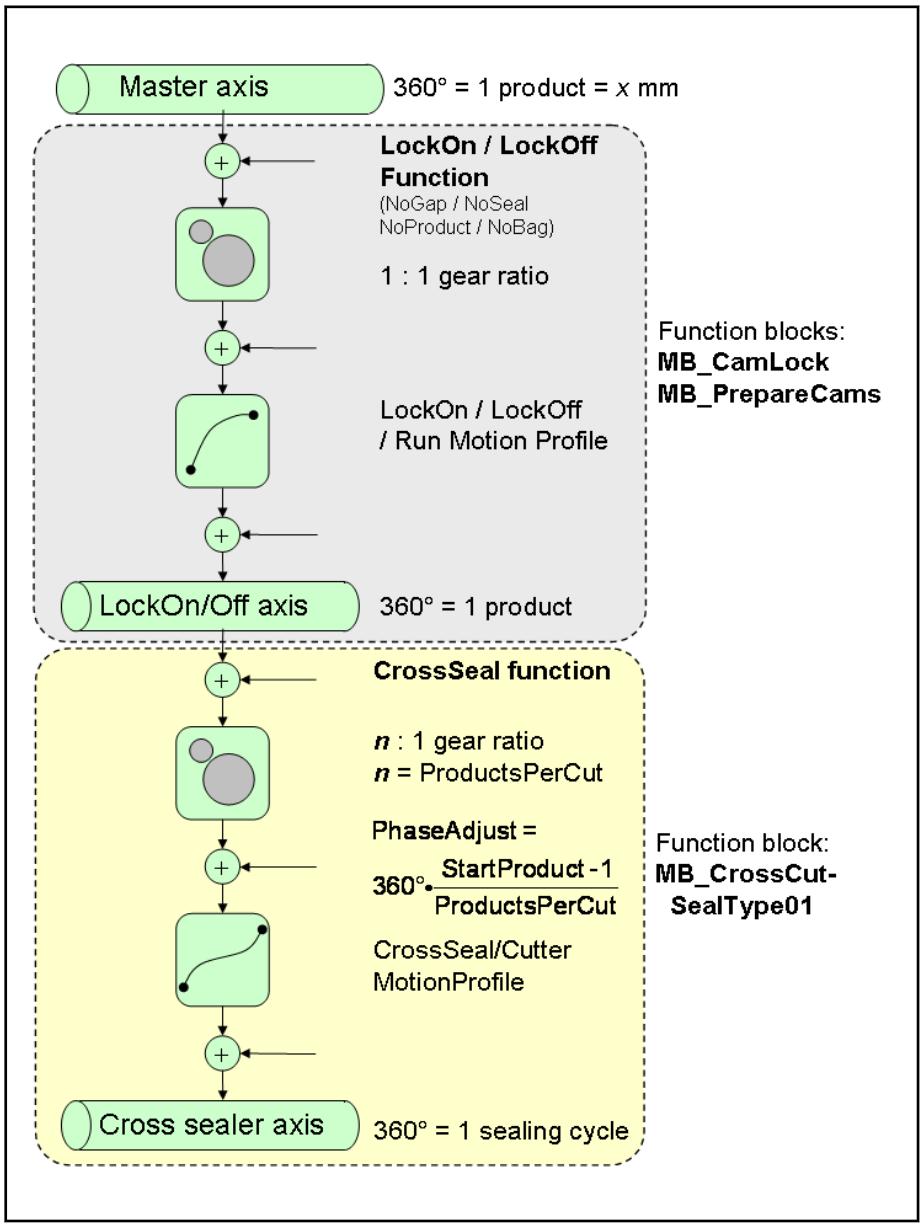


Fig. 10-6: Master axis structure for CrossSealer applications with lock on/lock off

The following figure shows an example of a signal-time diagram of the master axis and the CrossSealer axis for one product per package

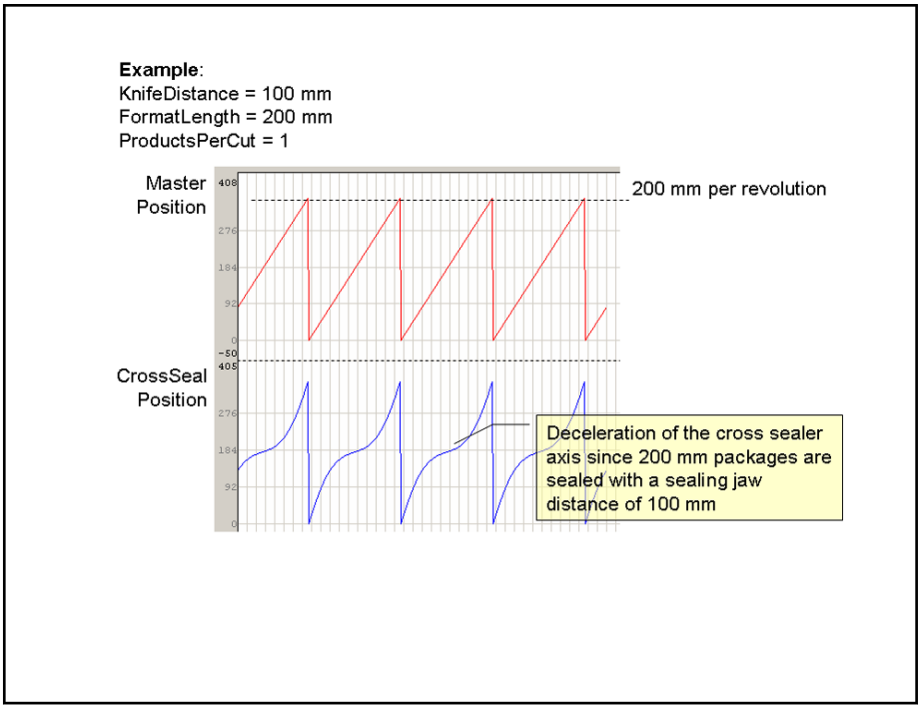


Fig.10-7: Example of a signal-time diagram with one product per sealing process

The following figure shows an example of a signal-time diagram of the master axis and the CrossSealer axis for 3 products per package:

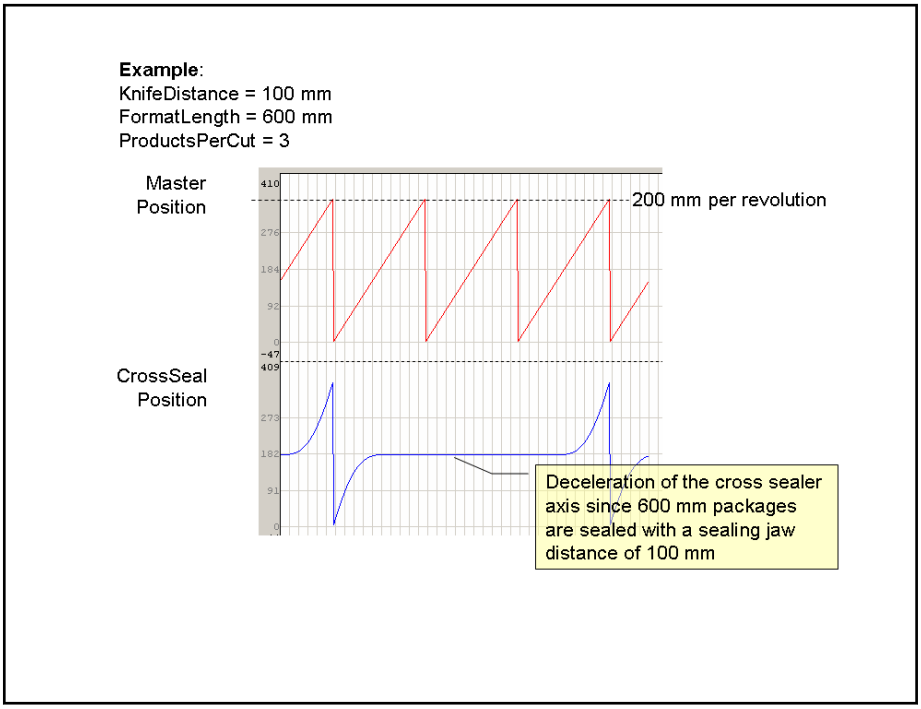


Fig.10-8: Example of a signal-time diagram with 3 products per sealing process

CrossCutter use case

Via the **CrossCutter** input, the function block generally differentiates between the CrossCutter application and the CrossSealer use case:

If the input **CrossCutter** is TRUE, a CrossCutter application is assumed. During the cutting process, the knife should move with the velocity of the product and with an optional correction profile. Outside the cutting area, a compen-

sating motion is to be performed if the format length **FormatLength** does not correspond to the distance of the sealing jaws **KnifeDistance**.

The CrossCutter use case provides the following characteristics:

- The input **CutSealArea** corresponds to the angular range of the CrossCutter in which it is traveling with the velocity of the product and the optional correction profile
- One revolution of the master axis corresponds to the material feed length which corresponds to the knife distance **KnifeDistance**
- One cut per product is assumed. The inputs **ProductsPerCut** and **StartProduct** are not effective (these inputs is internally set to 1)
- The output **ActualProduct** is always set to "1"
- The electronic gear of the MotionProfile used is set to the gear ratio **FormatLength**/**KnifeDistance**.

  This gear only allows integer values from 1 to 65535 as numerator and denominator of the gear ratio. Thus, there is a maximum cutting length deviation due to quantization.

  If a deviation of the cutting lengths occurs due to quantization, a respective gear ratio fine adjustment is calculated and specified. In this case, a drift between the master and the slave axis results due to the system conditions.

The following figure shows the master axis structure and the effects of the function block for the CrossCutter application:

RMB_TechCrossCutCrossSeal.library

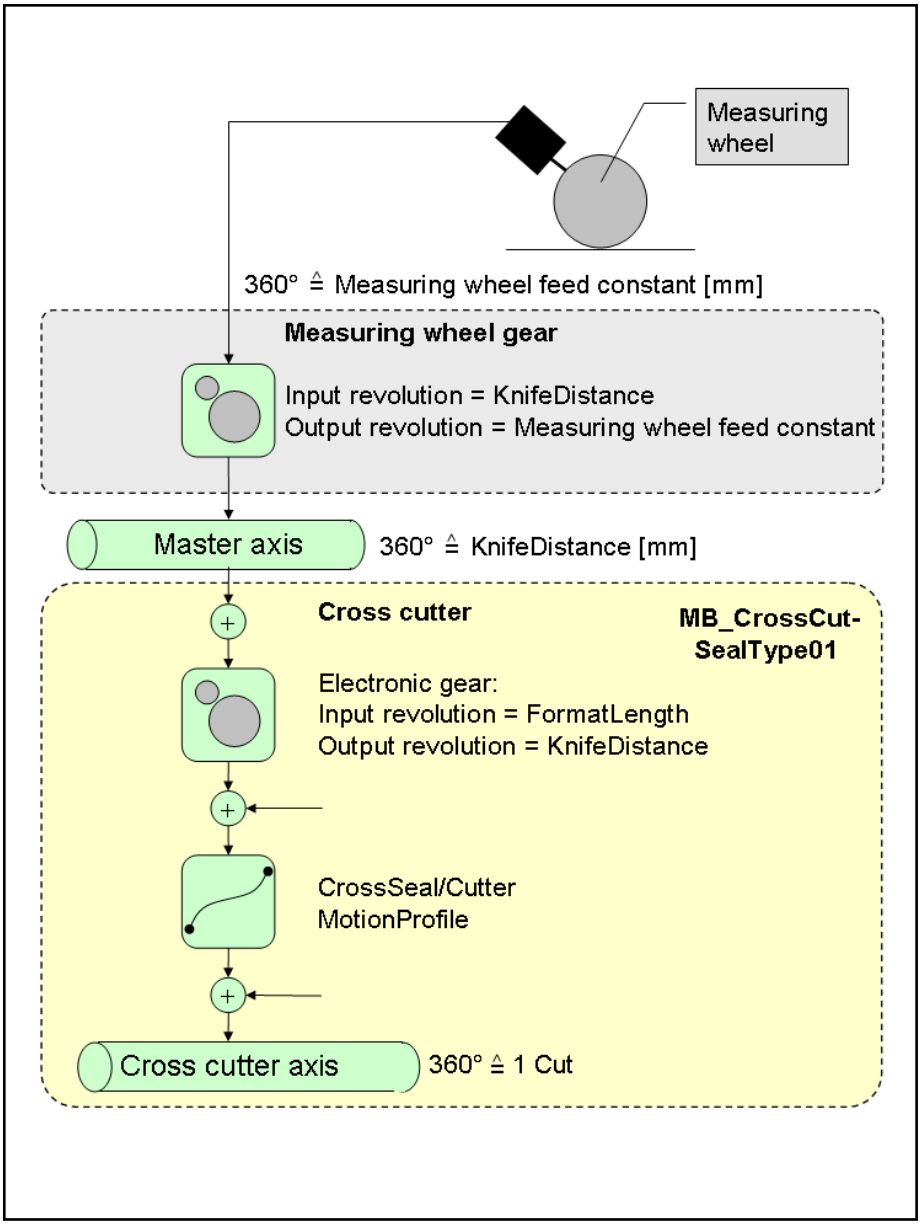


Fig.10-9: Master axis structure for the CrossCutter application

The following figure shows an example of a signal-time diagram of the measuring wheel axis, the master axis and the CrossCutter axis for a short format (format length = 0.5* knife distance):

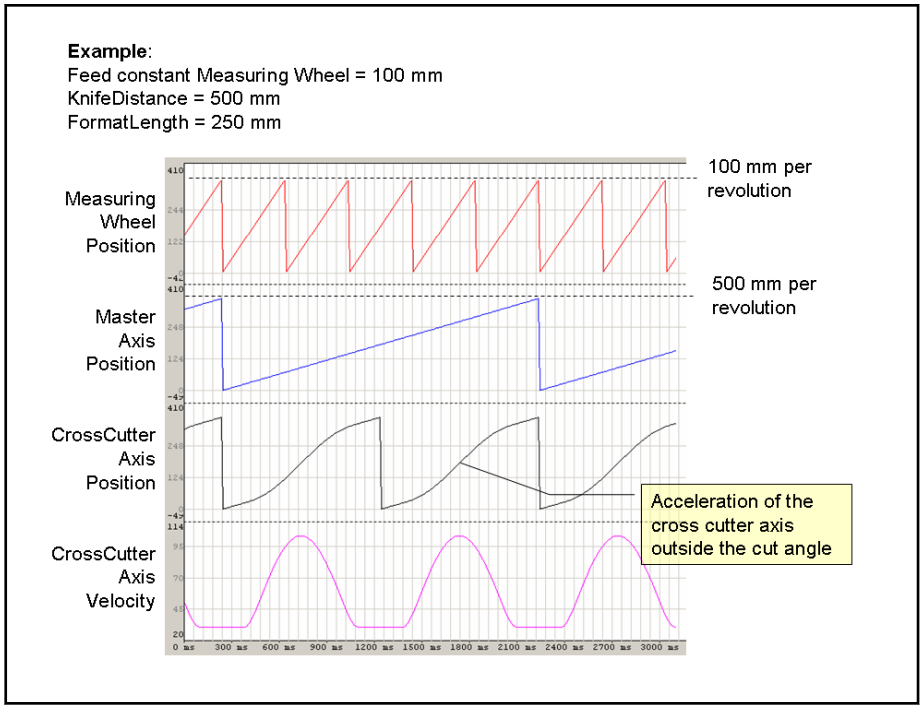


Fig. 10-10: Example of a signal-time diagram for a format length = 0.5 knife distance*

The following figure shows an example of a signal-time diagram of the measuring wheel axis, the master axis and the CrossCutter axis for a long format (format length = 2* knife distance):

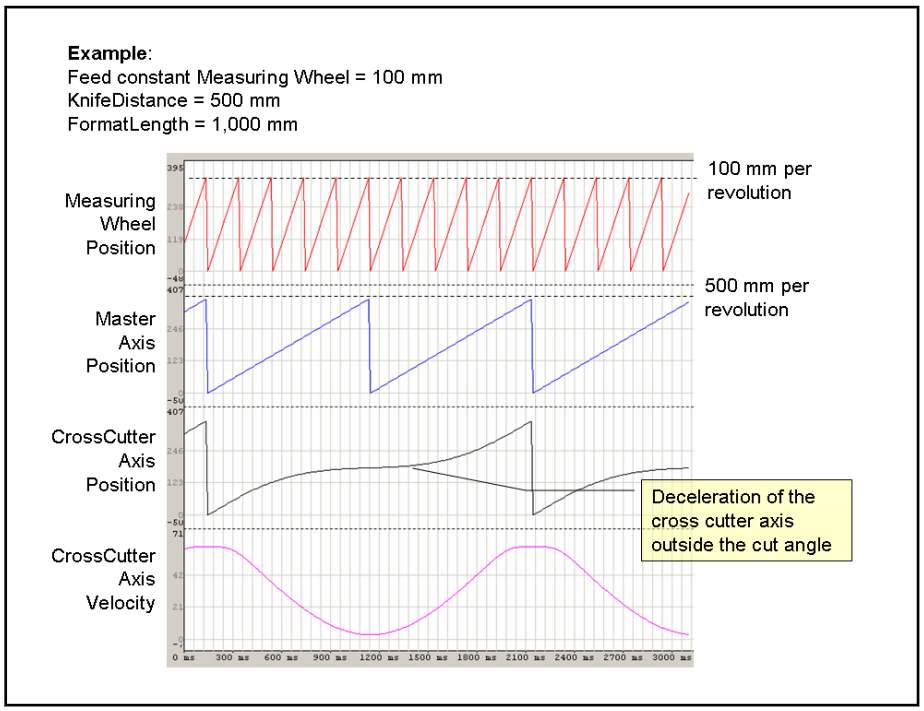


Fig. 10-11: Example of a signal-time diagram for a format length = 2 knife distance*

Necessary boundary conditions and requirements

- The sealing jaws/knives are symmetrically distributed over the circumference

- The 0° position of the slave axis is always exactly in the middle of CutSealArea

---

☞ **For drive parameterization:**

If there is more than one sealing jaw or knife per circumference, the settings for the mechanical gear (drive parameters: S-0-0121, S-0-0122) must be adapted. These parameters have to be set in a way so that a 360° electrical motion (modulo value of the slave axis) corresponds to the distance between two knives or sealing jaws.

Example of a gear configuration

The configuration of the mechanical gear when using a CrossCutter with four equidistantly distributed knives on the CrossCutter axis

Input variables:

Mechanical gear at the motor, input revolutions n_{input} = 10

Mechanical gear at the motor, output revolutions n_{output} = 1

Number of knives per circumference n_{knife} = 4

Calculation of the required gear setting:

$$\text{Mechanical gear} = \frac{\text{Input revolutions}}{\text{Output revolutions}} =$$

$$= \frac{n_{input}}{n_{output} \cdot n_{knife}} = \frac{10}{1 \cdot 4} = \frac{10}{4}$$

- S-0-0121 = 10
- S-0-0122 = 4

If, with this parameterization, the cutting cylinder moves mechanically by 90°, this corresponds to an electrical movement of 360° (= modulo value of the slave axis).

Error Handling

The function block does not perform an independent, motion-side error response. If an error is detected, no further Motion command is transmitted to the axis and the function block is exited via the **Error** output. A negative edge at "Enable" deletes the error output and the error state in the function block.

The function block uses the error table F_RELATED-TABLE (16#0170).

ErrorID	Additional1	Additional2	Description
ACCESS_ERROR (16#0004)	16#0003	16#0000	The function block was aborted by another function block
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0001	The "ProductsPerCut" input is not within the permitted range (1...32)
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0002	The "StartProduct" input is not within the permitted range (1...32)
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0003	The "NumberOfKnifes" input is not within the permitted range (1...8)
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0004	The "StartCamTableID" input is not within the permitted range (1...94)

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0005	The "KnifeDistance" input is not within the permitted range
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0006	The "FormatLength" input is not within the permitted range
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0007	The "CutSealArea" input is not within the permitted range
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0008	The "PendelumFactor" input is not within the permitted range
INPUT_RANGE_ERROR(16#0006)	16#0620	16#0009	The "Overspeed" input is not within the permitted range
INPUT_RANGE_ERROR(16#0006)	16#0620	16#000A	The "Resolution" input is not within the permitted range
INPUT_RANGE_ERROR(16#0006)	16#0621	16#0000	CutSealArea is too small with regard to the master axis velocity and the SERCOS cycle time

Fig.10-12: Error codes of the MB_CrossCutSealType01 function block

10.2 MB_CrossCutterCalcType04

10.2.1 General

Brief Description

The MB_CrossCutterCalcType04 function block provides all required data at its outputs which have to be loaded into the MotionProfile of an axis to operate a CrossCutter or a CrossSealer with the functions selected in the parameters of the function block. All inputs of the function block are transferred once at a rising edge at "Execute". In addition, the function block writes directly into the arrays specified by the pointers during its runtime. The written data is only valid if the function block was processed without errors and if it returns "Done".

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	RMB_TechCrossCutCrossSeal.compiled-library
IndraMotion MLD/MPx07VRS with MA function package	MX_Technology07.lib
IndraMotion MLD/MPx08VRS with MA function package	MX_Technology_08.lib (in preparation)
IndraMotion MLD/MPx17VRS with MA function package	MX_Technology_17.lib (in preparation)

Fig.10-13: Reference table of the MB_CrossCutterCalcType04 function block

RMB_TechCrossCutCrossSeal.library

Interface Description

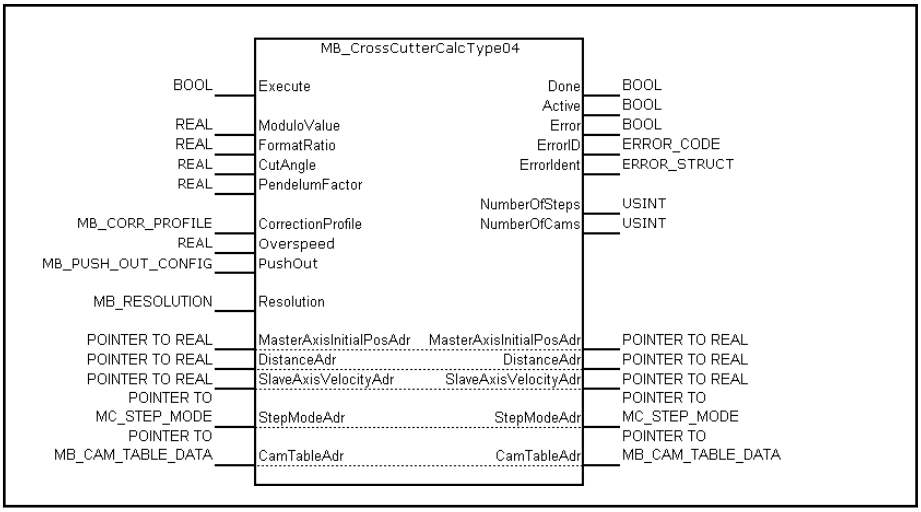


Fig.10-14: MB_CrossCutterCalcType04 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	MasterAxisInitialPosAdr	POINTER TO REAL	Pointer to an array with three elements to be created by the user in which the function block stores the starting positions of the master axis of the generated MotionProfile segments
	DistanceAdr	POINTER TO REAL	Pointer to an array with three elements to be created by the user in which the function block stores the distances of the generated MotionProfile segments
	SlaveAxisVelocityAdr	POINTER TO REAL	Pointer to an array with three elements to be created by the user in which the function block stores the velocities of the slave axis at the segment limits of the MotionProfile segments
	StepModeAdr	POINTER TO MC_STEP_MODE	Pointer to an array with three elements to be created by the user in which the function block stores the motion step modes of the generated MotionProfile segments.
	CamTableAdr	POINTER TO MB_CAM_TABLE_DATA	Pointer to an array with three elements to be created by the user in which the function block stores cam tables as well as meta data of the cams which are inserted as segments in the MotionProfile
VAR_INPUT	Execute	BOOL	Enabling the function block (once, edge-controlled)
	ModuloValue	REAL	Modulo value of the CrossCutter axis in [°]
	FormatRatio	REAL	Format ratio of the product to be cut (without unit)
	CutAngle	REAL	Length of the cutting area in [°]
	PendelumFactor	REAL	Maximum pendulum radius in [%] (0%: No pendular motion, 100%: Maximum pendular motion (pendular motion up to the cutting area limits)

I/O type	Name	Type	Comment
	CorrectionProfile	MB_CORR_PROFILE	Special profile selection. The inputs "Overspeed" or "PushOut" have to be used for the parameterization of the special profile. Further information on the options of the enumeration type MB_CORR_PROFILE can be found in the respective section
	Overspeed	REAL	Constant overspeed or velocity reduction in the cutting area [%] of the master axis velocity. (0%: No overspeed 100%: Double master axis velocity -50%: Half the master axis velocity) In the "CorrectionProfile", the CORR_PROFILE_OVERSPEED option has to be selected. Otherwise, "Overspeed" is deactivated and thus ineffective
	PushOut	MB_PUSH_OUT_CONFIG	Structure for a PushOut motion definition during the cut. In the "CorrectionProfile", either the option CORR_PROFILE_PUSH_OUT or CORR_PROFILE_COS_COMP_AND_PUSH_OUT has to be selected. Otherwise, "PushOut" is deactivated and thus ineffective. Further detailed information on the MB_PUSH_OUT_CONFIG structure can be found in the respective section.
	Resolution	MB_RESOLUTION	The resolution of the cam calculation is specified by determining the number of data points in three stages. Further information on the options of the enumeration type MB_RESOLUTION can be found in the respective section
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, the output contains detailed error information
	NumberOfSteps	USINT	Number of generated and to be downloaded MotionProfile segments
	NumberOfCams	USINT	Number of generated and to be downloaded cams

Fig.10-15: Interface variables of the MB_CrossCutterCalcType04 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

RMB_TechCrossCutCrossSeal.library

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
ModuloValue	REAL	>0.0°	n.def.	360.0°	At a rising edge at "Execute"
FormatRatio	REAL	0.2	30.0	1.0	At a rising edge at "Execute"
CutAngle	REAL	0.0°	< "ModuloValue"	12.0°	At a rising edge at "Execute"
PendelumFactor	REAL	0.0%	100.0%	0.0%	At a rising edge at "Execute"
CorrectionProfile	MB_CORR_PRO-FILE			CORR_PRO-FILE_NONE	At a rising edge at "Execute"
Overspeed	REAL	-50.0%	100.0%	0.0%	At a rising edge at "Execute"
PushOut	MB_PUSH_OUT_CONFIG				At a rising edge at "Execute"
Resolution	MB_RESOLU-TION			RESOLU-TION_HIGH	At a rising edge at "Execute"

Fig.10-16: Min./max. and default values of the MB_CrossCutterCalcType04 function block

Signal-time diagram

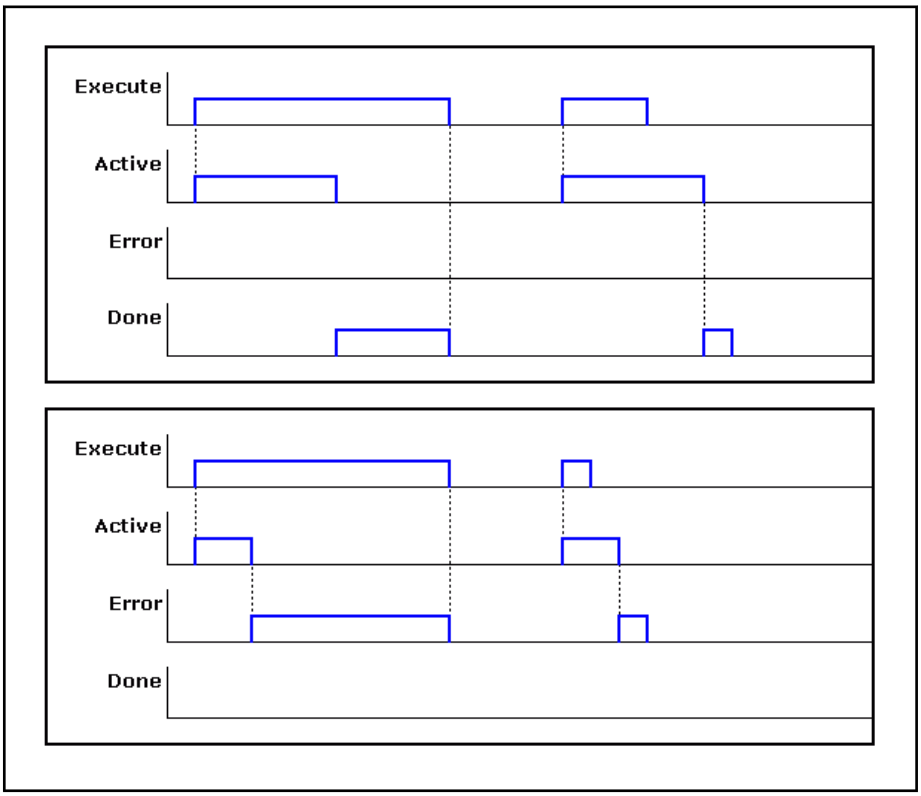


Fig.10-17: Signal-time diagram of the MB_CrossCutterCalcType04 function block

Functional Description

Based on the operation mode "Electronic MotionProfile", the function block calculates profiles for the "CrossCutter" application. The individual segments consist either of analytical segments or cam tables. After having enabled the processing with "Execute", an "Electronic MotionProfile" consisting of three

segments is calculated once based on the input variables. The parameters describing the profile are saved in the respective arrays which have to be provided by pointers at the inputs of the "MB_CrossCutterCalcType04" function block. The arrays have to be large enough to be able to save the number of calculated motion steps (currently three steps). The pointer handling is similar to the "MB_ChangeProfileStep"/"MB_ChangeProfileSet" function blocks since these function blocks have to be called by the user after the calculation to write the calculated profile to the respectively free set of the "Electronic MotionProfile".

The following list describes the meaning of the individual inputs of the "MB_CrossCutterCalcType04" function block in detail:

- The "ModuloValue" input specifies the modulo value of the CrossCutter axis. 360.0° are specified by default
- The "FormatRatio" input specifies the product length to be cut. It describes the ratio of the desired product length and the circumference of the CrossCutter cylinder. This means that if "FormatRatio==1", the length of the cut product corresponds to the circumference of the CrossCutter cylinder (synchronous format). If "FormatRatio==5", the length of the product cut is five times the circumference of the CrossCutter cylinder
- The "CutAngle" input specifies the cutting area of the CrossCutter. In the cutting area, the CrossCutter either rotates with the material feed velocity or with a (selectable) correction motion
- The "PendelumFactor" input specifies the maximum pendular motion of the CrossCutter axis in percent. This means that if "PendelumFactor==0%", the CrossCutter axis never rotates with a negative velocity (backwards). If "PendelumFactor==100%", the CrossCutter axis rotates to the border of the cutting area to minimize the energy loss and the required acceleration. This provides special benefits in case of higher values for "FormatRatio". If "PendelumFactor==100%", the CrossCutter does not generally rotate up to the border of the cutting area since this is not always advantageous. The respective optimum utilization of the available area is always automatically calculated by the function block
- The "CorrectionProfile" input allows different correction motions in the cutting zone required for the respective use case. The following can be selected:
  - "CorrectionProfile==CORR_PROFILE_NONE"

    (no correction: CrossCutter axis rotates with the master axis velocity)
  - "CorrectionProfile==CORR_PROFILE_OVERSPEED"

    (constant overspeed of the CrossCutter axis in the cutting area)
  - "CorrectionProfile==CORR_PROFILE_COS_COMP"

    (constant velocity vector in material direction, e. g. for guillotine cutters)
  - "CorrectionProfile==CORR_PROFILE_PUSH_OUT"

    (PushOut of the material) as well as
  - CorrectionProfile==CORR_PROFILE_COS_COMP_AND_PUSH_OUT

    (constant velocity vector in material direction as well as simultaneous PushOut, e.g. for guillotine cutters)

- The "Overspeed" input allows the exact setting of the overspeed in [%] in the cutting area if

  "CorrectionProfile==CORR_PROFILE_OVERSPEED"

  Thus, entering a value of 10 [%] increases the velocity of the CrossCutter axis in the cutting area by 10% compared to the material velocity
- The input "PushOut" parameterizes the PushOut correction

  The input is only processed if

  "CorrectionProfile==CORR_PROFILE_PUSH_OUT"

  or

  "CorrectionProfile==CORR_PROFILE_COS_COMP_AND_PUSH_OUT"
- The "Resolution" input allows the selection of the resolution of the cam tables used in 3 steps. Selecting a lower resolution saves calculation time, but reduces accuracy. High resolution increases the calculation time and the accuracy respectively

The successful calculation of a profile is signaled via the "Done" output. The parameters are written to the data arrays addressed by the provided pointers. Subsequently, these have to be written on a free MotionProfile using "MB_ChangeProfileSet"/"MB_ChangeProfileStep". The same procedure applies for the cams. It must be ensured that the table into which the cams have to be loaded is defined in the "MB_CAM_TABLE_DATA" structure. Then, the loaded MotionProfile is to be activated by using the "MB_MotionProfile" function block. The exact cutting length is defined with the electronic gear at the "MB_MotionProfile" function block. The gear ratio corresponds to the "FormatRatio" value. That means that a new profile is calculated for each product length possible. Each profile considers one input parameter only (especially "CutAngle" and "PendelumFactor". The following figure shows a sequential chart describing the usage of the MB_CrossCutterCalcType04 function block.

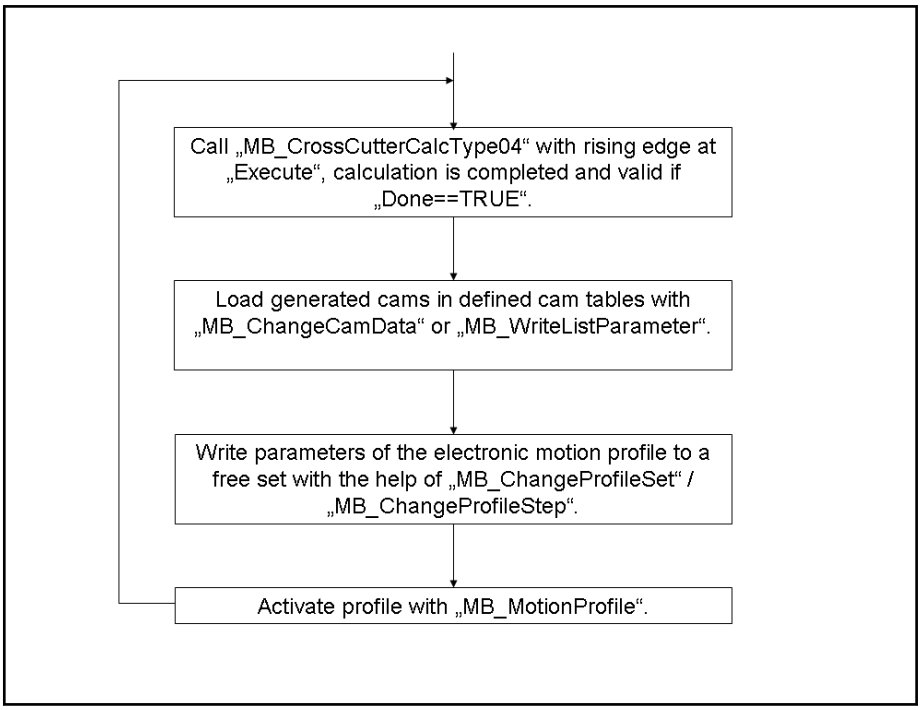


Fig.10-18: Sequence diagram for using MB_CrossCutterCalcType04

To use several equidistantly distributed knives, a mechanical gear is to be configured in IndraWorks with reference to the load side. The number of input

revolutions is set to 1 and the number of output revolutions is set to the number of the knives n used. Since no real mechanical gear exists, the drive only performs one n-fold revolution. Thus, the velocity of the slave axis in one revolution is n times identical to the velocity of the master axis.

Example:

The configuration of the mechanical gear when using a CrossCutter with four equidistantly distributed knives on the CrossCutter axis

Input variables:

Mechanical gear at the motor, input revolutions n_{input} = 10

Mechanical gear at the motor, output revolutions n_{output} = 1

Number of knives per circumference n_{knife} = 4

Calculation of the required gear setting:

$$\text{Mechanical gear} = \frac{\text{Input revolutions}}{\text{Output revolutions}} =$$

$$= \frac{n_{input}}{n_{output} \cdot n_{knife}} = \frac{10}{1 \cdot 4} = \frac{10}{4}$$

- S-0-0121 = 10
- S-0-0122 = 4

If, with this parameterization, the cutting cylinder moves mechanically by 90°, this corresponds to an electrical movement of 360° (= modulo value of the slave axis).

To adjust the master axis velocity and the slave axis velocity in the cutting area to the same value, the feed constant between the measuring wheel and the internal virtual master axis has to be adjusted. Therefore, the circumference of the internal virtual master axis is specified with the length between two knives of the real slave axis. This parameterization permits the cutting of each product with a different length at a corresponding preparation time.

RMB_TechCrossCutCrossSeal.library

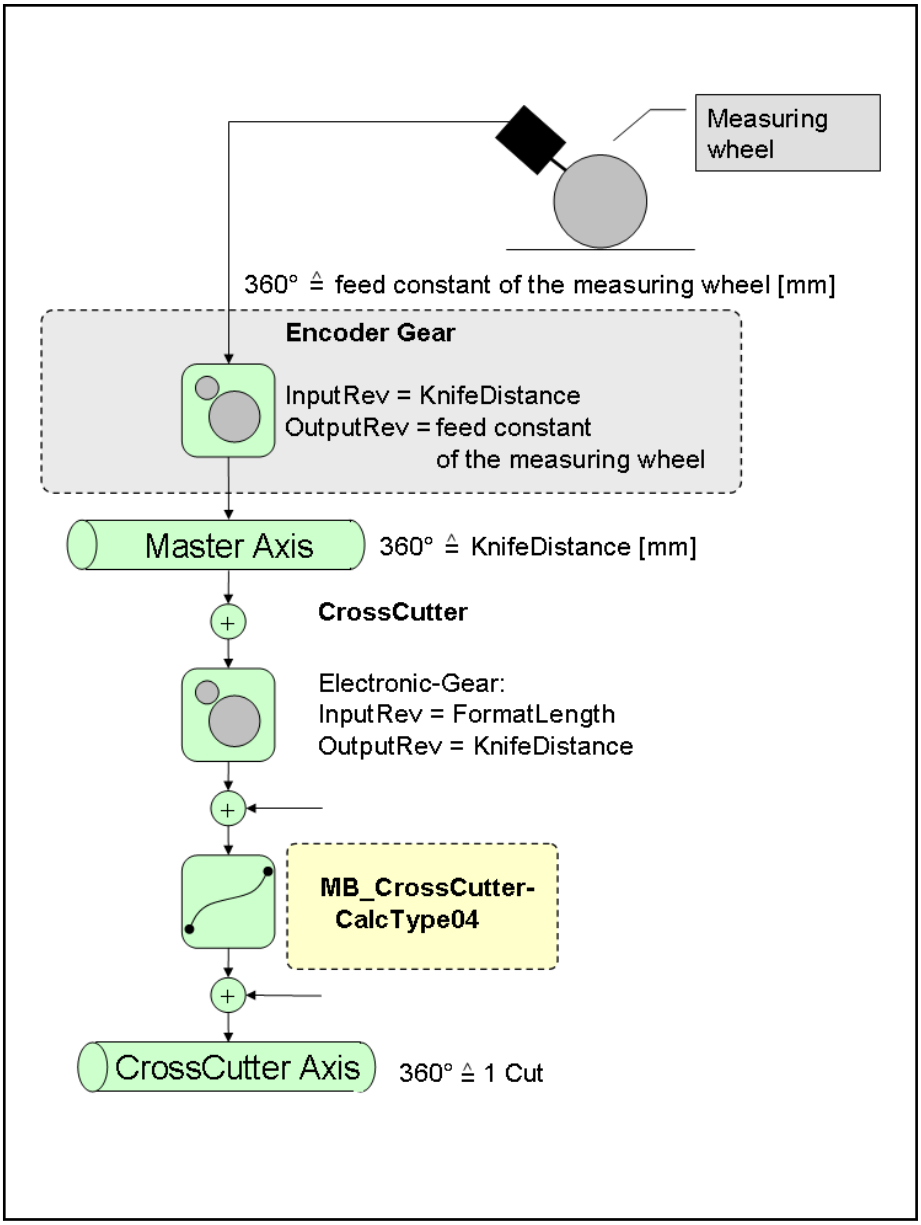


Fig.10-19: Master axis structure when operating a CrossCutter

Error Handling

The function block uses the error table **F_RELATED_TABLE**, 16#061x. It can generate the following error messages in Additional1 and Additional2:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0610	16#0001	Invalid value of the input "Modulo-Value"
INPUT_RANGE_ERROR(16#0006)	16#0610	16#0002	Invalid value of the input "Forma-tRatio"
INPUT_RANGE_ERROR(16#0006)	16#0610	16#0003	Invalid value of the input "CutAn-gle"
INPUT_RANGE_ERROR(16#0006)	16#0610	16#0004	Invalid value of the "CutAngle" in-put combined with the cosine com-pensation

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR(16#0006)	16#0610	16#0005	Invalid value of the input "PendulumFactor"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#0006	Invalid value of the input "CorrectionProfile"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#0007	Invalid value of the input "Overspeed"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#0008	Invalid starting point for the Push-Out function
INPUT_INVALID_ERROR(16#0001)	16#0610	16#0009	Invalid value for "Overspeed_At_Cut" in the structure "MB_PUSH_OUT_CONFIG"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#000A	Invalid value for "Overspeed_Max" in the structure "MB_PUSH_OUT_CONFIG"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#000B	Invalid combination of "Overspeed_At_Cut" and "Overspeed_Max"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#000C	Value of the "CutAngle" input too small combined with PushOut
INPUT_INVALID_ERROR(16#0001)	16#0610	16#000D	Invalid value of the input "Resolution"
INPUT_INVALID_ERROR(16#0001)	16#0610	16#000E	Invalid values of the "POINTER" inputs
STATE_MACHINE_ERROR (16#0005)	16#0006	xxyy	Invalid state of the state machine xx specifies the invalid step yy specifies the invalid state

Fig. 10-20: Error codes of the MB_CrossCutterCalcType04 function block

10.2.2 Special Features of IndraMotion MLC

Commissioning Notes

With regard to the IndraMotion MLC, the following commissioning notes are to be considered when operating the technology function block MB_CrossCutterCalcType04, downloading the generated MotionProfiles and cams and when linking the CrossCutter axis with the master axis:

- The modulo scaling with the modulo value 360° is to be activated for the CrossCutter axis
- The MB_MotionProfile function block always operates with the setting "0: Absolute synchronization without resynchronization" for the "StartMode" input to ensure the exact exchange of the profiles
- For the electronic gear, the MB_MotionProfile function block only allows integer values from 1 to 65535 as numerator and denominator of the gear ratio. Thus, there is a maximum cutting length deviation due to quantization. This can be minimized by multiplication with a big constant to be able to almost completely use the value range
- The generated MotionProfile can be downloaded on the IndraMotion MLC using the MB_ChangeProfileSet function block and the generated cams can be downloaded using either the MB_ChangeCamData func-

tion block or the MB_WriteListParameter function block. In contrast to the MB_WriteListParameter function block, the MB_ChangeCamData function block is faster. However, the cam is not saved persistently and has to be reloaded after a control reboot

When downloading the cams, it is to be ensured that the respective cam is loaded at its assigned position (see "CamTableID" element in the "Interface variables of the MB_CAM_TABLE_DATA function block" table in chapter 10.2.3 "MB_CAM_TABLE_DATA" on page 520).

- A small format ratio (< 1) should be selected carefully since the CrossCutter axis reaches very high velocities within the compensation range (up to eight times the master axis velocity)
- Bit 1 is to be set in the parameter A-0-2930 of the CrossCutter axis so that the gear is exchanged at the same time as the MotionProfile at a switching phase of 0° and not when calling the MB_MotionProfile function block

Format ratio	Ratio of the slave axis velocity to the master axis velocity
1	1
0,8	1,5
0,6	2,2
0,4	3,7
0,2	8

Fig.10-21: Effects of the format ratio on the maximum velocity of the CrossCutter axis within the compensation range

10.2.3 MB_CAM_TABLE_DATA

Brief Description

The MB_CAM_TABLE_DATA structure is used by the MB_CrossCutterCalcType04 function block. It contains an array with a cam table as well as meta data describing the number of array points as well as the position at which the table has to be loaded into the control/drive.

Interface Description

The following table lists the interface variables of the function block.

Structure element	Type	Description
CamTableID	DINT	Number of the cam table into which the described cam has to be loaded
NumberOfElements	UINT	Number of points of the described cam
CamTable	ARRAY [0...1023] OF DINT	Array with the cam data points

Fig.10-22: Interface variables of the MB_CAM_TABLE_DATA structure

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value
CamTableID	DINT	0		0
NumberOfElements	UINT	0	1024	0

Fig.10-23: Min./max. and default values of the MB_CAM_TABLE_DATA structure

Functional Description

In the structure element "CamTableID", the number of the cam table is saved in the format required by the MB_ChangeCamData, MB_ChangeProfileStep

and MB_ChangeProfileSet function blocks. fig. 10-24 "Assigning CamTableID to parameter numbers" on page 521 shows the mappings between "CamTableID" and the corresponding parameter number in the MLC and MLD.

CamTableID	Parameter number IndraMotion MLC	Parameter number IndraMotion MLD
1	C-0-2001	P-0-0072
2	C-0-2002	P-0-0092
3	C-0-2003	P-0-0780
4	C-0-2004	P-0-0781
5	C-0-2005	P-0-0784
6	C-0-2006	P-0-0785
7-99	C-0-2007 - C-0-2099	Not available

Fig. 10-24: Assigning CamTableID to parameter numbers

The following diagram shows how to use the data of the MB_CAM_TABLE_DATA structure to load the cam either into the control or into the drive.

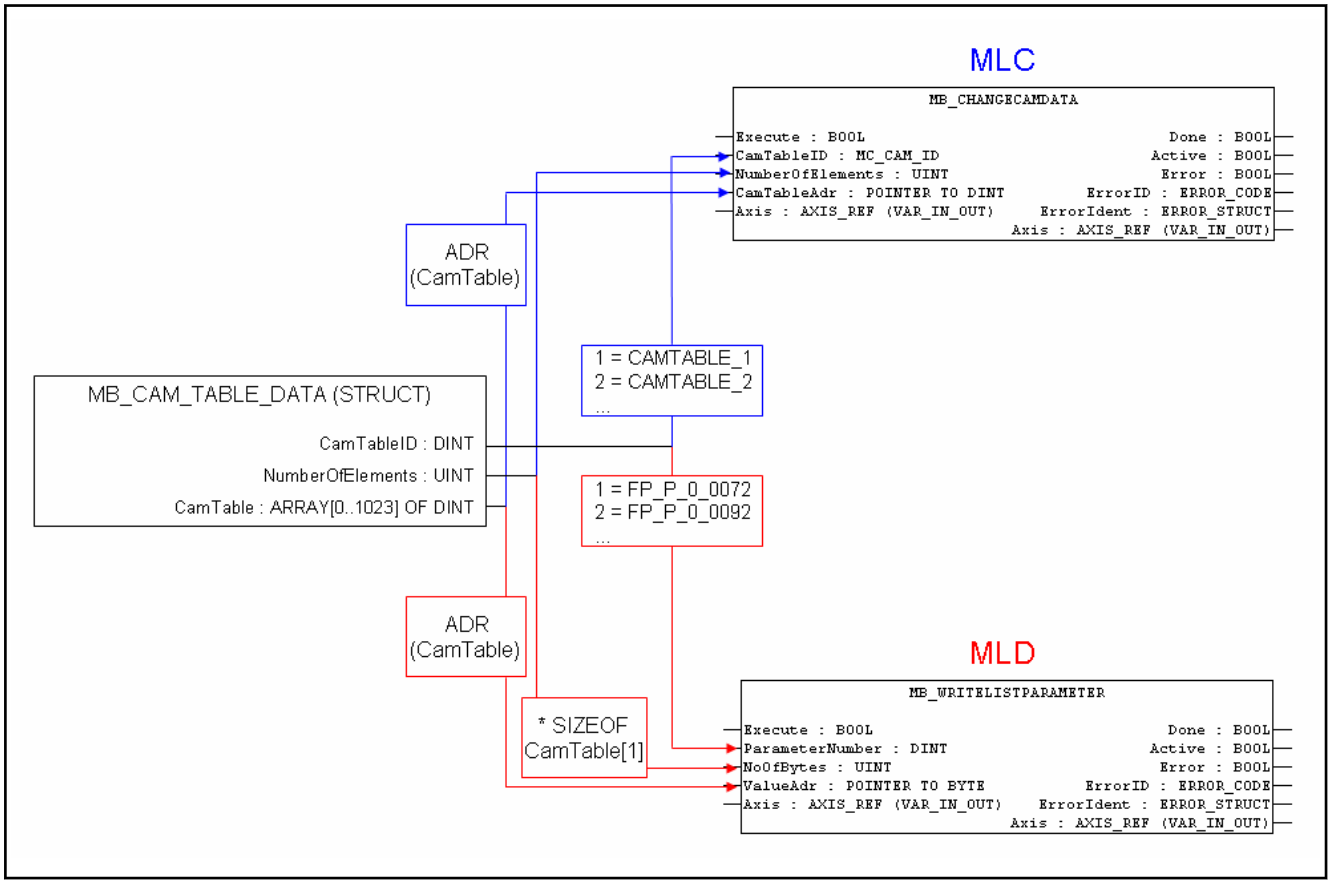


Fig. 10-25: Connection plan of the MB_CAM_TABLE_DATA structure

10.2.4 MB_CORR_PROFILE

Brief Description

The enumeration MB_CORR_PROFILE is used by the MB_CrossCutterCalcType04 function block to define a correction motion in the cutting area. The possible elements of the enumeration type MB_CORR_PROFILE as well as their meanings are described in the following.

RMB_TechCrossCutCrossSeal.library

Enumeration type elements The following table lists the elements of the enumeration type.

Element	Value	Description
CORR_PROFILE_NONE	0	No correction in the cutting zone. The CrossCutter axis velocity corresponds to the master axis velocity
CORR_PROFILE_OVERSPEED	1	Constant overspeed or reduction of the CrossCutter axis velocity in the cutting area. The CrossCutter axis velocity is faster or slower than the master axis velocity by the value parameterized at the "Overspeed" input of the MB_CrossCutterCalcType04 function block
CORR_PROFILE_COS_COMP	2	Correction of the CrossCutter axis velocity The velocity is changed in a way that the CrossCutter axis rotates in material direction (the x component of the velocity vector) at a constant velocity. The speed of the CrossCutter axis has to be adapted depending on the current position in the cut
CORR_PROFILE_PUSH_OUT	3	Executing a PushOut motion at the end of the cut. Parameterization via the "PushOut" input of the MB_CrossCutterCalcType04 function block. The speed of the CrossCutter axis has to be adapted depending on the current position in the cut. See also chapter 10.2.6 "MB_PUSH_OUT_CONFIG" on page 523
CORR_PROFILE_COS_COMP_AND_PUSH_OUT	4	Combination of the corrections "CORR_PROFILE_COS_COMP" and "CORR_PROFILE_PUSH_OUT". Parameterization via the "PushOut" input of the MB_CrossCutterCalcType04 function block. See also chapter 10.2.6 "MB_PUSH_OUT_CONFIG" on page 523

Fig.10-26: Elements of the enumeration type MB_CORR_PROFILE

10.2.5 MB_RESOLUTION

Brief Description The enumeration MB_RESOLUTION is used by the MB_CrossCutterCalcType04 function block. With this enumeration, the number of cam table data points and thus the required calculation time can be modified. The elements of the enumeration MB_RESOLUTION as well as their meanings are described in the following.

Enumeration type elements The following table lists the elements of the enumeration type.

Element	Value	Description	Number of data points (cutting segment + compensating segment + cutting segment)
RESOLUTION_LOW	1	Low resolution, short calculation time required (faster by approx. factor 3 in contrast to medium resolution)	10 + 100 + 10
RESOLUTION_MIDDLE	2	Medium resolution, average calculation time required (faster by approx. factor 3 in contrast to high resolution)	25 + 300 + 25
RESOLUTION_HIGH	3	High resolution, much calculation time required	50 + 1000 + 50

Fig.10-27: Elements of the enumeration type MB_RESOLUTION

Functional Description

With the enumeration MB_RESOLUTION, the number of cam table data points and thus the required calculation time can be modified.

For short formats and little preparation time for a format change, the calculation time of the function block can be reduced selecting the "RESOLUTION_LOW" option. However, it is be taken into account that the resolution of the generated cams is reduced and that the machine might not be running smoothly. For long formats and high preparation times for format changes, the "RESOLUTION_HIGH" option can be selected to generate the created cams with the highest resolution possible. Basically, this setting ("RESOLUTION_HIGH") it to be preferred.

The option to be selected depends on the application used and is to be specified by means of tests during the commissioning.

10.2.6 MB_PUSH_OUT_CONFIG

Brief Description

The MB_PUSH_OUT_CONFIG structure is used by the MB_CrossCutterCalcType04 function block. This structure can parameterize a defined PushOut at the end of the cut. The elements of the MB_PUSH_OUT_CONFIG structure as well as their minimum, maximum and default values are shown in the following interface description.

Interface Description

The following table lists the interface variables of the function block.

Structure element	Type	Description
PushOutBegin	REAL	Beginning of the PushOut area (parameter A). The value must not be equal to 0, unit in [°]
Overspeed_At_Cut	REAL	Overspeed during a cut (parameter B). The value must not be equal to 0, unit in [%]
Overspeed_Max	REAL	Maximum overspeed (parameter C), unit in [%]

Fig.10-28: Interface variables of the MB_PUSH_OUT_CONFIG structure

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

RMB_TechCrossCutCrossSeal.library

Name	Type	Min. value	Max. value	Default value
PushOutBegin	REAL	$-\frac{"CutAngle"}{2}$	$\frac{"CutAngle"}{2}$	-5.0°
Overspeed_At_Cut	REAL	-100.0%	100.0%	5.0%
Overspeed_Max	REAL	>0.0%	100.0%	10.0%

Fig.10-29: Min./max. and default values of the MB_PUSH_OUT_CONFIG structure

Functional Description

The following graphics display the velocity characteristics of the CrossCutter roll in the cutting area. The individual parameters can be adjusted using the elements of the MB_PUSH_OUT_CONFIG structure.

A curve profile with a cut angle of 20° has been assumed for the graphics. The degree of the velocity change is determined with the help of the two variables "Overspeed_At_Cut" and "Overspeed_Max". The beginning of the velocity increase is determined with the help of "PushOutBegin". For a unique parameterization, it is required that "PushOutBegin" and "Overspeed_At_Cut" are not equal to zero. Both figures show an example of the parameterization. The green line characterizes the degree of the increase and can be determined by connecting the points A, B and C.

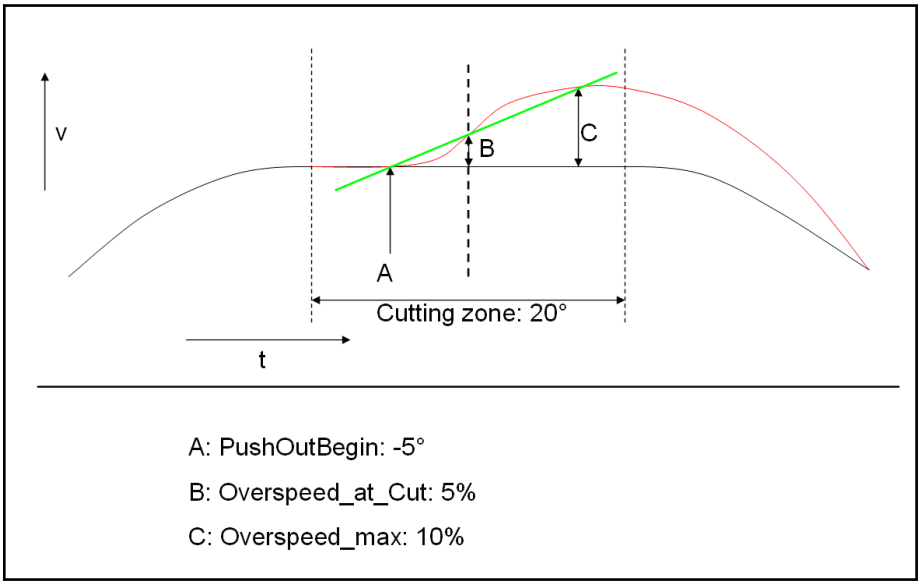


Fig.10-30: Definition of PushOut.PushOutBegin < 0

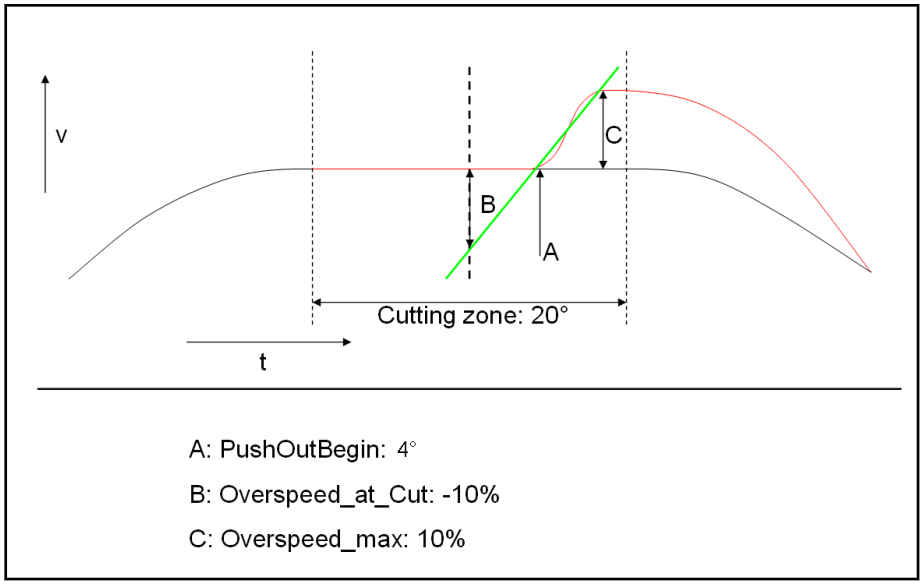


Fig. 10-31: Definition of PushOut.PushOutBegin > 0

11 RMB_TechCam.library

11.1 Introduction and Overview

The library described includes function blocks to create cam tables.

The function blocks are available for the different applications such as feed motion, CrossCutter etc.

The function blocks chapter 11.2 "MB_FeedMovementCalcType01" on page 527, chapter 11.3 "MB_FeedMovementCalcType02" on page 533 and chapter 11.4 "MB_CoordinatePairCamType01" on page 540 operate with the enumeration chapter 5.14.3 "MB_CAMPROF" on page 156 and the structure chapter 5.14.4 "MB_CAMTABLE" on page 157 located in the "ML_TechBase.compiled-library". Furthermore, cam tables of the mentioned function block can be loaded via the chapter 5.14.2 "MB_LoadCamData" on page 154 function block of the "ML_TechBase.compiled-library" library to a cam in the control or drive.

Overview on Function Blocks

Function block	Description
MB_FeedMovementCalcType01	Calculates a cam table with up to 1,024 data point elements for a feed motion
MB_FeedMovementCalcType02	Calculates a cam table with up to 1,024 data points with a polynomial 7th order in which the inputs of the jerk, acceleration and velocity can be limited in order to ensure optimum processing
MB_CoordinatePairCamType01	Calculates a cam table with up to 1,024 data point elements that are determined depending on the selected coordinate pair type

Fig.11-1: RMB_TechCam.compiled-library

11.2 MB_FeedMovementCalcType01

Brief Description

The function block calculates a cam table with up to 1,024 data point elements for a feed motion.

Assignment: Target system/library

IndraMotion MLC 12VRS	RMB_TechCam.compiled-library

Fig.11-2: Reference table of the MB_FeedMovementCalcType01

Interface Description

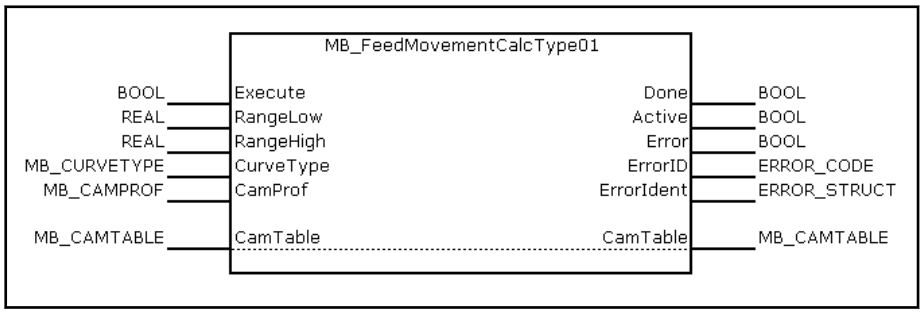


Fig.11-3: MB_FeedMovementCalcType01 function block

RMB_TechCam.library

I/O type	Name	Type	Comment
VAR_IN_OUT	CamTable	MB_CAMTABLE	Structure with the calculated cam table elements and information on the number of elements
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	RangeLow	REAL	Starting point of the feed motion. First point of the MotionProfile
	RangeHigh	REAL	End point of the feed motion. Last point of the MotionProfile
	CurveType	MB_CURVETYPE	Curve type selection
	CamProf	MB_CAMPROF	Cam profile selection
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid, error outputs valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, it contains a detailed error description (see "Error Handling" on page 533)

Fig.11-4: Interface variables of the MB_FeedMovementCalcType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
RangeLow	REAL	0.0	<360.0	0.0	Rising edge at "Execute"
RangeHigh	REAL	>0.0	360.0	360.0	Rising edge at "Execute"
CurveType	MB_CURVETYPE			CURVE-TYPE_POLY_5	Rising edge at "Execute"
CamProf	MB_CAMPROF			CAM_PROF_STANDARD	Rising edge at "Execute"

Fig.11-5: Min./max. and default values of the MB_FeedMovementCalcType01 function block

The "MB_CURVETYPE" enumeration selects between different compensation motions from rest-to-rest.

Element	Value	Description
CURVETYPE_QUADRATIC_PARABLE	0	Square parabola
CURVETYPE_SIMPLE_SINUSLINE	1	Simple sine line
CURVETYPE_POLY_5	2	Polynomial 5th order
CURVETYPE_SLOPING_SINUSLINE	3	Inclined sine line

Element	Value	Description
CURVETYPE_MODIFIED_ACCELERATION-TRAPEZE	4	Modified acceleration trapezoid
CURVETYPE_MODIFIED_SINUSLINE	5	Modified sine line

Fig.11-6: Elements of the enumeration type "MB_CURVETYPE"

The enumeration chapter 5.14.3 "MB_CAMPROF" on page 156 selects between different cam profiles.

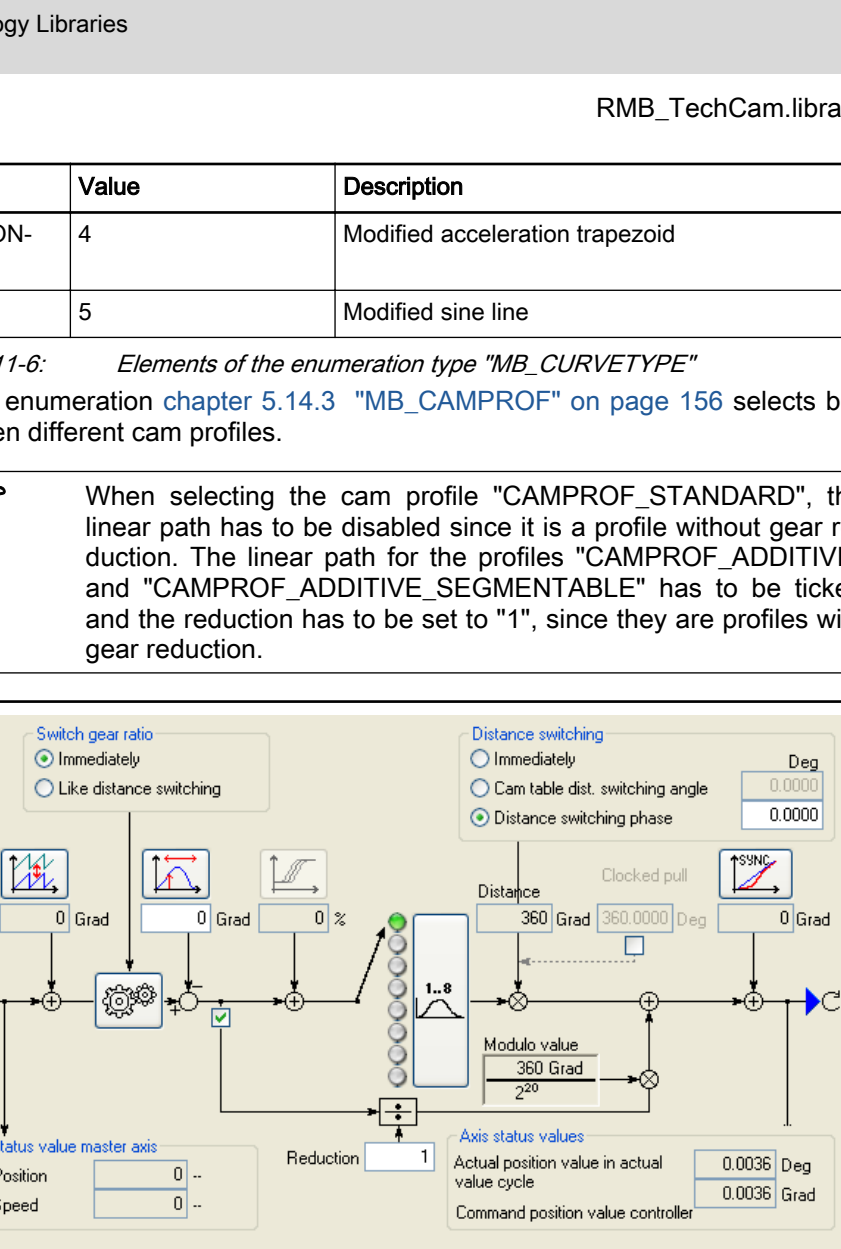


Fig.11-7: Enabling the linear path

The structure chapter 5.14.4 "MB_CAMTABLE" on page 157 is provided with the "CamTable" array containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

Functional Description

With the specification of a feed range defined by the limits "RangeLow" and "RangeHigh", the ML_FeedMovementCalcType01 function block calculates a cam table with 1,024 data point elements one time after enabling the processing with "Execute" based on the input values. The number of data point elements of a cam table is set by the "NumberOfElements" element of the "CamTable" structure.

The motion from rest-to-rest can be carried out with different compensation motions. The adjustable motions are:

- Square parabola
- Simple sine line
- Polynomial 5th order
- Inclined sine line

- Modified acceleration trapezoid
- Modified sine line

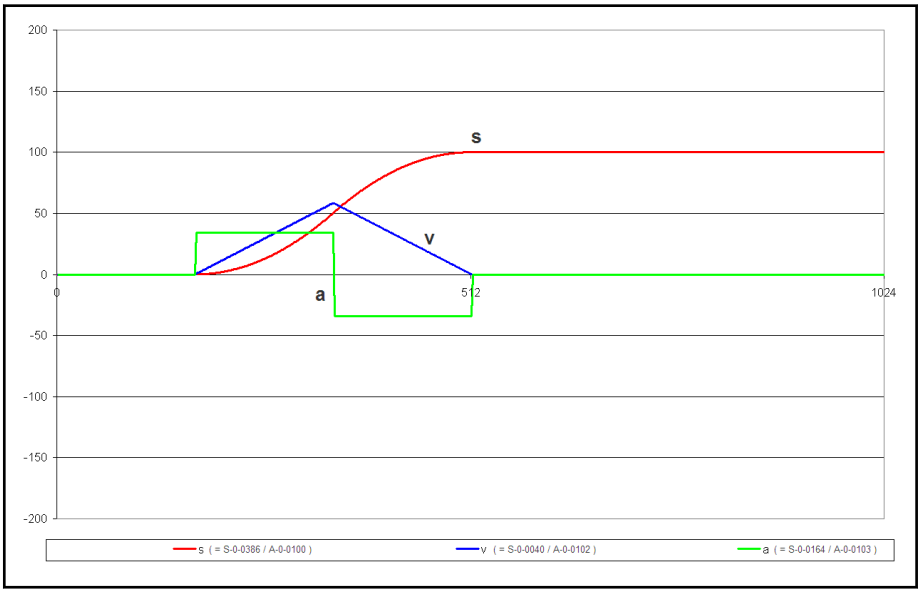
The cam table to be calculated is stored in the "CamTable" structure with scaling in per cent. The validity of the cam elements is signaled by the "Done" output. The cam table can be loaded to the control or drive using the chapter 5.14.2 "MB_LoadCamData" on page 154 function block.

If an error occurs while processing the function block, this is signaled by the "Error" output. The elements of the "CamTable" structure are not updated in case of error.

Cam characteristics

The cam table of the MB_FeedMovementCalcType01 provides the following characteristics:

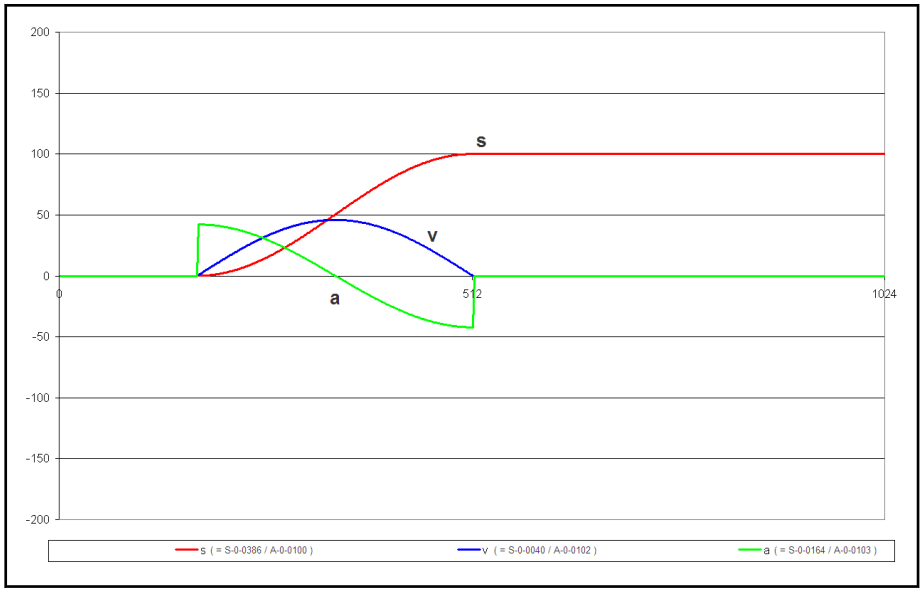
- The cam table generated by the function block describes a rest-in-rest motion
- The algorithm to calculate the cam data points depends on a difference (RangeHigh-RangeLow) of 1,026 cycles max, i.e. calls of the instance to avoid a considerable effect on the PLC cycle time
- The cam profile is provided for the cyclic processing in the drive as parameters P-0-0072, P-0-0092 etc.
- CurveType: CURVETYPE_QUADRATIC_PARABLE
  - The MotionProfile characterized by the cam is constant up to the acceleration (s, v, a) and shows a jump in the jerk (j)



RangeLow = 60; RangeHigh = 180

Fig.11-8: MB_FeedMovementCalcType01, MotionProfile QUADRATIC_PARABLE

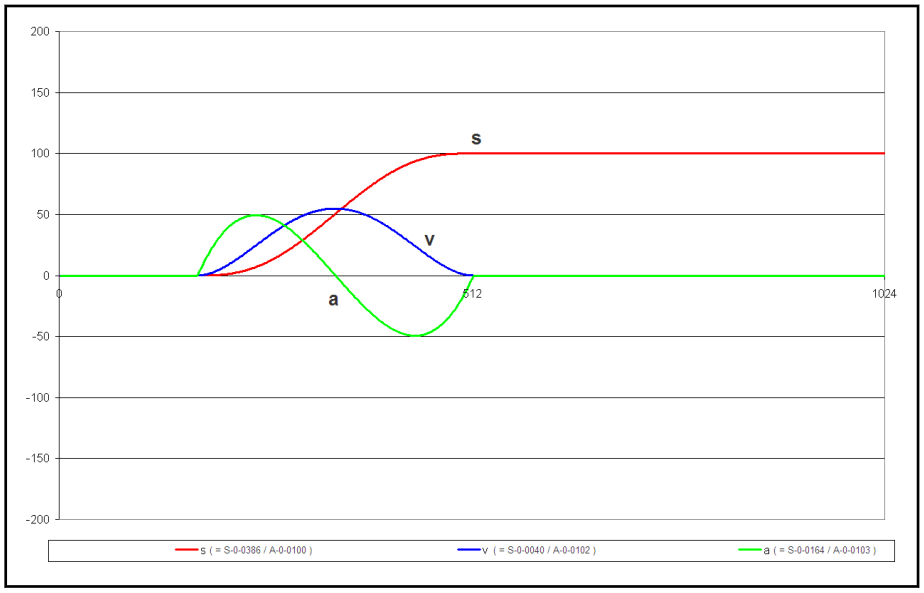
- CurveType: CURVETYPE_SIMPLE_SINUSLINE
  - The MotionProfile characterized by the cam is constant up to the acceleration (s, v, a) and shows a jump in the jerk (j)



RangeLow = 60; RangeHigh = 180

Fig.11-9: MB_FeedMovementCalcType01, MotionProfile SIMPLE_SINUS-LINE

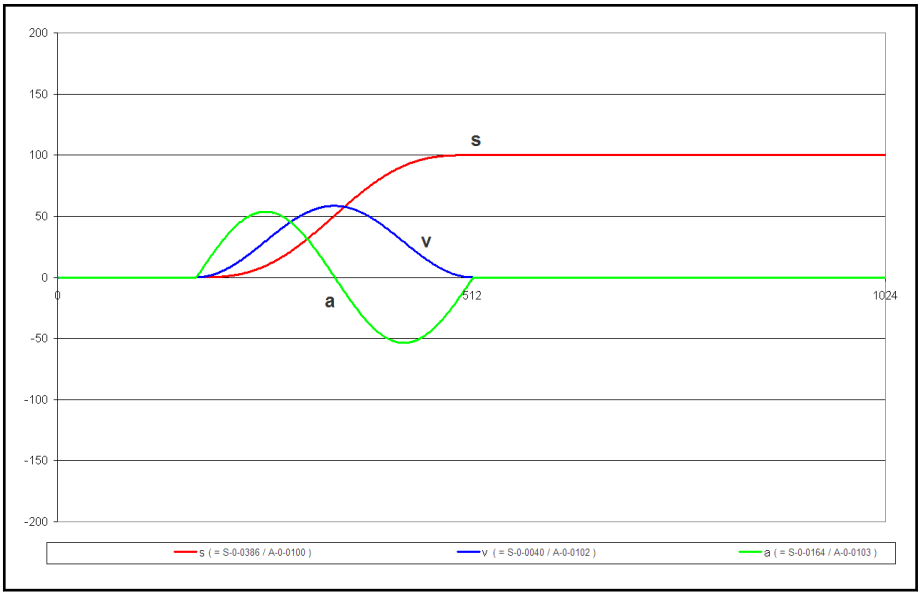
- CurveType: CURVETYPE_POLY_5
  - The MotionProfile characterized by the cam is constant up to the jerk (s, v, a, j)



RangeLow = 60; RangeHigh = 180

Fig.11-10: MB_FeedMovementCalcType01, MotionProfile POLY_5

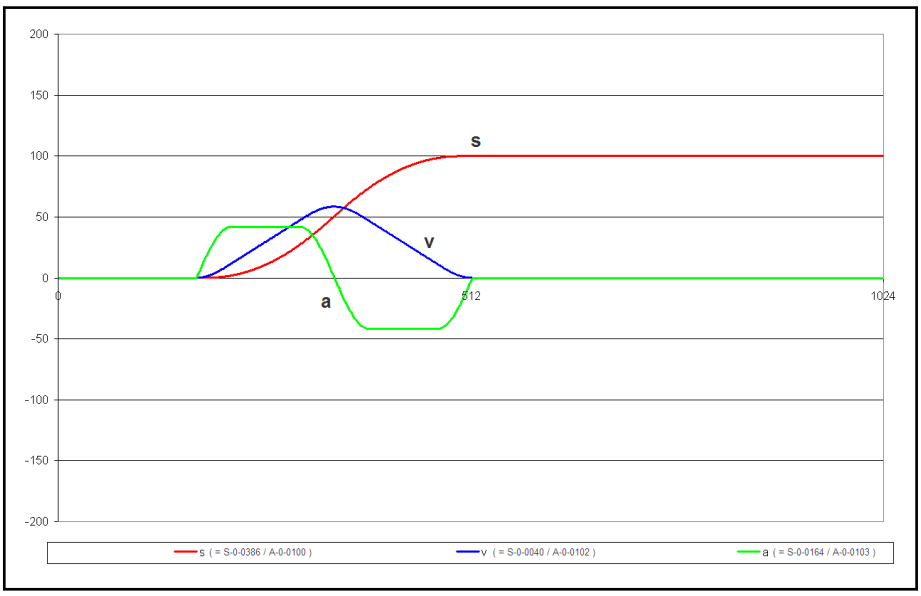
- CurveType: CURVETYPE_SLOPING_SINUSLINE
  - The MotionProfile characterized by the cam is constant up to the jerk (s, v, a, j)



RangeLow = 60; RangeHigh = 180

Fig.11-11: MB_FeedMovementCalcType01, MotionProfile SLOPING_SINUSLINE

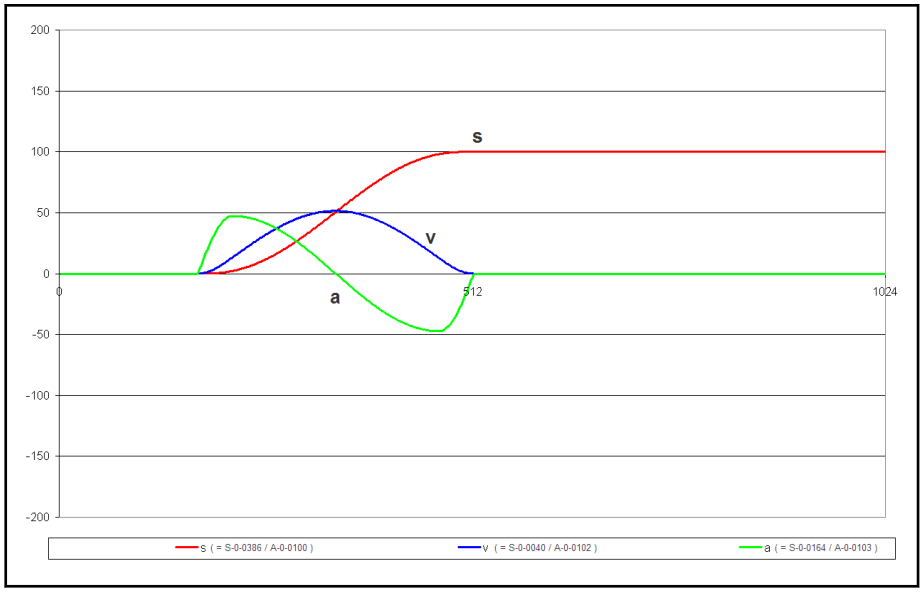
- CurveType: CURVETYPE_MODIFIED_ACCELERATIONTRAPEZE
  - The MotionProfile characterized by the cam is constant up to the jerk (s, v, a, j)



RangeLow = 60; RangeHigh = 180

Fig.11-12: MB_FeedMovementCalcType01, MotionProfile MODIFIED_ACCELERATIONTRAPEZE

- CurveType: CURVETYPE_MODIFIED_SINUSLINE
  - The MotionProfile characterized by the cam is constant up to the jerk (s, v, a, j)



RangeLow = 60; RangeHigh = 180

Fig.11-13: MB_FeedMovementCalcType01, MotionProfile MODIFIED_SINUSLINE

Error Handling

The function block uses the F_RELATED_TABLE, 16#0170x error table. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#00000006	16#00000000	Invalid state of the function block
CALCULATION_ERROR	16#0000000C	16#00000000	Division by Zero
INPUT_RANGE_ERROR	16#00002200	16#00000001	The "NumberOfElements" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002220	16#00000001	The "RangeLow " input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002220	16#00000002	The "RangeHigh " input is lower than the minimum or higher than the maximum
INPUT_INVALID_ERROR	16#00002201	16#00000001	Address of the "CamTable" array is zero
INPUT_INVALID_ERROR	16#00002221	16#00000001	The "RangeLow" input is higher than or equal to "RangeHigh"

Fig.11-14: Error codes of the MB_FeedMovementCalcType01 function block

11.3 MB_FeedMovementCalcType02

Brief Description

The function block calculates a cam table with up to 1,024 data points with a polynomial 7th order in which the inputs of the jerk, acceleration and velocity can be limited in order to ensure optimum processing.

Assignment: Target system/library

IndraMotion MLC 12VRS	RMB_TechCam.compiled-library

Fig.11-15: Reference table of the MB_FeedMovementCalcType02

RMB_TechCam.library

Interface Description

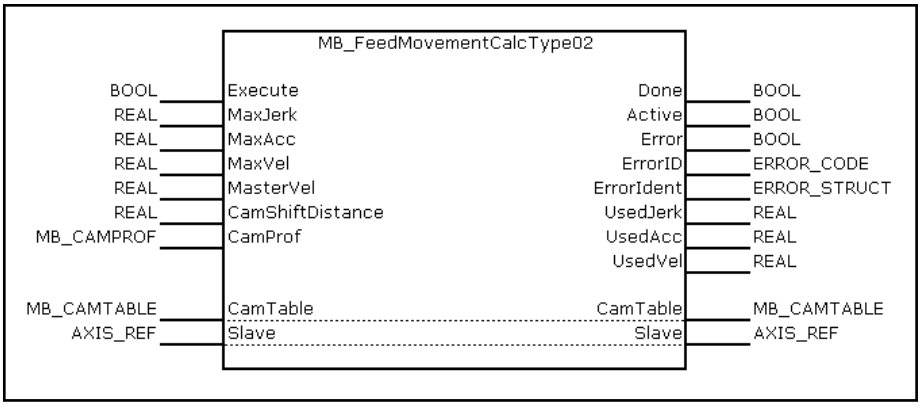


Fig.11-16: MB_FeedMovementCalcType02 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	CamTable	MB_CAMTABLE	Structure with the calculated cam table elements and information on the number of elements
	Slave	AXIS_REF	Reference to the slave axis
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	MaxJerk	REAL	Maximum jerk
	MaxAcc	REAL	Maximum acceleration
	MaxVel	REAL	Maximum velocity
	MasterVel	REAL	Velocity of the master axis
	CamShaftDistance	REAL	Cam length
	CamProf	MB_CAMPROF	Cam profile selection
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid, error outputs valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, it contains a detailed error description (see "Error Handling" on page 539)
	UsedJerk	REAL	Jerk used related to the input "MaxJerk" in [%]
	UsedAcc	REAL	Acceleration used related to the input "MaxAcc" in [%]
	UsedVel	REAL	Velocity used related to the input "MaxVel" in [%]

Fig.11-17: Interface variables of the MB_FeedMovementCalcType02 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
MaxJerk	REAL	>0.0	n.def.	20.0	Rising edge at "Execute"
MaxAcc	REAL	>0.0	n.def.	20.0	Rising edge at "Execute"
MaxVel	REAL	>0.0	n.def.	20.0	Rising edge at "Execute"
MasterVel	REAL	>0.0	n.def.	10.0	Rising edge at "Execute"
CamShaftDis-tance	REAL	>0.0	n.def.	360.0	Rising edge at "Execute"
CamProf	MB_CAMPROF			CAM_PROF_STANDARD	Rising edge at "Execute"

Fig. 11-18: Min./max. and default values of the MB_FeedMovementCalcType02 function block

The enumeration chapter 5.14.3 "MB_CAMPROF" on page 156 selects between different cam profiles.

When selecting the cam profile "CAMPROF_STANDARD", the linear path has to be disabled since it is a profile without gear reduction. The linear path for the profiles "CAMPROF_ADDITIVE" and "CAMPROF_ADDITIVE_SEGMENTABLE" has to be ticked and the reduction has to be set to "1", since they are profiles with gear reduction.

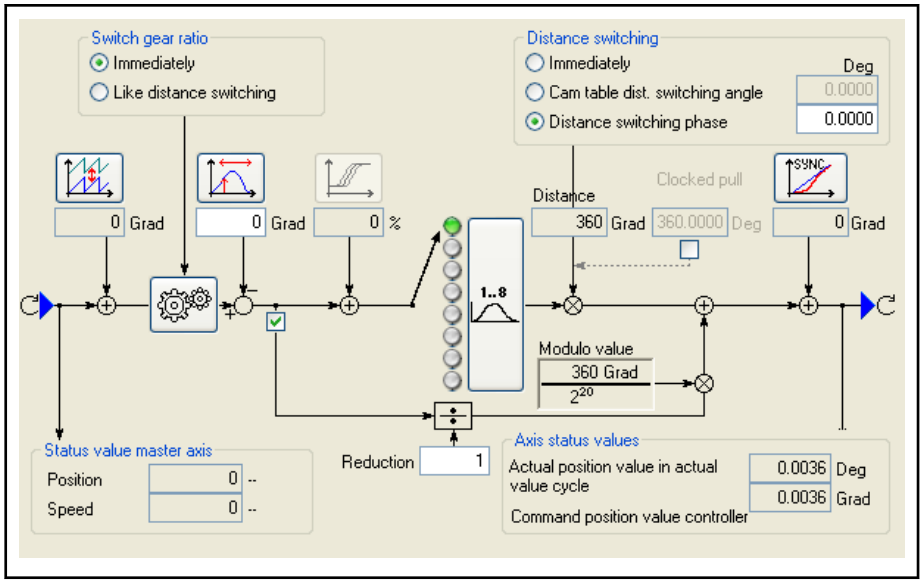


Fig. 11-19: Enabling the linear path

The structure chapter 5.14.4 "MB_CAMTABLE" on page 157 is provided with the "CamTable" array containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

RMB_TechCam.library

Functional Description

The MB_FeedMovementCalcType02 function block calculates a cam table with up to 1,024 data points. At this cam table, jerk, acceleration and velocity can be limited. The number of data point elements of a cam table is set by the "NumberOfElements" element of the "CamTable" structure.

The user sets the master axis velocity, cam distance, maximum velocity, acceleration and jerk. The function block uses these specified values to calculate a MotionProfile as cam indicating a rest-in-rest motion. The jerk profile either consists of individual parabola parts or calculated as polynomial 7th order. The MotionProfile is calculated from this jerk profile. The MotionProfile allows an endless motion of the slave axis.

Note:

The variant of the function block for drive cams gets more information from the parameters of the respective drive required for the calculation of the function block. That information includes for example the maximum jerk of the drive or the modulo value which means that parameters are read from the drive while the function block is active. If the specified values are possible, a cam is calculated or an error is output stating that the motion with this specified values is not possible.

The modulo value of the drive has to correspond to the value of the input variable "CamShaftDistance". Otherwise, the function block is completed with error. The parameter "CamShaftDistance" of the command "MC_CamIn" has thus to be set as high as the modulo value of the slave axis. Otherwise, an endless motion is not possible.

The cam table to be calculated is stored in the "CamTable" structure with scaling in per cent. The validity of the cam elements is signaled by the "Done" output. The cam table can be loaded to the control or drive using the chapter 5.14.2 "MB_LoadCamData" on page 154 function block.

If an error occurs while processing the function block, this is signaled by the "Error" output. The elements of the "CamTable" structure as well as the outputs "UsedJerk" and "UsedAcc" and "UsedVel" are not updated in case of error.

Cam characteristics

The cam table of the MB_FeedMovementCalcType02 provides the following characteristics:

- The MotionProfile characterized by the cam table is constant up to the jerk (s, v, a, j)
- The algorithm to calculate the cam data points is distributed across several cycles, i.e. calls of the instance to avoid a considerable effect on the PLC cycle time

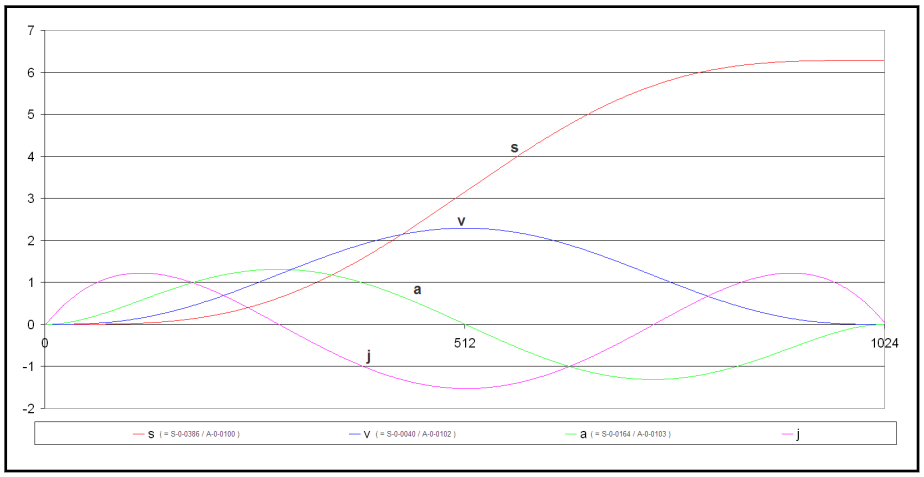
Overview on possible calculation procedures

Switching between the individual variants is executed by the function block. Preferably, a cam with polynomial 7th order is calculated. If this is not possible due to the set limit values, it is calculated using parabola parts instead. If this option does also not allow to maintain the limit values, the "Error" output of the function block is set.

Note:

When calculating using parabola parts, inaccuracies might occur if the ratio of jerk and acceleration is most unfavorable (high jerk at low acceleration) since the parabola becomes very high for the jerk and very narrow at the same time. Due to the limited accuracy, inaccuracy can result.

Variant 1: Polynomial 7th order

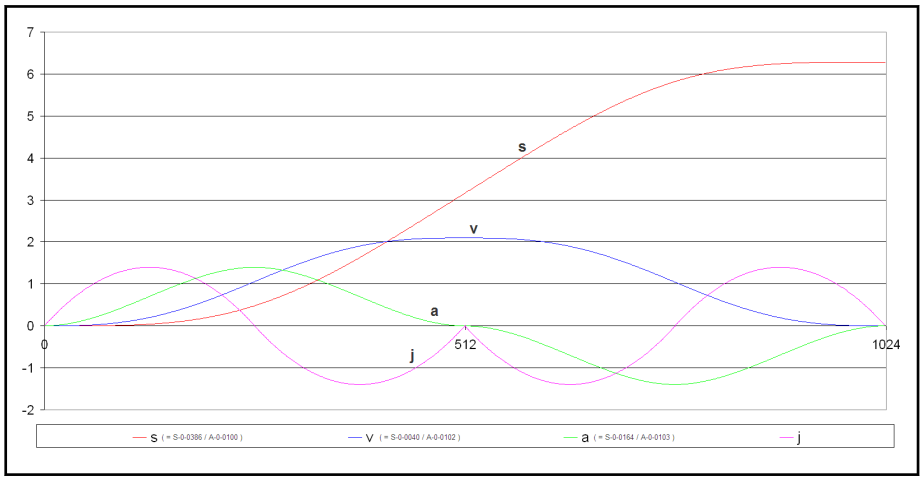


MaxJerk = 10 rad/s^3; MaxAcc = 10 rad/s^2; MaxVel = 10 rad/s; MasterVel= 10 rpm; modulo value = 2*Pi rad

Calculation as polynomial 7th order

Fig.11-20: Cam profile of the MB_FeedMovementCalcType02 [slave = f(master)]

Variant 2: Velocity and acceleration limit values are not reached (variant is rarely used in practice)

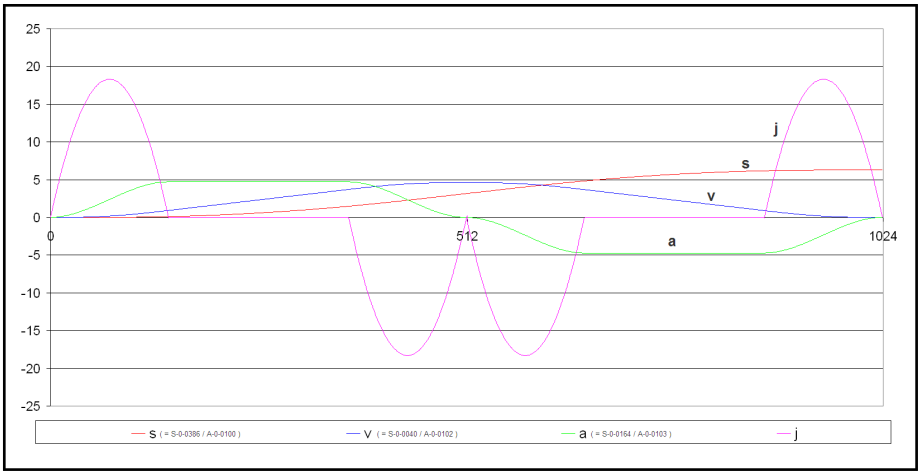


MaxJerk = 10 rad/s^3; MaxAcc = 10 rad/s^2; MaxVel = 10 rad/s; MasterVel= 10 rpm; modulo value = 2*Pi rad

Calculation from parabola parts

Fig.11-21: Cam profile of the MB_FeedMovementCalcType02 [slave = f(master)]

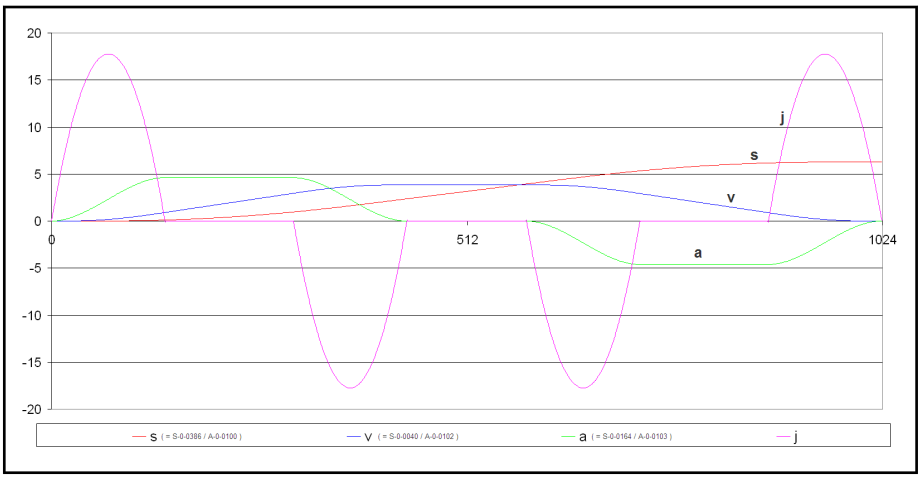
Variant 3: Velocity limit value is not reached, acceleration limit value is reached (variant is rarely used in practice)



MaxJerk = 20 rad/s^3; MaxAcc = 5 rad/s^2; MaxVel = 5 rad/s; MasterVel= 22 rpm; modulo value = 2*Pi rad
Calculation from parabola parts

Fig.11-22: Cam profile of the MB_FeedMovementCalcType02 [slave = f(master)]

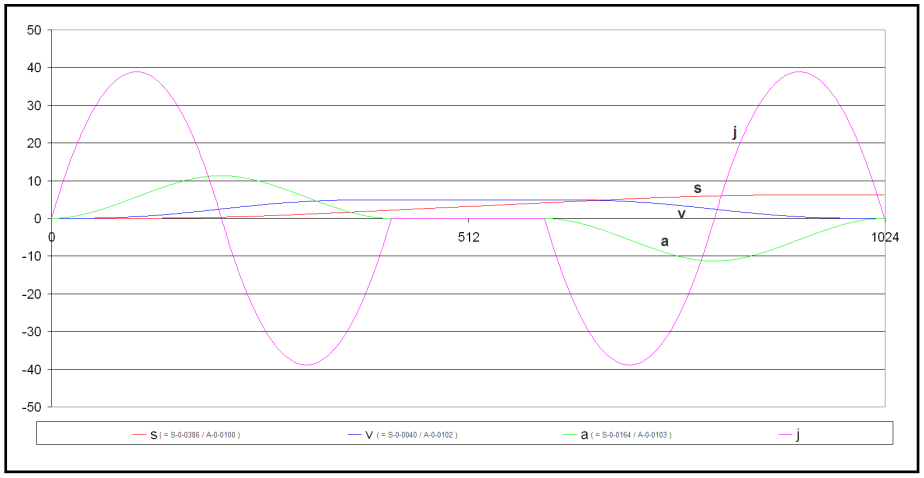
Variant 4: Velocity and acceleration limit values are reached (variant is rarely used in practice)



MaxJerk = 20 rad/s^3; MaxAcc = 5 rad/s^2; MaxVel = 4 rad/s; MasterVel= 21 rpm; modulo value = 2*Pi rad
Calculation from parabola parts

Fig.11-23: Cam profile of the MB_FeedMovementCalcType02 [slave = f(master)]

Variant 5: Velocity limit value is reached, acceleration limit value is not reached (variant is rarely used in practice)



MaxJerk = 40 rad/s^3; MaxAcc = 20 rad/s^2; MaxVel = 5 rad/s; MasterVel= 28 rpm; modulo value = 2*Pi rad
Calculation from parabola parts

Fig.11-24: Cam profile of the MB_FeedMovementCalcType02 [slave = f(master)]

Error Handling

The function block uses the error table F_RELATED_TABLE, 16#0170. It can generate the following error messages in Additional1/Additional2.

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#00000006	16#00000000	Invalid state of the function block
INPUT_RANGE_ERROR	16#00002200	16#00000001	The "NumberOfElements" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002230	16#00000001	The "MaxJerk" input is lower than the minimum value.
INPUT_RANGE_ERROR	16#00002230	16#00000002	The "MaxAcc" input is lower than the minimum value
INPUT_RANGE_ERROR	16#00002230	16#00000003	The "MaxVel" input is lower than the minimum value.
INPUT_RANGE_ERROR	16#00002230	16#00000004	The "MasterVel" input is lower than the minimum value
INPUT_RANGE_ERROR	16#00002230	16#00000005	The input "CamShaftDistance" is lower than the minimum value
INPUT_INVALID_ERROR	16#00002201	16#00000001	Address of the "CamTable" array is zero
INPUT_INVALID_ERROR	16#00002231	16#00000001	CamShaftDistance unequal to the modulo value of the axis reference
INPUT_INVALID_ERROR	16#00002231	16#00000002	Input values for jerk, acceleration or velocity are higher than preset in the axis parameters
CALCULATION_ERROR	16#00002232	16#00000001	Velocity of the master axis is too high or the maximum jerk, the maximum acceleration or the maximum velocity is too low.

Fig.11-25: Error codes of the MB_FeedMovementCalcType02 function block

RMB_TechCam.library

11.4 MB_CoordinatePairCamType01

Brief Description

The function block calculates a cam table with up to 1,024 data point elements that are determined depending on the selected coordinate pair type.

Assignment: Target system/library

IndraMotion MLC 12VRS	RMB_TechCam.compiled-library

Fig.11-26: Reference table of the MB_CoordinatePairCamType01

Interface Description

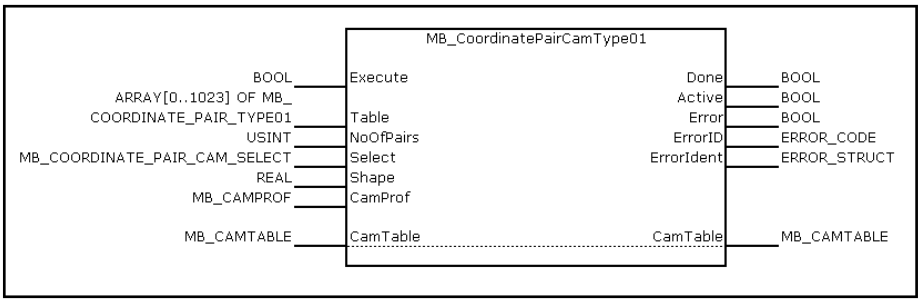


Fig.11-27: MB_CoordinatePairCamType01 function block

I/O type	Name	Type	Comment
VAR_IN_OUT	CamTable	MB_CAMTABLE	Structure with the calculated cam table elements and information on the number of elements
VAR_INPUT	Execute	BOOL	Processing of the function block enabled (once, edge-controlled)
	Table	ARRAY[0..1023] OF MB_COORDINATE_PAIR_TYPE01	List of coordinate pairs
	NoOfPairs	USINT	Number of valid coordinate pairs
	Select	MB_COORDINATE_PAIR_CAM_SELECT	Selection of the coordinate pair types
	Shape	REAL	Factor for jerk limitation
	CamProf	MB_CAMPROF	Cam profile selection
VAR_OUTPUT	Done	BOOL	Processing completed without error, output data valid
	Active	BOOL	Processing not yet completed, output data invalid
	Error	BOOL	Processing completed with error, output data invalid, error outputs valid
	ErrorID	ERROR_CODE	If the "Error" output is set, it contains a broad error classification
	ErrorIdent	ERROR_STRUCT	If the "Error" output is set, it contains a detailed error description (see "Error Handling" on page 547)

Fig.11-28: Interface variables of the MB_CoordinatePairCamType01 function block

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Execute	BOOL			FALSE	Continuous
NoOfPairs	USINT	1 or 2	1023	0	Rising edge at "Execute"
Select	MB_COORDI-NATE_PAIR_CAM_SELECT			COORDI-NATE_PAIR_POS_VEL_TYPE01	Rising edge at "Execute"
Shape	REAL	0.0	1.0	0.0	Rising edge at "Execute"
CamProf	MB_CAMPROF			CAM_PROF_STANDARD	Rising edge at "Execute"

Fig.11-29: Min./max. and default values of the MB_CoordinatePairCamType01 function block

Name	Type	Min. value	Max. value	Default value	Effective	Description
X	LREAL	0.0	<360.0 or 360.0	0.0	Rising edge at "Execute"	X-coordinate of the coordinate pair (position)
Y	LREAL	n.def.	n.def.	0.0	Rising edge at "Execute"	Y-coordinate of the coordinate pair (position, velocity or acceleration)

Fig.11-30: Min./max. and default values of the MB_COORDI-NATE_PAIR_TYPE01 structure

The enumeration "MB_COORDINATE_PAIR_CAM_SELECT" is used to select between the different types (position, velocity and acceleration) of the coordinate pairs.

Element	Value	Description
COORDINATE_PAIR_POS_VEL_TYPE01	0	Coordinate pairs from position (X) and velocity (Y)
COORDINATE_PAIR_POS_VEL_TYPE02	1	Coordinate pairs from position (X) and velocity (Y) considering the "Shape" input
COORDINATE_PAIR_POS_ACC	2	Coordinate pairs from position (X) and acceleration (Y) considering the "Shape" input
COORDINATE_PAIR_POS_POS_TYPE01	3	Coordinate pairs from position (X) and position (Y)
COORDINATE_PAIR_POS_POS_TYPE02	4	Coordinate pairs from position (X) and position (Y) considering the "Shape" input

Fig.11-31: Elements of the enumeration type MB_COORDI-NATE_PAIR_CAM_SELECT

The enumeration chapter 5.14.3 "MB_CAMPROF" on page 156 selects between different cam profiles.

When selecting the cam profile "CAMPROF_ADDITIVE_SEGMENTABLE", the following is to be considered:

- For cam tables for motions with unlimited (infinite) travel range whose first and last element are identical, the linear path has to be ticked and the reduction has to be set to "1" , since it is a profile with gear reduction. For all other cam tables (|last_Element - first_Element| = 100%), the linear path has to be disabled, since they are profiles without gear reduction.
- For cam tables for motions with limited (finite) travel range, the linear path has to be disabled.

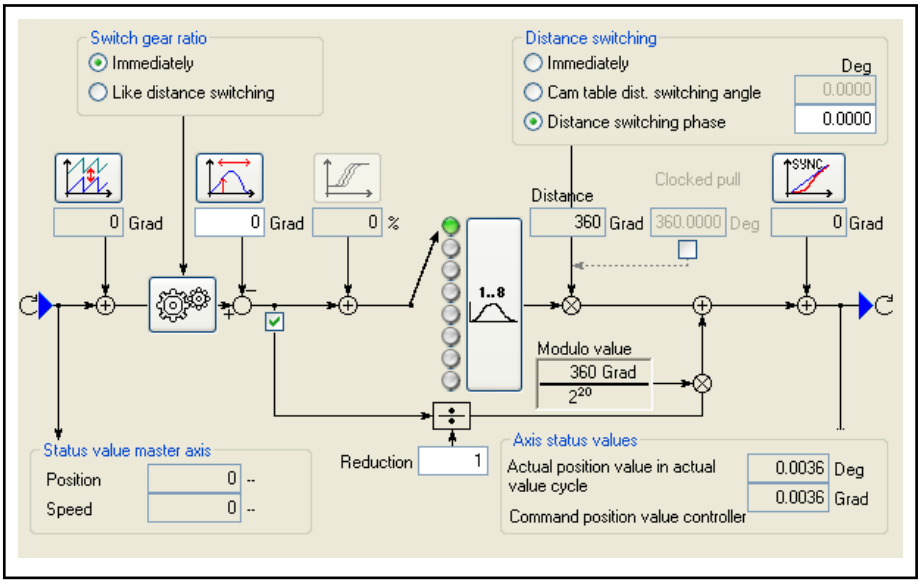


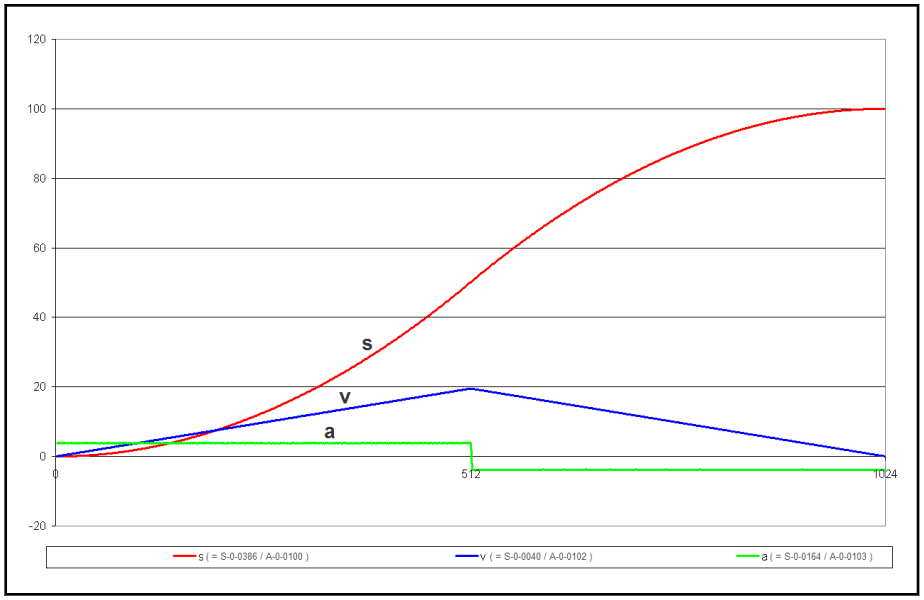
Fig.11-32: Enabling the linear path

The structure chapter 5.14.4 "MB_CAMTABLE" on page 157 is provided with the "CamTable" array containing the calculated cam points and the variable "NumberOfElements" which shows the number of valid cam points of the "CamTable" array.

Functional Description

With the specification of the coordinate pairs, the ML_CoordinateCamPairType01 function block carries out a one-time calculation of a cam table with up to 1,024 data point elements after processing enabled via "Execute" based on the input values. The number of data point elements of a cam table is set by the "NumberOfElements" element of the "CamTable" structure.

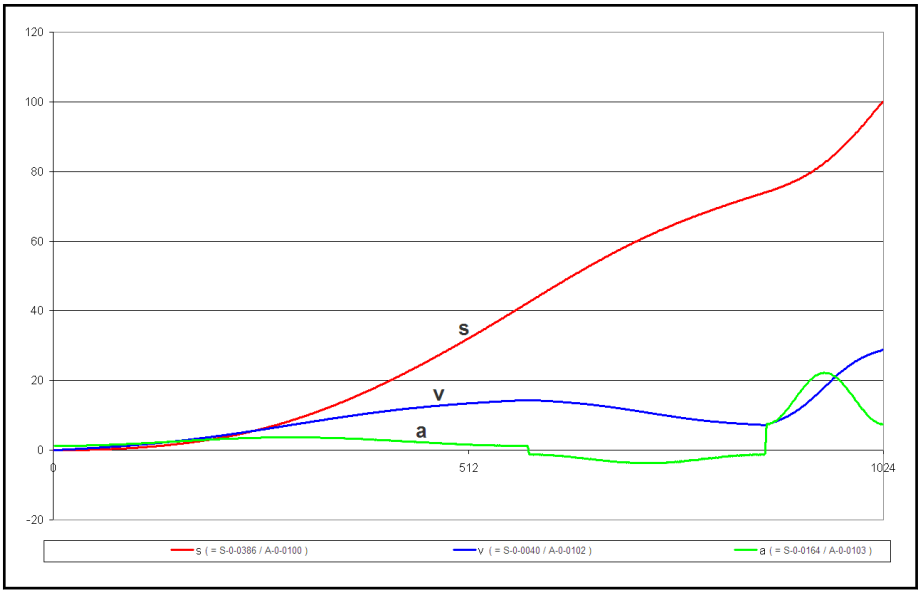
- Select: COORDINATE_PAIR_POS_VEL_TYPE01
  - If the data point (0;0) is no specified coordinate pair, it is automatically considered when calculating the cam table.



Coordinate pairs: (180;100)

Fig.11-33: MB_CoordinatePairCamType01, MotionProfile COORDINATE_PAIR_POS_VEL_TYPE01

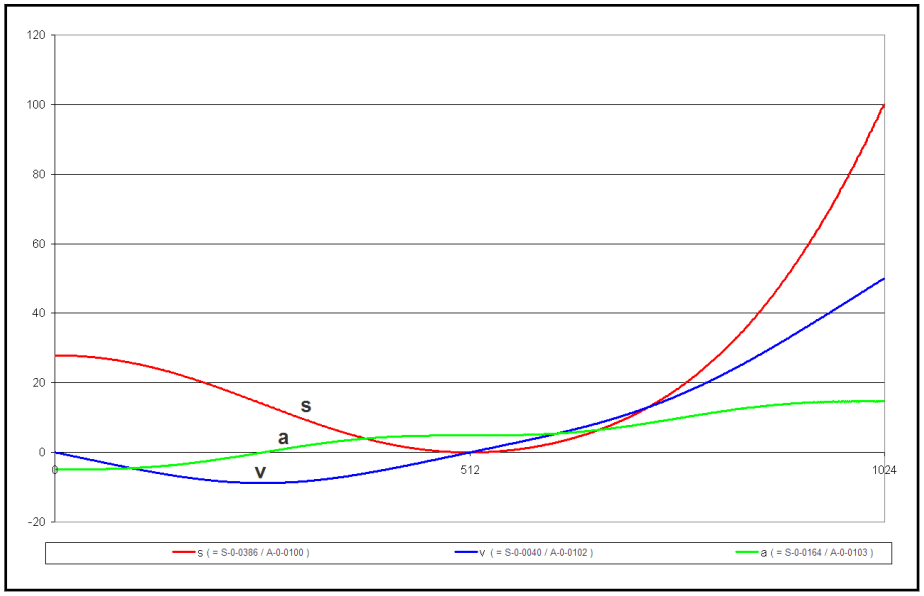
- Select: COORDINATE_PAIR_POS_VEL_TYPE02
  - To calculate the cam table, at least two coordinate pairs have to be given.
  - Due to the "Shape" factor provided for the jerk limitation, the velocity is rounded.
  - The cam table calculated can only be used as an endless cam if the first data point is (0;0) and the last data point is (360;0). If this is not the case, there are position irregularities at the cam transitions.
  - If the above mentioned data points (0; 0 and 360; 0) are inserted and the "Shape" factor is set to 0%, the created cam is identical to the cam created due to the selection of COORDINATE_PAIR_POS_VEL_TYPE01.



Coordinate pairs: (0;0),(200;200),(300;100),(350;400);
Shape = 0.5

Fig.11-34: MB_CoordinatePairCamType01, MotionProfile COORDINATE_PAIR_POS_VEL_TYPE02

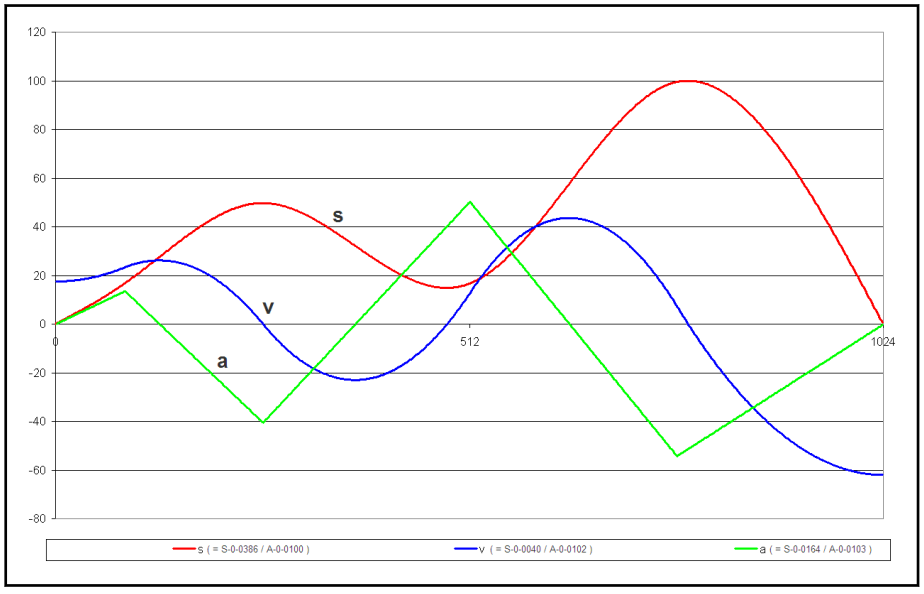
- Select: COORDINATE_PAIR_POS_ACC
  - This selection has to be used when the desired cam profile is specified by individual data points to be traveled with a "rest-in-rest" motion. The target points are connected with a polynomial 5th order. The consequence is that every target point from the input data is approached with an acceleration/deceleration motion, i.e. the velocity is = 0 at every target point.
  - To calculate the cam table, at least two coordinate pairs have to be given.
  - Due to the "Shape" factor provided for the jerk limitation, the velocity is rounded.



Coordinate pairs: (180;100),(360;300); Shape = 1

Fig.11-35: MB_CoordinatePairCamType01, MotionProfile COORDINATE_PAIR_POS_ACC

- Select: COORDINATE_PAIR_POS_POS_TYPE01
  - To calculate the cam table, at least two coordinate pairs have to be given.
  - It is traveled through the individual value pairs without rest. The target positions are connected with a spline polynomial 3rd order.



Coordinate pairs: (0;0),(30;10),(90;30),(180;10),(270;60),(360;0)

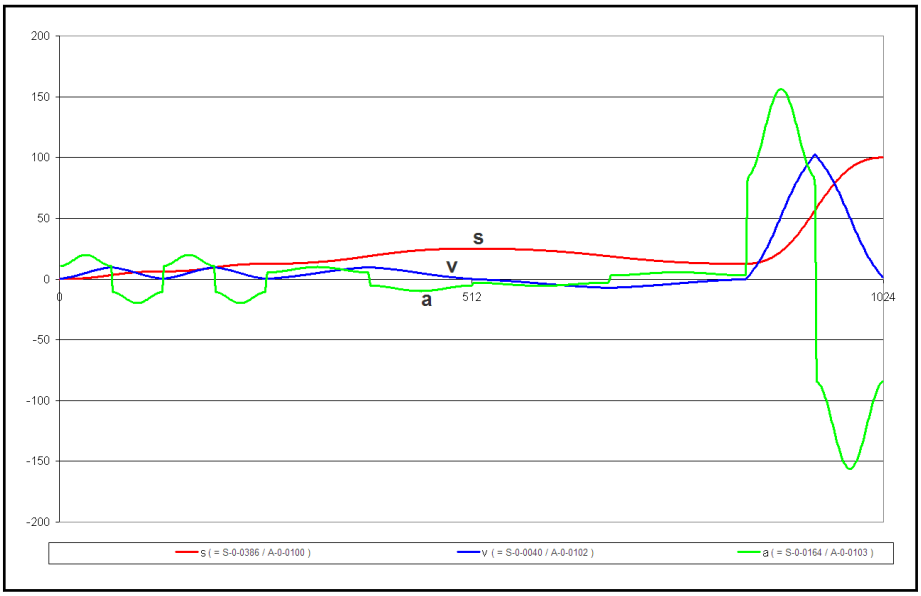
Fig.11-36: MB_CoordinatePairCamType01, MotionProfile COORDINATE_PAIR_POS_POS_TYPE01

- Select: COORDINATE_PAIR_POS_POS_TYPE02
  - This selection has to be used when the desired cam profile is specified by individual data points to be traveled with a "rest-in-rest" motion. The target points are connected with a polynomial 5th

order. The consequence is that every target point from the input data is approached with an acceleration/deceleration motion, i.e. the velocity is = 0 at every target point.

– To calculate the cam table, at least two coordinate pairs have to be given.

– Due to the "Shape" factor provided for the jerk limitation, the velocity is rounded.



Coordinate pairs: (0;0),(45;50),(90;100),(180;200), (300;100),(360;800); Shape = 0.3

Fig.11-37: MB_CoordinatePairCamType01, MotionProfile COORDINATE_PAIR_POS_POS_TYPE02

The cam table to be calculated is stored in the "CamTable" structure with scaling in per cent. The validity of the cam elements is signaled by the "Done" output. The cam table can be loaded to the control or drive using the chapter 5.14.2 "MB_LoadCamData" on page 154 function block.

If an error occurs while processing the function block, this is signaled by the "Error" output. The elements of the "CamTable" structure are not updated in case of error.

Cam characteristics

The cam table of the ML_CoordinatePairCamType01 provides the following characteristics:

- The algorithm to calculate the cam data points is divided into several cycles, i.e. calls the instance to avoid a considerable effect on the PLC cycle time.
- Select: COORDINATE_PAIR_POS_VEL_TYPE01
  – The MotionProfile characterized by the cam table is constant up to the velocity (s, v) and shows a jump in the acceleration (a)
- Select: COORDINATE_PAIR_POS_VEL_TYPE02
  – The MotionProfile characterized by the cam table is constant up to the acceleration (s, v, a) depending on the "Shape" factor and shows a jump in the jerk (j)
- Select: COORDINATE_PAIR_POS_ACC

  - The MotionProfile characterized by the cam table is constant up to the acceleration (s, v, a) depending on the "Shape" factor and shows a jump in the jerk (j)
- Select: COORDINATE_PAIR_POS_POS_TYPE01
  - The MotionProfile characterized by the cam table is constant up to the acceleration (s, v, a) and shows a jump in the jerk (j)
- Select: COORDINATE_PAIR_POS_POS_TYPE02
  - The MotionProfile characterized by the cam table is constant up to the acceleration (s, v, a) depending on the "Shape" factor and shows a jump in the jerk (j)

Error Handling

The function block uses the F_RELATED_TABLE, 16#0170x error table. It can generate the following error messages in Additional1/Additional2:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR	16#00000006	16#00000000	Invalid state of the function block
CALCULATION_ERROR	16#0000000C	16#00000000	Division by Zero
INPUT_RANGE_ERROR	16#00002200	16#00000001	The "NumberOfElements" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002270	16#00000001	The "NoOfPairs" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002270	16#00000002	The "Shape" input is lower than the minimum or higher than the maximum
INPUT_RANGE_ERROR	16#00002270	16#00000003	The input element of "table" input is lower than the minimum or higher than the maximum
INPUT_INVALID_ERROR	16#00002201	16#00000001	Address of the "CamTable" array is zero
INPUT_INVALID_ERROR	16#00002271	16#00000001	Input element of "Table" is lower than the previous input element

Fig.11-38: Error codes of the MB_CoordinatePairCamType01 function block

12 RIL_ParameterChannel.library

12.1 Introduction and Overview

The RIL_ParameterChannel with the IL_ParameterChannel function block allows an acyclic communication via the cyclic channel of a field bus connection.

12.2 IL_ParameterChannel

Brief Description

The block allows an acyclic communication via the cyclic data channel. The data exchange is carried out via a function block on the control as well as via a corresponding function block in the field bus master.

Assignment: Target system/library

Target system	Library
IndraMotion MLC/IndraLogic XLC 12VRS	RIL_ParameterChannel.compiled-library

Fig. 12-1: Reference table of the IL_ParameterChannel

Interface Description

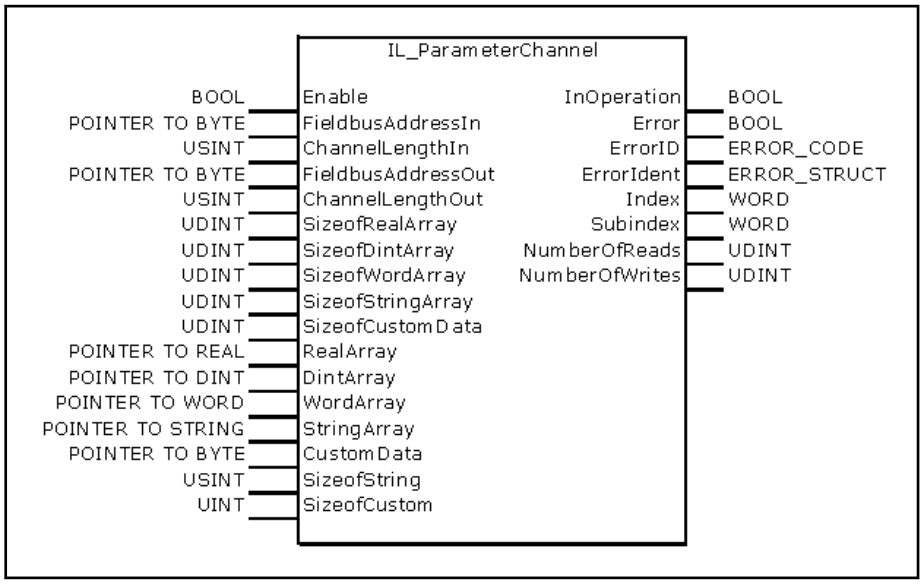


Fig. 12-2: Function block IL_ParameterChannel

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Processing of the function block enabled (once, edge-controlled)
	FieldbusAddressIn	POINTER TO BYTE	Address of the input range for the cyclic data traffic
	ChannelLengthIn	USINT	Length of the cyclic inputs for the parameter channel in bytes
	FieldbusAddressOut	POINTER TO BYTE	Address of the output range for the cyclic data traffic
	ChannelLengthOut	USINT	Length of the cyclic outputs for the parameter channel in bytes
	SizeofRealArray	UDINT	Number of REAL data accessible via the RealArray address in bytes
	SizeofDintArray	UDINT	Number of DINT data accessible via the DintArray address in bytes

RIL_ParameterChannel.library

I/O type	Name	Data type	Description
	SizeofWordArray	UDINT	Number of WORD data accessible via the WordArray address in bytes
	SizeofStringArray	UDINT	Number of STRING data accessible via the StringArray address in bytes
	SizeofCustomData	UDINT	Number of the complete Custom data accessible via the CustomData address in bytes
	RealArray	POINTER TO REAL	A pointer to an array of REAL values. The array can be provided with 4,845 elements
	DintArray	POINTER TO DINT	A pointer to an array of DINT values. The array can be provided with 4,845 elements
	WordArray	POINTER TO WORD	A pointer to an array of WORD values. The array can have 2,295 elements
	StringArray	POINTER TO STRING	A pointer to an array of STRING values. The array can have 2,040 elements
	CustomData	POINTER TO BYTE	A pointer to an array of BYTE values. The array can have data of up to 255 * 8Kb
	SizeofString	USINT	STRING size in the StringArray. Caution: This is the size given in the definition of the string (without end ID)
	SizeofCustom	UINT	Size of a data array (1 – 8,192 bytes)
VAR_OUTPUT	InOperation	BOOL	The function block operates correctly. The output data is valid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Describing diagnostics in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics
	Index	WORD	Last successfully used index
	Subindex	WORD	Last successfully used subindex
	NumberOfReads	UDINT	Number of successful read accesses by the master. The value can exceed (from 65535 to 0)
	NumberOfWrites	UDINT	Number of successful write accesses by the master. The value can exceed (from 65535 to 0)

Fig.12-3: Interface variables of the IL_ParameterChannel function block

Min./max. and default values and effective of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
FieldbusAddressIn	POINTER TO BYTE	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"
ChannelLengthIn	USINT	6	18	12	Rising edge at "Enable"
FieldbusAddressOut	POINTER TO BYTE	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"

Name	Type	Min. value	Max. value	Default value	Effective
ChannelLength-Out	USINT	6	18	12	Rising edge at "Enable"
SizeofRealArray	UDINT	0	19380(4845 * 4)	0	Rising edge at "Enable"
SizeofDintArray	UDINT	0	19380(4845 * 4)	0	Rising edge at "Enable"
SizeofWordArray	UDINT	0	4590(2295 * 2)	0	Rising edge at "Enable"
SizeofStringArray	UDINT	0	522240(2040 * 256)	0	Rising edge at "Enable"
SizeofCustomDa-ta	UDINT	0	2088960(255 * 8192)	0	Rising edge at "Enable"
RealArray	POINTER TO RE-AL	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"
DintArray	POINTER TO DINT	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"
WordArray	POINTER TO WORD	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"
StringArray	POINTER TO STRING	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"
CustomData	POINTER TO BYTE	0x00000000	0xFFFFFFFF	0x00000000	Rising edge at "Enable"
SizeofString	USINT	0	255	80	Rising edge at "Enable"
SizeofCustom	UINT	0	8192	128	Rising edge at "Enable"

Fig. 12-4: Min./max. and default values of the IL_ParameterChannel function block

Functional Description

The function blocks allows an acyclic communication via a cyclic channel. Therefore, the range from the cyclic data channel is used for communication. The inputs "FieldbusAddressIn" and "ChannelLengthIn" are used to transfer data from the master to the slave. The inputs "FieldbusAddressOut" and "ChannelLengthOut" are used for the direction from the slave to the master. The terms "out" and "in" refer to the direction seen from the slave. The data ranges can be at any position within the cyclic data and can also differ in length. The minimum length of the parameter channel is 6 bytes and the maximum length in both directions is 18 bytes.

PLC data can only be exchanged via data arrays. The different data types have to be considered. There are 5 different data types:

- REAL (up to 4845 data arrays)
- DINT (up to 4845 data arrays)
- WORD (up to 2295 data arrays)
- STRING (up to 2040 data arrays)
- Custom (up to 255 data arrays)

All data areas are assigned as pointer at the function block. The size available always has to be specified respectively. If either the data size or the assigned pointer is 0, the function block interprets it as a non-existing data range and outputs an error to these data requests. The data size must be in bytes, e.g. the SIZEOF() operator can be used.

For strings, the string size has to be specified at the "SizeofString" input apart from the array size. This is required since a string can be defined in different sizes. The size of the string must be uniform within the array and may consist of 255 characters max.

☞ The size of the string is only the size given when defining the string. The IndraLogic attaches one more byte for the end ID. For the definition

```
strExample: STRING(20);
```

21 bytes are assigned in the PLC since 1 end ID has to be added after the 20 characters. The IL_ParameterChannel function block considers this automatically when checking the sizes. Thus, "SIZEOF(strExample)" had to be specified as value for "SizeofString" in the previous example.

The CUSTOM data can be used to consistently transfer a higher number of data. The CUSTOM data can be imagined as data structure that should be completely transferred. Up to 255 of these structures can exist. To use this functionality, a pointer has to be assigned ("CustomData"). The size of the individual structure has also to be known. This is executed at the "SizeofCustom" input. Finally, the complete size is required ("SizeofCustomData").

Since there is no information on the content of the data in the function block, the data is copied byte by byte. The respective application has to ensure that the data format is consistent on slave and master.

☞ In case of the CUSTOM data, a data set has to be copied completely. Thus, the time required for copying is significantly important at runtime of the function block. Measurements have shown the following results regarding the maximum runtime for copying and editing the protocol:

- 256 bytes ca. 12us
- 4 Kbytes ca. 20us and
- 8 Kbytes ca. 40us required on an IndraControl L65

This has to be considered for the task configuration.

☞ If functions are used to check the validity of pointers and array limits (e.g. RIL_CheckRtv.library), the runtime of the function block drastically increases. For CUSTOM data of 4 Kb, the runtime can increase from 20us up to 30ms!

The outputs "NumberOfReads" and "NumberOfWrites" are used to identify an action in the slave by the master and to react respectively. That way, the respective values of "NumberOfReads" and "NumberOfWrites" can be saved on the slave in the PLC. Then, an action can be executed in case of a change. This allows to process the value just written by the master after a change of "NumberOfWrites" via the specified "index" and "subindex" for example.

The outputs are only updated if reading or writing was successful. If an error occurred, the counters are not increased. The counters are of type UDINT. The value range is from 0 to 65535 only. If the counter is at the maximum value of 65535 and it has to be incremented again, the counter starts again at 0.

Boundary conditions and restrictions:

- The function block may only be instantiated once
- Only the parameter set 0 can be read for SERCOS parameters
- SERCOS parameters can only be read as 16 bit "Ident Number" (IDN). "Extended Ident Number" (EIDN) cannot be read
- In the SERCOS parameters, only the SERCOS element 7 (data) can be read

The data can be queried from the master via the index or subindex. The following list specifies the respective values for the index/subindex.

Start index (16 bits)	End index (16 bits)	Subindex (8 bits)	Calculation rule
0x0000	0x1FFF		Reserved
0x2000	0x2FFF	Drive # (1-64)	Index = 0x2000 + S-parameter (only parameter set 0 is possible, no EIDN)
0x3000	0x3FFF	Drive # (1-64)	Index = 0x3000 +P-parameter (only parameter set 0 is possible, no EIDN)
0x4000	0x4FFF	Axis # (1-64)	Index = 0x4000 + A-parameter (only parameter set is 0 possible, no EIDN)
0x5000	0x5AEC		Free
0x5AED	0x5AED	0x01 – 0xFF	Index = 0x5AED; Subindex = Custom data array element; (elements 1 - 255)
0x5AEE	0x5AF7	0x01 – 0xFF	Index = (0x5AEE + [(WORD array element - 1) / 255]; subindex = (WORD array element) – [(index - 0x5AEE) * 255]; (elements 1 - 2295)
0x5AF8	0x5AFF	0x01 – 0xFF	Index = (0x5AF8+ [(STRING array element - 1) / 255]; subindex = (STRING array element) – [(index - 0x5AF8) * 255]; (elements 1 - 2040)
0x5B00	0x5CEB	0x00 – 0xFF	Free
0x5CEC	0x5CFF	0x01 – 0xFF	Index = (0x5CEC + [(REAL array element - 1) / 255]; subindex = (REAL array element) – [(index - 0x5CEC) * 255]; (elements 1 - 4845)
0x5D00	0x5D13	0x01 – 0xFF	Index = (0x5D00 + [(DINT array element - 1) / 255]; subindex = (DINT array element) – [(index - 0x5D00) * 255]; (elements 1 - 4845)
0x5D14	0x5FFF	0x00 – 0xFF	Free
0x6000	0x6FFF	0x00	Index = 0x6000 + C-parameter (only parameter set 0 is possible, no EIDN)
0x7000	0x7FFF	Kinematics # (1-16)	Index = 0x7000 + K-parameter (only parameter set 0 is possible, no EIDN)
0x8000	0x8FFF	Probe # (1-99)	Index = 0x(000 + M parameter (only parameter set 0 is possible, no EIDN)

RIL_ParameterChannel.library

0x9000	0x9FFF	PLS # (1-4)	Index = 0x9000 + N-parameter (only parameter set 0 is possible, no EIDN)
0xA000	0xAFFF	Osci # (1-16)	Index = 0xA000 + O-parameter (only parameter set 0 is possible, no EIDN)
0xB000	0xFFFF		Free

Fig.12-5: Assigned address ranges for index/subindex

Error handling

There are two different error types for the error handling.

- Error when accessing data and error when transferring data
- Error in the configuration of the function block

Errors in the configuration (e.g. parameter channel too short/long) are displayed as errors at the function block itself and have to be analyzed by the PLC program.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000001	"FieldbusAddressIn" is either not assigned or is provided with 0
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000002	"ChannelLengthIn" is either not assigned or is provided with an invalid value (valid values are in the range 6 <= "ChannelLengthIn" <= 18)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000003	"FieldbusAddressOut" is either not assigned or is provided with 0
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000004	"ChannelLengthOut" is either not assigned or provided with an invalid value (valid values are in the range 6 <= "ChannelLengthOut" <= 18)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000005	"SizeofRealArray" is provided with an invalid value (valid values are in the range 0 <= "SizeofRealArray" <= 19380)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000006	"SizeofDintArray" is provided with an invalid value (valid values are in the range 0 <= "SizeofDintArray" <= 19380)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000007	"SizeofWordArray" is provided with an invalid value (valid values are in the range 0 <= "SizeofWordArray" <= 4590)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000008	"SizeofStringArray" is provided with an invalid value (valid values are in the range 0 <= "SizeofStringArray" <= 522240)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#00000009	"SizeofCustomData" exceeds the value allowed (2088960)
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#0000000A	"SizeofRealArray" is provided with an invalid value. "SizeofRealArray" has to be divisible by 4 without remainder

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#0000000B	"SizeofRealArray" is provided with an invalid value. "SizeofDintArray" has to be divisible by 4 without remainder
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#0000000C	"SizeofWordArray" is provided with an invalid value. "SizeofWordArray" has to be divisible by 2 without remainder
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#0000000D	"SizeofStringArray" or "SizeofString" is provided with an invalid value. "SizeofStringArray" has to be divisible by "SizeofString" + 1 without remainder
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#0000000E	"SizeofCustomData" or "SizeofCustom" is provided with an invalid value. "SizeofCustomData" has to be divisible by "SizeofCustom" without remainder
INPUT_RANGE_ERROR (16#0001)	16#00000002	16#0000000F	"SizeofCustom" is provided with an invalid value (valid values are in the range 0 <= "SizeofCustom" <= 8192)
STATE_MACHINE_ERROR (16#0005)	16#00000006	16#0000000x	An invalid state has been detected within the function block. The function block must be restarted

Fig. 12-6: Error codes of the IL_ParameterChannel function block

Errors while accessing data (e.g. read/write non-existent index) and transferring data (e.g. wrong length specification) are not visible at the output of the function block. These are labeled as errors within the parameter channel and error values are specified. The master has to analyze the error values.

ErrorCode	Text
16#8500	The data volume to be transferred is too high for the internal cache
16#8B00	The format specification in the control word is wrong (must be 16#0C)
16#8C00	The given length is longer than the parameter channel itself
16#9500	Data was sent even though a read request was made
16#9600	Bit "C1" is not set in the control word
16#9700	The "last" bit was not sent even though it is a read request
16#1F00	Control-specific error. Further error codes (see error table of the control) follow in the next four words. The error code "Additonal1" is in the words 2 and 3 and the error code "Additional2" is in the words 4 and 5

ErrorCode	Text
16#F200	The given index is unknown
16#F300	The given subindex is unknown

Fig.12-7: Error codes within the parameter channel that can be created by the IL_ParameterChannel function block

12.3 IL_ParameterChannel - Protocol, Error Handling and Commissioning

12.3.1 Description of the Protocol Used

Parameter Channel in the Cyclic Data Channel

The protocol used in the parameter channel only uses a control word to send data from the master to the slave.

The slave uses the status word to be able to send data to the master.

The master can either write data or read data. A data transfer can be only activated by the master. The control word and the status word are always a fixed part of the parameter channel as shown below:

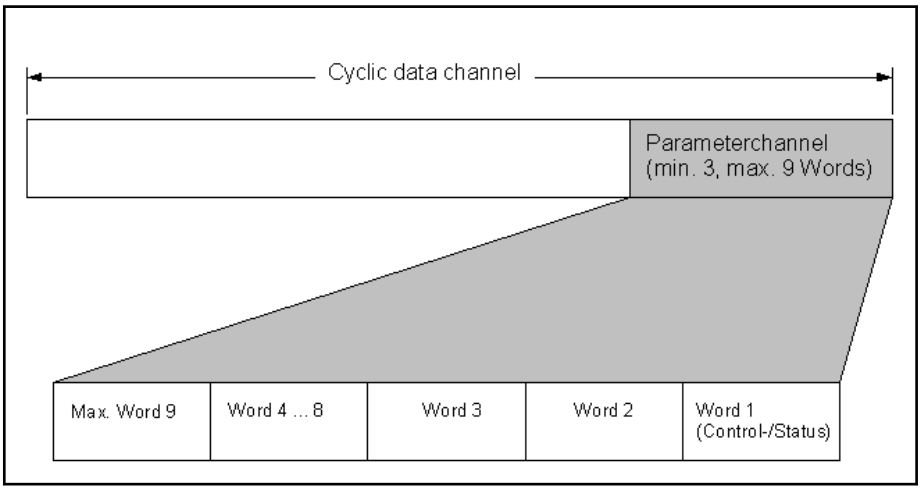


Fig.12-8: Data content of the field bus in the parameter channel

Control/Status Word

The control word is assigned as follows:

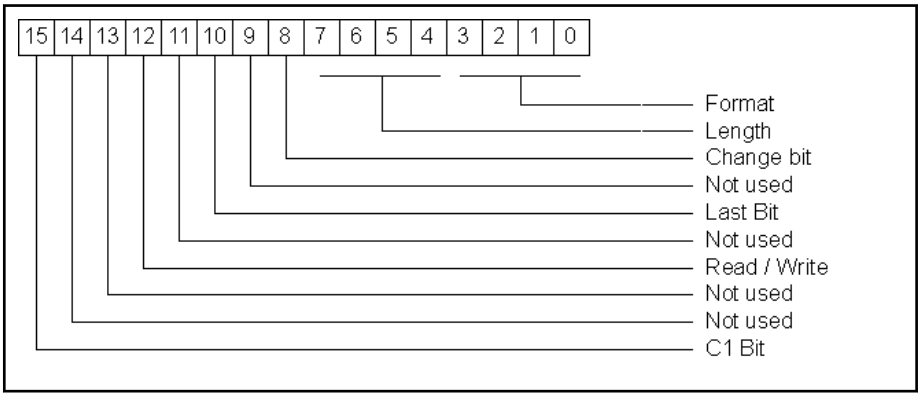


Fig.12-9: Assignment of the control word in the parameter channel

The status word is assigned as follows:

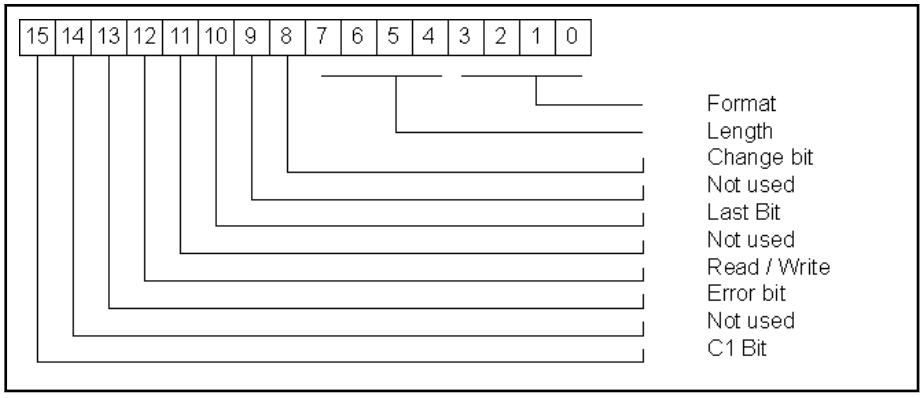


Fig.12-10: Assignment of the status word in the parameter channel

The individual bits have the following meaning:

Format: These bits indicate the purpose and meaning of the following data words in the parameter channel. The value is always 1100b except if the communication is aborted or initialized.

Length: These four bits specify the length of the valid data in bytes excluding the control word. The data in the remaining part of the parameter channel is not defined.

Toggle bit (T): This bit is only then toggled if a new set of data is sent. It creates a handshake between the master and the slave. The master may only toggle this bit if the toggle bit in the status word has the same state as the toggle bit sent in the control word.

Last bit (L): The last bit is set if the last part of the data set is sent. It is used for fragmentation.

Read/write bit (R/W): "Read" corresponds to the bit value 1 which means that the master wants to read data. If it should be written, this bit has to be 0.

C1: Due to the compatibility with the legacy systems, this bit has to be set to 1.

Error bit (E): This bit signalizes when an error occurred in the slave. The error cause (error code) is specified with the following data.

☞ Non-used bits are provided with a value of 0.

Index and Subindex

For the first request by the master, one word is provided for the index and one word for the subindex after the control word. Additionally, all available data can be addressed. The assignment of index/subindex to the control parameters is as follows:

RIL_ParameterChannel.library

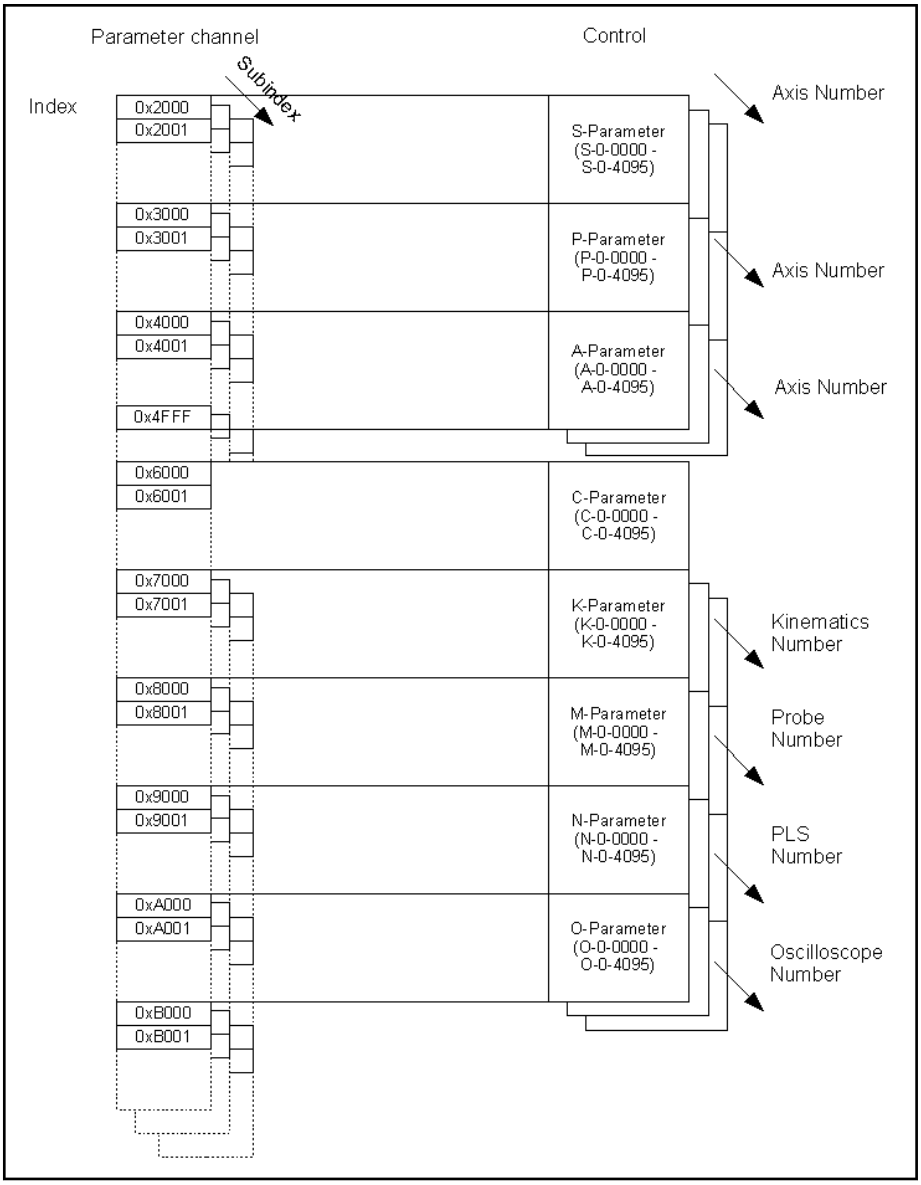


Fig.12-11: Assignment of index/subindex to the parameters

The assignment is very simple. To get from a parameter to the desired index/ subindex, only the parameter number has to be added to the start offset of the index. The subindex results from the respective number of the axis, the kinematics etc.

Example:

The axis parameter A-0-0100 of the axis 22 should be read.

In this case, 0x4000 + 100 = 0x4064 has to be selected for the index and the axis number 22 for the subindex.

The assignment of index/subindex to the PLC variables is shown below. It is important that the subindex 0 is not used. Thus, it is not shown below.

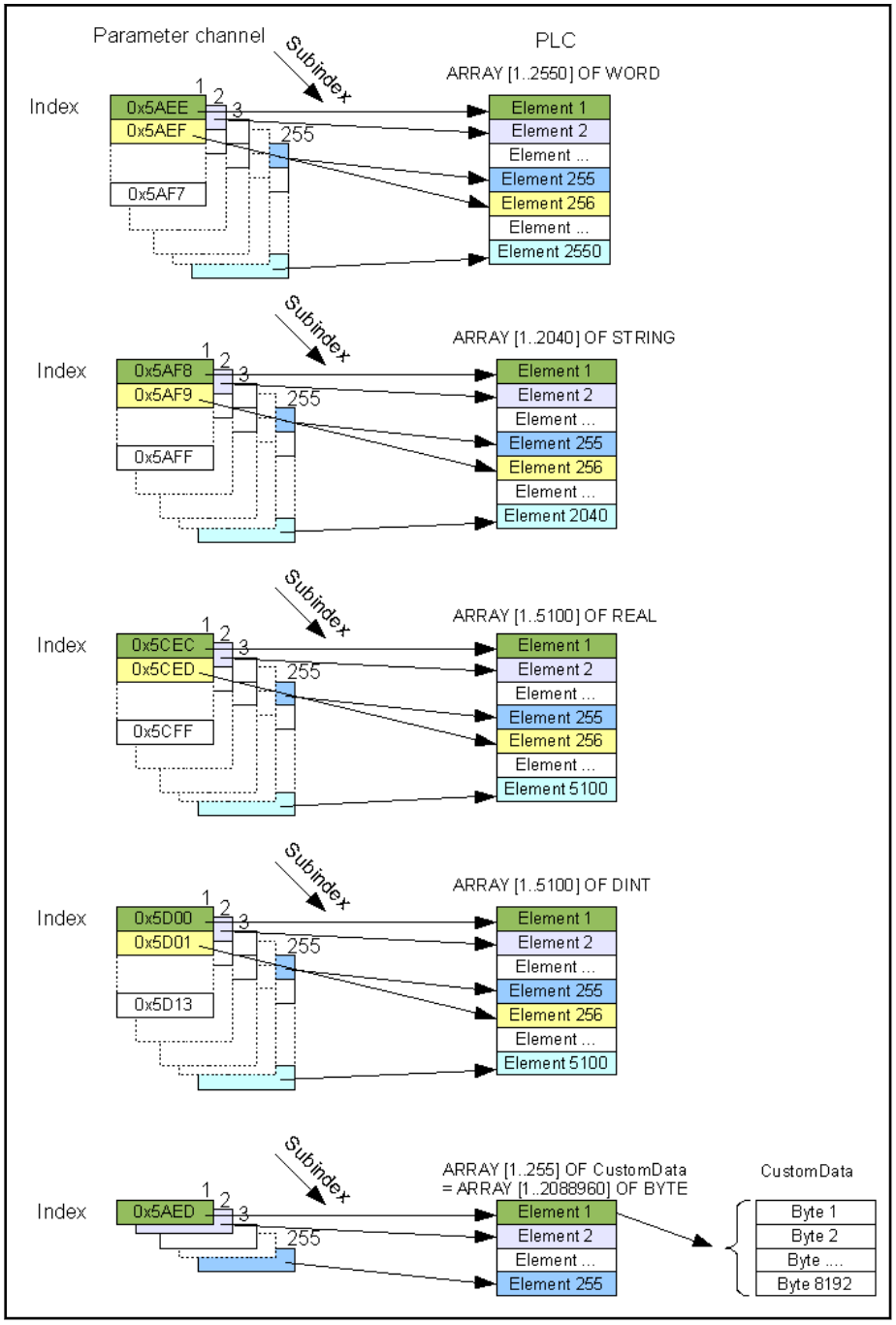


Fig. 12-12: Assignment of index/subindex to PLC variables

The mapping rules are more comprehensive. To get from a desired element number to the index/subindex, the following conversion formulae are used for the data types STRING, REAL, DINT and WORD:

Index = start index + [(element number - 1) / 255] (The division is integer which means the result is calculated without decimal places)

Subindex = element number – [(index - start index) * 255]

Example:

The 257th element of the REAL array should be read.

In this case, the start index is 0x5CEC and the element number is 257.

The following results for the index:

Index = 0x5CEC + (257 – 1) / 255 = 0x5CEC + (256 / 255) = 0x5CEC + 1 = 0x5CED

The following results for the subindex:

Subindex = 257 – [(0x5CED – 0x5CEC) * 255] = 257 – [1 * 255] = 2

To convert the index/subindex to the element number, the following formula applies:

Element number = [(index - starting index) * 255] + subindex

Example:

Which element is queried by the index 0x5D02 and by the subindex 3?

The index is in the range from 0x5D00 to 0x5D13 and thus, it is an element in the DINT array. The element number is:

Element number = [(0x5D02 – 0x5D00) * 255] + 3 = 2 * 255 + 3 = 513

The same mapping rules as above apply to the CUSTOM data. Since only one index is used, the mapping can be simplified respectively.

Index = start index = 0x5AED

Subindex = element number

To allow the calculation in the PLC program in reverse order (determine the element number from the index and the subindex), the following formulae can be used:

Element number = (index - starting index) * 255 + subindex

Example:

The function block indicates on its output that 0x5AF9 is written on the index and 5 on the subindex. On which element was thus written?

0x5AF9 is in the range from 0x5AF8 to 0x5AFF. Thus, it is a string.

The element number results from:

Element number = (0x5AF9 - 0x5AF8) * 255 + 5 = 0x1 * 255 + 5 = 260

It was written on the 260th element in the "STRING array".

Initializing the Communication

To initialize the communication or to cancel a pending query, the master can send a special request. The request is provided with the following content in the control word: 1000 000x 0000 1111b. The value of the toggle bit (T) is not important. If the master sends this control word, the slaves aborts all the processing and goes into a state in which it can process a new request. It responds in the status word with the content 1000 000x 0000 1111b.

Error Telegrams

If the slave detects an error, it sends a response telegram to the master in which the error bit (E) is set in the status word. There is an error code in the second word. This error code specifies the cause (also refer to the Error table, page 554).

The error code 0x1F00 is special here. It specifies the error that occurred within the control. In this case, the words 3 and 4 contain an error code of the control that can be found in the documentation of the control. Additional information on the error can be found in the words 5 and 6. If the parameter channel is shorter than 6 words, the telegram is fragmented.

Example:

Procedure of a read request

The following example is based on the assumptions below:

- The parameter channel is 8 bytes = 4 words long in both directions.
- The toggle bit is 0 at the beginning
- A PLC STRING variable (element 1) should be requested. Thus, the index 0x5AF8 and the subindex 1 have to be selected
- In this case, the slave responds with only one telegram. If the response is longer, the "last bit" would not be set in the status bit and the master would have to request telegrams until the "last bit" is set in the status word

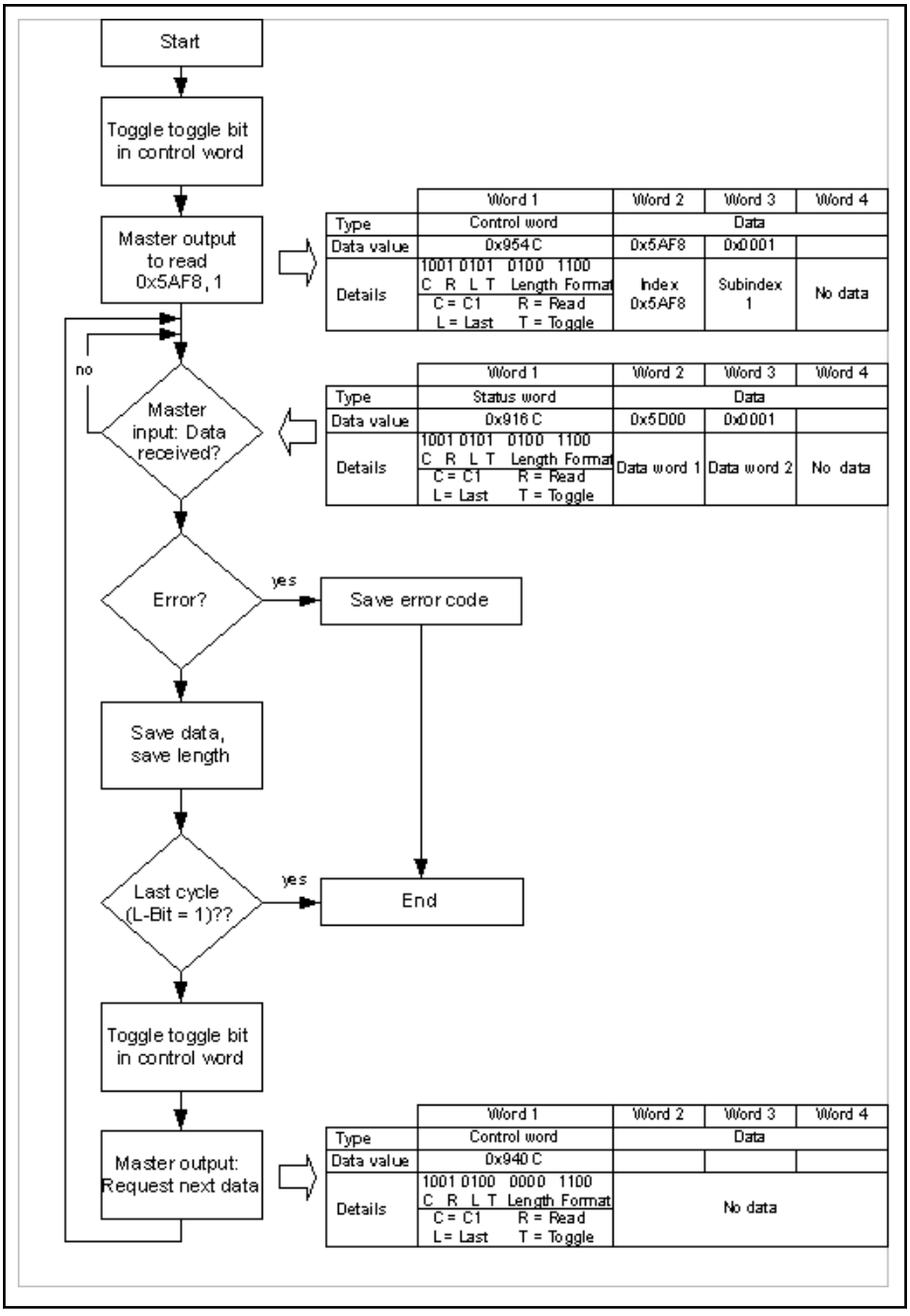


Fig.12-13: Example on how to read a PLC STRING variable

Example:

Procedure of a write request

The following example is based on the assumptions below:

- The parameter channel is 8 bytes = 4 words long in both directions.
- The toggle bit is 0 at the beginning
- The axis parameter "A-0-0002, Axis name" of the axis 22 should be written. Thus, the index 0x4002 and the subindex 22 have to be selected

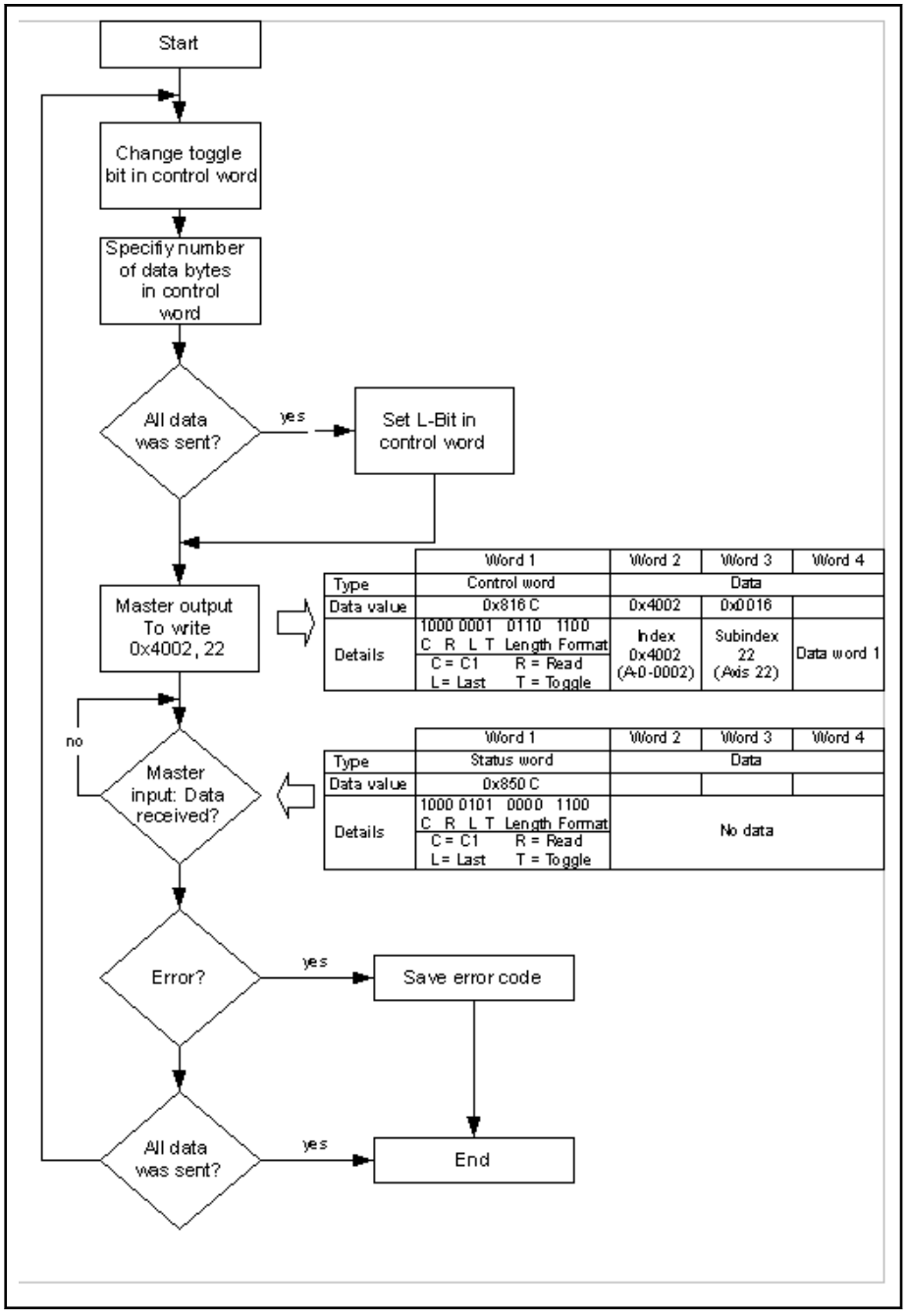


Fig.12-14: Example on how to write an axis parameter

12.3.2 Commissioning the function block

Before using the function block, a field bus connection must be established. The example of the PROFIBUS below shows how it is configured for operation.

Individual modules are configured for the PROFIBUS. It is recommended for the parameter channel to use own modules to achieve a structuring. Thus, there are modules of 8 and 16 bytes (parameter channel can obtain a length from 6 to 18 bytes). Alternatively, only parts of a module can also be used. This has to be considered when assigning the parameter channel function block. It is important that the inserted modules are modules with byte order. The following figure defines one module each with 16 byte inputs and outputs

("ParameterChannel_In" and "ParameterChannel_Out") for the parameter channel.

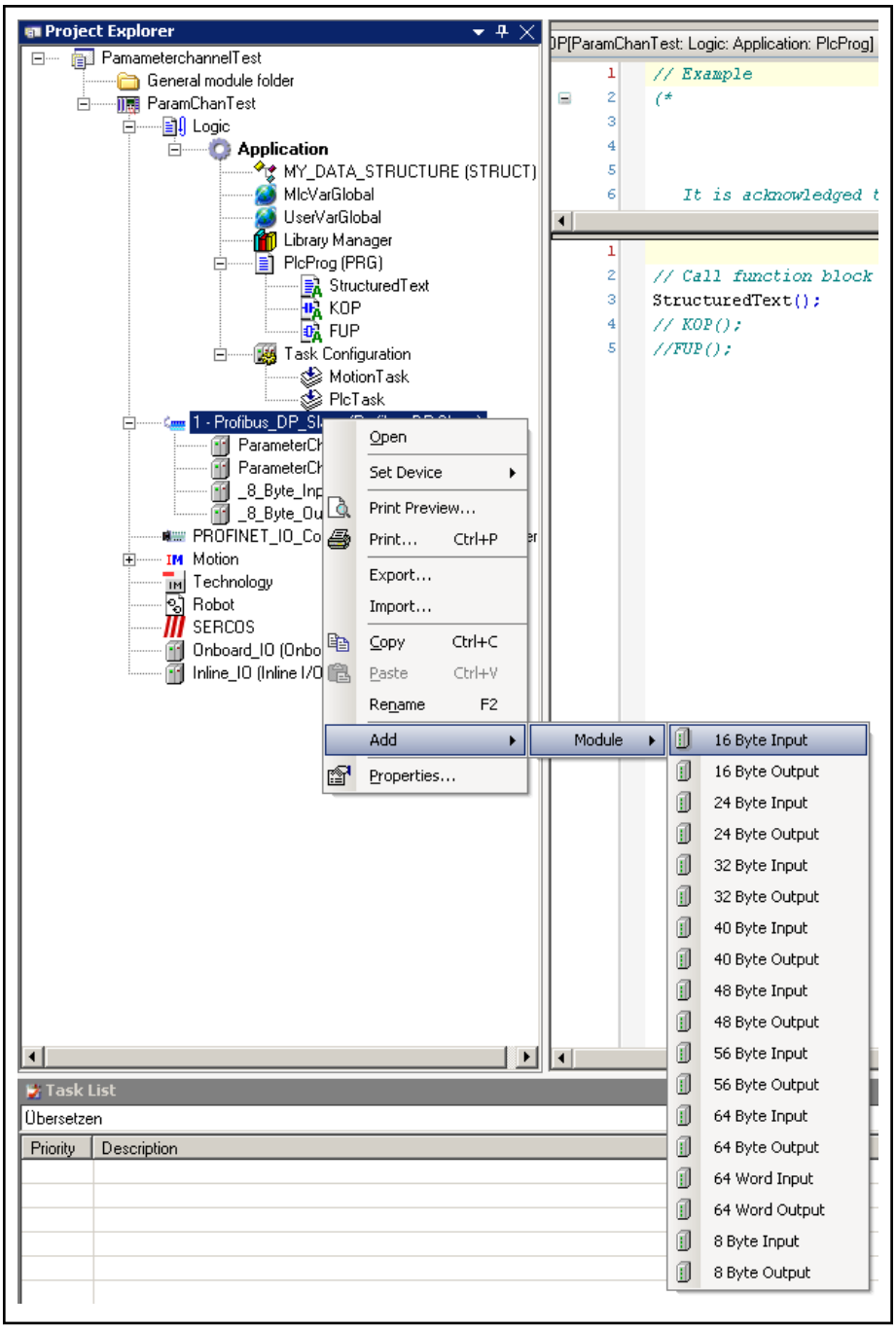


Fig.12-15: PROFIBUS module used as parameter channel

To operate the function block, the addresses for the field bus and the respective length have to be specified. There are several possibilities. A symbolic name can be given or the address of the I/O range can be used.

In the following figure, the name "myFieldbusOut " was specified for the output data of the field bus module "ParameterChannel_Out". This is carried out by double-clicking on the desired module and in the opening window in the tab "DP Modules I/O Mapping". The tab might have different names for different field buses. Alternatively, the address can also be used (here "%QB1").

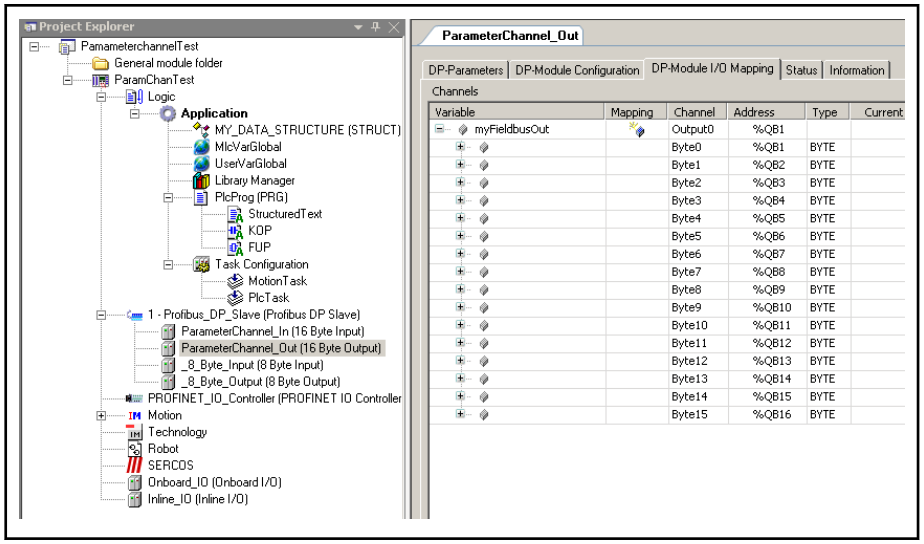


Fig. 12-16: Name and address of a PROFIBUS module

To assign the IL_ParameterChannel function block, default inputs/outputs have to be assigned. In addition, inputs for PLC objects must be assigned. Therefore, a pointer as well as the length of the data assigned are required. The STRING and CUSTOM data is something special since these can have user-specific lengths. Thus, the length of a string has also to be specified. In any case, the arrays always have to start with 1. Examples of the declaration of PLC variables are shown in the following source code. The maximum lengths are respectively specified for the arrays. The default inputs and outputs as well as the function block is declared.

Declaring an "IL_ParameterChannel" function block with data set

```
TYPE MY_DATA_STRUCTURE :
STRUCT
      // Custom structure, here 2048 * 4 = 8kB Data
      StructureData: ARRAY [1..2048] OF UDINT;
END_STRUCT
END_TYPE

VAR
      // Function block declaration
      Fb: IL_ParameterChannel;
      // PLC Data
      myRealArray:        ARRAY [1..4845] OF REAL;
      myDintArray:        ARRAY [1..4845] OF DINT;
      myWordArray:        ARRAY [1..2295] OF WORD;
      myStringArray:      ARRAY [1..2040] OF STRING(255);
      myCustomData:       ARRAY [1..255] OF MY_DATA_STRUCTURE;
      // In-/outputs of function block
      En:                 BOOL;
      InOp:               BOOL;
      Error:              BOOL;
      ErrorID:            ERROR_CODE;
      ErrorIdent:         ERROR_STRUCT;
      Index:              WORD;
      Subindex:           WORD;
      Reads:              UDINT;
      Writes:             UDINT;
END_VAR
```

The function block is called as follows:

RIL_ParameterChannel.library

```
1
2   Fb( Enable:= En,
3       FieldbusAddressIn:= %IB1,
4       ChannelLengthIn:= 16,
5       FieldbusAddressOut:= ADR(myFieldbusOut),
6       ChannelLengthOut:= 16,
7       SizeofRealArray:= SIZEOF(myRealArray),
8       SizeofDintArray:= SIZEOF(myDintArray),
9       SizeofWordArray:= SIZEOF(myWordArray),
10      SizeofStringArray:= SIZEOF(myStringArray),
11      SizeofCustomData:= SIZEOF(myCustomData),
12      RealArray:= ADR(myRealArray[1]),
13      DintArray:= ADR(myDintArray[1]),
14      WordArray:= ADR(myWordArray[1]),
15      StringArray:= ADR(myStringArray[1]),
16      CustomData:= ADR(myCustomData[1]),
17      SizeofString:= 255,
18      SizeofCustom:= SIZEOF(MY_DATA_STRUCTURE),
19      InOperation=> InOp,
20      Error=> Error,
21      ErrorID=> ErrorID,
22      ErrorIdent=> ErrorIdent
23      Index=> Index,
24      Subindex=> Subindex,
25      NumberOfReads=> Reads,
26      NumberOfWrites=> Writes);
27
```

Fig.12-17: Usage of the IL_ParameterChannel function block in structured text

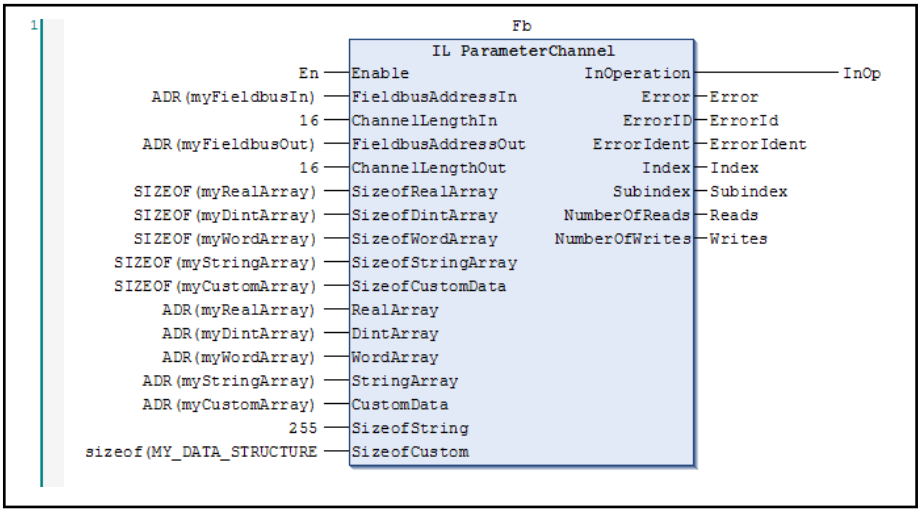


Fig.12-18: Usage of the IL_ParameterChannel function block in FBD

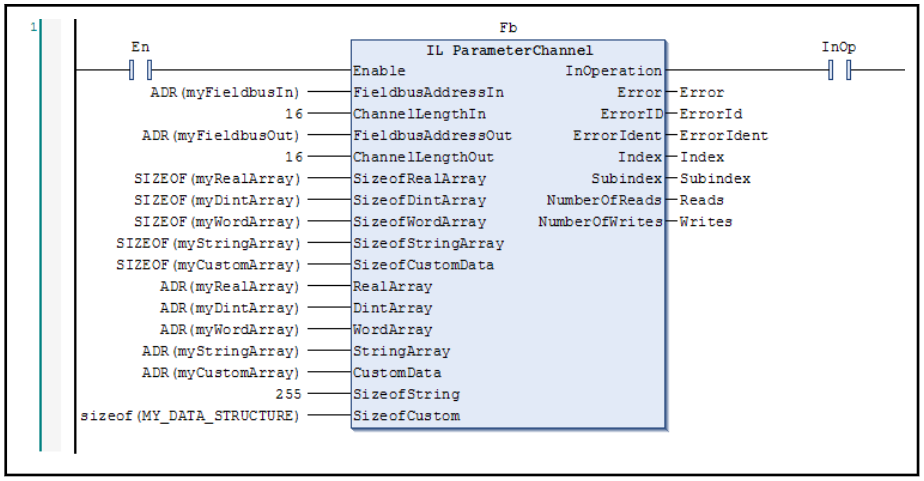


Fig. 12-19: Usage of the IL_ParameterChannel function block in LD

In the examples above, the method with the symbolic address ("myFieldbus-Out") or the method with the direct addressing ("%IB1") was used for the addresses of the field bus data in the structured text.

12.3.3 Differences to the Legacy Systems

General

The function block is a substitute for the already existing system VisualMotion. Existing differences are now described.

On the Field Bus Slave

To operate the parameter channel, the field bus has to be setup on the slave so that the additional cyclic data is configured for the parameter channel. This was also the case for the legacy system.

Additionally, the IL_ParameterChannel function block as well as different arrays have to be created for the IndraMotion MLC/IndraLogic XLC.

The function block may only be instantiated once. It is started by setting the "Enable" input and acts with regard to the default outputs ("InOperation", "Error" etc.) like other IndraMotion MLC/IndraLogic XLC function blocks.

The arrays to be created represent the variables and the registers in the VisualMotion. In any case, the arrays always have to start at the index 1. The arrays of strings and CustomData are additionally available and did not have comparable variables in the VisualMotion.

IndraMotion MLC/IndraLogic XLC		VisualMotion	
Declaration	**Data type**	**Name**	**Data type**
myReals: ARRAY [1..4845] of REAL	REAL (32 bit)	Program floats (4845 pieces)	FLOAT (32 bit)
myDints: ARRAY [1..4845] of DINT	DINT (32 bit)	Program Ints (4845 pieces)	INT (32 bit)
myWord: ARRAY [1..2295] of WORD	WORD (16 bit)	Register (2295 pieces)	INT (16 bit)
myString: ARRAY [1..255] of STRING	STRING (myStringSize)	--	--

Fig. 12-20: Comparison of the PLC data IndraMotion MLC/IndraLogic XLC/VisualMotion

RIL_ParameterChannel.library

On the Field Bus Master

The existing function blocks on the side of the master can still be used. But adaptations in the function block assignment have to be executed.

Addressing via "Index" and "Subindex" was changed or extended.

The S-parameters and P-parameters are identical to the VisualMotion system.

Only the data contents change for registers, program floats and program Ints:

- "Index" 0x5AEE – 0x5AF7 addressed the registers in VisualMotion. Now, the WORD array is accessed in the PLC
- "Index" 0x5CEC – 0x5CFF addressed the program floats in VisualMotion. Now, the REAL array is accessed in the PLC
- "Index" 0x5D00 – 0x5D13 addressed the program integers in VisualMotion. Now, the DINT array is accessed in the PLC

The data types and the calculation of the "index" and the "subindex" do not change.

The A-parameters were extended and remained compatible. Up to now, only the A-parameters till the number 2047 are possible. This function block can also be used to access A-parameters up to number 4096.

All the other data is empty data or additional data that has just been added.

Start index (16 bits)	End index (16 bits)	Subindex (8 bits)	Calculation rule
0x0000	0x1FFF		Reserved
0x2000	0x2FFF	Drive # (1-64)	Index = 0x2000 + S-parameter
0x3000	0x3FFF	Drive # (1-64)	Index = 0x3000 +P-parameter
0x4000	0x4FFF	Axis # (1-64)	Index = 0x4000 + A-parameter
0x5000	0x5AEC		Free
0x5AED	0x5AED	0x01 – 0xFF	Index = 0x5AEDSubindex = Custom data array element (element 1 - 255)
0x5AEE	0x5AF7	0x01 – 0xFF	Index = (0x5AEE + [(WORD array index - 1) / 255] subindex = (WORD array index) – [(index - 0x5AEE) * 255] (index 1 - 2295)
0x5AF8	0x5BEC	0x01 – 0xFF	Index = (0x5AF8+ [(STRING array index - 1) / 255] subindex = (STRING array index) – [(index - 0x5AF8) * 255] (index 1 - 244)
0x5BED	0x5CEB	0x00 – 0xFF	Free
0x5CEC	0x5CFF	0x01 – 0xFF	Index = (0x5CEC + [(REAL array index - 1) / 255] subindex = (REAL array index) – [(index - 0x5CEC) * 255] (index 1 - 4845)
0x5D00	0x5D13	0x01 – 0xFF	Index = (0x5CEC + [(DINT array index - 1) / 255] subindex = (DINT array index) – [(index - 0x5CEC) * 255] (index 1 - 4845)
0x5D14	0x5FFF	0x00 – 0xFF	Free
0x6000	0x6FFF	0x00	Index = 0x6000 + C-parameter
0x7000	0x7FFF	Kinematics # (1-16)	Index = 0x7000 + K-parameter
0x8000	0x8FFF	Probe # (1-99)	Index = 0x8000 + M-parameter

0x9000	0x9FFF	PLS # (1-4)	Index = 0x9000 + N-parameter
0xA000	0xAFFF	Osci # (1-16)	Index = 0xA000 + O-parameter
0xB000	0xFFFF		Free

Fig.12-21: Assigned address ranges for index/subindex

The error codes have (partially) been changed in the parameter channel. The differences are shown in the table below:

ErrorCode IndraMotion MLC/ IndraLogic XLC	ErrorCode VisualMotion	Description
16#8500	16#8500	Data too long (VisualMotion: >128 bytes; MLC > 64kBytes)
--	16#8800	Error not available on the IndraMotion MLC/IndraLogic XLC
16#8B00	16#8B00	Identical [the format specification in the control word is wrong (must be 16#0C)]
--	16#8D00	Error case is already checked on the IndraMotion MLC/ IndraLogic XLC while starting the function block
16#8C00	16#8C00	Identical (the given length is longer than the parameter channel itself)
--	16#9000	The format bits 0 to 3 of the control word have been changed during the transmission. This error is indicated on the IndraMotion MLC/IndraLogic XLC with the error number 16#8B00
16#9500	16#9500	Identical (data was sent even though a read request was made)
16#9600	--	C1 bit has not been set. The error does not exist on Visual-Motion, since the old formats of the parameter channel are still supported
16#9700	--	The "last" bit was not sent even though it is a read request. This error can only occur if a read request can be transferred within a fragment. This is not guaranteed for the VisualMotion since the parameter channel may only contain 2 words maximum as well. However, it is specified on the IndraMotion MLC/IndraLogic XLC that the parameter channel must contain at least 3 words. Due to that reason, a read request can always be transferred with a telegram
16#1F00	16#1Fxx	Control-specific error. In VisualMotion, the control-specific error is indicated in the lower byte. In the IndraMotion MLC/ IndraLogic XLC, 8 more bytes follow after the error code. These contain the error codes of the function blocks used to read/write parameters. The "Additional1" error codes is in the words 3 and 4 of the parameter channel, "Additional2" in the words 5 and 6
16#F200	16#F200	Identical (the given index is unknown)
16#F300	16#F300	Identical (the given subindex is unknown)

Fig.12-22: Differences between the IndraMotion MLC/IndraLogic XLC and VisualMotion error codes within the parameter channel

13 ML_TechTensionAdvanced.library

13.1 Introduction and Overview

The ML_TechTensionAdvanced library provides function blocks for multi-axis tension controllers with decoupling network that are **subject to license**.

This documentation describes the inputs and outputs of the individual function blocks and provides notes on their usage.

The following overview shows all function blocks of the ML_TechTensionAdvanced library:

The function blocks of the ML_TechTensionAdvanced

Function block	Function block
MB_TensionControlLoadCellType02	Multi-axis tension controller function block with dynamic precontrol for up to eight axes
MB_TensionDecouplingNetworkType01	Decoupling network function block for multi-axis tension controller

Fig. 13-1: Function blocks of the ML_TechRegiAdvanced library that are subject to license

☞ The usage of the functions blocks of the ML_TechTensionAdvanced library is subject to license. The respective license FWS-IM*MLC*-TEC-NNVRS-NN-TENS*ADV to use the function blocks has to be purchased. For each control, on which the function blocks are used, one license is required each (runtime license).

One license must be purchased for each control only authorizing the usage of function blocks of the ML_TechTensionAdvanced library on this control system.

13.2 Multi-Axis Tension Controller MB_TensionControlLoadCell-Type02

Brief Description

Depending on the measured value of a load cell, the tension controller controls the tension of up to eight axes to the desired command value.

Assignment: Target system/library

Target system	Library
IndraMotion MLC 12VRS	ML_TechMotion.compiled-library
IndraMotion MLD/MPx18 with additional package MA (in preparation)	In preparation

Fig. 13-2: Reference table of the MB_TensionControlLoadCellType02 function block

☞ The usage of the function block is subject to license. Details on the licensing requirement are described in the introduction of the documentation about the library (see page 571).

☞ IndraMotion MLC 12VRS or higher additionally supports slave axes with interpolation on the control.

ML_TechTensionAdvanced.library

Interface Description

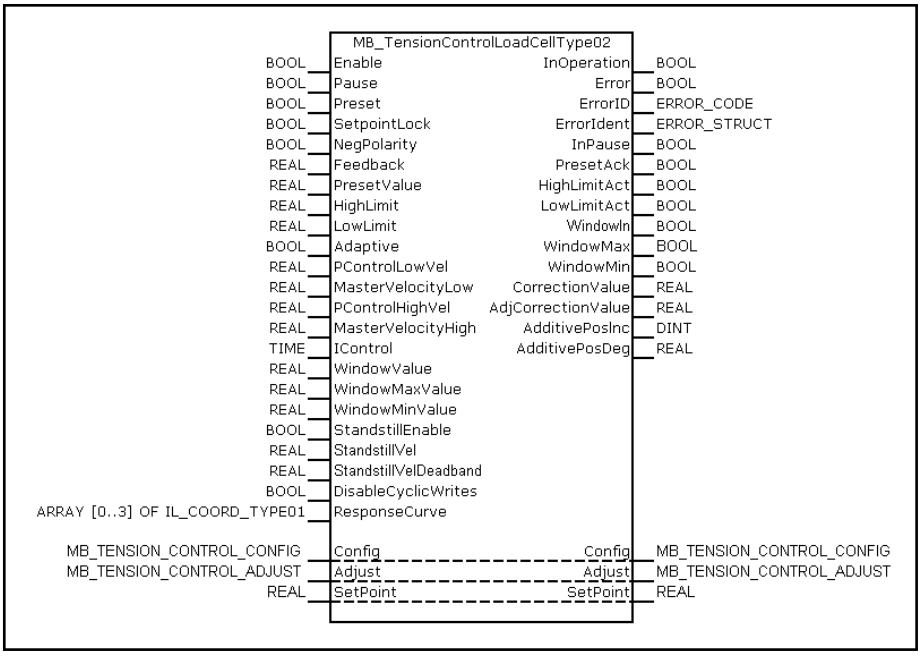


Fig.13-3: MB_TensionControlLoadCellType02 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	Config	MB_TENSION_CONTROL_CONFIG	Configuration of the tension controller (axes, tension distribution)
	Adjust	MB_TENSION_CONTROL_ADJUST	Structure for the slave axis settings at runtime
	Setpoint	REAL	Command value of the tension This is generally set by the user but can also be changed by the function block (see "SetpointLock" on page 582)
VAR_INPUT	Enable	BOOL	If this input is set, the function block operates
	Pause	BOOL	If this input is set, the tension controller is paused and the I-value of the output value is frozen. The "InPause" output signalizes an active "Pause"
	Preset	BOOL	At the rising edge of this input, the control output is set to the value parameterized in "PresetValue" and if successful, the "PresetAck" output is set
	SetpointLock	BOOL	At a rising edge at the input, the current actual tension value at "Feedback" is accepted as new command value for the tension controller
	NegPolarity	BOOL	If this input is set, the control variables "CorrectionValue" and "AdditivePosInc"/"AdditivePosDeg" are provided with a negative sign
	Feedback	REAL	Current actual tension
	PresetValue	REAL	The controller output is set to this value if the "Preset" input is set

I/O type	Name	Data type	Description
	HighLimit	REAL	Maximum value of the controller output. If the control variable calculated by the controller is greater than this value, the control variable is limited and the I-value of the controller is not summed up anymore
	LowLimit	REAL	Minimum value of the controller output. If the control variable calculated by the controller is lower than this value, the control variable is limited and the I-value of the controller is not summed up anymore
	Adaptive	BOOL	If this input is set, an adaptive P-gain is used for the closed-loop control
	PControlLowVel	REAL	P-gain of the controller if "Adaptive" = FALSE. Otherwise, the value corresponds to the P-gain at low master axis velocities
	MasterVelocityLow	REAL	Only effective if "Adaptive" = TRUE. Master axis velocity up to which "PControlLowVel" is used
	PControlHighVel	REAL	P-gain of the controller at high master axis velocities. If "Adaptive" = FALSE, this input is not effective
	MasterVelocityHigh	REAL	Only effective if "Adaptive" = TRUE. Master axis velocity from which "PControlHighVel" is used
	IControl	TIME	Integral action time of the I-controller If the value is set to 0, the I-controller is not effective
	WindowValue	REAL	Process variable window in [%]
	WindowMaxValue	REAL	Maximum value of the process variable in [%]
	WindowMinValue	REAL	Minimum value of the process variable in [%]
	StandstillEnable	BOOL	StandstillEnable = FALSE: "InVelocity Control" branch is active. Control variable: Gear ratio fine adjustment StandstillEnable = TRUE: "Standstill control" branch is active. Control variable: Master axis position, additive The additive velocity specified in "StandstillVel" is effective
	StandstillVel	REAL	Additive velocity for standstill control
	StandstillVelDeadband	REAL	This value determines the controller deviation value. Below this value, the tension velocity for the standstill control is reduced
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly to the optional cyclic data container, but only displayed at the output. The value is only evaluated at a rising edge at "Enable"
	ResponseCurve	ARRAY [0..3] OF IL_COORD_TYPE01	Characteristics to scale the control deviation in the "InVelocity Control" branch
VAR_OUTPUT	InOperation	BOOL	The tension controller operates and the outputs are valid

ML_TechTensionAdvanced.library

I/O type	Name	Data type	Description
	Error	BOOL	Indicates an error. If "Enable" = FALSE, the error is cleared
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	For a detailed error description according to the error table (see Error codes on page 585).
	InPause	BOOL	"Pause" is active
	PresetAck	BOOL	If this output is active, the "Preset" input was analyzed and the "CorrectionValue" output assumed the "PresetValue" value
	HighLimitAct	BOOL	This output is set if the control variable calculated by the controller is greater than the value specified in "HighLimit"
	LowLimitAct	BOOL	This output is set if the control variable calculated by the controller is lower than the value specified in "LowLimit"
	WindowIn	BOOL	If TRUE is set, the process variable is in the parameterized window
	WindowMax	BOOL	If TRUE is set, the process variable exceeded the parameterized maximum value
	WindowMin	BOOL	If TRUE is set, the process variable is below the parameterized minimum value
	CorrectionValue	REAL	Control variable for the tension controller
	AdjCorrectionValue	REAL	Control variable of the tension controller with dynamic precontrol
	AdditivePosInc	DINT	Additive master axis position in increments for each SERCOS cycle. It is used for the real slave axis with interpolation in the drive (target parameter: P-0-0692)
	AdditivePosDeg	REAL	12VRS or higher: Additive master axis position in degrees for each SERCOS cycle. It is used for the slave axis with interpolation on the control (e.g. for virtual axes or real axis with interpolation on the control. Target parameter: A-0-2600)

Fig.13-4: Interface variables of the MB_TensionControlLoadCellType02 function block

Min./max. and default values of the inputs

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
Enable	BOOL			FALSE	Continuous
Pause	BOOL			FALSE	Continuous
Preset	BOOL			FALSE	Continuous
SetPointLock	BOOL			FALSE	Continuous
NegPolarity	BOOL			FALSE	Continuous
Feedback	REAL	n.def.	n.def.	0.0	Continuous

Name	Type	Min. value	Max. value	Default value	Effective
PresetValue	REAL	n.def.	n.def.	0.0	Rising edge at "Preset"
HighLimit	REAL	>LowLimit	n.def.	0.00001	Continuous
LowLimit	REAL	-95.0	<HighLimit	-0.00001	Continuous
Adaptive	BOOL			FALSE	Continuous
PControlLowVel	REAL	>0.0	n.def.	0.0001	Continuous
MasterVelocity-Low	REAL	>0.0	<=MasterVelocity-High	0.0	Continuous
PControlHighVel	REAL	>0.0	n.def.	0.0	Continuous
MasterVelocity-High	REAL	>=MasterVelocity-Low	n.def.	1.0	Continuous
IControl	TIME	0 s	n.def.	20 ms	Continuous
WindowValue	REAL	0.0	100.0	0.0	Continuous
WindowMaxValue	REAL	>=WindowMinVal-ue	n.def.	0.0	Continuous
WindowMinValue	REAL	0.0	<=WindowMax-Value	0.0	Continuous
StandstillEnable	BOOL			FALSE	Continuous
StandstillVel	REAL	0.0	n.def.	0.0	Continuous
StandstillDead-band	REAL	0.0	n.def.	1.0	Continuous
DisableCyclic-Writes	BOOL			FALSE	Rising edge at "Enable"
ResponseCurve	ARRAY [0..3] OF IL_CO-ORD_TYPE01	See RIL_Loop-Control	See RIL_Loop-Control	See data struc-ture	Rising edge at "Enable"
Config	MB_TEN-SION_CON-TROL_CONFIG	See data struc-ture	See data struc-ture	See data struc-ture	Rising edge at "Enable"
Adjust	MB_TEN-SION_CON-TROL_ADJUST	See data struc-ture	See data struc-ture	See data struc-ture	Continuous
Setpoint	REAL	n.def.	n.def.	0.0	Continuous

Fig. 13-5: Min./max. and default values of the MB_TensionControlLoadCell-Type02 function block

Min./max. and default values of the MB_TENSION_CON-TROL_CONFIG structure

ML_TechTensionAdvanced.library

Name	Type	Min. value	Max. value	Default value	Description
Master	AXIS_REF			NoAxis	Master axis
Slave[1]	AXIS_REF			NoAxis	Pull roll 1
Slave[2]	AXIS_REF			NoAxis	Pull roll 2
Slave[3]	AXIS_REF			NoAxis	Pull roll 3
Slave[4]	AXIS_REF			NoAxis	Pull roll 4
Slave[5]	AXIS_REF			NoAxis	Pull roll 5
Slave[6]	AXIS_REF			NoAxis	Pull roll 6
Slave[7]	AXIS_REF			NoAxis	Pull roll 7
Slave[8]	AXIS_REF			NoAxis	Pull roll 8
Circumference	REAL	0.0	n.def.	0.0	Axis circumference of the first pull roll
DistanceRoll1To2	REAL	0.0	n.def.	0.0	Web length between feed roll and first pull roll

Fig.13-6: Structure description of the MB_TENSION_CONTROL_CONFIG

Structure element	Type	Default value	Description
SlavePreCtrl[1]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 1
SlavePreCtrl[2]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 2
SlavePreCtrl[3]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 3
SlavePreCtrl[4]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 4
SlavePreCtrl[5]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 5
SlavePreCtrl[6]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 6
SlavePreCtrl[7]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 7
SlavePreCtrl[8]	MB_TENSION_PRE_CTRL	NO_PRE_CTRL	Precontrol for slave axis 8

Fig.13-7: Structure description of the MB_TENSION_CONTROL_ADJUST

Element	Description
NO_PRE_CTRL	No precontrol/closed-loop control of the axis
PRE_CTRL_100	Precontrol of the axis without time constant
PRE_CTRL_DYNAMIC	Dynamic precontrol

Fig.13-8: Structure of the enumeration type MB_TENSION_PRE_CTRL

Functional Description

As shown in the following, several pull rolls are controlled by the function block MB_TensionControlLoadCellType02. The measured values of the load cell are assigned to the "Feedback" input of the function block. The command tension is set via the "SetPoint" input. In normal mode ("InVelocity Control" branch is active), the tension is controlled by the gear ratio fine adjustment of the pull rolls. Depending on the axis type, it is written on the respective parameter (P-0-0694 at interpolation in the drive or A-0-2605 at interpolation on the control). If the master axis is at standstill, the pull rolls are moved via the additive master axis position, since the gear ratio fine adjustment does not influence the tension. Depending on the axis type, it is written on the respective

parameter (P-0-0692 at interpolation in the drive or A-0-2600 at interpolation on the control).

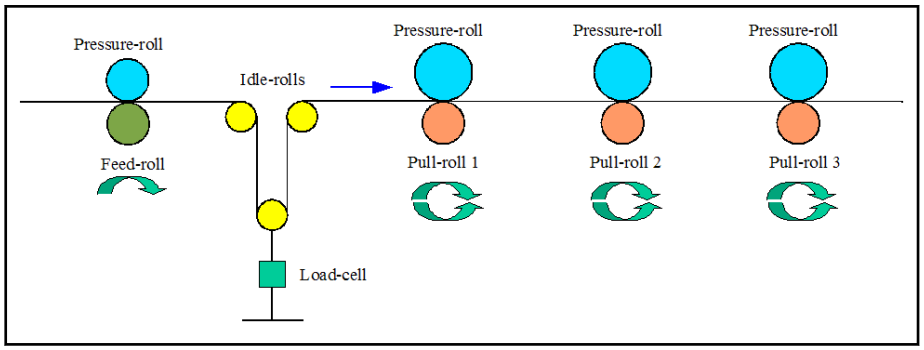


Fig. 13-9: Multi-axis tension controller

The following explains the individual functions of the function block inputs in detail. In general, all outputs are continuously analyzed if the "Enable" input is set. That means that changes to the inputs become immediately effective without toggling the "Enable" input again.

An exception is the "PresetValue" input which is only accepted at a rising edge at the "Preset" input. Likewise, the "DisableCyclicWrites" input is only adopted at a rising edge at the "Enable" input.

The complete mode of operation of the tension controller is shown in the following flowcharts. Depending on the "StandstillEnable" input, either the "InVelocity" path or "Standstill Control" is effective.

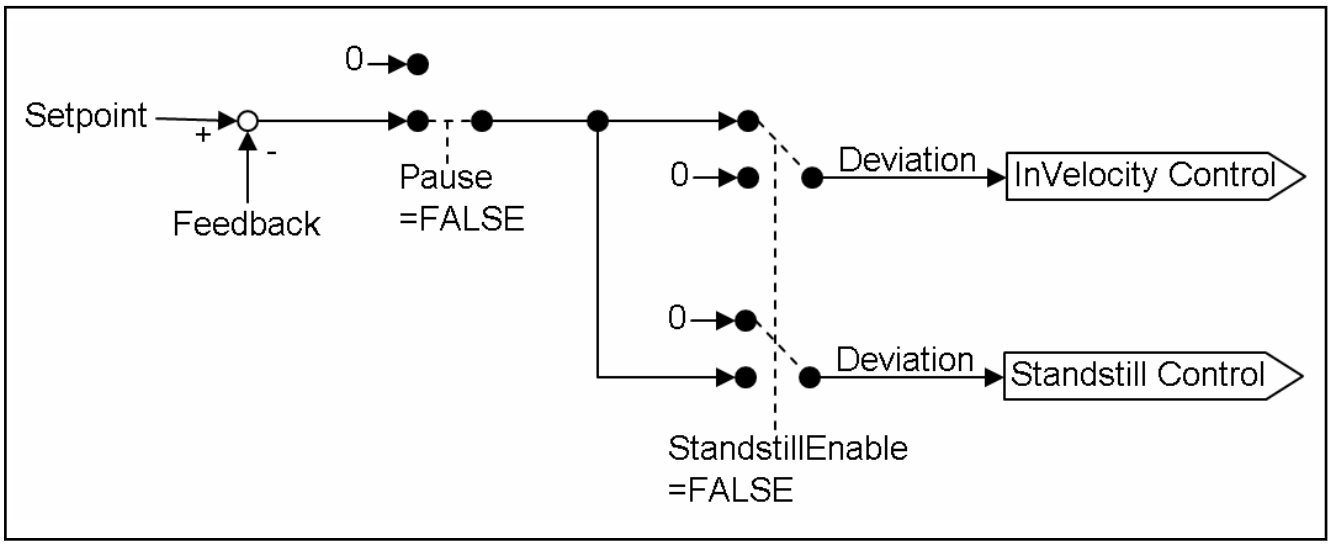


Fig. 13-10: Flowchart of the tension controller with branch for the control at standstill/velocity

The "InVelocity Control" path is effective if the "StandstillEnable" input is not set. The tension is controlled via the parameter "Gear ratio fine adjustment" P-0-0694 (slave axis with interpolation in the drive/IPO Drive) or A-0-2605 (slave axis with interpolation on the control/IPO Control). It is directly written on the parameters if the "DisableCyclicWrites" input is not set. For real axes with interpolation in the drive, the configuration of the parameter P-0-0694 in the optional cyclic SERCOS data (MDT) is thus required.

The control variable is additionally output via the "CorrectionValue" output.

ML_TechTensionAdvanced.library

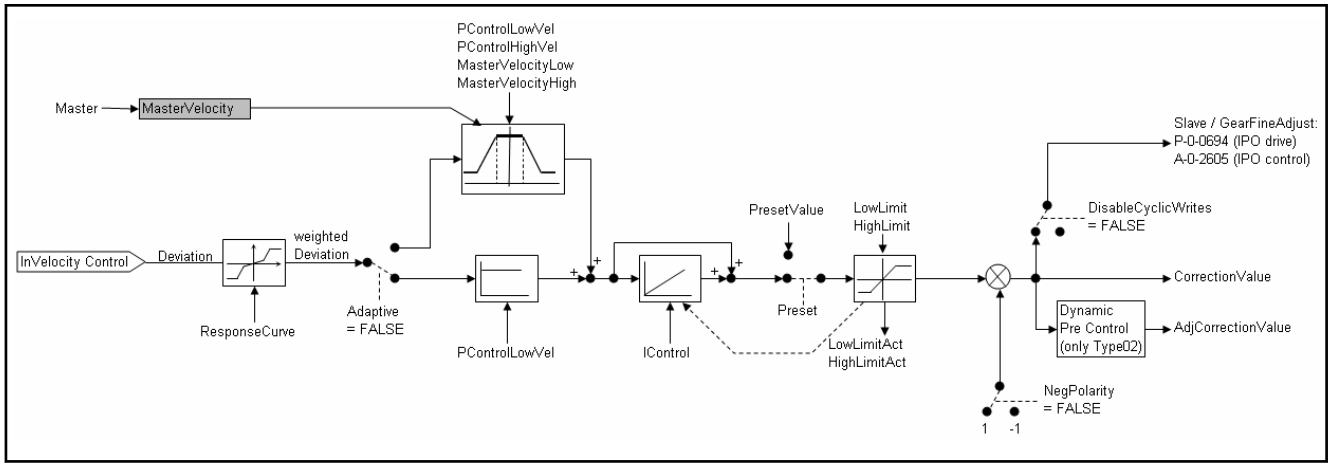


Fig.13-11: Flowchart of the tension controller for the "InVelocity Control" branch

The "Standstill Control" path is effective if the "StandstillEnable" input is set. This path has an effect on the additive master axis position of the slave axis to specify a velocity. It is directly written on the parameters if the "DisableCyclicWrites" input is not set. For slave axes with interpolation in the drive, it is written on the target parameter P-0-0692 as additive master axis position. For slave axes with interpolation on the control, it is written on the target parameter A-0-2600 as additive master axis position. For real axes with interpolation in the drive, the configuration of the parameter P-0-0692 in the optional cyclic SERCOS data (MDT) is thus required.

At the same time, the additive master axis position is available via the "AdditivePosInc" output (at interpolation in the drive) or "AdditivePosDeg" (at interpolation on the control).

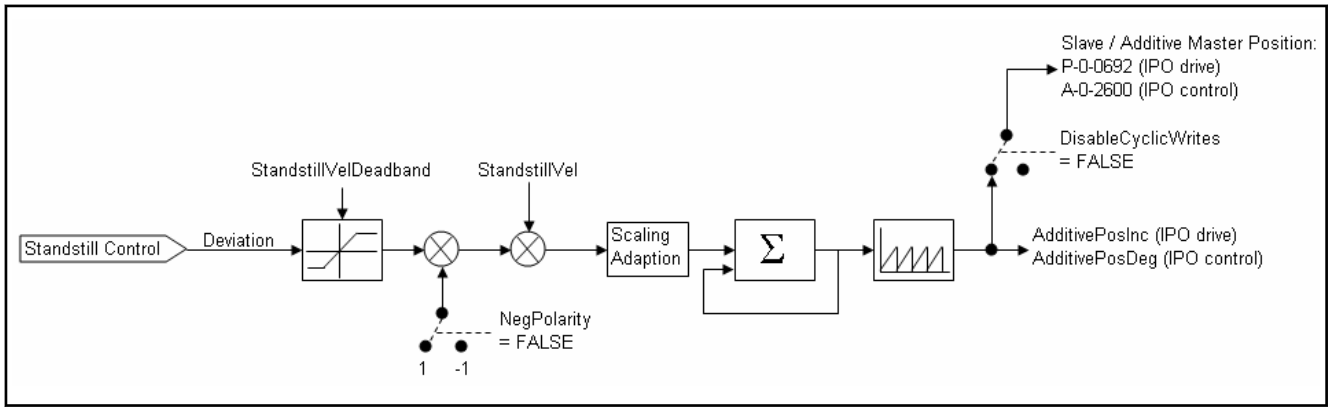


Fig.13-12: Flowchart of the tension controller for the closed-loop control at standstill (Standstill Control)

To ensure user-friendliness, the flowchart does not contain the "WindowValue", "WindowMaxValue" or "WindowMinValue" limitations and their status displays.

Commissioning

If "DisableCyclicWrites" is inactive, both the P-0-0694 (Gear ratio fine adjustment) and P-0-0692 (Additive master axis position) parameters have to be applied in the cyclic command value channel of the slave axes if real slave axes (Config.Slave[1..8]) with interpolation in the drive are used. At a rising edge at "Enable", the cyclic data channels are checked. If one of the two parameters is not found, a respective error message is returned at the error outputs of the function block. If "DisableCyclicWrites" is enabled, the configuration of the cyclic command value data channels is not checked.

For slave axes with interpolation on the control (e.g. virtual axis), the cyclic data channels do not have to be configured.

- The slave axes used have to be in the synchronous operation mode before the "Enable" input is activated. This can be executed by the AxisInterface or by the PLCopen function blocks "GearIn" or "GearInPos". Exception: The synchronous operation mode is not required if the "Pause" input is set
- The master axis has to be scaled to modulo 360 degrees
- The slave axes can be configured as combined (e.g. slave axis 1 with interpolation on the control, slave axis 2 with interpolation in the drive). **In case of a mixed configuration, the default value of the parameter P-0-0750 of each real slave axis with interpolation on the drive has to be 1.**

Config Static configuration parameters such as axis configuration and geometry are determined via the "MB_TENSION_CONTROL_CONFIG" structure. The "Master" input contains the axis reference to the master axis. All pull rolls are specified in the "Slave" array input. An axis that is not present is represented by the "NoAxis" constant. For the dynamic precontrol, the circumference of the first draw roll is required in the "Circumference" structure entry as well as the web length between the feed roll and the first pull roll in the "DistanceRoll1to2" structure entry.

Adjust The "MB_TENSION_CONTROL_ADJUST" structure contains settings that can be changed at runtime. The type of draw roll triggering is specified in the "SlavePreCtrl" array. If the constant "NO_PRE_CTRL" is entered for an axis, the axis is not controlled. The gear ratio fine adjustment (P-0-0694 or A-0-2605) is set to "0" and cannot be changed.

The constant "PRE_CTRL_100" causes the value of the output "CorrectionValue" to be written into the gear ratio fine adjustment.

In the "PRE_CTRL_DYNAMIC" setting, the "AdjCorrectionValue" output is written into the gear ratio fine adjustment of the axis. The dynamic draw roll control is used for all rolls following the first pull roll. To accomplish this, the constant "PRE_CTRL_DYNAMIC" has to be entered in "SlavePreCtrl" of the subsequent pull rolls. The first pull roll has to remain at "PRE_CRTL_100". Due to this setting, the tension of the following draw rolls is not affected when the velocity is modified.

The settings in "SlavePreCtrl" do not affect the motion in Standstill mode.

ML_TechTensionAdvanced.library

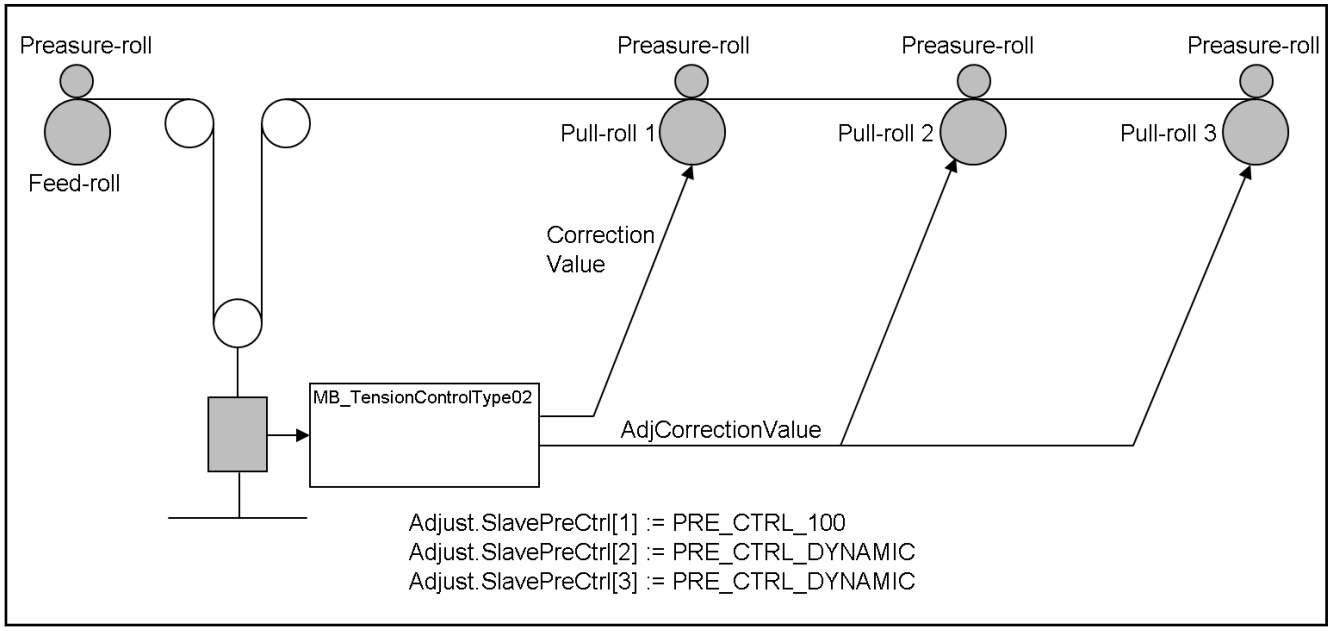


Fig.13-13: MB_TensionControlType02: Example cascade with three pull rolls

Preset/PresetValid

A positive edge at the "Preset" input specifies an initial value for the control variable using the "PresetValue" variable. At the same time, the internal variables of the controller are set in a way that the controller is approached without any jerk. "Preset" acts only on the gear fine adjustment in the path "InVelocity Control".

The "Preset" input is only evaluated when "Enable" is TRUE. At a rising edge at "Preset", the new value is accepted at "PresetValue" and "PresetAck" is set. A falling edge at "Preset" or "Enable" deletes "PresetAck". The "Preset" inputs dominates "Pause". That means that the value at "PresetValue" is applied at the output even though "Pause" is set. If "Preset" is active at the rising edge at "Enable", "PresetValue" is immediately assumed as control variable.

☞ The "Preset" function should be executed in standstill (StandstillEnable=TRUE). Otherwise, the velocity jumps can occur due to the setting of the ControlValue to the PresetValue. The value of the jump is proportional to the velocity.

The "Preset" input is only effective at the rising edge.

DisableCyclicWrites

The "DisableCyclicWrites" input determines whether the values calculated by the function block are to be written directly via the cyclic channel (FALSE) or whether they are only available at the function block output (TRUE).

Thus, it is possible to cascade several function blocks if "DisableCyclicWrites" is set to TRUE. Then, only the outputs ("CorrectionValue" and "AdditivePosInc" or "AdditivePosDeg") can be used and modified later on. However, the user still has to write the values into the target parameter.

If "DisableCyclicWrites" is set to TRUE, it is not checked whether the optional cyclic parameters are also configured.

ResponseCurve

Using the "ResponseCurve" input, the control deviation can be scaled. This is achieved with a characteristic curve with up to four data points. If the input is not assigned, a 1-to-1 characteristic curve is used as default value. That means that it is not scaled.

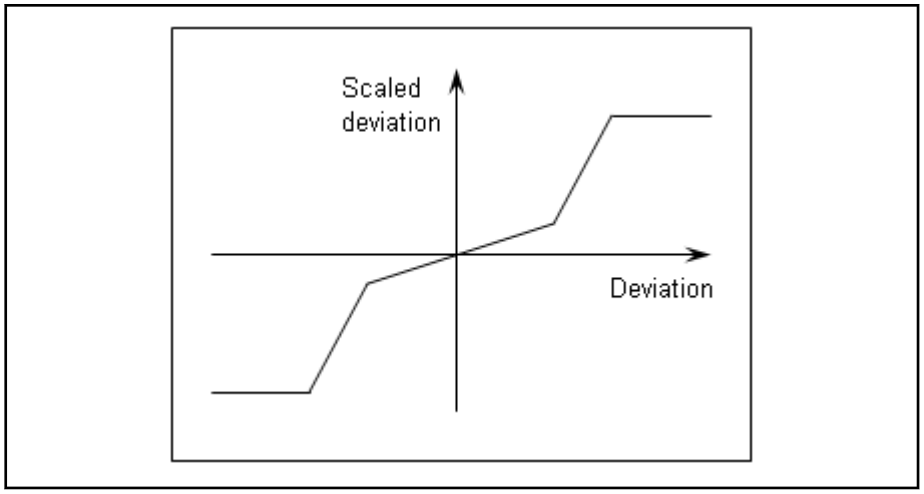


Fig.13-14: Scaling the control deviation using the "ResponseCurve" input

The function can for example be used to specify a smaller scaling of the control deviation for smaller control deviations. This results in a lower proportional gain and thus causes a smoother control behavior in steady state. When parameterizing the input, the controller parameters should first be set without evaluation. Subsequently, a lower scaling should be specified. This allows to set the controller parameters first with the system stability and then, the steady state is optimized.

Adaptive, PControlLowVel, PControlHighVel, MasterVelocityLow, MasterVelocityHigh

The P-gain can either be specified as constant P-gain or as adaptive P-gain. Via the "Adaptive" input, either an adaptive or a constant P-gain is selected. Setting "Adaptive" to TRUE selects the adaptive P-gain.

If a constant P-gain is selected, only the value in "PControlLowVel" affects the P-gain.

If an adaptive P-gain is selected, the "PControlLowVel", "PControlHighVel", "MasterVelocityLow" and "MasterVelocityHigh" inputs have to be assigned. The characteristic curve looks as follows:

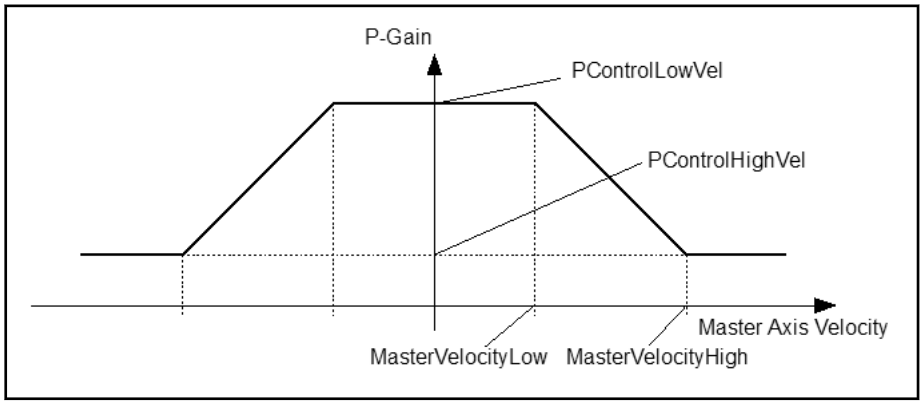


Fig.13-15: Characteristic curve for an adaptive P-gain

The individual inputs describe the characteristic curve for positive master axis velocities. The characteristic curve for negative master axis velocities results from its mirror image at the axis of the P-gain, i.e. only symmetric characteristics are possible. The P-gain between the master axis velocities "MasterVelocityLow" and "MasterVelocityHigh" is linearly interpolated. This results in some conditions for the value ranges of the inputs:

- The values for "MasterVelocityLow" and "MasterVelocityHigh" have to be positive

- The values for "MasterVelocityLow" and "MasterVelocityHigh" may not be identical
- The "MasterVelocityLow" value has to be lower than the "MasterVelocityHigh" value

If one of these conditions is not met, an error is generated at the output of the function block.

SetpointLock Setting the "SetpointLock" input applies the current actual tension value as the new command value for the tension controller. For example, the function allows the command value of the tension to obtain a reasonable initial value that reflects the current state of the machine.

Pause The "Pause" input enables the tension controller to pause. If this input is set to TRUE, the control deviation is set to "0", i.e. the controller output is frozen at the current I value. If "Pause" is active, the "InPause" output is set. At active pause, the synchronization mode is not required and the power (MC_Power) can be switched off.

☞ If no tension control is used at standstill (StandstillEnable is always FALSE), the "Pause" input has to be set at machine standstill.

NegPolarity The "NegPolarity" input affects the control direction. Depending on the position of the pull roll with regard to the blending direction of the material and the position of the feed roll, the control direction has to be either positive or negative. If this input is FALSE, the control direction is positive and a positive control deviation results in a positive control motion. The input also affects the standstill control.

Feedback This input represents the actual value of the tension. It can be read via an analog input at the drive or at the Inline I/O for example. The user is responsible for a reasonable scaling.

PresetValue Setting the "Preset" input specifies an initial value for the control variable using the "PresetValue" variable. At the same time, the internal variables of the controller are set in a way that the controller is approached without any jerk.

HighLimit, HighLimitAct This input limits the control variable of the controller to a maximum value. If the limitation becomes effective, the I-value of the controller is no longer summed up ("Anti-reset-windup"). The maximum value is determined by the maximum value for the gear ratio fine adjustment defined in the drive. If a limitation becomes effective, the "HighLimitAct" output is set.

LowLimit, LowLimitAct This input limits the control variable of the controller to a minimum value. If the limitation becomes effective, the I-value of the controller is no longer summed up ("Anti-reset-windup"). The minimum value is -95 irrespective of possible lower values in the drive (also refer to the section "Gear ratio fine adjustment"). If a limitation becomes effective, the "LowLimitAct" output is set.

WindowValue, WindowIn With the "WindowValue" value, a symmetric window can be defined containing the current actual tension value. The window is defined as follows:

$$\left(SetPoint - |SetPoint| \cdot \frac{WindowValue}{100\%}\right) \leq Feedback \leq \left(SetPoint + |SetPoint| \cdot \frac{WindowValue}{100\%}\right)$$

Fig.13-16: WindowValue, WindowIn

If this condition is met, the "WindowIn" output is set. If the condition is not met, the output is not set. A limitation is thus not carried out.

WindowMaxValue, WindowMax This value can be used to set the upper value for the actual tension value. The value is evaluated as follows:

$$Feedback > \left(SetPoint + |SetPoint| \cdot \frac{WindowMaxValue - 100}{100} \right)$$

Fig.13-17: WindowMaxValue, WindowMax

If this condition is met, the "WindowMax" output is set. If the condition is not met, the output is not set. A limitation is thus not carried out.

WindowMinValue, WindowMin

This value can be used to specify a lower value for the actual tension value. The value is evaluated as follows:

$$Feedback < \left(SetPoint + |SetPoint| \cdot \frac{WindowMinValue - 100}{100} \right)$$

Fig.13-18: WindowMinValue, WindowMin

If this condition is met, the "WindowMin" output is set. If the condition is not met, the output is not set. A limitation is thus not carried out.

StandstillEnable

By setting the "StandstillEnable" input, the velocity specified in "StandstillVel" is additively added to the axes to be controlled. This allows a closed-loop tension control at standstill as well.

☞ At lower velocities, the influence of the gear ratio fine adjustment decreases continuously. There is also an open control loop at master axis standstill, since the gear ratio fine adjustment has lost all its influence. Thus, the "StandstillEnable" input has to be set when reaching the standstill window in order not to increase the I-portion up to the limit in the "InVelocity Control" path by the open control loop.

☞ For axes with interpolation on the control, the "AdditivePosInc" output is given in modulo format 2^{20} (12VRS or higher).

☞ If this input is used, it must be ensured that the tension controller is only used in the SERCOS synchronous task. If this is not the case, the motor cannot run smoothly or oscillate around the standstill position.

StandstillVel

This value indicates the additive velocity sent to the drive when the "StandstillEnable" input is set. The units used for the velocity correspond to the scaling set in the drive.

☞ At standstill, the velocity is generated in jumps. Since the velocity has to be reached within one SERCOS cycle, the values specified cannot be indefinitely high. If the value selected for the standstill velocity is too high, oscillation of the drive can be caused. The maximum value depends on the scaling of the velocity, the maximum acceleration set and the mass connected to the drive. To avoid big velocity jumps, a filter time constant can be defined in the drive parameter P-0-0693. This smoothes the jump.

Background information

Gear ratio fine adjustment

The tension controller generally operates with velocities greater than "0". The gear ratio fine adjustment is an appropriate tool to maintain the tension. However, there are also some boundary conditions that are explained.

The gear ratio fine adjustment is internally limited to a minimum of -95%. If a lower value is entered, the function block outputs an error.

Reason for this limitation is the calculated output value that is defined as follows at a given input value:

$$Output\ value = Input\ value * \left(1 + \frac{Gear\ fine\ adjust}{100}\right)$$

Fig.13-19: Output value

Based on the equation, it can be seen that when the gear ratio fine adjustment approaches 100%, the output value approaches "0". Thus, the control loop practically opens since a modification of the input value does not cause any response at the output. It is even worse at a gear ratio fine adjustment of less than -100% because the sign of the output value reverses. In the control loop, positive feedback would occur that can result in an unstable system.

StandstillEnable, StandstillVel, StandstillVelDeadband

The standstill control can be executed using the "StandstillEnable" and "StandstillVel" inputs. This adds an additive velocity. To achieve better control quality, it is reduced by the command value. This can be executed with a modified signum function provided with the following transfer function:

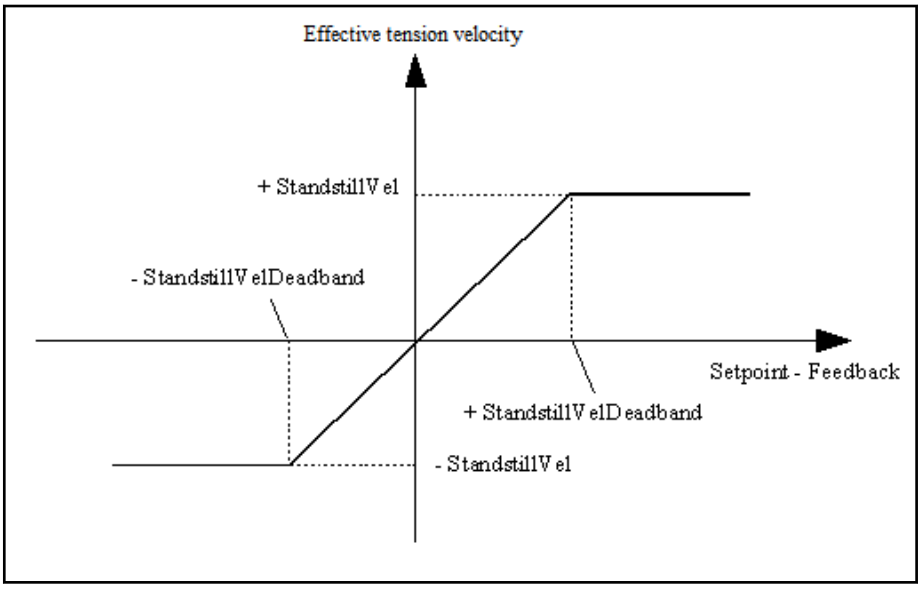


Fig.13-20: Modified signum function to limit the added "StandstillVel"

The control deviation is relativized in the "Setpoint" range. If the "Feedback" approaches the "Setpoint", the closed-loop control becomes "more sensitive" than a simple two-position controller. The value responsible for the limitation can be set at the "StandstillVelDeadband" input. The values are provided with the same units as the "Setpoint" input or "Feedback" input. Linear interpolation occurs between the two extreme values "+StandstillVel" and "-StandstillVel" so that the transfer function above results for control deviation and tension velocity.

Then, the factor determined is multiplied with the specified "StandstillVel". The velocity is processed as speed in the scaling set in the drive. If the web velocity should be entered instead of the speed, it is to be calculated before.

To reverse the direction of motion for the standstill control, set the "NegPolarity" input.

Subsequently, the result is multiplied by a correction factor. This correction factor results from the gear ratio. However, the gear ratio fine adjustment is not taken into consideration. In practice, the gear ratio fine adjustment is generally very small and is therefore not important. The gear ratio is only accepted at a rising edge at the "Enable" input.

The flowchart shows that the velocity specified is converted to the additive master axis position in increments or degrees. A few rules are to be considered:

- The specified velocity cannot be unlimited because the drive has to be able to reach the velocity within one SERCOS cycle. Thus, only very small values are allowed. The value size depends on the scaling used and the mass connected to the motor. The calculated increments/position are specified via the parameter P-0-0692 (slave axis with interpolation in the drive) or A-0-2600 (slave axis with interpolation on the control). It can also be smoothed via filters P-0-0693 (slave axis with interpolation in the drive) or A-0-2601 (slave axis with interpolation on the control) allowing also higher jumps
- The increments calculated are added before the gear. The gear ratio (A-0-2720 and A-0-2721 or P-0-0156 and P-0-0157) is included in the calculation as well. The gear ratio fine adjustment (P-0-0083/A-0-2722 and P-0-0694/A-0-2605) is not taken into consideration because it is generally very small. This can result in the axis moving at a slightly different velocity than the velocity set
- The units are scaled. In addition, the velocity value entered is not checked against the maximum and minimum values. This can result in very large values for the number of increments/position per SERCOS cycle. If the calculated interim value exceeds the maximum values by ± half of a modulo value, it is automatically limited. This results in a less additive velocity than specified at the "StandstillVel" input
- At very low velocities, it also has to be considered that the current velocity can deviate from the specified velocity. This can be caused due to the increments granulated per SERCOS cycle: e.g. if one increment corresponds to a velocity of 0.03 rpm, only velocities that are multiples of 0.03 rpm become effective. If this is not the case, it is rounded up to the next value.
- The calculated increments are processed with each SERCOS cycle. Thus, when using the standstill control, the function block has to be processed in the Motion/SERCOS synchronous task. If this is not the case, the drive moves one SERCOS cycle around the calculated number of increments and in the following SERCOS cycles it comes again to a standstill which results in a jerky motion. The filter in P-0-0693 can also be helpful.

For slave axes with interpolation in the drive, it is recommended to set the filter time constant of the P-0-0693 with a quadruple of the SERCOS cycle time. If the filter time constant is = "0", the drive outputs the warning E2070 "Velocity limitation activated" when tensioning at standstill since the additive master axis position transferred during the SERCOS cycle is not precisely interpolated within the position control cycle in the drive.

Error codes

The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**, 16#0170:

ML_TechTensionAdvanced.library

ErrorID	Additional1	Additional2	Description
ACCESS_ERROR (16#0004)	16#00000A00	16#00000001	The parameter P-0-0694 is not parameterized in the optional cyclic telegram of the slave axis
ACCESS_ERROR (16#0004)	16#00000A00	16#00000002	The parameter P-0-0692 is not parameterized in the optional cyclic telegram of the slave axis
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000001	The master axis velocity in the lower operating point ("MasterVelocityLow") is < 0
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000002	The master axis velocity in the upper operating point ("MasterVelocityHigh") is < 0
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000003	The master axis velocity in the lower operating point ("MasterVelocityLow") and upper operating point ("MasterVelocityHigh") are provided with the same values
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000004	The value of the master axis velocity in the lower operating point ("MasterVelocityLow") is greater than the value in the upper operating point ("MasterVelocityHigh")
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000005	The value of the "LowLimit" input is lower than the minimum allowed (-95.0)
INPUT_RANGE_ERROR(16#0006)	16#00000A01	16#00000006	The value of the inputs "PGainLowVel" or "PGainHighVel" is lower than or equal to 0
DEVICE_ERROR (16#0008)	16#00000A02	16#00000007	No power is added to the drive specified in "Slave". Closed-loop tension control is therefore not possible
DEVICE_ERROR (16#0008)	16#00000A02	16#00000008	The drive specified in the "Slave" is not in a synchronous operation mode
INPUT_INVALID_ERROR(16#0001)	16#00000A03	16#00000000	A "SlavePreCtrl" input has an invalid value
INPUT_INVALID_ERROR(16#0001)	16#00000A04	16#00000000	"Circumference" <= 0.0
INPUT_INVALID_ERROR(16#0001)	16#00000A04	16#00000001	"Distance" <= 0.0

Fig.13-21: Error codes of the MB_TensionControlLoadCellType02 function block

13.3 Function Blocks for the Usage of the "Decoupling Network"

13.3.1 MB_TensionDecouplingNetworkType01

Brief Description The tension controller controls the tension in a web section to the desired command value (MB_TensionControlLoadCellType01/02) depending measured value of a load cell. The decoupling network MB_TensionDecoupling-

NetworkType01 can be used to forward a control signal change of any web tensile stress controller of the continuous production facility to all axes required for decoupling. These decoupled precontrols impede a temporary and stationary change of all web tensile stresses in the machine, except the ones to be controlled locally.

Target system	Library
IndraMotion MLC 12VRS	ML_TechTensionAdvanced.compiled-library

Fig. 13-22: Reference table of the MB_TensionDecouplingNetworkType01 function block

The usage of the function block is subject to license. Details on the licensing requirement are described in the introduction of the documentation about the library (see page 571).

Interface description

Fig. 13-23: MB_TensionDecouplingNetworkType01 function block

I/O type	Name	Data type	Description
VAR_IN_OUT	ControllerValue	MB_TENSION_DECOUPLING_CONTROLLER_VALUE	Structure of the controller outputs (CorrectionValue, AdjCorrectionValue, AdditivePosInc and AdditivePosDeg)
	AxisValue	MB_TENSION_DECOUPLING_AXIS_VALUE	Structure of the axis inputs (CorrectionValue, AdditivePosInc and AdditivePosDeg)
	AxisRef	ARRAY [1..8] OF AXIS_REF	Control axes of the system (without "UnControlledAxis")
VAR_INPUT	Enable	BOOL	Enabling the function block (once, edge-controlled)
	StandstillEnable	BOOL	Closed-loop standstill control enabled if the input is set
	DisableCyclicWrites	BOOL	If this input is set, the cyclic data is not written directly into the optional cyclic data container, but only displayed at the output
	UnControlledAxis	INT	Axis in the system not used for the closed-loop control and decoupling of the web tensile stresses
VAR_OUTPUT	InOperation	BOOL	The decoupling network operates and the outputs are valid
	Error	BOOL	Processing completed with error
	ErrorID	ERROR_CODE	Diagnostic description in case of error
	ErrorIdent	ERROR_STRUCT	Detailed diagnostics

Fig. 13-24: Interface description of the MB_TensionDecouplingNetworkType01

ML_TechTensionAdvanced.library

Structure description The following table lists the structure of the MB_TENSION_DECOUPLING_CONTROLLER_VALUE.

Name	Type	Min. value	Max. value	Description
ControllerCorrectionValue [1]	REAL			CorrectionValue of the first web tensile stress controller
ControllerCorrectionValue [2]	REAL			CorrectionValue of the second web tensile stress controller
ControllerCorrectionValue [3]	REAL			CorrectionValue of the third web tensile stress controller
ControllerCorrectionValue [4]	REAL			CorrectionValue of the forth web tensile stress controller
ControllerCorrectionValue [5]	REAL			CorrectionValue of the fifth web tensile stress controller
ControllerCorrectionValue [6]	REAL			CorrectionValue of the sixth web tensile stress controller
ControllerCorrectionValue [7]	REAL			CorrectionValue of the seventh web tensile stress controller
ControllerCorrectionValue [8]	REAL			CorrectionValue of the eighths web tensile stress controller
ControllerAdjCorrectionValue [1]	REAL			AdjCorrectionValue of the first web tensile stress controller
ControllerAdjCorrectionValue [2]	REAL			AdjCorrectionValue of the second web tensile stress controller
ControllerAdjCorrectionValue [3]	REAL			AdjCorrectionValue of the third web tensile stress controller
ControllerAdjCorrectionValue [4]	REAL			AdjCorrectionValue of the forth web tensile stress controller
ControllerAdjCorrectionValue [5]	REAL			AdjCorrectionValue of the fifth web tensile stress controller
ControllerAdjCorrectionValue [6]	REAL			AdjCorrectionValue of the sixth web tensile stress controller
ControllerAdjCorrectionValue [7]	REAL			AdjCorrectionValue of the seventh web tensile stress controller
ControllerAdjCorrectionValue [8]	REAL			AdjCorrectionValue of the eighth web tensile stress controller
ControllerAdditivePosInc [1]	DINT			AdditivePosInc of the first web tensile stress controller
ControllerAdditivePosInc [2]	DINT			AdditivePosInc of the second web tensile stress controller
ControllerAdditivePosInc [3]	DINT			AdditivePosInc of the third web tensile stress controller
ControllerAdditivePosInc [4]	DINT			AdditivePosInc of the forth web tensile stress controller
ControllerAdditivePosInc [5]	DINT			AdditivePosInc of the fifth web tensile stress controller
ControllerAdditivePosInc [6]	DINT			AdditivePosInc of the sixth web tensile stress controller
ControllerAdditivePosInc [7]	DINT			AdditivePosInc of the seventh web tensile stress controller
ControllerAdditivePosInc [8]	DINT			AdditivePosInc of the eighth web tensile stress controller
ControllerAdditivePosDeg [1]	REAL			AdditivePosDeg of the first web tensile stress controller

Name	Type	Min. value	Max. value	Description
ControllerAdditivePosDeg [2]	REAL			AdditivePosDeg of the second web tensile stress controller
ControllerAdditivePosDeg [3]	REAL			AdditivePosDeg of the third web tensile stress controller
ControllerAdditivePosDeg [4]	REAL			AdditivePosDeg of the forth web tensile stress controller
ControllerAdditivePosDeg [5]	REAL			AdditivePosDeg of the fifth web tensile stress controller
ControllerAdditivePosDeg [6]	REAL			AdditivePosDeg of the sixth web tensile stress controller
ControllerAdditivePosDeg [7]	REAL			AdditivePosDeg of the seventh web tensile stress controller
ControllerAdditivePosDeg [8]	REAL			AdditivePosDeg of the eighth web tensile stress controller

Fig. 13-25: Structure description of the MB_TENSION_DECOUPLING_CONTROLLER_VALUE

Structure description

The following table lists the structure of the MB_TENSION_DECOUPLING_AXIS_VALUE.

Name	Type	Min. value	Max. value	Description
AxisCorrectionValue [1]	REAL			CorrectionValue of the first axis
AxisCorrectionValue [2]	REAL			CorrectionValue of the second axis
AxisCorrectionValue [3]	REAL			CorrectionValue of the third axis
AxisCorrectionValue [4]	REAL			CorrectionValue of the forth axis
AxisCorrectionValue [5]	REAL			CorrectionValue of the fifth axis
AxisCorrectionValue [6]	REAL			CorrectionValue of the sixth axis
AxisCorrectionValue [7]	REAL			CorrectionValue of the seventh axis
AxisCorrectionValue [8]	REAL			CorrectionValue of the eighth axis
AxisAdditivePosInc [1]	DINT			AdditivePosInc of the first axis
AxisAdditivePosInc [2]	DINT			AdditivePosInc of the second axis
AxisAdditivePosInc [3]	DINT			AdditivePosInc of the third axis
AxisAdditivePosInc [4]	DINT			AdditivePosInc of the forth axis
AxisAdditivePosInc [5]	DINT			AdditivePosInc of the fifth axis
AxisAdditivePosInc [6]	DINT			AdditivePosInc of the sixth axis
AxisAdditivePosInc [7]	DINT			AdditivePosInc of the seventh axis
AxisAdditivePosInc [8]	DINT			AdditivePosInc of the eighth axis
AxisAdditivePosDeg [1]	REAL			AdditivePosDeg of the first axis
AxisAdditivePosDeg [2]	REAL			AdditivePosDeg of the second axis
AxisAdditivePosDeg [3]	REAL			AdditivePosDeg of the third axis
AxisAdditivePosDeg [4]	REAL			AdditivePosDeg of the forth axis
AxisAdditivePosDeg [5]	REAL			AdditivePosDeg of the fifth axis

ML_TechTensionAdvanced.library

Name	Type	Min. value	Max. value	Description
AxisAdditivePosDeg [6]	REAL			AdditivePosDeg of the sixth axis
AxisAdditivePosDeg [7]	REAL			AdditivePosDeg of the seventh axis
AxisAdditivePosDeg [8]	REAL			AdditivePosDeg of the eights axis

Fig.13-26: Structure description of the MB_TENSION_DECOUPLING_AXIS_VALUE

Min./max. and default values

The following table lists the min./max. and default values of the function block inputs.

Name	Type	Min. value	Max. value	Default value	Effective
ControllerValue	MB_TENSION_DECOUPLING_CONTROLLER_ VALUE			0	Continuous
AxisValue	MB_TENSION_DECOUPLING_AXIS_ VALUE			0	Continuous
AxisRef	ARRAY [1..8] OF AXIS_REF			0	Rising edge at "Enable"
Enable	BOOL			FALSE	Continuous
StandstillEnable	BOOL			FALSE	Continuous
DisableCyclicWrites	BOOL			FALSE	Rising edge at "Enable"
UnControlledAxis	INT	1	9	1	Rising edge at "Enable"

Fig.13-27: Input behavior of the MB_TensionDecouplingNetworkType01

Functional Description

The following explains the individual functions of the function block inputs in detail. In general, all outputs are continuously evaluated if the "Enable" input is set. That means that input changes become immediately effective without toggling that input again. An exception of this rule are the inputs "UnControlledAxis", "DisableCyclicWrites" and the input/output array "AxisRef" which can only be evaluated at a rising edge at the "Enable" input. "0" or FALSE are assigned internally to all input values whose inputs are not connected. An exception of this rule is the "UnControlledAxis" input. If it is not assigned, it is set to "1". As for the web tensile stress controller "MB_Tension_ControlLoadCellType02", the control variable is written to the gear ratio fine adjustment in normal mode. Depending on the axis type, it is written directly on the respective parameter (P-0-0694 at interpolation in the drive or A-0-2605 at interpolation on the control), if "DisableCyclicWrites" is disabled. Additionally, the control signals of all axes are always provided via the "MB_TENSION_DECOUPLING_AXIS_VALUE" structure at the output of the decoupling network. If there are uniform axis types, it is sufficient to add only the respective standstill signal ("AdditivePosInc" at interpolation in the drive or "AdditivePosDeg" at interpolation in the control) from the controllers to the decoupling network. If there are mixed axis types (interpolation in the drive and interpolation in the control), it is absolutely necessary to add both standstill signals (AdditivePosInc and AdditivePosDeg) to the decoupling network for all controllers. Since the outputs "CorrectionValue", "AdjCorrectionValue", "AdditivePosInc" and "AdditivePosDeg" of the path tension controller may not be written directly on the axes anymore when using the decoupling network, it is absolutely necessary to disable the cyclic writing of values in all participating web tensile

stress controllers (DisableCyclicWrites:=TRUE). The complete mode of operation of the decoupling network is shown in the following diagrams.

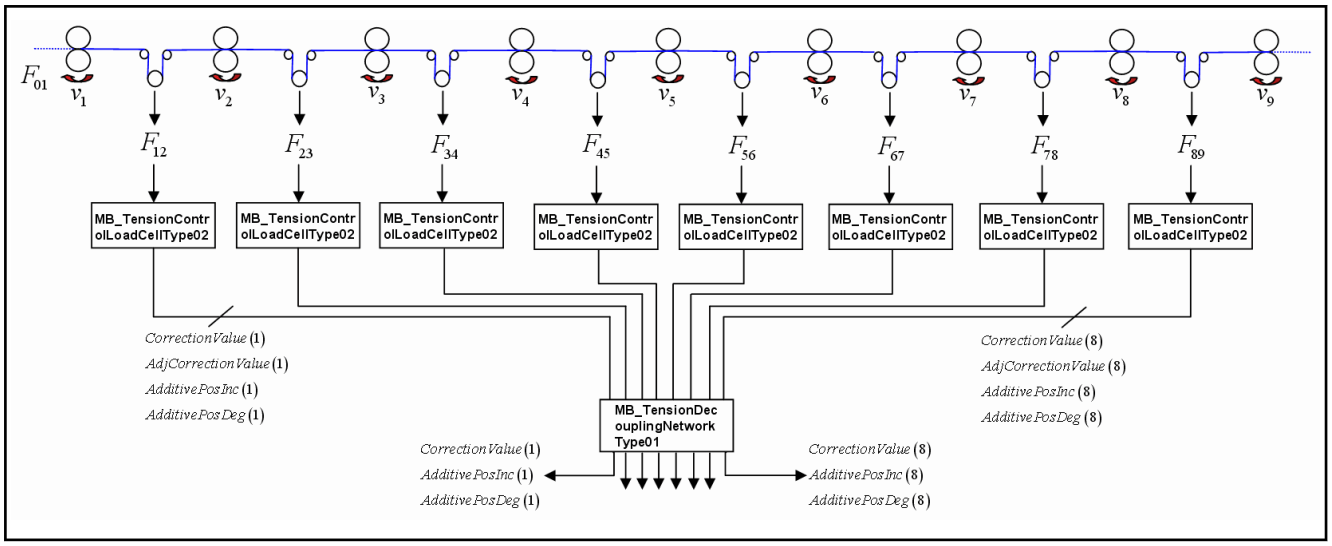


Fig. 13-28: General diagram of the MB_TensionDecouplingNetworkType01 function block

It can be seen that each web section to be controlled has to be equipped with a web tensile stress controller MB_TensionControlLoadCellType02. It is to be considered that all web tension controllers have to operate in positive polarity (NegPolarity:=FALSE). The controller outputs "CorrectionValue", "AdjCorrectionValue" , "AdditivePosInc" and "AdditivePosDeg" are then added to the decoupling network MB_TensionDecouplingNetworkType01. In the example shown, the web tensile stress control should be controlled independently from each other in eight web sections. At least eight control elements are required. Since n web sections to be controlled are always limited by n+1 axes, any axis can be selected that is exclusively operated with master axis velocity and does not have to execute additional control movements ("CorrectionValue" , "AdditivePosInc" or "AdditivePosDeg"). The following figure shows the first axis of the system to be excluded for the decoupled closed-loop control for the web tensile stresses (UnControlledAxis:=1). This decoupling variant also corresponds to the basic function of the MB_TensionControlLoadCellType02 if only the first web tensile stress controller is active.

ML_TechTensionAdvanced.library

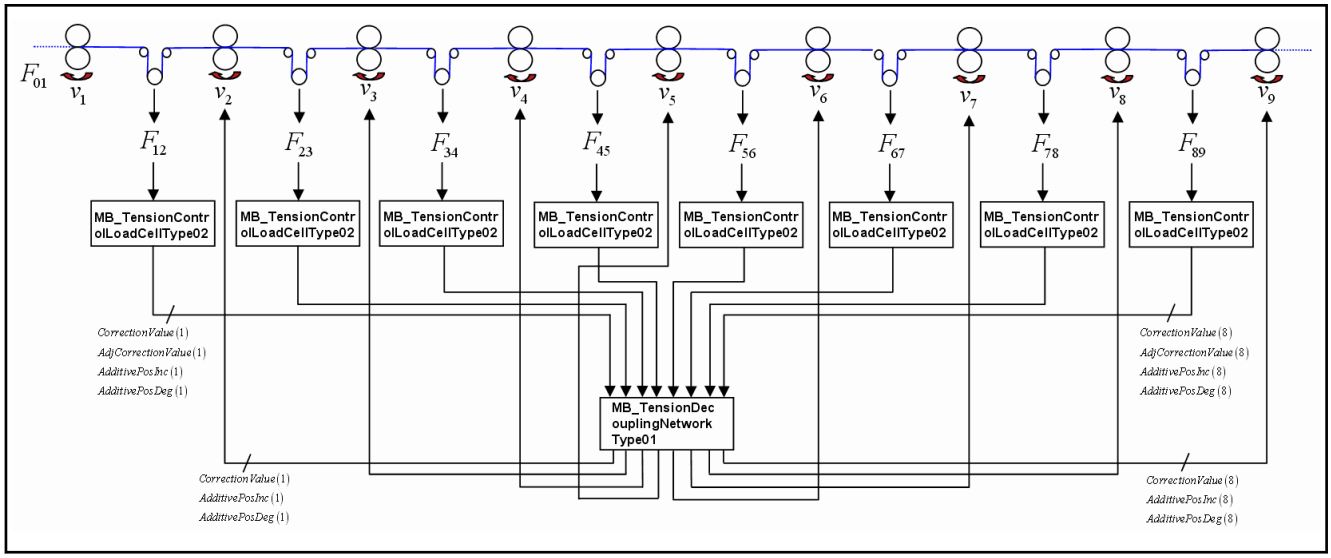


Fig. 13-29: Decoupling web tensile stresses by precontrol exclusively in web direction (leading nips are not required for the closed-loop control and decoupling)

If this variant is selected for a system with less than eight web sections to be controlled, the following web tensile stress controllers in web direction remain unconsidered.

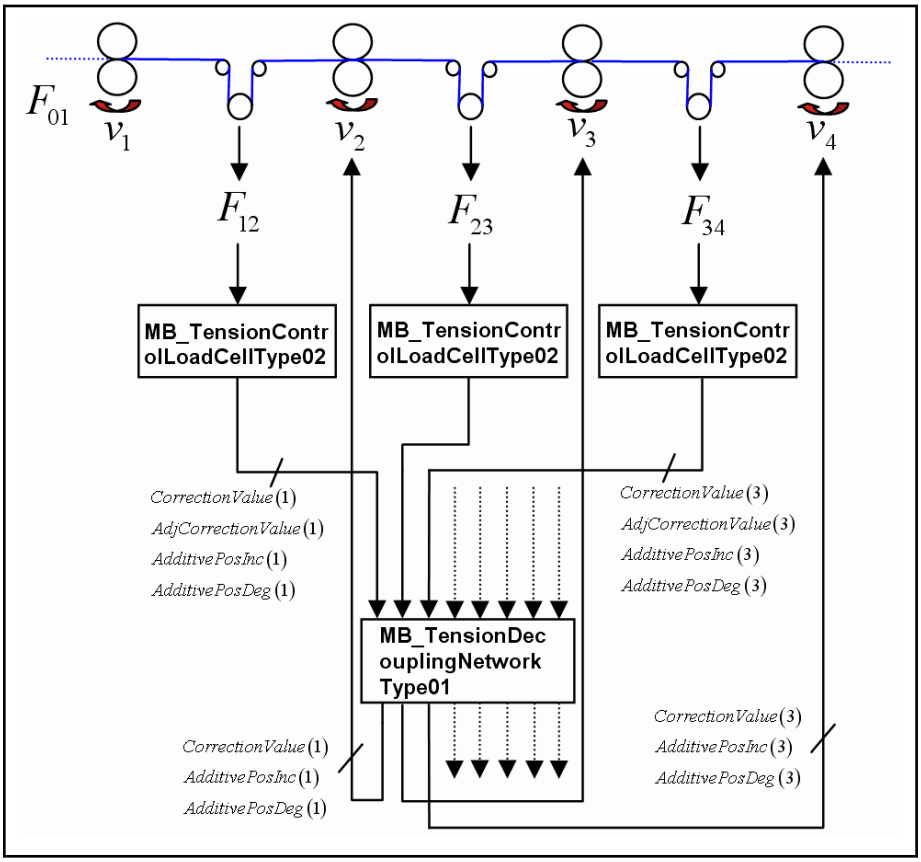


Fig. 13-30: Decoupling web tensile stresses by precontrol exclusively in web direction (representation for three web sections to be controlled)

Another option is shown in the following. The last axis of the system is excluded for the closed-loop control of the web tensile stresses (UnControlledAxis:=9).

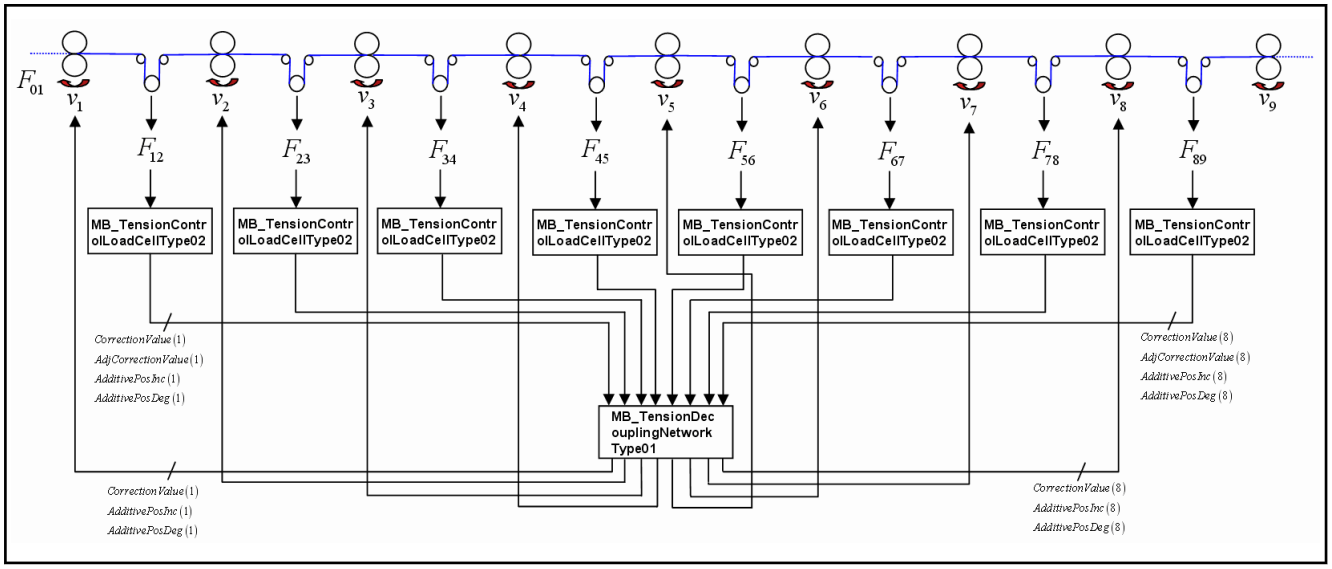


Fig. 13-31: Decoupling of web tensile stresses by precontrol exclusively opposite web direction (following nips are not required for the closed-loop control and decoupling)

If this variant is selected for a system with less than eight web sections to be controlled, the leading web tensile stress controllers opposite web direction remain unconsidered.

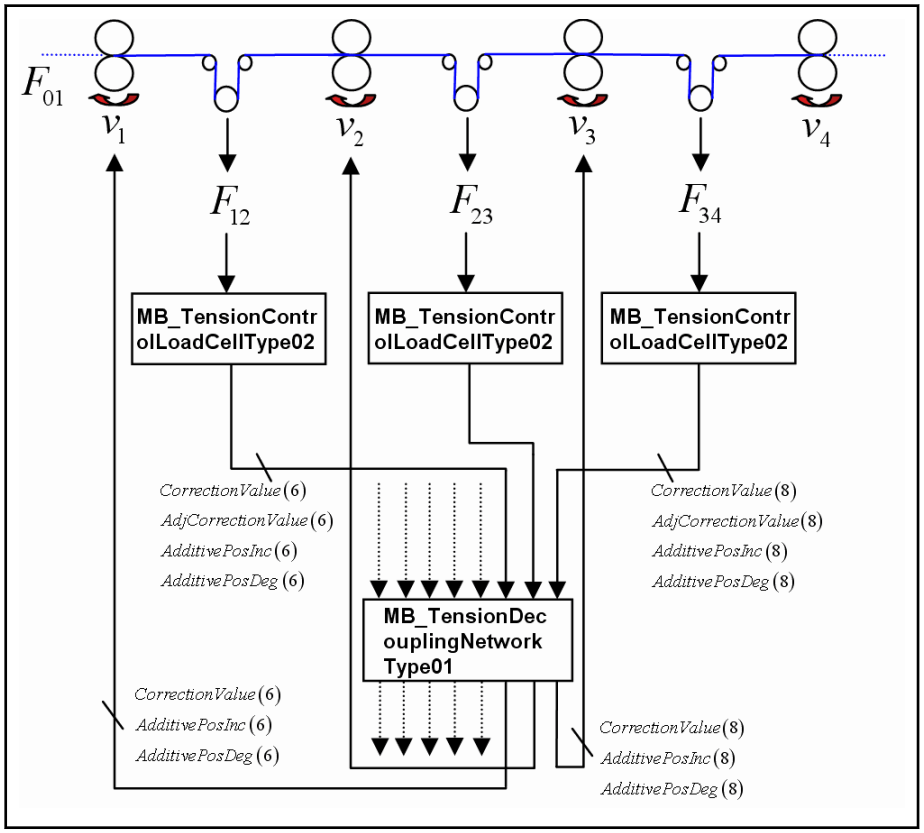


Fig. 13-32: Decoupling web tensile stresses by precontrol exclusively opposite web direction (representation for three web sections to be controlled)

Since any axis of the system can be excluded from controlling the web tensile stresses, another option is to exclude the centered axis. This corresponds to the behavior "UnControlledAxis":=2-8 and is exemplarily shown for the fifth axis (UnControlledAxis:=5).

ML_TechTensionAdvanced.library

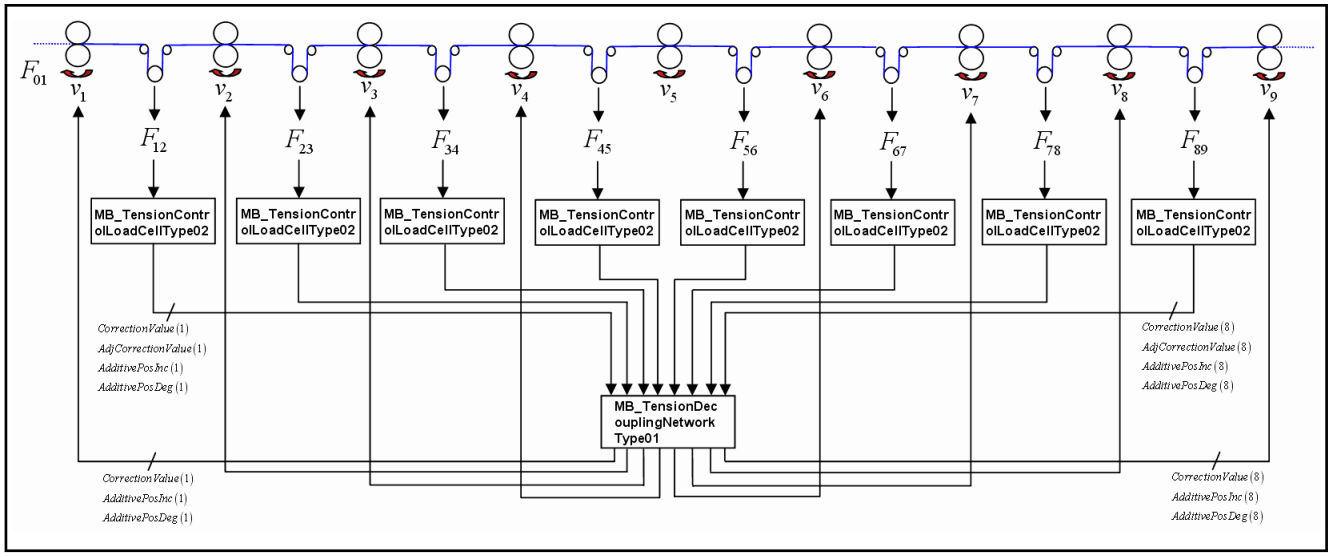


Fig. 13-33: Decoupling of web tensile stresses by precontrol in and opposite web direction (centered nips in figure "v_5" are not required for the closed-loop control and decoupling)

If this variant is selected for a system with less than eight web sections to be controlled, the leading and/or following web tensile stress controllers in web direction remain unconsidered as in the two previously presented variants. If the axis not to be controlled is selected according to the real system in web direction, it is only required to remain the following real non-existing web tensile stress controllers unconsidered.

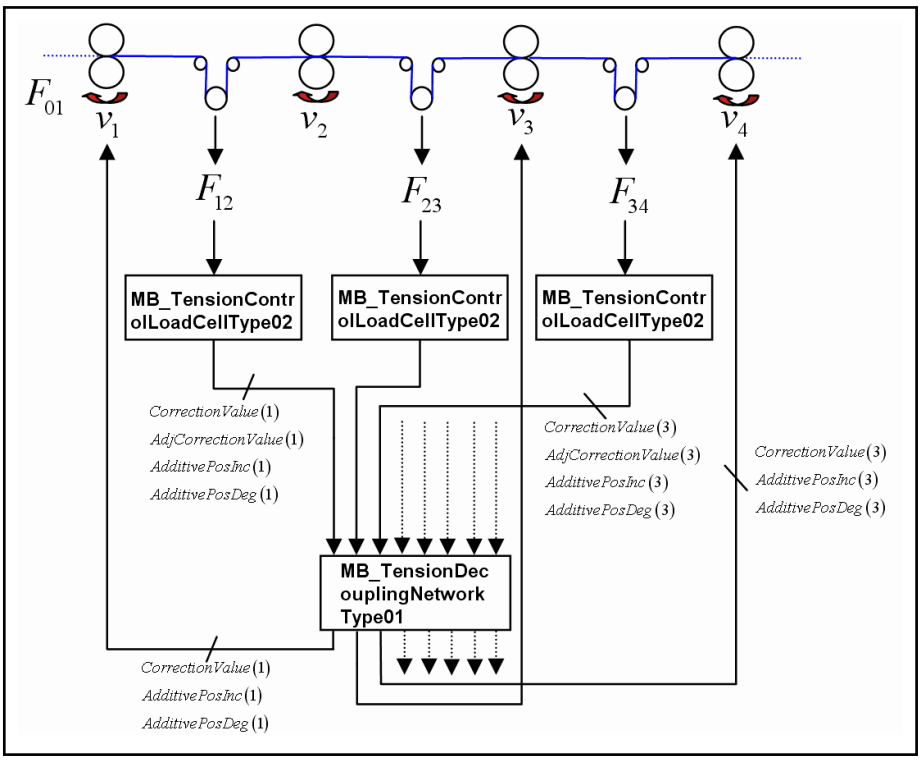


Fig. 13-34: Decoupling web tensile stresses by precontrol in and opposite web direction (representation for three web sections to be controlled)

Apart from these three variants of decoupling at which always one axis is not required for closed-loop control and decoupling, the decoupling network can

nearly be used as desired. Exemplarily, a system with up to 15 web sections to be controlled and the request for a minimum control effort is shown.

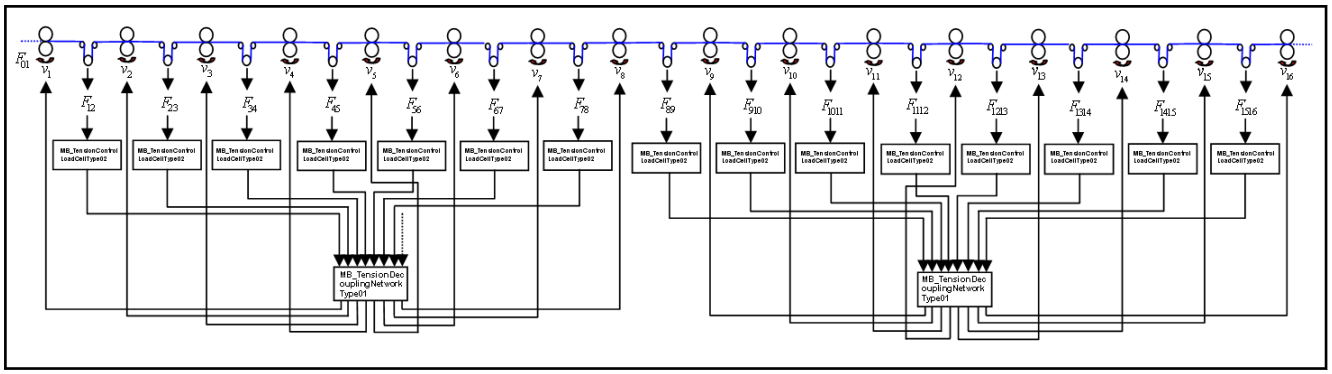


Fig. 13-35: Expansion of the decoupling network to 15 web sections to be controlled at an additional request to a minimum control effort

This figure shows that two function blocks MB_TensionDecouplingNetwork-Type01 are used and that a control signal change of a web tensile stress controller of a maximum of eight (controlled sections1/2) causes control movements. Furthermore, it can be seen that an upstream decoupling is used in the leading decoupling network (UnControlledAxis:=9) and a downstream decoupling (UnControlledAxis:=1) in the following decoupling network. Only the last web tensile stress controller of the leading decoupling network was not assigned. Since the last web section of the leading decoupling network is completely decoupled from all web sections in the front, this combination of the two decoupling networks can be used without cross communication or cross precontrol being required. If two or more decoupling networks are used, these can be combined to any number of variants. If another variant with multiple decoupling networks is reasonable for a special system, the inputs and outputs of the decoupling networks can be externally combined. Depending on the desired behavior, it might be additionally necessary to disable cyclic writing (DisableCyclicWrites:=TRUE). An example for these unlimited number of variants are systems with more than eight web sections und the requirements that the first, last or any axis may not be controlled and/or precontrolled. It is then reasonable to combine one or several outputs of a decoupling network with one or several inputs of another decoupling network. Detailed and advanced knowledge is required for these implementations.

Commissioning

If "DisableCyclicWrites" is inactive, both the P-0-0694 (Gear ratio fine adjustment) and P-0-0692 (Additive master axis position) parameters must be applied in the cyclic command value channel of the slave axis. At a rising edge at "Enable", the cyclic data channels are checked. If one of the two parameters is not found, a respective error message is returned at the error outputs of the function block. If "DisableCyclicWrites" is enabled, the configuration of the cyclic command value data channels is not checked.

Enable

The "Enable" input activates the function block.

StandstillEnable

The closed-loop tension control is enabled by setting the "StandstillEnable“ input. Thus, the inputs "AdditivePosInc“ and "AdditivePosDeg“ are evaluated and used to calculate and output the complete control variables "AdditivePosInc" and "AdditivePosDeg" of the individual axes.

DisableCyclicWrites

The "DisableCyclicWrites" input determines whether the values calculated by the function block are to be written directly via the cyclic channel (FALSE) or whether they are only available at the function block output (TRUE). With "DisableCyclicWrites":=TRUE, it is possible to combine multiple decoupling networks with each other by manually linking the outputs of multiple decoupling networks.

ML_TechTensionAdvanced.library

UnControlledAxis The "UnControlledAxis" input determines the decoupling variant as well as the position of the axis not to be controlled and precontrolled. By default, this input is set to 1. That means for the decoupling network that a precontrol should exclusively be executed in web direction (downstream decoupling). If this input is set to 9, the last axis of the system for the closed-loop control and decoupling is excluded (upstream decoupling). In this case, a precontrol is only executed in opposite web direction. Furthermore, any axis of the system can be excluded for the web tensile stress control and decoupling (UnControlledAxis:=2-8).

Error handling The function block generates the following error messages in Additional1/Additional2 for the table **F_RELATED_TABLE**, 16#0170:

ErrorID	Additional1	Additional2	Description
ACCESS_ERROR, 16#0004	16#00000A00	16#00000001	The parameter P-0-0694 is not parameterized in the optional cyclic telegram of the precontrolled axis
ACCESS_ERROR, 16#0004	16#00000A00	16#00000002	The parameter P-0-0692 is not parameterized in the optional cyclic telegram of the precontrolled axis
INPUT_RANGE_ERROR, 16#0006	16#00000A01	16#00000007	The "UnControlledAxis" input is not within the valid range
DEVICE_ERROR, 16#0008	16#00000A02	16#00000007	No power is added to a drive. Closed-loop tension control is therefore not possible
DEVICE_ERROR, 16#0008	16#00000A02	16#00000008	A drive is a not in a synchronous operation mode

Fig.13-36: Error codes of the MB_TensionDecouplingNetworkType01

14 Service and Support

The Bosch Rexroth service helpdesk at out headquarters in Lohr, Germany, and our worldwide service provide 24/7 support and assistance.

	Helpdesk	Worldwide service hotline
Phone	**+49 (0) 9352 40 50 60**	Outside Germany, please contact your sales/service office first. For hotline numbers, refer to the sales office addresses on the internet.
Fax	**+49 (0) 9352 40 49 41**	
E-mail	service.svc@boschrexroth.de	
Internet	For additional information on service, maintenance (e.g. delivery addresses) and training, please go to http://www.boschrexroth.com	

Required information

Please provide the following information:

- Detailed description of malfunction and circumstances leading to the malfunction
- Type plate name of the affected products, in particular type codes and serial numbers
- Phone and fax numbers as well as your e-mail address

Index

Index

F

F

Index

F

G

H

H

I

L

L

M

M

Index

M

M

M

N

O

P

P

R

S

Index

S

T

U

W

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 (0)93 52-40-0
Fax +49 (0)93 52-48 85
www.boschrexroth.com/electrics

R911333868

Printed in Germany
DOK-MLC***-TF*LIB**V12-LI02-EN-P