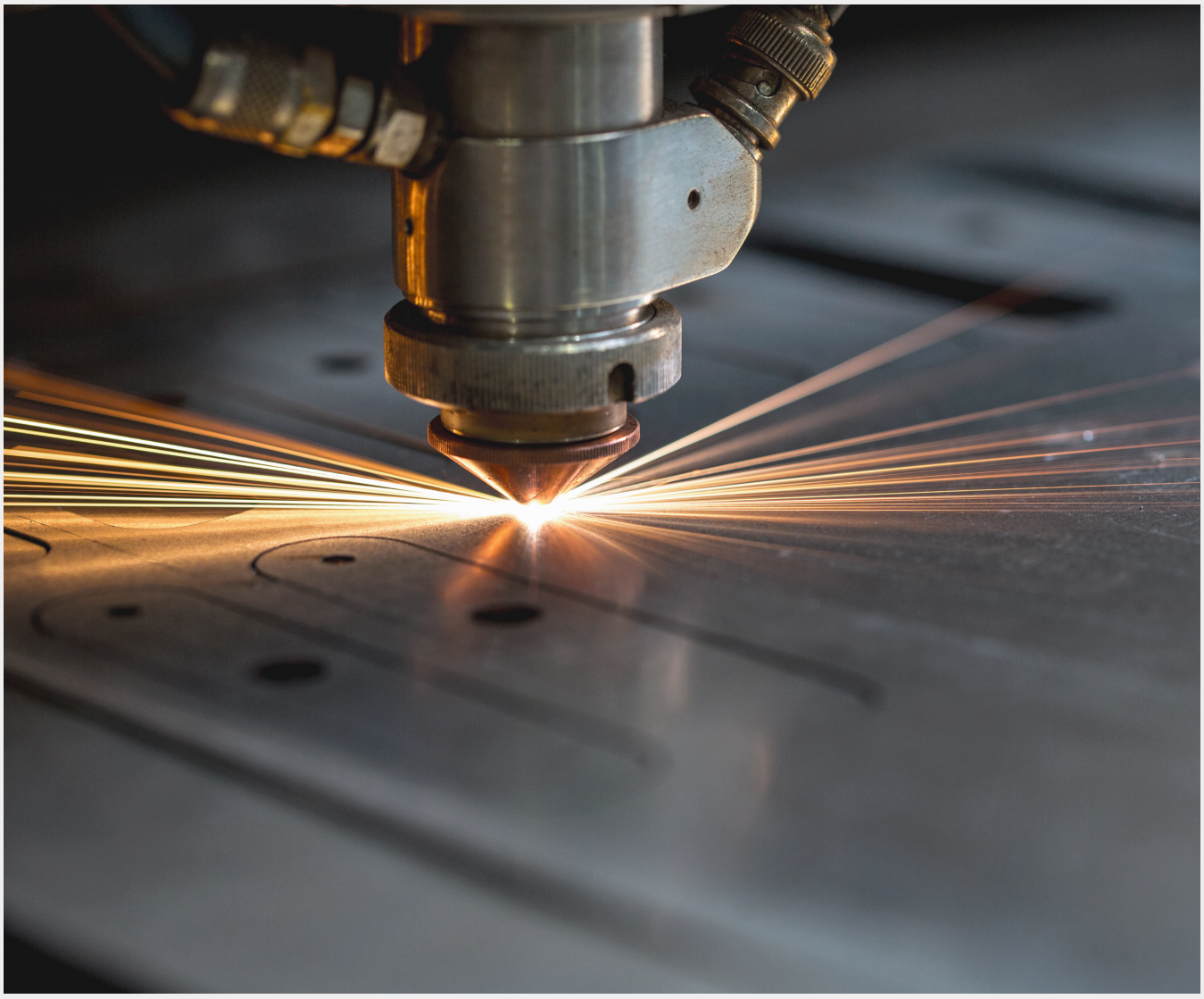
rexroth
A Bosch Company

MTX 15VRS

Shape Cutting Technology

Commissioning Manual
R911399147

Edition 06



Title	MTX 15VRS Shape Cutting Technology
Type of Documentation	Commissioning Manual
Document Typecode	DOK-MTX***-TECHCUT*V15-CO06-EN-P
Internal File Reference	RS-8b1dbb9153ed03aa0a347e8633e67190-6-en-US-3

Change record	Edition 06, 2021-09 See chapter 1.1 "Change record" on page 1
Copyright	© Bosch Rexroth AG 2021 All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.
Liability	The specified data is intended for product description purposes only and shall not be deemed to be a guaranteed characteristic unless expressly stipulated in the contract. All rights are reserved with respect to the content of this documentation and the availability of the product.
Editorial Department	System Development MTX, WeZh (MiSc/PiaSt)

Table of Contents

Page

1 About this documentation ... 1
1.1 Change record ... 1
1.2 Purpose ... 1
1.3 Overview on target groups and product phases ... 1
1.4 Required and supplementing documentation MTX ... 2
1.4.1 Selection/compilation ... 2
1.4.2 Configuration ... 2
1.4.3 Commissioning ... 3
1.4.4 Operation ... 4
1.4.5 OEM Engineering ... 5
1.4.6 AddOns ... 5
1.5 Information representation ... 6
1.5.1 Terms and abbreviations ... 6
1.6 Customer feedback ... 6

2 Important instructions on use ... 7
2.1 Intended use ... 7
2.1.1 Introduction ... 7
2.1.2 Areas of use and application ... 7
2.2 Unintended use ... 8

3 Introduction to the shape cutting technology ... 9

4 Shape cutting technology project template ... 11
4.1 Introduction and requirements ... 11
4.2 Creating an IndraWorks project from the project template ... 12
4.3 Project structure ... 12
4.4 Commissioning the NC ... 14
4.5 Commissioning of the PLC ... 18
4.6 Function test ... 19
4.7 Sample program ... 21
4.8 M-auxiliary functions overview ... 23

5 Fast data exchange ... 25
5.1 Introduction and overview ... 25
5.2 Main_FA program ... 25
5.3 Fast inputs and outputs ... 26
5.4 Configuring the inputs and outputs of the FA ... 30
5.5 Adding a new output value to the system data ... 31
5.6 Error analysis with the NC analyzer ... 33

Page

6 Height control ... 37
6.1 Overview ... 37
6.1.1 Characteristics of the height control ... 37
6.1.2 Control loop ... 38
6.1.3 Special features of the height control ... 38
6.1.4 Sensor selection ... 39
6.2 Project module ... 40
6.2.1 Height control project module ... 40
6.2.2 Retract project module ... 42
6.3 Commissioning the height control ... 43
6.3.1 Creating test program ... 43
6.3.2 Controller settings ... 44
6.4 Control behavior of the height control ... 48
6.5 Setting parameters of the height control ... 49
6.6 NC commands for the height control ... 50
6.6.1 Overview ... 50
6.6.2 Sensor calibration: SENSORCALIB ... 50
6.6.3 Structure of the sensor calibration file ... 51
6.6.4 Enabling/disabling the height control HEIGHTCTRL ... 51
6.6.5 Sensor calibration: PREC16P_CALIB ... 52
6.6.6 PlateSearch ... 53
6.7 Troubleshooting ... 54

7 FlyCut ... 57
7.1 Introduction ... 57
7.2 Variants ... 57
7.3 FlyCut project module ... 59
7.4 Parameterization ... 61
7.5 Commissioning ... 62

8 LeapFrog ... 65
8.1 Introduction ... 65
8.2 Project module LeapFrog ... 66
8.3 Parameterization ... 68
8.4 NC commands ... 69
8.4.1 LEAPFROG ... 69
8.4.2 ResetLeapFrog ... 69
8.4.3 M187 and CheckLeapFrogPara ... 69
8.5 Commissioning ... 70
8.6 Example ... 73

9 Laser power control ... 75
9.1 Introduction ... 75
9.2 Laser power control project module ... 76
9.3 Parameterization ... 78

Page

9.4 NC commands for the laser power control ... 81
9.4.1 Overview ... 81
9.4.2 Laser control: CUT ... 81
9.4.3 Piercing: PIERCING ... 82

10 Gas control ... 85
10.1 Introduction ... 85
10.2 Project module gas control ... 86
10.3 Parameterization ... 87
10.4 Gas control: GAS ... 87

11 Focus control ... 89
11.1 Introduction ... 89
11.2 Project module focus control ... 89
11.3 Parameterization ... 90
11.4 Focus control: FOCUS ... 90

12 Axis group (Gantry) ... 91
12.1 Overview on Gantry ... 91
12.2 Control behavior ... 91
12.3 Commissioning ... 93
12.4 PLC plug-in ... 94

13 Cutting width compensation (kerf) ... 97
13.1 Introduction ... 97
13.2 D-corrections ... 98

14 Re-enter and Retrace functionality ... 101
14.1 Introduction ... 101
14.2 Re-entry functions of the NC program ... 101
14.2.1 Overview ... 101
14.2.2 NC block search ... 101
14.2.3 Leaving and returning to contour ... 102
14.2.4 Block pre-run ... 102
14.2.5 Re-entry ... 102
14.3 Retrace ... 103

15 Technology tables ... 105
15.1 Introduction ... 105
15.2 Structure and storage location ... 105
15.3 Loading technology tables: TECHTABLE ... 108
15.4 Overview of SD variables ... 108

Page

16 MT_Technology ... 115
16.1 Introduction and overview ... 115
16.2 Height control ... 116
16.2.1 Introduction and overview ... 116
16.2.2 MT_HeightRegulationType01 ... 117
16.2.3 MT_DistanceSensorType01 ... 121
16.2.4 MT_DistanceSensorType02 ... 124
16.2.5 MT_HCSimType01 ... 126
16.2.6 MT_HCSimType02 ... 128
16.2.7 MT_PlateSearchType01 ... 130
16.2.8 MT_PlateSearchType02 ... 133
16.2.9 EN_HC_STATE_t ... 136
16.2.10 EN_HC_USRMODE_t ... 137
16.2.11 MT_RaiseAxis ... 138
16.3 LeapFrog ... 140
16.3.1 Introduction and overview ... 140
16.3.2 MT_LeapFrogType01 ... 140
16.4 Power control ... 145
16.4.1 Introduction and overview ... 145
16.4.2 MT_GetPowerCtrlMode ... 145
16.4.3 MT_LaserPowerControlType01 ... 146
16.4.4 MT_LaserPowerControlType02 ... 149
16.4.5 MT_LaserPowerControlType03 ... 151
16.5 Gas control ... 154
16.5.1 MT_GasControlType01 ... 154
16.6 Focus control ... 156
16.6.1 MT_FocusControlType01 ... 156
16.7 FlyCut ... 158
16.7.1 Introduction and overview ... 158
16.7.2 MT_FlyCutType01 ... 158
16.7.3 MT_FlyCutType02 ... 161
16.8 Auxiliary functions for oversampling ... 165
16.8.1 Introduction and overview ... 165
16.8.2 MT_Oversampling_AND ... 165
16.8.3 MT_Oversampling_OR ... 167

17 Service and support ... 169

Index ... 171

1 About this documentation

1.1 Change record

Editions of this documentation

Edition	Release date	Note
01	2019-07	First edition
02	2019-10	Chapter "Height control" updated and supplemented General updates and supplements implemented.
03	2020-07	Chapter "Height control" updated and supplemented Update of cycle commands General updates and supplements implemented.
04	2021-01	New chapter about gas actuation and focus control Chapter "Height control" updated and supplemented Updates in chapter "Laser power control" Updates and additions in chapter "MT technology" (MT_HeightRegulationType01, MT_PlateSearchType02, MT_RaiseAxis)
05	2021-03	Updates in chapter "LeapFrog" Updates in chapter "Technology tables"
06	2021-09	Modifications in chapter 7.3 "FlyCut project module" on page 59 Changes in chapter 8.5 "Commissioning" on page 70 New: chapter 16.2.10 "EN_HC_USRMODE_t" on page 137

Tab. 1-1: Change record

1.2 Purpose

This documentation is intended for users applying the project template for shape cutting technology functions and thus commissioning an MTX control. In addition to the general overview on the operating principle, the configuration and the commissioning on the NC and PLC as well as the diagnostic options are explained.

1.3 Overview on target groups and product phases

The following graphic refers to the bordered activities, product phases and target groups of the present documentation.

In the product phase "Engineering", the target group "programmer" can execute the activities "programming and configuring" using this documentation.

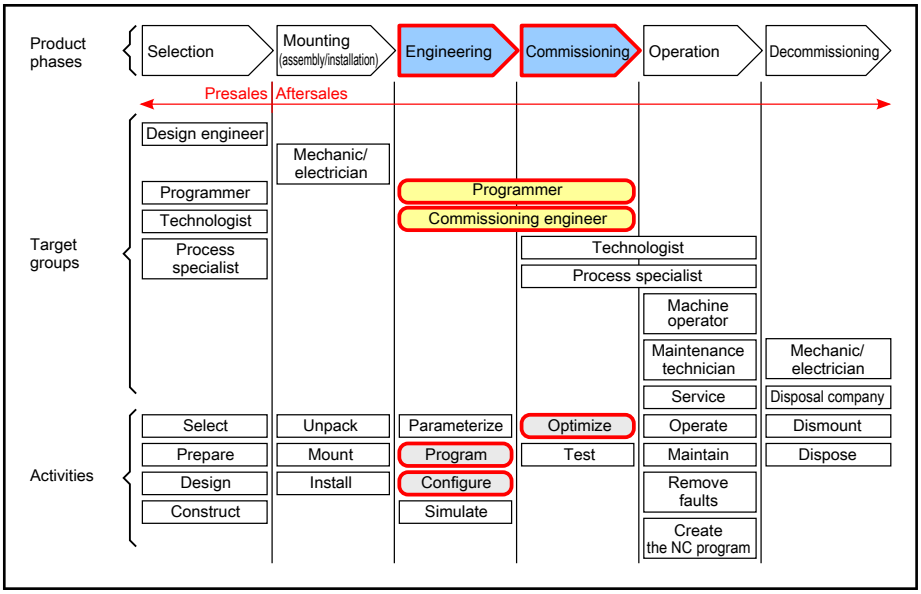


Fig. 1-1: Assigning this documentation to the target groups, product phases and target group activities

1.4 Required and supplementing documentation MTX

1.4.1 Selection/compilation

Documentation titles with type codes and part numbers

MTX 15VRS System Description DOK-MTX***-SYS*DES*V15-PRRS-EN-P, R911397742 This documentation describes the MTX control. It includes the designs, technical data, interfaces as well as the configuration of the control components.
MTX 15VRS SafeLogic System Overview DOK-MTX***-SL**SYS*V15-PRRS-EN-P, R911398637 This documentation describes the use of the safety control SafeLogic in the MTX.
IndraControl XM42 Controls DOK-CONTRL-XM4X*CTRL**-ITRS-EN-P, R911345566 This documentation describes the IndraControl XM42 controls.

RS Corresponding edition

Tab. 1-2: MTX documentation overview - Selecting/composing

1.4.2 Configuration

Documentation titles with type codes and part numbers

MTX 15VRS Machine Parameters DOK-MTX***-MA*PAR**V15-RERS-EN-P, R911400176 This documentation describes structure and the adjustment of available parameters of the MTX. It also includes the functions of the NC configurator and its operation.
MTX 15VRS PLC Interface DOK-MTX***-PLC*INT*V15-PRRS-EN-P, R911400172 This documentation describes the interface signals and the program function blocks for the integrated PLC.

MTX 15VRS Functional Description - Basics
DOK-MTX***-NC*F*BA*V15-RERS-EN-P, R911401401
This documentation describes the basic functions of the MTX. The basic commissioning steps and the control functions are provided as description and handling instruction.
MTX 15VRS Functional Description - Extension
DOK-MTX***-NC*F*EX*V15-RERS-EN-P, R911393316
This documentation describes the extended functions of the MTX. The basic commissioning steps and the control functions are provided as description and handling instruction.
MTX 15VRS Functional Description - Special Functions
DOK-MTX***-NC*F*SP*V15-RERS-EN-P, R911393309
This documentation describes the special functions of the MTX. The basic commissioning steps and the control functions are provided as description and handling instruction.
MTX Free Form Surface Milling
DOK-MTX***-FREEFORM***-APRS-EN-P, R911341435
This documentation describes the free form surface milling process wit the MTX control. CNC programs generated by a CAD/CAM system are used as basis for the entire process. Overview over the following topics: Description of the MTX or CNC function for freeform surface milling, strategy during the NC parameterization process, boundary conditions when generating CNC programs with CAM software.
MTX Conversion of MTX Projects
DOK-MTX***-PROCONV****-PRRS-EN-P, R911342484
This documentation provides support during the conversion of MTX 1.x projects to MTX 2G. This project planning manual provides information on how to convert a project and identifies potential difficulties during the conversion.

RS Corresponding edition

Tab. 1-3: MTX documentation overview - Configuring

1.4.3 Commissioning

Documentation titles with type codes and part numbers

MTX 15VRS Commissioning
DOK-MTX***-STARTUP*V15-CORS-EN-P, R911393281
This documentation describes the commissioning of the MTX control. Apart from a complete overview, commissioning and configuration of the axes and the user interface as well as the PLC data are described.
IndraWorks 15VRS Basic Libraries, IndraLogic 2G
DOK-IL*2G*-BASLIB**V15-LIRS-EN-P, R911398633
This documentation describes the system-comprehensive PLC libraries.
IndraWorks 15VRS Field Buses
DOK-IWORKS-FB******V15-APRS-EN-P, R911393284
This documentation describes the field bus and local periphery connections supported by the MLC and MTX systems. The focus of this documentation lies in the configuration, parameterization, commissioning and diagnostics of different periphery connections. It is the basis for the online help.
IndraWorks 14VRS WinStudio 7.4
DOK-IWORKS-WINSTUD*V14-APRS-EN-P, R911341585
This “User Manual and Technical Reference Book” supports the user in achieving the best results with the “Rexroth WinStudio”™ software. The document provides technical information and step-by-step instructions to create web-enabled HMI/SCADA programs.

IndraWorks 15VRS Software Installation
DOK-IWORKS-SOFTINS*V15-CORS-EN-P, R911393450
This documentation describes the IndraWorks installation.
IndraWorks 15VRS Engineering
DOK-IWORKS-ENGINEE*V15-APRS-EN-P, R911393303
This documentation describes the use 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.
IndraWorks 15VRS PLC Programming System IndraLogic 2G
DOK-IWORKS-IL2GPRO*V15-APRS-EN-P, R911396137
This documentation describes the PLC programming tool IndraLogic 2G and its use. The documentation includes the basic use, first steps, visualization, menu items and editors.
IndraWorks 15VRS HMI
DOK-MLC***-HMI*****V15-APRS-EN-P, R911399270
This documentation describes the functions, configuration and operation of the user interfaces IndraWorks HMI Engineering and IndraWorks HMI Operation.
MTX 15VRS Shape Cutting Technology
DOK-MTX***-TECHCUT*V15-CORS-EN-P, R911399147
This documentation describes the shape cutting technology used by the machines to cut materials using an NC-controlled tool head.
MTX-integrated CAD/CAM System for Shape Cutting
DOK-MTX***-CAD*TECHCUT-CO01-EN-P, R911393320
This documentation describes the mode of operation and the area of application of the CAD/CAM nesting software Lantek Expert Inside.

RS Corresponding edition

Tab. 1-4: MTX documentation overview - Commissioning

1.4.4 Operation

Documentation titles with type codes and part numbers

MTX 12VRS Block Pre-Run
DOK-MTX***-BLK*RUN*V12-APRS-EN-P, R911334379
This documentation explains to the machine manufacturer how to setup the "Block pre-run" function at the machine for the end user.
MTX 15VRS Programming Manual
DOK-MTX***-NC**PRO*V15-RERS-EN-P, R911393318
This documentation describes the standard programming of the MTX control. Apart from the basics of NC programming, the use of NC functions according to DIN 66025 as well as the NC functions with high-level language syntax and CPL functions are described.
MTX 15VRS Standard NC Operation
DOK-MTX***-NC*OP***V15-APRS-EN-P, R911393314
This documentation describes the operation of the standard user interface of the NC control of the MTX. It includes the operation of the interface, the NC program development as well as the tool management.
MTX 15VRS Multitouch
DOK-MTX***-MULTI***V15-APRS-EN-P, R911393311
The Multitouch user interface of the NC control MTX is described in this documentation.

MTX 15VRS Standard NC Cycles
DOK-MTX***-NC*CYC**V15-PRRS-EN-P, R911394940
This documentation describes the application of the standard cycles of the different technologies for the MTX control.
MTX 15VRS NC Simulation
DOK-MTX***-NC*SIM**V15-APRS-EN-P, R911393273
This documentation describes the NC simulation for the MTX control.
MTX 15VRS Measuring Functions
DOK-MTX***-MES*FUN*V15-APRS-EN-P, R91194938
This documentation describes the measuring cycles of the MTX control.

RS Corresponding edition

Tab. 1-5: MTX documentation overview - Operating

1.4.5 OEM Engineering

Documentation titles with type codes and part numbers

MTX 13VRS Automation Interface
DOK-MTX***-AUT*INT*V13-APRS-EN-P, R911337274
This documentation describes the script-based access to IndraWorks project data via the interface of the Automation Interface. Different objects including code examples are described. The Automation Builder is also described in this manual.
MTX 15VRS OPC Communication
DOK-MTX***-OPC*COM*V15-PRRS-EN-P, R911399272
This documentation describes the syntax and the structure of the items for the communication with Bosch Rexroth devices.
IndraWorks OPC UA Communication
DOK-IWORKS-OPC*UA*****-APRS-EN-P, R911379309
This documentation describes the OPC UA communication of the MLC and MTX control systems.

RS Corresponding edition

Tab. 1-6: MTX documentation overview - OEM engineering

1.4.6 AddOns

Documentation titles with type codes and part numbers

MTX 11VRS Action Recorder
DOK-MTX***-ACR*****V11-APxx-EN-P, R911329943
This documentation describes the MTX action recorder. It includes the installation and commissioning as well as interface signals, application and operation.
MTX 15VRS Efficiency Workbench MTX cta, MTX ega
DOK-MTX***-EWB*****V15-APRS-EN-P, R911400178
This documentation describes the mode of operation and the area of application of the analysis tool MTX cta and MTX ega.

MTX Remote Condition Monitoring
DOK-MTX***-RCM*****V01-APRS-EN-P, R911334383
This documentation describes the operation of the Remote Condition Monitoring System.
MTX visIREC User Documentation
DOK-MTX***-VISIREC*V01-APRS-EN-P, R911344242
This documentation describes the analysis tool visIREC used to optimize the free form surface milling process. 2D or 4D display of path-related data. 2D or 4D display of coordinate-related data. Analysis of critical areas (path and orientation deviation). Comparing the programmed NC blocks to the interpolated NC blocks.

xx / RS Corresponding edition

Tab. 1-7: MTX documentation overview - AddOns

1.5 Information representation

1.5.1 Terms and abbreviations

Term	Explanation
IWE	IndraWorks Engineering
IWO	IndraWorks Operation
OWG	Optical waveguide
NC	Numerical Control
OEM	Original Equipment Manufacturer
Profibus	Communication connection
Sercos	Communication connection
HC	Height control
LF	LeapFrog
FC	FlyCut
LP	Laser power control
SD	System date

Tab. 1-8: Terms and abbreviations

1.6 Customer feedback

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

2 Important instructions on use

2.1 Intended use

2.1.1 Introduction

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

The products may only be used as intended. If they are not used as intended, situations occur that result in damage to property or injury to persons.

☞ Bosch Rexroth 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 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:

- Anyone handling 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. It its not permitted to decompile software products or alter source codes.
- Do not install damaged or defective products or use them in operation.
- It has to be ensured that the products have been installed as described in the relevant documentation.

☞ Ensure that the data present in the control or entered or read in by the user is correct before applying it to exclude unwanted axis motion. It can be the following invalid or old data:

- Part programs
- ZO tables
- Compensation tables
- Tool tables
- Permanent CPL variables
- Remanent PLC data
- Permanent system data

2.1.2 Areas of use and application

The MTX control is used to

- program contour and machining technology (path feed, spindle speed, tool change) of a workpiece.
- guide a machining tool along a programmed path.

Feed drives, spindles and auxiliary axes of a machine tool are activated via Sercos interface.

This additionally requires I/O components for the integrated PLC which - together with the actual CNC - control the machining process as a whole and also monitors this process with regard to technical safety.

It may only be operated with the explicitly specified hardware component configurations and combinations and only with the software and firmware specified in the appropriate documentations and functional descriptions.

The CNC system MTX provides the perfect CNC system solution for cutting and forming for the following technologies:

- Rotate
- Milling
- Drilling
- Grinding
- Bending
- Nibbling
- Punching
- Contour cutting
- Handling

2.2 Unintended use

Using the CNC system MTX outside the previously specified areas of application or under operating conditions other than the conditions described in the documentation and the specified technical data, is defined as "unintended use".

The CNC system MTX must not be used if...

- it is subjected to operating conditions not corresponding to the specified ambient conditions. Operation under water, under extreme temperature fluctuations or under extreme maximum temperatures is prohibited.
- Furthermore, the CNC system MTX shall not be used for applications not expressly approved of by Bosch Rexroth. Therefore, please read the information given the general safety instructions!
- The CNC system MTX may not be used in systems or machines connected to the internet via an unsecure network connection. Otherwise, malfunctions or a control failure can result due to unauthorized access.

3 Introduction to the shape cutting technology

Shape cutting machines are systems that can cut materials up to a certain thickness using a NC-guided machining head. In general, cutting is not done mechanically but with laser, plasma, oxyfuel and water jet cutting.

Shape cutting machines usually have a Cartesian axis arrangement, i.e. x-, y- and z-axis (see fig. 3-2 "Logical axes of a shape cutting machine" on page 10). The x- or y-axis is normally created as a combination of two axes (see fig. 3-1 "Physical drive names of shape cutting machine kinematics" on page 10). One or several cutting heads move perpendicularly to the machining plane. In individual cases, the cutting heads are pivotally mounted.

Typical shape cutting machines are:

- Plasma cutting machines
- Oxyfuel cutting machines
- Laser cutting machines
- Water jet cutting machines
- Combined water jet cutting and sawing machines
- and other 2D applications

The presented project template for shape cutting machines is applicable for the above-mentioned machine types.

Furthermore, additional shape cutting applications such as pipe cutting machines can be implemented with the MTX system. However, the project template for shape cutting machines can only be applied to these types of machines to a limited extent or not at all.

Apart from the normal NC functionality of the MTX system, additional special functions are required for shape cutting machines. These technology functions include:

- Height control
- FlyCut
- LeapFrog
- Power control
- Re-enter and Retrace functionality
- Cutting width compensation
- Axis group (Gantry)
- Continuous Cut
- Material tables

The function and commissioning of the above-mentioned technology functions are described in detail in the following chapters.

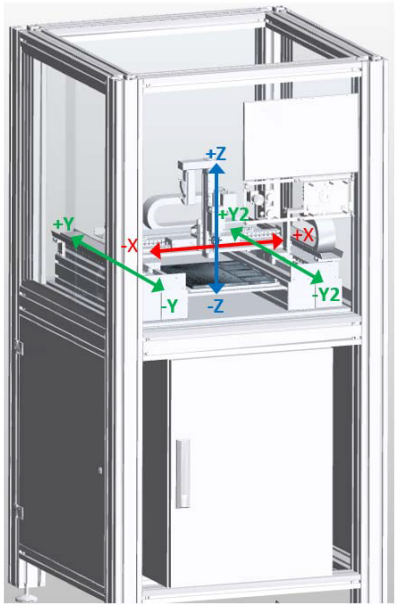


Fig. 3-1: Physical drive names of shape cutting machine kinematics

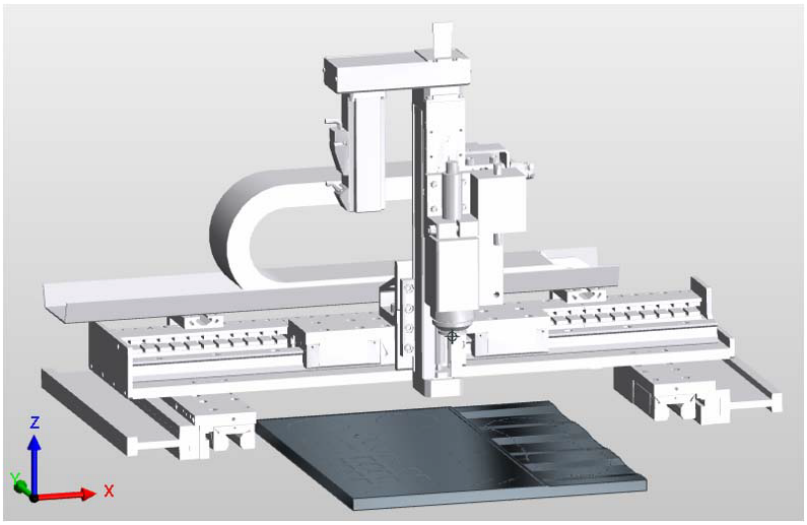


Fig. 3-2: Logical axes of a shape cutting machine

4 Shape cutting technology project template

4.1 Introduction and requirements

To ease the commissioning and to quickly access the technology functions required to operate a shape cutting machine, all relevant functions are summarized in the "Shape cutting technology" project template. Select this project template when creating a new IndraWorks project.

When designing this project template, it was focused on a simple structure and on great extendibility. However, manual work is necessary in some areas to adapt the template to the target machine.

The following chapters describe the functions, the commissioning, the diagnostic options and the adjustment options to the own application.

Hardware requirements

The following hardware is required to use the shape cutting technology project template:

- MTX control (XM42-S, XM42-P or XM42-A)
- Drive controller (IndraDrive CS/M/Mi)

For height control:

- Distance/position sensor with an analog output
- IndraDrive with an analog input (X32 or X38) or another Sercos field bus device with an analog input

  Other field buses are possible, but they are not time-synchronous to the actual position value and the command value of the axes.

For laser power control:

- Analog output:
  - Drive controller with analog output (IndraDrive CS/M)
  - S20-AIAO-2, S20-AO-4, S20-AO-8 analog output module
- PWM output:
  - S20-DIDO-2-CSOVS high-speed output module

For FlyCut:

- MTX control (XM42-S, XM42-P or XM42-A)
- S20-DIDO-2-CSOVS high-speed output module

Software requirements

The following software is required to use the shape cutting technology project template:

- IndraWorks MTX 15V04 or higher
- MTX-FW 15V04 or higher
- IndraDrive FW MPx20V24 or higher

Software licenses

To use the shape cutting technology project template or its implemented functions or the functions provided by the MT_Technology PLC library, the following software license is required:

R911396571: FWS-MTX-SHC2*****-NNVRS-NN-NNN

It includes the following licenses:

- R911320636: "Shape cutting 1" technology package
- R911337420: Fast Actions MTX fa Runtime
- R911318989: "Electronic gear" technology package

4.2 Creating an IndraWorks project from the project template

This chapter describes how to create a new project using the project template. It is required that IndraWorks and all necessary SW licenses have already been installed.

First, a new IndraWorks project is created using the "Technology project shape cutting" project template. This project template is part of the IndraWorks installation. It can be selected when creating a new IndraWorks project (**File ▸ New ▸ Project...**) (see fig. 4-1 "Creating a new project using the project template" on page 12). Select the "MTX" entry in the dialog under "Categories". Select the "Technology project shape cutting" under "Templates".

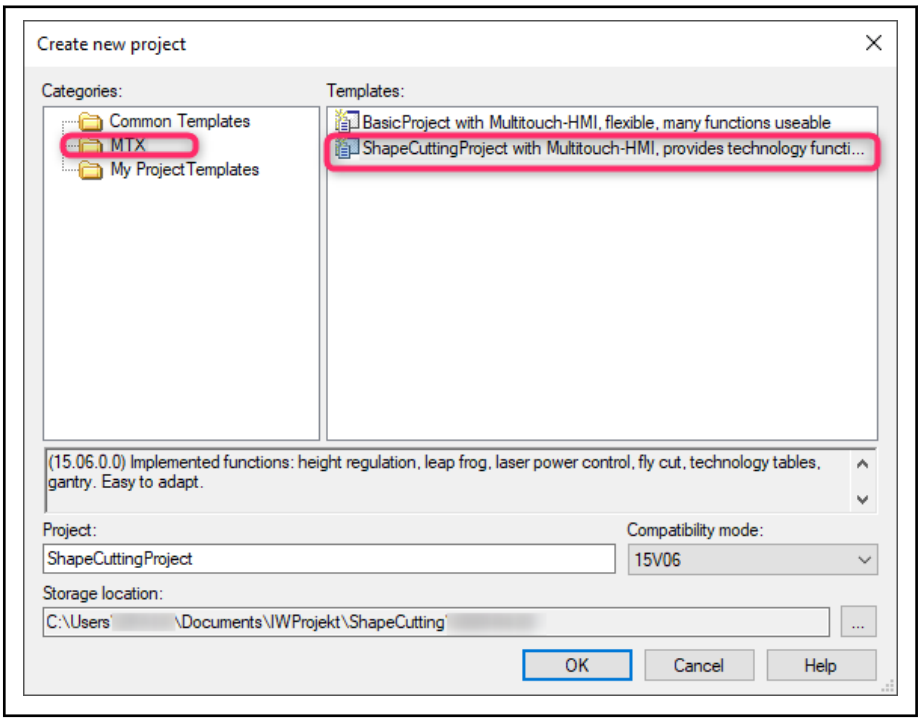


Fig. 4-1: Creating a new project using the project template

4.3 Project structure

The project template is structured like a normal MTX project. Special features and storage locations of the functions are explained in the following.

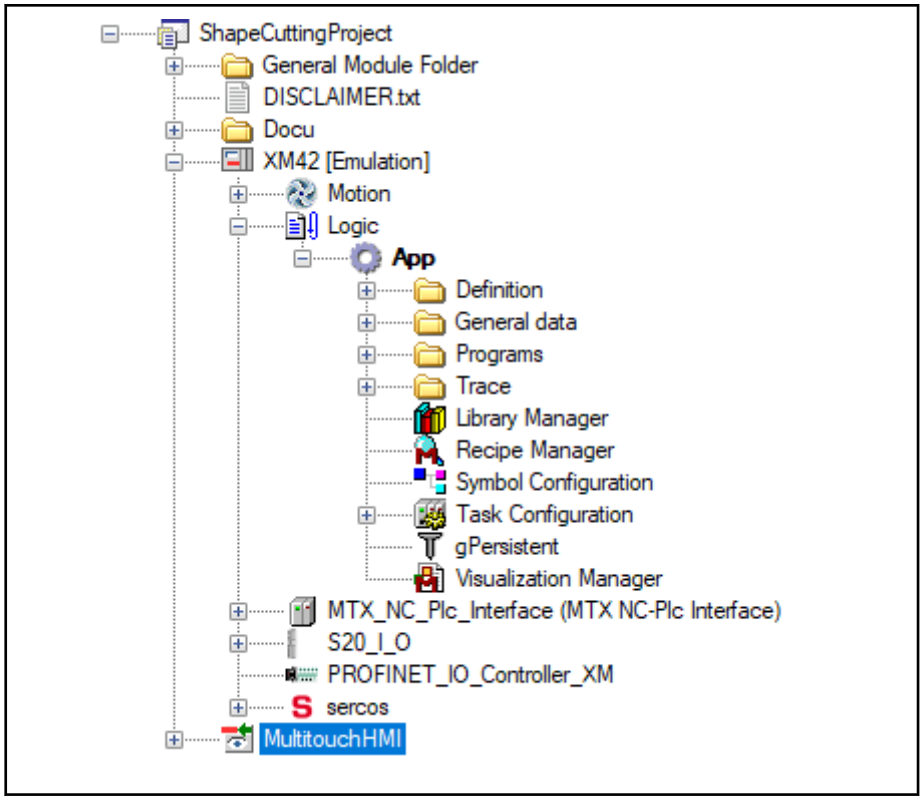


Fig. 4-2: Project structure of the shape cutting technology template

As shown in 4-2, the project includes a control (**XM42**) as well as an HMI device (**VEP15.6_Multitouch**).

The **Docu** folder contains the documentation for the project template and individual technology functions. The template does not require this folder. It is freely editable, e.g. to store own project documentations.

Below the control, the programs and the settings of the PLC are stored in the **Logic** ▸ **App** node.

The **Logic** ▸ **App** ▸ **Definition** folder includes general settings for the project. The "gDimensioning" file is important. It includes the settings for the following parameters:

- Number of drives
- Number of spindles (normally 0)
- Number of NC channels
- Number of visualization devices

Functions, data structures and global variables required for the project are stored in the **Logic** ▸ **App** ▸ **General data** folder. Generally, the functions, data structures and standard variables do not have to be adjusted. Define the customer-specific variables in the folder **Global variables** ▸ **Specific**.

The actual PLC program parts are stored under the **Logic** ▸ **App** ▸ **Programs** folder. The PLC task "tskPLC" calls the "Main" program every 10 ms. All general machine functions are executed in the program. All functions called by Main are stored under the **Logic** ▸ **App** ▸ **Programs** ▸ **Machine** folder.

All PLC-controlled special technology functions in the shape cutting area are stored under the **Logic** ▸ **App** ▸ **Programs** ▸ **Technology** folder. The following chapters describe these functions in detail.

Most technology functions require a synchronous data exchange to the NC. The required definitions and programs are stored under the **Log-**

ic ▸ App ▸ Programs ▸ FA_PLC folder. The synchronous data exchange is described in detail in chapter 5 "Fast data exchange" on page 25.

For the sample trace diagrams of the monitored technology function signals, go to the folder **Logic ▸ App ▸ Trace**

All PLC functions are called by three PLC tasks that are stored under **Logic ▸ App ▸ Task Configuration**:

- tskPLC: PLC task that is called every 10 ms. It is used for the machine control and operates technology functions that are not time-sensitive.
- tskFA: Fast Action task that is synchronously called after editing the NC interpolator. In this task, all time-sensitive technology functions are processed.
- tskFastIO: PLC task in which the I/O signals are exchanged.

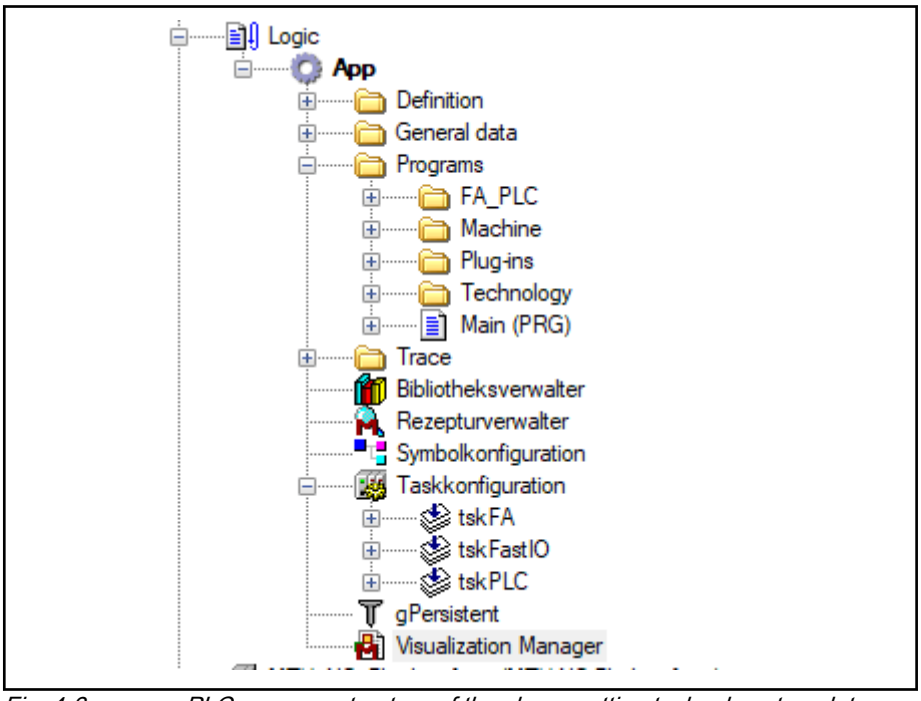


Fig. 4-3: PLC program structure of the shape cutting technology template

4.4 Commissioning the NC

The shape cutting project consists of two parts: The PLC part and the NC part. The PLC and the NC communicate via auxiliary variables and the FA task.

The NC part consists of several files in the file system of the control. There are two commissioning options:

- Installation by restoring the archive (see "Restoring the archive" on page 14). All required files are installed at the correct location.
- Copying and renaming all required files from the `/feprom` directory (see "Files on the control" on page 16).

Restoring the archive

The following steps describe how to restore the archive and therefore all NC functions on the control required for the shape cutting project.

1. Select the control

   Selection of the control and the corresponding firmware with subsequent network configuration according to 4-4.

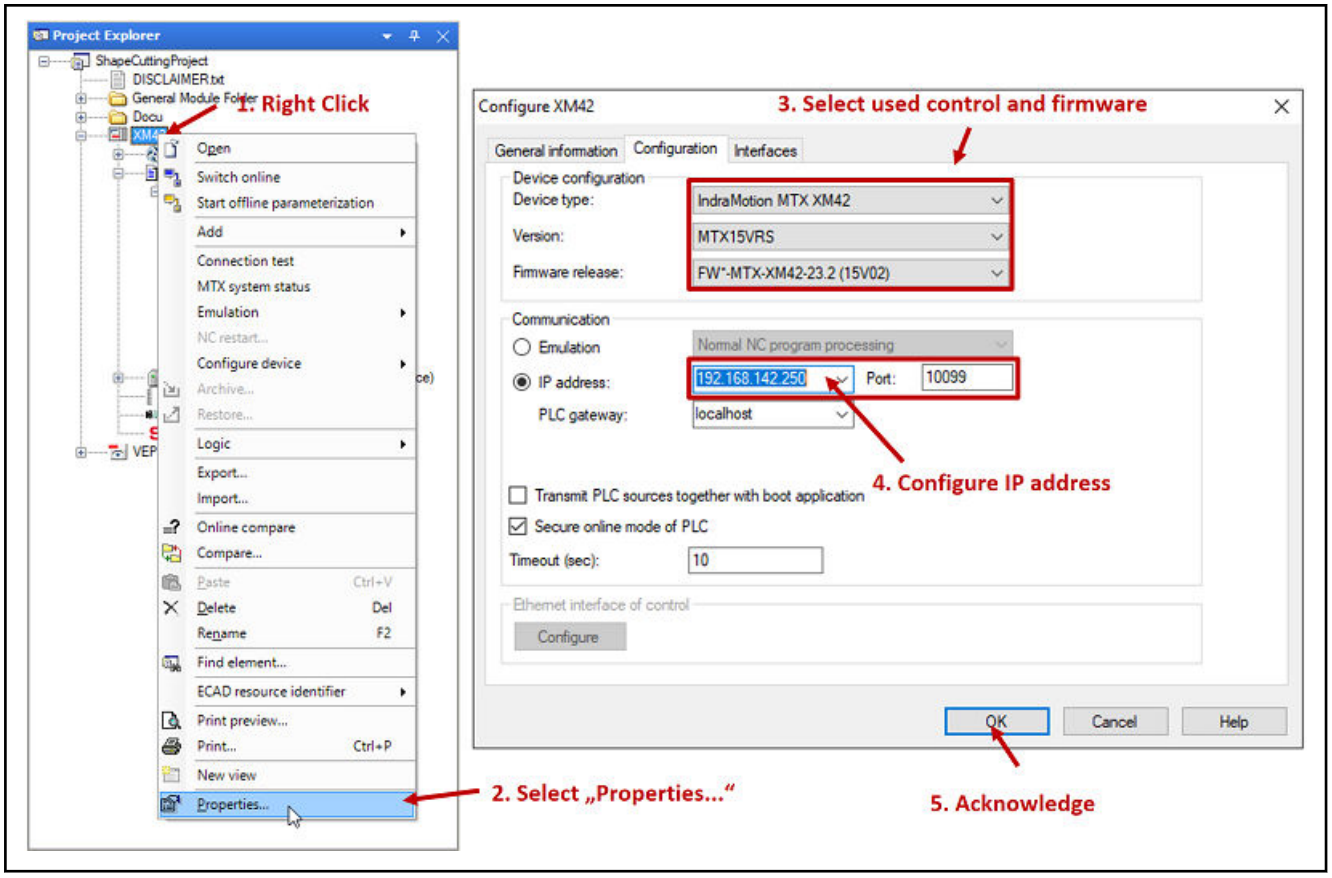


Fig. 4-4: Selecting the MTX control and the network configuration

2. Switch the control online

   After switching the control online, the file system on the control can be accessed (see 4-5)

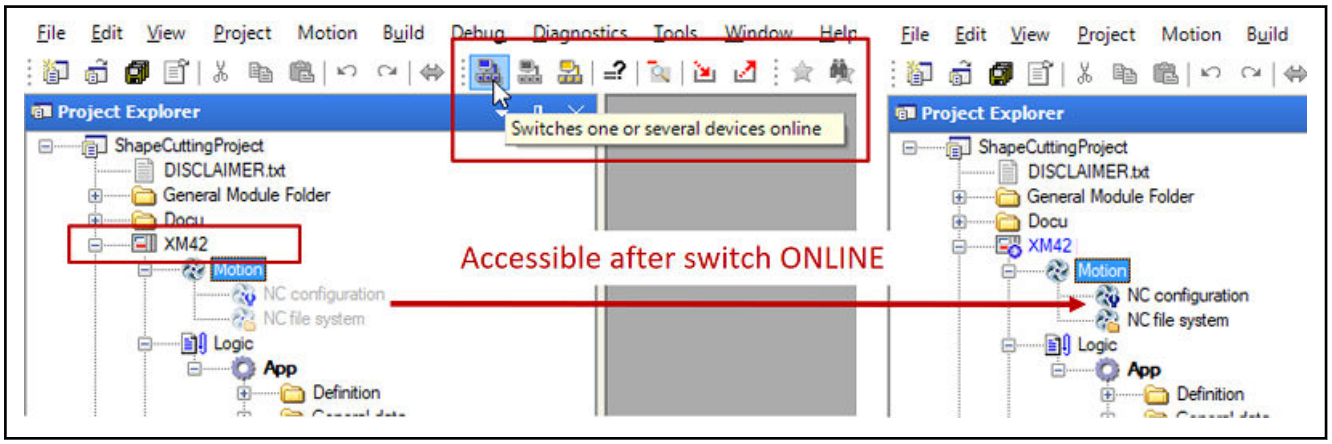


Fig. 4-5: Switching the MTX control online

3. Restore the TAR file

   The TAR file is restored as shown under Restoring the standard settings for the shape cutting project on the control.

   It includes the NC parts of the following modules:

   - Axis configuration (incl. Gantry)
   - Settings for the NC part of the FA task

- Height control
- Laser power control
- LeapFrog
- FlyCut

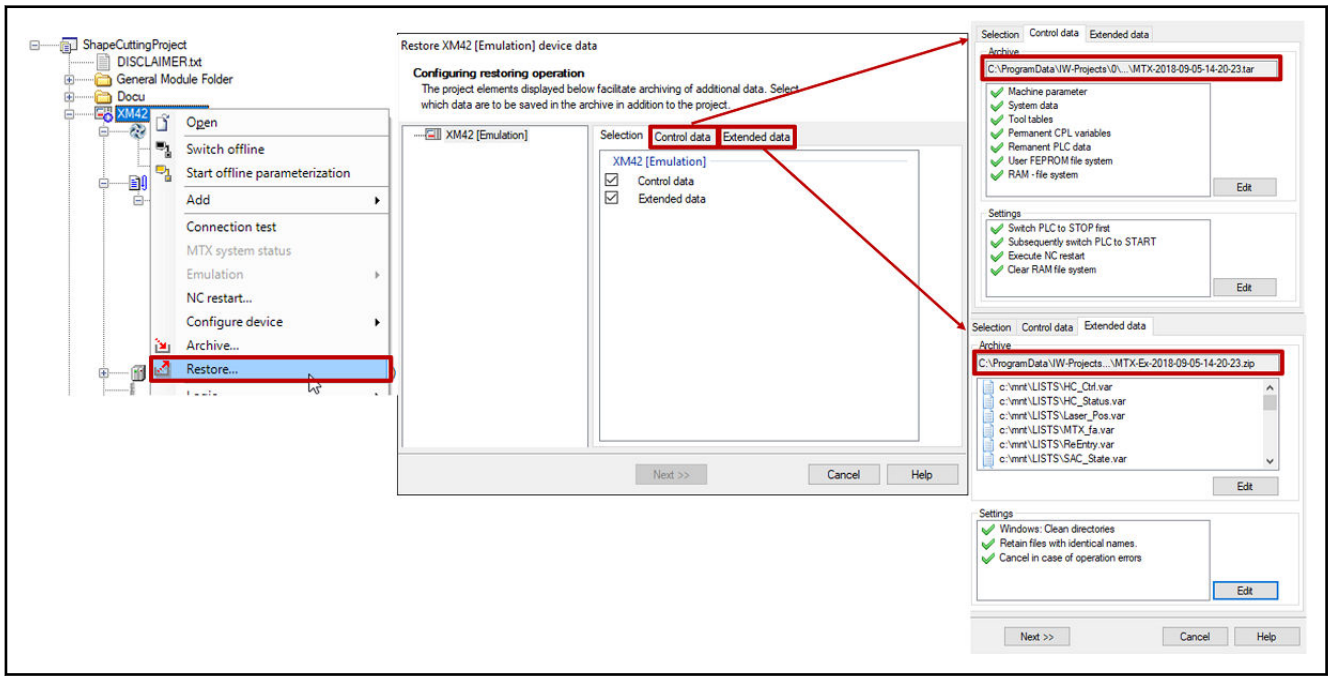


Fig. 4-6: Restoring the standard settings for the shape cutting project on the control

4. Adjust the axis configuration

   Adjust the axis configuration under **Motion ▸ NC Configuration** (see 4-5). The number of the channels and axes used is provided under **XM42 ▸ Logic ▸ App ▸ Definition ▸ gDimensioning**. This configuration is explained in the "MTX Machine Parameter" documentation (see Required and supplementing documentation MTX, page 2).

Files on the control

The shape cutting project includes several files stored on the file system of the control. They are usually transferred to the control with the TAR file (see Restoring the standard settings for the shape cutting project on the control). If the TAR file is not used, the files can also be adjusted individually.

The following cycles are required:

- `/feprom/shapecut/TECHTABLE`: Implementation of the `TECHTABLE` macro to load the technology tables
- `/feprom/shapecut/HEIGHTCTRL`: Implementation of the `HEIGHTCTRL` macro to disable the height control
- `/feprom/shapecut/SENSORCALIB`: Implementation of the `SENSORCALIB` macro to calibrate the sensor
- `/feprom/shapecut/CUT`: Implementation of the `CUT` macro to enable and disable the laser power control
- `/feprom/shapecut/GAS`: Implementation of the `GAS` macro to enable and disable the protective gas supply
- `/feprom/shapecut/FOCUS`: Implementation of the `FOCUS` macro to enable and disable the focus control

- `/feprom/shapecut/subLEAPFROG`: Macro implementation`LEAPFROG` to execute the LeapFrog functionality
- `/feprom/shapecut/subDELAY`: Implementation of the `DELAY` macro to execute the time delay functionality

To facilitate the call of the cycles, they were implemented into macros. They are included in the following file:

- `/feprom/macro/MacroDef_ShapeCut.xml_`: Implementing the macros for the cycles `TECHTABLE`, `HEIGHTCTRL`, `SENSORCALIB`, `CUT`, `GAS`, `FOCUS`, `LEAPFROG`, etc. You can define own macros in this file.

Furthermore, the following definitions are required for the system data:

- `/feprom/SDDefShapeCut.xml_`: Inclusion of height control, laser power control and LeapFrog in the system data of the control
- `/feprom/SDDatFAShapeCut.xml_`: Parameterization of the FA task
- `/feprom/schemas/SDHeightCtrl.xsd_`: Definition of the data types for the height control
- `/feprom/schemas/SDLeapFrog.xsd_`: Definition of the data types for the LeapFrog function
- `/feprom/schemas/SDPowerCtrl.xsd_`: Definition of the data types for the laser power control
- `/feprom/schemas/sdadvdistsig.xsd_`: Definition of the data types for the ADS function
- `/feprom/schemas/SDFlyCut.xsd_`: Definition of the data types for the FlyCut function

Installation

Extend the NC search path by the `/feprom/shapecut` entry to include to above-mentioned cycles. This is done in the NC configuration.

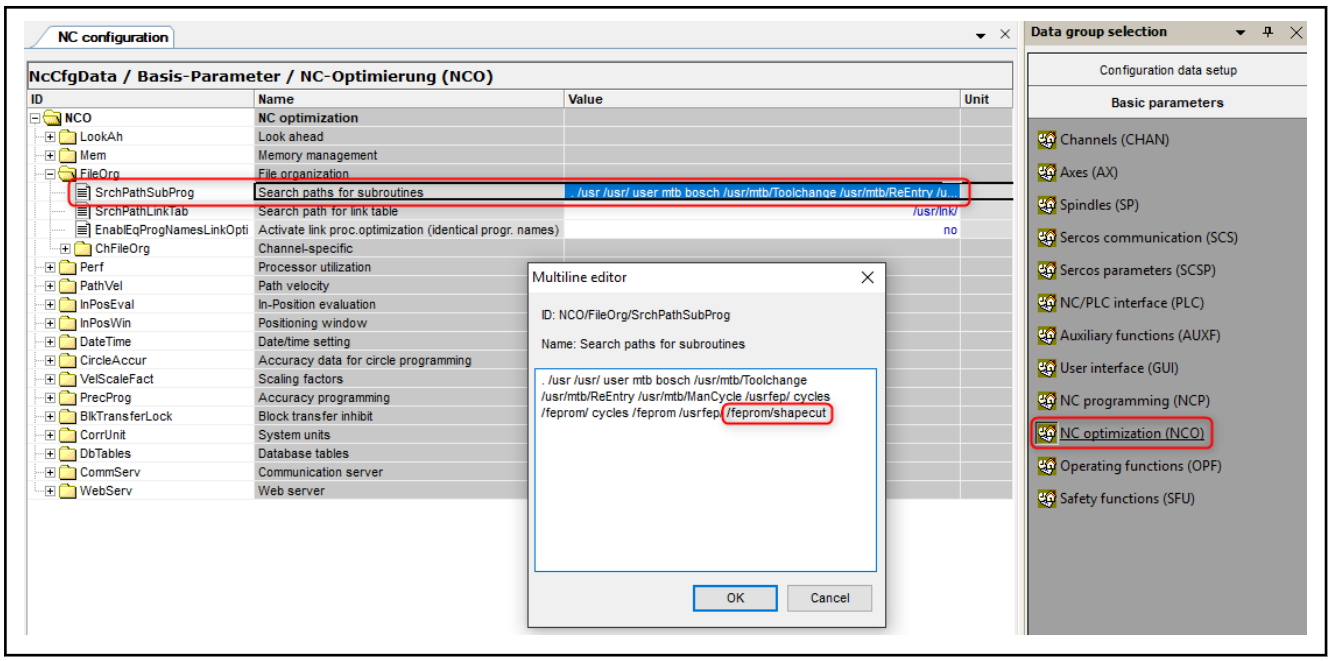


Fig. 4-7: Adding the cycles to the NC search path for shape cutting.

Copy the following files to the control and rename them to enable the shape cutting function:

Source	Target
/feprom/SDDefShapeCut.xml_	/usrfep/SDDefShapeCut.xml
/feprom/SDDatFAShapeCut.xml_	/usrfep/SDDatFAShapeCut.xml
/feprom/macro/MacroDef_Shape-Cut.xml_	/usrfep/macro/MacroDef_Shape-Cut.xml
/feprom/schemas/sdadvdist-sig.xsd_	/usrfep/schemas/sdadvdist-sig.xsd
/feprom/schemas/sdfa.xsd_	/usrfep/schemas/sdfa.xsd
/feprom/schemas/sdrtpos.xsd_	/usrfep/schemas/sdrtpos.xsd
/feprom/shapecut/HCSAC_Z_OFF	/usrfep/HCSAC_Z_OFF
/feprom/shapecut/SACVALID	/usrfep/SACVALID

Tab. 4-1: List of the files to be copied

Parameterization

The settings of all modules have to be adapted to the machine. For this purpose, edit the /usrfep/techtable/machine file. Proceed as described in Storage location of the technology tables and the MACHINE file.

For more information on the setting parameters, refer to the chapter Technology tables, page 105.

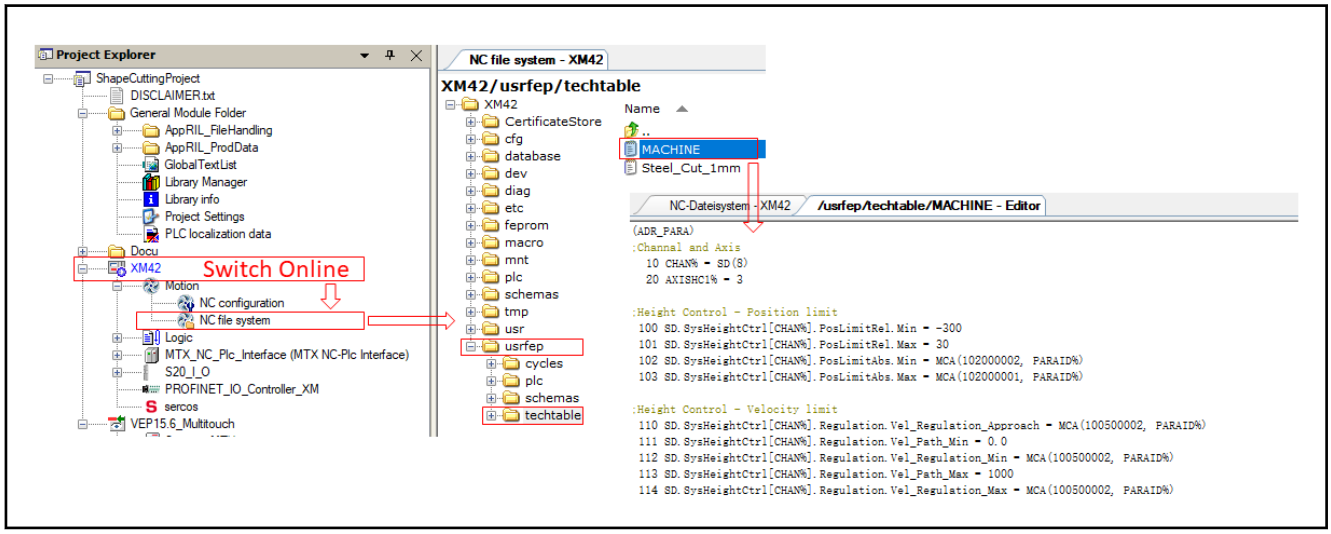


Fig. 4-8: Storage location of the technology tables and the MACHINE file

4.5 Commissioning of the PLC

Before the shape cutting project can be commissioned, the configuration has to be transferred to the control:

1. PLC log in

   Log the project into the PLC with and start the PLC with . Check that the "tskFA" and "tskPLC" tasks are performed (see Function test, page 19).

2. Transferring the HMI

   Transfer the visualization data to IndraWorks Operation (4-9) and activate the Operation simulator or an external IW Operation.

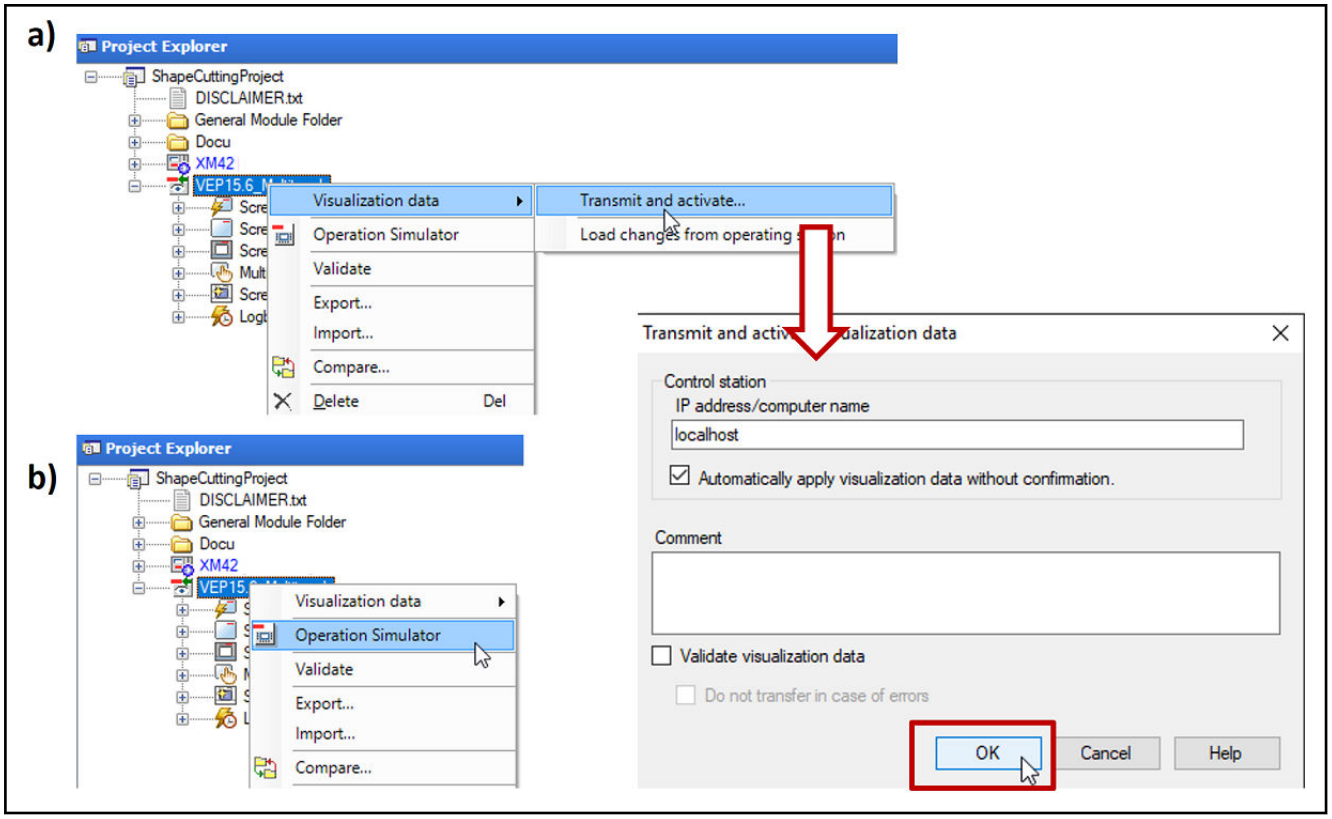


Fig. 4-9: Activating the IndraWorks Operation simulator with transfer of the visualization data

4.6 Function test

After all project parts have been loaded to the machine, the machine functions have to be tested. It is recommended to perform the test by means of the following steps.

Function test of the machine

In order to perform the function test, the following prerequisites have to be fulfilled:

- *All project parts have been loaded to the control*
- *The PLC is in the RUN state*
- *The HMI is connected and indicates the machine state*
- *The height sensor is installed and working*

1. Sercos bus test

   The Sercos bus has to be in P4. If this is not the case, please go to chapter 6.7 "Troubleshooting" on page 54 to correct the error.

2. Testing the PLC tasks

   When executing the task, its status switches to "Valid" and the "Cycle Count" counts up (see 4-10). If "Cycle Count" remains 0, please go to chapter 6.7 "Troubleshooting" on page 54 to correct the error.

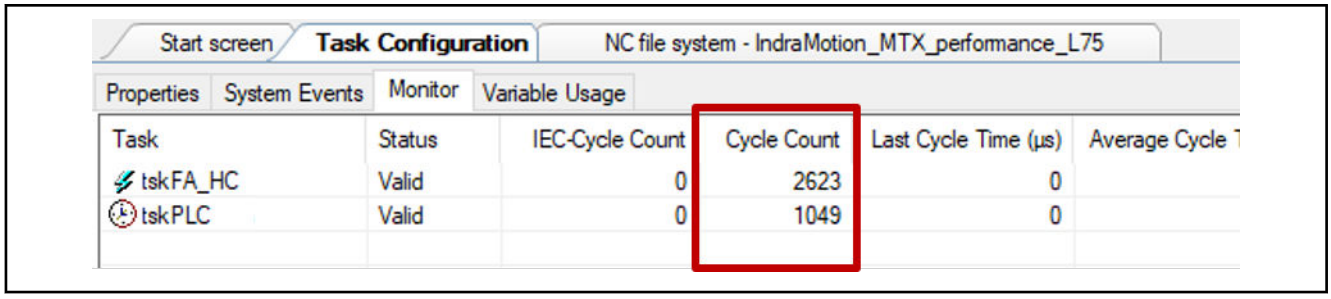


Fig. 4-10: Calling the PLC tasks

3. Testing the height sensor

   If the FA task is configured correctly and the height sensor is connected, the sensor value is displayed in the input data of the FA task.

   It has to be checked that the Main_FA.stFAInputData.iSensorVal variable displays the current sensor value and changes when approaching metal. If this is not the case, proceed as described in chapter Troubleshooting, page 54.

4. Testing the axis motion

   The axes are tested in the manual operation mode of IndraWorks Operation (see 4-11):

   - Operate the machine in "Manual mode"
   - Determine the axis velocity with the "Override" controller
   - Modify the positions of the x-/y-/z-axes and check that the axes move correctly and that their positions are shown in the "Command Position" area
   - Check the maximum and minimum positions of the respective x/y/z-axes

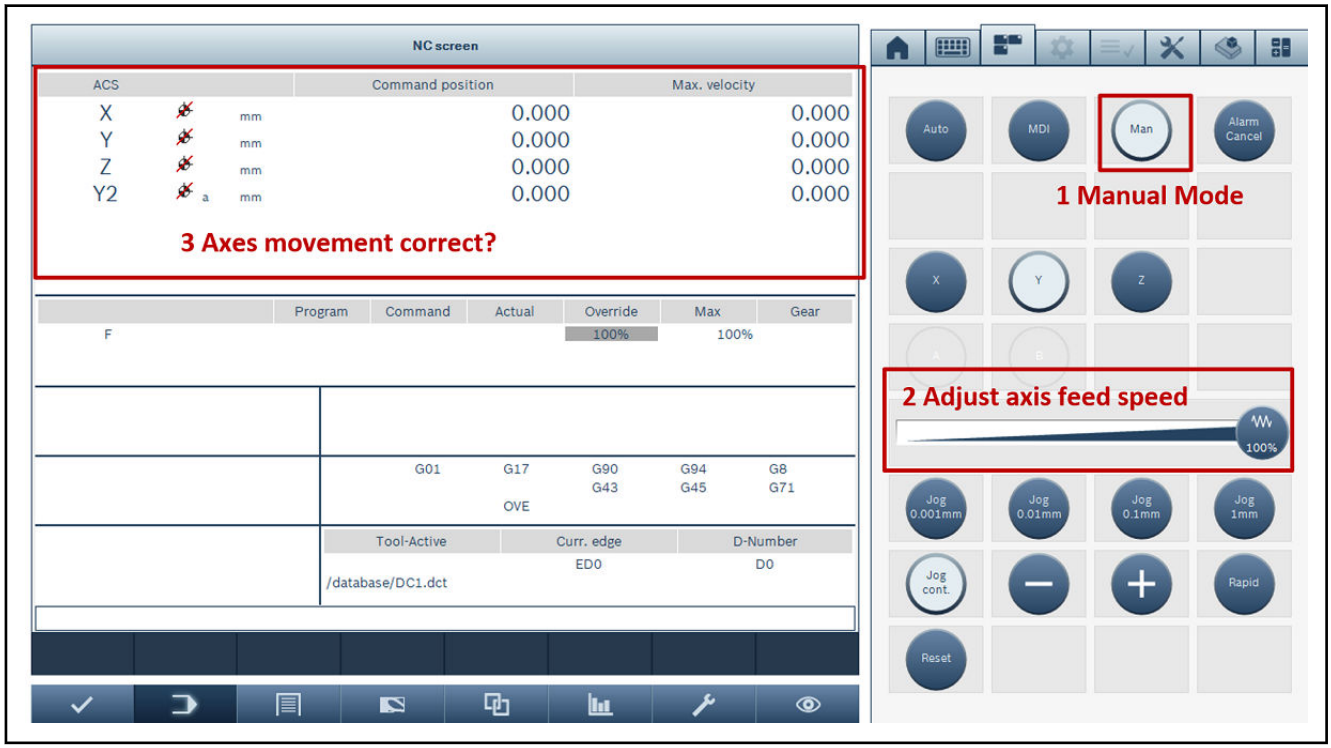


Fig. 4-11: Test the axis by typing manually

4.7 Sample program

The following small sample program runs with the template project. It is used as an introduction and template for own NC programs. The calls and macros used in the program are described step by step. For further details, refer to the corresponding chapters.

Sample program

```
N10 TECHTABLE("Steel_Cut_1mm");Load material table
N20 FFW(Z1)                   ;pre control Z axis
N30 HEIGHTCTRL(0)             ;switch off HC
N40 G1Z0                      ;move to reference position of z axis
N45 G0X45Y-40                 ;move to start position X/Y axis
N50 SENSORCALIB("CALIB")      ;Calibrate sensor
N60 G1Z0                      ;Move to start position of contour

N70 GAS(1)                    ;Switch on gas
N80 HEIGHTCTRL(1)             ;Switch on height regulation
N90 CUT(1)                    ;switch on laser power
N95 RETRACE(1)                ;switch on retrace

;cut the contour
N100 G1Y40
N110 G1Y230
N120 CUT(0)                   ;switch off laser power
N130 LEAPFROG X20Y0           ;fast repositioning
N140 CUT(1)                   ;switch on laser power
N150 G1Y230

N900 RETRACE(0)               ;switch off retrace
N910 CUT(0)                   ;switch off laser power
N920 HEIGHTCTRL(0)            ;Switch off height regulation
N930 GAS(0)                   ;Switch off gas
```

The program can be divided into three sections. The lines are numbered for a better overview. The first section from `N10` to `N95` prepares the machine for the cutting process. In the section from line `N100`, the workpiece is cut. In the last section from `N900`, the machine is set to its initial mode.

In the following, the individual program sections are described in detail.

Initialization

A technology table is loaded with the `TECHTABLE` subroutine call in line `N10`. It defines all parameters that are important for the machine and the material to be cut such as cutting velocity, laser control, height of the cutting head above the material, etc. The technology tables are stored as editable files in a directory on the control or in a directory connected via NFS. The file name is used as parameter in the `TECHTABLE` command. Technology tables save additional parameters for later function and subroutine calls and allow to change material without editing the actual cutting program.

In line `N30`, the height control is disabled by the `HEIGHTCTRL(0)` subroutine call. This call ensures that the cutting head is not moved close to the material during a subsequent motion.

The starting point for the calibration of the height control is approached in the lines `N40` and `N45`. Adjust the position for the calibration if required. The `G1` command is used for the referencing motion. The velocity of the motion is specified in the `Steel_Cut_1mm`. The cutting head is then moved to the starting position with `G1`.

The `SENSORCALIB` call in line `N50` performs the sensor calibration with a subsequent check of the determined measured values. If the `CALIB` parameter is specified, it is calibrated again. If the `LOAD` parameter is specified, only the last calibration is loaded. This calibration file is named as SD variable in `Steel_Cut_1mm`. It is recommended to perform a new calibration after every change of material.

Line N60 moves the cutting head to the starting position of the contour.

GAS(1) in line N70 activates the cutting gas. Call before activating the height control, as the cutting gas affects the scattering of the height sensor.

In following line N80, the height control is activated. The cutting head is moved to the material. The process is finished when the controlled state is reached.

Line N90 activates the laser with the CUT(1) command. The laser automatically performs a piercing process if it is configured accordingly in the material table.

The RETRACE function is activated in line N95. This function enables the following forward and backward traversing motion. It is thus possible to move forward and backward on the contour to restart the cutting process at a suitable position after an interruption.

Cutting process

The actual cutting process starts from line N100. Here, the contour is completely traversed once. To interrupt the cutting process, stop the program at any time. The laser is disabled automatically.

In the lines N120 to N140, quickly position the material with a so-called Leap-Frog. For this purpose, the laser is disabled at the target position of the preceding traversing command. The cutting head is then lifted and quickly moved by the LEAPFROG call. The laser is enabled again at the target point.

Finish

To set the machine to its initial state, all functions are disabled subsequently.

In N900, the RETRACE function is disabled first. It is recommended to enable the retrace function only once per contour or contour group.

N910 disables the laser by starting CUT(0).

Then, the height control is disabled in line N920. This results in lifting of the cutting head to a safe distance.

The cutting gas is disabled in N930 using GAS(0).

Command reference

In the following, all commands used in the sample program are listed. For a detailed description of the functionality, refer to the referenced chapters.

- TECHTABLE: Loads a material table. (see Technology tables, page 105)
- HEIGHTCTRL: Enables or disables the height control. (see Height control, page 37)
- SENSORCALIB: Calibrates the sensor and loads an existing calibration. (see Sensor calibration: SENSORCALIB, page 50)
- GAS: Enables and disables the cutting gas. (see Gas control: GAS, page 87)
- CUT: Enables and disables the laser. (see Laser power control, page 75)
- LEAPFROG: Repositions the cutting head quickly. For this purpose, the cutting head is lifted and lowered again. (see LeapFrog, page 65)
- RETRACE: Controls the ability to move forward and backward on the path (see Retrace, page 103)

Demo program

Two demo programs MAIN and LASERDEMO and the respective post- and pre-process programs are stored in the mount directory **mnt ▸ NPG**. The post-process is adapted to the machine used. The pre-process is designed for different machining processes. The machine performance can be tested when implementing both programs.

4.8 M-auxiliary functions overview

M-auxiliary functions apply the configuration and some subfunctions. All assigned M-auxiliary functions are listed in the following.

Auxiliary function	Module	Use
M0	In general	Interrupting the NC program
M1	In general	Interrupting NC program conditionally
M2	In general	Ending NC program
M169	HC	Loading configuration
M170	HC	Waiting for stabilization of the sensor value
M171	HC	Resetting the calibration
M172	HC	Loading calibration data
M174	HC	Reading in HC mode
M175	HC	Resetting PlateSearch
M176	HC	Saving PlateSearch results
M177	HC	Loading Precitec calibration file
M178	HC	Resetting Precitec calibration
M184	LP	Reading in the configuration of gas, focus and laser power control
M185	GS	Waiting for stable gas pressure
M186	FO	Waiting for focus position
M187	LF	Trigger LeapFrog
M188	LF	Waiting for LeapFrog execution
M210	FC	Loading configuration

Tab. 4-2: M-auxiliary functions of the shape cutting templates

5 Fast data exchange

5.1 Introduction and overview

Many control processes require parts of the PLC program running in sync with the drive bus and the NC control. The transfer from and to sensors and actuators requires the control process to be synchronous as well.

There are synchronous data telegrams in the Sercos bus that can be transferred with each bus cycle. They contain the actual and nominal values of the drives as well as the sensors and actuators, if required.

5.2 Main_FA program

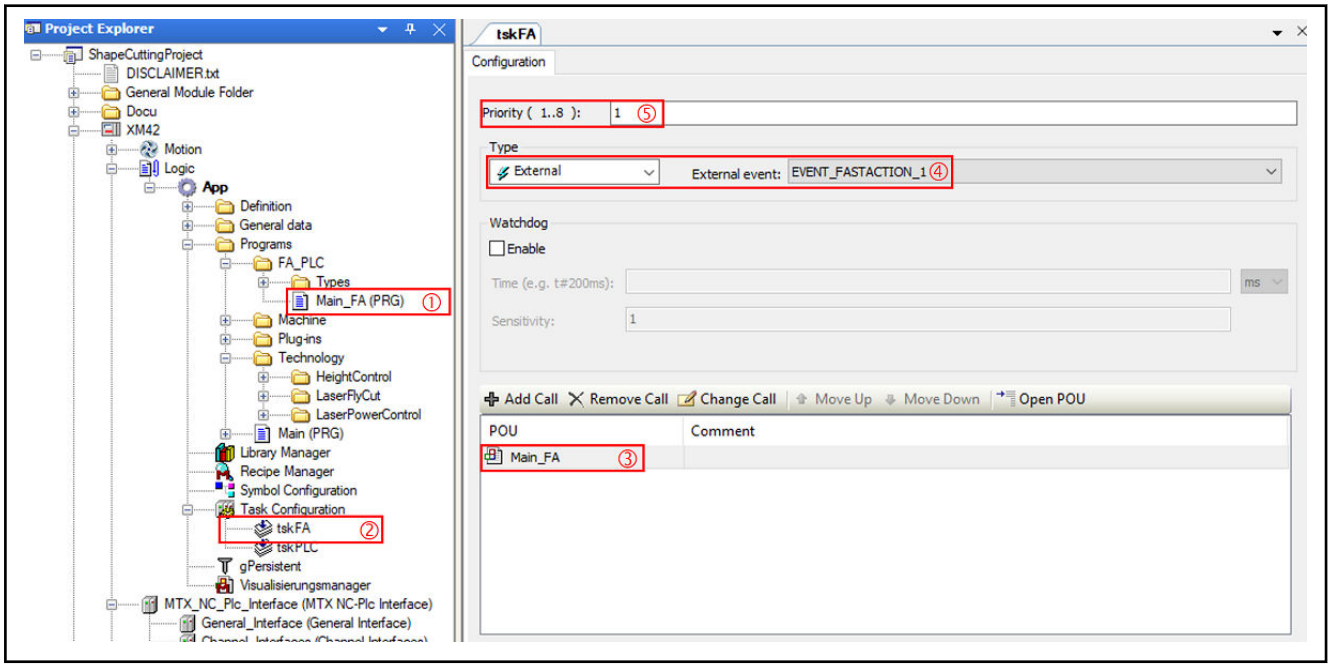
The synchronous data exchange between the NC and the PLC is done via a fast PLC task – the Fast Action (FA) task. It is called by the NC directly after calculating the new axis positions. Up to 100 values can be transferred from the NC to the PLC and from the PLC to the NC via two data exchange storages.

This task is required for the height control, laser power control and FlyCut technology functions. It guarantees a synchronous and therefore exact calculation and transfer of the values necessary for the execution of the technology functions.

In the shape cutting project, the "Main_FA" FA task is located at:

XM42 ▸ Logic ▸ App ▸ Programs ▸ FA_PLC.

Block 1 in fig. 5-1 "Fast Action task of the shape cutting project" on page 25 shows the storage location of the PLC program.



1 "Main_FA" PLC program
2 "tskFA" PLC task
3 Call of the "Main_FA" PLC program within the "tskFA" PLC task
4 Task setting EVENT_FASTACTION_x external event
5 Task priority 1 (highest priority)

Fig. 5-1: Fast Action task of the shape cutting project

A FA task (block 2, 3) calls the program "Main_FA". The Fast Action task type has to be configured to "External". Either `EVENT_FASTACTION_1` or `EVENT_FASTACTION_2` can be selected as the corresponding event. In general, it is sufficient to work with one Fast Action task. The Fast Action task should have the highest priority (1).

5.3 Fast inputs and outputs

Files on the control

The fast action functionality is mapped in the following files:

- `/feprom/SDDatFAShapeCut.xml_`: Definition of the values to be transferred cyclically between NC and PLC
- `/feprom/SDDefShapeCut.xml_`: Inclusion of the ShapeCut functionality in the system data of the control
- `/feprom/SDDeffa.xml_`: Inclusion of the FA functionality in the system data of the control
- `/feprom/schemas/SDHeightCtrl.xsd_`: Definition of the data types for the ShapeCut functionality in the system data of the control
- `/feprom/schemas/sdfa.xsd_`: Definition of the data types for the FA functionality in the system data of the control

Installation

The files listed above are either installed by restoring the archive (see chapter 4.5 "Commissioning of the PLC" on page 18) or by copying and renaming them:

`/feprom/SDDatFAShapeCut.xml_` ⇒ `/usrfep/SDDatFAShapeCut.xml`

`/feprom/SDDefShapeCut.xml_` ⇒ `/usrfep/SDDefShapeCut.xml`

`/feprom/SDDeffa.xml_` ⇒ `/usrfep/SDDeffa.xml`

`/feprom/schemas/SDHeightCtrl.xsd_` ⇒ `/usrfep/schemas/SDHeightCtrl.xsd`

`/feprom/schemas/sdfa.xsd_` ⇒ `/usrfep/schemas/sdfa.xsd`

The file `SDDatFAShapeCut.xml` is just a template and is then to be filled with the parameters to be transferred. This is described under chapter 5.5 "Adding a new output value to the system data" on page 31.

The content of the `SDDatFAShapeCut.xml` file is shown in the following example. This example can also be used as template.

Program:

```
<?xml version="1.0"?>
<SD>
  <SysFA>
    <PLCTask>
      <Control>
        <Enable>1</Enable>
        <Reset>0</Reset>
      </Control>
      <Config>
        <Base>
          <IECTaskName>tskFA</IECTaskName>
          <Cycle>1</Cycle>
          <MaxPLCTaskTime>400</MaxPLCTaskTime>
        </Base>
        <CycData>
          <Cmd>AxisPos</Cmd>
          <ID>100000</ID>
          <Access>0</Access>
          <Index1>1</Index1>
          <Index2>0</Index2>
```

```
        </CycData>
      </Config>
    </PLCTask>
    <NCTask>
      <MaxFARunTime>400</MaxFARunTime>
    </NCTask>
  </SysFA>
</SD>
```

- The maximum permitted execution time of the PLC task should not exceed 80% of the interpolation cycle time (note: if 2 fast PLC tasks are configured, the sum has to be < 80%): **MaxPLCTaskTime < 0.8 * IPOCycleTime**
- We recommend setting the maximum permissible execution time of the FA task to approx. 3/10 of the interpolator cycle time: **MaxFARunTime < 0.3 * IPOCycleTime**
- After copying the files, restart the NC to activate functionality.

MT_FADataRead, MT_FADataWrite

Most technology functions use the FA functionality (Fast Action) for a fast data exchange between the NC and the PLC. The data of the technology functions to be read or written uses the `MT_FADataRead` and `MT_FADataWrite` functions. These two functions copy the data between the PLC and the NC. They are started in the "Main_FA" PLC program.

In this chapter, the FA functionality and the FA data exchange of the linked variables, the so-called system data, are presented. Then, an example for the adjustment of the FA configuration by adding a new output is shown. For more information on the topic "FA task", see the documentation "MTX 15VRS Functional Description - Special Functions" in the chapter "Fast Actions MTX fa"

For detailed information on the `MT_FADataRead` and `MT_FADataWrite` functions, refer to the documentation "Rexroth IndraMotion MTX Functional Description 15VRS Special Functions".

In the following, the input variables read out by the `MT_FADataRead` FA reading function as well as the output variables exported by the `MT_FADataWrite` FA writing function are presented. fig. 5-2 "Schematic representation of the data exchange via the FA task" on page 28 shows the cyclic data exchange using the FA function.

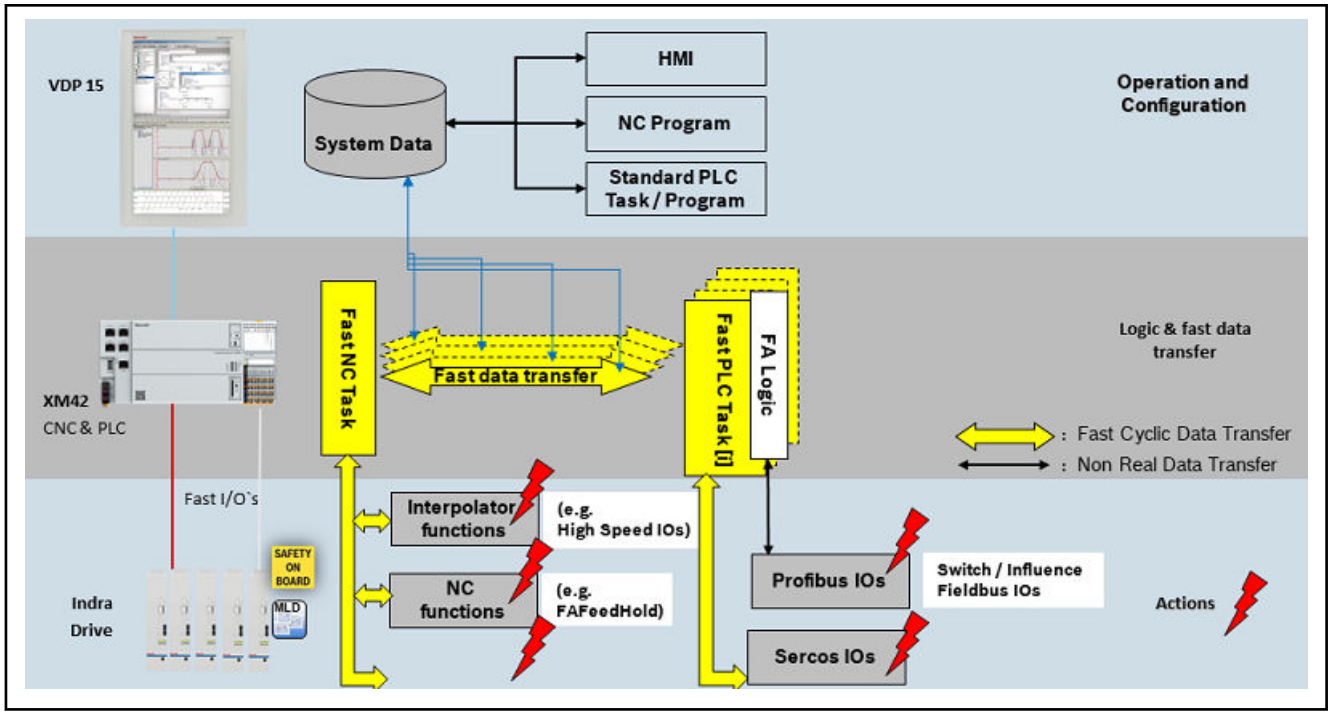


Fig. 5-2: Schematic representation of the data exchange via the FA task

The data structures of the inputs and outputs are defined in the configuration files "SDDefShapeCut.xml“, „SDDatFAShapeCut.xml" in "/usrfep" and "SDHeightCtrl.xsd" in "/usrfep/schemas". The structural element has to be selected from the allowed FA types. A maximum of 100 elements may be used in one FA task. This means that the number of inputs and outputs may not exceed 100.

☞ For detailed information on the FA data types and the general operating principle, see the documentation "Rexroth MTX Functional Description 15VRS Special Functions".

The default HC input data contains the axis positions, the sensor values, the axis coupling status and the HC mode. The HC output exports the height correction for the subsequent interpolator cycle. tab. 5-1 "Default inputs of the FA task" on page 28 and tab. 5-2 "Default outputs of the FA task" on page 29 show the FA types and the data types of the most important HC input and output data.

Tech function	Data	Variable	FA type	Data type	Length
HC	Command position of the z-axis	lrAxisPosCmd_Z	Axis position	LREAL	8 bytes
HC	Actual position of the z-axis (height)	lrAxisPosAct_Z	Axis position	LREAL	8 bytes
FC	ADSLab signal	udiAdvDistSig	System date	UDINT	4 bytes
FC	ADSLab validity	iADSValidity	System date	INT	2 bytes
FC	ADSLab remaining time	diADSRemTime	System date	DINT	4 bytes

Tech function	Data	Variable	FA type	Data type	Length
HC	Sensor value (analog input IndraDrive: P-0-0210)	iSensorVal	Sercos	INT	2 bytes
HC	Height control mode	udiHCMode	System date	UDINT	4 bytes
HC	Coupling status	usiCoupleSta	System date	USINT	1 byte
LF	LeapFrog trigger	rLeapfrogTrigger	System date	REAL	4 bytes
FC,LF,LP	Distance to go in the block	lrPathDistToGo	System date	LREAL	8 bytes
HC,FC,LF,LP	Path velocity	dwPathVelo	System date	DWORD	4 bytes
FC	Event status	bEvent	Event	BOOL	1 byte
LP	Laser Power Mode	udiSysPowerMode	System date	UDINT	4 bytes

HC Height control
FC FlyCut
LP Laser power control
LF LeapFrog

Tab. 5-1: Default inputs of the FA task

Tech function	Data	Variable	FA type	Data type	Length
HC	Height correction	rCorrValue	System date	REAL	4 bytes
HC	Status code	diExternalStatus	System date	DINT	4 bytes
LF	LeapFrog status	diLeapfrogSta	System date	REAL	4 bytes
LF	FA path override	rFeedOvrCmd	FAPathOvr	REAL	4 bytes

HC Height control
FC FlyCut
LP Laser power control
LF LeapFrog

Tab. 5-2: Default outputs of the FA task

The FA type "System date" (abbreviation: SD) describes xml-based data structure used to read and influence states or functions of the control during the operation. It provides specific data already structured and formatted for all system parts (see "Rexroth MTX Functional Description 15VRS Special Function", chapter 1.4 "Required and supplementing documentation MTX" on page 2)]. The used FA system data has to be declared in the file "SD*.xsd". The configuration of the FA task and HC inputs/outputs is explained in chapter 5.4 "Configuring the inputs and outputs of the FA" on page 30 using an example.

- The size of the structure for the input/output data has to be the same as the size of the data configured on the NC.
- The system data does not allow the use of the `DOUBLE` data type.

- When using the default FA inputs and outputs, the "SDDefShapeCut.xml", and "SDHeightCtrl.xsd" configuration files may **not** be changed.
- For the emulation, the FA input data "Measured values of the distance sensor" in the FA configuration file "SDDatFAShapeCut.xml" have to be configured as SD variable `SysHCSim`.
- For the machine implementation, the FA input data "Measured values of the distance sensor" in the FA configuration file "SDDatFAShapeCut.xml" has to be configured as "Analog-SercosData". The Sercos address depends on the hardware.

5.4 Configuring the inputs and outputs of the FA

The inputs and outputs of the Fast Action use the `MT_FADataRead` FA reading function and the `MT_FADataWrite` FA writing function. In fig. 5-3 "Implementation of the FA reading/writing functions to exchange data" on page 30, the data exchange is presented using an FA task. The data types of the inputs and outputs are defined as structures below the PLC folder **App ▸ Programs ▸ FA_PLC ▸ Types**. The two structures are presented in fig. 5-4 "Structure definition of the inputs and outputs of the FA" on page 30. One structure is the measured value (`iSensorVal` sensor value) and the other structure is the controller output value (`rCorrValue` height correction).

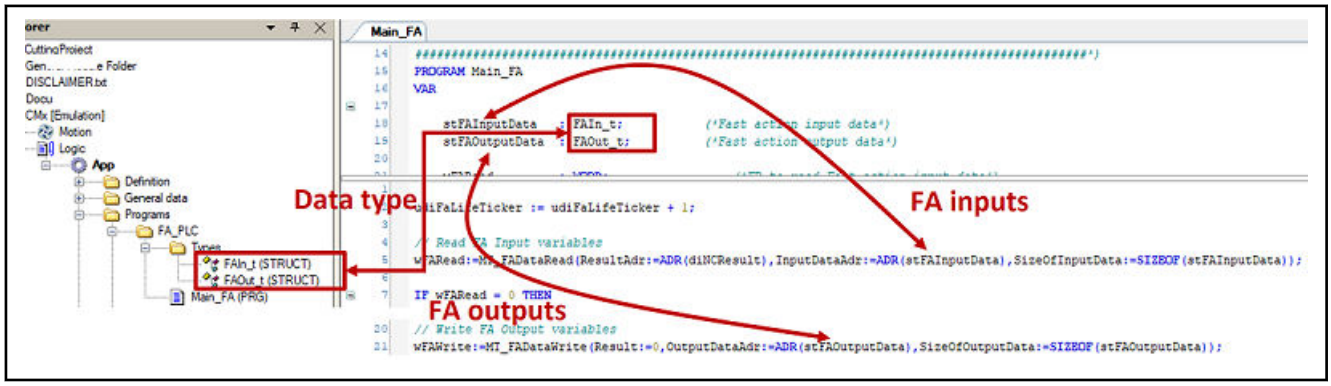


Fig. 5-3: Implementation of the FA reading/writing functions to exchange data

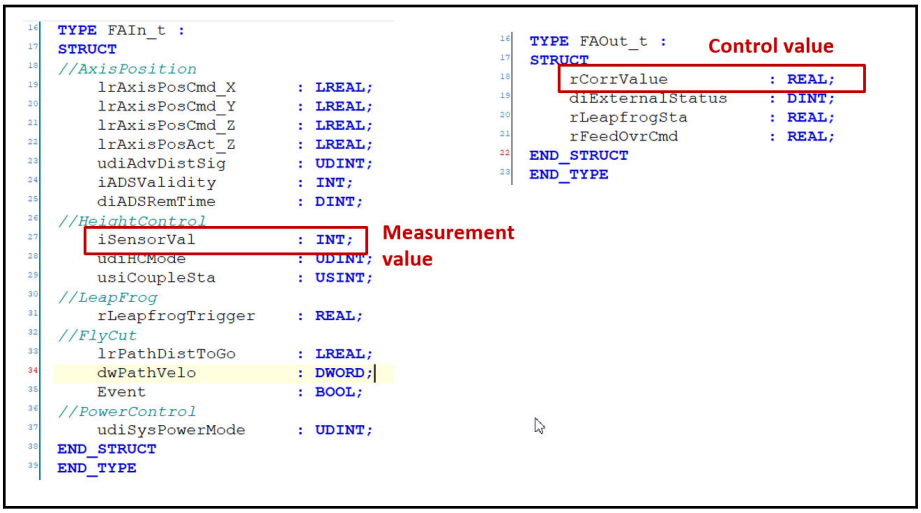


Fig. 5-4: Structure definition of the inputs and outputs of the FA

5.5 Adding a new output value to the system data

This chapter describes the adjustment of cyclic data transferred in the FA task. An adjustment may be necessary for example to add more input or output data or the required operating data for a second cutting head.

The inputs and outputs can be adjusted according to the requirements. In the following example, the difference between the nominal z-position and the real z-position as a new output value is added to the exchange data of the FA task. In this case, the original output value `wDigitalOutput_1` is deleted and a new element `rActToCmd` is created. The new output is defined as system date. As the use of the `LREAL` data type is not allowed by the FA system data, the element is defined as `REAL`. This restriction applies only to system variables. However, the use of LREAL is allowed if the axis coordinates are read. The fig. 5-5 "Example of adding an output" on page 31 shows the computation of the values before the FA writing function in the "Main_FA" FA main program and the definition of the new structure of the `FAOut_t` type.

FAOut_HC_t

```
20  TYPE FAOut_t :
21  STRUCT
22      rCorrValue          : REAL;
23      diExternalStatus    : DINT;
24      rLeapfrogSta        : REAL;
25      rFeedOvrCmd         : REAL;
26      rActToCmd           : REAL;
27  END_STRUCT
28  END_TYPE
```

Main_FA_HC

```
64  // Caculate the difference between actual and command position
65  stFAOutputData.rActToCmd  := LREAL_TO_REAL(stFAInputData.lrAxisPosAct_Z
66                                             - stFAInputData.lrAxisPosCmd_Z);
67
68  // Write FA Output variables
69  wFAWrite:=MT_FADataWrite(Result:=0,
70                           OutputDataAdr:=ADR(stFAOutputData),
71                           SizeOfOutputData:=SIZEOF(stFAOutputData));
```

Fig. 5-5: Example of adding an output

The FA inputs and outputs are configured on the NC by the "SDDefFA.xml", "SDDatFAHc.xml" and "SDHeightCtrl.xsd" files. To adapt the NC file, the project has to be switched online first before the file system of the control can be accessed.fig. 5-6 "Declaration of system data" on page 31 shows how the `SysHeightCtrl` system date used in the height control is declared as a `SysHeightControl_t` structure in "SDDefHC.xml".

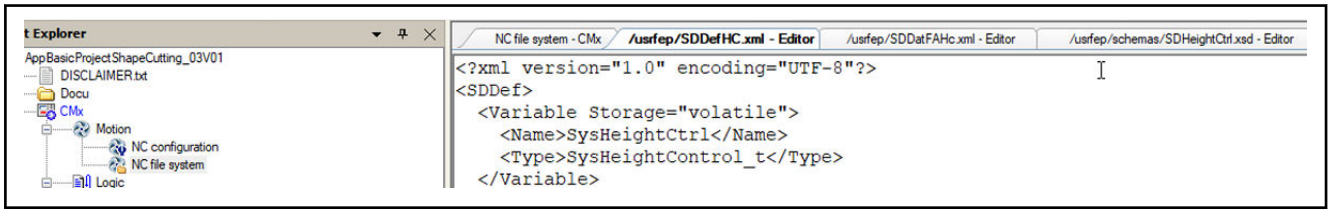


Fig. 5-6: Declaration of system data

The `SysHeightControl_t` structure is defined in the "SDHeightCtrl.xsd" file. When adding a system date, the new element has to be registered at this position. The "SDHeightCtrl.xsd" file is modified in fig. 5-7 "Adding a system date to the SDHeightCtrl.xsd file" on page 32. The PLC type `REAL` is indicated as the SD type `Float_t`.

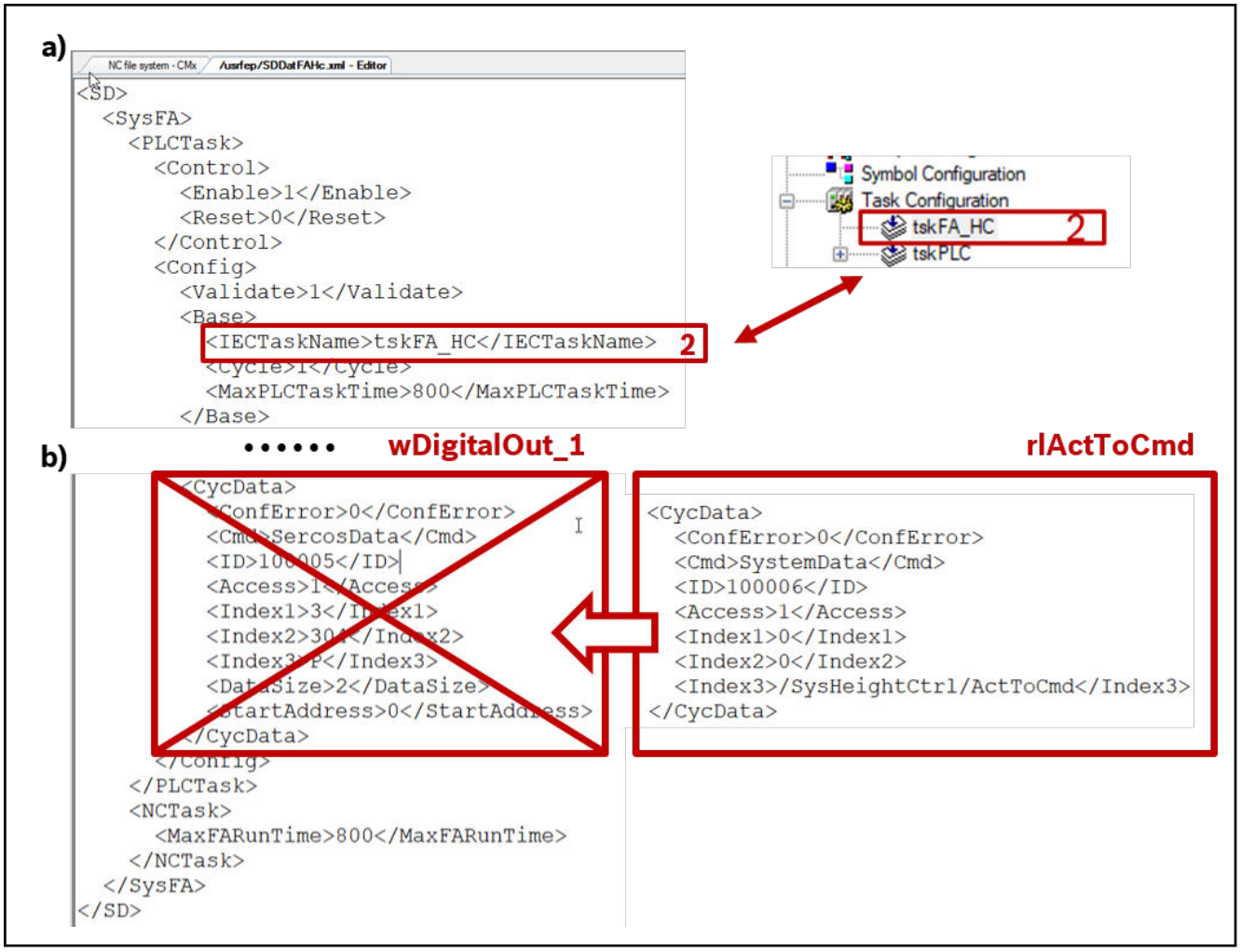
```
<xs:complexType name="SysHeightControl_t">
  <xs:annotation>
    <xs:documentation>Adaptive height control parameters</xs:documentation>
    <xs:documentation>Used by FA height control logic</xs:documentation>
    <xs:documentation>Autor: michluft, Bosch Rexroth AG</xs:documentation>
  </xs:annotation>

  <xs:sequence>
        ......
          <xs:element name="ActToCmd" type="Float_t">
              <xs:annotation>
                <xs:documentation>Defines a new parameters
                </xs:documentation>
              </xs:annotation>
      </xs:element>
     </xs:sequence>
   </xs:complexType>

 </xs:schema>
```

Added element for the new HC-Output

Fig. 5-7: Adding a system date to the "SDHeightCtrl.xsd" file

The file "SDDatFA.xml" defines all cyclically updated FA data on the NC. fig. 5-8 "Adding a system date in the "SDDatFA.xml" file" on page 33 a) shows the called FA task corresponding to the task name in chapter 5.2 "Main_FA program" on page 25.

The maximum number of exchanged elements is 100. The sequence and the data types have to completely match the PLC definitions in fig. 5-4 "Structure definition of the inputs and outputs of the FA" on page 30. In this example, the last `wDigitalOut_1` output was replaced by `rlActToCmd`. This change is made in fig. 5-8 "Adding a system date in the "SDDatFA.xml" file" on page 33 b).



a) Configuring the FA task
b) Changing an output

Fig. 5-8: Adding a system date in the "SDDatFA.xml" file

After the adjustments in the sections chapter 5.4 "Configuring the inputs and outputs of the FA" on page 30 and chapter 5.5 "Adding a new output value to the system data" on page 31, the control can be started with the new output.

☞ After changing these three files in the NC directory, restart the NC to apply the changes.

5.6 Error analysis with the NC analyzer

With the “NC Analyzer" tool in IndraWorks Operation, the FA exchange data (see 5-9) and the statistical setting data (see 5-10) of the HC can be viewed.

The data, the cycle time, the status of the FA task and the outputs are listed in the area **System Data ▸ SysFA**. The sequence of the PLC tasks is determined by the number of the `EVENT_FASTACTION_i`, i=1,2 external event (see Main_FA program, page 25).

If the `CycData` in 5-9 a) does not match the HC inputs and outputs in the PLC program, the HC FA task cannot be started. In the area **System Data ▸ SysFA ▸ Status/Info** of 5-9 b), the error codes of NC/PLC/User as well as the total data length for the HC inputs and outputs are shown. The FA error

codes are described in chapter 12 of the MTX Functional Description 15VRS - Special Functions (see Configuration, page 2).

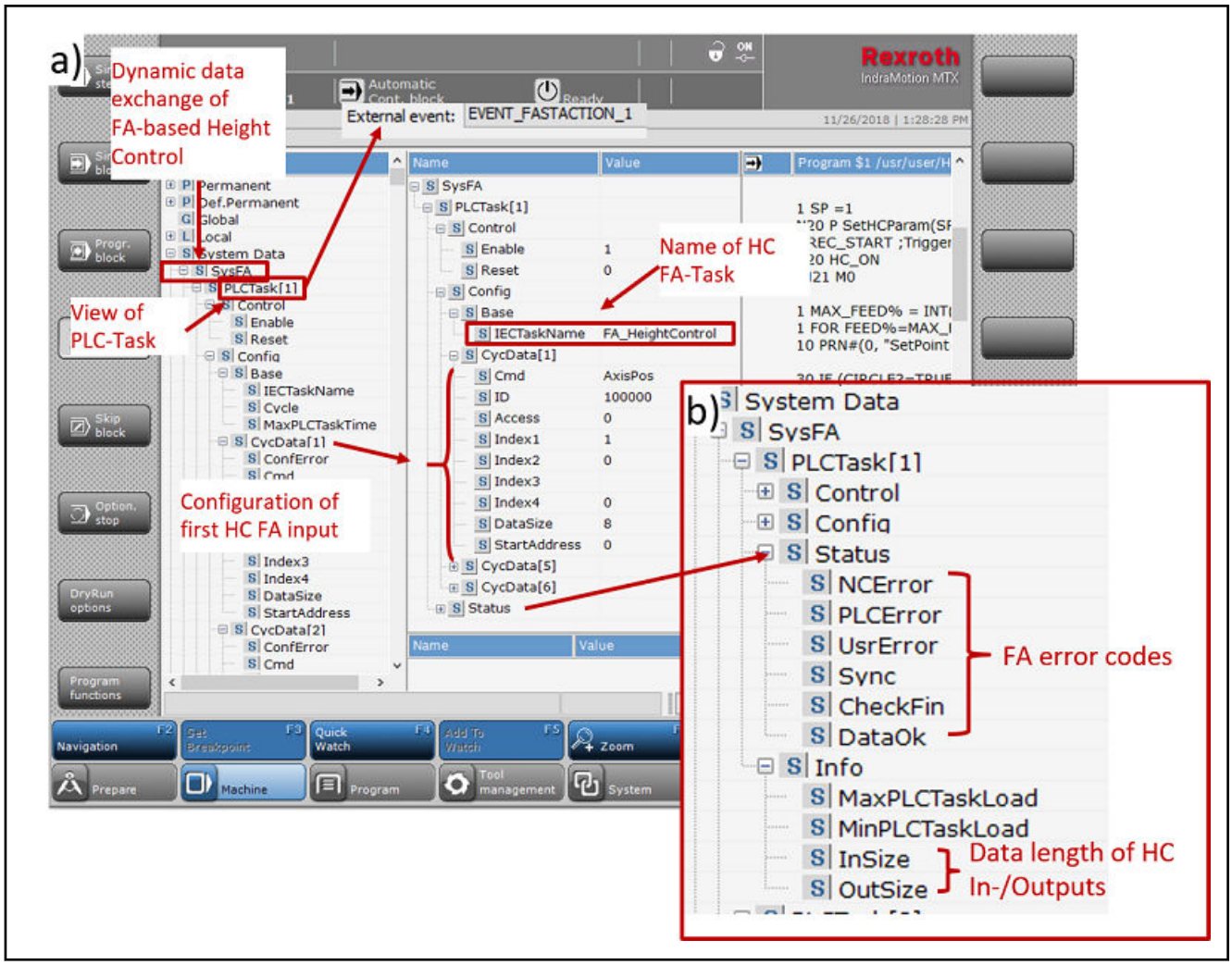


Fig. 5-9: Performing the "NC analysis" for the HC FA data exchange

The area **System Data ▸ SysHeightCtrl** in fig. 5-10 "Using the "NC analysis" for the HC settings" on page 35 shows the current settings of the HC that are helpful for the error analysis.

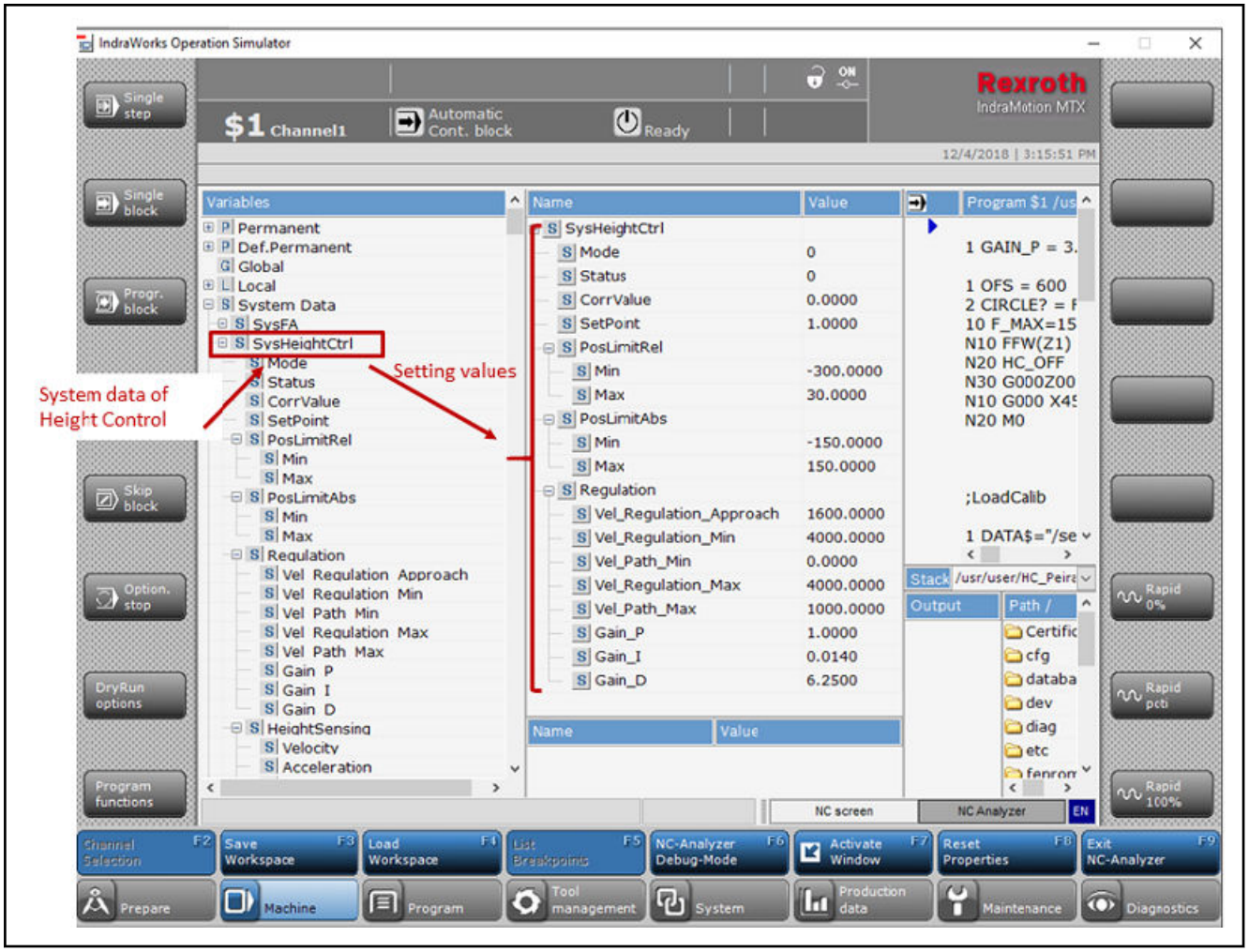


Fig. 5-10: Using the "NC analysis" for the HC settings

6 Height control

6.1 Overview

6.1.1 Characteristics of the height control

Different profiles are created using shape cutting procedures. The raw workpiece (sheet) shows manufacturing tolerances. Deformation due to the handling of sheets can also result. Furthermore, the processing table of the machine is not completely flat. This results in a required distance guidance of the machining head. It has to keep exactly a specific distance to the workpiece. With the height control (abbreviation: HC), the distance between the laser machining head and the material surface remains almost constant. This means that the position of the vertical coordinate automatically adapts to the workpiece surface while machining. The processing of slightly corrugated sheet is thus possible for example.

Compared to conventional laser processing without HC, the following advantages result:

- A collision is avoided between the material and the machining head
- The laser focus is not lost due to the material profile
- Reduced additional programming to follow up the vertical coordinate while machining a structured surface

To further increase the efficiency of the MTX system in laser processing, Bosch Rexroth provides an intelligent height control module and thus further advantages:

- Fast approach of the machining head to the material within the movement limits
- Jerk-limited transition between the approach of the machining head and the height control in the area closed by
- The automatic detection of gaps and cracks in the surface to be machined (gap detection) allows a vibration-free crossing and avoids collisions with the material.
- The automatic detection of the starting point offset of the material (PlateSearch)
- Avoiding an collision by detecting an imminent collision between the material and the machining head with corresponding evasive movement (Retract)
- Short settling time in milliseconds and high flexibility under customer-specific operating conditions
- Adaptability to various customer requirements with a configurable, flexible design
- Compatible with LeapFrog

The HC module uses the IndraWorks library (abbreviation: IW-Lib) "MT_Technology".

In order to execute the height control, the following hardware components are required:

- MTX control (XM42-S, XM42-A or XM42-P)
- Distance/position sensor with an analog output
- IndraDrive with an analog input or Sercos field bus I/O device with an analog input

The resulting settling time and the control deviation depend on the measurement duration and the resolution of the sensor used.

Furthermore, the following SW license is required:

R911396571: FWS-MTX-SHC2*****-NNVRS-NN-NNN

It includes the following licenses:

- R911320636: "Shape cutting 1" technology package
- R911337420: Fast Actions MTX fa Runtime
- R911318989: "Electronic gear" technology package

6.1.2 Control loop

An example of the height control concept with an analog sensor is shown in fig. 6-1 "Illustration of the Bosch Rexroth height control concept with an analog sensor" on page 38.

A mostly capacitive position sensor measures the distance between the cutting head and the material surface to be processed. The sensor converts the measured distance into an analog voltage signal and transfers it to the analog input of the drive controller (IndraDrive). Then, the measurement signals are converted into digital values by the AD converter integrated in IndraDrive and forwarded to the MTX control via Sercos which transfers the input values to the HC. There, the measured distance is compared to the predetermined set point and controlled accordingly. The vertical correction is issued to IndraDrive again and therefore moves the vertical coordinate.

Control processes and data exchange are performed in a fast PLC task (FA task). In this documentation, the predetermined nominal distance is referred to as "SetPoint" and the vertical direction as "z"-direction.

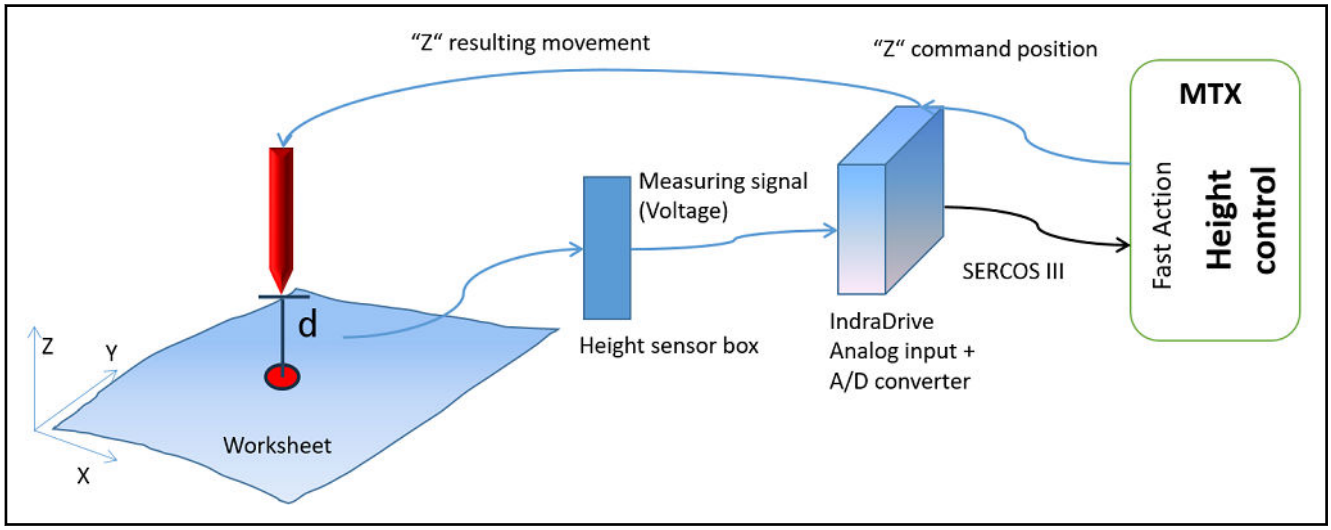


Fig. 6-1: Illustration of the Bosch Rexroth height control concept with an analog sensor

6.1.3 Special features of the height control

A great advantage of the intelligent height control is that the vertical travel speed as well as the control parameters are automatically adapted according to the current position and the horizontal feed velocity.

In order to realize this adaptive control, various HC states are defined for the controller parameterization. fig. 6-2 "States of the height control" on page

39 schematically shows the HC states depending on the height. The vertical work area is divided into a remote, center, near and collision area:

- Remote area: The sensor has not detected the material surface yet
- Center area: The sensor has already detected the material surface but is still far away from the SetPoint
- Immediate area: The sensor measuring value is in the area around the SetPoint
- Collision area: The sensor is below the SetPoint and therefore too close to the material surface

If the sensor is in the remote area, the z-axis moves towards material at full speed. The z-axis is decelerated if the sensor detects the material surface. To ensure a minimum settling time, the z-axis is first moved to the immediate area with a quick P controller. As soon as the cutting head has reached the SetPoint, the z-axis is precisely controlled with a configurable PID controller. If the cutting head enters the collision area, die z-axis moves away from the material at full speed. The machine stops with an error. Processing can be continued at a suitable point with the re-entry function (see Re-enter and Re-trace functionality, page 101).

The required settings as well as the calculation of the control parameters in the SetPoint, collision area and for the retreat path in case of imminent collision are described in the following sections.

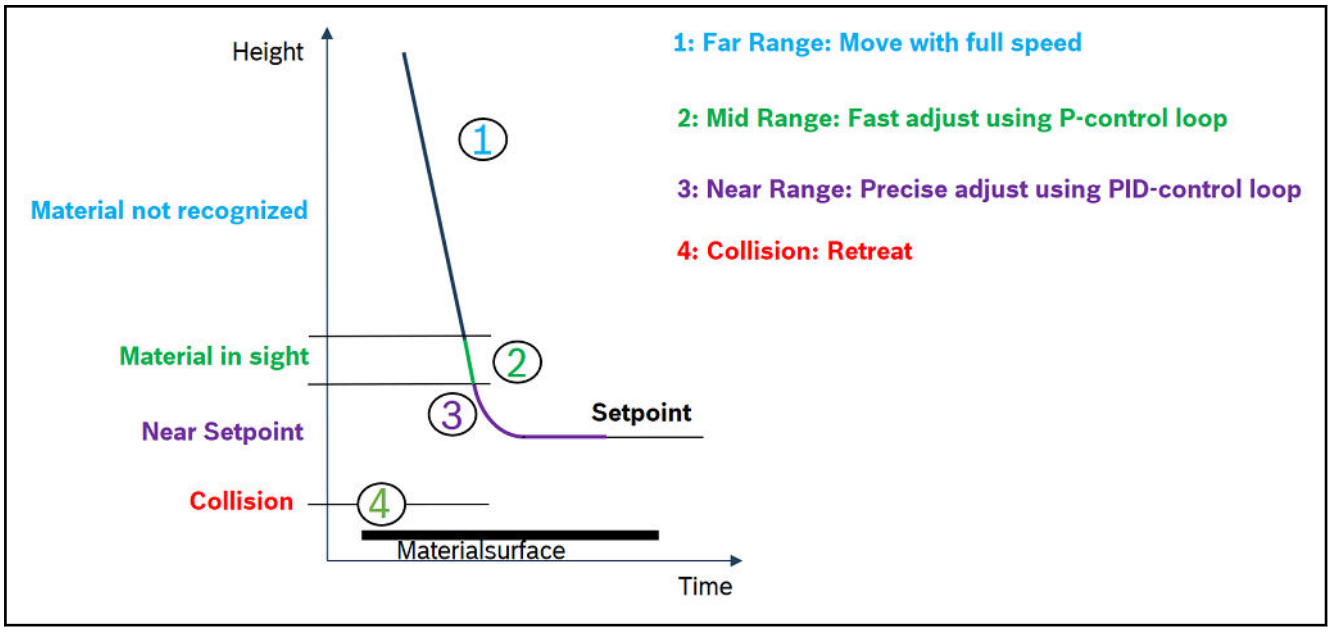


Fig. 6-2: States of the height control

6.1.4 Sensor selection

A distance sensor has to be installed before commissioning of the height control. In order to ensure a good control dynamism, the actualization frequency of the sensor should be higher than 1 kHz and the resolution should be higher than 0.1 mm. Furthermore, the sensor should have a measuring range between 0.1 and 35 mm. A non-contact capacitive sensor is recommended for our HC. In this document, commissioning is performed with a capacitance sensor with an analog interface.

6.2 Project module

6.2.1 Height control project module

All program parts of the height control module are located in the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ HeightControl**. It is divided into two program parts: "HeightControl_FA" and "HeightControl_PLC" (refer to fig. 6-3 "Storage place of the height control module within the shape cutting project" on page 40).

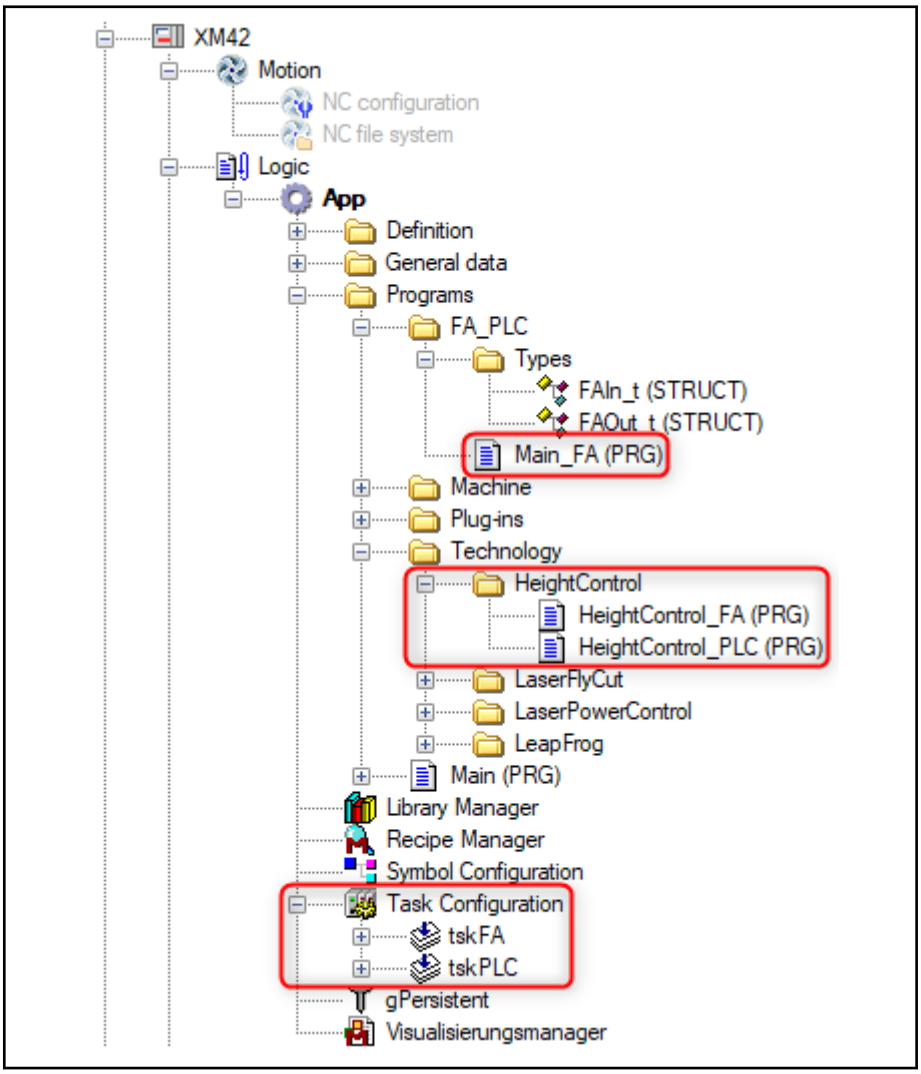


Fig. 6-3: Storage place of the height control module within the shape cutting project

HeightControl_FA

The program "HeightControl_FA" contains the actual control routine. It is started by the program "Main_FA" while executing the Fast Action task ("tskFA"). In the program, all cyclic data required for the height control is read and written after the computation of the height control.

The following variables are transferred cyclically between the NC and the PLC within the FA task:

Direction	Data	Variable	FA type	Data type	Length
Input	Command position of the z-axis	lrAxisPosCmd_Z	Axis position	LREAL	8 bytes
Input	Actual position of the z-axis (height)	lrAxisPosAct_Z	Axis position	LREAL	8 bytes
Input	Sensor value (analog input IndraDrive: P-0-0210)	iSensorVal	Sercos	INT	2 bytes
Input	Path velocity	dwPathVel	System date	DWORD	4 bytes
Output	Height correction	rCorrValue	System date	REAL	4 bytes
Output	Status code	diExternalStatus	System date	DINT	4 bytes

Tab. 6-1: Cyclic exchange data of the height control

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

Implementation of HeightControl_FA

The HeightControl_FA program can be adjusted process-specifically. The following function blocks can be called in the FA program:

- MT_HCSimType01 chapter 16.2.5 "MT_HCSimType01" on page 126: Simulation of the sensor values for the distance between material and machining head using the Z-axis position and the sensor calibration curve. Validation of height control in the simulation.
- MT_HCSimType02 chapter 16.2.6 "MT_HCSimType02" on page 128: Simulation of a material plate to validate the "PlateSearch" technology function in the simulation.
- MT_DistanceSensorType01 chapter 16.2.3 "MT_DistanceSensorType01" on page 121: Function block for general sensor calibration, loading of the calibration curve, filtering the sensor values
- MT_DistanceSensorType02 chapter 16.2.4 "MT_DistanceSensorType02" on page 124: Function block for "Precitec" sensor calibration and filtering of the sensor values
- MT_HeightRegulationType01 chapter 16.2.2 "MT_HeightRegulationType01" on page 117: Implementation of height control
- MT_PlateSearchType01 chapter 16.2.7 "MT_PlateSearchType01" on page 130: Implementation of the "PlateSearch" technology function
- MT_RaiseAxischapter 16.2.11 "MT_RaiseAxis" on page 138: Realizes fast lifting of cutter head in case of impending collision

Combination with LeapFrog

The height control can be used together with the LeapFrog. The LeapFrog function is described in chapter 8 "LeapFrog" on page 65.

During upward motion of the LeapFrog, the height control is switched to "Freeze Mode". The distance between the setpoint and the Freeze position "HeightLimiter" is specified by the "LeapFrogAddPosZ" of the Leapfrog function.

If the freeze mode is inactive, the downward motion of the z-axis is applied by the height control. The LeapFrog function block returns its current state at the "LeapFrogState" output.

HeightControl_PLC

The "HeightControl_PLC" program processes the auxiliary variables for reading sensor values. It is started by the cyclical task "tskPLC" which runs in a 10 ms cycle.

The following functions can be mapped using auxiliary variables:

- Applying the configuration data of the height control: M169

- Waiting for stabilization of the measured values: M170
- Resetting the calibration values: M171
- Reading the value pair (sensor value and distance) and saving to the calibration table: M172
- Reading the HC mode to enable/disable the height control: M174

6.2.2 Retract project module

During the laser beam cutting process, the machining head has to automatically be lifted when it comes into contact with material or in case of an external signal.

The retract module program is located in the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ RetractControl**. The folder contains the program: "RetractControl_FA".

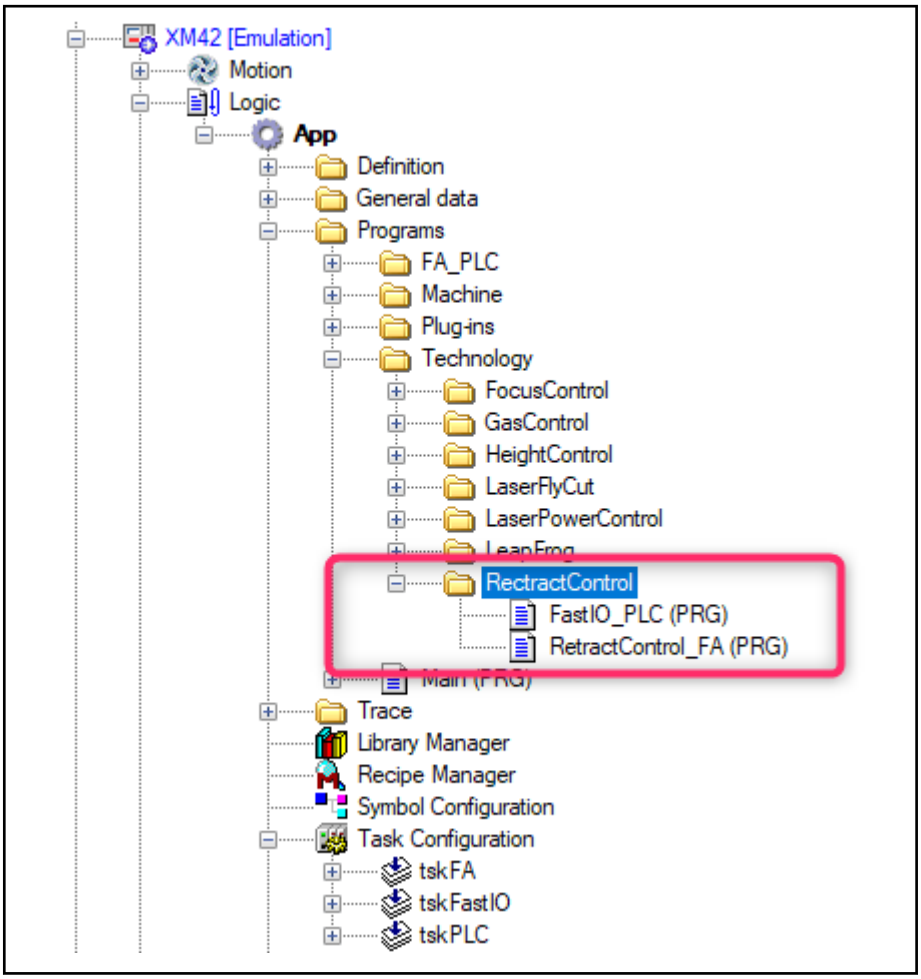


Fig. 6-4: Storage of the retract module within the laser beam cutting project

RetractControl_FA

The "RetractControl_FA" program contains the actual control routine. It is started by the program "Main_FA" while executing the Fast Action task ("tskFA"). In the program, all cyclic data required for retract is read and written after the computation of the height control.

The following variables are transferred cyclically between the NC and the PLC within the FA task:

Direction	Data	Variable	FA type	Data type	Length
Input	Command position of the z-axis	lrAxisPosCmd_Z	Axis position	LREAL	8 bytes
Input	Actual position of the z-axis (height)	lrAxisPosAct_Z	Axis position	LREAL	8 bytes
Input	Sensor value (analog input IndraDrive: P-0-0210)	iSensorVal	Sercos	INT	2 bytes
Output	FA path override	rFeedOvrFactor	FAPathOvr	REAL	4 bytes
Output	Height correction	rCorrValue	System date	REAL	4 bytes
Output	Status code	diExternalStatus	System date	DINT	4 bytes

Tab. 6-2: Cyclic exchange data of the height control

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

Implementation of RetractControl_FA

As the machining conditions differ, the "RetractControl_FA" program has to be adjusted process-specifically. The following measures can be included in the program:

- The retract motion is realized by the function block "MT_RaiseAxis"chapter 16.2.11 "MT_RaiseAxis" on page 138. The function block is enabled by the "RetractZ" signal
- The retract motion activation is triggered by the internal logic in case of material contact. This is detected by using the output signal of the height control "FAOutputData.diExternalStatus".
- The retract motion activation is triggered by an external signal, e.g.
- Retract height specification during cutting process (SD configuration: HeightControl_FA.fbSC.RetractWay) and during the LeapFrog process (constant: HeightLF)
- Enabling/disabling the laser using the "LaserEnable" signal
- Adjusting the feed motion in different process types using variable "CmdFeedrateOvr"

6.3 Commissioning the height control

6.3.1 Creating test program

After linking the project with the NC/PLC and pretesting the axes and the sensor, the HC module can be enables via NC commands. As shown in fig. 6-5 "Creating an NC program for the height control" on page 44, an example NC program ("Example NC program for height control:" on page 44) is created in the NC directory "/mnt/SimpleTest" for HC.

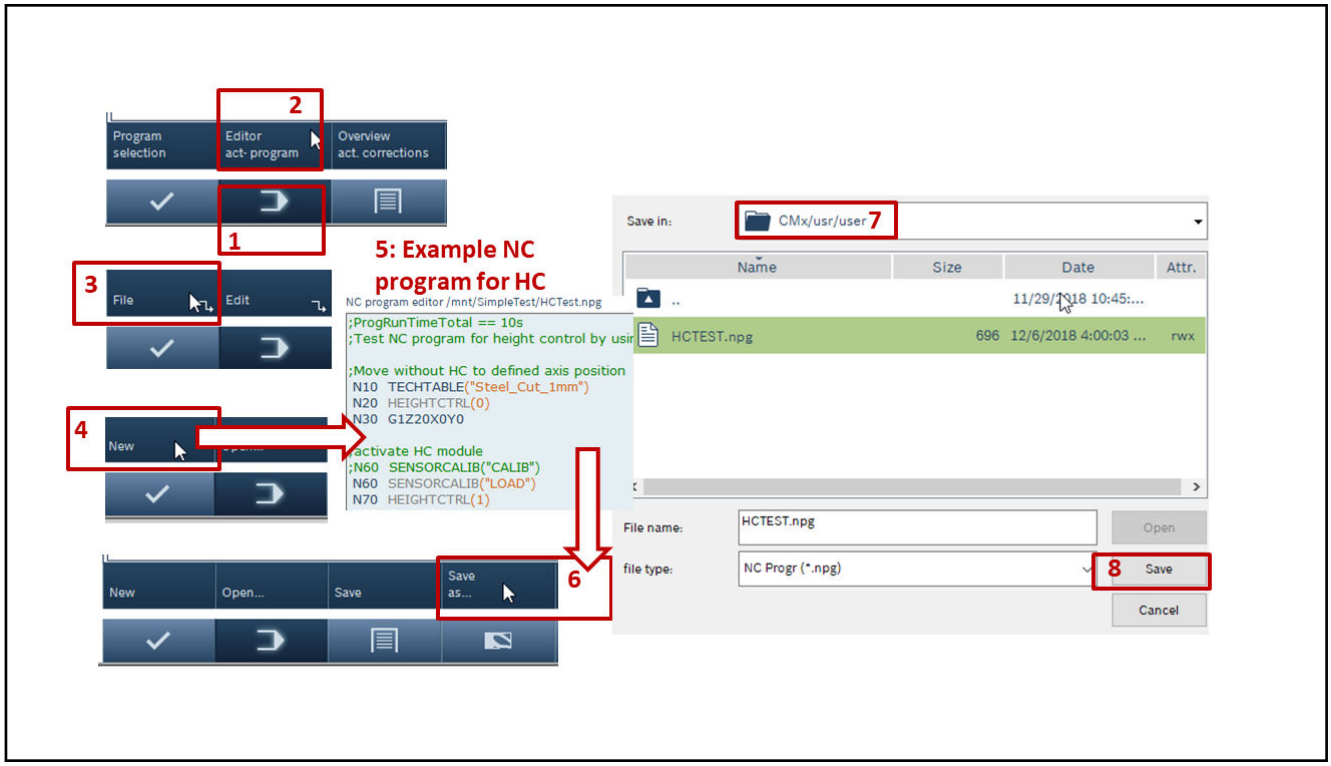


Fig. 6-5: Creating an NC program for the height control

Example NC program for height control:

```
;Move without HC to defined axis position
N10    TECHTABLE("Steel_Cut_1mm")  ;load config parameters (SetPoint, PID, ...)
N20    HEIGHTCTRL(0)               ;switch off HC
N30    G1Z0X0Y0                    ;move to reference position of z axis
N40    M0                          ;wait for PLC

;Activate HC module
;N50    SENSORCALIB("CALIB")       ;calibrate sensor
N50    SENSORCALIB("LOAD")         ;load sensor calibration

N60    HEIGHTCTRL(1)               ;switch on HC, SetPoint to 1mm
N70    G1X200Y200                  ;move to end position with speed 5000 mm/min
N80    HEIGHTCTRL(0)               ;switch off HC module
N90    G1Z0X0Y0                    ;move to reference position
```

Fig. 6-6: Test program

Execute the NC program in "Example NC program for height control:" on page 44 and the sensor is calibrated and the HC is parameterized and enabled or disabled:

- Call the NC program `SENSORCALIB` and the sensor is calibrated automatically. The calibration result is saved to the root directory under the file name "SensorCalib.txt".
- The height control is enabled and disabled via the `HEIGHTCTRL(1/0)` command.

6.3.2 Controller settings

The PID controller is set in several steps: These are described in the following. It is useful to store the determined parameters in the file `/usrfep/techtable/MACHINE` on the memory card of the control. Thus, they do not have to be stored each time in the technology table. All parameters set in the following are included in this file.

Setting the height control

*The limit values of the machine (max. velocities, max. acceleration, max. jerk) were entered into the NC configuration. 10 has to be set in the NC parameter **NC configuration ▸ Basic parameters ▸ CHAN ▸ Shape ▸ CHshapeG8**.*

1. Specifying the z-acceleration allowed

   The allowed acceleration is specified in the parameter `SD.SysHeightCtrl[*].Regulation.Acceleration_Max`:

   Example: `SD.SysHeightCtrl[*].Regulation.Acceleration_Max = 20: REM mm/S2`

2. Configuring the approaching velocity of the z-axis

   To set the approaching velocity of the cutting head from a greater distance, use the parameter `SD.SysHeightCtrl[*].Regulation.Vel_Approach`. Start with low values.

   Example: `SD.SysHeightCtrl[*].Regulation.Vel_Approach = 2000: REM mm/min`

3. Setting the P-value

   The P-value is set in the next step. The I-value and the D-value are set to 0. Lower values are safer, as they slow down the closed-loop control.

Program:

```
SD.SysHeightCtrl[*].Regulation.Gain_P = 0.01
SD.SysHeightCtrl[*].Regulation.Gain_I = 0       : REM ms
SD.SysHeightCtrl[*].Regulation.Gain_D = 0       : REM ms
SD.SysHeightCtrl[*].Regulation.Gain_PAPP = 0.01
```

NOTICE **If the P-value is too high or the I-/D-value is set incorrectly, the head can oscillate.**

Select lower values first and then slowly increase the P-value.

4. Checking the response

   Open Trace1_HCandLF under **XM42 ▸ Logic ▸ App ▸ Trace** and load it to the control. The trace shows the z-position of the head and the "HCInPos" output. If the z-axis is stable after enabling the HC, the approaching velocity as well as the P-value can be increased.

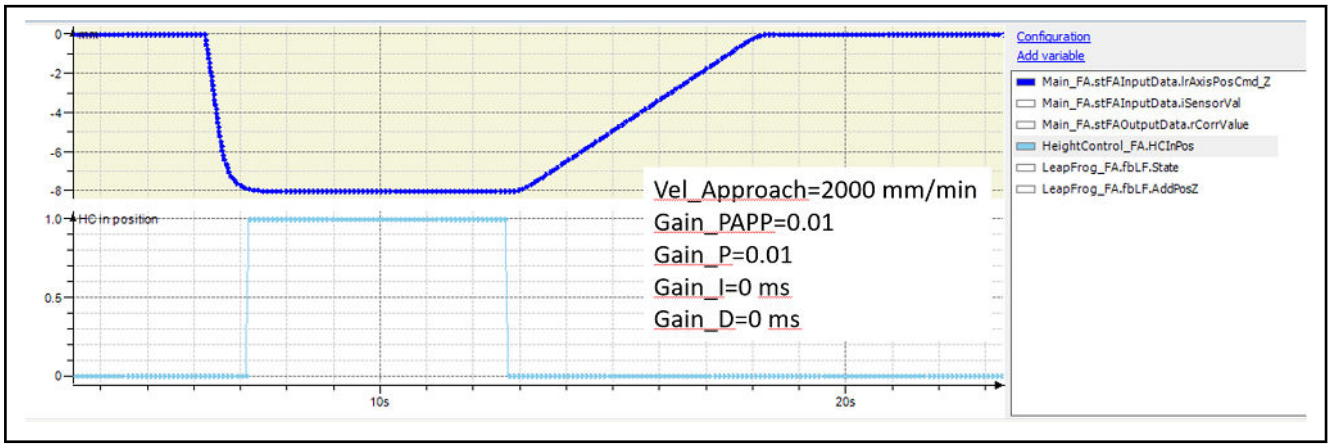


Fig. 6-7: PLC trace example when enabling the height control.

Change only one parameter at a time per attempt. Set the same value for `Gain_P` and `Gain_PAPP` in this step. The approaching velocity may not be beyond the set machine limits.

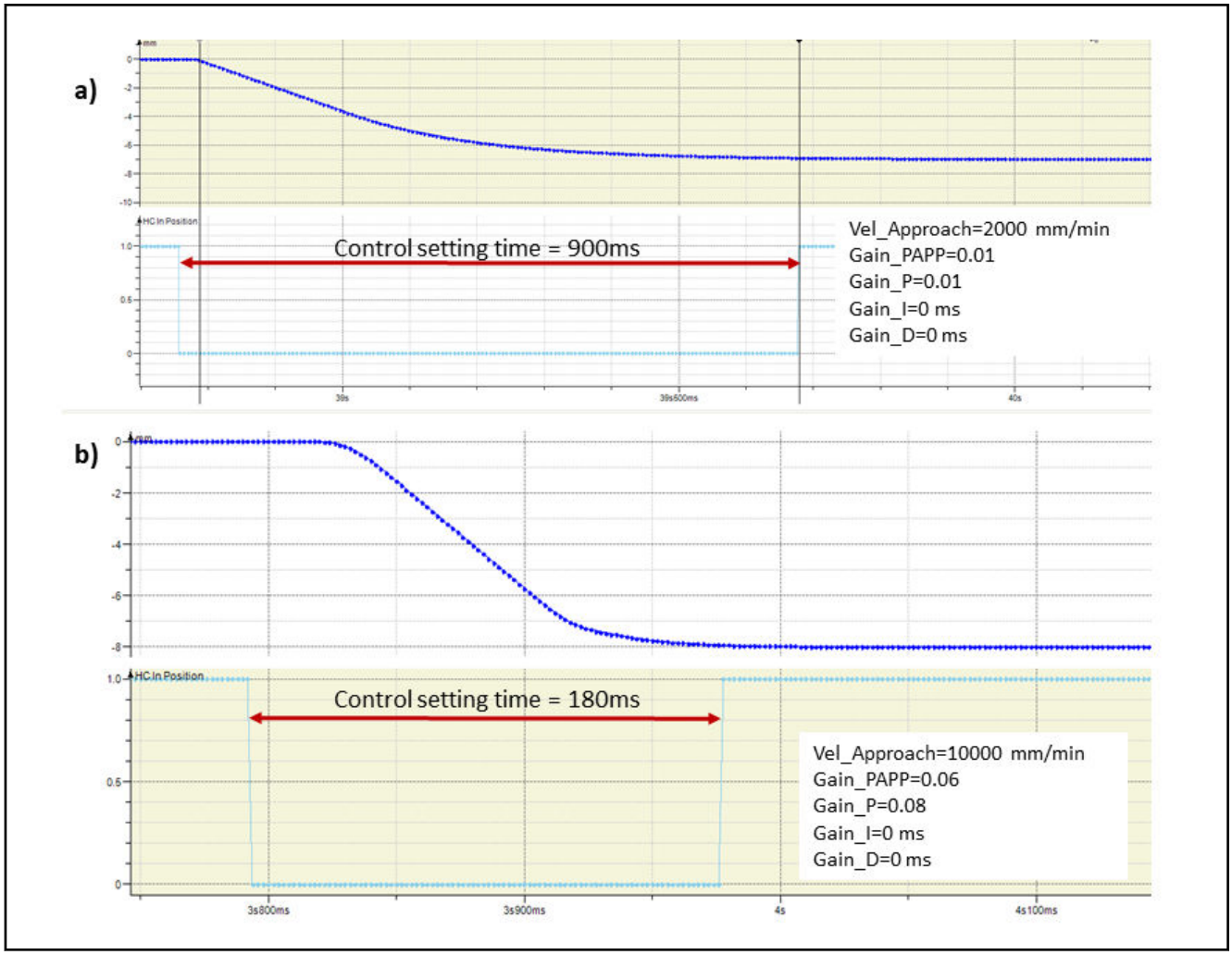


Fig. 6-8: Control response in non-optimized state a) and in optimized state b)

5. Setting< the I-value

   A control deviation remains if only the P-value is set. The cutting head will never reach the target position. This remaining deviation is compensated with the I-value. The control response can be optimized further at the same time. By experience, the optimum I-value is between the single and the double of the resonance frequency of the control settings. It is 20 Hz in this example.

   The I-value of the controller is set in ms. The 20 Hz mentioned before result in 50 ms. The double is 100 ms. If the resonance frequency is known, a good starting value can be determined. If the resonance frequency is unknown, try higher values. Decrease the values until an oscillation is present.

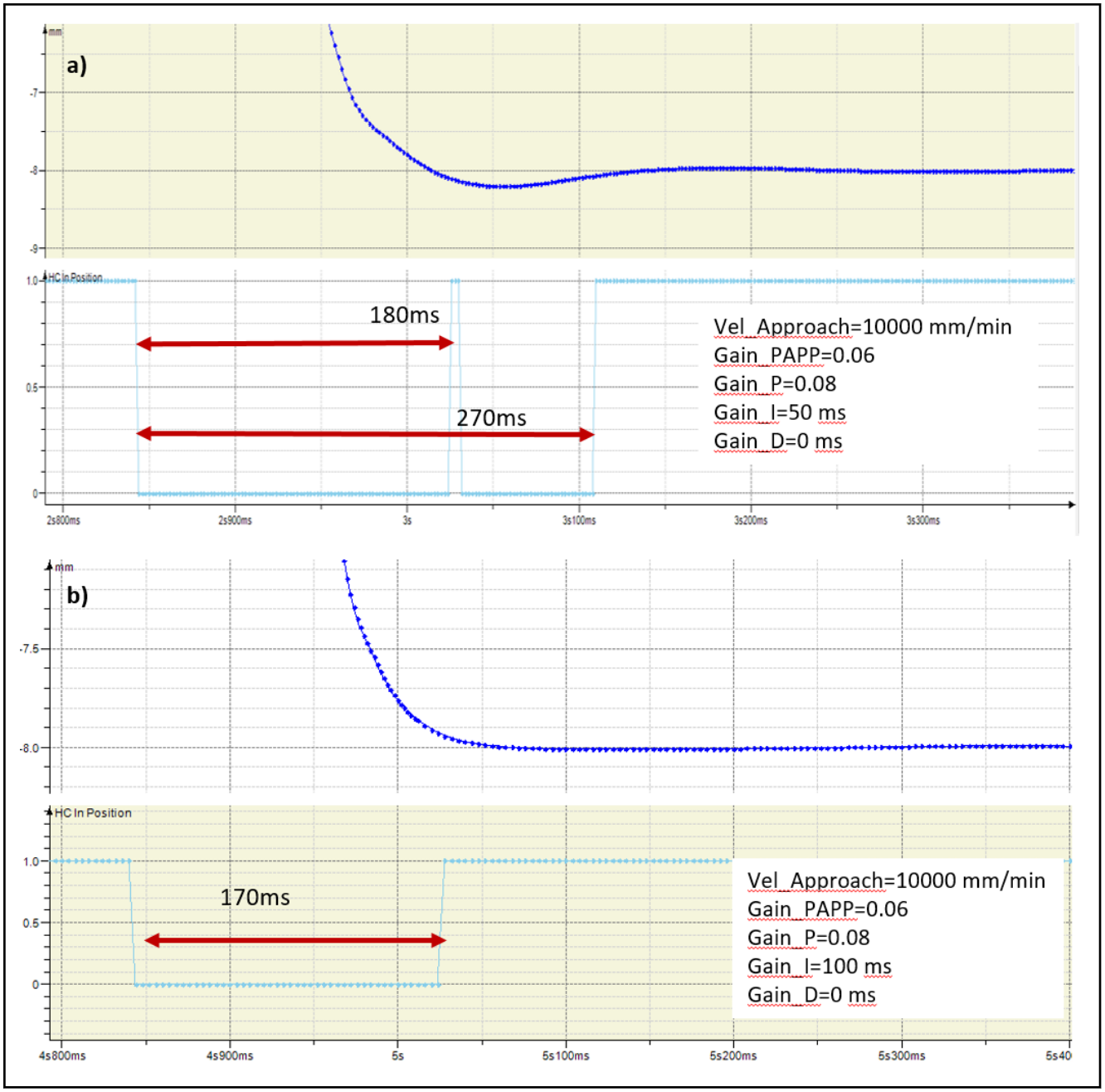


Fig. 6-9: Setting< the I-value

NOTICE **If the I-value is too low, the head can oscillate.**

Select higher values first and then slowly decrease the I-value.

The setting of the controller should now be at its best. When enabling the height control, the set SetPoint is reached after the fastest time possible. The closed-loop control should now be insensitive to interferences. If the control dynamics is insufficient, use the D-value for improving. In most cases, this is not required.

6.4 Control behavior of the height control

This chapter presents the control behavior of the height control using three examples.

1. Structured material surface

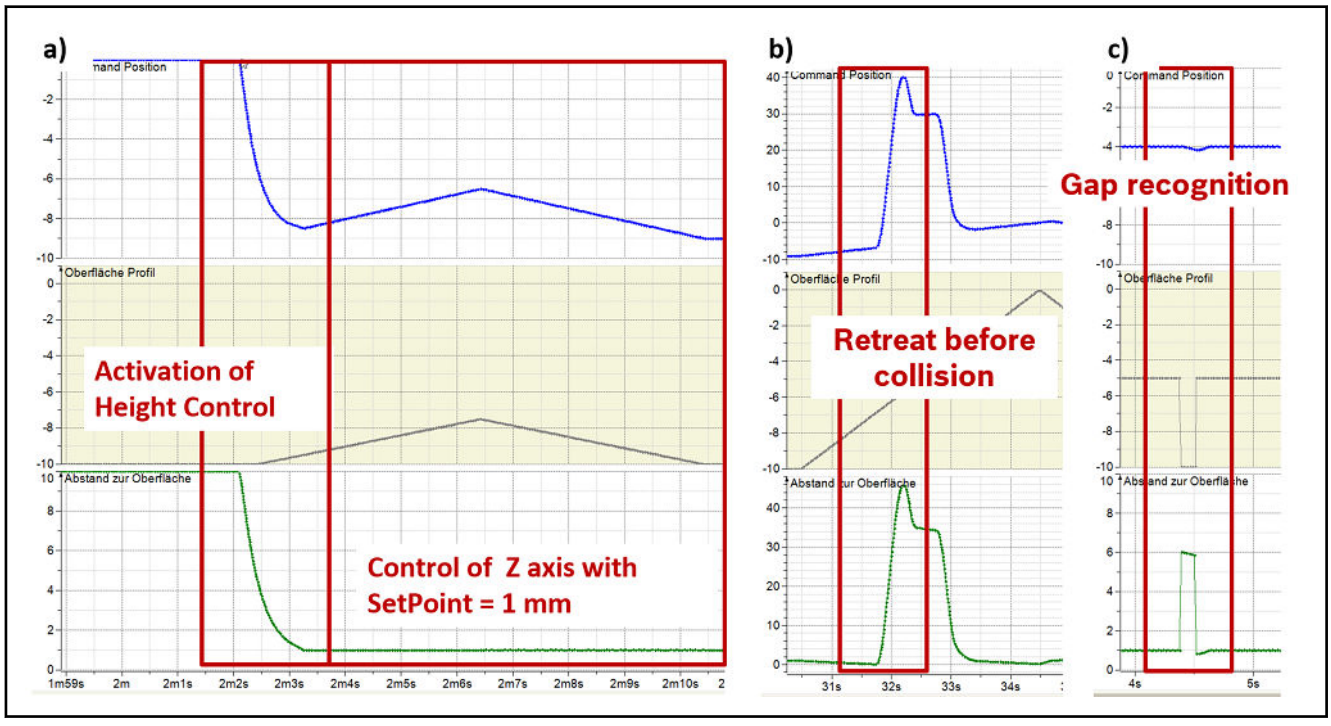
   fig. 6-10 "Three behavioral examples with SetPoint = 1 mm" on page 48 a) shows the movement of the cutting head on a structured material surface with height control. After activating the HC, the cutting head approaches the SetPoint. The controller then keeps a constant distance from the material profile. In this example, the predetermined SetPoint is 1 mm.

2. Collision avoidance

   fig. 6-10 "Three behavioral examples with SetPoint = 1 mm" on page 48 b) shows the control behavior for the retreat of the cutting head in case of imminent collision with the material. In practice, an unexpected steep increase of the surface is possible. In this case, the cutting head can be damaged by the stop on the material. In order to avoid collision, the retreat function of the HC is executed. If the cutting head enters the collision area, e.g. 10 % of the SetPoint, the head is immediately lifted to a defined height.

3. Gap recognition

   Gaps or cracks can be detected with the HC. In this case, the cutting head does not follow the material gap downwards but holds the current position and thus moves directly over it. The execution of the gap recognition function is shown in fig. 6-10 "Three behavioral examples with SetPoint = 1 mm" on page 48 c).



a) for the structured material surface
b) Collision avoidance
c) Detection of a material crack

Fig. 6-10: Three behavioral examples with SetPoint = 1 mm

6.5 Setting parameters of the height control

The setting parameters of the height control are defined in the system data. In general, they are written while loading the technology tables (see Technology tables, page 105).

Parameter	Unit	Description
SD.SysHeightCtrl[*].SetPoint	mm	SetPoint, nominal distance between the machining head and the material surface
SD.SysHeightCtrl[*].Regulation.Vel_Approach	mm/min	Approaching velocity of the machining head
SD.SysHeightCtrl[*].Regulation.Vel_Regulation_Max	mm/min	Upper limit of the travel velocity in z-direction
SD.SysHeightCtrl[*].Regulation.Acceleration_Max	mm/s²	Max. acceleration allowed
SD.SysHeightCtrl[*].Regulation.Gain_P		Controller parameters: P-value
SD.SysHeightCtrl[*].Regulation.Gain_I		Controller parameters: I-value
SD.SysHeightCtrl[*].Regulation.Gain_D		Controller parameters: D-value
SD.SysHeightCtrl[*].Regulation.Gain_PAPP		Controller parameters: P-controller for the approach
SD.SysHeightCtrl[*].HeightSensing.OvershootRatio	%	Maximum overshooting permitted: (Earth position - MaxAmpl.) / (Earth position - SetPoint) 0: No overshooting permitted
SD.SysHeightCtrl[*].HeightSensing.InPosTime	ms	In-Pos-Time
SD.SysHeightCtrl[*].HeightSensing.InPosWindow	mm	In-Pos-Window
SD.SysHeightCtrl[*].HeightSensing.InPosTimeout	ms	In-Pos-Timeout
SD.SysHeightCtrl[*].PosLimitRel.Min	mm	Workspace parameters: Lower limit of the relative motion
SD.SysHeightCtrl[*].PosLimitRel.Max	mm	Workspace parameters: Upper limit of the relative motion
SD.SysHeightCtrl[*].PosLimitAbs.Min	mm	Lower limit of the absolute position
SD.SysHeightCtrl[*].PosLimitAbs.Max	mm	Upper limit of the absolute position
SD.SysHeightCtrl[*].Sensor.OutOfRangeValue	mm	Out-of-range value, maximum measuring range of the sensor
SD.SysHeightCtrl[*].Sensor.CalibPoints		Number of the measuring points for the sensor calibration
SD.SysHeightCtrl[*].Sensor.CalibFilename		File name to store the calibration file
SD.SysHeightCtrl[*].Sensor.CollisionSensorVal		Sensor value for collision detection
SD.SysHeightCtrl[*].Collision.Mode		1: Collision detection by pre-contact; other specification is reserved
SD.SysHeightCtrl[*].Collision.RetractWay	mm	Retract path during collision

Parameter	Unit	Description
SD.SysHeightCtrl[*].Collision.PreCollisionDist	mm	Pre-contact Limit of the collision area in mm. Setting range: 0.0 < PreCollisionDist < SetPoint
SD.SysHeightCtrl[*].Collision.Delay	ms	Time after which a collision was detected before a retract motion is initiated
SD.SysHeightCtrl[*].Gapdetection.Inclination	degree	Inclination of the gap edge: acttan(z_2-z_1)/(xy_2-xy_1)
SD.SysHeightCtrl[*].Gapdetection.Delay	ms	Time after which a gap/edge is detected before a retract motion is inititated
SD.SysHeightCtrl[*].Gapdetection.DistUp	mm	Retract distance after detecting a gap/an edge

* Channel number

Tab. 6-3: Setting parameters of the height control

6.6 NC commands for the height control

6.6.1 Overview

In the previous chapter, the height control was used in an example program. The HC module is operated with special NC commands. The definition of the NC commands is stored in the NC directory "feprom/shapecut/" (cf. Commissioning the NC, page 14). After switching the IndraWorks shape cutting project online, the NC directory can be accessed.

In this chapter, the following NC commands of the height control are presented:

- `SENSORCALIB` sensor calibration command, see Sensor calibration: SENSORCALIB, page 50
- PreciTech sensor calibration command `PREC16P_CALIB`/`PREC16P_LOAD`, seechapter 6.6.5 "Sensor calibration: PREC16P_CALIB" on page 52
- `HEIGHTCTRL` height control command, see Enabling/disabling the height control HEIGHTCTRL, page 51
- PlateSearch command `PLATESEARCH`, see chapter 6.6.6 "PlateSearch" on page 53

6.6.2 Sensor calibration: SENSORCALIB

The calibration of a general distance sensor is loaded or performed again with the `SENSORCALIB` cycle. In this case, specify the global variable `usiSensorType` in **Logic** ▸ **App** ▸ **General data** ▸ **Global variables** ▸ **Specific** ▸ **gSpecific** `usiSensorType` as `1`.

The command has the parameter: "Mode". It accepts the "LOAD" or "CALIB" options that determine whether to load an already existing calibration (`Mode="LOAD"`) or perform a new calibration (`Mode="CALIB"`).

Parameterization

The SENSORCALIB cycle uses the following settings for the calibration which are usually defined by the machine settings in the technology tables (see chapter 15 "Technology tables" on page 105):

- `SD.SysHeightCtrl[*].Sensor.OutOfRangeValue`
- `SD.SysHeightCtrl[*].Sensor.CalibPoints`
- `SD.SysHeightCtrl[*].Sensor.Calibfilename`

`*` determines the channel number. This number is usually 1.

Performing the sensor calibration

The sensor calibration is started with the macro`SENSORCALIB("CALIB")`. The calibration of the sensor and the subsequent check of the measured values are performed automatically. The result is stored on the file system of the control. Then, the sensor values are already stored in the memory and do not have to be reloaded.

Loading the sensor calibration

Loading a calibration is started with the `SENSORCALIB("LOAD")` call. The values stored in the calibration file during the preceding calibration run are retransferred to the control memory.

6.6.3 Structure of the sensor calibration file

The calibration file contains the assignment of a sensor measuring value to a specific sensor distance. Each line contains a pair of values, the first value corresponding to the real distance and the second value corresponding to the sensor measuring value for this distance.

The file is read and written by the SENSORCALIB NC cycle (see chapter 6.6.2 "Sensor calibration: SENSORCALIB" on page 50). The storage of the file in the data system of the control is defined by the `SD.SysHeightCtrl[*].Sensor.Calibfilename` system date. In this case, `*` corresponds to the channel number.

The sensor calibration processes in section chapter 6.6.2 "Sensor calibration: SENSORCALIB" on page 50 are summarized in "Two methods for the sensor calibration in the NC program" on page 51 in an exemplary manner.

Two methods for the sensor calibration in the NC program

```
SENSORCALIB("CALIB")  ;automatic calibration

SENSORCALIB("LOAD")   ;load calibration file
```

Example

```
  0.000     80
  0.050    125
  0.100    160
  0.150    189
  0.200    214
  0.250    252
  0.300    291
  0.350    323
  0.400    352
  0.450    380
  0.500    403
  0.550    424
...
 18.600   9510
 18.750   9571
 18.900   9608
 19.050   9669
 19.200   9745
 19.350   9818
 19.500   9862
 19.650   9959
 19.800   9986
 19.950  10000
```

6.6.4 Enabling/disabling the height control HEIGHTCTRL

The commands to enable or disable the height control are `HEIGHTCTRL(1)` and `HEIGHTCTRL(0)`:

Program:

```
N10 TECHTABLE("Steel_cut_1mm")    ; Technologietabelle laden
N20 HEIGHTCTRL(1)                 ; Einschalten der HC
N30 HEIGHTCTRL(0)                 ; Ausschalten der HC
```

The TECHTABLE command loads the technology table and therefore all settings for the height control. They are transferred to the height control by calling the M169 M-function internally.

The HC-enabling and -disabling commands contain subfunctions for the axis coupling and the HC initialization. Execute HEIGHTCTRL(1) and the axes used are coupled. Then, the HCMode system date is set to 1 to activate the height control.

If the HEIGHTCTRL(0) command is called, the axes are decoupled and the HCMode system date is reset to 0. The cutting head is lifted to a safe position and the HC is disabled.

6.6.5 Sensor calibration: PREC16P_CALIB

The 16-point Precitec sensor calibration is loaded with the PREC16P_LOAD cycle or executed again with PREC16P_CALIB. For the specific I/O signals of the Precitec sensor, go to the Precitec documentation "TD_EG8030_BA_EN". In this case, the PLC has to set the global variable usiSensorType in **Logic ▸ App ▸ General data ▸ Global variables ▸ Specific ▸ gSpecific** as 2

The positions of the 16 points are configured by the PLC global structure usiSensorType declared as type ST_CalibRepository_t in **Logic ▸ App ▸ General data ▸ Global variables ▸ Specific ▸ gSpecific**. 16 calibration points are configured in the following for the sensor with a measured range of 15 mm. The structure "stCalibInit" can save up to 8 calibration curves "SensorCurve"*(calibration index: 0 to 7)* as well as the relevant 16 calibration reference position "SensorcurveArray" (position index: 1 to 16) and 1 zero position (position index 0).

HeightControl_FA.PRECSensorInit

```
stCalibInit.SensorCurve[0].SensorCurveArray[0].Way := 0.0;
stCalibInit.SensorCurve[0].SensorCurveArray[1].Way := 0.2;
stCalibInit.SensorCurve[0].SensorCurveArray[2].Way := 0.5;
stCalibInit.SensorCurve[0].SensorCurveArray[3].Way := 0.7;
stCalibInit.SensorCurve[0].SensorCurveArray[4].Way := 1.0;
stCalibInit.SensorCurve[0].SensorCurveArray[5].Way := 1.2;
stCalibInit.SensorCurve[0].SensorCurveArray[6].Way := 1.5;
stCalibInit.SensorCurve[0].SensorCurveArray[7].Way := 1.8;
stCalibInit.SensorCurve[0].SensorCurveArray[8].Way := 2.0;
stCalibInit.SensorCurve[0].SensorCurveArray[9].Way := 2.5;
stCalibInit.SensorCurve[0].SensorCurveArray[10].Way := 3.0;
stCalibInit.SensorCurve[0].SensorCurveArray[11].Way := 4.0;
stCalibInit.SensorCurve[0].SensorCurveArray[12].Way := 6.0;
stCalibInit.SensorCurve[0].SensorCurveArray[13].Way := 8.0;
stCalibInit.SensorCurve[0].SensorCurveArray[14].Way := 10.0;
stCalibInit.SensorCurve[0].SensorCurveArray[15].Way := 12.0;
stCalibInit.SensorCurve[0].SensorCurveArray[16].Way := 15.0;
```

Fig. 6-11: Configuration of the 16-calibration sensor of the Precitec sensor with a measured range of 15 mm

Parameterization

The PREC16P_CALIB cycle uses the following settings for this calibration which are usually defined by the machine settings in the technology tables (see Technology tables, page 105):

- SD.SysHeightCtrl[*].Sensor.OutOfRangeValue
- SD.SysHeightCtrl[*].Sensor.CalibPoints**

* determines the channel number. This number is usually 1.

** The CalibPoints of the 16-point calibration is 16.

Performing the sensor calibration

The Precitec sensor calibration is started with the PREC16P_CALIB call and executed automatically. The result is output as PLC structure in the type ST_CalibRepository_t.

Then, the sensor values are already stored in the memory and do not have to be reloaded.

Loading the sensor calibration

Loading a calibration is started with the `PREC16P_LOAD` call. The values stored in the calibration repository during the preceding calibration run are retransferred to the control memory.

6.6.6 PlateSearch

Command to automatically determine the position and orientation of the rectangular material according to `PlateSearch`. Calling the command set the PLC global variable `usiSensorType` in **Logic ▸ App ▸ General data ▸ Global variables ▸ Specific ▸ gSpecific** as `TRUE`.

Parameterization

PlateSearch has the following settings which are also used for the gap detection. The parameters are generally specified via the machine setting in the technology tables (see Technology tables, page 105).

- `SD.SysHeightCtrl[*].Gapdetection.Inclination`: Inclination of the edges, *acttan(z2-z1)/(xy2-xy1)*
- `SD.SysHeightCtrl[*].Gapdetection.Delay`: Time after which a retract motion is initiated if an edge is detected
- `SD.SysHeightCtrl[*].Gapdetection.DistUp`: Retract distance after edge detection

`*` determines the channel number. This number is usually 1.

Executing PlateSearch

The process is executed automatically when calling the NC program `PlateSearchProg`. To determine the position and the angle of rotation, the machining head is traveled twice over two adjacent edges. The following example program describes the process to find the edges. The measurement result is output in the system data:

- `SD.PlateSearch[*].OffsetX`: Distance between the determined zero point of the material and the zero position in MCS in X-direction `Offset_X`
- `SD.PlateSearch[*].OffsetY`: Distance between the determined zero point of the material and the zero position in MCS in Y-direction `Offset_Y`
- `SD.PlateSearch[*].RotationsAngle`: Angle of rotation of the material

Example program of the PlateSearch

```
; Find WCS with detection of two edges
; Defination a simulated material surface
STOCK("BLOCK",500,500,1)
; Load cutting parameter table, 70ms
N10 TECHTABLE("Steel_Cut_1mm")
N20 G1X60Y60Z30
N30 M175                        ; Initial plate search
 10 CURFEED% =SD(5,1,2)         ; Save the current feedrate
N40 SENSORCALIB("LOAD")         ; Load Calibration
N50 HEIGHTCTRL(1)               ; Height control enable
N60 PLATESEARCH                 ; Plate search enable
N100 Y=IC(-100) F2000           ; Move over edge 1 to calculate point 1
N110 X=IC(10)
N120 Y=IC(110)                  ; Move over edge 1 to calculate point 2
N130 X=IC(-110)                 ; Move over edge 2 to calculate point 3
N140 Y=IC(-10)
N150 X=IC(100)                  ; Move over edge 2 to calculate point 4
N200 HEIGHTCTRL(0)
N210 M176                       ; Transfer the Plate offset to SD Variable
N220 F[CURFEED%]                ; Turn to the normal feedrate
```

“Lane” of the PlateSearch

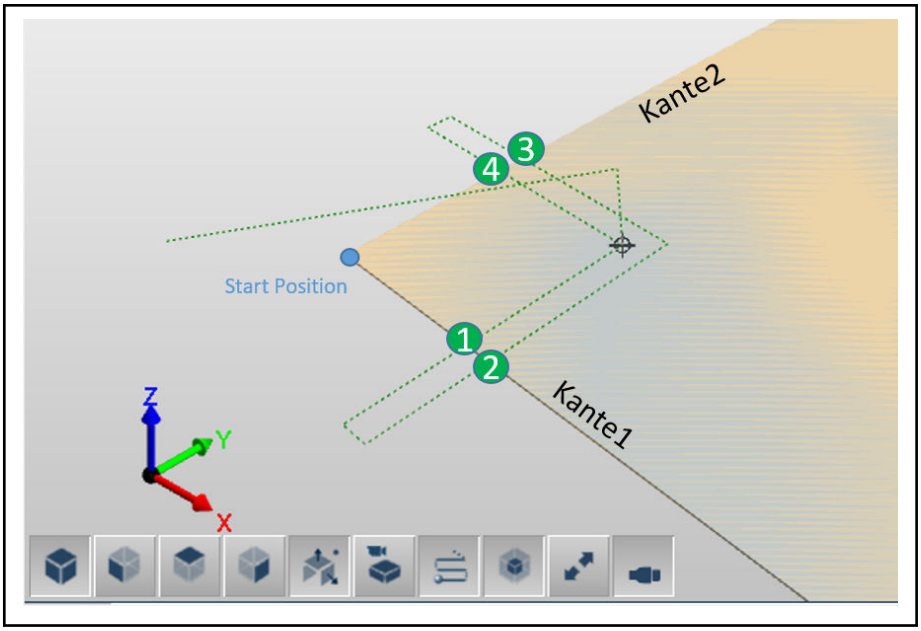


Fig. 6-12:

6.7 Troubleshooting

- **Problem:** The height control cannot be started

  Cause: The `.xml` and `.xsd` files belonging to the HC as well as the subroutines are missing in the NC directory.

  Remedy: Recovery of the TAR files with subsequent NC restart (see also 4-6, page 16).

- **Problem:** The FA task for HC does not run after logging in and starting.

  Cause: The configured inputs and outputs of the FA task have a wrong data length or are parameterized incorrectly.

  Remedy:

  – Check that the FA inputs/outputs in the PLC program match the ones on the NC side (see also chapter 5.4 "Configuring the inputs and outputs of the FA" on page 30)

  – The analysis can also be performed with the NC analyzer. Please check the area **System Data ▸ SysFA ▸ Status/Info**. (see also chapter 5.6 "Error analysis with the NC analyzer" on page 33)

- **Problem:** The `iSensorVal` PLC variable in the HC main program shows incorrect sensor values.

  Cause: Sensor is not connected correctly and no material is within sensor range

  Remedy:

  – Check the sensor connection

  – Check the configuration of the cyclic data traffic

  – Check that the sensor is installed and calibrated correctly

  – Manually move the sensor to 5 mm above the material surface

- **Problem:** The z-axis is unstable if the HC is active

  Cause: The feed forward is not activated or the control parameters are not set correctly

Remedy:

– Activate the feed forward of the axes with the `FFW` command

– Optimize the controller parameters in the "/usrfep/techtable/MACHINE" file (fig. 4-8 "Storage location of the technology tables and the MACHINE file" on page 18)

- **Problem:** The FA task does not deliver values for the height control.

  Cause: The "MT_FADataRead" FA function is not zero.

  Remedy: This error message is described in the "Rexroth MTX Functional Description 15VRS Special Functions" documentation in the chapter about the fast FA task. Troubleshooting can be performed with the NC analyzer in chapter 5.6 "Error analysis with the NC analyzer" on page 33.

7 FlyCut

7.1 Introduction

FlyCut is a process during which the cutting device (e.g. a laser) is switched on and off in sync to the position at the highest possible velocity of motion. FlyCut allows, for example, cutting grids without running every rectangle individually. Instead, the attempt is made to run all edges in a straight line once and to control the laser accordingly. This is also possible for round and curved trajectories.

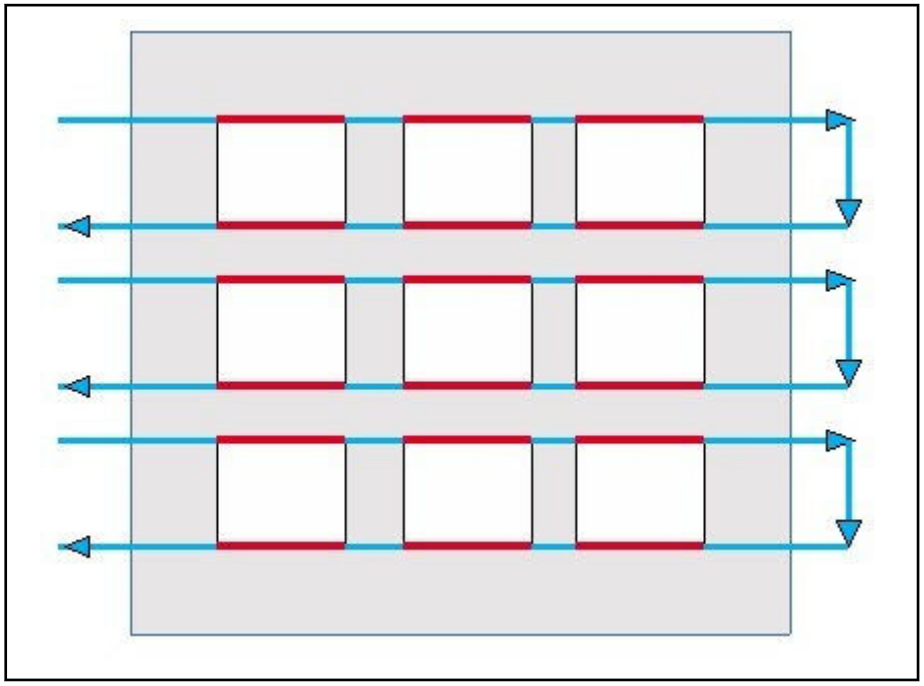


Fig. 7-1: FlyCut principle of operation

Due to the high path velocity, the laser has to be controlled highly synchronous to the position. In order to compensate the signal propagation delay and control-internal delays, controlling has to be done with a certain lead. The functions described below ensure the synchronization with the position and the required lead.

7.2 Variants

The FlyCut functionality was implemented differently. Each functionality has advantages and disadvantages depending on the application. There is one implementation based on the NC and two implementations based on the PLC running via the Fast Action task.

All variants use the `CUT()` call. This command can trigger up to 8 different signals per NC channel that control the laser. Each signal can be parameterized individually (see Parameterization, page 61).

Implementation into the NC

The macros reference to the system data `SD.SysPowerCtrl[SD(8)].LaserCtrlEvent`. This system data calls the NC commands `ADSENA()` and `ADSLAB()` internally. The ADS function is parameterized via the SD variables `SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].*`. The resolution is one NC or one Sercos cycle. The digital outputs of the drive are available as outputs.

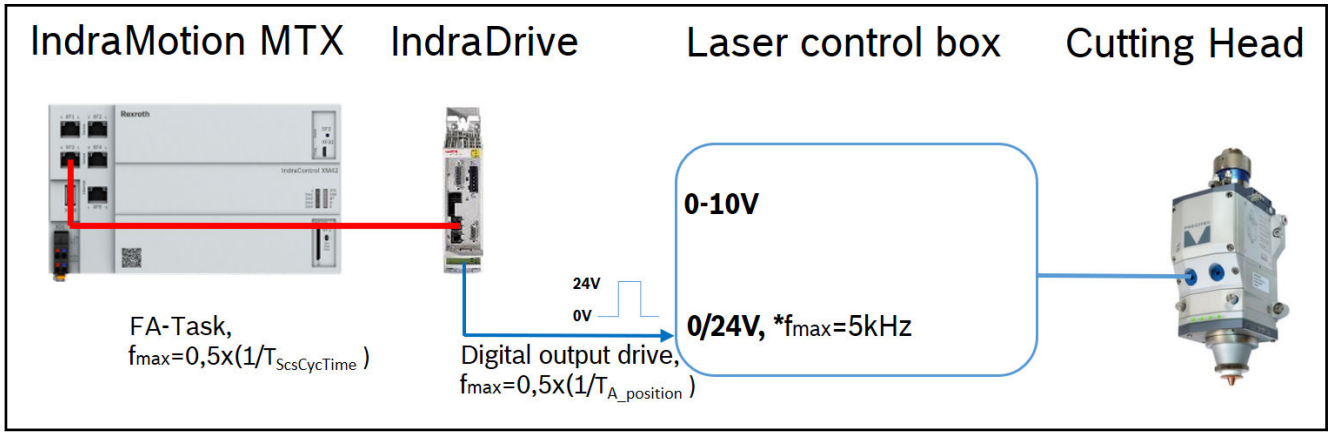


Fig. 7-2: System architecture to use the NC-based implementation

MT_FlyCutType01

This function block implements the FlyCut function in a PLC function block. It is mainly called in the Fast Action task. The resolution is one NC or one Sercos cycle. All Sercos I/O devices as well as I/Os at the local S20 bus of the control are available as outputs.

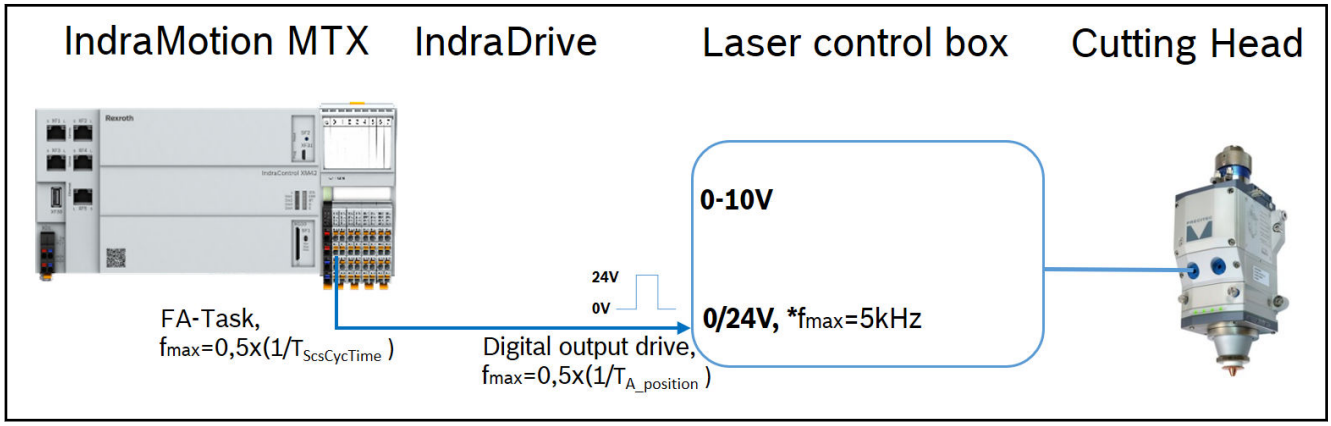


Fig. 7-3: System architecture to use the MT_FlyCutType01 function block

MT_FlyCutType02

The MT_FlyCutType02 function block also implements the FlyCut functionality into a PLC function block. Unlike the MT_FlyCutType01, this function block is used with the S20-DIDO-2/1-OVS high-speed oversampling module. It is called in a Fast Action task. The resolution is up to 10 µs.

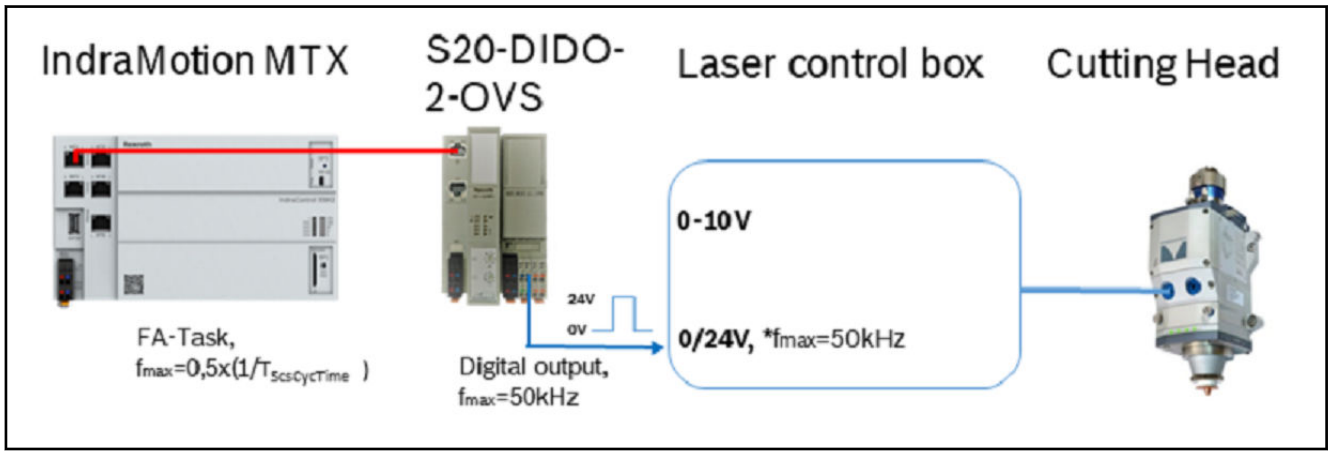


Fig. 7-4: System architecture to use the MT_FlyCutType02 function block

Functional selection

	NC	MT_FlyCutType01	MT_FlyCutType02
Required resolution	≥ 1 ms	≥ 1 ms	< 1 ms
Switching lead time	> 5 ms	> 2 ms	> 1 ms
Sercos cycle time	≤ 1 ms	≤ 1 ms	≥ 1 ms
Cutting velocity	≤ 30000 mm/min	≤ 30000 mm/min	≥ 30000 mm/min

Tab. 7-1: Selection criteria for different implementations

7.3 FlyCut project module

All program parts of the FlyCut module are located under the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ FlyCut**. It is divided into two program parts: "FlyCut_FA" and "FlyCut_PLC"

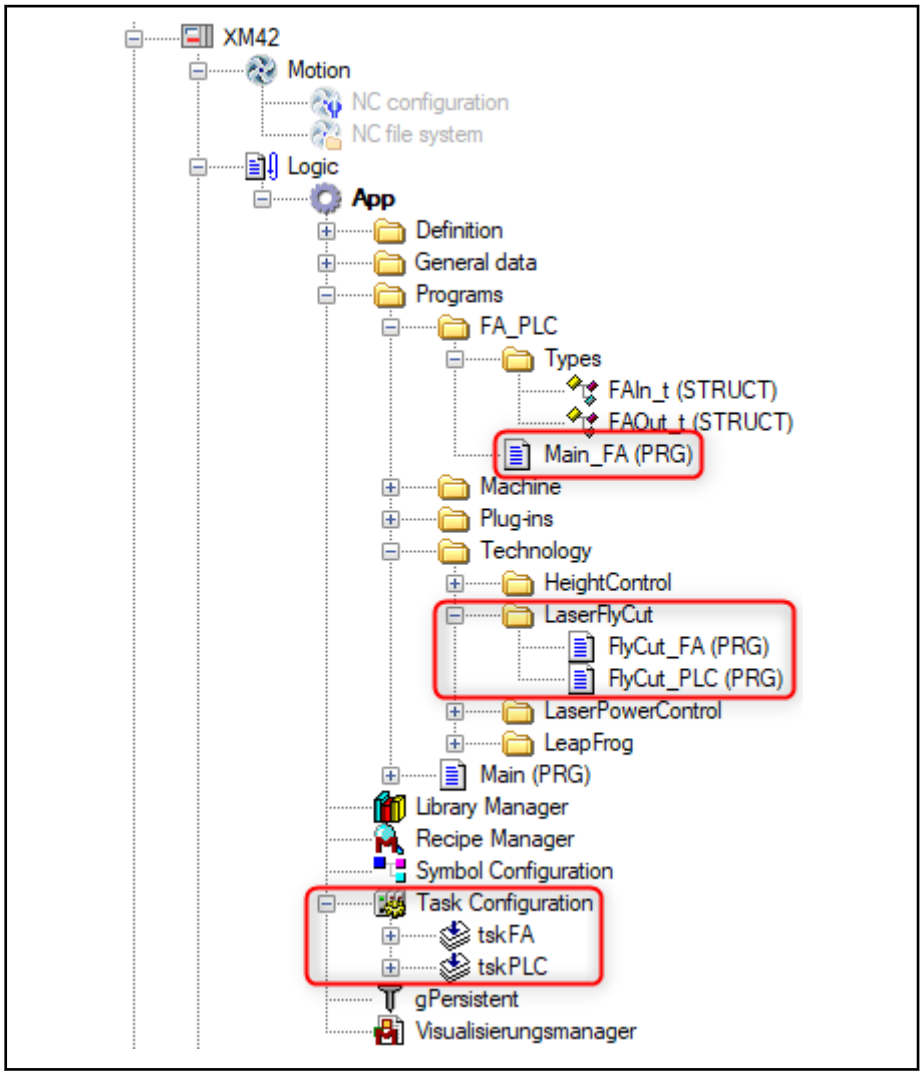


Fig. 7-5: Storage location of the FlyCut module within the shape cutting project

FlyCut_FA

The "FlyCut_FA" program contains the routine for switching the cutting jet on or off. It is started by the program "Main_FA" while executing the Fast Action task ("tskFA"). Within the program, all cyclic data required to execute the FlyCut functionality is read. After the switching operation has been computed, it is written to the corresponding outputs.

To trigger the FlyCut functionality, call the `CUT` command in the NC code. It uses the NC command `ADSLAB` internally to determine the distance to go up to the switching position. Input variables send the distance to go and the target condition to the FA task.

The following variables are transferred cyclically between the NC and the PLC within the FA task:

Direction	Data	Variable	FA type	Data type	Length
Input	Signal status from ADSLAB	udiAdvDistSig	System date	UDINT	8 bytes
Input	Validation of the ADSLAPB time slot	iADSValidity	System date	INT	2 bytes
Input	Remaining time	diADSRemTime	System date	DINT	4 bytes
Input	Distance to go in the block	lrPathDistToGo	System date	LREAL	8 bytes
Input	Path velocity	dwPathVelo	System date	DWORD	4 bytes
Input	Event	bEvent	Write event	BOOL	1 byte

Tab. 7-2: Cyclic exchange data of FlyCut

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

FlyCut_PLC

The "FlyCut_PLC" program processes the auxiliary variables to read configuration values. It is started by the cyclic task "tskPLC" which runs in a 10 ms cycle.

To send the configuration values to the PLC, call the `TECHTABLE` command. It uses the auxiliary variables `M210` internally to apply the configuration to the PLC.

Configuration of the oversampling module

The following configurations are required to use the OVS module:

- User-defined parameters

  An example, the settings for channel 1 are represented with a Sercos cycle time of 1 ms and 0.5 ms.

  The OVS sampling time for a Sercos cycle time of 1 ms is 1 ms / 64 = 15.6 µs.

  The OVS sampling time for a Sercos cycle time of 1 ms is 0.5 ms / 50 = 10 µs.

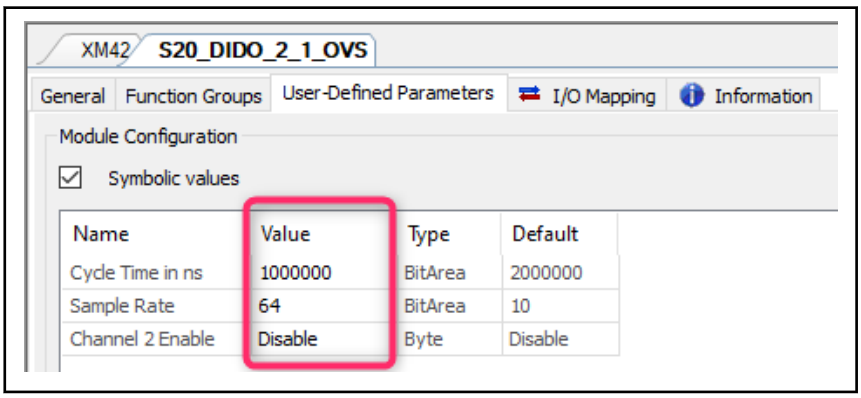


Fig. 7-6: OVS setting for channel 1 with a Sercos cycle time of 1 ms

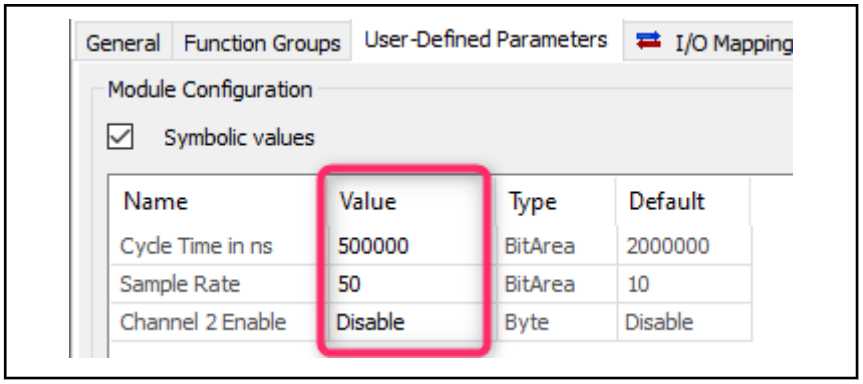


Fig. 7-7: OVS setting for channel 1 with a Sercos cycle time of 0.5 ms

- Configuration of the XM42 control

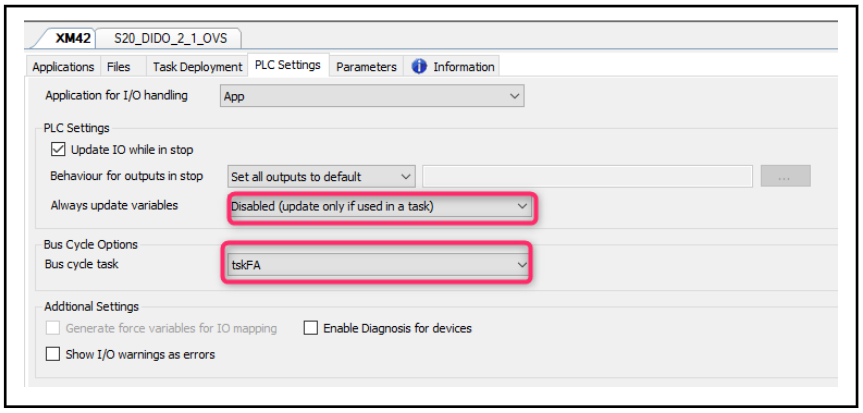


Fig. 7-8: OVS setting for the XM42 control

7.4 Parameterization

The FlyCut functionality is parameterized in the NC via system data. When using the MT_FlyCutType01 and MT_FlyCutType02 PLC function blocks, it is possible to use the inputs of the function blocks.

The switch-on and switch-off time can be set path-based and time-based. This way, the switching time can be set to the technology used and the response behavior of the cutting head. The parameters usually have to be determined experimentally.

The following system data is intended for use with FlyCut:

System variable	Minimum	Maximum	Unit	Description
`SD.SysFlyCut[*].Mode`	0	1		0, ADSLAB; 1, PWM or OVS
`SD.SysFlyCut[*].DistOn`	0.0	Path velocity * remaining machining time	mm	Indicates the switch-on distance in mm relative to the end point of the NC block
`SD.SysFlyCut[*].DistOff`	0.0	Path velocity * remaining machining time	mm	Indicates the switch-off distance in mm relative to the end point of the NC block

System variable	Minimum	Maximum	Unit	Description
SD.SysFlyCut[*].TimeOn	0.0	4.0	ms	Indicates the switch-on point in time in ms relative to the end point of the NC block
SD.SysFlyCut[*].TimeOff	0.0	4.0	ms	Indicates the switch-off point in time in ms relative to the end point of the NC block

* NC channel, normally 1

Tab. 7-3: FlyCut functionality settings

The **remaining machining time** is calculated by the ADS function and is saved in the SD variable: `SysAdvDistSigA[*].RemTime`. The default value is `4.0 ms`.

The **path velocity** results from FA input: `udiPathVel` in `mm/100*s`.

Example: the maximum DistOn and DistOff `SD.SysFlyCut[*].DistOn = 4.0*udiPathVel/10^5`

7.5 Commissioning

To set the parameters, proceed as follows:

The following preconditions have to be met:

- *The drive bus is running*
- *The PLC is loaded and started*
- *The height control has already been commissioned*
- *The NC program is loaded for testing*
- *The machine is ready to start*

1. Set the following parameters in the Techtable"FlyCut" on page 112 to the specified values:

ADS setting to calibrate the FlyCut module

```
1010 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].DistOn= 0      : REM mm
1020 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].DistOff= 0     : REM mm
1030 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].TimeOn= 0      : REM ms
1040 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].TimeOff= 0     : REM ms
; 0 is current plane, 1 is space
1050 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].DistCalc=0
; -1 the signal level is only switched on by ADSLAB()
1060 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].Duration=-1
1070 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].Type=10
1080 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].Index=0
```

"DistOn", "DistOff", "TimeOn", "TimeOff" to 0.

2. The following NC program cuts two lines in opposite direction into the material. Adapt it to the machine. The acceleration distances have to be long enough to achieve a sufficiently high velocity.

Sample program to calibrate the FlyCut module

```
;Move without HC to defined axis position
;Load material table with fly cut configuration
N10   TECHTABLE("Steel_Cut_1mm")
N30   HEIGHTCTRL(0)          ;switch off HC
```

```
N40  CUT(0)                  ;switch off laser power
N50  G1Z0                    ;move to reference position of z axis
N60  G1 X0Y0                 ;move to start position X/Y axis
N70  M0                      ;wait for PLC


N80  SENSORCALIB("CALIB")    ;calibrate the HC sensor
N100 GAS(1)                  ;switch on gas
N110 HEIGHTCTRL(1)           ;switch on HC
N120 G1 X50 Y0 CUT(1)        ;move, start laser at endpoint
N130 X80 Y0 CUT(0)           ;move, stop laser at endpoint
N140 X100 Y0                 ;move
N150 X100 Y3                 ;move
N160 X80 Y3 CUT(1)           ;move, start laser at endpoint
N170 X50 Y3 CUT(0)           ;move, stop laser at endpoint
N180 X0 Y3                   ;move
N190 CUT(0)                  ;switch off laser power
N200 HEIGHTCTRL(0)           ;switch off HC
N210 GAS(0)                  ;switch off gas
```

3. Check the result. The two lines should start and end at the same level. Correct the setting for the parameters "TimeOn" and "TimeOff" and repeat the test.

- In general, the setting 0 for "TimeOn" and "TimeOff" is correct for a laser.
- For the settings and application notes of ADS, refer to **external link: R911342634: Programming Instructions, chapter 6.4**.

8 LeapFrog

8.1 Introduction

The LeapFrog function is used to quickly move the cutting head to a new position. For this purpose, the cutting head is lifted at the starting point and set down again at the target point. This is done quick and smooth.

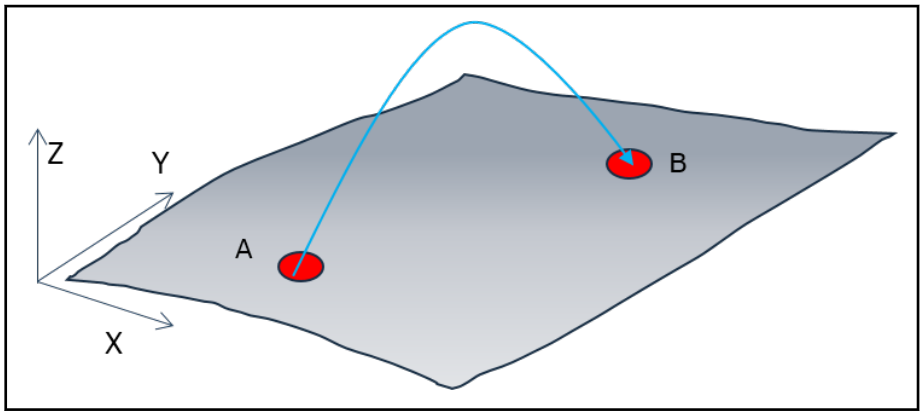


Fig. 8-1: LeapFrog structure

Lifting the cutting head is necessary to avoid a collision with parts that have already been cut out and may protrude from the surface. The cutting head cannot drive directly over such obstacles.

The LeapFrog function is triggered by calling the `LEAPFROG` cycle or optionally with the G-code `G00`. A LeapFrog has already to be programmed in the G-code as the NC does not know where critical traversing motions occur. In general, the CAD system producing the G-code determines the blocked areas and inserts LeapFrog commands into the G-code. The NC automatically performs evasive movements when the `LEAPFROG` command occurs and determines the quickest route over the material.

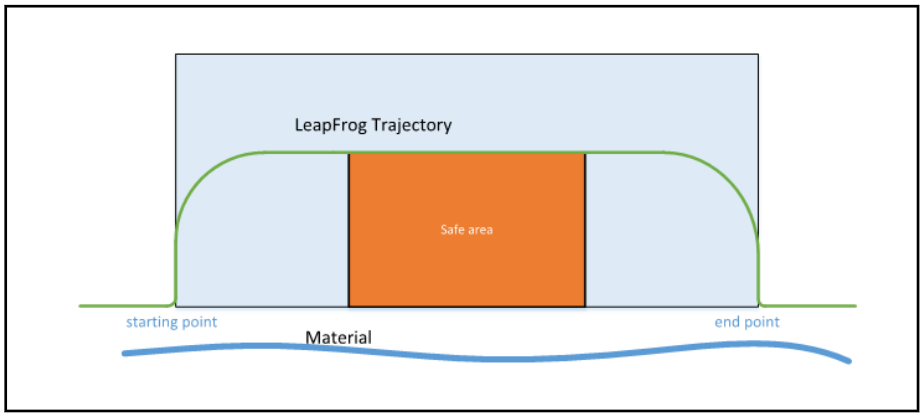


Fig. 8-2: Path structure of a LeapFrog

The LeapFrog adjusts to different scenarios. It detects the material height automatically and moves to the target point as fast as possible.

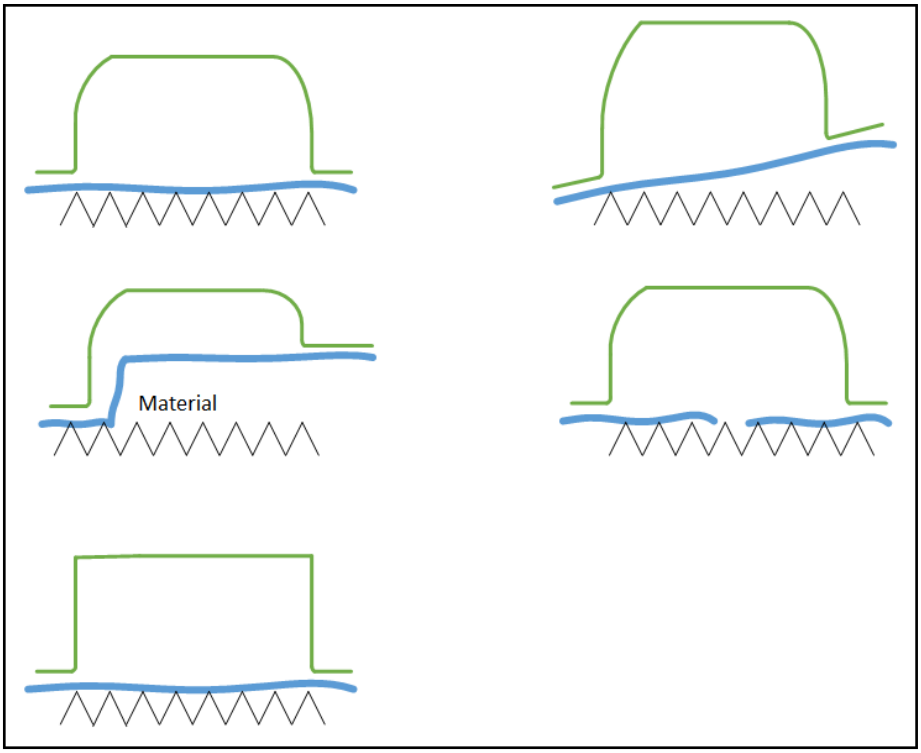


Fig. 8-3: LeapFrog areas of application

Data flow

The path generator implemented in the FA task outputs a new additive position command value for the z-axis upon each cycle. The position of the z-axis from the NC is superimposed with this position command value. The path computation depends on the LeapFrog command and its parameters.

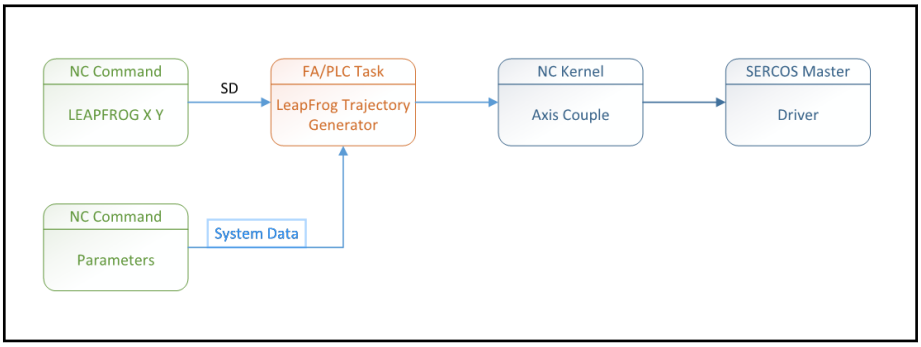


Fig. 8-4: LeapFrog data flow

8.2 Project module LeapFrog

All program parts of the LeapFrog module are located under the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ LeapFrog**. It is divided into two program parts: "LeapFrog_FA" and "LeapFrog_PLC"

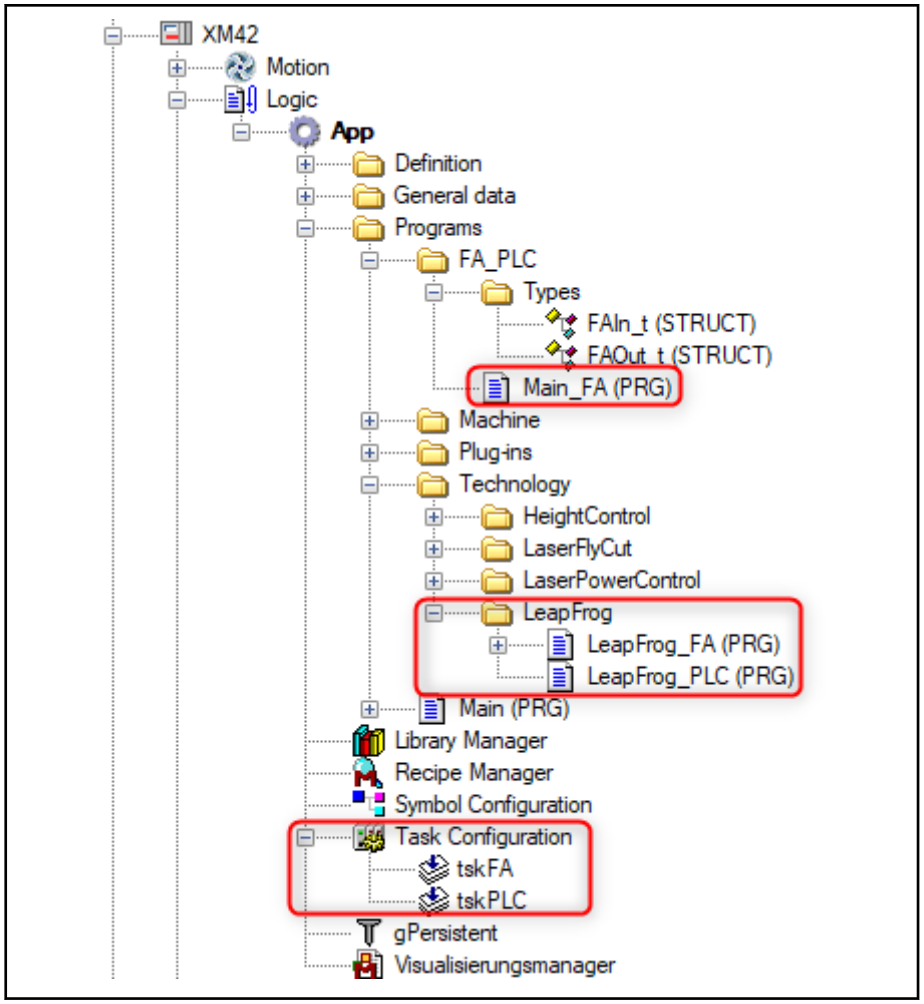


Fig. 8-5: Storage place of the LeapFrog module within the shape cutting project

LeapFrog_FA

The program "LeapFrog_FA" contains the actual control routine. It is started by the program "Main_FA" while executing the Fast Action task ("tskFA"). Within the program, all cyclic data required to execute the LeapFrog functionality is read and transferred to the height control after computing the height. The following variables are transferred cyclically between the NC and the PLC within the FA task:

Direction	Data	Variable	FA type	Data type	Length
Input	Trigger signal	fLeapfrogSta	System date	REAL	4 bytes
Input	Distance to go in the block	lrPathDistToGo	PathDistToGo	LREAL	8 bytes
Input	Path velocity	udiPathVel	PathVelo	UDINT	4 bytes
Output	Correction value	rCorrValue	System date	REAL	4 bytes
Output	LeapFrog status	diLeapFrogSta	System date	DINT	4 bytes
Output	FA path override	rFeedOvrFactor	FAPathOvr	REAL	4 bytes

Tab. 8-1: Cyclic exchange data of LeapFrog

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

LeapFrog_PLC

The "LeapFrog_PLC" program processes the auxiliary variables to read configuration values. It is started by the cyclical task "tskPLC" which runs in a 10 ms cycle.

The calling of the `LEAPFROG` command in the NC-code triggers the LeapFrog. Internally, the LeapFrog uses auxiliary variable `M187` to apply the parameterization to the PLC.

The LeapFrog tests the stability during each PLC cycle. `M188` is reset if LEAPFROG is stable. The NC program waits for the `M188`.

8.3 Parameterization

To use the LeapFrog functionality, a locked and a safe movement area have to be defined. System data variables define these areas. They are specified as follows:

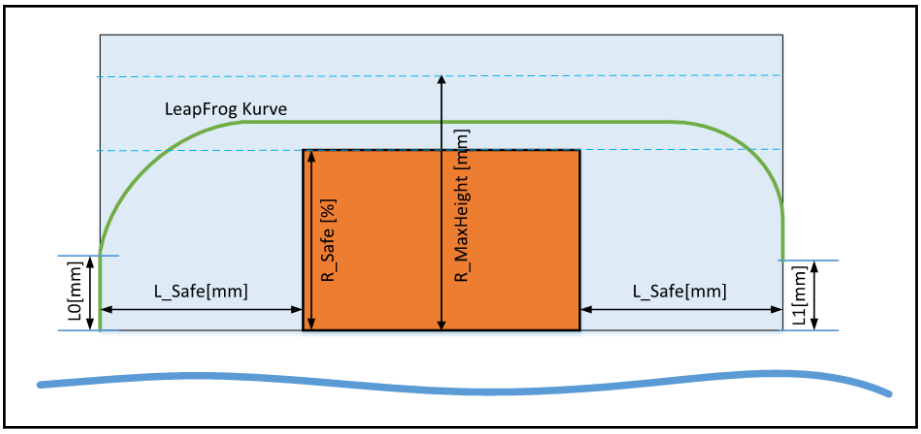


Fig. 8-6: LeapFrog parameter

Parameter	Unit	Description
`SD.SysLeapFrog[*].Mode`		0: Fastest path possible 1: Inverted u-curve
`SD.SysLeapFrog[*].R_MaxHeight`	mm	Maximum height of the safe area
`SD.SysLeapFrog[*].R_Safe`		Safe height as part of `R_MaxHeight` The value has to be between 0.1 and 1.0.
`SD.SysLeapFrog[*].L_Safe`	mm	Maximum distance between the starting point and reaching the height defined in `R_Safe` in mm
`SD.SysLeapFrog[*].L0`	mm	Lifting height at the starting point
`SD.SysLeapFrog[*].L1`	mm	Safe height at the target point
`SD.SysLeapFrog[*].MinDist`	mm	Defines a minimum LeapFrog distance from which the height at the target point is not L1, but L0. A value of 0 switches off the function.
`SD.SysLeapFrog[*].Feedrate`	mm/min	Maximum path velocity during the LeapFrog.
`SD.SysLeapFrog[*].CornerRadius`	mm	Corner radius of the inverted U-curve. It is only used in mode 1.

* Channel number

Tab. 8-2: Setting parameters for the MT_LeapFrogType01

It is usually parameterized in the technology tables (see Technology tables, page 105).

8-6 graphically displays the parameters of the LeapFrog functionality.

8.4 NC commands

8.4.1 LEAPFROG

The NC command LEAPFROG and its abbreviation LF start a LeapFrog motion ending in a stable height control.

Program:

```
TECHTABLE("Steel_Cut_1mm") ; Load Tech Table
LEAPFROG X0 Y0             ; LeapFrog absolute position to X0 Y0
G91
G01 X10 F1000              ; Cutting some contour
G01 Y10
G01 X-10
G01 Y-10
LF X100 Y100               ; LeapFrog relative distance with X100 & Y100
G01 X10                    ; Cutting some contour again
G01 Y10
G01 X-10
G01 Y-10
G90
LF X0 Y0                   ; LeapFrog absolute position to X0 Y0
M30
```

LTS/LCTS are disabled during the LeapFrog motion and restarted after the LeapFrog has been completed. However, LTS or LCTS had to be enabled before.

If the following conditions are met, the LeapFrog motion is replaced by a G01:

- The jump distance is smaller than the minimum distance defined in SD.SysLeapFrog[].MinDist.

G01 uses the feed defined in SD.SysLeapFrog[*].Feedrate. The previous feed is restored after the command.

8.4.2 ResetLeapFrog

The NC command ResetLeapFrog (its abbreviation RLF) deletes an error of the LeapFrog path generator. It is also used for the channel reset (e.g. M30).

Program:

```
; *******************************************************
; * Subprogram after program end (M30) of main program *
; *******************************************************
N1000 LPS ProgEndProg
; Reset Height control
11031 CALL HCSAC_Z_OFF [1,3]
11032 SD.SysHeightCtrl[1].Mode = 0
; Reset LeapFrog
N1041 RLF

N1990 PEND
```

8.4.3 M187 and CheckLeapFrogPara

M187 is used to load the LeapFrog parameters to the PLC.

The NC function CheckLeapFrogPara can be used to check the parameters on NC level. A warning or an error is issued for invalid parameters.

Program:

```
; Modify LeapFrog parameters
1001 SD.SysLeapFrog[1].Feedrate = 100000 ; change LeapFrog feed rate
1002 CheckLeapFrogPara                   ; check LeapFrog parameters
N003 M187                                ; load parameters into PLC task
```

It is recommended to use the command TECHTABLE to load parameter changes.

8.5 Commissioning

There are two options to commission the LeapFrog functionality:

- Installation by restoring the archive (see chapter 4.5 "Commissioning of the PLC" on page 18). All required files except for the macro to replace the G0 and G00 codes are installed.
- Copying and renaming all required files from the /feprom directory.

Files on the control

The LeapFrog functionality is mapped in the following files:

- /feprom/shapecut/subLEAPFROG: Implementation of the LeapFrog functionality as macro LEAPFROG
- /feprom/SDDefShapeCut.xml_: Including the LeapFrog functionality in the system data of the control
- /feprom/schemas/SDLeapFrog.xsd_: Definition of the data types for the LeapFrog functionality in the system data of the control
- /feprom/macro/MacroDef_ShapeCut.xml_: Macro to define the LEAPFROG and LF cycle commands.

After copying the files, restart the NC to activate functionality.

G-code programming

The LeapFrog function can be triggered by calling the G-code G00 if the G-code is defined in the macro file MacroDef_ShapeCut.xml with the following blocks.

Program:

```
<Macro Type="DIN" Overlay="Yes">
    <Name>G00</Name>
    <Replace>SubPConv subLEAPFROG</Replace>
  </Macro>
```

The following example demonstrates the use of the LeapFrog functionality if the G00 macro is not activated:

Program:

```
TECHTABLE("Steel_Cut_1mm") ; Load Tech Table
; Actual G Code
G00 X0 Y0                  ; Move to reference point
LEAPFROG X10 Y100          ; LeapFrog to X=10mm and Y=100mm
LEAPFROG X90 Y300          ; LeapFrog to X=90mm and Y=300mm
```

The following example demonstrates the use of the LeapFrog functionality if the G00 macro is activated:

Program:

```
TECHTABLE("Steel_Cut_1mm") ; Load Tech Table
; Actual G Code
G0 X0 Y0                   ; Move to reference point
G00 X10 Y100               ; LeapFrog to X=10mm and Y=100mm
G00 X90 Y300               ; LeapFrog to X=90mm and Y=300mm
```

Inverted U-curve mode

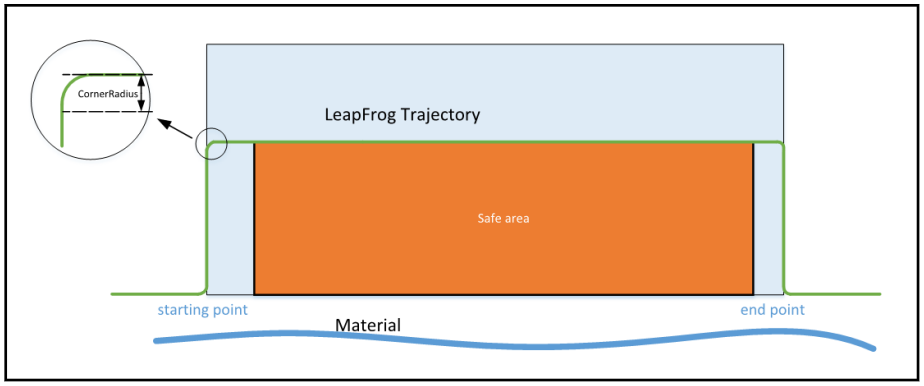


Fig. 8-7: Inverted U-curve mode

The LeapFrog can generate an inverted U-curve for special applications. The functionality is enabled using the following parameters:

- `SD.SysLeapFrog[*].Mode` = 1
- `SD.SysLeapFrog[*].CornerRadius` > 0

The parameter `CornerRadius` determines the radius from the upward to the sideward movement and back to the downward movement. This parameter is only active in mode 1.

Program:

```
; Leapfrog table in "Steel_Cut_1mm"
10210 SD.SysLeapFrog[1].Mode = 1
10220 SD.SysLeapFrog[1].R_MaxHeight = 50 : REM mm
10230 SD.SysLeapFrog[1].R_Safe = 0.6     : REM 0.1 .. 1.0
10240 SD.SysLeapFrog[1].L_Safe = 30      : REM mm
10250 SD.SysLeapFrog[1].MinDist = 1      : REM mm
10260 SD.SysLeapFrog[1].Feedrate = 80000 : REM mm/min
10270 SD.SysLeapFrog[1].CornerRadius = 2 : REM mm
```

Program:

```
; Invert-U Curve LeapFrog example
TECHTABLE("Steel_Cut_1mm") ; Load Tech Table
G0 X0 Y0                   ; Move to reference point
LEAPFROG X10 Y100          ; LeapFrog to X=10mm and Y=100mm
LEAPFROG X90 Y300          ; LeapFrog to X=90mm and Y=300mm
```

Scaling the path

The complete LeapFrog path is reduced if the jump distance is smaller than double the `L_Safe`.

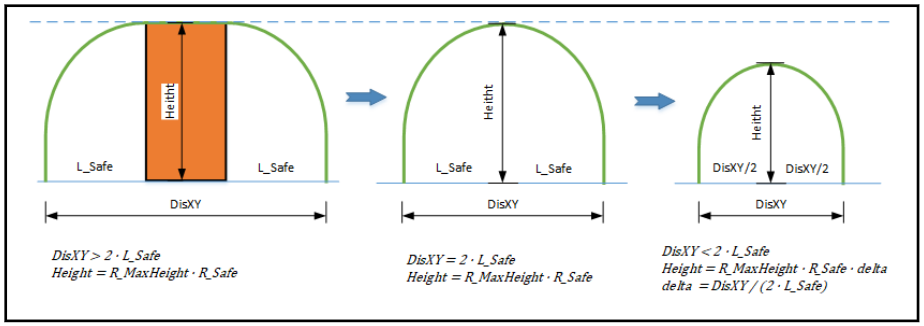


Fig. 8-8: LeapFrog scaling

`L_Safe` should be between 0.1-times and twice of `R_MaxHeight · R_Safe`.

Transition to the height control

A complete LeapFrog motion has seven states. Depending on the parameters set, some are only used for one cycle.

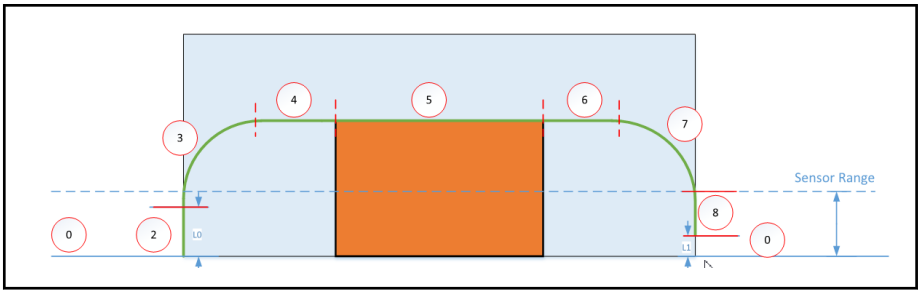


Fig. 8-9: LeapFrog state description

Value	Name	Description
0	Idle	LeapFrog is inactive.
1	Preparing	The LeapFrog prepares the computation.
2	UpwardBeforeL0	In upward motion before L0.
3	UpwardAfterL0	In upward motion after reaching L0, but before reaching the height.
4	TopBeforeSafeZone	Height reached, but still in front of the safe area.
5	TopSafeZone	In the safe area. The z-axis may not be below the height of the safe area.
6	TopAfterSafeZone	Beyond the safe area The z-axis may only be moved downward.
7	Downward	The z-axis moves downward, but has not yet reached the detection range of the sensor.
8	SteelDetected	The sensor detected the material, L1 (setpoint) has not yet been reached.
9	Finished	The LeapFrog was exited.

Tab. 8-3: LeapFrog states

A LeapFrog motion automatically transitions into the height control.

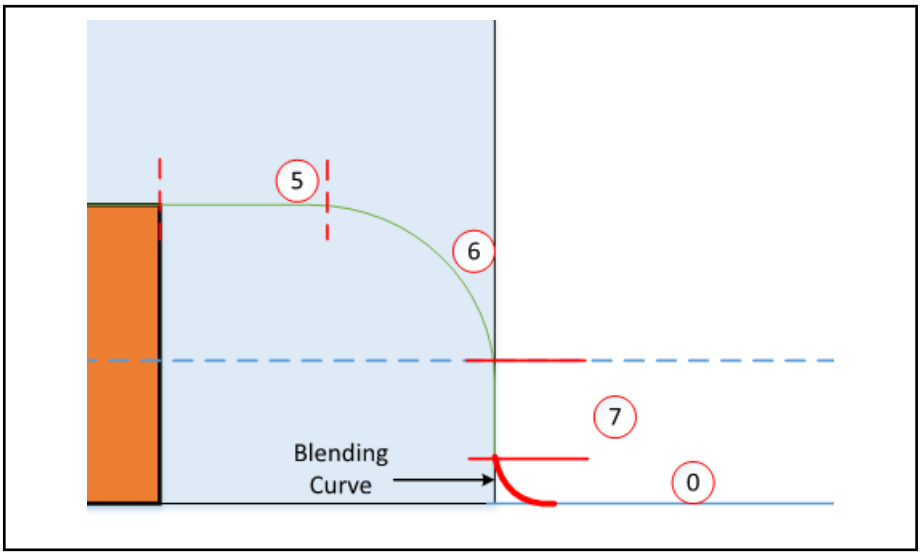


Fig. 8-10: Transition of the LeapFrog to the height control

The user can affect this transition. Change the following program location under **Logic ▸ App ▸ Technology ▸ HeightControl ▸ HeightControl_FA (PRG)**

```
82  // ###############################################################################
83  // Switch from LeapFrog to HeightControl
84  // Downward: HeightControl take corver control of Z axis when Z can move downward
85  LFActive := ((LeapFrogState>EN_LF_STATE_t.Idle) AND (LeapFrogState<EN_LF_STATE_t.Downward));
86
```

Fig. 8-11: Program location to be changed

8.6 Example

The following example demonstrates the use of the LeapFrog functionality if the G00 macro is **not** activated:

Program:

```
;Load Tech Table
TECHTABLE("Steel_Cut_1mm")
; Actual G Code
G00 X0 Y0           ; Move to reference point
LEAPFROG X10 Y100   ; LeapFrog to X=10mm and Y=100mm
LEAPFROG X90 Y300   ; LeapFrog to X=90mm and Y=300mm
```

The following example demonstrates the use of the LeapFrog functionality if the G00 macro is activated:

Program:

```
;Load Tech Table
TECHTABLE("Steel_Cut_1mm")
; Actual G Code
G000 X0 Y0     ; Move to reference point
G0 X10 Y100    ; LeapFrog to X=10mm and Y=100mm
G00 X90 Y300   ; LeapFrog to X=90mm and Y=300mm
```

9 Laser power control

9.1 Introduction

For laser cutting machines, the laser has to be controlled depending on the cutting velocity due to the following reasons:

- Control of the heat input into the material
- Control of the cutting depth
- Clean surfaces when cutting edges and tips

Power curve depending on the velocity

The power input into the material by the laser is usually controlled via a PWM signal (pulse width modulation). The ratio between the deactivated and activated laser part varies depending on the desired power input. Furthermore, the length of the PWM cycle can be changed.

The PWM signal for the laser control is usually generated using an external switching device adapted to the laser. This requires an analog input signal from the control. It can be generated by the analog output of an IndraDrive for example (cf. fig. 9-1 "Laser power control via an analog signal" on page 75).

Other control devices require a PWM signal with a specific frequency. It can be generated with the S20-DIDO-2-OVS high-speed oversampling module for example (cf. fig. 9-2 "Laser power control via a PWM" on page 75).

If the cutting head is not moved, the laser has to be switched off.

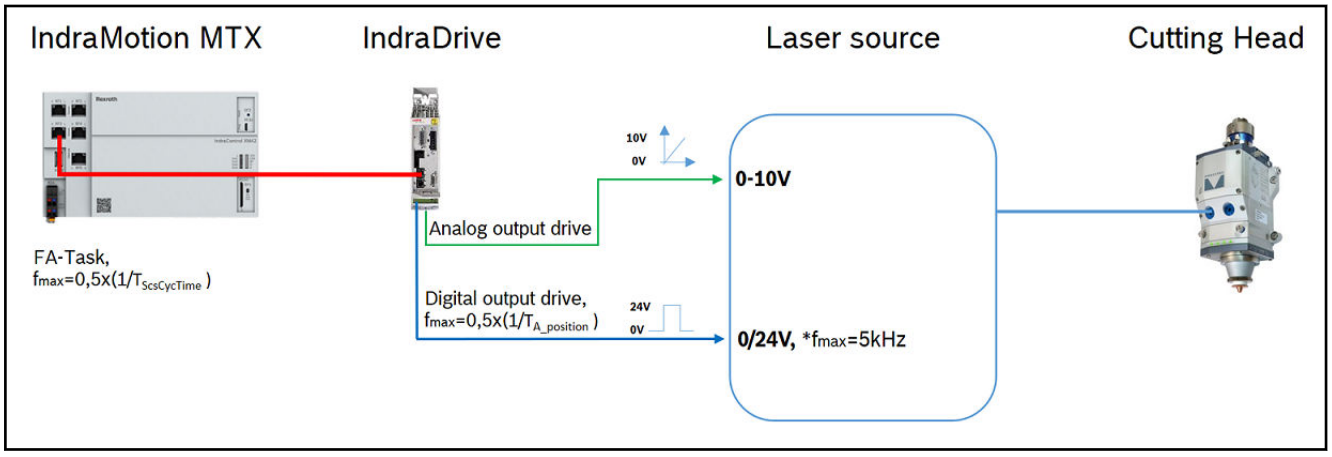


Fig. 9-1: Laser power control via an analog signal

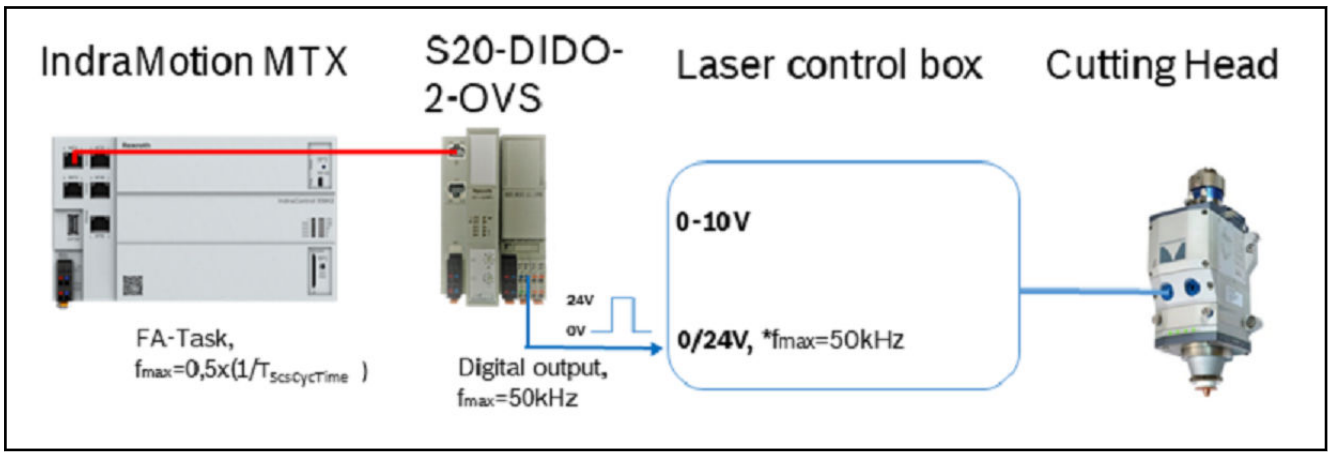
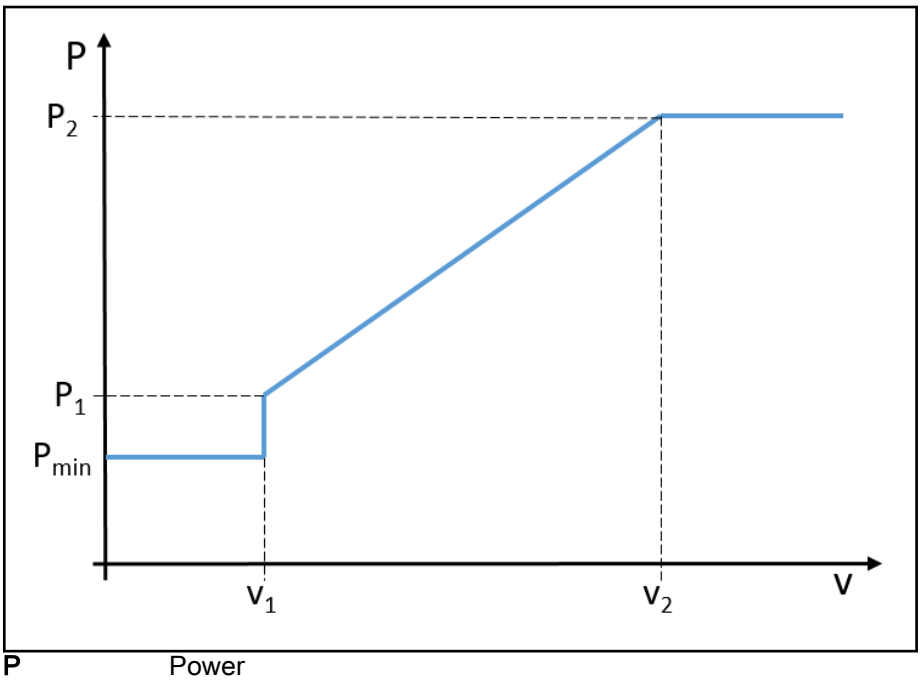


Fig. 9-2: Laser power control via a PWM

Both procedures require a laser power control depending on the velocity. An appropriate control curve is shown in fig. 9-3 "Laser power curve depending on the velocity" on page 76.



P Power
v Velocity

Fig. 9-3: Laser power curve depending on the velocity

Piercing

Another aspect of the laser power control is the piercing, i.e. the start of a new cut beginning with a hole. To avoid fraying of the edges, the pierce point is machines with short laser pulses. The power is increased with every pulse. This process causes small starting holes. Material deformation and material splatters caused by high heat exposure are prevented.

The template provides multiple options to modify this process. The process can be modified to the application using the program.

9.2 Laser power control project module

All program parts of the height control module are located below the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ LaserPowerControl**. It is divided into two program parts: "PowerControl_FA" and "PowerControl_PLC"

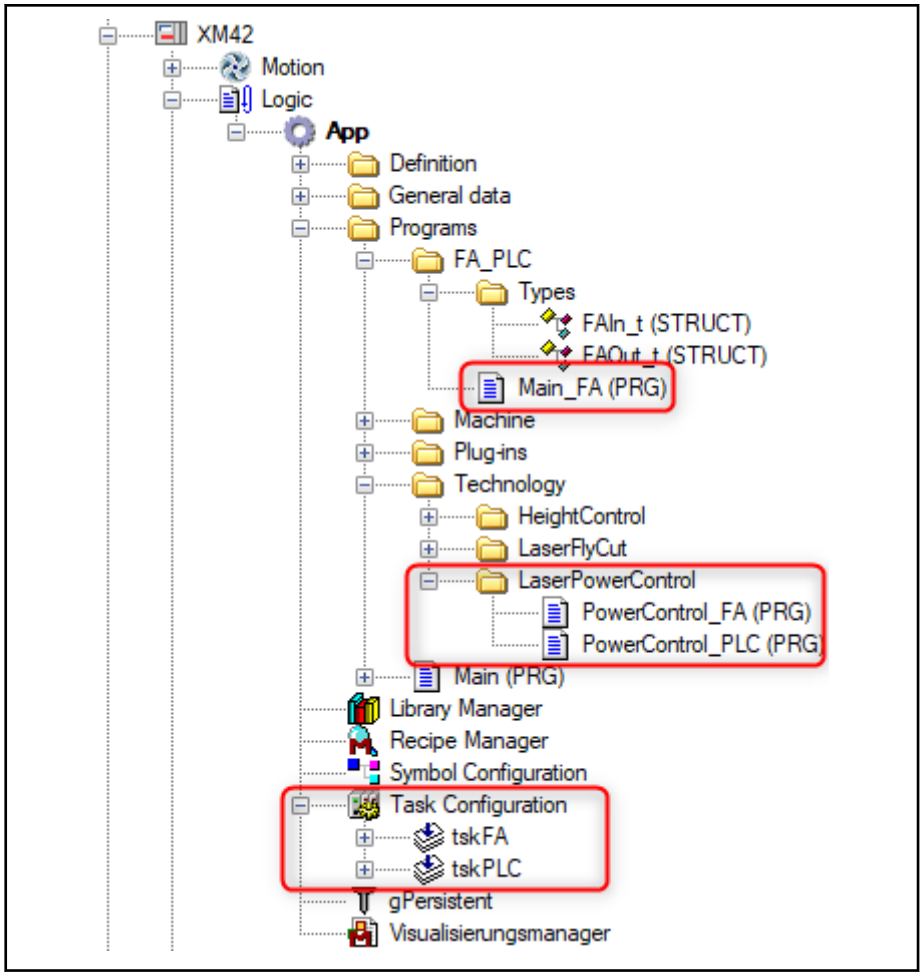


Fig. 9-4: Storage location of the laser power control module in the shape cutting project

PowerControl_FA

The program "PowerControl_FA" contains the actual control routine. The program “Main_FA” starts it while executing the Fast Action task ("tskFA"). In the program, all cyclic data required for the power control is read and written after computing the laser power. The following variables are transferred cyclically between the NC and the PLC within the FA task:

Direction	Data	Variable	FA type	Data type	Length
Input	Path velocity	udiPathVel	System date	DWORD	4 bytes
Input	Reading in the configuration of gas, focus and power status	udiSysPowerMode	System date	DWORD	4 bytes

Tab. 9-1: Cyclic exchange data of the height control

☞ `udiSysPowerMode` links the SD.SysPowerCtrl.SignalStatus system date in NC. Bit 0 - 3 of the signals refers to AnalogMode; bit 4 - 7 refers to the PWMMode and bit 8 - 11 refers to the LaserMode.

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

PowerControl_PLC The "PowerControl_PLC" program processes the auxiliary variables to read the sensor values. It is started by the cyclic task "tskPLC" which runs in a 10 ms cycle. The gas and focus control also requires the program.

The following functions are mapped using auxiliary variables:

- Applying a new configuration to the laser power control: M184

9.3 Parameterization

The laser power control is parameterized using so-called system variables. They are usually configured via the parameterization file `/userfep/techtable/MACHINE` on the file system of the control.

When the laser power is parameterized, it is basically differentiated between an analog output, a PWM on a PWM module and a PWM via the oversampling module. All setting parameters for the analog control begin with `SD.SysPowerCtrl[`Channel number`].Analog`, the setting parameters for the PWM-based control begin with `SD.SysPowerCtrl[`Channel number`].Duty` and the setting parameters for the oversampling module begin with `SD.SysPowerCtrl[`Channel number`].OVS`.

The analog and the PWM part can be active at the same time.

Analog `SD.SysPowerCtrl[*].Analog.Mode`:

This system variable selects the control mode for the analog signal. Possible values:

- 0: No analog signal is used
- 1: Fix power value
  - `SD.SysPowerCtrl[*].Analog.NormalPower` sets the specific power. Value range: 0 ... 10000 mV
- 2: Linear velocity/power control
  - `SD.SysPowerCtrl[*].Analog.SpeedLinear.LowestPower` sets the voltage value for velocities below the minimum velocity. Value range: 0 ... 10000 mV
  - `SD.SysPowerCtrl[*].Analog.SpeedLinear.LowerPoint.Velocity` sets the minimum velocity for the linear conversion in mm/min.
  - `SD.SysPowerCtrl[*].Analog.SpeedLinear.LowerPoint.Power` specifies the voltage at minimum velocity. Value range: 0 ... 10000 mV
  - `SD.SysPowerCtrl[*].Analog.SpeedLinear.UpperPoint.Velocity` sets the maximum velocity for the linear conversion in mm/min.
  - `SD.SysPowerCtrl[*].Analog.SpeedLinear.UpperPoint.Power` specifies the voltage at maximum velocity. Value range: 0 ... 10000 mV

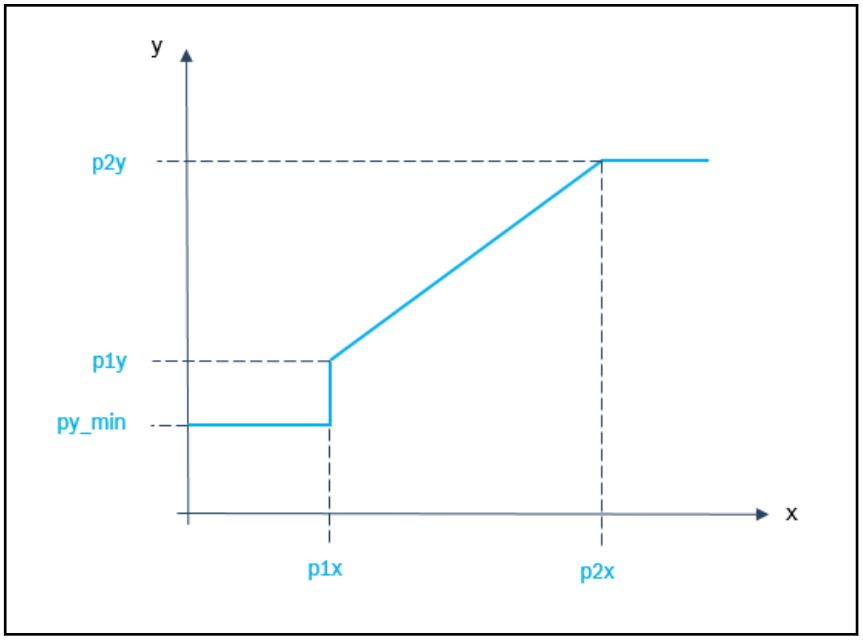


Fig. 9-5: Parameter for the linear velocity/power control

- 3: Velocity/power curve

  The curve is stored in a text file. It is organized in lines. Each line contains a pair of values consisting of velocity and performance separated by spaces.

  – `SD.SysPowerCtrl[*].Analog.SpeedCurve.FilePath` defines the storage location of the curve file as string.

- 4: Time/power curve

  The curve is stored in a text file. It is organized in lines. Each line contains a pair of values consisting of time and performance separated by spaces.

  – `SD.SysPowerCtrl[*].Analog.TimeCurve.FilePath` defines the storage location of the curve file as string.

  – `SD.SysPowerCtrl[*].Analog.TimeCurve.Cycles` specifies the number of repetitions of the time curve.

PWM `SD.SysPowerCtrl[*].PWM.Mode`:

This system variable selects the control mode for the PWM signal. Possible values:

- 0: No PWM signal is used
- 1: Fix duty value

  – `SD.SysPowerCtrl[*].Duty.NormalPower` sets the duty value range: 0 ... 100 %

- 2: Linear velocity/duty control

  – `SD.SysPowerCtrl[*].Duty.SpeedLinear.LowestPower` sets the duty for velocities below the minimum velocity. Value range: 0 ... 100 %

  – `SD.SysPowerCtrl[*].Duty.SpeedLinear.LowerPoint.Velocity` sets the minimum velocity for the linear conversion in mm/min.

- `SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Power` sets the duty at minimum velocity. Value range: 0 ... 100 %
- `SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Velocity` sets the maximum velocity for the linear conversion in mm/min.
- `SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Power` sets the duty at maximum velocity. Value range: 0 ... 100 %

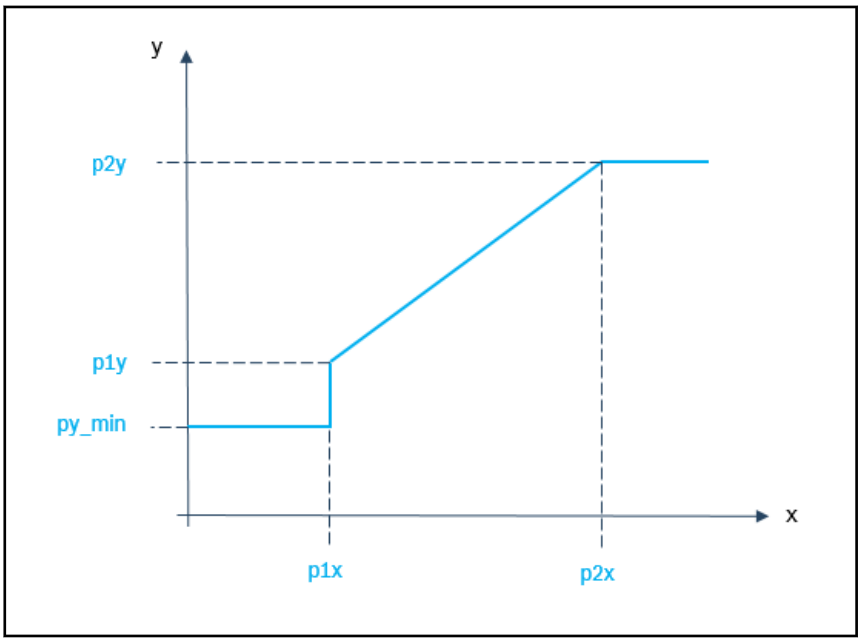


Fig. 9-6: Parameter for the linear velocity/power control

- 3: Velocity/duty curve

  The curve is stored in a text file. It is organized in lines. Each line contains a pair of values consisting of velocity and duty separated by spaces.

  - `SD.SysPowerCtrl[*].Duty.SpeedCurve.FilePath` defines the storage location of the curve file as string.

- 4: Time-linear duty curve
  - `SD.SysPowerCtrl[*].Duty.TimeLinear.LowestPower` sets the duty for the time below the starting time. Value range: 0 ... 100 %
  - `SD.SysPowerCtrl[*].Duty.TimeLinear.Lower-Point.Time` sets the starting time for the linear conversion in ms
  - `SD.SysPowerCtrl[*].Duty.TimeLinear.Lower-Point.Power` sets the duty at the starting time. Value range: 0 ... 100 %
  - `SD.SysPowerCtrl[*].Duty.TimeLinear.Upper-Point.Time` sets the end time for the linear conversion in ms
  - `SD.SysPowerCtrl[*].Duty.TimeLinear.Upper-Point.Power` sets the duty at the end time. Value range: 0 ... 100 %

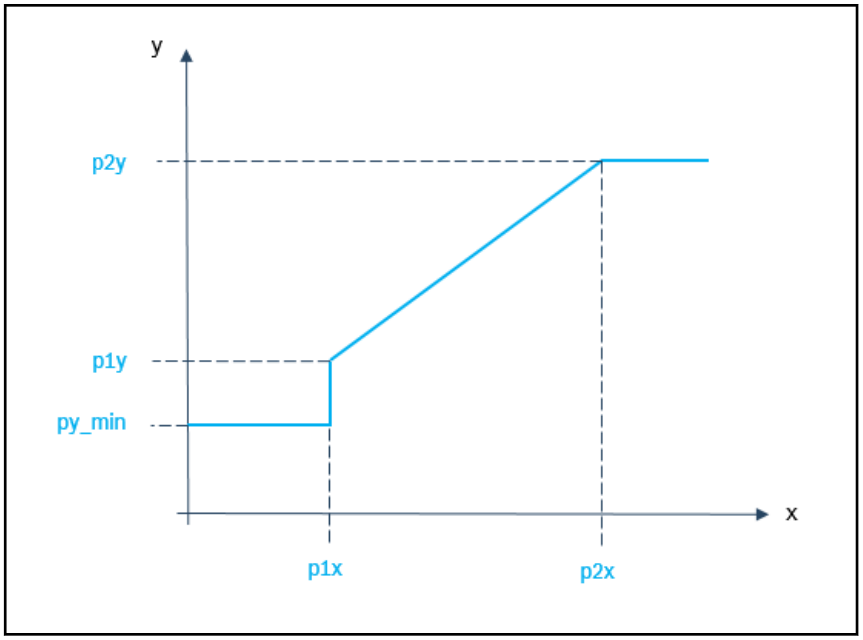


Fig. 9-7: Parameter for the linear time/power control

Piercing

The name of an NC program can be stored for the piercing. It is started when the calling `PIERCING` command and controls the actual procedure. A time can be specified additionally. It can be used directly from the NC program.

`SD.SysPiercing[*].ProgName` name of the NC program called with the `PIERCING` command.

`SD.SysPiercing[*].Time` Time parameter in milliseconds for the piercing.

`*` Channel number

9.4 NC commands for the laser power control

9.4.1 Overview

The definition of the NC commands is stored in the NC directory "usrfep/cycles/" (cf. Commissioning the NC, page 14). After switching the IndraWorks shape cutting project online, the NC directory can be accessed.

In this chapter, the following NC commands of the laser power control are presented:

- Laser switch-on and switch-off command `CUT`, see Laser control: CUT, page 81
- Piercing `PIERCING`, see Piercing: PIERCING, page 82

9.4.2 Laser control: CUT

The laser power control is activated or deactivated using the `CUT` cycle.

The command has a parameter which can either be 1 for switching on or 0 for switching off the laser power control.

Parameterization

The parameterization of the laser power control is usually defined by the machine settings in the technology tables (see chapter 15 "Technology tables" on page 105). The required parameters are described in chapter Parameterization, page 78.

Example:

```
N10 TECHTABLE("steel_cut_1mm"); load Parameters
N20 CUT(1)                    ; switch on laser power control
N90 CUT(0)                    ; switch off laser power control
```

9.4.3 Piercing: PIERCING

The NC command `PIERCING` executes a LeapFrog to the specified position and starts plunging. The NC program stored in `SD.SysPiercing[` channel number `].ProgName` is started.

The command requires the x-position and the y-position as parameters.

During the piercing process, the gas control can be used, using the "GAS" command and the focus control can be used, using the "FOCUS" command, see chapter 10 "Gas control" on page 85 and chapter 11 "Focus control" on page 89.

Example

```
; Main program
N10 X0
N20 TECHTABLE(,"Piercing001",) ; Load cutting parameter table, 70ms
N30 PIERCING X30 Y30  ; Leapfrog to target pos then call piercing program
N40 GAS(1)            ; switch on gas
N50 HEIGHTCTRL(1)     ; Height control enable
N60 CUT(1)            ; Laser on directly, V=0, FlyCut off
N70 X40 Y40 G1 F1000
N80 X50 Y50
N90 X200 Y200
N100 CUT(0)                ; Laser off at X=200, V=0, FlyCut off.
N110 LEAPFROG Y40
N120 G1 F2000 Y140 CUT(1)  ; Laser on at Y=140, V=2000, FlyCut on
N130 Y180 CUT(0)
N140 Y200 CUT(1)
N150 Y400 CUT(0)           ; Laser off at Y=400, V=2000, FlyCut on
N160 Y500
N170 HEIGHTCTRL(0)
N180 GAS(0)                ; switch off gas
```

Fig. 9-8: NC sample program

Piercing001

```
; Technology table
100 SD.SysPiercing.ProgName = "/usr/bosch/Steel_PiercingProg01"
110 SD.SysPiercing.Time = 70
;Height control
120 SD.SysHeightCtrl[CHAN%].SetPoint = 5
;Power control
130 SD.SysPowerCtrl.Analog.Mode =1
...
;Focus control parameters
200 SD.SysFocusCtrl[CHAN%].FocalPos = 4

;Gas control parameters
300 SD.SysGasCtrl[CHAN%].GasType = 1
310 SD.SysGasCtrl[CHAN%].GasPressure = 20
320 SD.SysGasCtrl[CHAN%].GasTime = 2000
...
```

Fig. 9-9: Example of a technology table with Piercing configuration.

/usr/bosch/Steel_PiercingProg01

```
GAS(1)                          ; Open gas
HEIGHTCTRL(1)                   ; Height control enable, Z axis move to P1
CUT(1)
DELAY([SD.SysPiercing.Time])    ; Piercing
CUT(0)
FOCUS                           ; Change focial position
CUT(1)
DELAY([SD.SysPiercing.Time])    ; Piercing
```

```
CUT(0)
GAS(0)
```

Fig. 9-10: Sample of a “Piercing” file

10 Gas control

10.1 Introduction

In case of laser beam cutting, the gas pressure as well as the gas type has to be controlled using the different machining processes.

The command gas pressure control is realized using an analog signal. There are three types of gas control, telling by the sensor feedback:

- Gas control without pressure sensor: The waiting time until the target pressure is reached can be parameterized
- Gas control using the digital pressure sensor: Sensor is generated "DONE" signal after reaching the target pressure
- Gas control with analog pressure sensor: The actual gas pressure is measured continuously and is comparted to the command gas pressure. If both pressures are identical, the "DONE" signal is generated

Supplying different gas types supports the cutting and ensures clean cutting edges. The gas is often used to blow the melted material out of the cutting channel. Depending on the material, the process and the currently executed step, different gas types and gas pressures can be selected

The gas types oxygen (O_2), nitrogen (N_2), helium (He) and air are the most commonly used gas types. These are selected via valves. An optional pressure sensor and a proportional valve control the gas pressure. fig. 10-1 "Gas control design" on page 85 shows the general design of a gas control.

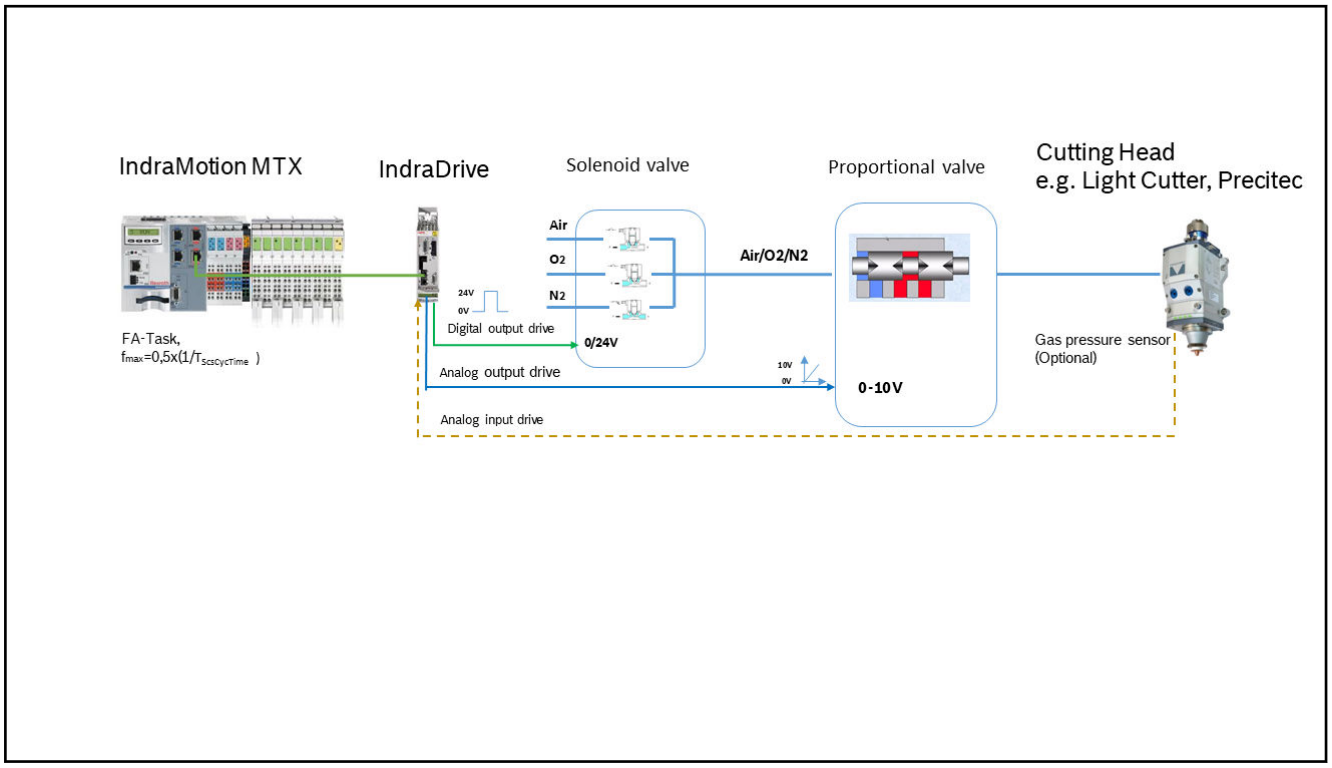


Fig. 10-1: Gas control design

10.2 Project module gas control

All program parts of the gas control module are located in the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ GasControl**. It contains the program: "GasControl_FA".

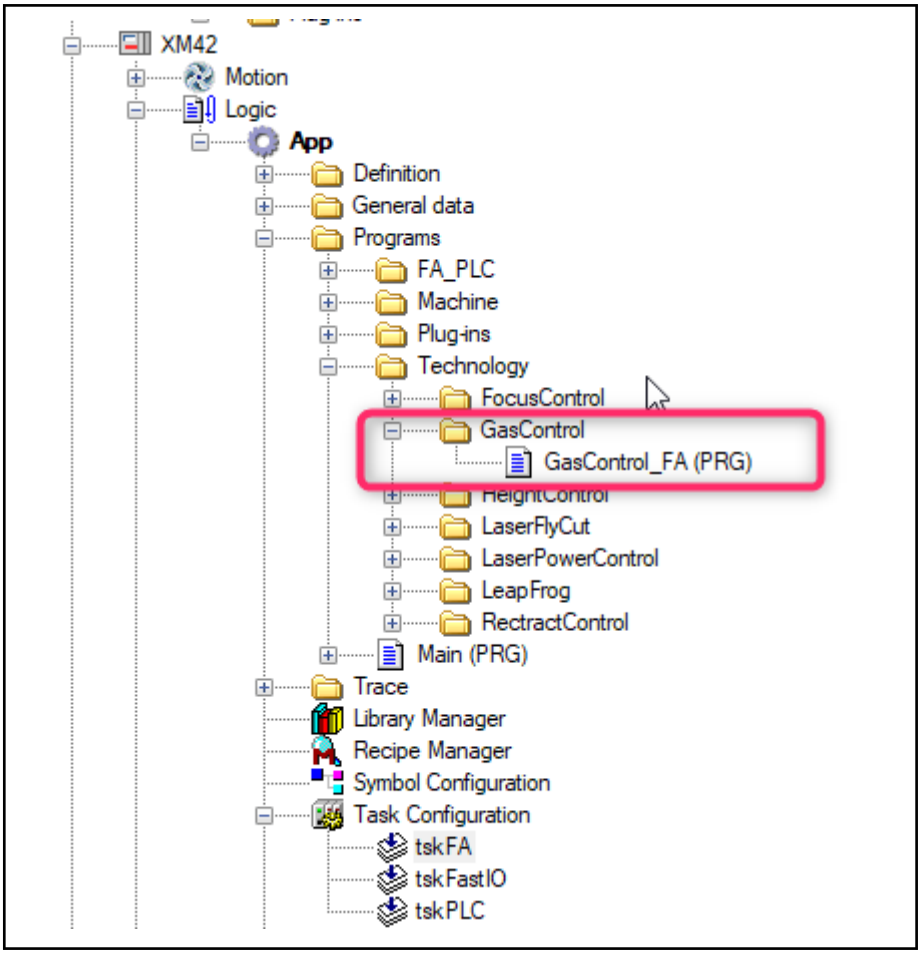


Fig. 10-2: Storage place of the gas control module within the laser beam cutting project

Gas control in FA task: GasControl_FA

The program "GasControl_FA" contains the actual control routine. It is started by the program "Main_FA" while executing the Fast Action task ("tskFA"). Within the program, all cyclic data required for gas control is processed. The following variable contains the status information of gas, focus and power control. It is transmitted to the PLC cyclically within the FA task between the system date `SD.SysPowerCtrl.SignalStatus` in NC and the FA input `udiSysPowerMode`. The bits 12-15 of the signal refer to the gas mode.

Direction	Data	Variable	FA type	Data type	Length
Input	Reading in the configuration of gas, focus and power status Bit 12 - 15: Gas mode	udiSysPowerMode	System date	DWORD	4 bytes

Tab. 10-1: Cyclic exchange data of the gas control

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

Gas control in PLC task: PowerControl_PLC

The auxiliary variables to read the gas control parameters are located in the program "PowerControl_PLC", also required by the laser beam control. It is called by the cyclic task "tskPLC" with a 10 ms cycle.

The following functions are mapped using auxiliary variables:

- Applying a new configuration to the gas, the focus and the laser power control: M184
- Waiting for reaching of gas pressure: M185

The gas control can be implemented independently of the laser power control.

10.3 Parameterization

The gas control is parameterized using so-called system variables. They are usually configured via the parameterization file `/userfep/techtable/MACHINE` on the file system of the control.

All setting parameters of the gas control start with `SD.SysGasCtrl[*]`. Set the gas control parameter in the `SysGasCtrl` section of the technology table.

Parameter	Unit	Description
`SD.SysGasCtrl[*].SensorType`	/	Sensor type: 0: without sensor, 1: analog sensor for gas pressure, 2: digital sensor with fixed command pressure
`SD.SysGasCtrl[*].GasType`	/	Gas type: `O2`, `N2`, `Air`, `etc.`
`SD.SysGasCtrl[*].GasPressure`	bar	Command gas pressure
`SD.SysGasCtrl[*].GasTime`	ms	Waiting time until gas pressure is reached
`SD.SysGasCtrl[*].WaitTimeout`	ms	Waiting time to reach the gas pressure before an error is output
`SD.SysGasCtrl[*].UnstableTimeout`	ms	Maximum permitted time for instable gas pressure during machining
`SD.SysGasCtrl[*].StableWindow`	bar	Tolerance window of the analog feedback of the gas pressure in bar

* Channel number

Tab. 10-2: Setting parameters of the gas control

10.4 Gas control: GAS

The gas required for the cutting process is activated or deactivated with the `GAS` cycle.

The command has a parameter which can either be 1 for switching on or 0 for switching off the gas supply.

Example

```
N20 GAS(1)                    ; switch on gas
N90 GAS(0)                    ; switch off gas
```

11 Focus control

11.1 Introduction

The focus position of the laser beam has to be controlled in laser beam cutting machines.

The command focus control is realized using an analog signal. There are two types of focus control:

- Focus control without feedback sensor: The waiting time to reach the target position is parameterized
- Focus control with handshake signal: The sensor generates a "Pos-Reached" signal after reaching the target position

11.2 Project module focus control

All program parts of the focus control module are located in the folder **Logic ▸ App ▸ Programs ▸ Technology ▸ FocusControl**. It contains the program: "FocusControl_FA".

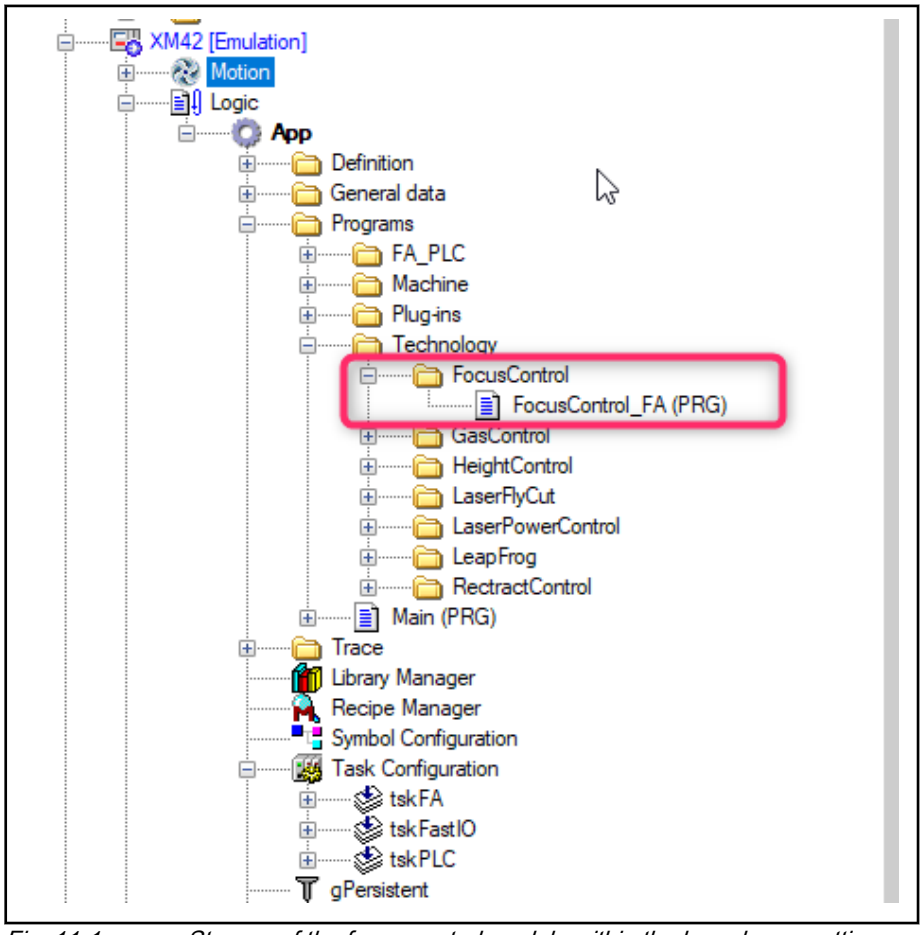


Fig. 11-1: Storage of the focus control module within the laser beam cutting project

Focus control in FA task: FocusControl_FA

The "FocusControl_FA" program contains the actual control routine. It is started by the program "Main_FA" while executing the Fast Action task ("tskFA"). Within the program, all cyclic data required for focus control is processed. The following variable contains the status information of gas, focus and power control. It is transmitted to the PLC cyclically within the FA task between

the system date `SD.SysPowerCtrl.SignalStatus` in NC and the FA input `udiSysPowerMode`.

Direction	Data	Variable	FA type	Data type	Length
Input	Gas, focus and power status Bit 16 - 19: FOCUS mode	udiSysPowerMode	System date	DWORD	4 bytes

Tab. 11-1: Cyclic exchange data of the focus control

For more information on the cyclic data exchange, refer to chapter 5 "Fast data exchange" on page 25.

Focus control in PLC task: PowerControl_PLC

The auxiliary variables to read the focus control parameters are located in the program "PowerControl_PLC", also required by the laser beam control. It is started by the cyclic task "tskPLC" which runs in a 10 ms cycle.

The following functions are mapped using auxiliary variables:

- Applying a new configuration to the gas, focus and laser power control: M184
- Focus position is reached: M186

11.3 Parameterization

The focus control is parameterized using so-called system variables. They are usually configured via the parameterization file `/userfep/techtable/MACHINE` on the file system of the control.

All setting parameters for the focus control start with `SD.SysFocusCtrl[*]`. Set the focus control parameter in the `SysFocusCtrl` section of the technology table.

Parameter	Unit	Description
`SD.SysFocusCtrl[*].SensorType`	/	Sensor type: 0: without sensor, 1: digital sensor with handshake signal
`SD.SysFocusCtrl[*].FocalPos`	mm	Command focus position
`SD.SysFocusCtrl[*].WaitTime`	ms	Waiting time until the focus position is reached
`SD.SysFocusCtrl[*].InPosTimeout`	ms	Waiting time until the focus position is reached before an error is output

* Channel number

Tab. 11-2: Setting parameters of the focus control

11.4 Focus control: FOCUS

The focus control is activated using the `FOCUS` cycle.

The command does not have any parameter.

Example

```
N20 FOCUS                    ;Positionierung durch Fokussteuerung
```

12 Axis group (Gantry)

12.1 Overview on Gantry

An axis group of two or more axes that follow one common preset is referred to as Gantry. In general, the axis group is not mechanically connected. Each axis has its own controller and motor. When controlling the axis group, the positions of the axis as well as the axial limits (position, velocity, acceleration and torque) are compared against each other and controlled according to the weakest axis.

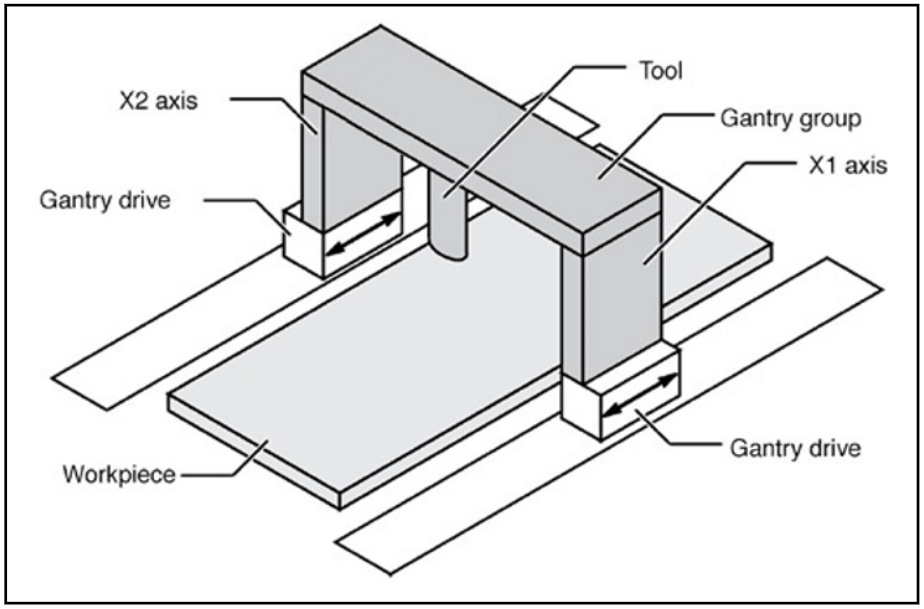


Fig. 12-1: Gantry axis group

A Gantry axis group has to meet the following conditions:

- The measuring system has to be identical on all axes.
- All axes have identical axis parameters

The Gantry axis group works for the following weighting methods:

- Linear absolute
- Linear modulo
- Rotary endless
- Rotary modulo

12.2 Control behavior

The control behavior of a Gantry axis group is specified in the following. The reasons for position deviations of individual axes as well as the control behavior resulting thereof are discussed.

For a detailed description of the Gantry function, see the documentation "Rexroth IndraMotion MTX Functional Description 15VRS - Extension" (see chapter 1.4 "Required and supplementing documentation MTX" on page 2)

During the axes movement, the NC automatically compensates the deviations of the axis position to the end of the NC block.

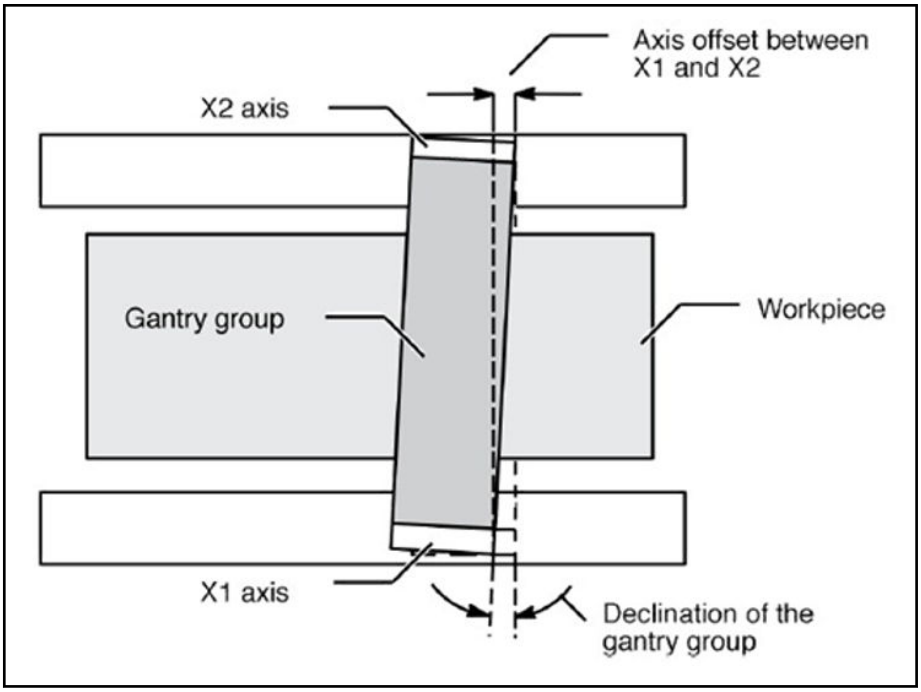


Fig. 12-2: Deviation of the axis position from the nominal value

Requirements

The following requirements have to be fulfilled for the operation of a Gantry axis group:

- All axes belonging to the group were commissioned correctly.
- All axes belonging to the group are referenced.
- No offset is set for the target position.

Reasons for deviations

A deviation of the axis position during processing can have the following reasons:

- a unilateral blockage
- an uneven load of the individual axes

During axial standstill or braking, a deviation of the axis position may occur for the following reasons:

- Different braking response during delay
- A value difference could not be compensated during the motion path as the drives were switched off.
- The axes have shifted each other when switched off.
- Open loop axes drifted away.

Starting behavior

When the axes are switched on, the NC specifies identical target values for all axes of the Gantry axis group. If the resulting position leap of individual axes is too big, an error (F2037: excessive command value specification or F2028: excessive control deviation) is triggered in the drive. If the deviation remains, it is applied and considered as static offset during later movements by the NC. This occurs under the following conditions:

- For drives with absolute encoders at control startup if the axis is referenced
- For drives with incremental encoders when referencing
- When driving to a reference point with mechanical switch

Compensation motion A command value offset can be compensated during axis interpolation. An asynchronous axis movement of one or more axes of the axis group compensates an actual value offset.

12.3 Commissioning

This chapter describes the commissioning of the Gantry axis group. The conditions, parameterization and the diagnostic options are described.

Before commissioning of the Gantry axis group, the drives in the Sercos bus have to be available. The axis names and the Sercos addresses have to be entered beforehand.

1. Activating the settings for the axis coupling

This step can be skipped after loading the TAR files (see Commissioning the NC, page 14).

Open the NC configuration by double-clicking on **XM42 ▸ Motion ▸ NC configuration**. Select the first entry "Selection of Optional Parameters (SUP)" on the right-hand side of the IndraWorks window. Activate the entry "Electronic Couplings (COUP)" in the selection list.

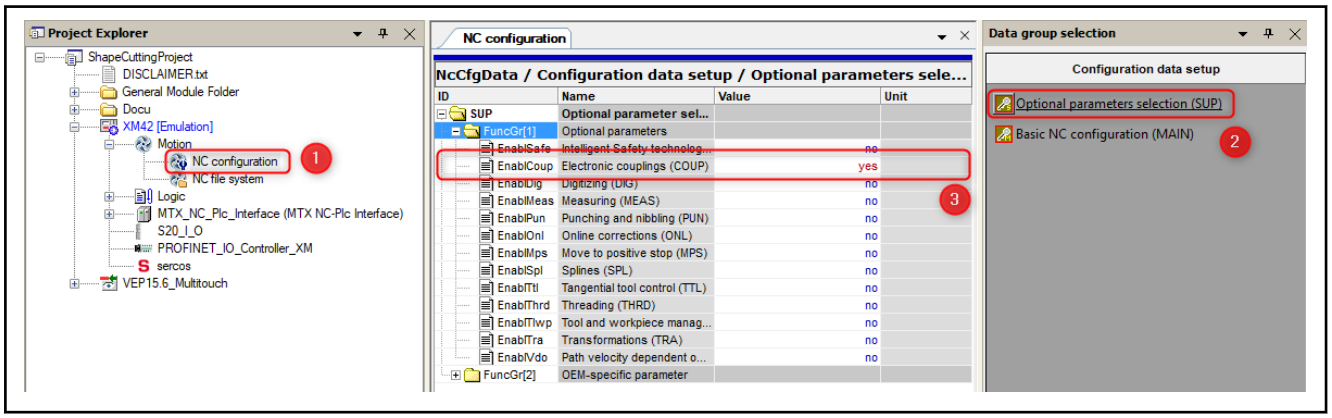


Fig. 12-3: Activating the settings for the axis coupling

2. Parameterization of the Gantry axis group

   Open the NC configuration by double-clicking on **XM42 ▸ Motion ▸ NC configuration**. Select the first entry "Electronic Couplings (COUP)" on the right-hand side of the IndraWorks window (①).

   Adjust the settings for the axes (②) to be operated in the Gantry axis group according to fig. 12-4 "Parameterizing Gantry" on page 94. The upper axis is parameterized as the group master.

   Select the same "axle group index" (③) for all axes in the group.

   Change the "Maximum lag distance" according to the machine (④). If the lag distance is larger than the specified value, the NC generates an error message.

   Parameterize the entry "Upper limit for Gantry command value offset" (⑥). This value determines the maximum deviation of the command values between the master and the slave axis. This value can only be set if the entry "Balanced Gantry command value offset" is displayed on the selection "Max. value from "Maximum lag distance"" (⑤).

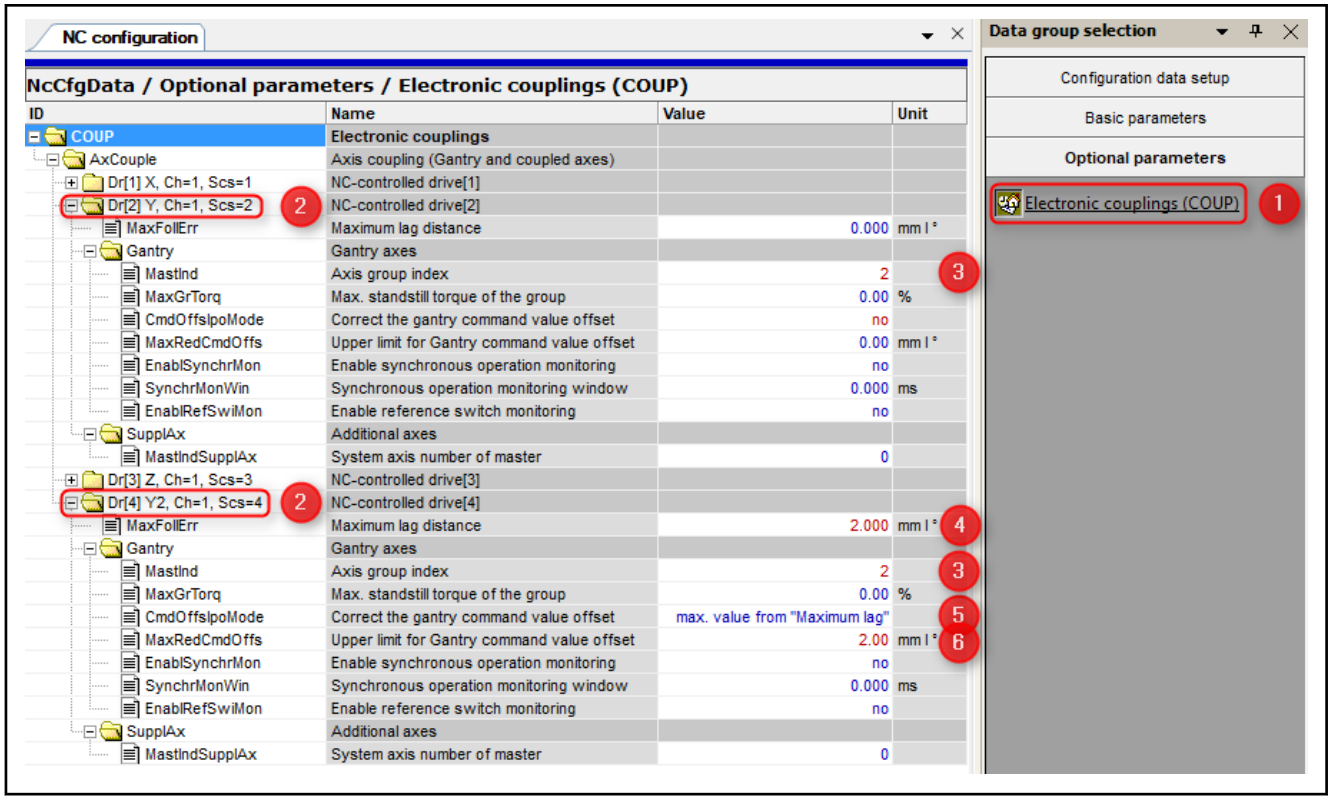


Fig. 12-4: Parameterizing Gantry

3. Complete parameterization

   Close the window "NC configuration" to complete the parameterization. IndraWorks asks whether the NC should be restarted. This has to be done in order to activate the new parameterization.

12.4 PLC plug-in

In addition to the NC-controlled axis Gantry function, there is another PLC-controlled auxiliary function which compensates the position offset of the axes at machine start. This function is executed as a plug-in in the shape cutting project.

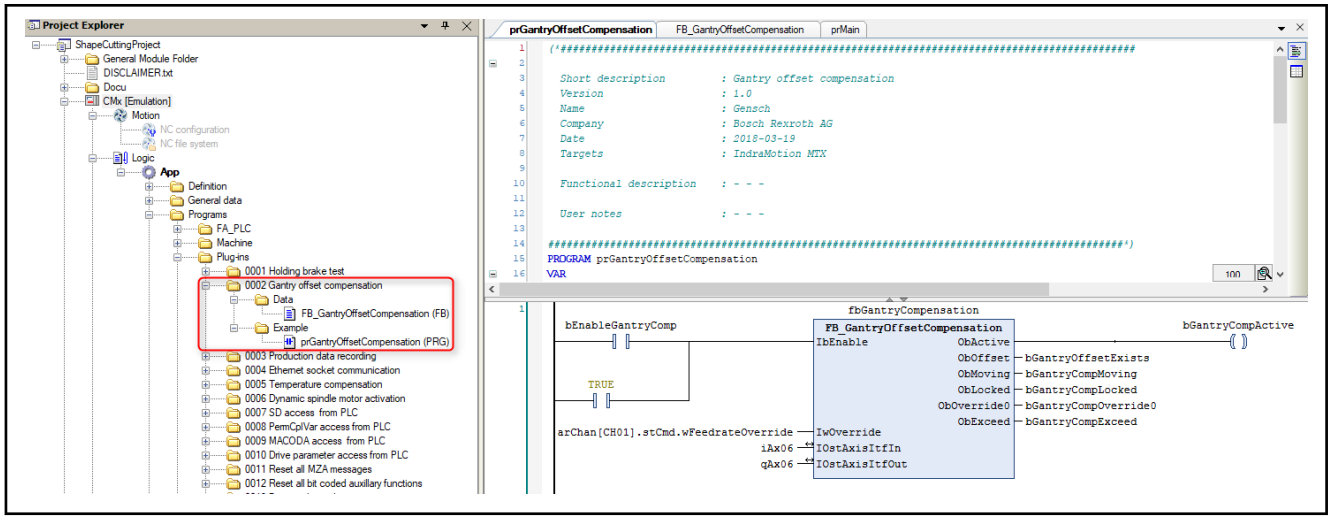


Fig. 12-5: Plug-in Gantry offset compensation in the shape cutting project

The plug-in can be integrated by calling in the PLC main program "Main".

Requirements

In order to use the plug-in, the following requirements have to be fulfilled:

- In the case of absolute encoders, a master and at least one slave axis exist
- In the case of an incremental measuring system with a relevant position measurement, there has to be one reference point for all axes of the Gantry axis group.
- All axes have to be enabled and referenced
- The difference between the commanded and the actual position is smaller than the maximum permissible deviation set in the parameterization (cf. chapter 12.3 "Commissioning" on page 93)
- The upper limit for the deviation of the commanded positions is not exceeded or the monitoring is switched off (cf. chapter 12.3 "Commissioning" on page 93)

13 Cutting width compensation (kerf)

13.1 Introduction

Cutting of material inevitable leads to a loss of material. The cut has a specific width which has to be considered when cutting. Depending on the cutting technology used and the thickness of the material, different cutting widths are possible:

Technology	Diameter of the jet	Material thickness	Cutting width
Laser cutting	0.01 – 0.02 mm	≤ 1 mm	0.05 – 0.2 mm
	0.2 mm	10 mm	0.1 – 1 mm
	0.5 mm	50 mm	2.5 mm
Plasma cutting	2 mm	≤ 150 mm	3 mm
Water jet cutting	0.8 – 1 mm	≤ 150 mm	0.1 – 1 mm

Tab. 13-1: Cutting widths of different cutting technologies

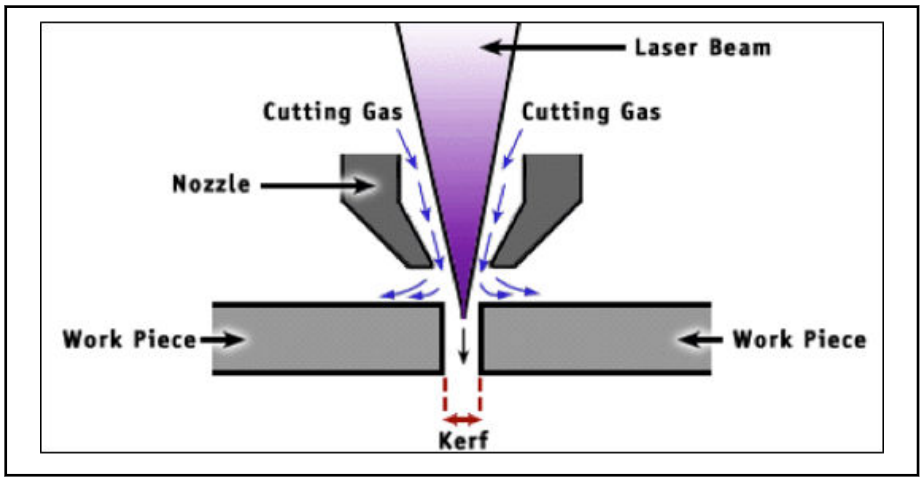


Fig. 13-1: Cutting width of a laser beam

The cutting width is transferred via the tool radius in MTX. The tool radius has to be set to half the cutting width. Up to 99 tools can be defined. The cutting width can be compensated depending on the material thickness and set performance.

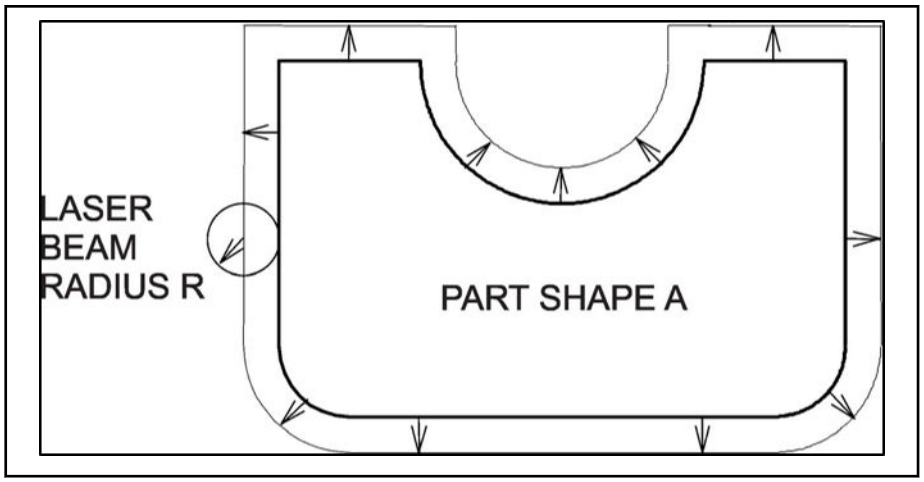


Fig. 13-2: Cutting width around the workpiece

13.2 D-corrections

The tool correction compensates the width of the cutting jet. For this purpose, the width of the resulting cut divided by two has to be indicated as the tool radius in the D-correction.

The tool database can be accessed via the HMI in the "Machine" screen with the "D-corrections" F key. The radius for each tool in the "R" field can be set there. Other fields are not relevant for the shape cutting technology.

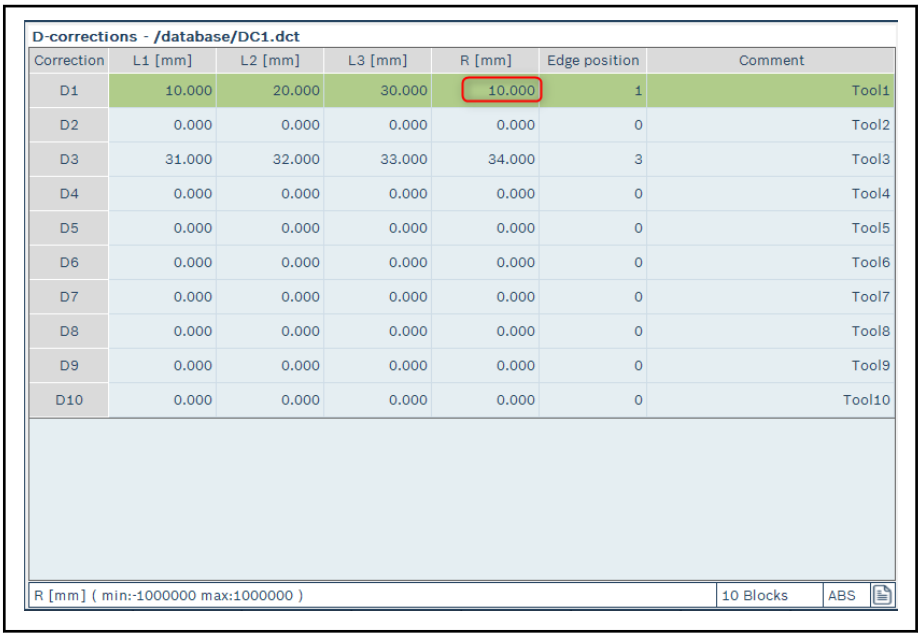
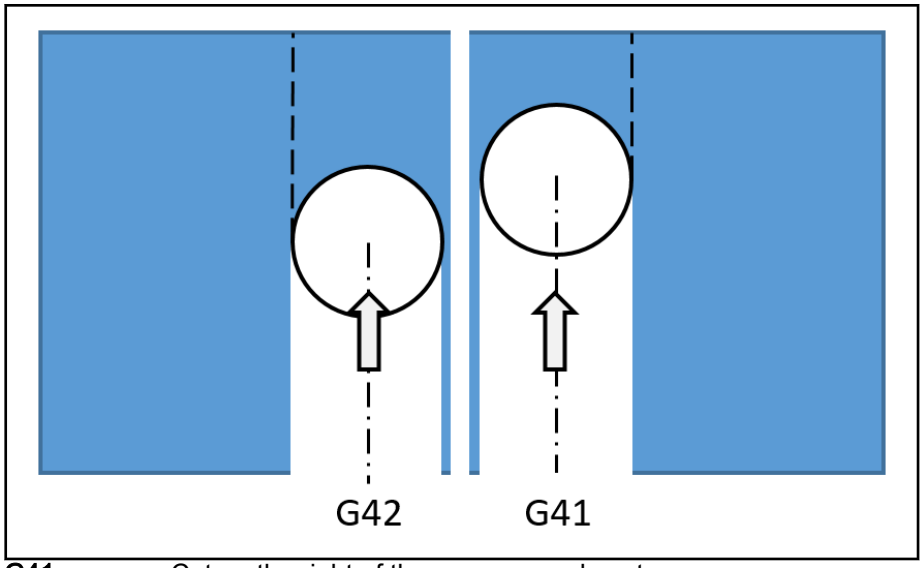


Fig. 13-3: D-corrections screen

Whether and how the D-correction is applied is determined via the commands `G40`, `G41` and `G42` in the NC program. The following determination applies:

G-code	Behavior
G40	Cut on the contour, a compensation movement reduces a correction compensation that may have existed before
G40(NOM)	Cut on the contour, a correction compensation that may have existed before is not reduced
G41	Cut on the left of the contour
G42	Cut on the right of the contour

Tab. 13-2: Behavior of the D-correction



G41 Cut on the right of the programmed contour
G42 Cut on the left of the programmed contour

Fig. 13-4: Positioning of the cutting jet according to G41 and G42

The D-correction to be used has to be indicated with the command `G41`/`G42`. This is done via the G-code `D`. `D0` means that no D-correction is applied. `D1` to `D99` selects one of the existing entries from the D-correction table.

In the following example, the programming is depicted with and without cutting width compensation.

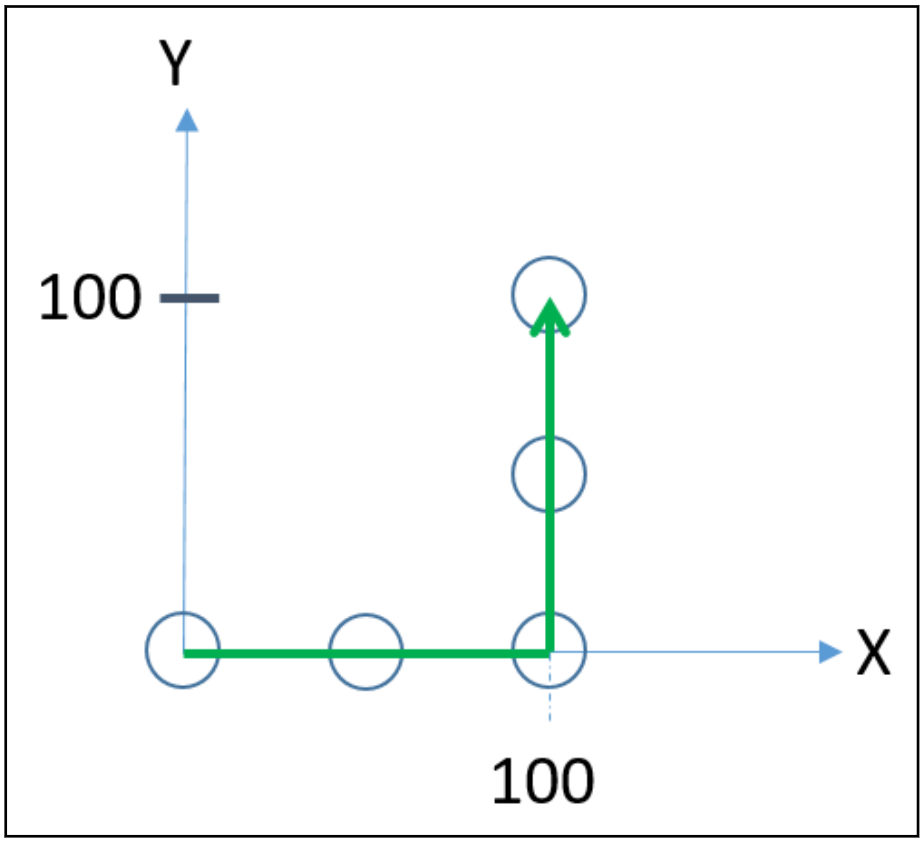


Fig. 13-5: Without cutting width compensation

Program:

```
G90 G1 X0 Y0 Z0 F10000
G1 D0 G40 X100 F1000
Y100
M30
```

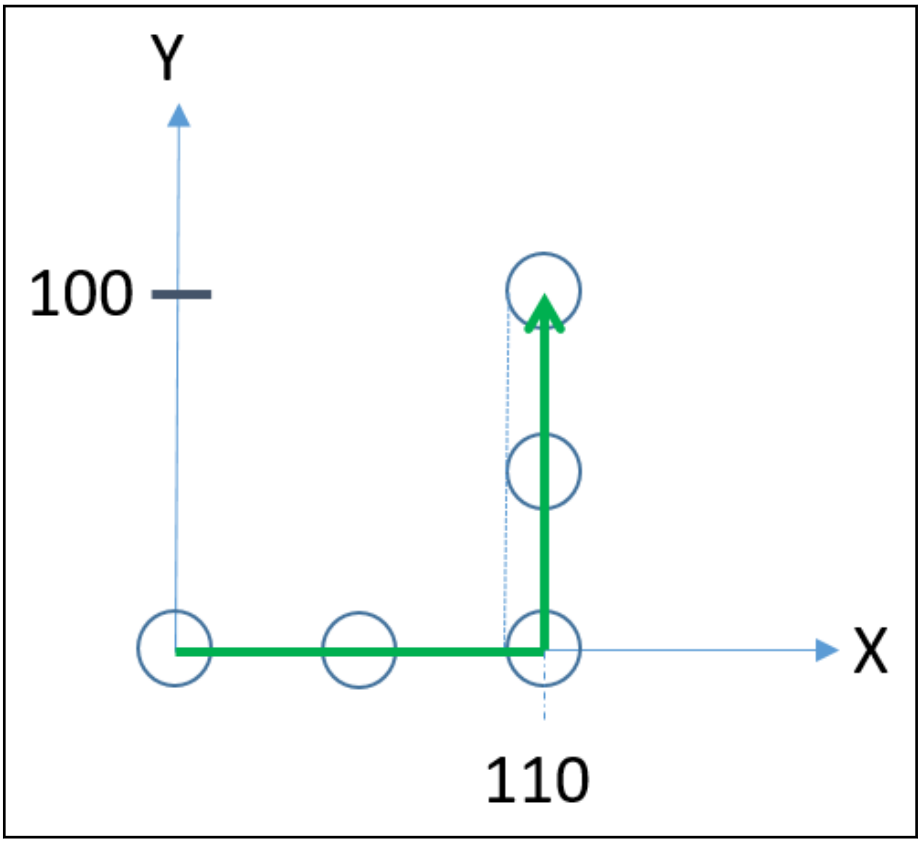


Fig. 13-6: With cutting width compensation

Program:

```
G90 G1 X0 Y0 Z0 F10000
G1 X100 F1000
G1 X100 D1 G42
Y100
D0 G40(NOM)
M30
```

14 Re-enter and Retrace functionality

14.1 Introduction

In the processing of workpieces, interrupting the process and continuing at a defined location is often necessary. There are many reasons for this. It may be necessary to set the machine parameters or the processing operation had to be interrupted due to an error. As the processing of workpieces often takes longer to complete, it makes sense to continue processing as close to the point of abortion as possible.

For this purpose, Rexroth MTX offers the functions Restart and Retrace. Restart enables re-entering at a defined location with normal program processing afterwards whereas the Retrace function allows for forward and backward movement on the contour.

14.2 Re-entry functions of the NC program

14.2.1 Overview

There are various options for re-entering a NC program. Each option has different areas of application and possibilities of influencing programs and parameters.

The individual procedures for re-entering are described below.

Procedure	NC program editable	NC block target
NC block search, page 101	Yes	Selectable in the editor
Leaving and returning to contour, page 102	No	Interrupted NC block
Block pre-run, page 102	Yes	Selectable in the editor
Re-entry, page 102	No	RCB mark or breakpoint

Tab. 14-1: Re-entry functions to an NC program

14.2.2 NC block search

The NC block search is the simplest form of re-entering. It allows for re-entering an NC program on each main block of a subprogram. The previous program sequence is not taken into account. The starting block has to be chosen in such a way that all required settings for the correct program sequence are loaded. In case of a shape cutting machine, it is an NC block containing the `TECHTABLE` call.

The NC block search is used for testing and changing subprograms. The block search allows changing the program as well as the involved machine and material settings as all parameters can be reloaded via the `TECHTABLE` call (see chapter 15 "Technology tables" on page 105).

Requirements

- The NC program has to receive numbered NC blocks
- Reset the channel

Procedure

1. Changing the NC program or the material parameters
2. Select "Block search" under "Program functions".
3. Position the cursor on the starting block in the NC editor
4. Press the "Apply block" F key

The program is started at the selected block.

☞ The block number does not have to exist more than once as the first block found with this number is always jumped at.

14.2.3 Leaving and returning to contour

This function is used to set the zero position and the jet width (see chapter 13 "Cutting width compensation (kerf)" on page 97). Unlike the NC block search, this procedure does not require a channel reset. The NC program cannot be edited during interruption.

The program is re-entered at the start, at the end or at the point of abortion of the interrupted NC block.

Requirements

- The NC program has been stopped but not reset

Procedure

1. Select the "Retrace and Re-enter" function
2. Lift and, if required, move the cutting head
3. Move the zero position or edit the D-correction
4. Select "Re-enter"
5. Check the settings for repositioning
6. Start NC program

   The program is continued with the changed settings.

14.2.4 Block pre-run

The block pre-run function allows for re-entering a NC program on any NC block. The program sequence up to the re-entry point is considered. The NC program can be edited during interruption.

Requirements

- Target block is not locked

  `SD.SysChSRun.LockTargBlock = 0`
- A channel reset has been carried out

Procedure

1. Changing the NC program or the material parameters
2. Select the "Block scan" function under "Program functions"
3. Select the target NC block and confirm with "Apply block"
4. Check the restart options
5. Start NC program

   The NC program is continued at the specified point.

14.2.5 Re-entry

The re-entry function allows the restart of a NC program at specific points marked with the `RCB` command.

Additional conditions can be added to the RCB call in order to allow for re-start at a specific meter reading in a loop, for example.

For detailed information on the RCB command, see the documentation MTX Programming Manual (see chapter 1.4 "Required and supplementing documentation MTX" on page 2).

Requirements

- Target block is not locked

  `SD.SysChSRun.LockTargBlock = 0`

- A channel reset has been carried out
- The NC program contains RCB marks or a breakpoint has been saved before the channel reset (`SD.SysChSRun.ReEntryStore = 1`)

Procedure

1. Select the "Re-entry" function under "Program functions"
2. Select "ReEntry List"
3. Select the RCB mark or the breakpoint
4. Check the re-entry options
5. Select the "Start re-entry" function
6. Start the NC program

   The NC program is started at the selected re-entry point.

In order to continue the program sequence at the last saved re-entry point, the following shorter sequence is possible:

1. Select the "Re-entry" function
2. Select "ReEntry last RCB"
3. Start the NC program

   The NC program is started at the last re-entry point.

In order to continue the program sequence at the last breakpoint, the following shorter sequence is possible:

1. Select the "Re-entry" function
2. Select "ReEntry last Breakpoint"
3. Start the NC program

   The NC program is started at the last re-entry point.

14.3 Retrace

The Retrace functionality allows typing on a contour that has already been run in the normal program sequence after a breakpoint. The required NC programs are internally generated in the control. CPL commands and M functions are not mapped, with a few exceptions.

After typing, the Program can be continued directly at the point.

Parameterization

The Retrace functionality has to be parameterized in order to be used. For this purpose, the number of NC blocks that can be stored for the Retrace has to be configured. After changing the parameter, the NC has to be restarted.

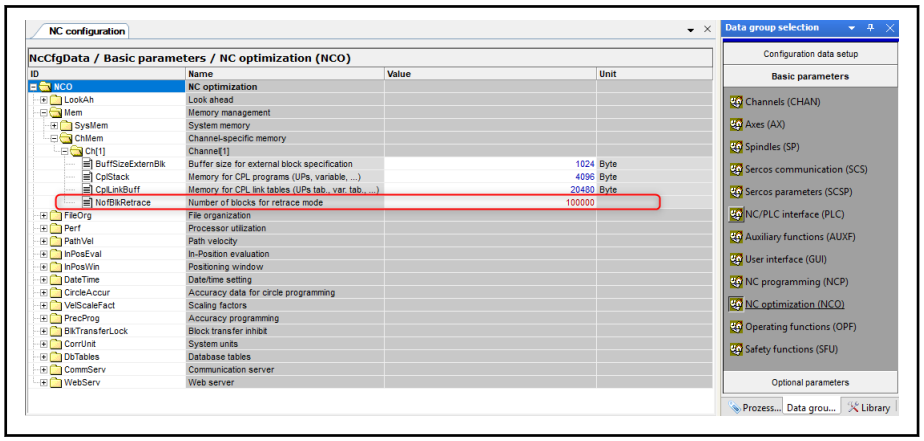


Fig. 14-1: Parameterization of the NC blocks usable for the Retrace function

Activation in the NC program

The area of a NC program to be covered by the Retrace begins with the call `RETRACE(1)` and ends with the call of `RETRACE(0)`.

Program:

```
N100 RETRACE(1)
N110 G1  X253 F[#FEED] ADSLAB(8)
N120 G3 I0 J-11.5 ADSLAB(-8)
N130 G1  X223 ADSLAB(8)
N140 G3 I0 J-11.5 ADSLAB(-8)
N150 G1  X193 ADSLAB(8)
N160 G3 I0 J-11.5 ADSLAB(-8)
N170 G1  X181.5
N180 RETRACE(0)
```

Fig. 14-2: Normal NC program with active Retrace

PLC Interface

The following variables in the PLC channel interface control the Retrace functionality. They are operated from the HMI.

`qCh_Retrace`	TRUE: Activates Retrace mode FALSE: Deactivates Retrace mode
`qCh_RetraceMoveFwd`	TRUE: Forward movement on the path
`qCh_RetraceModeRev`	TRUE: Backward movement on the path
`iCh_RetraceAct`	Shows whether the Retrace mode has been activated

Tab. 14-2: PLC interface for Retrace

15 Technology tables

15.1 Introduction

For processing of the materials, appropriate settings on the machine and the technology functions are required. Therefore, the cutting velocity and the laser power, for example, have to be adapted to the sheet thickness. Furthermore, various methods of processing such as cutting, engraving and nibbling require different settings.

The settings are summarized in so-called technology or material tables. They are loaded in the NC program during processing. This can also be done in the middle of program in order to change the method of processing.

When loading a technology table, the global machine settings are loaded first and then individual process-dependent parameters are specifically overwritten. This way, a consistent behavior of all material tables is ensured. At the same time, material tables can be used for different machines without changes.

15.2 Structure and storage location

Technology tables are usually located on the file system of the control. If required, an NFS file system, in which the files are centrally stored, can also be registered. All technology tables are located in the directory `/usrfep/techtable` on the file system of the control.

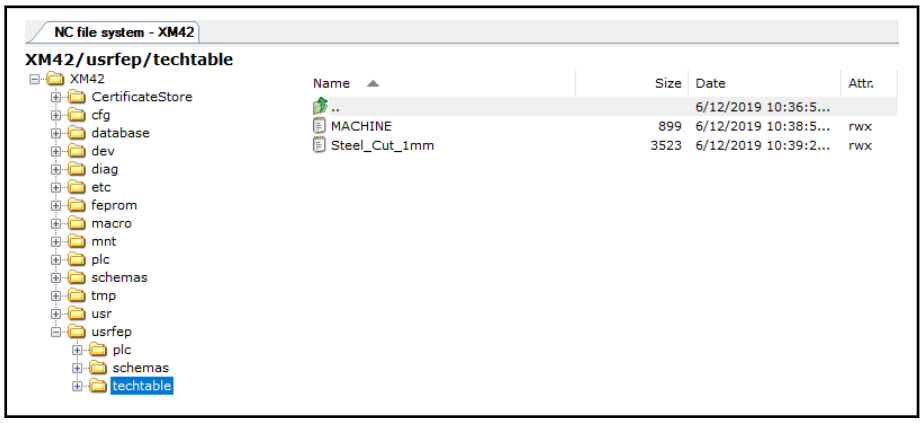


Fig. 15-1: Storage location of the technology tables in the file system of the control

All technology tables are CPL scripts writing values to SD variables. Individual technology functions use them as parameters.

As the machine settings are loaded first, this file should contain a complete set of all parameters. The machine settings include all settings specified by the physics of the machine. Many parameters defined here do not have to be adjusted for individual materials. The machine parameterization is stored in the file `/usrfep/techtable/MACHINE`. It is usually structured as follows:

/usrfep/techtable/MACHINE

```
(ADR_PARA)

;Channel and Axis
  10 CHAN% = SD(8)
  20 SD.SysHeightCtrl[CHAN%].CtrlAxis = 3
  30 PARAID% = SD.SysHeightCtrl[CHAN%].CtrlAxis -1

F  1000 ; Cuting feedrate
F2=5000 ; Positioning feedrate
```

```
;Height Control - Position limit
 100 SD.SysHeightCtrl[CHAN%].PosLimitRel.Min = -300             : REM mm
 101 SD.SysHeightCtrl[CHAN%].PosLimitRel.Max = 30               : REM mm
 102 SD.SysHeightCtrl[CHAN%].PosLimitAbs.Min = MCA(102000002, PARAID%)
 103 SD.SysHeightCtrl[CHAN%].PosLimitAbs.Max = MCA(102000001, PARAID%)

;Height Control - Velocity limit
 110 SD.SysHeightCtrl[CHAN%].Regulation.Vel_Approach = 5000     : REM mm/min
 111 SD.SysHeightCtrl[CHAN%].Regulation.Vel_Regulation_Max = 0.5 * MCA(100500002, PARAID%)
 112 SD.SysHeightCtrl[CHAN%].Regulation.Acceleration_Max = 20   : REM  mm/S2

;Height Control - PID regulation
 115 SD.SysHeightCtrl[CHAN%].Regulation.Gain_P = 0.01
 116 SD.SysHeightCtrl[CHAN%].Regulation.Gain_I = 500            : REM ms
 117 SD.SysHeightCtrl[CHAN%].Regulation.Gain_D = 0              : REM ms
 118 SD.SysHeightCtrl[CHAN%].Regulation.Gain_PAPP = 0.01

;Height Control - Sensor config
 120 SD.SysHeightCtrl[CHAN%].Sensor.OutOfRangeValue = 15        : REM mm
 121 SD.SysHeightCtrl[CHAN%].Sensor.CalibPoints = 100
 122 SD.SysHeightCtrl[CHAN%].Sensor.CollisionSensorVal = 1000   : REM mV
 123 SD.SysHeightCtrl[CHAN%].Sensor.CalibFilename = "/SensorCalib"

;Height Control - Height sensing
 130 SD.SysHeightCtrl[CHAN%].HeightSensing.InPosTime = 20       : REM ms
 131 SD.SysHeightCtrl[CHAN%].HeightSensing.InPosWindow = 0.1    : REM mm
 132 SD.SysHeightCtrl[CHAN%].HeightSensing.InPosTimeout = 3000  : REM ms
 134 SD.SysHeightCtrl[CHAN%].HeightSensing.OvershootRatio = 0.0 : REM %

;Height Control - Collision detection
;mode selection of Collision detection:
;1: collision detection with pre contact, other configuration is reserved
 140 SD.SysHeightCtrl[CHAN%].Collision.Mode = 1
 141 SD.SysHeightCtrl[CHAN%].Collision.RetractWay = 50          : REM mm
 142 SD.SysHeightCtrl[CHAN%].Collision.PreCollisionDist = 0.1   : REM mm
 143 SD.SysHeightCtrl[CHAN%].Collision.Delay = 5                : REM ms

;Height Control - Set point
 150 SD.SysHeightCtrl[CHAN%].SetPoint = 3.0                     : REM mm

;HeightControl-Gap detection
 160 SD.SysHeightCtrl[CHAN%].Gapdetection.Inclination = 10      : REM Degree
 161 SD.SysHeightCtrl[CHAN%].Gapdetection.Delay = 0.0           : REM ms
 162 SD.SysHeightCtrl[CHAN%].Gapdetection.DistUp = 0.0          : REM mm

;Fly Cut
;Mode=0(disable type1 and type2 use ADSLAB )
;Mode=1(enable type1 and type2: PWM and OVS)
 200 SD.SysFlyCut[CHAN%].Mode = 0
 201 SD.SysFlyCut[CHAN%].DistOn = 0.0                           : REM mm
 202 SD.SysFlyCut[CHAN%].DistOff = 0.0                          : REM mm
 203 SD.SysFlyCut[CHAN%].TimeOn = 0.0                           : REM ms
 204 SD.SysFlyCut[CHAN%].TimeOff = 0.0                          : REM ms

; ADS parameters
 210 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].DistOn= 0      : REM mm
 211 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].DistOff= 0     : REM mm
 212 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].TimeOn= 0      : REM ms
 213 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].TimeOff= 0     : REM ms
 214 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].DistCalc=0     : REM 0 is current plane, 1 is space
 215 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].Duration=-1    : REM -1:the signal level is only
                                                               : switched on by ADSLAB()
 216 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].Type=10
 217 SD.SysAdvDistSig[CHAN%].Sig[OUTPUTCHANNEL%].Index=0

;LeapFrog
;LeapFrog mode selection: 0, normal use case; 1, invert-U case
 300 SD.SysLeapFrog[CHAN%].Mode = 0                             : REM mm
 301 SD.SysLeapFrog[CHAN%].R_MaxHeight = 30                     : REM mm
 301 SD.SysLeapFrog[CHAN%].R_Safe = 0.6                         : REM %
 301 SD.SysLeapFrog[CHAN%].L_Safe = 30                          : REM mm
 302 SD.SysLeapFrog[CHAN%].L0 = 0                               : REM mm
 303 SD.SysLeapFrog[CHAN%].L1 =  0                              : REM mm
 304 SD.SysLeapFrog[CHAN%].MinDist = 1                          : REM mm, 0 for MinDist check disabled
 305 SD.SysLeapFrog[CHAN%].Feedrate = 20000                     : REM mm/min, 0 for max feedrate
                                                               : machine parameter)
 306 SD.SysLeapFrog[CHAN%].CornerRadius   = 0                   : REM mm, only for invert-U leapfrog

; Gas control parameters
; SensorType 0: For open loop control
; SensorType 1: Analog sensor for pressure feedback
; SensorType 2: Digital sensor
 400 SD.SysGasCtrl[CHAN%].SensorType = 0
```

```
 401 SD.SysGasCtrl[CHAN%].GasType = 1
 402 SD.SysGasCtrl[CHAN%].GasPressure = 2                    : REM bar
 403 SD.SysGasCtrl[CHAN%].GasTime = 2000                     : REM ms
 404 SD.SysGasCtrl[CHAN%].WaitTimeout = 2000                 : REM ms, Gas stable timeout during gas on
 405 SD.SysGasCtrl[CHAN%].UnstableTimeout = 2000             : REM ms. Gas stable timeout during
                                                             : normal operation
 406 SD.SysGasCtrl[CHAN%].StableWindow = 0.2                 : REM bar

; Focus control parameters
; SensorType 0: For open loop control
; SensorType 1: Close loop with POS.REACHED signal
 500 SD.SysFocusCtrl[CHAN%].SensorType = 0
 501 SD.SysFocusCtrl[CHAN%].FocalPos = 1                     : REM mm
 502 SD.SysFocusCtrl[CHAN%].WaitTime = 100                   : REM ms
 503 SD.SysFocusCtrl[CHAN%].InPosTimeout = 6000              : REM ms


;PowerControl parameters
;;Power control mode setting
; Analog mode select:
;  0: Not used, 1: Fixed power, 2: Power/Speed linear, 3: Power/Speed curve, 4: Power/Time curve
 600  SD.SysPowerCtrl[CHAN%].Analog.Mode = 1
; PWM module mode select:
;  0: Not used, 1: Fixed power, 2: Power/Speed linear line, 3: Power/Speed curve, 4: Power/Time curve
;  5: OVS: Fixed frequency and duty, 6: OVS: Duty/Speed linear line,
;  7: OVS Duty/Speed curve, 8: OVS Duty/Time curve,
;  9: Fixed TotalTime and OnTime, laser fire continuously with delay time. (Frequency = 1000*1000/
TotalTime)
 601  SD.SysPowerCtrl[CHAN%].PWMMode = 0

;;Analog mode 1: fix normal power
;;Analog mode 2: power/speed linear line
 610  SD.SysPowerCtrl[CHAN%].Analog.NormalPower = 2000          : REM mV range(0-10000)
 612  SD.SysPowerCtrl[CHAN%].Analog.SpeedLinear.LowestPower = 1000          : REM mV (0-10000)
 613  SD.SysPowerCtrl[CHAN%].Analog.SpeedLinear.LowerPoint.Velocity = 900.0   : REM mm/min
 614  SD.SysPowerCtrl[CHAN%].Analog.SpeedLinear.LowerPoint.Power = 2000       : REM mV (0-10000)
 615  SD.SysPowerCtrl[CHAN%].Analog.SpeedLinear.UpperPoint.Velocity =  5000.0 : REM mm/min
 616  SD.SysPowerCtrl[CHAN%].Analog.SpeedLinear.UpperPoint.Power = 10000      : REM mV (0-10000)

;;Analog mode 3: power/speed curve: file path max length 100, file name max length 30
 617  SD.SysPowerCtrl[CHAN%].Analog.SpeedCurve.FilePath = "/usrfep/curveparas/Ana_SpeedCurve.txt"
;;Analog mode 4: time/speed curve
 618  SD.SysPowerCtrl[CHAN%].Analog.TimeCurve.FilePath = "/usrfep/curveparas/Ana_TimeCurve.txt"
 619  SD.SysPowerCtrl[CHAN%].Analog.TimeCurve.Cycles = 1                    : REM TimeCurve cycles

;;Duty mode selection
;;Duty mode 1: fix normal power;
 620 SD.SysPowerCtrl[CHAN%].Duty.Mode = 1
 621  SD.SysPowerCtrl[CHAN%].Duty.NormalPower = 0                           : REM percent range (0-100)

;;Duty mode 2: power/speed linear line
 622  SD.SysPowerCtrl[CHAN%].Duty.SpeedLinear.LowestPower = 15              : REM percent range (0-100)
 623  SD.SysPowerCtrl[CHAN%].Duty.SpeedLinear.LowerPoint.Velocity = 850.0   : REM mm/min
 624  SD.SysPowerCtrl[CHAN%].Duty.SpeedLinear.LowerPoint.Power = 20         : REM percent range (0-100)
 625  SD.SysPowerCtrl[CHAN%].Duty.SpeedLinear.UpperPoint.Velocity =  8000.0 : REM mm/min
 626  SD.SysPowerCtrl[CHAN%].Duty.SpeedLinear.UpperPoint.Power = 100        : REM percent range (0-100)

;;Duty mode 3: power/speed curve: file path max length 100, file name max length 30
 627  SD.SysPowerCtrl[CHAN%].Duty.SpeedCurve.FilePath = "/usrfep/curveparas/Duty_SpeedCurve.txt"

;;Duty mode 4: power/time curve
 628  SD.SysPowerCtrl[CHAN%].Duty.TimeLinear.LowestPower = 0                : REM percent range (0-100)
 629  SD.SysPowerCtrl[CHAN%].Duty.TimeLinear.LowerPoint.Time = 0            : REM ms
 630  SD.SysPowerCtrl[CHAN%].Duty.TimeLinear.LowerPoint.Power = 10          : REM percent range (0-100)
 631  SD.SysPowerCtrl[CHAN%].Duty.TimeLinear.UpperPoint.Time =  3000        : REM ms
 632  SD.SysPowerCtrl[CHAN%].Duty.TimeLinear.UpperPoint.Power = 80          : REM percent range (0-100)

; Frequency control parameters
; Frequency mode selection
; mode 1: fix normal frequency
 700  SD.SysPowerCtrl[CHAN%].Freq.Mode = 1
 701  SD.SysPowerCtrl[CHAN%].Freq.NormalPower = 1000                        : REM Hz

; mode 2: fix normal frequency
 702  SD.SysPowerCtrl[CHAN%].Freq.SpeedLinear.LowestPower = 1000            : REM Hz
 703  SD.SysPowerCtrl[CHAN%].Freq.SpeedLinear.LowerPoint.Velocity = 1000.0  : REM mm/min
 704  SD.SysPowerCtrl[CHAN%].Freq.SpeedLinear.LowerPoint.Power = 1000       : REM Hz
 705  SD.SysPowerCtrl[CHAN%].Freq.SpeedLinear.UpperPoint.Velocity =  8000.0 : REM mm/min
 706  SD.SysPowerCtrl[CHAN%].Freq.SpeedLinear.UpperPoint.Power = 5000       : REM Hz

; mode 3: file path max length 100, file name max length 30
 707  SD.SysPowerCtrl[CHAN%].Freq.SpeedCurve.FilePath = "/usrfep/curveparas/Freq_SpeedCurve.txt"
```

Only the deviating parameters are defined in the individual technology tables. Such a file looks as follows:

Example: /usrfep/techtable/Steel_Cut_1mm

```
;Feedrate
F10000

;Height Control
100 SD.SysheightControl[CHAN%].SetPoint = 3 : REM mm

;LeapFrog
200 SD.SysLeapFrog[CHAN%].Height = 20       : REM mm
210 SD.SysLeapFrog[CHAN%].L_Safe = 60       : REM mm

;Power Control
300 SD.SysPowerCtrl[CHAN%].Analog.SpeedLinear.EnergyLevel = 0.6
```

For a summary of all parameters, go to the chapter Overview of SD variables, page 108.

15.3 Loading technology tables: TECHTABLE

A technology table is loaded using the macro `TECHTABLE`. The file name of the technology table to be used is specified as parameter. Up to three technology tables can be uploaded using the command. The command without parameters resets all settings to the default machine table `"MACHINE"`.

```
TECHTABLE("P1","P2","P3")
```

The technology table can be loaded in the NC program at any time. It is recommended to execute the `TECHTABLE` command at the beginning of the NC program. A certain processing time (< 100 ms) is required for the loading.

Program:

```
;Load default MACHINE table
N100 TECHTABLE()
;Reset to default MACHINE table, then load 3 material tables successively
N200 TECHTABLE("Al_8.0_1000W_N2_12345","Piercing001","Quality001")
;Reset to default MACHINE table, then load quality table
N300 TECHTABLE("Quality001")
;Update the SD variable in quality table, and the other configurations remain
;the previous value
N400 TECHTABLE(,"Quality001")
```

Fig. 15-2: Example of the TECHTABLE command call

15.4 Overview of SD variables

All setting parameters for technology functions are stored in so-called system data or SD variables. The NC and the PLC can read and write them.

SD variables enable a structured storage of all setting parameters. They are generally assigned with values using the technology tables.

In the following, all SD variables used for shape cutting technology are listed according to function groups.

Height control

Parameter	Unit	Description
SD.SysHeightCtrl[*].SetPoint	mm	SetPoint, nominal value distance between the machining head and the material surface
SD.SysHeightCtrl[*].Regulation.Vel_Approach	mm/min	Approaching velocity of the machining head
SD.SysHeightCtrl[*].Regulation.Vel_Regulation_Max	mm/min	Upper limit of the travel velocity in z-direction
SD.SysHeightCtrl[*].Regulation.Acceleration_Max	mm/s²	Max. acceleration allowed
SD.SysHeightCtrl[*].Regulation.Gain_P		Controller parameters: P-value
SD.SysHeightCtrl[*].Regulation.Gain_I		Controller parameters: I-value
SD.SysHeightCtrl[*].Regulation.Gain_D		Controller parameters: D-value
SD.SysHeightCtrl[*].Regulation.Gain_PAPP		Controller parameters: P-controller for the approach
SD.SysHeightCtrl[*].HeightSensing.OvershootRatio	%	Maximum overshooting permitted: (Earth position - MaxAmpl.) / (Earth position - SetPoint) 0: No overshooting permitted
SD.SysHeightCtrl[*].HeightSensing.InPosTime	ms	In-Pos-Time
SD.SysHeightCtrl[*].HeightSensing.InPosWindow	mm	In-Pos-Window
SD.SysHeightCtrl[*].HeightSensing.InPosTimeout	ms	In-Pos-Timeout
SD.SysHeightCtrl[*].PosLimitRel.Min	mm	Workspace parameters: Lower limit of the relative motion
SD.SysHeightCtrl[*].PosLimitRel.Max	mm	Workspace parameters: Upper limit of the relative motion
SD.SysHeightCtrl[*].PosLimitAbs.Min	mm	Lower limit of the absolute position
SD.SysHeightCtrl[*].PosLimitAbs.Max	mm	Upper limit of the absolute position
SD.SysHeightCtrl[*].Sensor.OutOfRangeValue	mm	Out-of-range value, maximum measuring range of the sensor
SD.SysHeightCtrl[*].Sensor.CalibPoints		Number of the measuring points for the sensor calibration
SD.SysHeightCtrl[*].Sensor.CalibFilename		File name to store the calibration file
SD.SysHeightCtrl[*].Sensor.CollisionSensorVal		Sensor value for collision detection
SD.SysHeightCtrl[*].Collision.Mode		1: Collision detection due to pre-contact; Other values: Reserved
SD.SysHeightCtrl[*].Collision.RetractWay	mm	Retract path during collision
SD.SysHeightCtrl[*].Collision.PreCollisionDist	mm	Limit of the collision area, pre-contact value. Setting range: 0.0 < PreCollisionDist < SetPoint
SD.SysHeightCtrl[*].Collision.Delay	ms	Time after which a collision was detected before a retract motion is initiated

Parameter	Unit	Description
SD.SysHeightCtrl[*].Gapdetection.Inclination	degree	Inclination of the gap edge: acttan(z2-z1)/(xy2-xy1)
SD.SysHeightCtrl[*].Gapdetection.Delay	ms	Time after which a gap was detected before a retract motion is initiated
SD.SysHeightCtrl[*].Gapdetection.DistUp	mm	Retract distance after detecting a gap/an edge

* Channel number

Tab. 15-1: Setting parameters of the height control

LeapFrog

Parameter	Unit	Description
SD.SysLeapFrog[*].Height	mm	Maximum height of the safe area
SD.SysLeapFrog[*].L_Safe	mm	Maximum distance between the starting point and reaching the height defined in R_Safe in mm
SD.SysLeapFrog[*].L0	mm	Lifting height at the starting point
SD.SysLeapFrog[*].L1	mm	Safe height at the target point
SD.SysLeapFrog[*].MinDist	mm	Defines a minimum LeapFrog distance from which the height at the target point is not L1, but L0. A value of 0 switches off the function.
SD.SysLeapFrog[*].Feedrate2	mm/min	Maximum path velocity during the LeapFrog.

* Channel number

Tab. 15-2: LeapFrog setting parameters

Laser power control

Parameter	Unit	Description
SD.SysPowerCtrl[*].AnalogMode		0: No analog signal is used 1: Fix power value 2: Linear velocity/power control 3: Velocity/power curve 4: Time/power curve
SD.SysPowerCtrl[*].Analog.NormalPower	mV	specifies the fixed power. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedLinear.Lowest-Power	mV	sets the voltage for velocities below the minimum velocity. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedLinear.Lower-Point.Velocity	mm/min	sets the minimum velocity for the linear conversion.
SD.SysPowerCtrl[*].Analog.SpeedLinear.Lower-Point.Power	mV	specifies the voltage at minimum velocity. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedLinear.Upper-Point.Velocity	mm/min	sets the maximum velocity for the linear conversion.
SD.SysPowerCtrl[*].Analog.SpeedLinear.Upper-Point.Power	mV	specifies the voltage at maximum velocity. Value range: 0 .. 10000 mV

Parameter	Unit	Description
SD.SysPowerCtrl[*].Analog.SpeedCurve.FilePath		Defines the storage location of the curve file as string.
SD.SysPowerCtrl[*].Analog.TimeCurve.FilePath		Defines the storage location of the curve file as string.
SD.SysPowerCtrl[*].Analog.TimeCurve.Cycles		specifies the number of repetitions of the time curve.
SD.SysPowerCtrl[*].PWM.Mode		0: No PWM signal is used 1: Fix duty value 2: Linear velocity/duty control 3: Velocity/duty curve
SD.SysPowerCtrl[*].Duty.NormalPower	%	sets the fixed duty. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedLinear.Lowest-Power	%	sets the duty for velocities below the minimum velocity. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Velocity	mm/min	sets the minimum velocity for the linear conversion in mm/min.
SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Power	%	sets the duty at minimum velocity. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Velocity	mm/min	sets the maximum velocity for the linear conversion in mm/min.
SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Power	%	sets the duty at maximum velocity. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedCurve.FilePath		Defines the storage location of the curve file as string.
SD.SysPowerCtrl[*].OVS.Frequency	Hz	Frequency for the PWM

* Channel number

Tab. 15-3: Setting parameters for the laser power control

Gas actuation

Parameter	Unit	Description
SD.SysGasCtrl[*].SensorType		Pressure sensor type 0: Without sensor 1: Analog sensor 2: Digital sensor (pressure reached)
SD.SysGasCtrl[*].GasType		Selected gas type 1 ... 4
SD.SysGasCtrl[*].GasPressure	bar	Gas pressure for analog sensor
SD.SysGasCtrl[*].GasTime	ms	Minimum waiting time to reach the gas pressure
SD.SysGasCtrl[*].WaitTimeout	ms	Maximum waiting time to reach the gas pressure

Parameter	Unit	Description
SD.SysGasCtrl[*].UnstableTimeout	ms	Maximum time till pressure stabilization
SD.SysGasCtrl[*].StableWindow	bar	Window for stable pressure

* Channel number

Tab. 15-4: Setting parameters for the gas control

Focus control

Parameter	Unit	Description
SD.SysFocusCtrl[*].SensorType		Type of focus monitoring 0: Open Loop 1: Closed-loop - Waiting for "FocalPosReached" input
SD.SysFocusCtrl[*].FocalPos	mm	Target focus position
SD.SysFocusCtrl[*].WaitTime	ms	Waiting time for focus (only type 0)
SD.SysFocusCtrl[*].InPosTimeout	ms	Minimum waiting time to reach the focus

* Channel number

Tab. 15-5: Setting parameters for the focus control

FlyCut

System variable	Unit	Description
SD.SysFlyCut[*].Mode		0, ADSLAB; 1, PWM or OVS
SD.SysFlyCut[*].DistOn	mm	Indicates the switch-on distance in mm relative to the end point of the NC block
SD.SysFlyCut[*].DistOff	mm	Indicates the switch-off distance in mm relative to the end point of the NC block
SD.SysFlyCut[*].TimeOn	ms	Indicates the switch-on point in time in ms relative to the end point of the NC block
SD.SysFlyCut[*].TimeOff	ms	Indicates the switch-off point in time in ms relative to the end point of the NC block

* Channel number

Tab. 15-6: FlyCut settings

ADSLAB

System variable	Unit	Description
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].DistOn	mm	Indicates the switch-on distance in mm relative to the end point of the NC block
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].DistOff	mm	Indicates the switch-off distance in mm relative to the end point of the NC block
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].TimeOn	ms	Indicates the switch-on point in time in ms relative to the end point of the NC block
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].TimeOff	ms	Indicates the switch-off point in time in ms relative to the end point of the NC block

System variable	Unit	Description
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].DistCalc	-	0: Coordinates of the active plane 1: Coordinates of the active space
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].Duration	-	-1: The signal level is only switched on by ADSLAB () 0.5 - 10000.0: Signal duration
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].Type	-	0: iCh_Customx; 1: iCh_Cplxx; 10: iCh_Customx + The HS output of the PosDepHSOut function; 11: iCh_Cplxx + The HS output of the PosDepHSOut function
SD.SysAdvDistSig[*].Sig[OUTPUTCHANNEL%].Index	-	0: (not used). No IF signal is used 1 ... 8: Index for type = 0 and 10 (iCh_Custom1 ... iCh_Custom8) 1 .. 16: Index for type = 1 and 11 (iCh_Cpl01 ... iCh_Cpl16)

* Channel number

Tab. 15-7: FlyCut settings

16 MT_Technology

16.1 Introduction and overview

General information

Technology function blocks (Tech FB) extend the basic functionality of the target system MTX and provide application-specific functionalities. The described library includes technology functions that are used in the shape cutting area or for other applications. All function blocks can also be used individually.

The function blocks described are provided via the "MT_Technology" library for MTX.

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

Requirements

Technology function blocks of the "MT_Technology" library require the firmware support of the target system. Specific prerequisites of the technology function blocks are documented with the corresponding function blocks.

PLC libraries used

The PLC library MT_Technology uses the following libraries internally:

- RIL_CommonTypes
- RIL_Utilities
- RIL_LoopControl
- MT_Base

All libraries used are automatically included internally into the correct library version using library placeholders.

Areas of application and function blocks of the MT_Technology

Function blocks	Description
Height control	
MT_HeightRegulationType01, page 117	Monitors and controls the distance between the machining head and the material
MT_DistanceSensorType01, page 121	Calibration, filtering and linearization of the measured value of a general distance sensor
MT_DistanceSensorType02, page 124	Calibration, filtering and linearization of the measured value of a Precitec distance sensor
MT_HCSimType01, page 126	Generating a vertical distance between the z-position and the simulated material
MT_HCSimType02, page 128	Simulating a rectangular material surface with a configurable starting point position.
MT_PlateSearchType01, page 130	Determining the positioning offset of a rectangular material
MT_PlateSearchType02, page 133	Determining the positioning offset of a rectangular material in the emulation
MT_RaiseAxis, page 138	Lifting the Z-axis to the specified position
EN_HC_STATE_t, page 136	State of the height control and sensor calibration
EN_HC_USRMODE_t, page 137	Height control modes
LeapFrog	
MT_LeapFrogType01, page 140	Implements the LeapFrog auxiliary function

Power control	
MT_GetPowerCtrlMode, page 145	Decodes the settings for the laser power control from a system variable
MT_LaserPowerControlType01, page 146	Controls the laser power depending on the cutting head velocity above the material at an analog output
MT_LaserPowerControlType02, page 149	Controls the laser power depending on the cutting head velocity above the material at a PWM output
MT_LaserPowerControlType03, page 151	Controls the laser power depending on the cutting head velocity above the material with a PWM signal on a S20 oversampling module
Gas control	
MT_GasControlType01, page 154	Controlling and monitoring the gas valves of the cutting head
Focus control	
MT_FocusControlType01, page 156	Controlling and monitoring the focus of the laser cutting head
FlyCut	
MT_FlyCutType01, page 158	Position-accurate activation and deactivation of the cutting jet in the Sercos cycle
MT_FlyCutType02, page 161	Position-accurate activation and deactivation of the cutting jet with an accuracy of up to 10 µs on a S20 oversampling module.
Auxiliary functions for the S20 oversampling module	
MT_Oversampling_AND, page 165	AND on a S20 oversampling module per channel.
MT_Oversampling_OR, page 167	OR on a S20 oversampling module per channel.

Tab. 16-1: MT_Technology library

16.2 Height control

16.2.1 Introduction and overview

For shape cutting machines, a cutting head is horizontally guided over the workpiece to be machined in the x- and y-coordinate. As the workpiece is usually not completely even and the distance between the machining head and the workpiece has to be kept constant during processing, a height control of the machining head is required. For this purpose, the z-coordinate is controlled by a distance to the material surface detected by a sensor.

The function blocks described in this chapter cover this application case.

- MT_HeightRegulationType01, page 117 keeps a constant distance between the machining head and a material that is nearly flat.
- MT_DistanceSensorType01, page 121 linearizes and filters the raw measured value and generates an evaluable measurement signal from the general distance sensor.
- MT_DistanceSensorType02, page 124 linearizes and filters the raw measured value and generates an evaluable measurement signal from the Precitec distance sensor.
- MT_HCSimType01, page 126, computes the vertical distance between the z-axis and the simulated material.
- MT_HCSimType02, page 128, simulates a rectangular material surface with a configured starting point position.
- MT_PlateSearchType01, page 130, determines the position offset and the orientation offset of the rectangular material.

- MT_PlateSearchType02, page 133, determines the position offset and the orientation offset of the rectangular material in the emulation.
- EN_HC_STATE_t, page 136, provides the states of the height control and the sensor calibration.
- EN_HC_USRMODE_t, page 137, provides the height control modes.
- MT_RaiseAxis, page 138, lifts the z-axis to a pre-defined position.

16.2.2 MT_HeightRegulationType01

Brief description

The MT_HeightRegulationType01 function block is used to keep a constant distance between the machining head and a material that is nearly flat.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-2: Reference table of the MT_HeightRegulationType01

Interface description

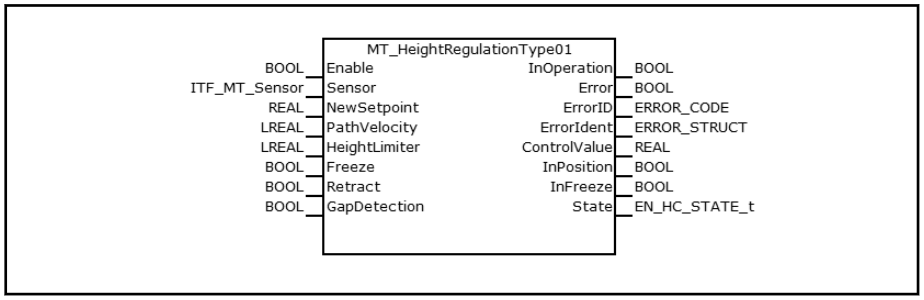


Fig. 16-1: MT_HeightRegulationType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Sensor	ITF_MT_Sensor	Function block for the sensor processing
	NewSetpoint	REAL	Distance to be kept from the surface
	PathVelocity	LREAL	Path velocity from the NC
	HeightLimiter	LREAL	Distance between the setpoint and the “Freeze” position
	Freeze	BOOL	Prevents the lowering under the current z-value, used for LeapFrog
	Retract	BOOL	Retract position of a cutting head FALSE: Cutting head in normal position TRUE: Cutting head is lifted to a safe position

I/O type	Name	Data type	Description
	GapDetection	BOOL	Configuration detection of gaps in the material FALSE: Detection of gaps and cracks in the material if the height control is stable. TRUE: Continuous detection of gaps and cracks in the material. Note: Disabling the gap detection by SD config: `SD.SysHeightCtrl.Gapdetection.Delay >= SD.SysHeightCtrl.HeightSensing.InPosTimeout`
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 121
	ControlValue	REAL	Correction value for the z-coordinate
	InPosition	BOOL	Sensor is regulated and oscillated
	InFreeze	BOOL	"Freeze" is active. The "ControlValue" output cannot be lowered.
	State	EN_HC_STATE_t	Machining state of the height control

Tab. 16-3: Interface variables of the MT_HeightRegulationType01 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
Sensor	ITF_MT_Sensor			NULL	Once, at a rising edge at "Enable"
NewSetpoint	REAL	0.0	0.5 * sensor, measuring range	0.0	Continuous if "Enable" is TRUE
PathVelocity	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
HeightLimiter	LREAL	0.0	N. def.	0.0	Continuous if "Enable" is TRUE
Freeze	BOOL			FALSE	Continuous if "Enable" is TRUE
Retract	BOOL			FALSE	Continuous if "Enable" is TRUE
GapDetection	BOOL			FALSE	Continuous if "Enable" is TRUE

Tab. 16-4: Overview on the min./max. values and default values of the inputs

Functional description

The MT_HeightRegulationType01 function block is used to keep a constant distance between the machining head and a material that is nearly flat.

It requires the current sensor value of the distance sensor. This value is provided in a compatible sensor function block (e.g. chapter 16.2.3 "MT_DistanceSensorType01" on page 121) at the "Sensor" entry. Furthermore, the distance to be kept at the "Setpoint" input is required. The "PathVelocity" input is required to approach the machining head to the material. The correction value for the z-coordinate is output at the "ControlValue" output.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- PathVelocity: SD.SysIpoPathFeed[*].ActFeed

Normal operation

Each cycle, the MT_HeightRegulationType01 computes a new value for the "ControlValue" output.

Configuration

The configuration contains all settings for the height control. The function block applies the following system variables when calling the `MT_DistanceSensorType01/02.UpdateConfig()` method on the function block instance:

Parameter	Unit	Description
`SD.SysHeightCtrl[*].CtrlAxis`		Axis number of the z-axis
`SD.SysHeightCtrl[*].SetPoint`	mm	SetPoint, nominal distance between the cutting head and the material surface
`SD.SysHeightCtrl[*].Regulation.Vel_Regulation_Approach`	mm/min	Approaching velocity of the machining head
`SD.SysHeightCtrl[*].Regulation.Vel_Regulation_Max`	mm/min	Upper limit of the travel velocity in Z-direction, reference to compute the P-controller
`SD.SysHeightCtrl[*].Regulation.Gain_P`		Controller parameters: P-value
`SD.SysHeightCtrl[*].Regulation.Gain_I`	ms	Controller parameters: I-value
`SD.SysHeightCtrl[*].Regulation.Gain_D`	ms	Controller parameters: D-value
`SD.SysHeightCtrl[*].Regulation.Gain_PAPP`		Controller parameters: P-controller for the approach
`SD.SysHeightCtrl[*].HeightSensing.OvershootRatio`	%	Maximum overshooting permitted: (End position - MaxAmpl.) / (End position - SetPoint) 0: No overshooting permitted
`SD.SysHeightCtrl[*].HeightSensing.Acceleration`	m/s²	Max. permitted acceleration when approaching the material
`SD.SysHeightCtrl[*].HeightSensing.Velocity`	mm/min	Max. permitted axis velocity when approaching the material
`SD.SysHeightCtrl[*].HeightSensing.InPosTime`	ms	In-Pos-Time
`SD.SysHeightCtrl[*].HeightSensing.InPosWindow`	mm	In-Pos-Window

Parameter	Unit	Description
SD.SysHeightCtrl[*].HeightSensing.InPosTimeout	ms	In-Pos-Timeout
SD.SysHeightCtrl[*].PosLimitRel.Min	mm	Workspace parameters: Lower limit of the relative motion
SD.SysHeightCtrl[*].PosLimitRel.Max	mm	Workspace parameters: Upper limit of the relative motion
SD.SysHeightCtrl[*].PosLimitAbs.Min	mm	Lower limit of the absolute position
SD.SysHeightCtrl[*].PosLimitAbs.Max	mm	Upper limit of the absolute position
SD.SysHeightCtrl[*].Sensor.OutOfRangeValue	mm	Out-of-range value, maximum measuring range of the sensor
SD.SysHeightCtrl[*].Sensor.CalibPoints		Number of the measuring points for the sensor calibration
SD.SysHeightCtrl[*].Sensor.CalibFilename		File name to store the calibration file
SD.SysHeightCtrl[*].Sensor.CollisionSensorVal		Sensor raw value below which a collision is detected
SD.SysHeightCtrl[*].Collision.Mode		1: Collision detection by "pre" contact
SD.SysHeightCtrl[*].Collision.RetractWay	mm	Retract path during collision
SD.SysHeightCtrl[*].Collision.PreCollisionDist	mm	"Pre"-contact: Limit of the collision area in mm
SD.SysHeightCtrl[*].Collision.Delay	ms	Time after which a collision was detected before a retract motion is initiated Disabling the collision detection: Delay >= InPosTimeout
SD.SysHeightCtrl[*].Gapdetection.Inclination	degree	Inclination of the gap edge: acttan(z_2-z_1)/(xy_2-xy_1)
SD.SysHeightCtrl[*].Gapdetection.Delay	ms	Time during which a gap was detected before a retract motion is initiated Disabling the gap detection: Delay >= InPosTimeout
SD.SysHeightCtrl[*].Gapdetection.DistUp	ms	Retract motion if gap was detected.

* NC channel, normally 1

Tab. 16-5: Settings of the MT_HeightRegulation function block

Call the `MT_DistanceSensorType01/02.UpdateConfig()` method in a cyclic PLC task.

SetPoint If the input variable `NewSetPoint` is set to 0.0, the standard command value in the Techtable `SD.SysHeightCtrl[*].SetPoint` is used for the height control.

`NewSetPoint` can be changed permanently for an adaptive command value.

Freeze If "Freeze" is set, the machining head cannot continue approaching towards the workpiece in the z-coordinate. The current ControlValue is practically limited downwards.

This function is required for the LeapFrog. The machining head is lifted for a short time to evade obstacles upwards. When repositioning the machining head, the control is active, but it does not actively move the head towards the material.

Retract If the “Retract” input is TRUE, the cutting head returns to the safe height configured to the system variable "SD.SysHeightCtrl[ChannelNr].Collision.RetractWay".

GapDetection The detection of gaps and cracks in the material is controlled by the "GapDetection" input.

- GapDetection = FALSE: Detection of gaps and cracks in the material if the machining head is stable.
- GapDetection = TRUE: Continuous detection of gaps and cracks in the material.
- Disabling the gap detection by SD config: `SD.SysHeightCtrl.Gapdetection.Delay >= SD.SysHeightCtrl.HeightSensing.InPosTimeout`
- The gap is detected by an edge angle `SD.SysHeightCtrl[*].Gapdetection.Inclination`.
- Retract motion after gap detection.

Troubleshooting 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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3121	16#0000	HC configuration error
RESOURCE_ERROR (16#0003)	16#3122	16#0000	Sensor calibration is missing, height control cannot be activated
INPUT_RANGE_ERROR (16#0006)	16#3123	16#0000	Invalid "SetPoint"
OTHER_ERROR (16#7FFE)	16#3124	16#0000	Collision detected
OTHER_ERROR (16#7FFE)	16#3125	16#0000	Timeout while activating the height control

Tab. 16-6: Error codes MT_HeightRegulationType01 function block

16.2.3 MT_DistanceSensorType01

Brief description The MT_DistanceSensorType01 function block filters and linearizes the measured value of a distance sensor.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-7: Reference table of the MT_DistanceSensorType01

Interface description

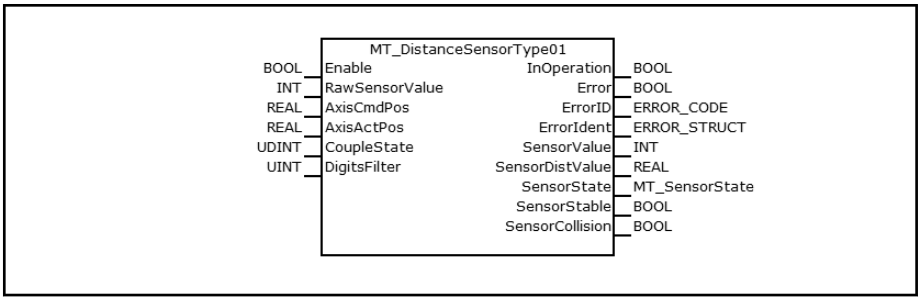


Fig. 16-2: MT_DistanceSensorType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	RawSensorValue	INT	Raw value of the sensor
	AxisCmdPos	REAL	Commanded axis position (required for calibration)
	AxisActPos	REAL	Current axis position (required for calibration)
	CoupleState	UDINT	Axis coupling status of the system axis coupling
	DigitsFilter	UINT	Noise value of the sensor
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 123
	SensorValue	REAL	Linearized and filtered sensor value
	SensorState	HC_SensorState	Function block state (for debug purposes)
	Stable	BOOL	TRUE: Sensor value is stable FALSE: Sensor value is not yet oscillated

Tab. 16-8: Interface variables of the MT_DistanceSensorType01 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
RawValue	INT	-32768	32767	0	Continuous if "Enable" is TRUE
AxisCmdPos	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
AxisActPos	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Name	Type	Min. value	Max. value	Default value	Effective
CoupleState	BOOL	N. def.	N. def.	FALSE	Continuous if "Enable" is TRUE
DigitsFilter	UINT	0	65335	0.0	Continuous if "Enable" is TRUE

Tab. 16-9: Overview on the min./max. values and default values of the inputs

Functional description

The MT_DistanceSensorType01 function block filters and linearizes the measured value of a distance sensor. It is used together with the function block chapter 16.2.2 "MT_HeightRegulationType01" on page 117.

The function block has two states:

1. Calibration of the sensor
2. Filtering and linearization

If the sensor is not calibrated, the "SensorValue" output is set to 0.0.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- RawValue: Sensor value of an analog input (e.g. X32 of an IndraDrive)
- AxisActPos: Current position of the vertical coordinate
- AxisCmdPos: Commanded position of the vertical coordinate
- CoupleState: Axis coupling state

Normal operation

Upon each cycle, the MT_DistanceSensorType01 computes a new filtered value for the "SensorFiltValue" output and a converted measuring distance for the "SensorDistValue" output.

AxisCmdPos, AxisActPos

These two inputs are required to calibrate the sensor. They determine at which height the cutting head is positioned above the material.

CoupleState

The "CoupleState" input includes the axis coupling state. It is read from the system variable SD.SysAxCoupleState[<Axis number>].State which has to be applied in the FA task.

DigitsFilter

The sensor value is usually slightly noisy. The noise increases if the protective gas is activated. The noise can be deducted to receive a signal that is as clean and quick-responding as possible. The noise value has to be specified. This value can easily be determined from a record.

Troubleshooting

The function block generates the following error messages in Additional1/ Additional2 using the "**F_RELATED_TABLE**", 16#3180:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid function block state
INPUT_INVALID_ERROR(16#0001)	16#3181	16#0000	Configuration error
DEVICE_ERROR (16#0008)	16#3182	16#0000	Axis coupling still active
RESOURCE_ERROR (16#0003)	16#3183	16#0000	Timeout during calibration
RESOURCE_ERROR (16#0003)	16#3184	16#0000	Calibration check failed
RESOURCE_ERROR (16#0003)	16#3185	16#0000	Calibration check not possible
DEVICE_ERROR (16#0008)	16#3186	16#0000	Position outside the machine limit

Tab. 16-10: Error codes MT_DistanceSensorType01 function block

16.2.4 MT_DistanceSensorType02

Brief description

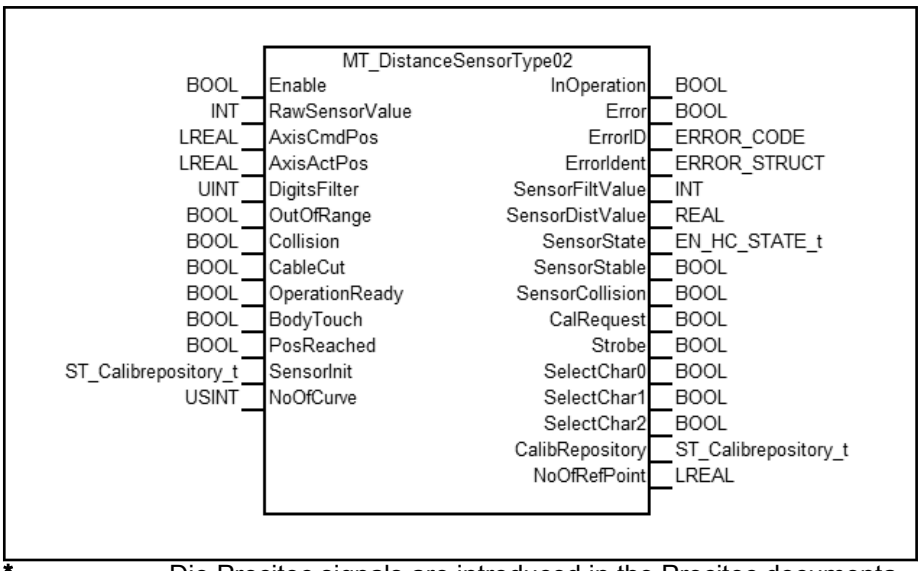
The MT_DistanceSensorType02 function block filters and linearizes the measured value of a Precitec distance sensor.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-11: Reference table of the MT_DistanceSensorType02

Interface description



* Die Precitec signals are introduced in the Precitec documentation TD_EG8030_BA_EN, chapter 2-15.

Fig. 16-3: MT_DistanceSensorType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	RawSensorValue	INT	Raw value of the sensor
	AxisCmdPos	REAL	Commanded axis position (required for calibration)
	AxisActPos	REAL	Current axis position (required for calibration)
	DigitsFilter	UINT	Noise value of the sensor
	OutOfRange	BOOL	Beyond the measuring range PrecitecSignal* A10
	Collision	BOOL	PrecitecSignal A11
	CableCut	BOOL	PrecitecSignal A12
	OperationReady	BOOL	PrecitecSignal A13
	BodyTouch	BOOL	PrecitecSignal A14
	PosReached	BOOL	PrecitecSignal A15
	SensorInit	ST_Calibrepository	Initializing the calibration curve with reference positions
	NoOfCurve	USINT	Number of the active curve
VAR_OUTPUT	InOperation	BOOL	Function block is operating

I/O type	Name	Data type	Description
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 126
	SensorValue	REAL	Linearized and filtered sensor value
	SensorState	HC_SensorState	Function block state (for debug purposes)
	Stable	BOOL	TRUE: Sensor value is stable FALSE: Sensor value is not yet oscillated

Tab. 16-12: Interface variables of the MT_DistanceSensorType02 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
RawValue	INT	-32768	32767	0	Continuous if "Enable" is TRUE
AxisCmdPos	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
AxisActPos	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
DigitsFilter	UINT	0	65335	0.0	Continuous if "Enable" is TRUE
OutOfRange	BOOL	N. def.	N. def.	FALSE	Continuous
Collision	BOOL	N. def.	N. def.	FALSE	Continuous
CableCut	BOOL	N. def.	N. def.	FALSE	Continuous
OperationReady	BOOL	N. def.	N. def.	FALSE	Continuous
BodyTouch	BOOL	N. def.	N. def.	FALSE	Continuous
PosReached	BOOL	N. def.	N. def.	FALSE	Continuous
SensorInit	ST_Calibreposito-ry	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
NoOfCurve	USINT	1	8	1	Continuous if "Enable" is TRUE

Tab. 16-13: Overview on the min./max. values and default values of the inputs

Functional description

The MT_DistanceSensorType02 function block filters and linearizes the measured value of a distance sensor. It is used together with the function block chapter 16.2.2 "MT_HeightRegulationType01" on page 117.

The function block has two states:

1. Calibration of the sensor
2. Filtering and linearization

If the sensor is not calibrated, the "SensorValue" output is set to 0.0.

Circuitry Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- RawValue: Sensor value of an analog input (e.g. X32 of an IndraDrive)
- AxisActPos: Current position of the vertical coordinate
- AxisCmdPos: Commanded position of the vertical coordinate
- DigitsFilter: Band width of the digital filter
- OutOfRange, Collision, CableCut, OperationReady, BodyTouch, PosReached: A11-A15 signals from the PreciTech sensor
- NoOfCurve: Number of the active curve

Normal operation Upon each cycle, the MT_DistanceSensorType02 computes a new filtered value for the "SensorFiltValue" output and a converted measuring distance for the "SensorDistValue" output.

AxisCmdPos, AxisActPos These two inputs are required to calibrate the sensor. They determine at which height the cutting head is positioned above the material.

DigitsFilter The sensor value is usually slightly noisy. The noise increases if the protective gas is activated. The noise can be deducted to receive a signal that is as clean and quick-responding as possible. The noise value has to be specified. This value can easily be determined from a record.

SensorInit The accessing repository for sensor calibration, filtering and linearization. To calibrate the sensor using the cycle command "`PREC16P_CALIB`", `SensorInit.SensorCurve[*].SensorCurveArray[**].Way` initializes the 16 reference positions. The `CalibRepository` output outputs the calibration result. chapter 6.6.5 "Sensor calibration: PREC16P_CALIB" on page 52

If an existing calibration curve is used with the command "`PREC16P_LOAD`", raw measured values are filtered and linearized and exported via the outputs `SensorFiltValue` and `SensorDistValue` In this case, `SensoreInit` remains unchanged and it is output directly as `CalibRepository` output.

* Number of the selected calibration curve
** Number of the reference points

Troubleshooting The function block generates the following error messages in Additional1/ Additional2 using the "**F_RELATED_TABLE**", 16#3180:

ErrorID	Additional1	Additional2	Description
STATE_MACHINE_ERROR (16#0005)	16#0006	16#0000	Invalid function block state
INPUT_INVALID_ERROR(16#0001)	16#3181	16#0000	Configuration error
DEVICE_ERROR (16#0008)	16#3182	16#0000	Axis coupling still active
RESOURCE_ERROR (16#0003)	16#3183	16#0000	Timeout during calibration
RESOURCE_ERROR (16#0003)	16#3184	16#0000	Calibration check failed
RESOURCE_ERROR (16#0003)	16#3185	16#0000	Calibration check not possible
DEVICE_ERROR (16#0008)	16#3186	16#0000	Position outside the machine limit

Tab. 16-14: Error codes MT_DistanceSensorType02 function block

16.2.5 MT_HCSimType01

Brief description The function block MT_HCSimType01 computes the vertical distance between the z-axis and the simulated material.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-15: Reference table of the MT_HCSimType01

Interface description

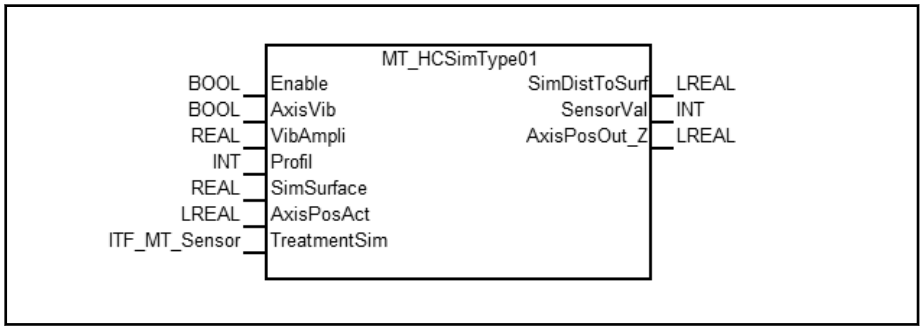


Fig. 16-4: MT_HCSimType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block if set to TRUE
	AxisVib	BOOL	Enabling the fault simulation
	VibAmpli	REAL	Amplitude of the fault signals
	Profile	INT	Profile of the simulated lines: "0" straight line "1" zick-zack
	SimSurface	REAL	Simulated material position
	AxisPosAct	LREAL	Current axis position (required for calibration)
	TreatmentSim	ITF_MT_Sensor	Function block for the sensor processing
VAR_OUTPUT	SimDistToSurf	LREAL	Distance between the z-position and the simulated material
	SensorVal	INT	Simulated sensor measured values
	AxisPosOut	LREAL	Simulated z-position including the fault

Tab. 16-16: Interface variables of the MT_DistanceSensorType01 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	N. def.	N. def.	FALSE	Continuous
AxisVib	BOOL	N. def.	N. def.	FALSE	Continuous if "Enable" is TRUE
VibAmpli	REAL	0.0	N. def.	0.0	Continuous if "Enable" is TRUE
Profile	INT	0	1	0	Continuous if "Enable" is TRUE

Name	Type	Min. value	Max. value	Default value	Effective
SimSurface	REAL	Negative travel limit of the z-axis in the NC configuration	Positive travel limit of the z-axis in the NC configuration	0.0	Continuous if "Enable" is TRUE
AxisActPos	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Tab. 16-17: Overview on the min./max. values and default values of the inputs

Functional description

The function block MT_HCSimType01 computes the vertical distance between the z-axis and the simulated material and outputs the respective sensor measured value. It is used together with the function block chapter 16.2.2 "MT_HeightRegulationType01" on page 117 for the emulation. Disable the function block for the real control and the distance sensor.

Circuitry

Call the function block in a fast Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- Enable: Simulation status `stCtrl.bSimModeAct`
- AxisActPos: Current position of the vertical coordinate

Normal mode

Each cycle, the MT_HCSimType01 computes a distance between the z-position and the simulated material for the "SimDistToSurf" output and the simulated sensor values for the "SensorVal" output.

16.2.6 MT_HCSimType02

Brief description

The function block MT_HCSimType02 simulates a rectangular material surface with a configured starting point position.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-18: Reference table of the MT_HCSimType02

Interface description

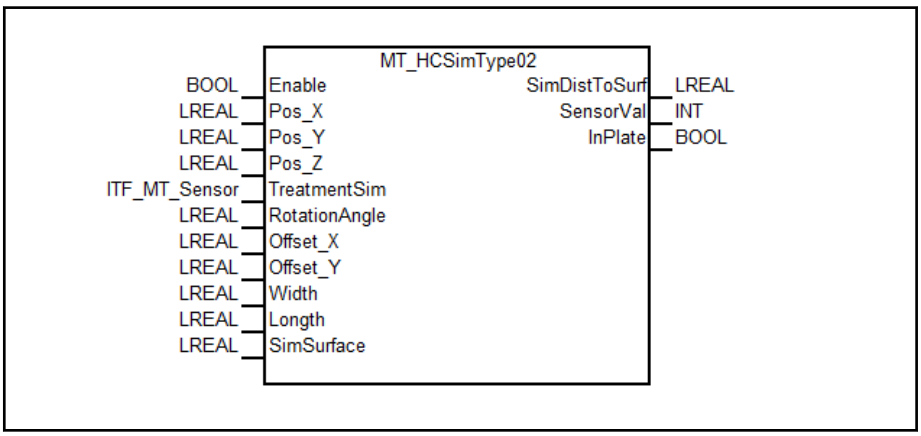


Fig. 16-5: MT_HCSimType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pos_X	LREAL	Current x-axis position
	Pos_Y	LREAL	Current y-axis position

I/O type	Name	Data type	Description
	Pos_Z	LREAL	Current z-axis position
	TreatmentSim	ITF_MT_Sensor	Function block for the sensor processing
	RotationAngle	LREAL	Simulated angle of rotation of the material
	Offset_X	LREAL	Simulated x-offset of the starting point position
	Offset_Y	LREAL	Simulated y-offset of the starting point position
	Width	LREAL	Current axis position (required for calibration)
	Longth	LREAL	
	SimSurface	LREAL	Function block for the sensor processing
VAR_OUTPUT	SimDistToSurf	LREAL	Distance between the z-position and the simulated material
	SensorVal	INT	Simulated sensor measured values
	InPlate	BOOL	

Tab. 16-19: Interface variables of the MT_HCSimType02 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	N. def.	N. def.	FALSE	Continuous
Pos_X	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
Pos_Y	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
Pos_Z	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
RotationAngle	LREAL	0	90	0.0	Continuous if "Enable" is TRUE
Offset_X	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
Offset_Y	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
Width	LREAL	0	N. def.	0.0	Continuous if "Enable" is TRUE
Longth	LREAL	0	N. def.	0.0	Continuous if "Enable" is TRUE
SimSurface	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Tab. 16-20: Overview on the min./max. values and default values of the inputs

Functional description

The function block MT_HCSimType02 simulates a rectangular material surface. The dimensions `Width`, `Length`, the z-position `SimSurface`, die starting point position`Offset_X`, `Offset_Y` and the angle of rotation of the material `RotationAngle` can be configured.

The vertical distance between the z-axis and the simulated material as well as the corresponding measures sensor value are calculated using the function block. The output `InPlate` indicates whether the z-axis is within the material area.

The function block is used together with the function blocks chapter 16.2.2 "MT_HeightRegulationType01" on page 117 and chapter 16.2.8 "MT_PlateSearchType02" on page 133 for the PlateSearch emulation. In case of a real control with distance sensor, disable the function block.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- Enable: Simulation status `stCtrl.bSimModeAct`
- Pos_X: Current position of the x-axis
- Pos_Y: Current position of the y-axis
- Pos_Z: Current position of the vertical coordinate

Normal operation

Each cycle, the MT_HCSimType02 computes whether the z-axis is within the material range and it computes the distance between the z-position and the simulated material for the "SimDistToSurf" output and the simulated sensor values for the "SensorVal" output.

16.2.7 MT_PlateSearchType01

Brief description

The MT_PlateSearchType01 function block determines the position offset of a rectangular material.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-21: Reference table of the MT_PlateSearchType01

Interface description

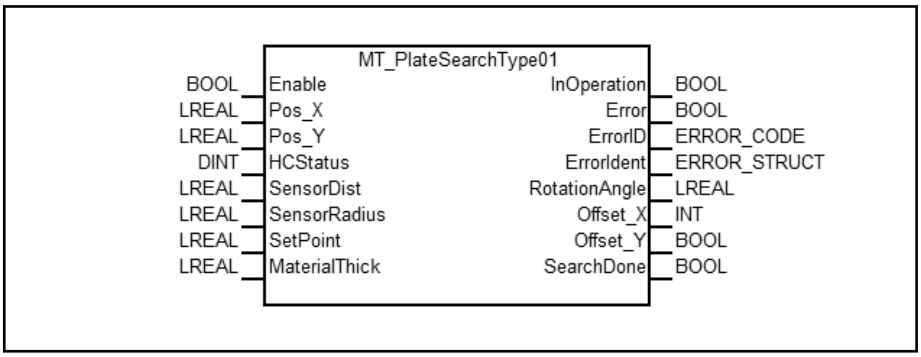


Fig. 16-6: MT_PlateSearchType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pos_X	LREAL	Current x-axis position
	Pos_Y	LREAL	Current y-axis position
	HCStatus	LREAL	Current status of the height regulation (is required to asses the material edge)
	SensorDist	LREAL	Distance between machining head and material
	SensorRadius	LREAL	Sensor radius

I/O type	Name	Data type	Description
	SetPoint	LREAL	Setpoint of the height control
	MaterialThick	LREAL	Material thickness
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Error codes of the MT_PlateSearchType01 function block, page 133
	RotationAngle	LREAL	Angle of rotation of the material
	Offset_X	LREAL	X-offset of the starting point position
	Offset_Y	LREAL	Y-offset of the starting point position
	SearchDone	BOOL	"`PlateSearch`" Process completed

Tab. 16-22: Interface variables of the MT_PlateSearchType01 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	N. def.	N. def.	FALSE	Continuous
Pos_X	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
Pos_Y	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
HCStatus	INT	0	N. def.	N. def.	Continuous if "Enable" is TRUE

Tab. 16-23: Overview on the min./max. values and default values of the inputs

Functional description

The MT_PlateSearchType01 function block determines the position offset `Offset_X`, `Offset_Y` and the orientation offset `RotationsAngle` of rectangular material.

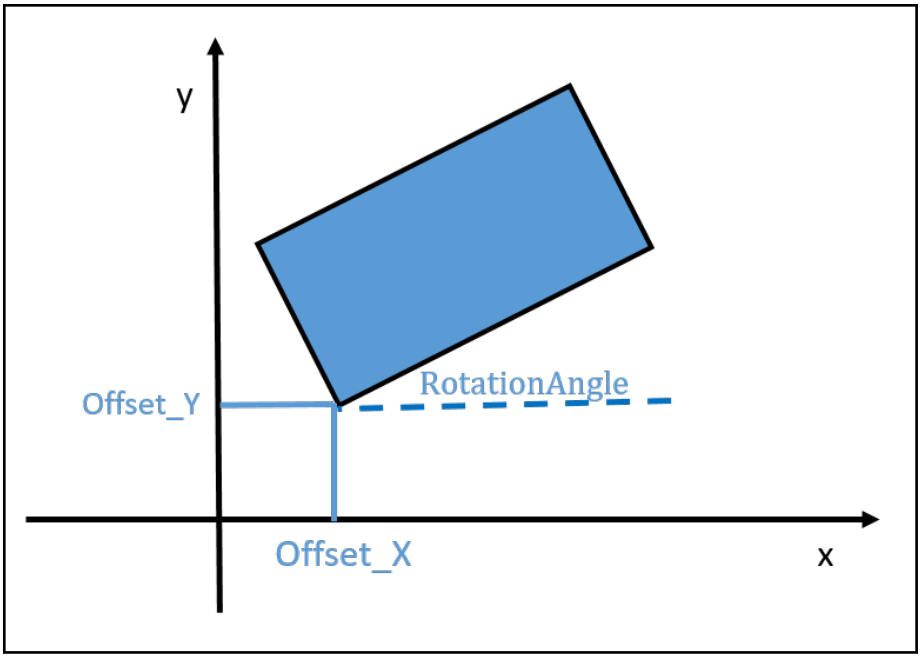


Fig. 16-7: Output signals of the MT_PlateSearchType01 function block

The function block is used together with the function block chapter 16.2.2 "MT_HeightRegulationType01" on page 117. Use the chapter 16.2.8 "MT_PlateSearchType02" on page 133 function block for the emulation.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- Pos_X: Current position of the x-axis
- Pos_Y: Current position of the y-axis
- HCStatus: Current height control status
- Offset_X: System date `SD.PlateSearch[*].OffsetX`
- Offset_Y: System date `SD.PlateSearch[*].OffsetY`
- RotationsAngle: System date `SD.PlateSearch[*].RotationsAngle`

Normal operation

MT_PlateSearchType01 computes the position offset and the orientation offset by calling the NC program "PlateSearchProg". The `UpdatePlate()` method can export the measuring results as system date.

UpdatePlate

The `UpdatePlate()` method can reset and export the measuring results as system date. For this method, the reset mode `UpdateMode = 0` and the export mode `UpdateMode = 1` is used. The inputs `Angle`, `Pos_X` and `Pos_Y` are the outputs of the `MT_PlateSearchType01` function block. The inputs are set to "0.0" in reset mode.

The `UpdatePlate()` method has to be called in a cyclic PLC task.

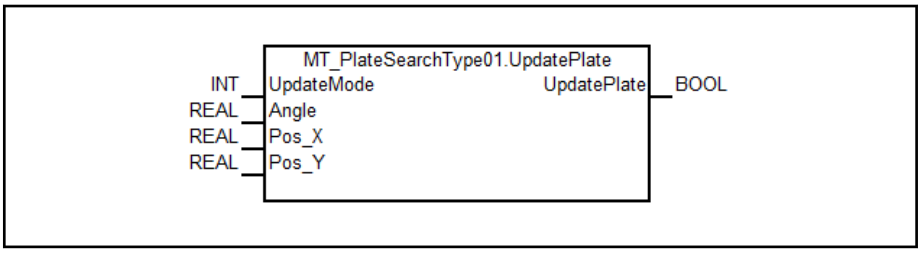


Fig. 16-8: Method: MT_PlateSearchType01.UpdatePlate

Configuration

The configuration contains all settings for the PlateSearch. The function block applies the following system variables when calling the `MT_DistanceSensorType01/02.UpdateConfig()` method on the function block instance:

Parameter	Unit	Description
`SD.SysHeightCtrl[*].Gapdetection.Inclination`	Degree	Inclination of the gap edge: acttan(z_2-z_1)/(xy_2-xy_1)
`SD.SysHeightCtrl[*].Gapdetection.Delay`	ms	Time after which a gap was detected before a retract motion is initiated
`SD.SysHeightCtrl[*].Gapdetection.DistUp`	mm	Retract motion if gap was detected. 0.0 in PlateSearch.

* NC channel, normally 1

Tab. 16-24: Settings of the MT_PlateSearchType01 function block

The `UpdateConfig()` method has to be called in a cyclic PLC task.

Troubleshooting

The function block generates the following error messages in Additional1/ Additional2 using the "**F_RELATED_TABLE**", 16#3120:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR16#0006)	16#3127	16#0000	No edge found
INPUT_RANGE_ERROR (16#0006)	16#3128	16#0000	No positioning can be calculated using the found edges
INPUT_RANGE_ERROR (16#0006)	16#3129	16#0000	Time out

Tab. 16-25: Error codes of the MT_PlateSearchType01 function block

16.2.8 MT_PlateSearchType02

Brief description

The MT_PlateSearchType02 function block determines the position offset of rectangular material. It is usually used in the emulation in connection with the chapter 16.2.6 "MT_HCSimType02" on page 128 function block.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-26: Reference table MT_PlateSearchType02

Interface description

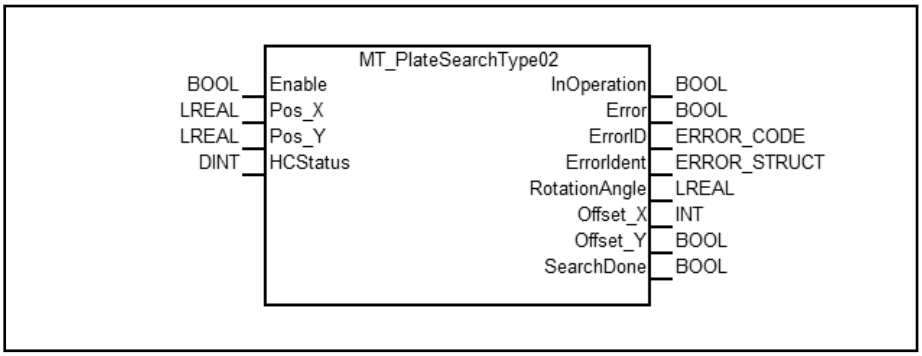


Fig. 16-9: MT_PlateSearchType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Pos_X	LREAL	Current x-axis position
	Pos_Y	LREAL	Current y-axis position
	HCStatus	LREAL	Current status of the height regulation (is required to asses the material edge)
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Error codes of the MT_PlateSearchType02 function block, page 136
	RotationAngle	LREAL	Angle of rotation of the material
	Offset_X	LREAL	X-offset of the starting point position
	Offset_Y	LREAL	Y-offset of the starting point position
	SearchDone	BOOL	"`PlateSearch`" Process completed

Tab. 16-27: Interface variables of the MT_PlateSearchType02 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	N. def.	N. def.	FALSE	Continuous
Pos_X	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
Pos_Y	LREAL	N. def.	N. def.	N. def.	Continuous if "Enable" is TRUE
HCStatus	INT	0	N. def.	N. def.	Continuous if "Enable" is TRUE

Tab. 16-28: Overview on the min./max. values and default values of the inputs

Functional description

The MT_PlateSearchType01 function block determines the position offset `Offset_X`, `Offset_Y` and the orientation offset `RotationsAngle` of rectangular material.

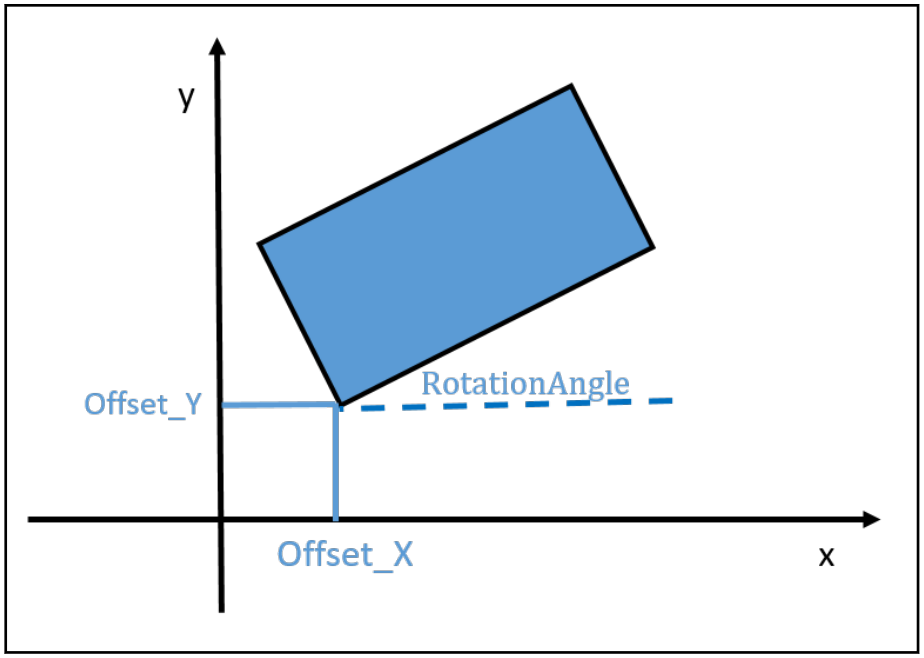


Fig. 16-10: Settings of the MT_PlateSearchType02 function block

The function block is used together with the function block chapter 16.2.2 "MT_HeightRegulationType01" on page 117. For the emulation, the workpiece can be simulated using the chapter 16.2.6 "MT_HCSimType02" on page 128 function block.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- Enable: Simulation status `stCtrl.bSimModeAct`
- Pos_X: Current position of the x-axis
- Pos_Y: Current position of the y-axis
- HCStatus: Current height control status
- Offset_X: System date `SD.PlateSearch[*].OffsetX`
- Offset_Y: System date `SD.PlateSearch[*].OffsetY`
- RotationsAngle: System date `SD.PlateSearch[*].RotationsAngle`

Normal operation

MT_PlateSearchType02 computes the position offset and the orientation offset by calling the NC program "PlateSearchProg". The `UpdatePlate()` method can export the measuring results as system date.

UpdatePlate

The `UpdatePlate()` method can reset and export the measuring results as system date. For this method, the reset mode `UpdateMode = 0` and the export mode `UpdateMode = 1` is used. The inputs `Angle`, `Pos_X` and `Pos_Y` are the outputs of the `MT_PlateSearchType02` function block. The inputs are set to "0.0" in reset mode.

The `UpdatePlate()` method has to be called in a cyclic PLC task.

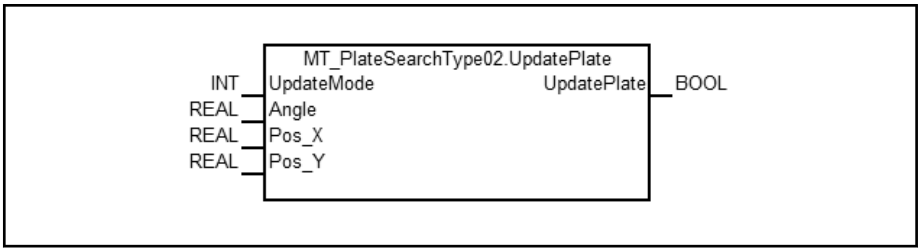


Fig. 16-11: Method: MT_PlateSearchType02.UpdatePlate

Configuration

The configuration contains all settings for the PlateSearch. The function block applies the following system variables when calling the `MT_DistanceSensorType01/02.UpdateConfig()` method on the function block instance:

Parameter	Unit	Description
`SD.SysHeightCtrl[*].Gapdetection.Inclination`	Degree	Inclination of the gap edge: acttan(z_2-z_1)/(xy_2-xy_1)
`SD.SysHeightCtrl[*].Gapdetection.Delay`	ms	Time after which a gap was detected before a retract motion is initiated
`SD.SysHeightCtrl[*].Gapdetection.DistUp`	mm	Retract motion if gap was detected. 0.0 in PlateSearch

* NC channel, normally 1

Tab. 16-29: Settings of the MT_PlateSearchType02 function block

The `UpdateConfig()` method has to be called in a cyclic PLC task.

Troubleshooting

The function block generates the following error messages in Additional1/ Additional2 using the "**F_RELATED_TABLE**", 16#3120:

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR16#0006)	16#3127	16#0000	No edge found
INPUT_RANGE_ERROR (16#0006)	16#3128	16#0000	No positioning can be calculated using the found edges.
INPUT_RANGE_ERROR (16#0006)	16#3129	16#0000	Time out

Tab. 16-30: Error codes of the MT_PlateSearchType02 function block

16.2.9 EN_HC_STATE_t

Brief description

The enumeration EN_HC_STATE_t shows the states of the height control and the sensor calibration.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-31: Reference table of the EN_HC_STATE_t

EN_HC_STATE_t

The following table lists the states of the height control and the sensor calibration.

Name	Values	Description
Status_NoError_C	0	No error
Status_HC_Calibratio-nActive_C	1	Sensor calibration is active
Status_HC_Calibration-CheckActive_C	2	Sensor calibration check is active
Status_HC_Calibration-WaitForInPos_C	3	Waiting for InPos during calibration
Status_HC_Frozen_C	4	Height control is in Freeze state, e.g. for PlateSearch
Status_HC_Calib-Check_OK_C	5	Sensor calibration done
Status_HC_SheetOutO-fRange_C	7	Material not in the measuring range
Status_HC_Contact_C	8	Collision
Status_HC_Retract_C	9	Lifting the machining head
Status_HC_Approach_C	10	Approach of the machining head
Status_HC_Regulat-ing_C	11	Regulation of the machining head
Status_HC_Gap_C	12	Gap is detected
Status_HC_Stable	13	Height control is stable
Status_SC2_Reset	20	Reset of the Precitec sensor
Status_SC2_CalibInit	21	Initializing the Precitec sensor
Status_SC2_Noz-zLost_OOR	22	Material is beyond the measured range of the Precitec sensor
Status_SC2_Noz-zLost_IR	23	Material is in the measured range of the Precitec sensor
Status_SC2_Approach	24	Approach of machining head in case of Precitec calibration
Status_SC2_RefCalib16	25	16 point calibration

Tab. 16-32: Overview of the status of height control and sensor calibration

Functional description

The enumeration EN_HC_STATE_t shows the states of the height control and the sensor calibration. It is the output of chapter 16.2.2 "MT_HeightRe-gulationType01" on page 117, chapter 16.2.3 "MT_DistanceSensorType01" on page 121 and chapter 16.2.4 "MT_DistanceSensorType02" on page 124.

16.2.10 EN_HC_USRMODE_t

Brief description

The enumeration EN_HC_USRMODE_t provides the modes of the height control.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-33: Reference table EN_HC_USRMODE_t

EN_HC_USRMODE_t

In the following table, the height control states and the sensor calibration are listed.

Name	Values	Description
EN_HC_HC_Off	0	Height control is disabled
EN_HC_Normal	1	Height control is active in normal mode
EN_HC_LFFreeze	2	Height control is frozen for the *LeapFrog* function. Machining head control is only possible upwards. The *HeightLimiter* input is the lower limit of the height
EN_HC_PSFreeze	3	Height control is frozen. Machining head control is only possible upwards
EN_HC_HoldPosition	4	Height control is frozen. The machining head remains at the current position

Tab. 16-34: Mode overview of the height control

Functional description

The enumeration EN_HC_USRMODE_t provides the modes of the height control. It is the output of chapter 16.2.2 "MT_HeightRegulationType01" on page 117, chapter 16.2.3 "MT_DistanceSensorType01" on page 121, and chapter 16.2.4 "MT_DistanceSensorType02" on page 124.

16.2.11 MT_RaiseAxis

Brief description

The MT_RaiseAxis function block raises the z-axis to a specified position. It is located in the "MT_Technology/Utility/GlobalFunctions" folder.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-35: Reference table MT_RaiseAxis

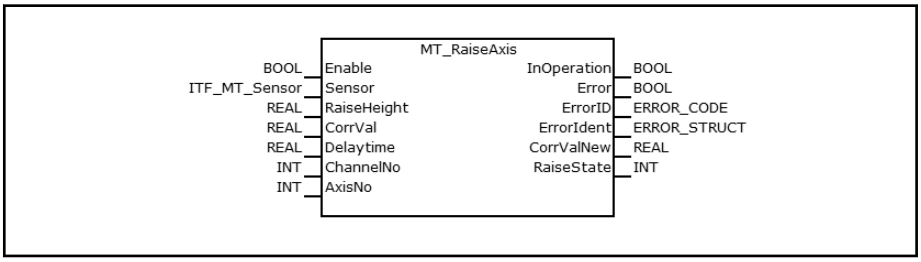


Fig. 16-12: Function block MT_RaiseAxis

Interface description

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Sensor	ITF_MT_Sensor	Function block for the sensor processing
	RaiseHeight	REAL	Target height of the axis (z-axis)
	CorrVal	REAL	Correction value for the z-coordinate after executing the function block, e.g.: output variable of the height control
	Delaytime	REAL	Delay troubleshooting after an Enable of the function block
	ChannelNo	INT	Channel number of the z-axis
	AxisNo	INT	Axis number of the z-axis
VAR_OUTPUT	InOperation	BOOL	Function block is operating

I/O type	Name	Data type	Description
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 139
	CorrValNew	REAL	Correction value for the z-coordinate after executing the function block
	RaiseState	HC_SensorState	Function block state: 0, no raising process active; 1, raising process active; 2, raising process completed

Tab. 16-36: Interface variables of the MT_RaiseAxis 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
Sensor	ITF_MT_Sensor	\	\	NULL	Once, at a rising edge at "Enable"
RaiseHeight	REAL	0.0	N. def.	0.0	Continuous if "Enable" is TRUE
CorrVal	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
Delaytime	REAL	0.0	N. def.	0.0	Continuous if "Enable" is TRUE
ChannelNo	INT	0	12	0	Continuous if "Enable" is TRUE
AxisNo	INT	0	256	0	Continuous if "Enable" is TRUE

Tab. 16-37: Overview on the min./max. values and default values of the inputs

Functional description

The MT_RaiseAxis function block raises the z-axis. The function block can be called in the FA program RetractControl_FA chapter 6.2.2 "Retract project module" on page 42.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- ChannelNo: Active channel number
- AxisNo: Axis number of the axis

Connect the "CorrValue" output of the MT_HeightRegulationType01 function block to the "CorrVal" input of this function block. The "CorrValNew" output controls the z-axis.

Normal operation

Each cycle, the MT_RaiseAxis function block computes a new value for the "CorrValNew" output.

Troubleshooting

The function block generates the following error messages in Additional1/ Additional2 using the "**F_RELATED_TABLE**", 16#3180:

ErrorID	Additional1	Additional2	Description
INPUT_INVALID_ERROR(16#0001)	16#312A	16#0000	Axis coupling is not active
INPUT_INVALID_ERROR(16#0001)	16#312B	16#0000	Configuration error

Tab. 16-38: Error codes MT_RaiseAxis function block

16.3 LeapFrog

16.3.1 Introduction and overview

In order to quickly place the cutting head on the material over long distances and avoid collisions with cut and possibly protruding parts, the cutting head has to be lifted during the positioning process and lowered at the target position. This process is called LeapFrog.

The vertical coordinate is controlled according to the movement over the material surface so that a fluent and fast movement is created and collisions are avoided. In doing so, the defined safety distances are maintained.

The function blocks described in this chapter provide the LeapFrog as an auxiliary function.

- MT_LeapFrogType01, page 140, performs the LeapFrog auxiliary function.

16.3.2 MT_LeapFrogType01

Brief description

The MT_LeapFrogType01 function block performs the LeapFrog auxiliary function. It is operated together with the function block MT_HeightRegulationType01, page 117.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-39: Reference table of the MT_LeapFrogType01

Interface description

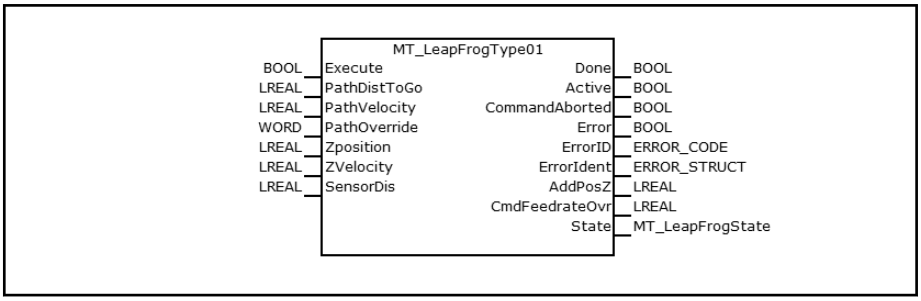


Fig. 16-13: MT_LeapFrogType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Execute	BOOL	Enables the function block at a rising edge
	PathDistToGo	LREAL	Distance up to block end in 10^{-4} mm
	PathVelocity	LREAL	Current path velocity in the plane in mm/s
	PathOverride	WORD	Feed for the x-axis and the y-axis in percent
	ZPosition	LREAL	Current position of the height axis in mm

I/O type	Name	Data type	Description
	ZVelocity	LREAL	Current velocity of the height axis in mm/s
	SensorDis	LREAL	Distance to the material from the sensor measurement in mm
VAR_OUTPUT	Done	BOOL	Completed the operation successfully
	Active	BOOL	The function block was enabled. The operation is executed.
	CommandAborted	BOOL	The operation was aborted
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 144
	AddPosZ	LREAL	Commanded position of the vertical axis in mm
	CmdFeedrateOvr	LREAL	FastAction path override
	State	MT_LeapFrogState	Execution state of the LeapFrog

Tab. 16-40: Interface variables of the MT_LeapFrogType01 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
PathDistToGo	LREAL	N. def.	N. def.	0.0	Continuous after a rising edge at "Execute"
PathVelocity	LREAL	N. def.	N. def.	0.0	Continuous after a rising edge at "Execute"
PathOverride	WORD	0	100	100	Continuous after a rising edge at "Execute"
ZPosition	LREAL	N. def.	N. def.	0.0	Continuous after a rising edge at "Execute"
ZVelocity	LREAL	N. def.	N. def.	0.0	Continuous after a rising edge at "Execute"
SensorDis	LREAL	N. def.	N. def.	0.0	Continuous after a rising edge at "Execute"

Tab. 16-41: Overview on the min./max. values and default values of the inputs

Functional description

The MT_LeapFrogType01 function block performs the LeapFrog auxiliary function. It is used for fast positioning on the material. The cutting head is lifted during positioning and set down again at the target position. Consider the protective distance determined in the system data.

The function block requires the current path velocity as well as the remaining distance in the block. They are created at the "PathDistToGo", "PathVelocity" and "ZVelocity" inputs. The computed control value is output at the "AddPosZ" output.

If the velocity on the plane becomes too high to lift the head in time, the function block can reduce the path velocity via the "CmdFeedrateOvr" output.

Parameterization

System data is used for the parameterization. The function block applies the following system variables when calling the `UpdateConfig()` method on the function block instance:

Parameters	Unit	Description
`SD.SysLeapFrog[*].Mode`		0: Fastest path possible 1: Inverted u-curve
`SD.SysLeapFrog[*].R_MaxHeight`	mm	Maximum height of the safe area
`SD.SysLeapFrog[*].R_Safe`		Safe height as part of `R_MaxHeight` The value has to be between 0.1 and 1.0.
`SD.SysLeapFrog[*].L_Safe`	mm	Distance between the starting point and reaching the height defined in `L_Safe` in mm
`SD.SysLeapFrog[*].L0`	mm	Lifting height at the starting point
`SD.SysLeapFrog[*].L1`	mm	Safe height at the target point
`SD.SysLeapFrog[*].MinDist`	mm	Defines a minimum LeapFrog distance from which the height at the target point is not L1, but L0. A value of 0 switches off the function.
`SD.SysLeapFrog[*].Feedrate`	mm/min	Maximum path velocity during the LeapFrog.
`SD.SysLeapFrog[*].CornerRadius`	mm	Corner radius of the inverted U-curve. It is only used in mode 1.

* Channel number

Tab. 16-42: Setting parameters for the MT_LeapFrogType01

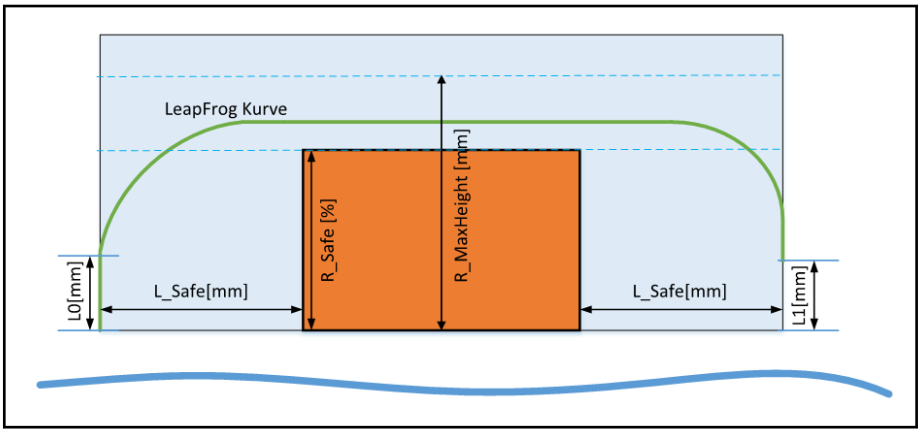


Fig. 16-14: Setting parameters for the MT_LeapFrogType01

The new configuration is applied via the "UpdateConfig()" method call of the function block. Call from a cyclic PLC task.

Trigger signal

A LeapFrog motion can be triggered or reset via an SD variable `SD.LeapFrog[*].Trigger`. Decode the variable first. The mDecodeCommand() method of the function block is used.

```
6   // start leapfrog
7   IF fbLF.mDecodeCommand(FAInputData.fLeapfrogSta) = Start THEN
8       bExecuteLF := TRUE;
9   END_IF
10
11  // reset leapfrog
12  IF fbLF.mDecodeCommand(FAInputData.fLeapfrogSta) = Reset THEN
13      bExecuteLF := FALSE;
14  END_IF
```

Fig. 16-15: Applying the DecodeTrigger() method

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

FB input	ID	Index 1	Index 2	Index 3	Access	Data type
PathDistToGo	100410	Channel number	-	-	0	LREAL
PathVelocity	100411	Channel number	-	-	0	UDINT
FB output	**ID**	**Index 1**	**Index 2**	**Index 3**	**Access**	**Data type**
CmdFeedrateOvr	100401	Channel number	-	-	1	REAL

Tab. 16-43: Link between the FA input data and the inputs of the function block

FB output	ID	Index 1	Index 2	Index 3	Access	Data type
CmdFeedrateOvr	100401	Channel number	-	-	1	REAL

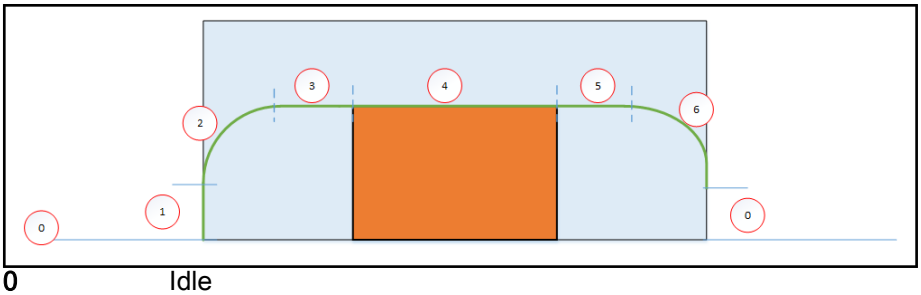
Tab. 16-44: Link between the FA output data and the outputs of the function block

Normal operation

After a rising edge at "Execute", the MT_LeapFrogType01 function block computes a new value at the "AddPosZ", "AddVeCmdFeedrateOvrlZ" outputs upon each cycle.

The "Active" output is TRUE while executing the LeapFrog. The current status of the process can be read at the "State" output. It can assume the following states:

- Idle: LeapFrog inactive
- UpwardBeforeL0: Upward movement before reaching L0
- UpwardAfterL0: Upward movement after reaching L0
- TopBeforeSafeZone: Above the safe height but not in the safe zone
- TopSafeZone: Above the safe zone
- TopAfterSafeZone: Above the safe height, behind the safe zone
- Downward: Lowering motion



0 Idle
1 UpwardBeforeL0
2 UpwardAfterL0
3 TopBeforeSafeZone
4 TopSafeZone
5 TopAfterSafeZone
6 Downward

Fig. 16-16: States of the MT_LeapFrogType01 function block

Troubleshooting 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 function block state
ACCESS_ERROR (16#0004)	16#3111	16#0000	Timeout while reading the system variables
INPUT_RANGE_ERROR (16#0006)	16#3112	16#0000	System variable .../Height invalid, less than 0 or less than L0/L1
INPUT_RANGE_ERROR (16#0006)	16#3113	16#0000	System variable .../L0 invalid or less than 0
INPUT_RANGE_ERROR (16#0006)	16#3114	16#0000	System variable .../L1 invalid or less than 0
INPUT_RANGE_ERROR (16#0006)	16#3115	16#0000	System variable ...L_SAFE is lower than 0.1
CALCULATION_ERROR (16#0007)	16#3116	16#0000	Internal error
CALCULATION_ERROR (16#0007)	16#3117	16#0000	Internal error
INPUT_RANGE_ERROR (16#0006)	16#3118	16#0000	System variable .../MinDist invalid or lower than 0.1 mm
INPUT_RANGE_ERROR (16#0006)	16#3119	16#0000	System variable .../Feedrate invalid or lower than 0.001 mm/min
INPUT_RANGE_ERROR (16#0006)	16#311A	16#0000	System variable .../R_Safe invalid, smaller than 0.1 or higher than 1.0
INPUT_RANGE_ERROR (16#0006)	16#311B	16#0000	System variable .../Mode invalid. Should be 0 or 1.
INPUT_RANGE_ERROR (16#0006)	16#311C	16#0000	System variable .../L1 invalid, smaller than 0.0 or higher than (R_MaxHeight x R_Safe x 0.5) .

Tab. 16-45: Error codes MT_LeapFrogType01 function block

16.4 Power control

16.4.1 Introduction and overview

The cutting power is controlled by various parameters depending on the machine type. In general, the power to be applied depends on the cutting velocity and the material used.

The function blocks described in this chapter control the cutting power or provide the required auxiliary functions.

- MT_GetPowerCtrlMode, page 145, decodes settings and status information for the power control.
- MT_LaserPowerControlType01, page 146, controls the power of the cutting laser via an analog signal.
- MT_LaserPowerControlType02, page 149, controls the power of the cutting laser via a PWM signal.
- MT_LaserPowerControlType03, page 151, controls the power of the cutting laser via a PWM signal at an S20-DIDO-2/1-OVS S20 bus module.

16.4.2 MT_GetPowerCtrlMode

Brief description

The MT_GetPowerCtrlMode function block is an auxiliary function block. It decodes the control word for the cutting jet and translate it into four control signals.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-46: Reference table of the MT_GetPowerCtrlMode

Interface description

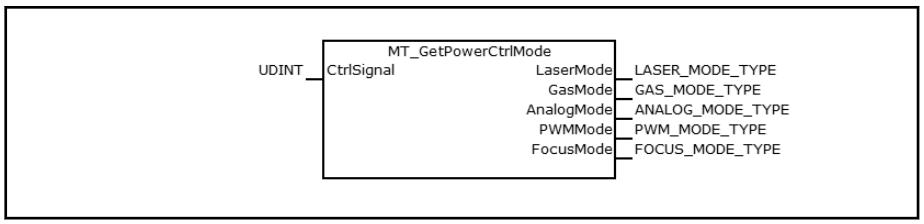


Fig. 16-17: MT_GetPowerCtrlMode function block

I/O type	Name	Data type	Description
VAR_INPUT	CtrlSignal	UDINT	Control signal from the NC
VAR_OUTPUT	LaserMode	LASER_MODE_TYPE	Control signal for the laser power control
	GasMode	GAS_MODE_TYPE	Control signal for the protective gas
	AnalogMode	ANALOG_MODE_TYPE	Control signal for the analog mode
	PWMMode	PWM_MODE_TYPE	Control signal for the PWM mode
	FocusMode	FOCUS_MODE_TYPE	Control signal for the laser focus

Tab. 16-47: Interface variables of the MT_GetPowerCtrlMode 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
CtrlSignal	UDINT	0	65535	0	Continuous

Tab. 16-48: Overview on the min./max. values and default values of the inputs

Functional description

The MT_GetPowerCtrlMode function block is an auxiliary function block. It decodes the control word for the cutting jet and translate it into four control signals.

It requires an input signal from the NC. Various modes for the laser power control are coded in this signal. The control signals for the following function blocks are created from it:

- MT_LaserPowerControlType01
- MT_LaserPowerControlType02
- MT_LaserPowerControlType03

CtrlSignal

This signal is transferred from the NC within a Fast Action task. It is binary coded and should not be modified.

LaserMode, GasMode, AnalogMode, PWMMode

These outputs are extracted from the "CtrlSignal". Each output is mapped on an ENUM and therefore easily legible.

Circuitry

Call the function block in a fast Fast Action task. Link the following inputs and outputs with the respective values from the NC:

FB input	ID	Index 1	Index 2	Index 3	Access	Data type
CtrlSignal	100006	-	-	/SysPowerCtrl[*]/SignalStatus	0	UDINT

* NC channel

Tab. 16-49: Link between the FA input data and the inputs of the function block

Normal operation

The MT_FlyCutType01 function block calculates a new value for the "LaserMode", "GasMode", "AnalogMode", "PWMMode" and "FocusMode" outputs with each cycle.

Troubleshooting

The function block does not generate error codes.

16.4.3 MT_LaserPowerControlType01

Brief description

The MT_LaserPowerControlType01 function block controls the laser power via an analog output by means of the path velocity.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-50: Reference table of the MT_LaserPowerControlType01

Interface description

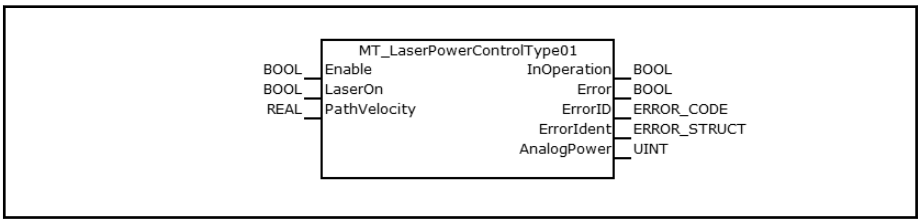


Fig. 16-18: MT_LaserPowerControlType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	LaserOn	BOOL	Laser state FALSE: Switch off laser TRUE: Switch on laser
	PathVelocity	REAL	Path velocity from the NC
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 148
	AnalogPower	UINT	Setpoint for the laser power

Tab. 16-51: Interface variables of the MT_LaserPowerControlType01 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
LaserOn	BOOL			FALSE	Continuous if "Enable" is TRUE
PathVelocity	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Tab. 16-52: Overview on the min./max. values and default values of the inputs

Functional description

The MT_LaserPowerControlType01 function block controls the laser power via an analog output by means of the path velocity.

It requires the current path velocity of the cutting head. The path velocity is created on the "PathVelocity" input. The calculated control value is issued at the "AnalogPower" output.

Circuitry

Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- PathVelocity: SD.SysIpoPathFeed[*].ActFeed

Normal operation

The MT_LaserPowerControlType01 function block calculate a new value for the "AnalogPower" output with each cycle.

Configuration

The configuration contains all setting parameters for the analog power control. The function block applies the following system variables when calling the `UpdateConfig()` method on the function block instance:

Parameters	Unit	Description
SD.SysPowerCtrl[*].Analog.AnalogMode		0: No analog signal is used 1: Fix power value 2: Linear velocity/power control 3: Velocity/power curve 4: Time/power curve
SD.SysPowerCtrl[*].Analog.NormalPower	mV	specifies the fixed power. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedLinear.Lowest-Power	mV	sets the voltage for velocities below the minimum velocity. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedLinear.Lower-Point.Velocity	mm/min	sets the minimum velocity for the linear conversion.
SD.SysPowerCtrl[*].Analog.SpeedLinear.Lower-Point.Power	mV	specifies the voltage at minimum velocity. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedLinear.Upper-Point.Velocity	mm/min	sets the maximum velocity for the linear conversion.
SD.SysPowerCtrl[*].Analog.SpeedLinear.Upper-Point.Power	mV	specifies the voltage at maximum velocity. Value range: 0 .. 10000 mV
SD.SysPowerCtrl[*].Analog.SpeedCurve.FilePath		Defines the storage location of the curve file as string.
SD.SysPowerCtrl[*].Analog.TimeCurve.FilePath		Defines the storage location of the curve file as string.
SD.SysPowerCtrl[*].Analog.TimeCurve.Cycles		specifies the number of repetitions of the time curve.

* NC channel, normally 1

Tab. 16-53: Settings of the MT_LaserPowerControlType01 function block

The `UpdateConfig()` method has to be called in a cyclic PLC task.

LaserOn If "LaserOn" is set, the laser is activated. Depending on the configuration, the laser power is controlled according to the path velocity.

Troubleshooting 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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3140	16#0000	"AnalogMode" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3141	16#0000	"PathVelocity" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3142	16#0000	Parameter "ChannelNumber" invalid
OTHER_ERROR (16#000F)	16#3143	16#0000	Timeout while reading the parameter
INPUT_RANGE_ERROR (16#0006)	16#3144	16#0000	Parameter "NormalPower" invalid
INPUT_RANGE_ERROR (16#0006)	16#3145	16#0000	Error in the parameterization of the power line

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR (16#0006)	16#3146	16#0000	Velocity/power curve do not rise monotonously
INPUT_RANGE_ERROR (16#0006)	16#3147	16#0000	Points of the velocity/power curve out of range
INPUT_RANGE_ERROR (16#0006)	16#3148	16#0000	Time/power curve does not rise continuously
INPUT_RANGE_ERROR (16#0006)	16#3149	16#0000	Points of the time/power curve out of range

Tab. 16-54: Error codes MT_LaserPowerControlType01 function block

16.4.4 MT_LaserPowerControlType02

Brief description

The MT_LaserPowerControlType02 function block controls the laser power via a PWM output by means of the path velocity.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-55: Reference table of the MT_HeightRegulationType02

Interface description

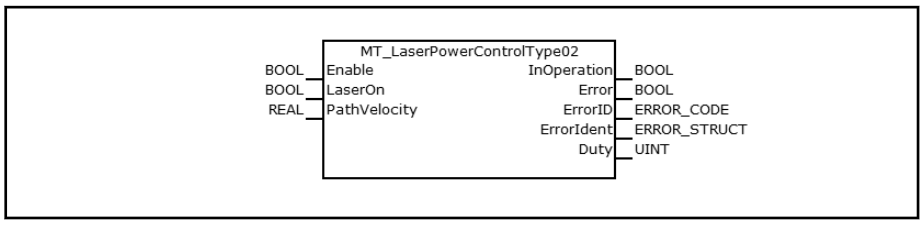


Fig. 16-19: MT_LaserPowerControlType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	LaserOn	BOOL	Laser state FALSE: Switch off laser TRUE: Switch on laser
	PathVelocity	REAL	Path velocity from the NC
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 151
	Duty	UINT	Setpoint for the laser power

Tab. 16-56: Interface variables of the MT_LaserPowerControlType02 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
LaserOn	BOOL			FALSE	Continuous if "Enable" is TRUE
PathVelocity	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Tab. 16-57: Overview on the min./max. values and default values of the inputs

Functional description The MT_LaserPowerControlType01 function block controls the laser power via a PWM output by means of the path velocity.

It requires the current path velocity of the cutting head. The path velocity is created on the "PathVelocity" input. The calculated control value is issued at the "Duty" output.

Circuitry Call the function block in a Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- PathVelocity: SD.SysIpoPathFeed[*].ActFeed

Normal operation The MT_LaserPowerControlType02 function block calculate a new value for the "Duty" output with each cycle.

Configuration The configuration contains all setting parameters for the PWM-based power control. The function block applies the following system variables when calling the `UpdateConfig()` method on the function block instance:

Parameter	Unit	Description
`SD.SysPowerCtrl[*].PWM.Mode`		0: No PWM signal is used 1: Fix duty value 2: Linear velocity/duty control 3: Velocity/duty curve
`SD.SysPowerCtrl[*].Duty.NormalPower`	%	sets the fixed duty. Value range: 0 .. 100 %
`SD.SysPowerCtrl[*].Duty.SpeedLinear.Lowest-Power`	%	sets the duty for velocities below the minimum velocity. Value range: 0 .. 100 %
`SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Velocity`	mm/min	sets the minimum velocity for the linear conversion in mm/min.
`SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Power`	%	sets the duty at minimum velocity. Value range: 0 .. 100 %
`SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Velocity`	mm/min	sets the maximum velocity for the linear conversion in mm/min.
`SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Power`	%	sets the duty at maximum velocity. Value range: 0 .. 100 %
`SD.SysPowerCtrl[*].Duty.SpeedCurve.FilePath`		Defines the storage location of the curve file as string.

* NC channel, normally 1

Tab. 16-58: Settings of the MT_LaserPowerControlType02 function block

The `UpdateConfig()` method has to be called in a cyclic PLC task.

LaserOn If "LaserOn" is set, the laser is activated. Depending on the configuration, the laser power is controlled according to the path velocity.

Troubleshooting 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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3140	16#0000	"PWMMode" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3141	16#0000	"PathVelocity" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3142	16#0000	Parameter "ChannelNumber" invalid
OTHER_ERROR (16#000F)	16#3143	16#0000	Timeout while reading the parameter
INPUT_RANGE_ERROR (16#0006)	16#3144	16#0000	Parameter "NormalPower" invalid
INPUT_RANGE_ERROR (16#0006)	16#3145	16#0000	Error in the parameterization of the power line
INPUT_RANGE_ERROR (16#0006)	16#3146	16#0000	Velocity/power curve do not rise monotonously
INPUT_RANGE_ERROR (16#0006)	16#3147	16#0000	Points of the velocity/power curve out of range

Tab. 16-59: Error codes MT_LaserPowerControlType02 function block

16.4.5 MT_LaserPowerControlType03

Brief description The MT_LaserPowerControlType03 function block controls the laser power via a PWM at a S20-DIDO-2/1-OVS S20 bus module by means of the path velocity.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-60: Reference table of the MT_HeightRegulationType03

Interface description

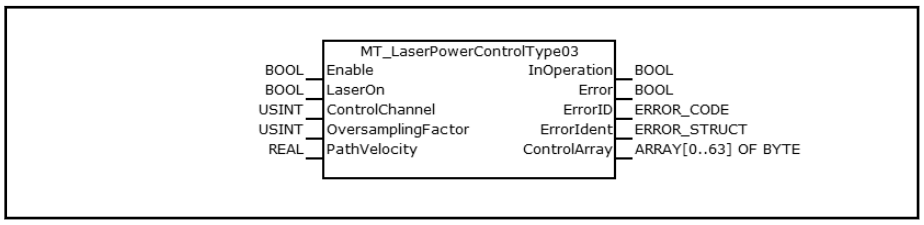


Fig. 16-20: MT_LaserPowerControlType03 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	LaserOn	BOOL	Laser state FALSE: Switch off laser TRUE: Switch on laser
	ControlChannel	USINT	Oversampling channel 1 or 2 to be used
	OversamplingFactor	USINT	Oversampling factor set at S20-DIDO-2/1-OVS

I/O type	Name	Data type	Description
	PathVelocity	REAL	Path velocity from the NC
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 153
	ControlArray	ARRAY[0..63] OF BYTE	Cyclic channel for the S20-DIDO-2/1-OVS module

Tab. 16-61: Interface variables of the MT_LaserPowerControlType03 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
LaserOn	BOOL			FALSE	Continuous if "Enable" is TRUE
ControlChannel	USINT	1	2	0	Once, at a rising edge at "Enable"
Oversampling-Factor	USINT	0	512	0	Once, at a rising edge at "Enable"
PathVelocity	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Tab. 16-62: Overview on the min./max. values and default values of the inputs

Functional description

The MT_LaserPowerControlType03 function block controls the laser power via a PWM at a S20-DIDO-2/1-OVS S20 bus module by means of the path velocity.

It requires the current path velocity of the cutting head. The path velocity is created on the "PathVelocity" input. The calculated control value is issued at the "ControlArray" output.

Circuitry

Call the function block in a fast Fast Action task. Link the following inputs and outputs with the respective values from the NC:

- PathVelocity: SD.SysIpoPathFeed[*].ActFeed

Normal operation

The MT_LaserPowerControlType02 function block calculate a new value for the "ControlArray" output with each cycle. The bit pattern in the array of the "ControlArray" output corresponds to a PWM modulation.

Configuration

The configuration contains all setting parameters for the PWM-based power control. The function block adopts the following system variables when calling the `UpdateConfig()` method on the function block instance:

Parameter	Unit	Description
SD.SysPowerCtrl[*].PWM.Mode		0: No PWM signal is used 1: Fix duty value 2: Linear velocity/duty control 3: Velocity/duty curve
SD.SysPowerCtrl[*].Duty.SpeedLinear.Lowest-Power	%	sets the duty for velocities below the minimum velocity. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Velocity	mm/min	sets the minimum velocity for the linear conversion in mm/min.
SD.SysPowerCtrl[*].Duty.SpeedLinear.Lower-Point.Power	%	sets the duty at minimum velocity. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Velocity	mm/min	sets the maximum velocity for the linear conversion in mm/min.
SD.SysPowerCtrl[*].Duty.SpeedLinear.Upper-Point.Power	%	sets the duty at maximum velocity. Value range: 0 .. 100 %
SD.SysPowerCtrl[*].Duty.SpeedCurve.FilePath		Defines the storage location of the curve file as string.
SD.SysPowerCtrl[*].OVS.Frequency	Hz	Frequency for the PWM

* NC channel, normally 1

Tab. 16-63: Settings of the MT_LaserPowerControlType03 function block

Calling the `UpdateConfig()` method has the be carried out in a cyclic PLC task.

LaserOn

If "LaserOn" is set, the laser is activated. Depending on the configuration, the laser power is controlled according to the path velocity.

ControlChannel, Oversampling-Factor

The "ControlChannel" input corresponds to the channel of the S20-DIDO-2/1-OVS S20 bus module to be used. The value of the OversamplingFactor input has to correspond to the selected setting at the S20-DIDO-2/1-OVS S20 bus module. For more information, see chapter chapter 16.8.1 "Introduction and overview" on page 165.

ControlArray

The ControlArray output contains the generated PWM signal. It can either be directly mapped on a cyclic channel of a S20-DIDO-2/1-OVS S20 bus module or be further processed by means of the chapter 16.8.3 "MT_Oversampling_OR" on page 167 and chapter 16.8.2 "MT_Oversampling_AND" on page 165 function blocks.

Troubleshooting

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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3140	16#0000	"PWMMode" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3141	16#0000	"PathVelocity" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3142	16#0000	Parameter "ChannelNumber" invalid
OTHER_ERROR (16#FFFE)	16#3143	16#0000	Timeout while reading the parameter

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR (16#0006)	16#3144	16#0000	Parameter "NormalPower" invalid
INPUT_RANGE_ERROR (16#0006)	16#3145	16#0000	Error in the parameterization of the power line
INPUT_RANGE_ERROR (16#0006)	16#3146	16#0000	Velocity/power curve do not rise monotonously
INPUT_RANGE_ERROR (16#0006)	16#3147	16#0000	Points of the velocity/power curve out of range
INPUT_RANGE_ERROR (16#0006)	16#314A	16#0000	Calculated switching frequency too high
INPUT_RANGE_ERROR (16#0006)	16#314B	16#0000	"ControlChannel" input not 1 or 2
	16#314C	16#0000	"OversamplingFactor" input too big

Tab. 16-64: Error codes MT_LaserPowerControlType03 function block

16.5 Gas control

16.5.1 MT_GasControlType01

Brief description

The function block MT_GasControlType01 switches and monitors the valves for air and protective gas of the cutting head.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-65: Reference table of the MT_GasControlType01

Interface description

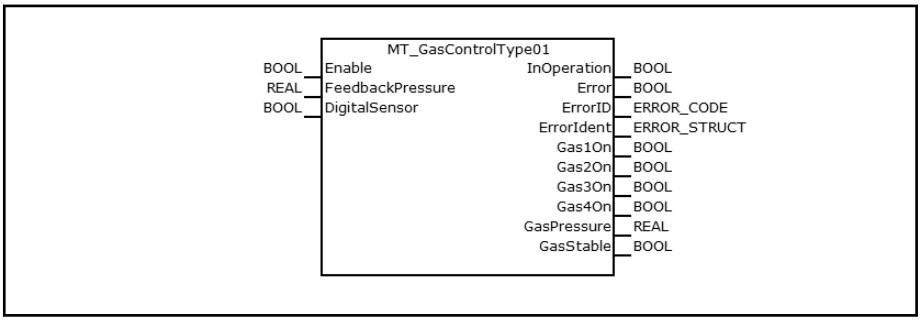


Fig. 16-21: MT_GasControlType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	FeedbackPressure	REAL	Measured gas pressure at the cutting head
	DigitalSensor	BOOL	Feedback of a digital pressure sensor
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description

I/O type	Name	Data type	Description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 156
	Gas1On	BOOL	Gas type 1 on/off
	Gas2On	BOOL	Gas type 2 on/off
	Gas3On	BOOL	Gas type 3 on/off
	Gas4On	BOOL	Gas type 4 on/off
	GasPressure	REAL	Measured gas pressure
	GasStable	BOOL	Gas pressure stable

Tab. 16-66: Interface variables of the MT_GasControlType01 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
FeedbackPres-sure	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
DigitalSensor	BOOL			FALSE	Continuous if "Enable" is TRUE

Tab. 16-67: Overview on the min./max. values and default values of the inputs

Functional description

The function block MT_GasControlType01 switches and monitors the valves for air and protective gas of the cutting head.

Circuitry

Call the function block in a Fast Action task. Link the following inputs to the respective values from the sensors:

- "FeedbackPressure" for an analog sensor
- "DigitalSensor" for a digital pressure sensor

Link the outputs "Gas1On" to "Gas4On" to the actuation of the respective valves.

Normal operation

Each cycle, the MT_GasControlType01 computes new values for all outputs.

Configuration

The configuration contains all settings for the gas control. The function block applies the following system variables when calling the `UpdateConfig()` method on the function block instance:

Parameters	Unit	Description
`SD.SysGasCtrl[*].SensorType`		Pressure sensor type 0: Without sensor 1: Analog sensor 2: Digital sensor (pressure reached)
`SD.SysGasCtrl[*].GasType`		Selected gas type 1 .. 4
`SD.SysGasCtrl[*].GasPressure`	bar	Gas pressure for analog sensor
`SD.SysGasCtrl[*].GasTime`	ms	Minimum waiting time to reach the gas pressure
`SD.SysGasCtrl[*].WaitTimeout`	ms	Maximum waiting time to reach the gas pressure

Parameters	Unit	Description
SD.SysGasCtrl[*].UnstableTimeout	ms	Maximum time till pressure stabilization
SD.SysGasCtrl[*].StableWindow	bar	Window for stable pressure

* NC channel, normally 1

Tab. 16-68: Settings of the MT_GasControlType01 function block

The `UpdateConfig()` method has to be called in a cyclic PLC task.

Troubleshooting 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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3150	16#0000	Cannel number invalid
INPUT_RANGE_ERROR (16#0006)	16#3151	16#0000	Timeout when reading SD variables
INPUT_RANGE_ERROR (16#0006)	16#3152	16#0000	An SD variable value is invalid
INPUT_RANGE_ERROR (16#0006)	16#3153	16#0000	“FeedbackPressure” input invalid
OTHER_ERROR (16#7FFE)	16#3154	16#0000	Timeout when reaching the stable gas pressure
OTHER_ERROR (16#7FFE)	16#3155	16#0000	Gas pressure is not in the stable area anymore

Tab. 16-69: Error codes of the MT_GasControlType01 function block

16.6 Focus control

16.6.1 MT_FocusControlType01

Brief description The function block MT_FocusControlType01 controls and monitor the focus of the laser cutting head.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-70: Reference table of the MT_FocusControlType01

Interface description

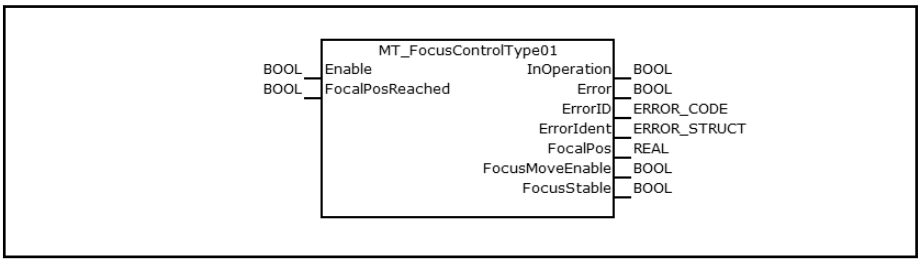


Fig. 16-22: MT_FocusControlType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	FocalPosReached	BOOL	Cutting head reached focus position (PRECITEC laser head POS.REACHED)

I/O type	Name	Data type	Description
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Troubleshooting, page 158
	FocalPos	REAL	Analog output signal on the laser head
	FocusMoveEnable	BOOL	Focus motion permitted (PRECITEC laser head AUTOMATIC)
	FocusStable	BOOL	Focus has reached the target position

Tab. 16-71: Interface variables of the MT_FocusControlType01 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
FocalPosReached	BOOL			FALSE	Continuous if "Enable" is TRUE

Tab. 16-72: Overview on the min./max. values and default values of the inputs

Functional description

The function block MT_FocusControlType01 controls and monitor the focus of the laser cutting head.

Circuitry

Call the function block in a Fast Action task. Link the following inputs to the respective values from the sensors:

- "FocalPosReached" output of the PRECITEC head

The output "FocusMoveEnable" is connected to the AUTOMATIC input of the PRECITEC head.

Normal operation

Each cycle, the MT_FocusControlType01 computes new values for all outputs.

Configuration

The configuration contains all settings for focus control. The function block applies the following system variables when calling the `UpdateConfig()` method on the function block instance:

Parameters	Unit	Description
`SD.SysFocusCtrl[*].SensorType`		Type of focus monitoring 0: Open Loop 1: Closed-loop - Waiting for "FocalPosReached" input
`SD.SysFocusCtrl[*].FocalPos`	mm	Target focus position
`SD.SysFocusCtrl[*].WaitTime`	ms	Waiting time for focus (only type 0)
`SD.SysFocusCtrl[*].InPosTimeout`	ms	Minimum waiting time to reach the focus

* NC channel, normally 1

Tab. 16-73: Settings of the MT_FocusControlType01 function block

The `UpdateConfig()` method has to be called in a cyclic PLC task.

Troubleshooting

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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3160	16#0000	Cannel number invalid
INPUT_RANGE_ERROR (16#0006)	16#3161	16#0000	Timeout when reading SD variables
INPUT_RANGE_ERROR (16#0006)	16#3162	16#0000	An SD variable value is invalid
INPUT_RANGE_ERROR (16#0006)	16#3163	16#0000	Wrong initial value at the "Focal-PosReached" input
OTHER_ERROR (16#7FFE)	16#3164	16#0000	Timeout when reaching the focus

Tab. 16-74: Error codes of the MT_FocusControlType01 function block

16.7 FlyCut

16.7.1 Introduction and overview

The velocity of the cutting head in shape cutting machines often has to be reduced in order to follow the shape of the workpiece. With simple, repeating geometrical shapes such as grids, it is easier to drive in straight lines and to activate or deactivate the cutting jet corresponding to the position. For this purpose, the switching time has to be calculated accurate to position.

The function blocks described in this chapter cover this application case.

- MT_FlyCutType01, page 158, activates or deactivates the cutting jet with the precision of the Sercos cycle.
- MT_FlyCutType02, page 161, activates or deactivates the cutting jet with a precision of up to 10 µs. The S20-DIDO-2-CSOVS I/O module is required for this purpose.

16.7.2 MT_FlyCutType01

Brief description

The MT_FlyCutType01 function block controls the activation and deactivation of a cutting head with regard to the position. It is used for implementing the FlyCut functionality.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-75: Reference table of the MT_FlyCutType01

Interface description

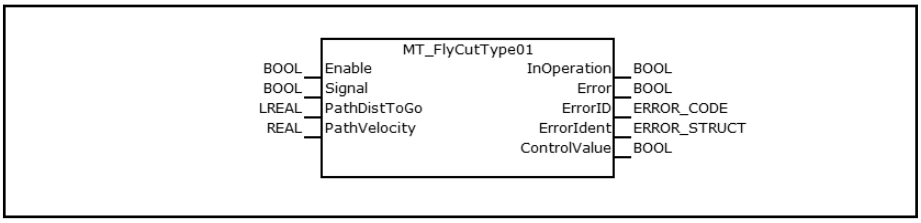


Fig. 16-23: MT_FlyCutType01 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Signal	BOOL	Input signal of the G code
	PathDistToGo	LREAL	Distance up to block end
	PathVelocity	REAL	Current path velocity
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Error handling, page 160
	ControlValue	BOOL	Signal for controlling the cutting head

Tab. 16-76: Interface variables of the MT_FlyCutType01 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
Signal	BOOL			FALSE	Continuous if "Enable" is TRUE
PathDistToGo	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
PathVelocity	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE

Tab. 16-77: Overview on the min./max. values and default values of the inputs

Functional description

The MT_FlyCutType01 function block controls the activation and deactivation of a cutting head with regard to the position. It is used for implementing the FlyCut functionality.

For this purpose, it requires some input data from the NC. It has to be transferred from the input data of the Fast Action task. Furthermore, a control signal that is activated in the NC block is required. This way, the function block knows that a switching operation has to be done at the block end. Delays due to the Fast Action task and other internal processes have already been calculated internally. The output signal can be directly issued to a high-speed output such as the local outputs of the control.

Signal, PathDistToGo, PathVelocity

These signals are transferred from the NC within a Fast Action task. "PathDistToGo" corresponds to the "PathDistToGo" command of the FA input data. "PathVelocity" is obtained from the "PathVelo" command of the FA input data.

The "Signal" input can be obtained from several sources. It can be generated via an event (`SEV()`, `REV()`). Both methods lead to different images in the FA input data.

Circuitry Call the function block in a fast Fast Action task. Link the following inputs and outputs with the respective values from the NC:

FB input	ID	Index 1	Index 2	Index 3	Access	Data type
Signal	100500	Event number (1..96)	-	-	0	BOOL
PathDistToGo	100410	Channel number	-	-	0	LREAL
PathVelocity	100411	Channel number	-	-	0	UDINT

Tab. 16-78: Link between the FA input data and the inputs of the function block

Normal operation MT_FlyCutType01 calculates a new value for the "ControlValue" output with each cycle.

Configuration The configuration contains the temporal behavior as well as the switch-on and switch-off positions with regard to the end of the NC block. The function block adopts the following system variables when calling the `UpdateConfig()` method on the function block instance:

System variable	Description
`SD.SysFlyCut[*].DistOn`	Indicates the switch-on distance in mm relative to the end point of the NC block
`SD.SysFlyCut[*].DistOff`	Indicates the switch-off distance in mm relative to the end point of the NC block
`SD.SysFlyCut[*].TimeOn`	Indicates the switch-on point in time in ms relative to the end point of the NC block
`SD.SysFlyCut[*].TimeOff`	Indicates the switch-off point in time in ms relative to the end point of the NC block

***** NC channel, normally 1

Tab. 16-79: Settings of the MT_FlyCutType01 function block

Calling the `UpdateConfig()` method has the be carried out in a cyclic PLC task.

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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3132	16#0000	Parameter "TimeOn" > T#1S
INPUT_RANGE_ERROR (16#0006)	16#3133	16#0000	Parameter "TimeOff" > T#1S
INPUT_RANGE_ERROR (16#0006)	16#3136	16#0000	Parameter "DistOn" < 0.0 mm
INPUT_RANGE_ERROR (16#0006)	16#3137	16#0000	Parameter "DistOf" < 0.0 mm

Tab. 16-80: Error codes MT_FlyCutType01 function block

16.7.3 MT_FlyCutType02

Brief description

The MT_FlyCutType02 function block controls the activation and deactivation of a cutting head with regard to the position. It is used for implementing the FlyCut functionality.

This function block requires the S20-DIDO-2/1-OVS S20 bus module for issuing the signal.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-81: Reference table of the MT_FlyCutType02

Interface description

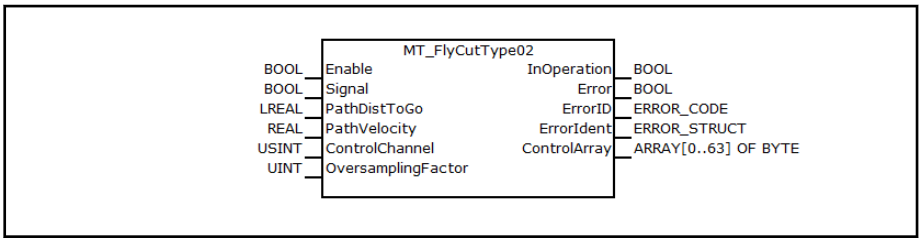


Fig. 16-24: IL_FIRType02 function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	Signal	BOOL	Input signal of the G code
	PathDistToGo	LREAL	Distance up to block end
	PathVelocity	REAL	Current path velocity
	ControlChannel	USINT	Output channel at the S20-DIDO-2/1-OVS module
	OversamplingFactor	UINT	Set oversampling rate at the S20-DIDO-2/1-OVS module
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Error handling, page 164
	ControlValue	ARRAY[0..63] OF BYTE	Signal for controlling the cutting head

Tab. 16-82: Interface variables of the MT_FlyCutType02 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
Signal	BOOL			FALSE	Continuous if "Enable" is TRUE
PathDistToGo	LREAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
PathVelocity	REAL	N. def.	N. def.	0.0	Continuous if "Enable" is TRUE
ControlChannel	USINT	1	2	0	Rising edge at "Enable"
OversamplingFactor	UINT	1	512	0	Rising edge at "Enable"

Tab. 16-83: Overview on the min./max. values and default values of the inputs

Functional description

The MT_FlyCutType02 function block controls the activation and deactivation of a cutting head with regard to the position. It is used for implementing the FlyCut functionality.

This function block requires the S20-DIDO-2/1-OVS S20 bus module for issuing the signal.

For this purpose, it requires some input data from the NC. It has to be transferred from the input data of the Fast Action task. Furthermore, a control signal that is activated in the NC block is required. This way, the function block knows that a switching operation has to be done at the block end. Delays due to the Fast Action task and other internal processes have already been calculated internally. The output signal can be directly issued to a high-speed output such as the local outputs of the control.

Signal, PathDistToGo, PathVelocity

These signals are transferred from the NC within a Fast Action task. "PathDistToGo" corresponds to the "PathDistToGo" command of the FA input data. "PathVelocity" is obtained from the "PathVelo" command of the FA input data.

The "Signal" input can be obtained from several sources. It can be generated via an event (`SEV()`, `REV()`). Both methods lead to different images in the FA input data.

ControlChannel, ControlArray

The S20-DIDO-2/1-OVS S20 module has two outputs. They are mapped in the "ControlArray" output. The "ControlChannel" input determines which of the two outputs is to be written to.

The "ControlArray" output has to be put on the digital output of the S20-DIDO-2/1-OVS S20 module. This can be done by mapping the I/O area to the output of the function block. If additional function blocks are to affect the S20 module, they have to be linked via the MT_Oversampling_AND and MT_Oversampling_OR function blocks.

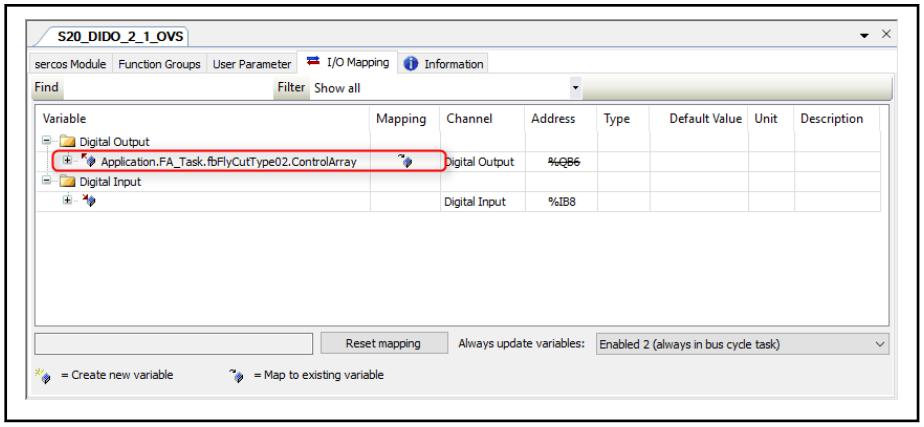


Fig. 16-25: I/O mapping in the S20-DIDO-2/1-OVS S20 module

OversamplingFactor

The "OversamplingFactor" input indicates how often a SERCOS cycle should be divided, i.e. how often switching can be performed within a Sercos cycle.

The shortest switching time of the module is 10 µs. When the Sercos bus is running with a cycle time of 1 ms, the maximum OversamplingFactor is 100.

The longest switching time is 1.3 ms.

The Sercos cycle time is automatically determined by the call rate of the function block. If the set value on the "OversamplingFactor" input is not within the permitted range, the function block switches to error state.

The value applied to the function block has to be identical to the value on the "Sample Rate" parameter set at the S20-DIDO-2/1-OVS S20 module.

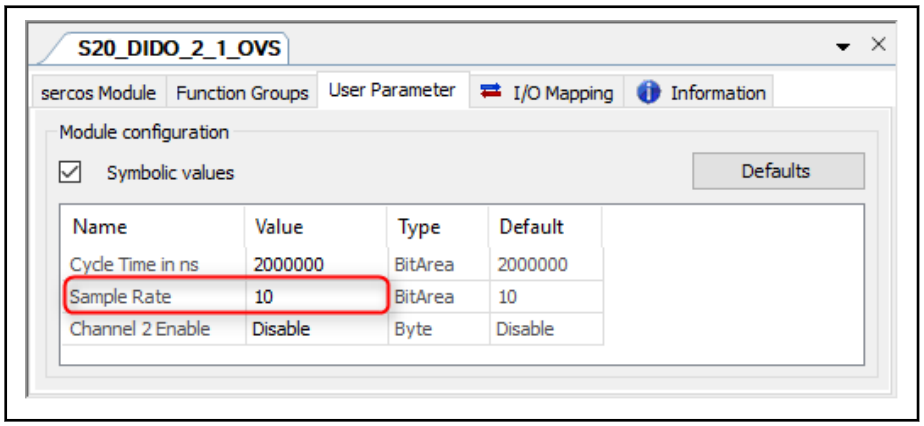


Fig. 16-26: Setting the "Sample Rate" parameter in the S20-DIDO-2/1-OVS S20 module

Circuitry

Call the function block in a fast Fast Action task. Link the following inputs and outputs with the respective values from the NC:

FB input	ID	Index 1	Index 2	Index 3	Access	Data type
Signal	100500	Event number (1..96)	-	-	0	BOOL
PathDistToGo	100410	Channel number	-	-	0	LREAL
PathVelocity	100411	Channel number	-	-	0	UDINT

Tab. 16-84: Link between the FA input data and the inputs of the function block

Normal operation

MT_FlyCutType02 calculates a new value for the "ControlArray" output with each cycle.

Configuration

The configuration contains the temporal behavior as well as the switch-on and switch-off positions with regard to the end of the NC block. The function block adopts the following system variables when calling the `Config.Update()` method on the function block instance:

System variable	Description
`SD.SysFlyCut[*].DistOn`	Indicates the switch-on distance in mm relative to the end point of the NC block
`SD.SysFlyCut[*].DistOff`	Indicates the switch-off distance in mm relative to the end point of the NC block
`SD.SysFlyCut[*].TimeOn`	Indicates the switch-on point in time in ms relative to the end point of the NC block
`SD.SysFlyCut[*].TimeOff`	Indicates the switch-off point in time in ms relative to the end point of the NC block

* NC channel, normally 1

Tab. 16-85: Settings of the MT_FlyCutType02 function block

Calling the `UpdateConfig()` method has the be carried out in a cyclic PLC task.

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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3132	16#0000	Parameter "TimeOn" > T#1S
INPUT_RANGE_ERROR (16#0006)	16#3133	16#0000	Parameter "TimeOff" > T#1S
INPUT_RANGE_ERROR (16#0006)	16#3136	16#0000	Parameter "DistOn" < 0.0 mm
INPUT_RANGE_ERROR (16#0006)	16#3137	16#0000	Parameter "DistOff" < 0.0 mm
INPUT_RANGE_ERROR (16#0006)	16#3134	16#0000	"OversamplingFactor" input invalid
INPUT_RANGE_ERROR (16#0006)	16#3135	16#0000	"ControlChannel" input invalid

Tab. 16-86: Error codes MT_FlyCutType02 function block

16.8 Auxiliary functions for oversampling

16.8.1 Introduction and overview

Auxiliary functions are required for merging several technology functions that are based on the oversampling technology for the S20-DIDO-2/1-OVS S20 oversampling module. They link two sources binary to a target. Here, a distinction is made between the two channels of the S20 oversampling module.

The S20-DIDO-2/1-OVS S20 bus module has two oversampling channels that can be interconnected, if required. The maximum oversampling factor is 256 if both channels are active. Therefore, a bus cycle is divided 256 times. Please note that the smallest switching unit must not exceed 10 µs. The maximum oversampling factor for a bus cycle of 1 ms is therefore 100.

If both oversampling channels are interconnected, an oversampling factor of up to 512 on the first channel is possible. However, the above-mentioned limitation of the minimum switching time of 10 µs remains valid.

64 bytes are transmitted in the cyclic channel of the oversampling module. They can be changed in every bus cycle. 512 bits of the 64 bytes are mapped in the two oversampling channels. Channel 1 starts from bit 0 (byte 0); channel 2 starts from bit 256 (byte 32). If only one channel is used, all 512 bits are available for the first channel.

The bits of the cyclic channel are issued in sequence in the cycle calculated on the basis of the oversampling factor. This is started with the most significant bit of the first byte. The same number of bits in a bus cycle is issued as specified in the oversampling factor. All other bits are ignored.

The function blocks described in this chapter cover this application case.

- MT_Oversampling_AND, page 165, executes an AND link per channel.
- MT_Oversampling_OR, page 167, executes an OR link per channel.

16.8.2 MT_Oversampling_AND

Brief description

The MT_Oversampling_AND function block executes an AND link on an oversampling channel for the S20-DIDO-2/1-OVS S20 oversampling module.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-87: Reference table of the MT_Oversampling_AND

Interface description

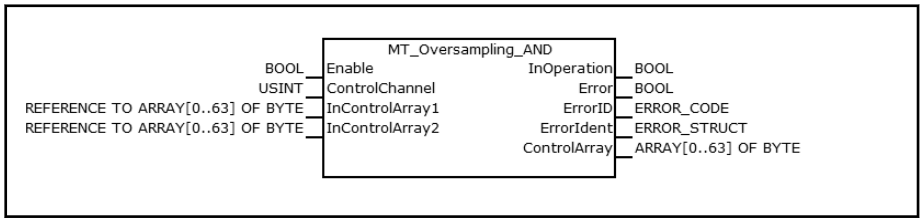


Fig. 16-27: MT_Oversampling_AND function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	ControlChannel	USINT	Channel to be linked 0 - Both channels 1: Channel 1 2: Channel 2
	InControlArray1	REFERENCE TO ARRAY[0..63] OF BYTE	First array to be linked
	InControlArray2	REFERENCE TO ARRAY[0..63] OF BYTE	Second array to be linked
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description
	ErrorIdent	ERROR_STRUCT	Detailed error description Error handling, page 167
	ControlArray	ARRAY[0..63] OF BYTE	AND-linked oversampling array

Tab. 16-88: Interface variables of the MT_Oversampling_AND 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
ControlChannel	USINT	0	2	0	Continuous
InControlArray1	REFERENCE TO ARRAY[0..63] OF BYTE				Continuous
InControlArray2	REFERENCE TO ARRAY[0..63] OF BYTE				Continuous

Tab. 16-89: Overview on the min./max. values and default values of the inputs

Functional description

The MT_Oversampling_AND function block executes an AND link bit by bit on one or both oversampling channels. It is intended for merging several technology functions on an oversampling channel as well as for merging several oversampling channels on the output array of the S20-DIDO-2/1-OVS S20 oversampling module.

ControlChannel

The "ControlChannel" input select which components of the two input arrays are linked:

- 0: Both arrays are AND-linked over the entire length.
- 1: The oversampling channel 1 (bit 0 to 255) is AND-linked; the remaining components are copied from the "InControlArray1" input.
- 2: The oversampling channel 2 (bit 256 to 511) is AND-linked; the remaining components are copied from the "InControlArray1" input.

InControlArray1, InControlArray2

Both "InControlArray1" and "InControlArray2" inputs have to be linked to the output of a corresponding function block (e.g. MT_FlyCutType02). If no array is created on the input, the function block reports an error.

ControlArray The "ControlArray" output contains the merged data of the "InControlArray1" and "InControlArray2" inputs. It can be connected further or interconnected to the HW output of the S20-DIDO-2/1-OVS S20 oversampling module as a reference.

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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3190	16#0001	InControlArray1 input not assigned
INPUT_RANGE_ERROR (16#0006)	16#3190	16#0002	InControlArray1 input not assigned
INPUT_RANGE_ERROR (16#0006)	16#3191	16#0000	ControlChannel input does not contain a valid value

Tab. 16-90: Error codes MT_Oversampling_AND function block

16.8.3 MT_Oversampling_OR

Brief description The MT_Oversampling_OR function block executes an OR link on an oversampling channel for the S20-DIDO-2/1-OVS S20 oversampling module.

Target system/library

Target system	Library
MTX 15VRS Function package TEC	MT_Technology

Tab. 16-91: Reference table of the MT_Oversampling_OR

Interface description

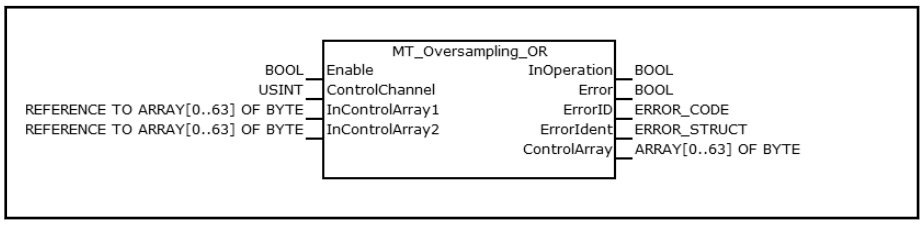


Fig. 16-28: MT_Oversampling_OR function block

I/O type	Name	Data type	Description
VAR_INPUT	Enable	BOOL	Enables the function block when set to TRUE
	ControlChannel	USINT	Channel to be linked 0 - Both channels 1: Channel 1 2: Channel 2
	InControlArray1	REFERENCE TO ARRAY[0..63] OF BYTE	First array to be linked
	InControlArray2	REFERENCE TO ARRAY[0..63] OF BYTE	Second array to be linked
VAR_OUTPUT	InOperation	BOOL	Function block is operating
	Error	BOOL	Indicates an error. Clear error with "Enable" = FALSE
	ErrorID	ERROR_CODE	Brief error description

I/O type	Name	Data type	Description
	ErrorIdent	ERROR_STRUCT	Detailed error description Error handling, page 168
	ControlArray	ARRAY[0..63] OF BYTE	AND-linked oversampling array

Tab. 16-92: Interface variables of the MT_Oversampling_OR 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
ControlChannel	USINT	0	2	0	Continuous
InControlArray1	REFERENCE TO ARRAY[0..63] OF BYTE				Continuous
InControlArray2	REFERENCE TO ARRAY[0..63] OF BYTE				Continuous

Tab. 16-93: Overview on the min./max. values and default values of the inputs

Functional description

The MT_Oversampling_OR function block executes an OR link bit by bit on one or both oversampling channels. It is intended for merging several technology functions on an oversampling channel as well as for merging several oversampling channels on the output array of the S20-DIDO-2/1-OVS S20 oversampling module.

ControlChannel

The "ControlChannel" input select which components of the two input arrays are linked:

- 0: Both arrays are OR-linked over the entire length.
- 1: The oversampling channel 1 (bit 0 to 255) is OR-linked; the remaining components are copied from the "InControlArray1" input.
- 2: The oversampling channel 2 (bit 256 to 511) is OR-linked; the remaining components are copied from the "InControlArray1" input.

InControlArray1, InControlArray2

Both "InControlArray1" and "InControlArray2" inputs have to be linked to the output of a corresponding function block (e.g. MT_FlyCutType02). If no array is created on the input, the function block reports an error.

ControlArray

The "ControlArray" output contains the merged data of the "InControlArray1" and "InControlArray2" inputs. It can be connected further or interconnected to the HW output of the S20-DIDO-2/1-OVS S20 oversampling module as a reference.

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 function block state
INPUT_RANGE_ERROR (16#0006)	16#3190	16#0001	InControlArray1 input not assigned
INPUT_RANGE_ERROR (16#0006)	16#3190	16#0002	InControlArray1 input not assigned
INPUT_RANGE_ERROR (16#0006)	16#3191	16#0000	ControlChannel input does not contain a valid value

Tab. 16-94: Error codes MT_Oversampling_OR function block

17 Service and support

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

Service Germany

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

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

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

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

Service worldwide

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

Preparing information

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

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

Index

A
Abbreviations ... 6
About this documentation ... 1
Auxiliary functions for oversampling ... 165
Axis group (Gantry) ... 91

C
Change record ... 1
Commissioning
  NC ... 14
  PLC ... 18
Complaints ... 6
Criticism ... 6
Customer Feedback ... 6
Cutting width compensation (kerf) ... 97

D
D-corrections ... 98
Documentation
  Change record ... 1

E
Enabling/disabling the height control ... 51
Error analysis with the NC analyzer ... 33
Error codes
  MT_DistanceSensorType01 ... 123
  MT_DistanceSensorType02 ... 126
  MT_FlyCut01 ... 160
  MT_FlyCutType02 ... 164
  MT_FocusControlType01 ... 158
  MT_GasControlType01 ... 156
  MT_HeightRegulationType01 ... 121
  MT_LaserPowerControlType01 ... 149
  MT_LaserPowerControlType02 ... 151
  MT_LaserPowerControlType03 ... 154
  MT_LeapFrogType01 ... 144
  MT_Oversampling_AND ... 167
  MT_Oversampling_OR ... 168
  MT_PlatesearchType01 ... 133
  MT_PlatesearchType02 ... 136
  MT_RaiseAxis ... 140

F
Fast data exchange ... 25
Fast inputs and outputs ... 26
Feedback ... 6
FlyCut ... 57, 158
Focus control ... 89, 156

G
Gantry ... 91
Gas control ... 85, 154

H
Height control ... 37
  Commissioning the project module ... 43
  Control behavior ... 48
  Control loop ... 38
  NC commands ... 50
  Project module ... 40, 42
  Sensor selection ... 39
  Setting parameters ... 49
  Special features ... 38
  Troubleshooting ... 54
HEIGHTCTRL
  Enabling/disabling the height control ... 51
Helpdesk ... 169
Hotline ... 169

I
Inputs and outputs of FA
  Configuration ... 30
Intended use
  Areas of application ... 7
  Introduction ... 7
  Use cases ... 7
Introduction to the shape cutting technology ... 9

K
Kerf ... 97

L
Laser power control ... 75
LeapFrog ... 65, 140

M
M-auxiliary functions
  Overview ... 23
Main_FA ... 25
MT_Technology ... 115
  FlyCut ... 158
  Focus control ... 156
  Gas control ... 154
  Height control ... 116
  LeapFrog ... 140
  Power control ... 145

N
NC
  Commission ... 14

O
Oversampling
  Auxiliary functions ... 165
Overview on target groups and product phases ... 1

P

PlateSearch..... 53
Purpose..... 1

R

Re-enter..... 101
Re-enter and Retrace functionality..... 101
Re-entry functions of the NC program..... 101
Retrace..... 101, 103

S

Sensor calibration file
Structure..... 51
Sensor calibration: PREC16P_CALIB..... 52
Sensor calibration: SENSORCALIB..... 50
Service hotline..... 169
Shape cutting machines..... 9
Shape cutting technology
Function test..... 19
Introduction..... 9
Project structure..... 12
Project template..... 11
Sample program..... 21
Shape cutting technology project template..... 11
Suggestions..... 6
Support..... 169
System data
Add a new output variable..... 31

T

Target groups..... 1
Technology tables..... 105

U

Unintended use..... 8
Consequences, disclaimer..... 7

Notes

rexroth
A Bosch Company

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

R911399147

DOK-MTX***-TECHCUT*V15-CO06-EN-P