Electric Drives and Controls | Hydraulics | Linear Motion and Assembly Technologies | Pneumatics | Service

Rexroth
Bosch Group

Rexroth IndraDrive
Rexroth IndraMotion MLD
Application Examples

R911324034
Edition 02

Application Manual

Title Rexroth IndraDrive
Rexroth IndraMotion MLD
Application Examples

Type of Documentation Application Manual

Document Typecode DOK-INDRV*-MLD-APPLI**-AW02-EN-P

Internal File Reference RS-5bed5147cc1fce5f0a6846a00169d1a1-1-en-US-3

Purpose of Documentation By means of four application examples, this documentation shows an introduction to the programming of the drive-integrated PLC (IndraMotion MLD).

Record of Revision

Edition	Release Date	Notes
DOK-INDRV*-MLD-APPLI**-AW01-EN-P	09/2008	First edition
DOK-INDRV*-MLD-APPLI**-AW02-EN-P	07/2010	Edition 02

Published by Bosch Rexroth AG

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

Telephone +49 (0)93 52/ 40-0 ■ Fax +49 (0)93 52/ 40-48 85

http://www.boschrexroth.com/

Dept. DC-IA/EDY

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

Table of Contents

Table of Contents

Page

1 Introduction

1.1 About This Documentation

Purpose of Documentation

By means of four application examples, this documentation shows an introduction to the programming of the drive-integrated PLC (IndraMotion MLD). It demonstrates various problems occurring in the production process which are resolved with IndraMotion MLD.

The following examples of application are introduced and implemented as MLD solutions:

- Double-axis positioning control (Pick and Place)
- Intelligent error reaction
- Synchronous multi-axis motion with virtual master axis
- Vibration damping with superimposed process loop

In a simple way, the program examples demonstrate how to realize the following topics:

- Reading and writing drive parameters
- Reading digital and analog inputs
- Setting digital outputs
- Implementing a process control
- Motion Control: (Positioning and axis-synchronous motion)
- Programming an internal error reaction

1.2 Reference Documentations

1.2.1 IndraMotion MLD

Title	Type of documentation	Document typecode[1)]	Part number
Rexroth IndraMotion MLD	Application Manual	DOK-INDRV*-MLD-**VRS**-AWxx-EN-P	R911306084
Rexroth IndraMotion MLD Library	Library Description	DOK-INDRV*-MLD-SYSLIB*-FKxx-EN-P	R911309224
Rexroth IndraMotion MLD Getting Started	Summary	DOK-IM*MLD*-F*STEP**V**-KB01-EN-P	R911319306

1) In the document typecodes, "xx" is a wild card for the current edition of the documentation (example: AW03 is the third edition of an Application Manual);

Fig. 1-1: Documentations – Overview

1.2.2 Firmware

Title Rexroth IndraDrive …	Type of documentation	Document typecode[1)]	Part number
Rexroth IndraDrive Firmware for Drive Controllers	Functional Description	DOK-INDRV*-MP*-05VRS**-FKxx-EN-P	R911320182

1) In the document typecodes, "xx" is a wild card for the current edition of the documentation (example: FK02 is the second edition of a Functional Description);

Fig. 1-2: Documentations – Overview

1.2.3 Drive System

Title Rexroth IndraDrive	Type of documentation	Document typecode[1)]	Part number
Rexroth IndraDrive Drive Controllers Control Sections	Project Planning Manual	DOK-INDRV*-CSH********-PRxx-EN-P	R911295012

1) In the document typecodes, "xx" is a wild card for the current edition of the documentation (example: PR01 is the first edition of a Project Planning Manual);

Fig. 1-3: Documentations – Overview

2 Important Directions for Use

2.1 Appropriate Use

2.1.1 Introduction

Rexroth products represent state-of-the-art developments and manufacturing. They are tested prior to delivery to ensure operating safety and reliability.

⚠ WARNING **Personal injury and property damage caused by incorrect use of the products!**

The products have been designed for use in the industrial environment and may only be used in the appropriate way. If they are not used in the appropriate way, situations resulting in property damage and personal injury can occur.

☞ Rexroth as manufacturer is not liable for any damages resulting from inappropriate use. In such cases, the guarantee and the right to payment of damages resulting from inappropriate use are forfeited. The user alone carries all responsibility of the risks.

Before using Rexroth products, make sure that all the pre-requisites for an appropriate use of the products are satisfied:

- Personnel that in any way, shape or form uses our products must first read and understand the relevant safety instructions and be familiar with appropriate use.
- If the products take the form of hardware, then they must remain in their original state, in other words, no structural changes are permitted. It is not permitted to decompile software products or alter source codes.
- Do not mount damaged or faulty products or use them in operation.
- Make sure that the products have been installed in the manner described in the relevant documentation.

2.1.2 Areas of Use and Application

Drive controllers made by Rexroth are designed to control electrical motors and monitor their operation.

Control and monitoring of the Drive controllers may require additional sensors and actors.

☞ The drive controllers may only be used with the accessories and parts specified in this documentation. If a component has not been specifically named, then it may neither be mounted nor connected. The same applies to cables and lines.

Operation is only permitted in the specified configurations and combinations of components using the software and firmware as specified in the relevant Functional Descriptions.

Drive controllers have to be programmed before commissioning, making it possible for the motor to execute the specific functions of an application.

Drive controllers of the Rexroth IndraDrive line have been developed for use in single- and multi-axis drive and control tasks.

To ensure application-specific use of Drive controllers, device types of different drive power and different interfaces are available.

Typical applications include, for example:

- Handling and mounting systems,
- Packaging and food machines,
- Printing and paper processing machines and
- Machine tools.

Drive controllers may only be operated under the assembly and installation conditions described in this documentation, in the specified position of normal use and under the ambient conditions as described (temperature, degree of protection, humidity, EMC, etc.).

2.2 Inappropriate Use

Using the Drive controllers outside of the operating conditions described in this documentation and outside of the indicated technical data and specifications is defined as "inappropriate use".

Drive controllers must not be used, if ...

- they are subject to operating conditions that do not meet the specified ambient conditions. This includes, for example, operation under water, under extreme temperature fluctuations or extremely high maximum temperatures.
- Furthermore, Drive controllers must not be used in applications which have not been expressly authorized by Rexroth. Please carefully follow the specifications outlined in the general Safety Instructions!

☞ Components of the drive system Rexroth IndraDrive are **products of category C3** (with restricted distribution) according to IEC 61800-3. These components are not provided for use in a public low-voltage mains supplying residential areas. If these components are used in such a mains, high-frequency interference is to be expected. This can require additional measures of radio interference suppression.

3 Safety Instructions for Electric Drives and Controls

3.1 Definitions of Terms

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

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

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

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

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

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

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

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

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

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

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

Qualified Persons In terms of this application documentation, qualified persons are those persons who are familiar with the installation, mounting, commissioning and operation of the components of the electric drive and control system, as well as with the hazards this implies, and who possess the qualifications their work requires. To comply with these qualifications, it is necessary, among other things,

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

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

3) to attend a course of instruction in first aid

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

3.2 General Information

3.2.1 Using the Safety Instructions and Passing Them on to Others

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

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

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

3.2.2 Requirements for Safe Use

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

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

concept in which measures of risk reduction for personal safety depend on electrical, electronic or programmable control systems.

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

  The machine and installation manufacturers must

  – make sure that the delivered components are suited for their individual application and check the information given in this application documentation with regard to the use of the components,
  – make sure that their individual application complies with the applicable safety regulations and standards and carry out the required measures, modifications and complements.
- Commissioning of the delivered components is only allowed once it is sure that the machine or installation in which the components are installed complies with the national regulations, safety specifications and standards of the application.
- Operation is only allowed if the national EMC regulations for the application are met.
- The instructions for installation in accordance with EMC requirements can be found in the section on EMC in the respective application documentation.

  The machine or installation manufacturer is responsible for compliance with the limit values as prescribed in the national regulations.
- The technical data, connection and installation conditions of the components are specified in the respective application documentations and must be followed at all times.

National regulations which the user must take into account

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

3.2.3 Hazards by Improper Use

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

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

3.3 Instructions with Regard to Specific Dangers

3.3.1 Protection Against Contact with Electrical Parts and Housings

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

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

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

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

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

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

- Before switching on and before commissioning, ground or connect the components of the electric drive and control system to the equipment grounding conductor at the grounding points.
- Connect the equipment grounding conductor of the components of the electric drive and control system permanently to the main power supply at all times. The leakage current is greater than 3.5 mA.
- Establish an equipment grounding connection with a copper wire of a cross section of at least 10 mm^2 (8 AWG) or additionally run a second equipment grounding conductor of the same cross section as the original equipment grounding conductor.

3.3.2 Protective Extra-Low Voltage as Protection Against Electric Shock

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

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

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

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

3.3.3 Protection Against Dangerous Movements

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

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

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

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

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

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

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

To avoid accidents, injury and/or property damage:

- Keep free and clear of the machine's range of motion and moving machine parts. Prevent personnel from accidentally entering the machine's range of motion by using, for example:
  - Safety fences
  - Safety guards
  - Protective coverings
  - Light barriers
- Make sure the safety fences and protective coverings are strong enough to resist maximum possible kinetic energy.
- Mount emergency stopping switches in the immediate reach of the operator. Before commissioning, verify that the emergency stopping equipment works. Do not operate the machine if the emergency stopping switch is not working.
- Prevent unintended start-up. Isolate the drive power connection by means of OFF switches/OFF buttons or use a safe starting lockout.
- Make sure that the drives are brought to safe standstill before accessing or entering the danger zone.
- Additionally secure vertical axes against falling or dropping after switching off the motor power by, for example,
  - mechanically securing the vertical axes,
  - adding an external braking/arrester/clamping mechanism or
  - ensuring sufficient counterbalancing of the vertical axes.
- The standard equipment **motor holding brake** or an external holding brake controlled by the drive controller is **not sufficient to guarantee personal safety**!
- Disconnect electrical power to the components of the electric drive and control system using the master switch and secure them from reconnection ("lock out") for:
  - Maintenance and repair work
  - Cleaning of equipment
  - Long periods of discontinued equipment use
- Prevent the operation of high-frequency, remote control and radio equipment near components of the electric drive and control system and their supply leads. If the use of these devices cannot be avoided, check the machine or installation, at initial commissioning of the electric drive and control system, for possible malfunctions when operating such high-frequency, remote control and radio equipment in its possible positions of normal use. It might possibly be necessary to perform a special electromagnetic compatibility (EMC) test.

3.3.4 Protection Against Magnetic and Electromagnetic Fields During Operation and Mounting

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

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

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

3.3.5 Protection Against Contact With Hot Parts

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

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

3.3.6 Protection During Handling and Mounting

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

- Observe the relevant statutory regulations of accident prevention.
- Use suitable equipment for mounting and transport.
- Avoid jamming and crushing by appropriate measures.

- Always use suitable tools. Use special tools if specified.
- Use lifting equipment and tools in the correct manner.
- Use suitable protective equipment (hard hat, safety goggles, safety shoes, safety gloves, for example).
- Do not stand under hanging loads.
- Immediately clean up any spilled liquids from the floor due to the risk of slipping.

3.3.7 Battery Safety

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

Risk of injury by improper handling!

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

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

3.3.8 Protection Against Pressurized Systems

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

Risk of injury by improper handling of pressurized lines!

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

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

3.4 Explanation of Signal Words and the Safety Alert Symbol

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

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

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

⚠ DANGER

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

⚠ WARNING

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

⚠ CAUTION

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

NOTICE

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

4 Requirements

4.1 Firmware and Hardware Requirements

Using the drive-integrated PLC (Rexroth IndraMotion MLD-S/M) requires the following hardware/firmware combinations:

Control section for IndraDrive C/M:

- CSH01.*C (as of FWA-INDRV-MPH02VRS)
- CSB01.1** (as of FWA-INDRV-MPB03VRS)

IndraDrive Cs:

- HCS01 (as of FWA-INDRV-MPB17VRS)

☞ With BASIC control sections (CSB01.1), using MLD-S has only been enabled for self-contained Bosch Rexroth system solutions ("technology functions")! MLD is not available for double-axis devices (HMD01.1 with CDB01.1C).

☞ As of firmware MPH04V06, IndraMotion MLD is available as a multi-axis PLC (MLD-M). Access to remote axes (CCD slaves) requires the control section hardware CSH01.2C (firmware versions MPH04/05VRS) or CSH01.3C (firmware version >= MPC06VRS) and the enabling of the functional firmware package "ML".

☞ Using the PLC functionality does not require any special optional card or control section configuration, because it is a PLC that is running in parallel in the drive processor in the real-time kernel.

4.2 Enabling of Functional Packages

In addition to the functional package "Closed Loop", the functional package "IndraMotion MLD" (drive PLC) must have been enabled in the drive so that IndraMotion MLD can be used.

Requirements

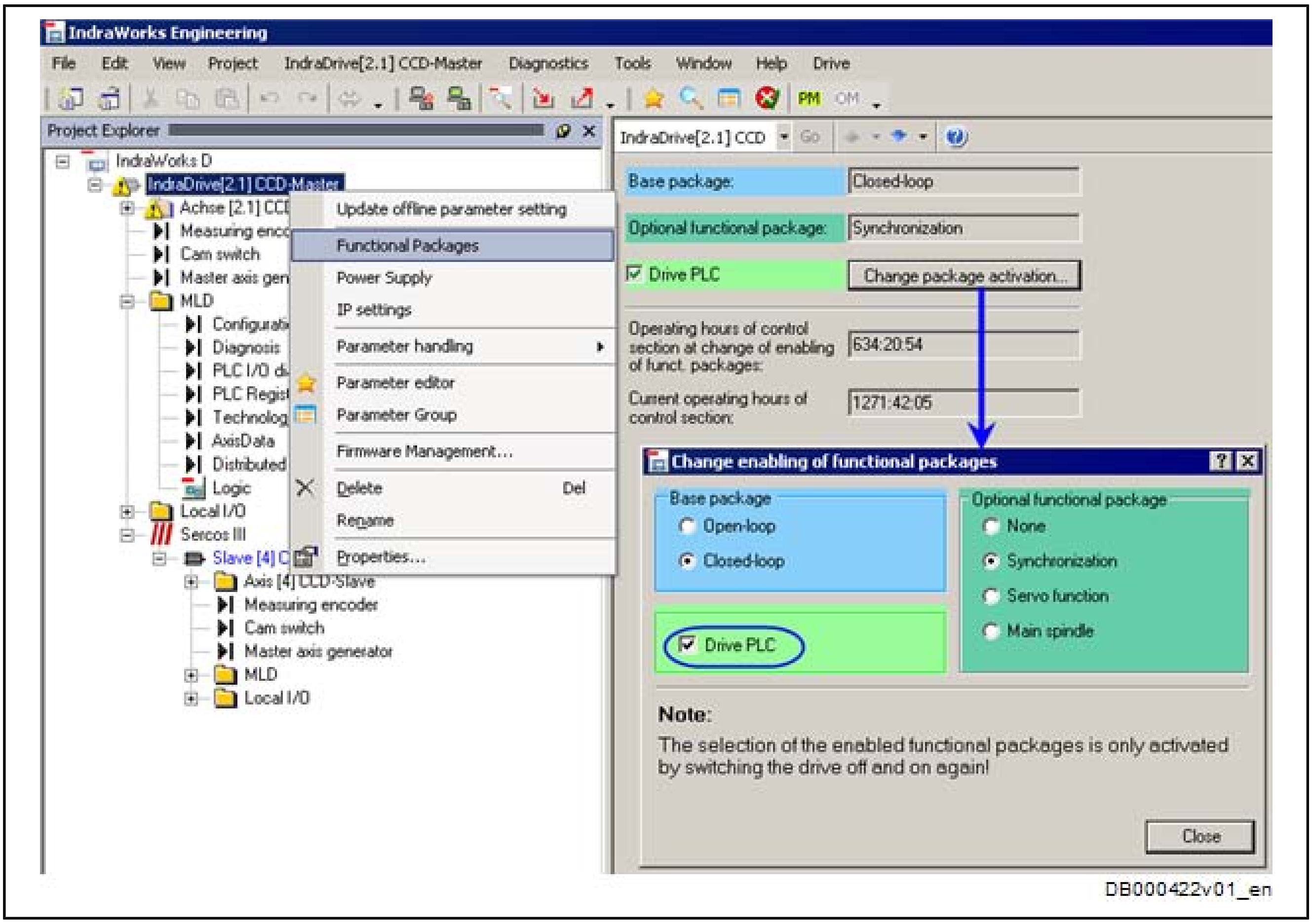


Fig.4-1: IndraWorks Dialog to Enable the Functional Package "IndraMotion MLD" (Drive PLC)

Possible configurations of IndraMotion MLD

- **TF:** IndraMotion MLD for using the self-contained Bosch Rexroth system solutions (technology functions) (with MPB firmware)
- **ML:** IndraMotion MLD for free programming of the single axis; including the use of the technology functions (with MPH/C firmware)
- **MA:** IndraMotion MLD Advanced for multi-axis systems (MLD-M) and turnkey solutions (with MPH/C firmware)

To implement the individual application examples, it might possibly be necessary to enable another functional package. This will be described within the corresponding chapter.

☞ It is only allowed to enable licensed functional packages!

4.3 Programming

We assume that you basically know how to handle the IndraLogic commissioning software. For more information, see the following documentations:

- IndraMotion MLD - Getting Started "R911319306"
- Rexroth IndraMotion MLD "R911306084"

The MLD applications examples are available on the following media:

Requirements

- Installation data carrier IndraWorks MLD in the "AddOns" directory
- Media directory

5 Double-Axis Positioning Control (Pick and Place)

5.1 Task Definition – Application Description

5.1.1 General Information

Workpieces are to be moved from one place to another place. The required axis motions are to be carried out one after the other. In addition, the control (digital output) and feedback (digital input) of the pneumatic picker are to be handled via IndraMotion MLD. The procedure is to be started via a switch-key which is read in at a digital input at the master.

Control ("close" picker, "open" picker) and feedback (picker "closed") for the pneumatic picker are to be controlled via digital inputs/outputs at the SERCOS slave. This sets the additional task to access remote inputs/outputs with MLD.

5.1.2 Mechanical Configuration

The figure below illustrates the mechanical scheme of the double-axis positioning control "Pick and Place".

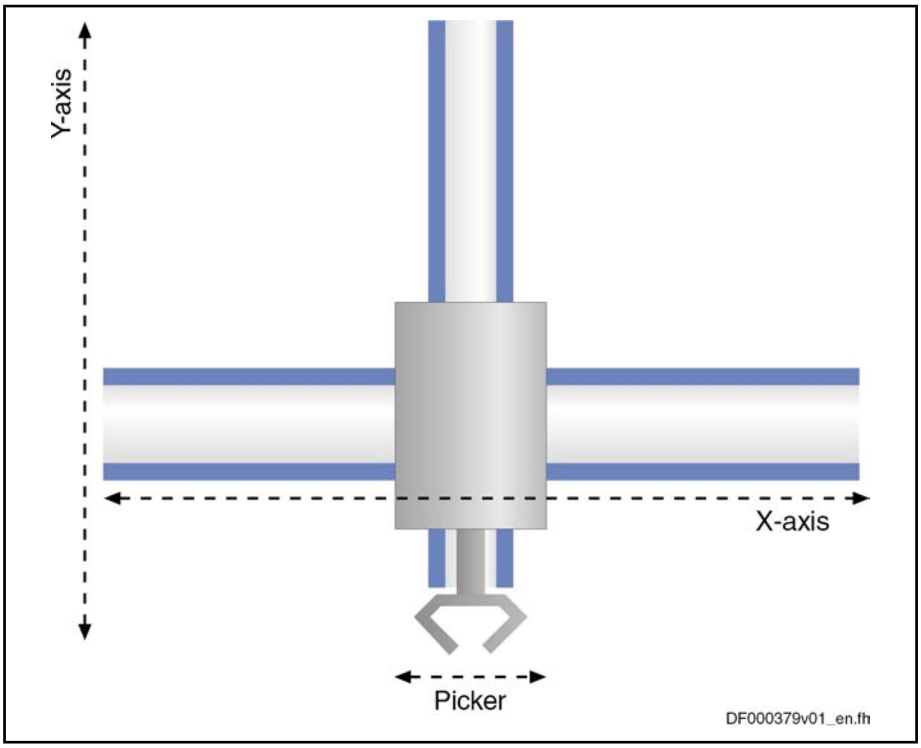
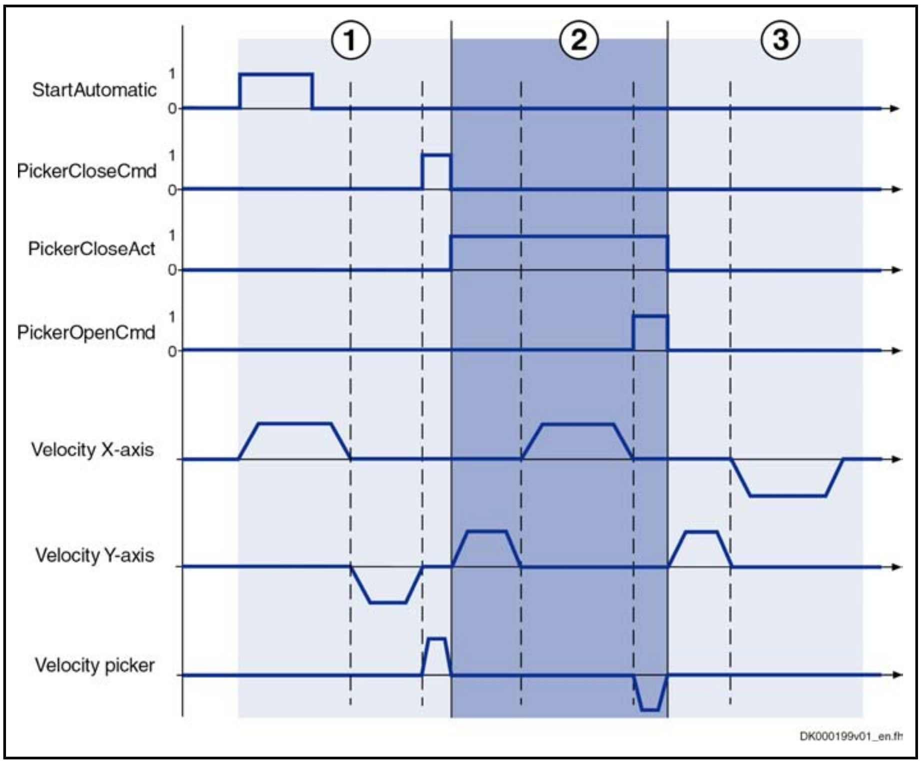


Fig.5-1: Mechanical Scheme of the Application "Pick and Place"

5.1.3 Sequence of Motion

The chronological diagram below illustrates the sequence of motion of the double-axis positioning control "Pick and Place".



1	Pick up product
2	Transport to placing position
3	Return to start position
Fig.5-2:	*Sequence of Motion of the Application "Pick and Place"*

Step 1:

Upon a positive edge at the "bStartAutomatic" input (P-0-1390, bit 0, %IX0.0), the X- and Y-axes are switched to enable. The X-axis first and then the Y-axis move to the picking position. When the 1st positioning process of both axes has been completed, the "bPickerCloseCmd" output (P-0-1411, bit 8, %QX1.8) is set whereby the picker closes and takes up the workpiece.

Step 2:

When the picker has closed, this is signaled by the feedback "bPickerCloseAct" (P-0-1440, bit 1, %IX50.1) and the movement to the placing position is carried out. In this case, it is first the Y-axis and then the X-axis which is moved. When the placing position has been reached, the "bPickerOpenCmd" output (P-0-1411, bit 9, %QX1.9) is set upon which the picker opens and places the workpiece.

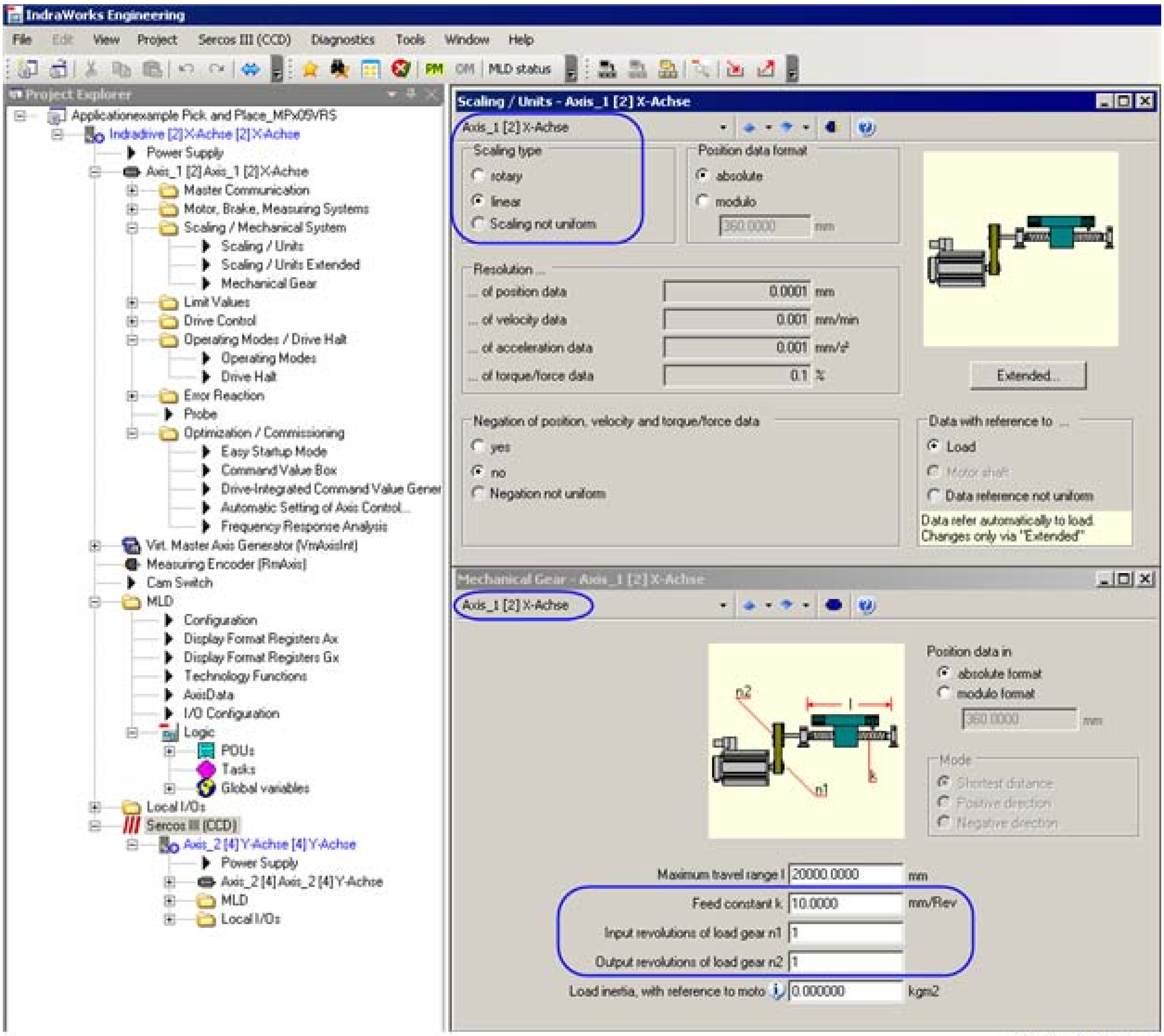
Step 3:

The 0-signal of the "bPickerCloseAct" input (P-0-1440, bit 1, %IX50.1) signals that the picker has opened and this triggers the movement to the start position. For this purpose, it is first the Y-axis and then the X-axis which positions. When the travel process has been completed, the enable signal is removed at the axes.

Starting from the basic parameters, you have to make some fundamental settings for the example of application "Pick and Place". The following paragraphs will explain these settings in short form.

Mechanical Data

According to the mechanical configuration, you have to set the scaling, gear and feed constant for the X- and Y-axes.



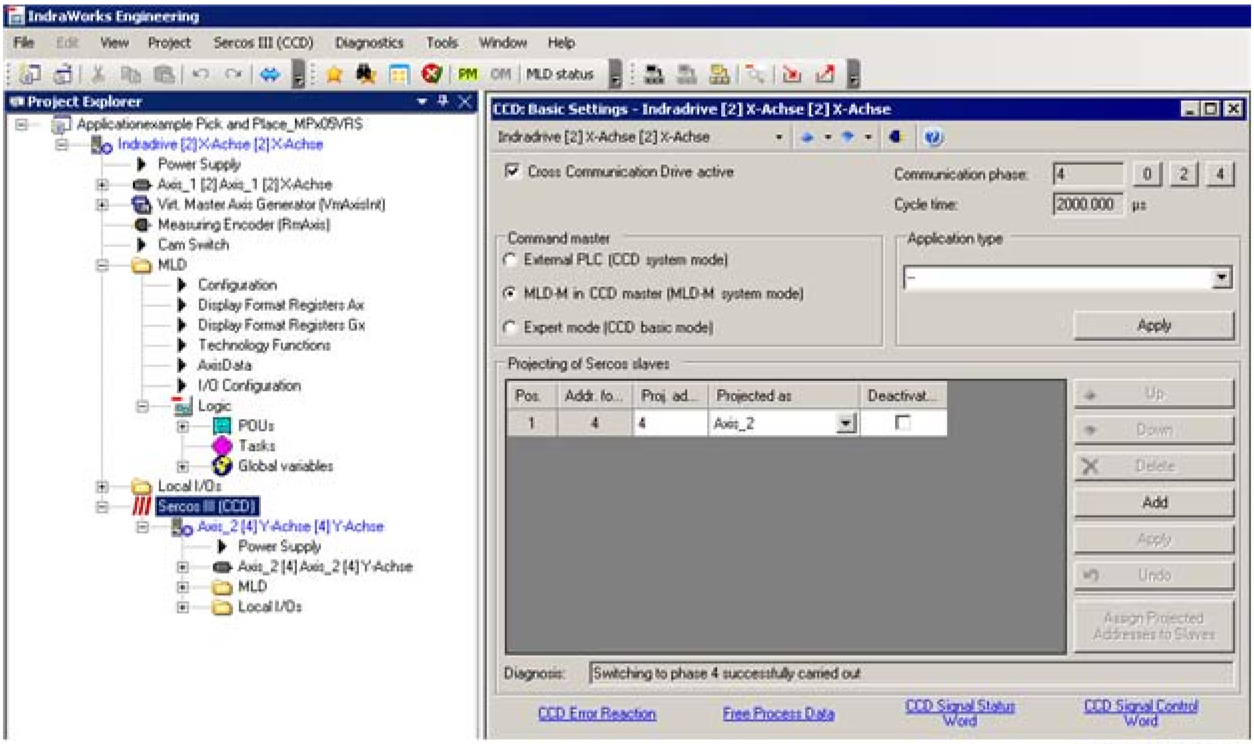
DB000273v01_en

Fig.5-3: Example: Mechanical Data for X-Axis

CCD Configuration

First you have to activate the CCD communication and select the MLD-M system mode. The axis with address "4" has been configured as CCD slave (Y-axis).

Double-Axis Positioning Control (Pick and Place)



DB000274v01_en

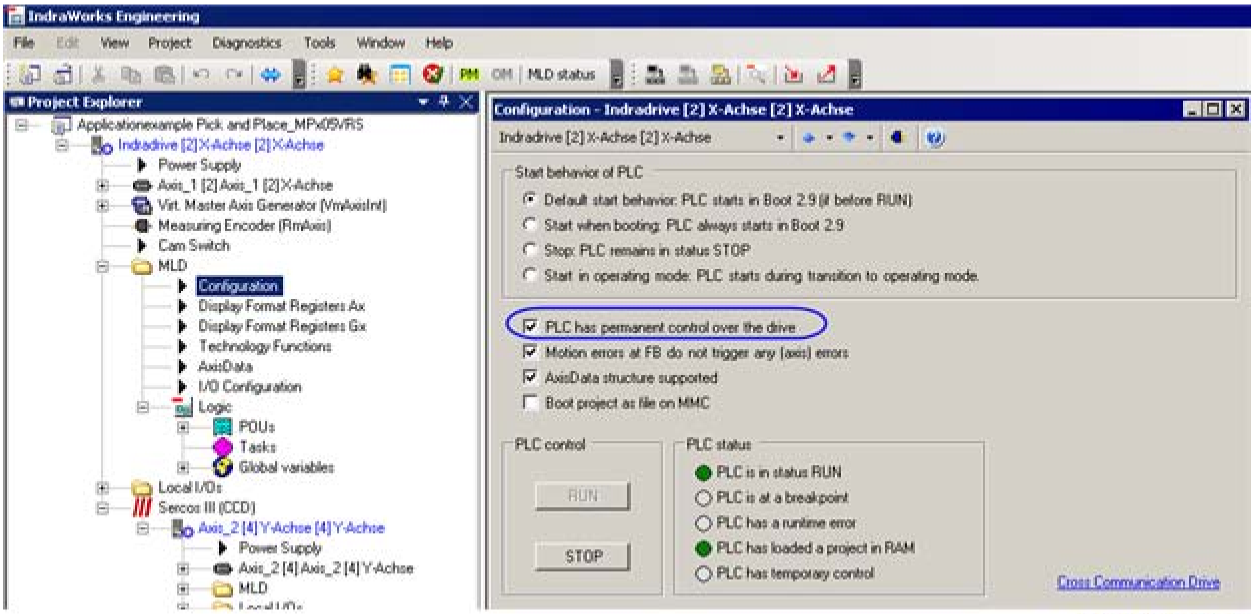
Fig.5-4: IndraWorks Dialog for CCD Settings

The resulting axis addressing in MLD-M is:

- X-axis (axis address 2) → Axis 1 in MLD
- Y-axis (axis address 4) → Axis 2 in MLD

MLD Configuration

In the drive PLC, you have to select permanent control for the CCD master.

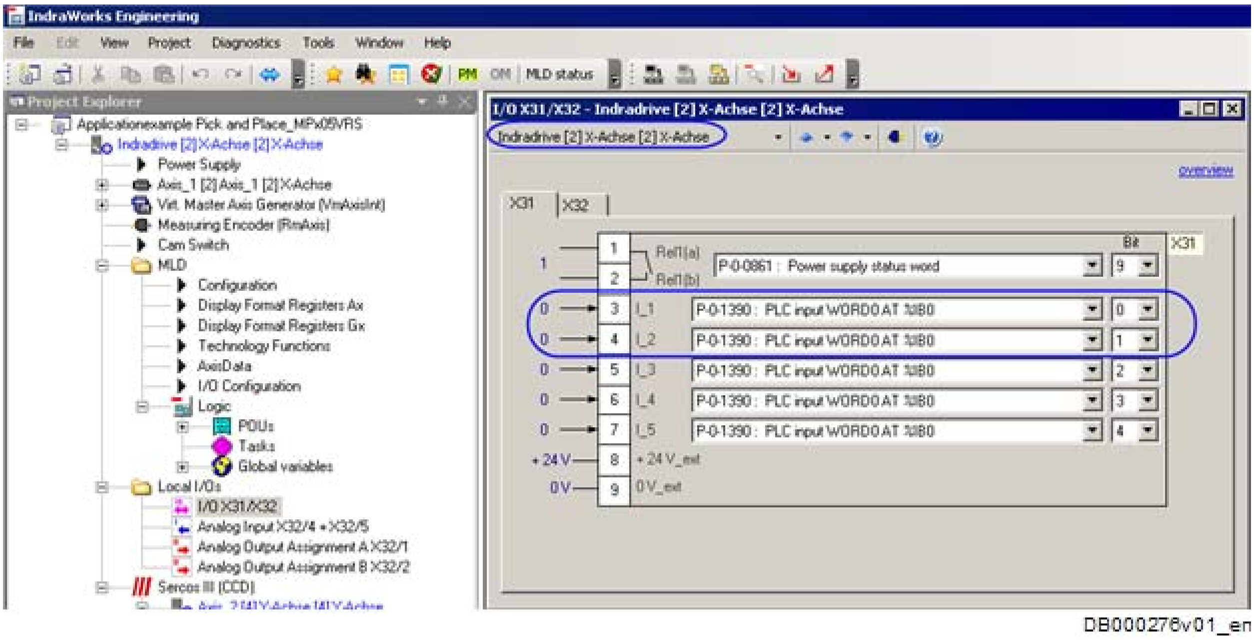


DB000275v01_en

Fig.5-5: IndraWorks Dialog for MLD Configuration

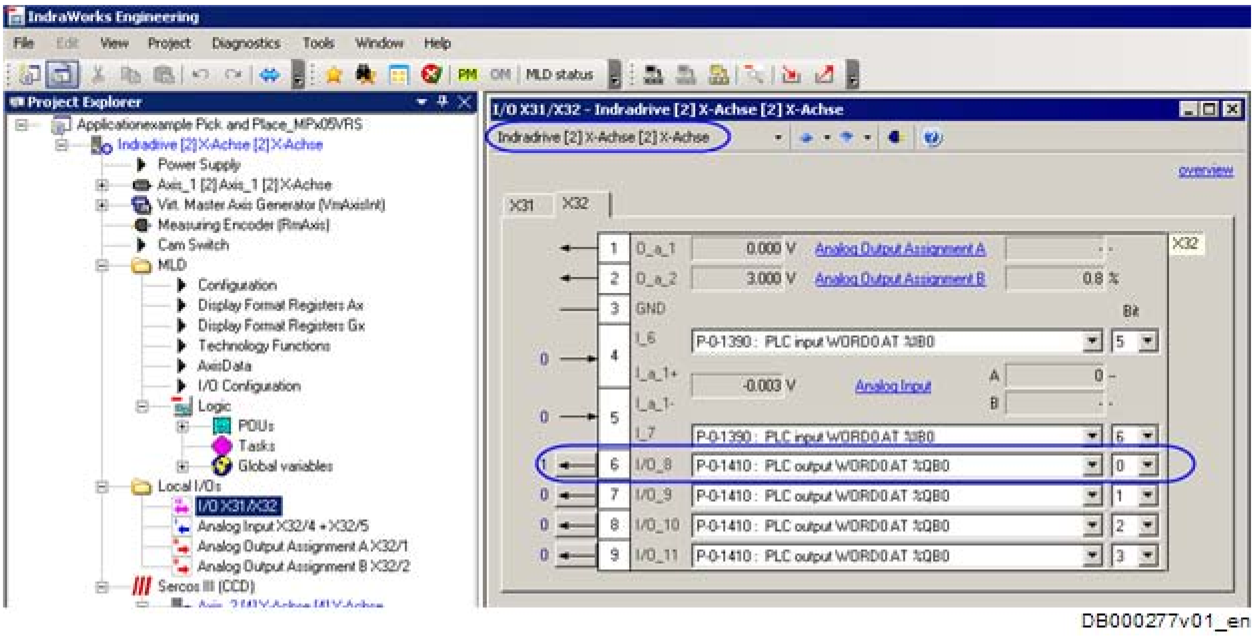
Configuring the Digital Inputs/Outputs at the Master

The digital inputs and outputs at the terminals X31/32 have to be parameterized at the X-axis (CCD master) in accordance with the following IndraWorks dialog.



X31.3 P-0-1390, bit 0 (%IX0.0) → bStartAutomatic
X31.4 P-0-1390, bit 1 (%IX0.1) → bProgramReset

Fig.5-6: Configuration "X31" of X-Axis (CCD Master)



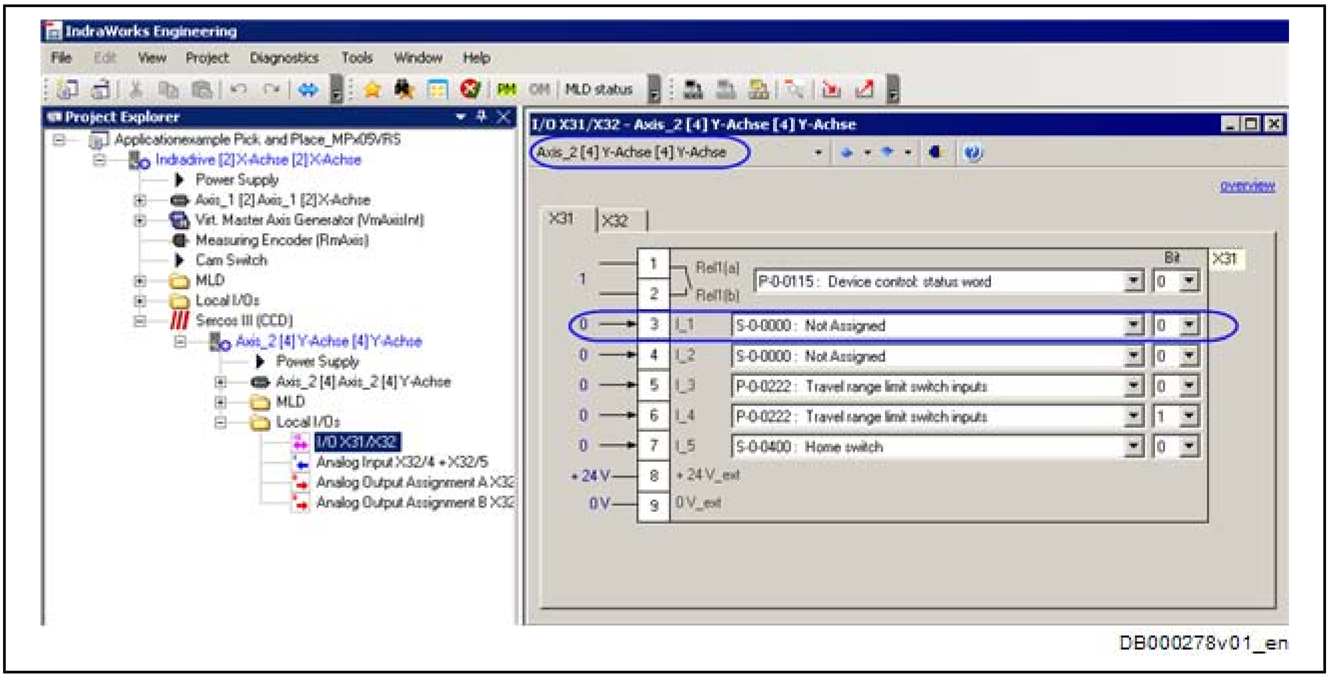
X32.3 P-0-1410, bit 0 (%QX0.0) → bPickerActive

Fig.5-7: Configuration "X32" of X-Axis (CCD Master)

Configuring the Digital Inputs/Outputs at the Slave

The digital inputs and outputs at X31/32 have to be parameterized at the Y-axis (CCD slave) in accordance with the following IndraWorks dialog.

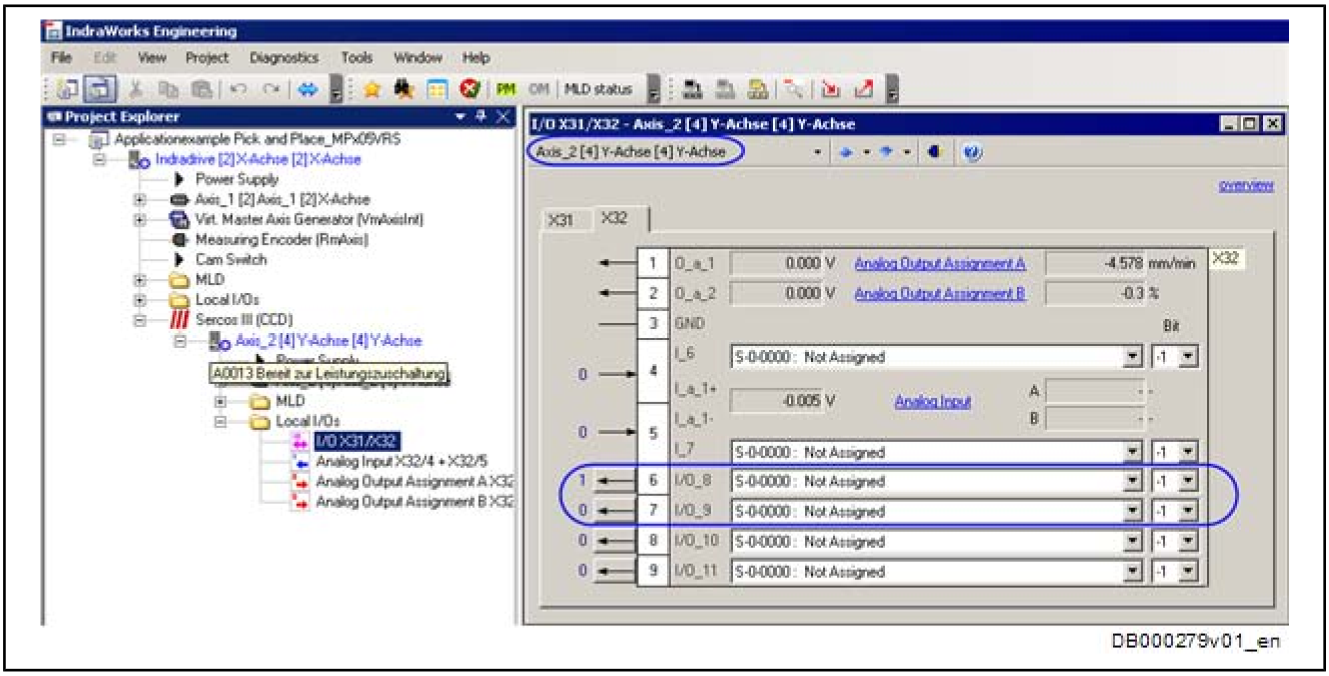
Double-Axis Positioning Control (Pick and Place)



X31.3 P-0-0303, bit 1 → bPickerCloseAct

Fig.5-8: Configuration "X31" of Y-Axis (CCD Slave)

You can simply configure a dummy parameter for the digital input "I_1" of the CCD slave, because the status of the input is copied directly from parameter P-0-0303 (signal status of the digital inputs) to the CCD master (see also figure "Configuring the Distributed Inputs/Outputs").



X32.6 P-0-0304, → bit 8 bPickerCloseCmd

X32.7 P-0-0304, bit 9 → bPickerOpenCmd

Fig.5-9: Configuration "X32" of Y-Axis (CCD Slave)

The digital outputs only have to be configured as outputs and a dummy parameter can be assigned to them as it is done for the inputs. The status of the output is influenced by the CCD master by direct writing of the parameter

P-0-0304 (signal status of the digital outputs) (see also figure "Configuring the Distributed Inputs/Outputs").

Configuring the "Distributed Inputs/ Outputs"

The settings shown in the following IndraWorks dialog are required to transmit the input which has been read in from the Y-axis (CCD slave) to the X-axis (CCD master) or to set the outputs at the Y-axis (CCD slave) from the MLD-M of the X-axis (CCD master).

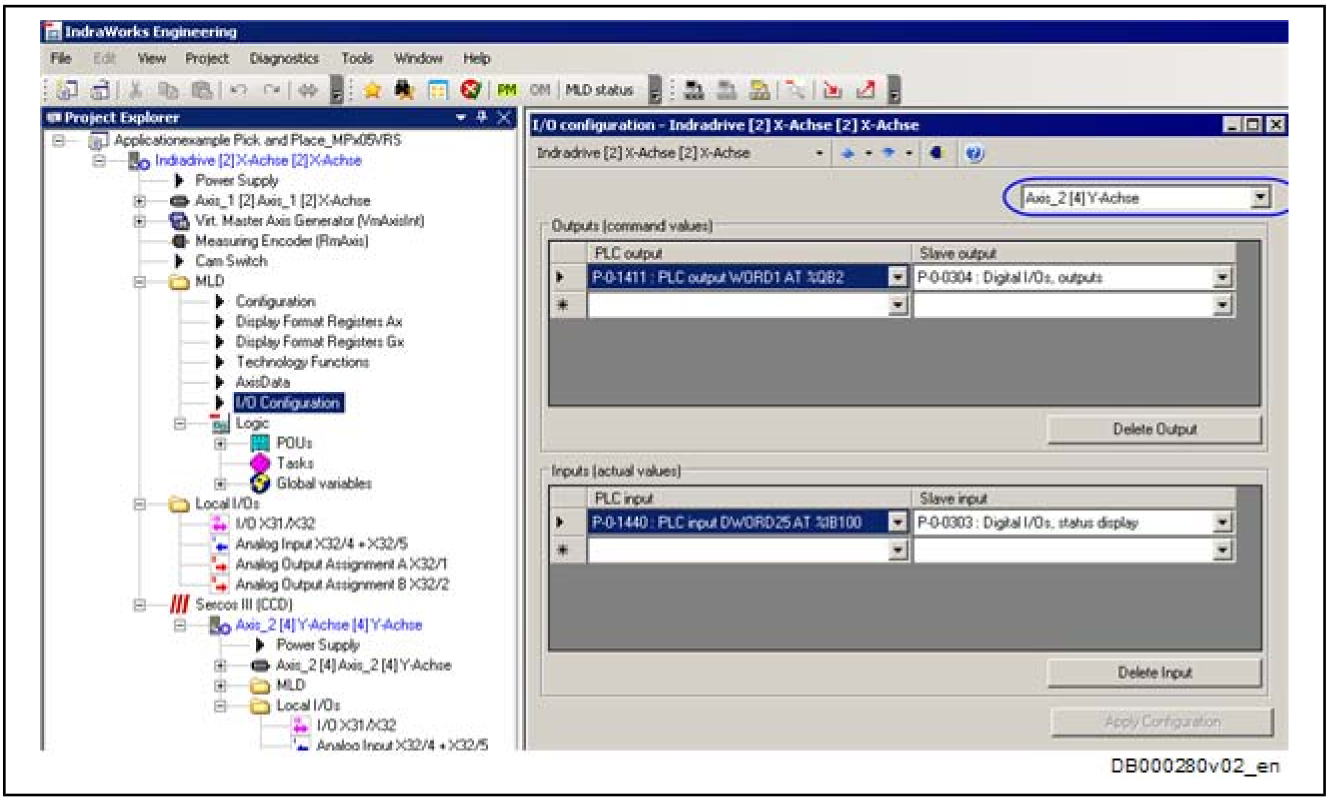


Fig.5-10: IndraWorks Dialog for Configuring the Distributed Inputs/Outputs

In the application example, the parameter P-0-1411 is written by MLD-M. By the above-mentioned configuration, this parameter directly takes effect on the status of the digital outputs (P-0-0304) in the Y-axis (CCD slave).

The status of the digital inputs of the CCD slave (P-0-0303) is copied to the parameter P-0-1440 of the CCD master and evaluated there by MLD-M. As the parameter P-0-0303 is a 32-bit value, it has to be assigned to a 32-bit process image register, such as parameter P-0-1440.

You have to observe in which bits the corresponding terminals take effect. For example, the output "I/O_8" which is used has to be addressed in the CCD slave via bit 8 of the parameter P-0-0304.

ℹ See also Parameter Description for "P-0-0303, Digital I/Os, inputs" and "P-0-0304, Digital I/Os, outputs"

Double-Axis Positioning Control (Pick and Place)

5.2 Programming

1. **Variable declaration**

   In the variable declaration, the variables which are used are created and assigned to the inputs and outputs.

```
PROGRAM PLC_PRG
VAR
    bStartAutomatic     AT %IX0.0   : BOOL      (*P-0-1390, Bit0*);
    bProgramReset       AT %IX0.1   : BOOL      (*P-0-1390, Bit1*);
    bPickerCloseAct     AT %IX50.1  : BOOL      (*P-0-1440, Bit1*);
    bPickerActiv        AT %QX0.0   : BOOL      (*P-0-1410, Bit0*);
    bPickerCloseCmd     AT %QX1.8   : BOOL      (*P-0-1411, Bit8*);
    bPickerOpenCmd      AT %QX1.9   : BOOL      (*P-0-1411, Bit9*);

    iStep                           : INT;
    bStart                          : BOOL;

    fbStartEdge                     : R_TRIG;
    fbMC_PowerXAxis                 : MC_Power;
    fbMC_PowerYAxis                 : MC_Power;
    fbMX_MoveAbsoluteXAxis          : MX_MoveAbsolute;
    fbMX_MoveAbsoluteYAxis          : MX_MoveAbsolute;
END_VAR
```

DB000281v01_nn.tif

Fig.5-11: Variable Declaration

2. **Initialization**

   In the first initialization step, all variables or function blocks are brought to a defined status.

```
(*Reset of Motion function blocks and variables, if Picker is not active or ProgramReset*)
IF NOT bPickerActiv OR bProgramReset  THEN
    fbMX_MoveAbsoluteXAxis(Execute:= FALSE,
                           Axis:= Axis1);
    fbMX_MoveAbsoluteYAxis(Execute:= FALSE,
                           Axis:= Axis2);

    fbMC_PowerXAxis (Enable:=FALSE ,
                     Axis:= Axis1);
    fbMC_PowerYAxis (Enable:=FALSE ,
                     Axis:= Axis2);

    istep:=0;
    bPickerActiv:=FALSE;
    bPickerCloseCmd:=FALSE;
    bPickerOpenCmd:=FALSE;
END_IF
```

DB000282v01_nn

Fig.5-12: Initialization

3. **Generating the start edge**

   After a positive edge at the "bStartAutomatic" input (P-0-1390, bit 0, %IX0.0), the automatic sequence of steps is processed.

```
(*Generating a positive Edge for start*)
fbStartEdge(CLK:= bStartAutomatic , Q=>bStart);

IF (bStart OR bPickerActiv) AND NOT bProgramReset  THEN
    CASE iStep OF

```

DB000283v01_nn

Fig.5-13: Starting the Application "Pick und Place"

4. **Setting drive enable**

   In the first step (step 0), the X- and Y-axes are switched to enable. When the axes are in control, the program jumps to the next step.

```
        0:      (*Enable the X-Axis and Y-Axis*)
                fbMC_PowerXAxis(Enable:=TRUE ,
                                Axis:= Axis1);
                fbMC_PowerYAxis(Enable:=TRUE ,
                                Axis:= Axis2);

                (*Setting State: Picker activ*)
                bPickerActiv := TRUE;

                (*if Picker closed -> next step*)
                IF fbMC_PowerYAxis.Status  THEN
                        iStep := 10;
                END_IF

```

DB000284v01_nn

Fig.5-14: Step 0: Setting Drive Enable

5. **Moving to the picking position**

   In the second step (step 10), it is first the X-axis and then the Y-axis which move to the picking position. When the 1st positioning process of the axes has been completed, the "bPickerCloseCmd" output (P-0-1411, bit 8, %QX1.8) is set. The picker closes and the workpiece is taken up. When the picker has closed, this is signaled by the feedback "bPickerCloseAct" (P-0-1440, bit 1, %IX50.1) and the program switches to the next step.

Double-Axis Positioning Control (Pick and Place)

```
10:     (*Positioning to pick position*)
        (*1st movement of X-Axis*)
        fbMX_MoveAbsoluteXAxis(
                                Execute:= TRUE,
                                Position:= 500*10000,
                                Velocity:= 1000,
                                Acceleration:=1000 ,
                                Deceleration:= 1000,
                                Axis:=Axis1);
        (*1st movement of Y-Axis, if positionig of X-Axis is finished*)
        fbMX_MoveAbsoluteYAxis(
                                Execute:= fbMX_MoveAbsoluteXAxis.Done,
                                Position:= -100*10000,
                                Velocity:= 1000,
                                Acceleration:=1000 ,
                                Deceleration:= 1000,
                                Axis:=Axis2);

        (*if Y-Axis is finished, set Command Close Picker*)
        IF fbMX_MoveAbsoluteYAxis.Done THEN
            bPickerCloseCmd := TRUE;

            (*if Picker closed -> next step; reset motion function blocks*)
            IF bPickerCloseAct THEN
                iStep := 20;

                fbMX_MoveAbsoluteXAxis(Execute:= FALSE,
                                        Axis:= Axis1);
                fbMX_MoveAbsoluteYAxis(Execute:= FALSE,
                                        Axis:= Axis2);
            END_IF
        END_IF
```

DB000285v01_nn

Fig.5-15: Step 10: Moving to the Picking Position

6. **Moving to the placing position**

   In the third step (step 20), the movement to the placing position is carried out. In this case, it is first the Y-axis and then the X-axis which is moved. When the placing position has been reached, the "bPickerOpenCmd" output (P-0-1411, bit 9, %QX1.9) is set. The picker opens and the workpiece is placed. The "bPickerCloseAct" input (P-0-1440, bit 1, %IX50.1) signals that the picker has opened and the program switches to the next step.

```
    20:     (*Positioning to place position*)
            (*Reset Command clsoe Picker*)
            bPickerCloseCmd := FALSE;

            (*2nd movement of Y-Axis*)
            fbMX_MoveAbsoluteYAxis(
                                    Execute:= TRUE,
                                    Position:= -50*10000,
                                    Velocity:= 1000,
                                    Acceleration:=1000 ,
                                    Deceleration:= 1000,
                                    Axis:=Axis2);
            (*2nd movement of X-Axis, if positionig of Y-Axis is finished*)
            fbMX_MoveAbsoluteXAxis(
                                    Execute:= fbMX_MoveAbsoluteYAxis.Done,
                                    Position:= 1000*10000,
                                    Velocity:= 1000,
                                    Acceleration:=1000 ,
                                    Deceleration:= 1000,
                                    Axis:=Axis1);

            (*if X-Axis is finished, set Command Open Picker*)
            IF fbMX_MoveAbsoluteXAxis.Done THEN
                bPickerOpenCmd := TRUE;

                (*if Picker open -> next step; reset motion function blocks*)
                IF NOT bPickerCloseAct  THEN
                    iStep := 30;

                    fbMX_MoveAbsoluteXAxis(Execute:= FALSE,
                                           Axis:= Axis1);
                    fbMX_MoveAbsoluteYAxis(Execute:= FALSE,
                                           Axis:= Axis2);
                END_IF
            END_IF
```

DB000286v01_nn

Fig.5-16: *Step 20: Moving to the Placing Position*

7. **Moving to the start position**

   In the fourth step (step 30), it is first the Y-axis and then the X-axis which move to the start position. When the travel process has been completed, the program switches to the next step.

Double-Axis Positioning Control (Pick and Place)

```
    30:     (*Positioning to start position*)
            (*Reset Command open Picker*)
            bPickerOpenCmd := FALSE;

            (*3rd. movement of Y-Axis*)
            fbMX_MoveAbsoluteYAxis(
                                Execute:= TRUE,
                                Position:= 0,
                                Velocity:= 1000,
                                Acceleration:=1000 ,
                                Deceleration:= 1000,
                                Axis:=Axis2);
            (*3rd movement of X-Axis, if positionig of Y-Axis is finished*)
            fbMX_MoveAbsoluteXAxis(
                                Execute:= fbMX_MoveAbsoluteYAxis.Done,
                                Position:= 0,
                                Velocity:= 1000,
                                Acceleration:=1000 ,
                                Deceleration:= 1000,
                                Axis:=Axis1);

            (*if positionig of X-Axis is finished -> next step; reset motion function blocks*)
            IF fbMX_MoveAbsoluteXAxis.Done  THEN
                iStep := 100;

                fbMX_MoveAbsoluteXAxis(Execute:= FALSE,
                                       Axis:= Axis1);
                fbMX_MoveAbsoluteYAxis(Execute:= FALSE,
                                       Axis:= Axis2);

            END_IF
```

DB000287v01_nn

Fig.5-17: Step 30: Moving to the Start Position

8. **Resetting the sequence of steps**

   In the fifth step (step 100), the "bPickerActiv" signal and the sequence of steps are reset. The sequence of steps must be restarted. Step 40 has been prepared for further functionality and can be included by the corresponding changes in the program. Step 99 has been prepared for an error reaction, but the reaction has not been programmed in this example of application.

```
    40:     (*free step for further functionality*);

    99:     (*Errorhandling*)
            iStep := 100;

    100:    (*Restet State: Picker activ*)
            bPickerActiv := FALSE;

            iStep := 0;

    END_CASE
END_IF
```

DB000288v01_nn

Fig.5-18: Step 100: Resetting the Sequence of Steps

5.3 Commissioning and Testing

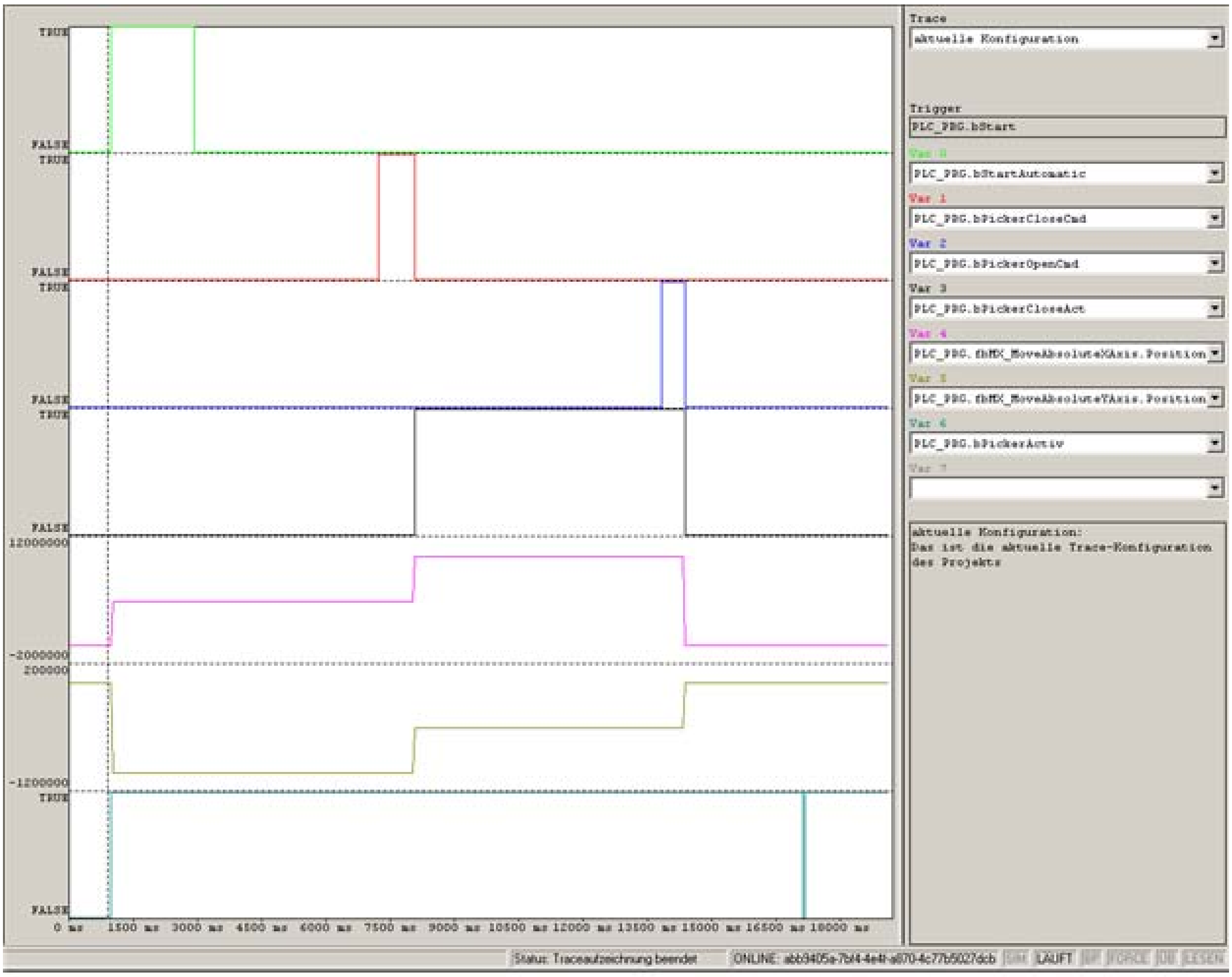
For commissioning and testing, the following steps have to be carried out:

1. Compile program and then load it to drive
2. Start drive PLC
3. Switch both axes to operating mode (OM); clear any present error messages via "Esc" key
4. Switch power on → axes must show the status "Ab"
5. Establish position data reference for both axes (e.g. " set absolute position")
6. Application can be started via input "I_1" at CCD master

5.4 Visualization and Diagnostics

There are different options for visualizing the signals:

- Online display
- IndraLogic trace function
- Oscilloscope function of the drive



DB000289v01_de

Fig.5-19: Sampling Trace via IndraLogic

Double-Axis Positioning Control (Pick and Place)

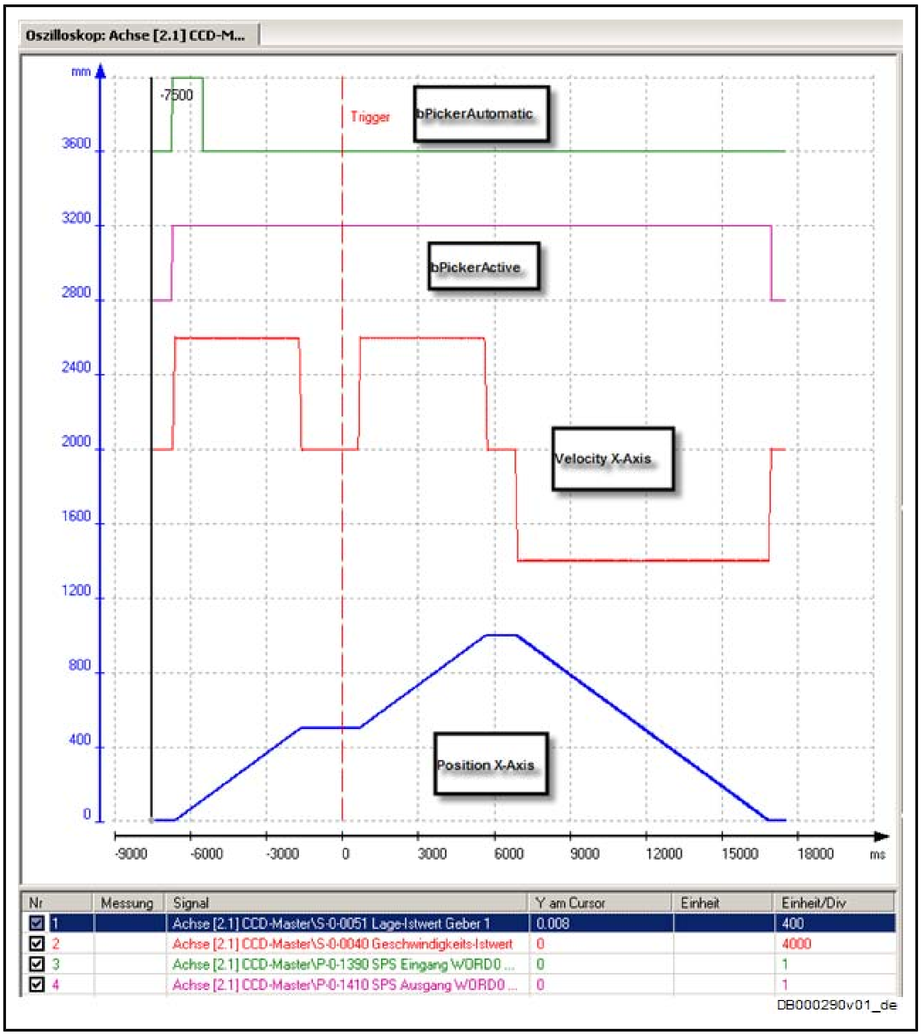


Fig.5-20: Oscilloscope Recording of X-Axis

6 Intelligent Error Reaction

6.1 Task Definition - Application Description

☞ This is an exemplary project for an error reaction by IndraMotion MLD and MPx04 firmware. By means of this example, we will describe the basic options of an MLD error reaction. When used in real machines, the error reaction requires application-specific adjustments!

Task Definition

When an error occurs (F2xxx, F3xxx or F4xxx), an intelligent error reaction is to be carried out by means of IndraMotion MLD. As the standard error reaction "best possible deceleration" is unsuitable, a local error reaction is to take place, particularly when the master communication fails (F4009).

☞ The intelligent error reaction does **not** work when errors of the error classes F8xxx, F7xxx and F6xxx occur! In these cases, it is always the error reaction defined by the drive which is carried out (see section "Error Reactions" in the Functional Description of the IndraDrive firmware).

Function

The figure below contains an overview of the functional structure of MLD's application example "intelligent error reaction".

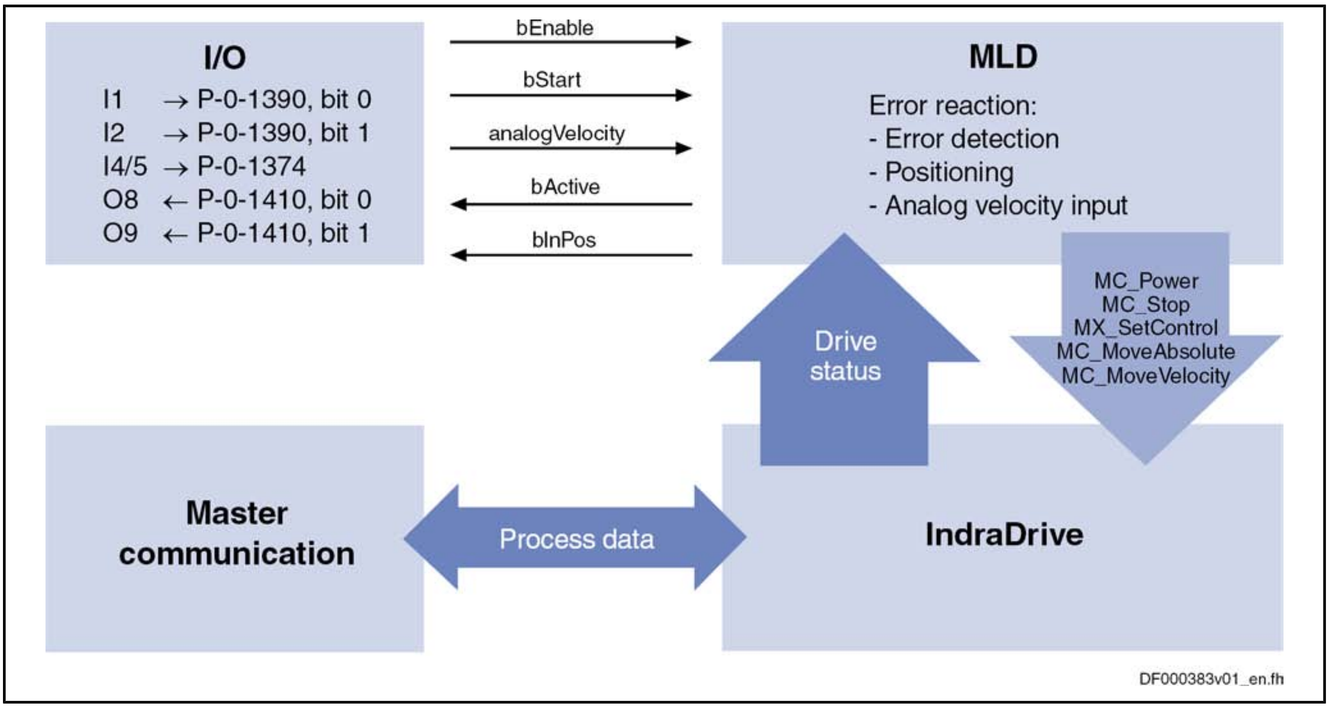


Fig.6-1: Structural Overview of the Application Example

The intelligent error reaction is characterized by the following features:

- The error reaction can be parameterized or controlled via global registers (P-0-1370 ff.), as well as analog and digital inputs at the control section.
- Via an analog input, a velocity command value can be preset with which the drive continues moving after an error has occurred (e.g. for stirring machines).

- As an alternative, the drive moves to a defined absolute position, when a digital input is set.
- Removing the Enable signal at the digital input terminates/aborts the error reaction of the drive.
- When the cause of the error is removed within 30 s after the error has occurred, and the error message is cleared, the reaction can take longer than 30 s (e.g. analog velocity input). Otherwise, the drive aborts the MLD error reaction after 30 s by means of best possible deceleration when an error is present.

Notes on Utilization

Observe the following aspects when using the MLD application "intelligent error reaction":

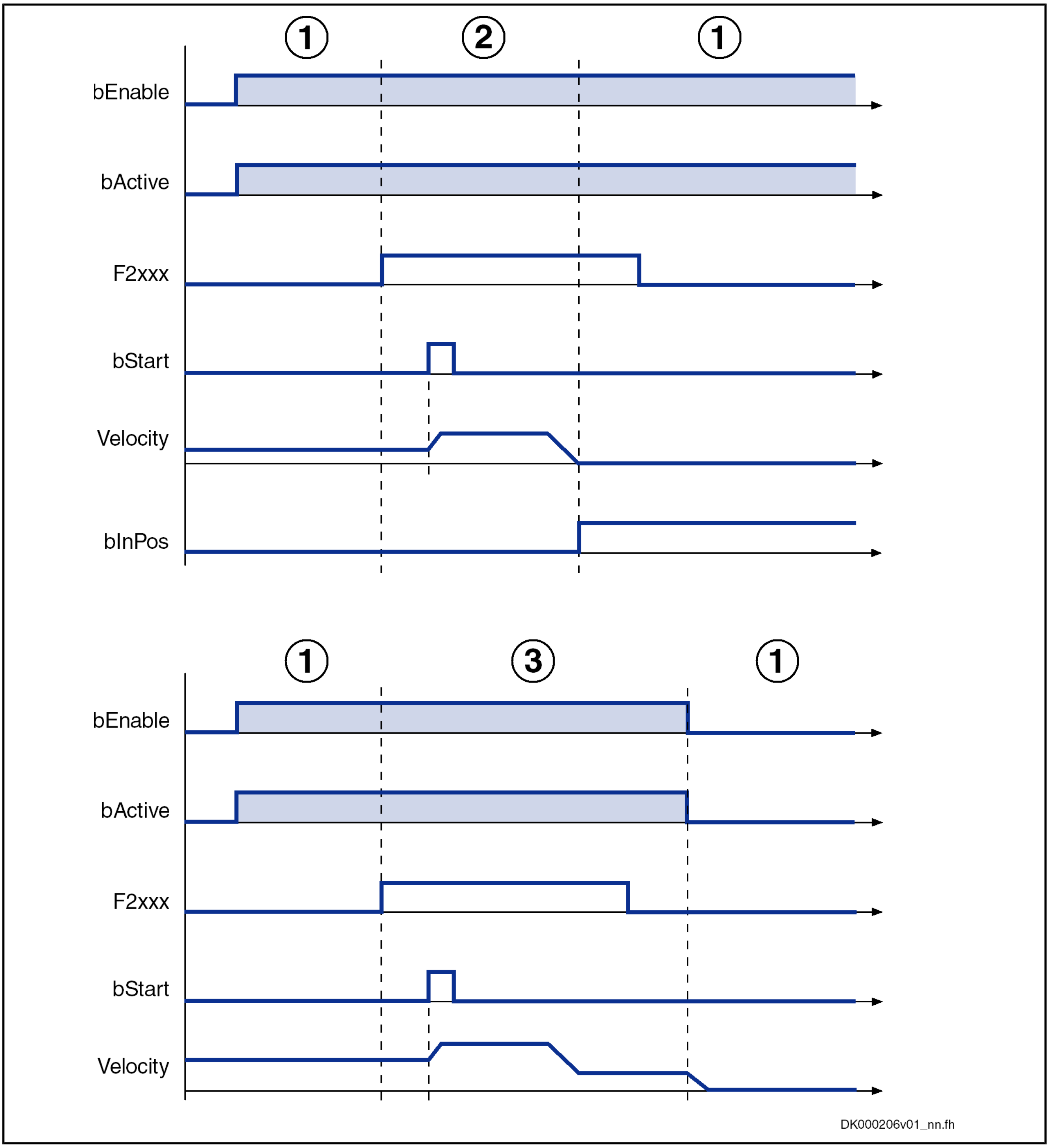
- During the entire error reaction of the drive, MLD has control over the axis. Only upon abortion/end of reaction is control ceded to the external control unit (via master communication).
- In order that MLD can react to an error with the corresponding reaction/ motion, the respective error reaction must have been set in the drive (see below).

☞ The error reaction described in this application example may only be used with the MPx04 firmware.

As of firmware version MPx05, the "easy startup mode" is available for the local emergency mode in case master communication fails (see "Setting-Up Mode (Easy Startup Mode)" and "Local Mode" in the section "Operating Modes of Master Communication" of the Functional Description MPx05).

Sequence of the Error Reaction

The chronological diagram below illustrates the sequence of motion of the intelligent error reaction.



1 Error detection
2 Positioning
3 Analog velocity input
Fig.6-2: Sequence of Motion of the Application "Intelligent Error Reaction"

Step 1: Error detection

As soon as the function has been activated via the Enable signal "bEnable" (P-0-1390, bit 0, %IX0.0), the feedback takes place with "bActive" (P-0-1410, bit 0, %QX0.0). Afterwards, permanent monitoring checks whether a new drive error has occurred and whether a reaction to an occurred error is to take place. According to the settings, the parameterized reaction is carried out. The setting as to which error class the reaction is to take place is made in the parameter P-0-1370, bit 1 to 3 (bit 1: reaction to F2xxx; bit 2: reaction to F3xxx; bit 3: reaction to F4xxx). In bit 0, set the type of reaction, that is the moving to a return position or an analog velocity input.

Step 2: Positioning

If the moving to a preset return position is to be the reaction to the error which is present, the program waits for a rising edge of "bStart" (P-0-1390, bit 1, %IX0.1). After the rising edge has been detected, MLD takes control and the drive moves to the position set in P-0-1372. When the position has been reached, this is signaled with "bInPos" (P-0-1410, bit 1, %QX0.1). Afterwards, control is ceded to the master communication and the error reaction has been completed.

Step 3: Analog velocity input

If the error reaction is to be such that the axis follows an analog velocity command value, the program waits for a rising edge of "bStart" (P-0-1390, bit 1, %IX0.1). After the rising edge has been detected, MLD takes control and the drive follows the velocity value preset in parameter P-0-1374 via the analog input. In the case of a negative edge of the Enable signal "bEnable" (P-0-1390, bit 0, %IX0.0), the drive no longer follows the analog command value input and is stopped. Control is ceded to the master communication and the error reaction has been completed.

6.2 Parameterizing/Configuring the Drive

Starting from the basic parameters, you have to make some fundamental settings for the example of application "intelligent error reaction"; the following paragraphs will only give a brief explanation of these fundamental settings.

☞ The global registers used for parameterizing the error reaction mustn't have been configured in the cyclic data, because otherwise these parameters are set to zero when the master communication fails.

Error Reaction

Parameter setting of the error reaction of the drive. This is necessary so that the drive PLC can carry out the corresponding reaction/motion in spite of the drive error being present.

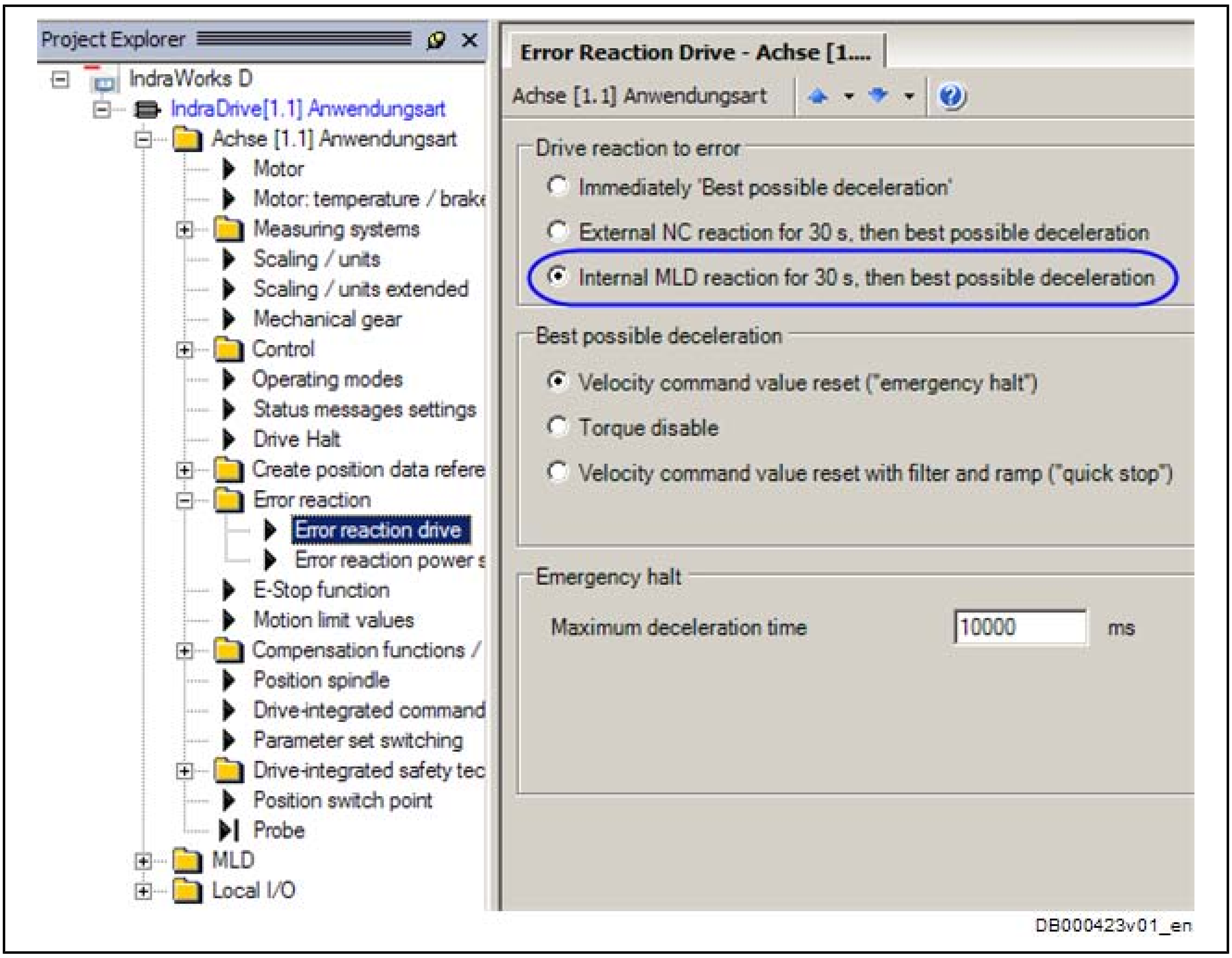


Fig.6-3: IndraWorks Dialog for Parameterizing the Error Reaction

Mechanical Data

According to the mechanical configuration, you have to set the scaling, gear and feed constant.

Intelligent Error Reaction

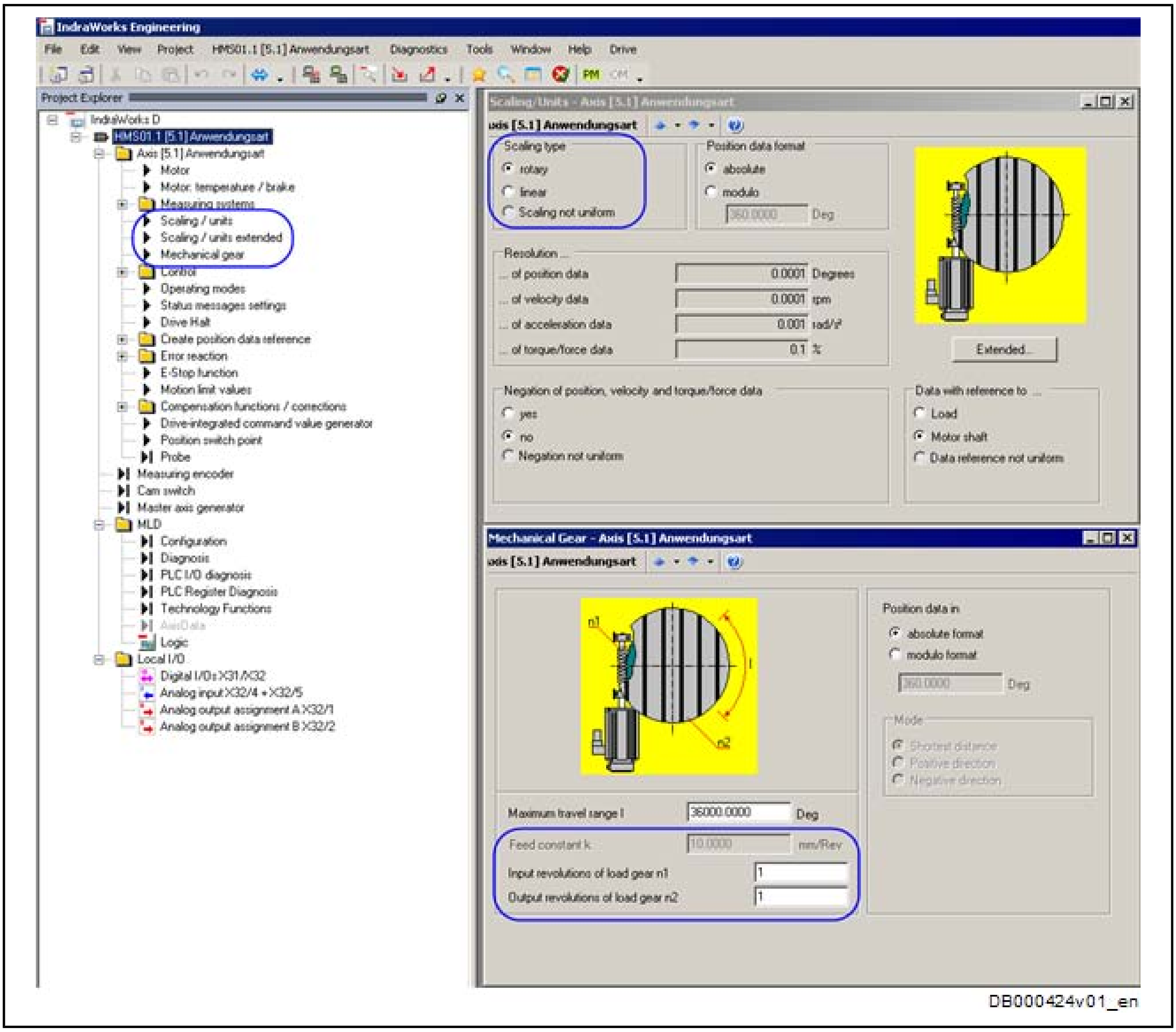


Fig.6-4: IndraWorks Dialog for Setting the Mechanical Data (Example)

MLD Configuration

The drive PLC does not have permanent control. Motion and control in "normal operation" take place via the master communication. The drive PLC takes over control only for the error reaction.

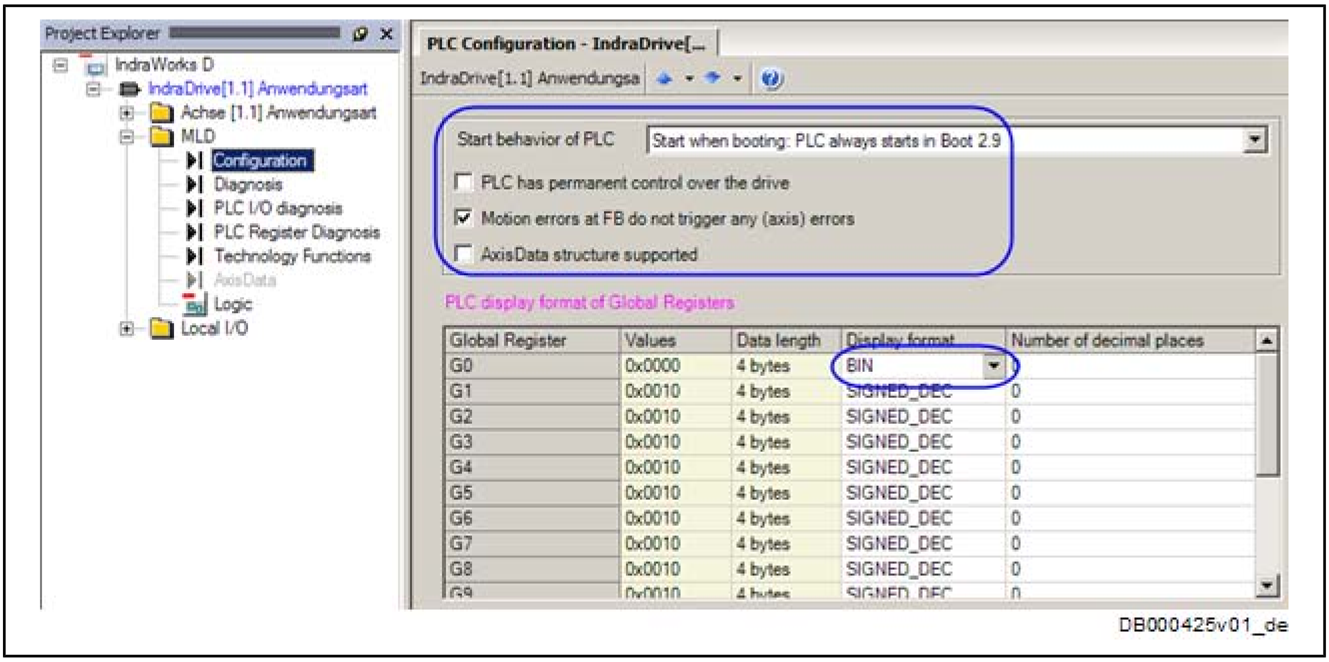


Fig.6-5: IndraWorks Dialog for MLD Configuration

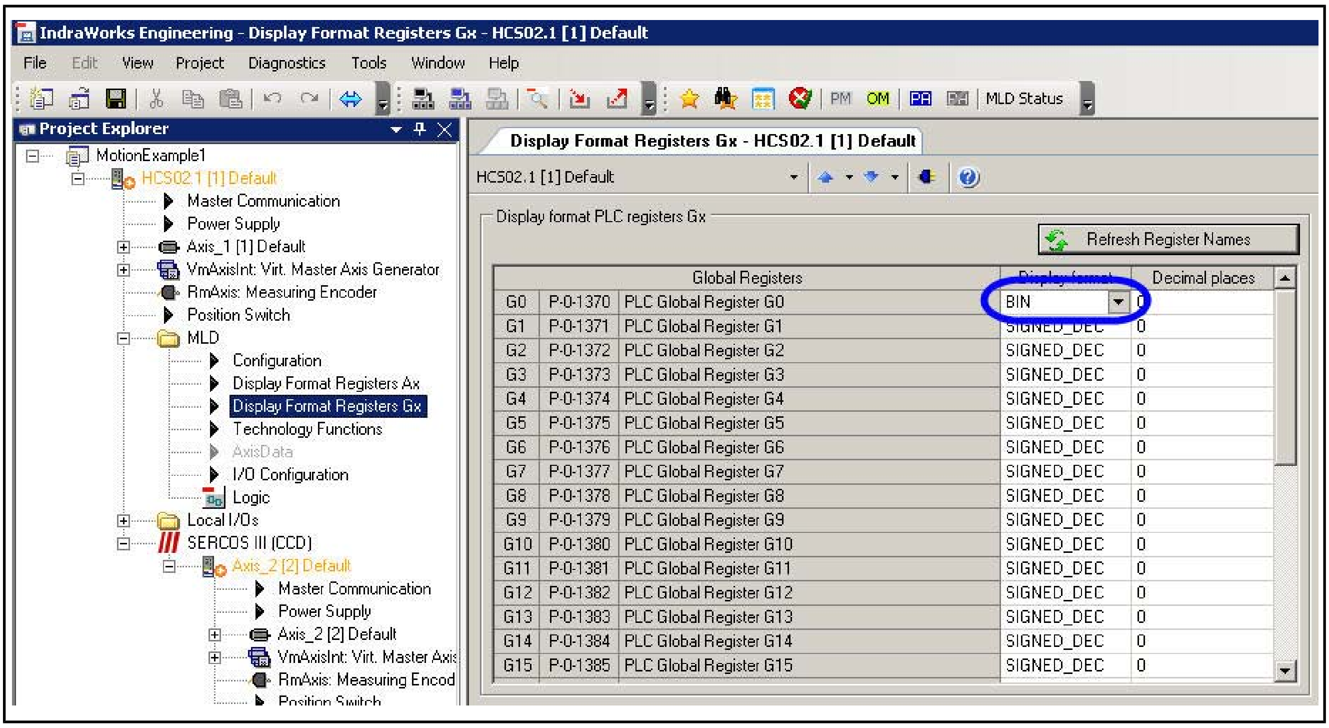
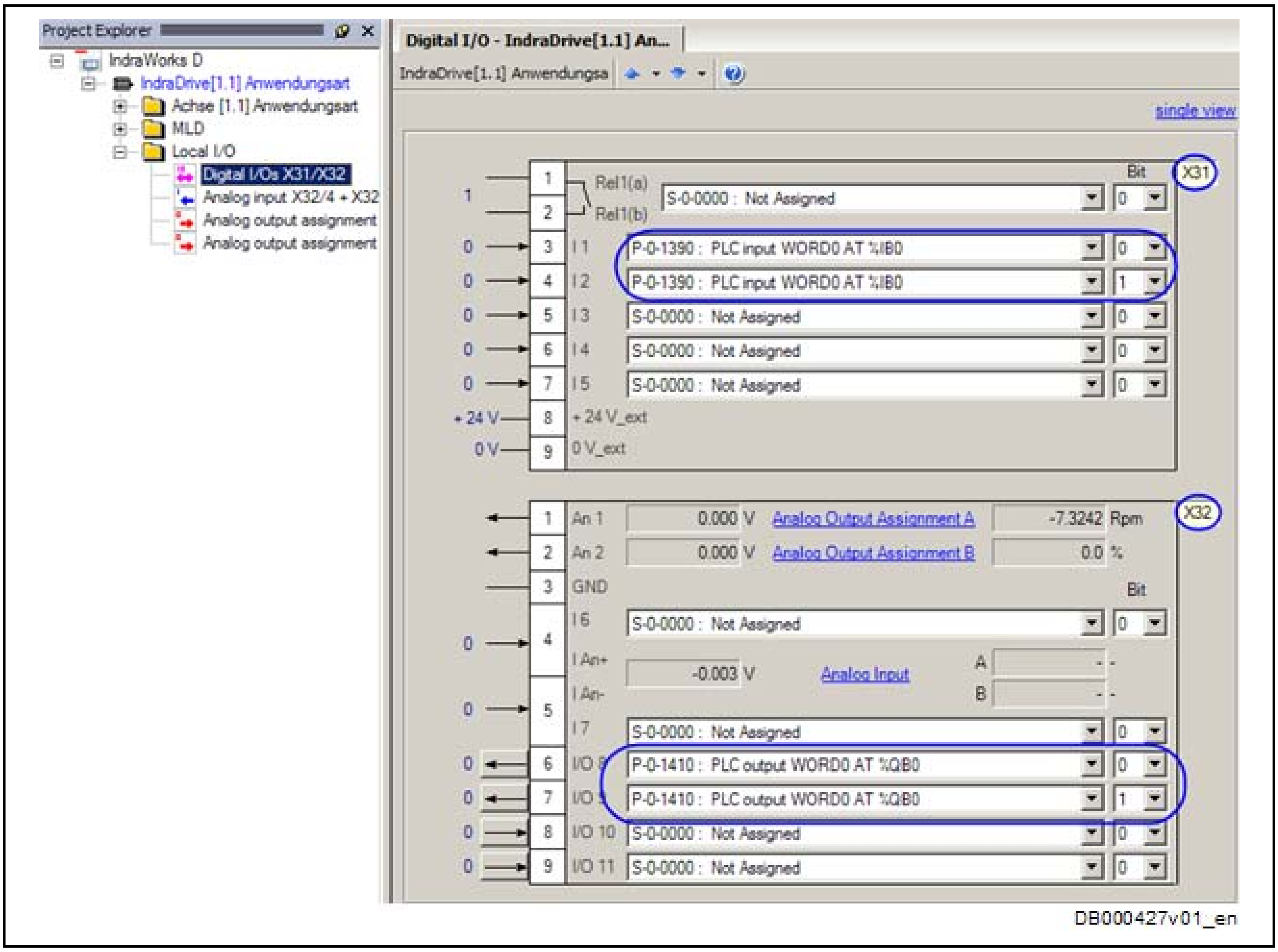


Fig.6-6: MLD Configuration Display Format "Register Gx"

Configuring the Digital I/Os

The screenshot below shows the IndraWorks dialog for setting the digital inputs/outputs (X31/X32).

Intelligent Error Reaction



X31.3	P-0-1390, bit 0 (%IX0.0) bEnable
X31.4	P-0-1390, bit 1 (%IX0.1) bStart
X32.6	P-0-1410, bit 0 (%QX0.0) bActive
X32.7	P-0-1410, bit 0 (%QX0.1) bInPos

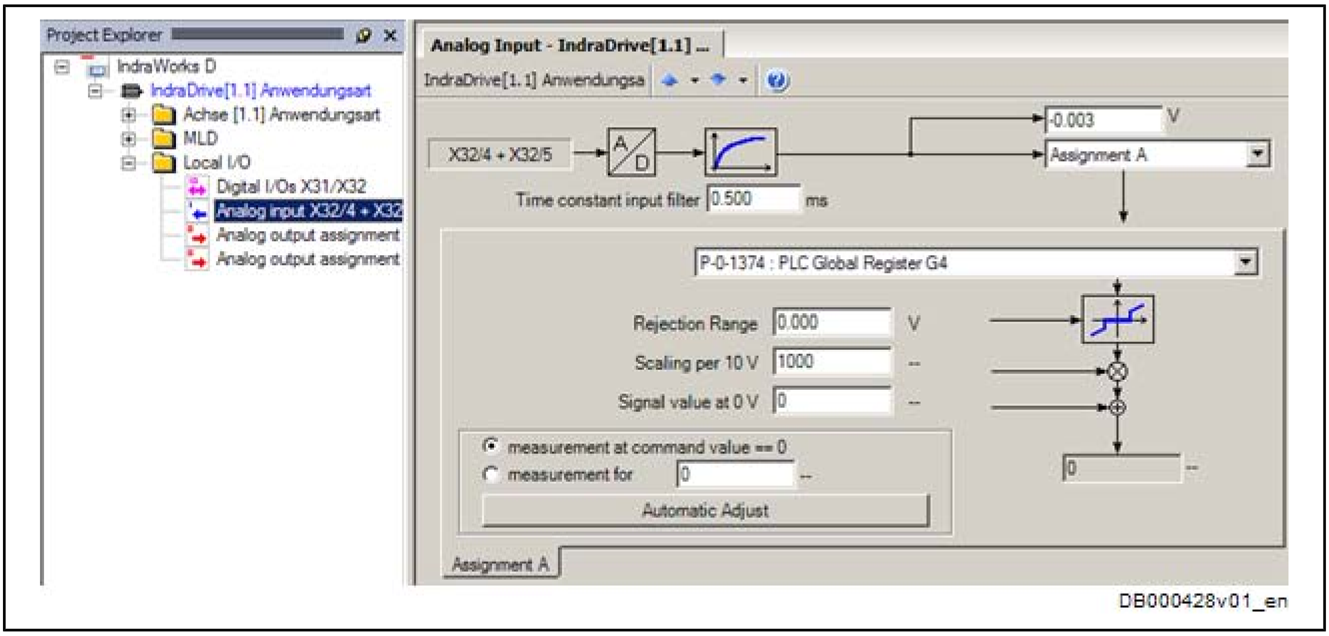
Fig.6-7: IndraWorks Dialog for Configuring the Digital Inputs/Outputs X31/X32

The states of the digital inputs are evaluated by assigning the bits of P-0-1390 directly in MLD.

In the application example, the parameter P-0-1410 is written by the drive PLC. By the assignment shown above, this parameter directly takes effect on the digital outputs.

Configuring the Analog Input

The screenshot below shows the IndraWorks dialog for setting the analog input.



X32.4/ X32.5 P-0-1374

Fig.6-8: IndraWorks Dialog for Configuring the Analog Input

Via the analog input, an analog velocity command value is input in the global register of the drive PLC (P-0-1374).

6.3 Programming

1. **Variable declaration**

   In the variable declaration, the variables which are used are created and assigned to the inputs and outputs.

```
PROGRAM errorreaction
VAR
    bEnable   AT%IX0.0 :  BOOL; (*P-0-1390, Bit 0: enable/abort*)
    bStart    AT%IX0.1 :  BOOL; (*P-0-1390: Bit 1: start reaction*)
    bActive   AT%QX0.0:   BOOL; (*P-0-1410, Bit 0: monitoring activ*)
    bInPos    AT%QX0.1:   BOOL; (*P-0-1410, Bit 1: inposition*)

    iStep: INT;
    diDiagnosticNumber: DINT;
    dwErrornumber: DWORD;

    fbR_TRIG: R_TRIG;
    fbMX_SetControl: MX_SetControl;
    fbMC_Stop: MC_Stop;
    fbMC_Power: MC_Power;
    fbMC_MoveAbsolute: MC_MoveAbsolute;
    fbMC_MoveVelocity: MC_MoveVelocity;
END_VAR
                                                    DB000414v01_nn
```

Fig.6-9: Variable Declaration

2. **Error detection**

   When the function for error reaction has been activated via "bEnable" (P-0-1390, bit 0, %IX0.0), "bActive" is set (P-0-1410, bit 0, %QX0.0) in the first step (Step 0). If a new drive error is then detected (P-0-0115, 0 → 1 in bit 13 and S-0-0390 → F2xxx/F3xxx/F4xxx) to which a reaction is to take place (P-0-1370, bit 1 to 3), the function blocks required for the reaction and "bInPos" (P-0-1410, bit 1, %QX0.1) are reset and the program jumps to the next step.

```
CASE iStep OF
0:  (*monitoring*)
    IF bEnable THEN (*error reaction enabled*)
        bActive := TRUE;   (*status: functionality enabled*)
        fbR_TRIG(CLK:=EXTRACT(DINT_TO_DWORD (DV_P_0_0115),13) , Q=>);   (*edgedetection on errorbit in P-0-0115, Bit 13*)
        dwErrornumber :=DINT_TO_DWORD(DV_S_0_0390) AND 16#FF000;    (*detection of errornumber Fx*)
        IF   fbR_TRIG.Q AND (
                (dwErrornumber = 16#F2000) AND EXTRACT(DINT_TO_DWORD (DV_P_0_1370),1)    (*F2xxx active, F2xxx reaction configured*)
            OR (dwErrornumber = 16#F3000) AND EXTRACT(DINT_TO_DWORD (DV_P_0_1370),2)    (*F3xxx active, F3xxx reaction configured*)
            OR (dwErrornumber = 16#F4000) AND EXTRACT(DINT_TO_DWORD (DV_P_0_1370),3))   (*F4xxx active, F4xxx reaction configured*)
        THEN
            bInPos := FALSE;
            fbMX_SetControl(Execute:= FALSE,Axis:=axis1);(*reset FB*)
            fbMC_Power(Enable:=FALSE ,Axis:= axis1); (*reset FB*)
            fbMC_Stop(Execute:=FALSE ,Axis:=axis1);   (*reset FB*)
            fbMC_MoveAbsolute(Execute:= FALSE,Axis:=axis1 );    (*reset FB*)
            iStep := 10;    (*next step: drive control to drivePLC*)
        END_IF
    ELSE
        bActive := FALSE;  (*Error reaction disabled*)
    END_IF
                                                    DB000415v01_nn
```

Fig.6-10: Step 0: Error Detection

3. **Taking control to the drive PLC**

   As long as "bEnable" has been set, the program in step 2 (Step 10) waits for a rising edge of "bStart" (P-0-1390, bit 1, %IX0.1). After the edge has been detected, MLD gets control. As soon as IndraMotion MLD has control, the program jumps to the step with the corresponding reaction, depending on the configured type of reaction (P-0-1370, bit 0). When "bEnable" was reset, the error reaction is completed in Step 100 (see below).

```
10: (*drive control to drivePLC*)
    IF bEnable  THEN (*error reaction enabled*)
        fbR_TRIG(CLK:=bStart , Q=>);    (*edgedetection for start of reaction*)
        IF fbR_TRIG.Q OR fbMX_SetControl.Execute THEN
            fbMX_SetControl(Execute:= TRUE,PLCControl:= TRUE,Axis:=axis1);   (*drive control to drivePLC*)
        END_IF
        IF fbMX_SetControl.PLCControlStat THEN    (*drive control in drive PLC*)
            IF EXTRACT(DINT_TO_DWORD (DV_P_0_1370),0) THEN (*kind of reaction*)
                iStep := 20;    (*next Step: return motion*)
            ELSE
                iStep := 30;    (*next Step: constant velocity*)
            END_IF
        END_IF
    ELSE   (*error reaction disabled*)
        iStep := 100;   (*next step: error reaction disabled/finished*)
    END_IF
                                                                     DB000416v01_nn
```

Fig.6-11: Step 10: Setting Drive Enable

4. **Positioning**

   As long as "bEnable" has been set, the return motion is started or processed in step 3 (Step 20). When the return position has been reached, i.e. when "bInPos" (P-0-1410, bit 1, %QX0.1) has been set or "bEnable" was reset, the error reaction is completed in Step 100 (see below).

```
20: (*return motion*)
    IF bEnable AND NOT bInPos THEN  (*error reaction enabled and position not reached*)
        fbMC_MoveAbsolute(
            Execute:= TRUE,
            Position:= DINT_TO_REAL(DV_P_0_1372),
            Velocity:= 1000,
            Acceleration:=1000 ,
            Deceleration:= 1000,
            Axis:=axis1,
            Done=> bInPos ); (*return motion*)
    ELSE  (*error reaction disabled or position reached*)
        iStep := 100;   (*next step: error reaction disabled/finished*)
    END_IF
                                                         DB000417v01_nn.tif
```

Fig.6-12: Step 20: Positioning

5. **Analog velocity input**

Intelligent Error Reaction

As long as "bEnable" has been set, the drive moves in step 4 (Step 30) with command velocity preset via the analog input and parameter P-0-1374. When "bEnable" is reset, the error reaction is completed in Step 100 (see below).

```
30: (*constant velocity*)
    IF bEnable THEN (*error reaction enabled*)
        fbMC_MoveVelocity(
            Execute:= TRUE,
            Velocity:= DINT_TO_REAL(DV_P_0_1374),
            Acceleration:=1000 ,
            Deceleration:= 1000,
            Axis:= Axis1 ); (*constant velocity*)
    ELSE (*error reaction disabled*)
        iStep := 100;   (*next step: error reaction disabled/finished*)
    END_IF
```

DB000418v01_nn

Fig.6-13: Step 30: Analog Velocity Input

6. **Error reaction aborted/completed**

   In the fifth step (Step 100), "bActive" is set to the value of "bEnable", "bTriggered" is reset, the axis is stopped, the enable signal of the drive is removed and the drive PLC cedes control. When control is no longer in the drive PLC, the program jumps back to the monitoring step (Step 0). Step 40 has been prepared for further functionality and can be included by the corresponding changes in the program. "Step 99" has been prepared for an additional error reaction, but it has not been programmed in this example of application.

```
40: (*free step for further functionality*)

99: (*FB-error reaction*)

100: (*error reaction disabled/finished*)
    bActive := bEnable;    (*state of enable signal*)
    bTriggered := FALSE;
    fbMC_Stop(Execute:=TRUE ,Deceleration:= 1000,Axis:=axis1); (*stop axis *)
    fbMC_Power(Enable:=NOT fbMC_Stop.Done,Axis:= axis1); (*power off*)
    fbMX_SetControl(Execute:= FALSE,PLCControl:= FALSE,Axis:=axis1);
    fbMX_SetControl(Execute:= NOT fbMC_Power.Status,PLCControl:= FALSE,Axis:=axis1);(*drive control to master communication*)
    IF NOT fbMX_SetControl.PLCControlStat    THEN
        iStep := 0; (*next step: monitoring*)
    END_IF

END_CASE;
```

DB000419v01_nn

Fig.6-14: Step 100: Error Reaction Aborted/Completed

6.4 Commissioning and Testing

For commissioning and testing, the following steps have to be carried out:

1. Compile program and then load it to drive
2. Start drive PLC
3. Switch axis to operating mode (OM); clear any present errors via "Esc" key
4. Switch power and drive enable on → axes must show the status "AF"
5. Establish position data reference (e.g. set absolute position)
6. Configure function:
   - Type of reaction (P-0-1370, bit 0); depends on type of error (P-0-1370, bit 1-3)
   - Return position (P-0-1374)
   - In analog form, preset velocity via analog input I An+, I An- (X32/4, X32/5)
7. Function can be enabled via input "I1" (X31/3).
8. Generate an error (e.g. F2021)
9. Start error reaction via input "I2" (X31/4)

6.5 Visualization and Diagnostics

There are different options for visualizing the signals:

- Online display
- IndraLogic trace function
- Oscilloscope function of the drive

Intelligent Error Reaction

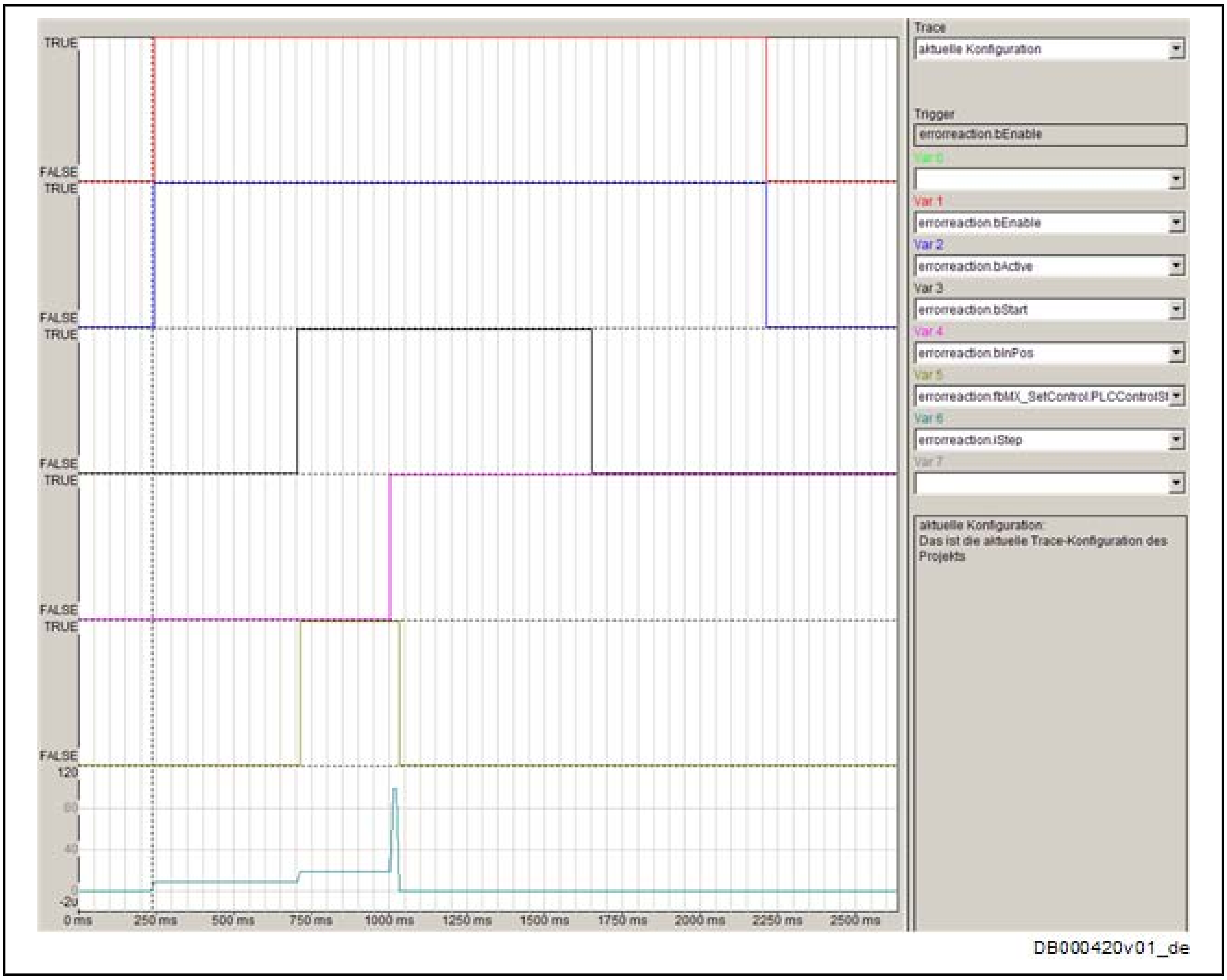


Fig.6-15: Sampling Trace via IndraLogic

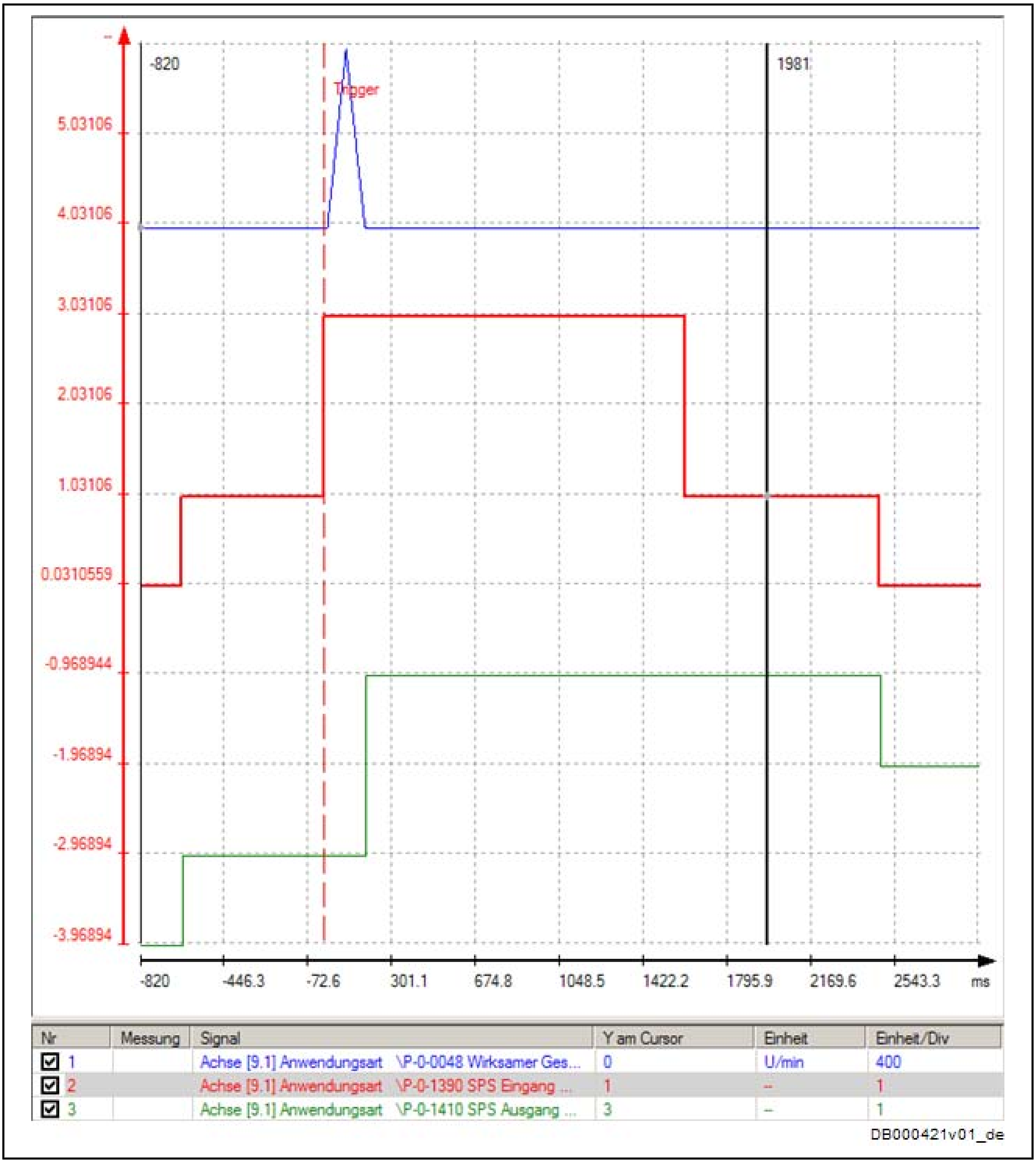


Fig.6-16: Oscilloscope Recording of the Axis

7 Synchronous Multi-Axis Motion With Virtual Master Axis

7.1 Task Definition – Application Description

7.1.1 General Information

This part is based on the application example 1 "double-axis positioning control (Pick and Place)". The following application example shows mechanisms of how to realize synchronous multi-axis motions with IndraMotion MLD-M via the CCD communication.

The example below illustrates master axis linking between virtual master axis and real axes.

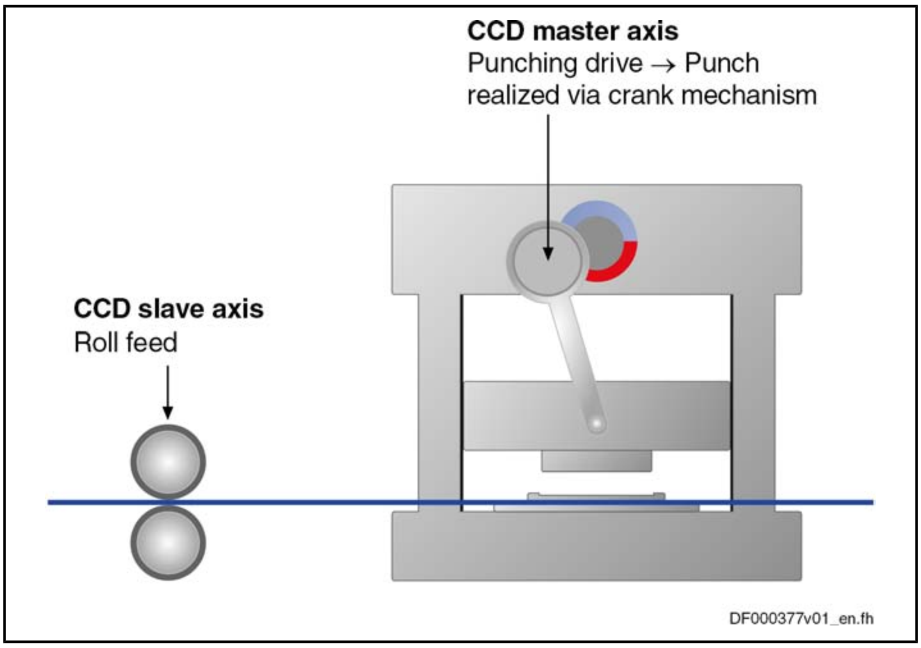


Fig.7-1: Simple Punching Machine With Roll Feed

Shaped pieces are to be punched out of a sheet metal strip. One drive (CCD slave axis) carries out material feed and a second drive (CCD master axis) moves the punching head via a crank mechanism. Material feed mustn't take place during the actual punching operation. Only when the punching head has left the material may the material be infed. MLD-M of the CCD master axis controls or commands the axes.

7.1.2 Sequence of Motion

To avoid the collision of punching head and material when the operation modes are activated, the two axes are aligned with one another.

The sequence of motion starts with the punching drive being moved to the position at which material feed is to start. For this purpose, the operation mode "phase synchronization" is activated via the synchronous motion function block "MB_GearInPos" in the punching axis (CCD master axis). Simultaneously, the virtual master axis (the virtual master axis is the master axis which the punching drive and the roll drive follow synchronously) is moved to the start position of the material feed (0 degrees) via the motion function block "MC_MoveAbsolute". When the start position has been reached and the punching drive has absolutely synchronized to the master axis, the operation mode "electronic motion profile" is activated for the roll feed axis (CCD slave axis) via the synchronous motion function block "MB_MotionProfile". When the roll feed axis has relatively synchronized to the master axis, the virtual master axis is continuously moved via the function block "MC_MoveVelocity".

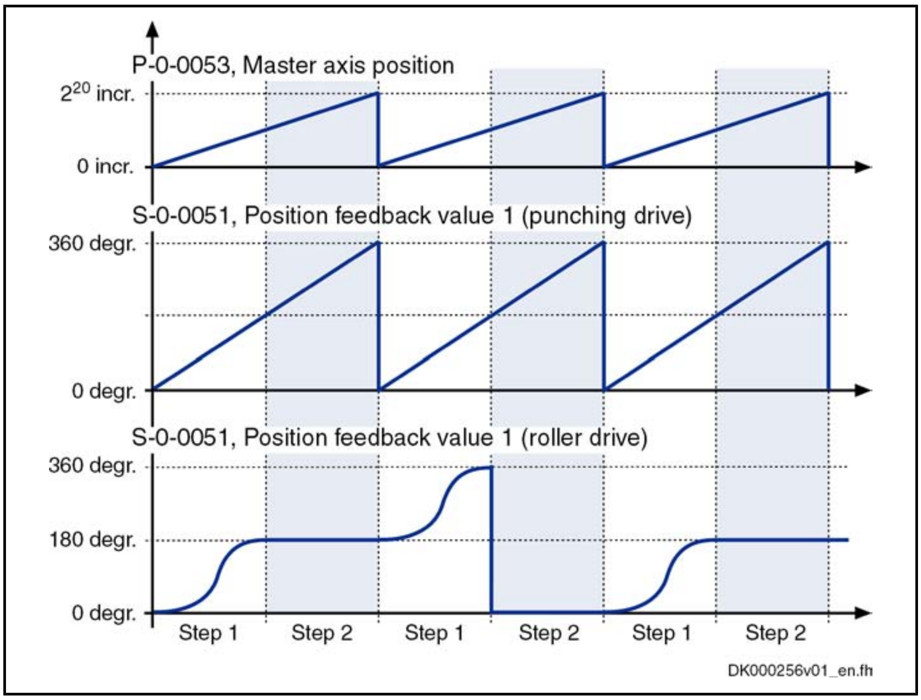
The cyclic sequence of motion consists of two motion steps:

- **Motion step 1**

  Within this motion step, material feed from the master axis position "0 degrees" (master axis start position for material feed) to "180 degrees" (master axis end position for material feed) takes place.

- **Motion step 2**

  The second motion step defines the punching range. Within this range, material feed mustn't take place. The punching range reaches from the master axis end position for material feed (180 degrees) to the master axis start position for material feed (0 degrees).



Step 1	Material feed takes place from master axis position "0 increments" to position "524288 increments"
Step 2	Punching takes place within this range

Fig.7-2: Sequence of Motion

7.2 Parameterizing/Configuring the Drive

7.2.1 Overview

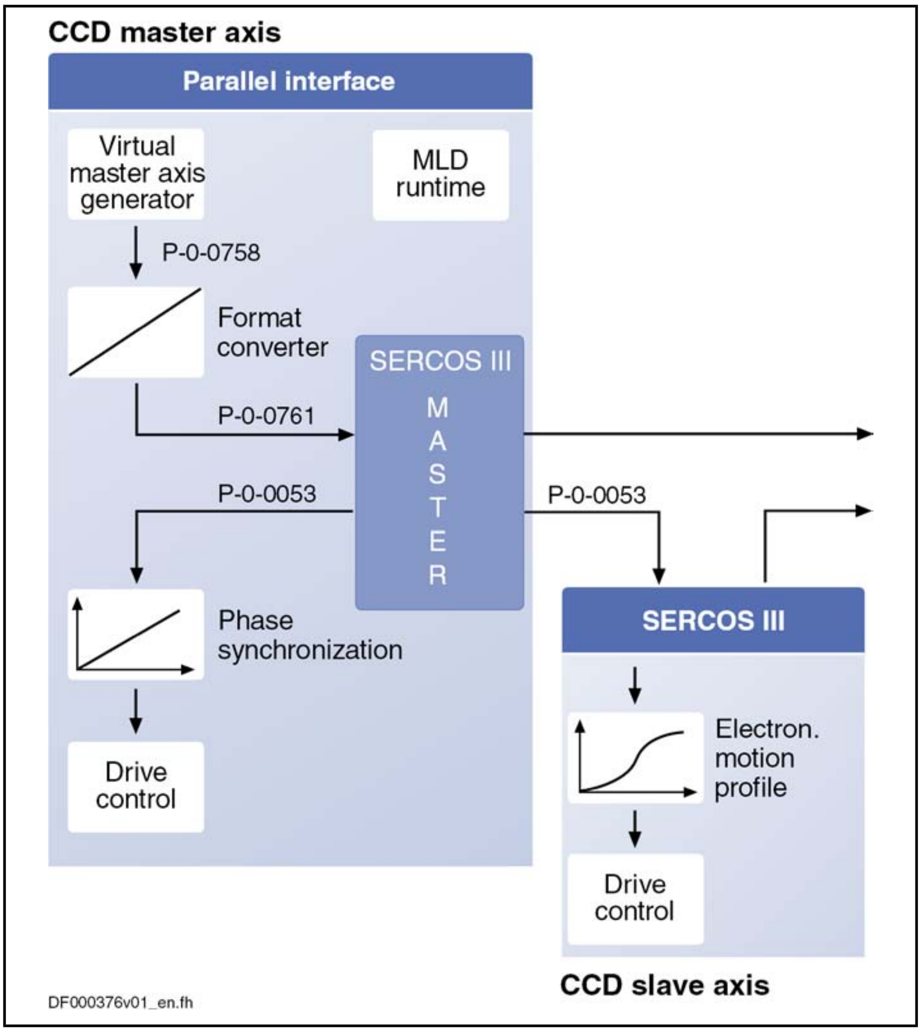


Fig. 7-3: Configuring the Application Example

Starting from basic parameters in the CCD master axis, you have to make some fundamental settings for the example of application "synchronous multi-axis motion with virtual master axis". These settings are described below.

Synchronous Multi-Axis Motion With Virtual Master Axis

7.2.2 CCD Master Axis

Enabling of Functional Packages

This application example additionally requires the base package "Closed-Loop" and the enabling of the optional functional package "Synchronization".

☞ It is only allowed to enable licensed functional packages!

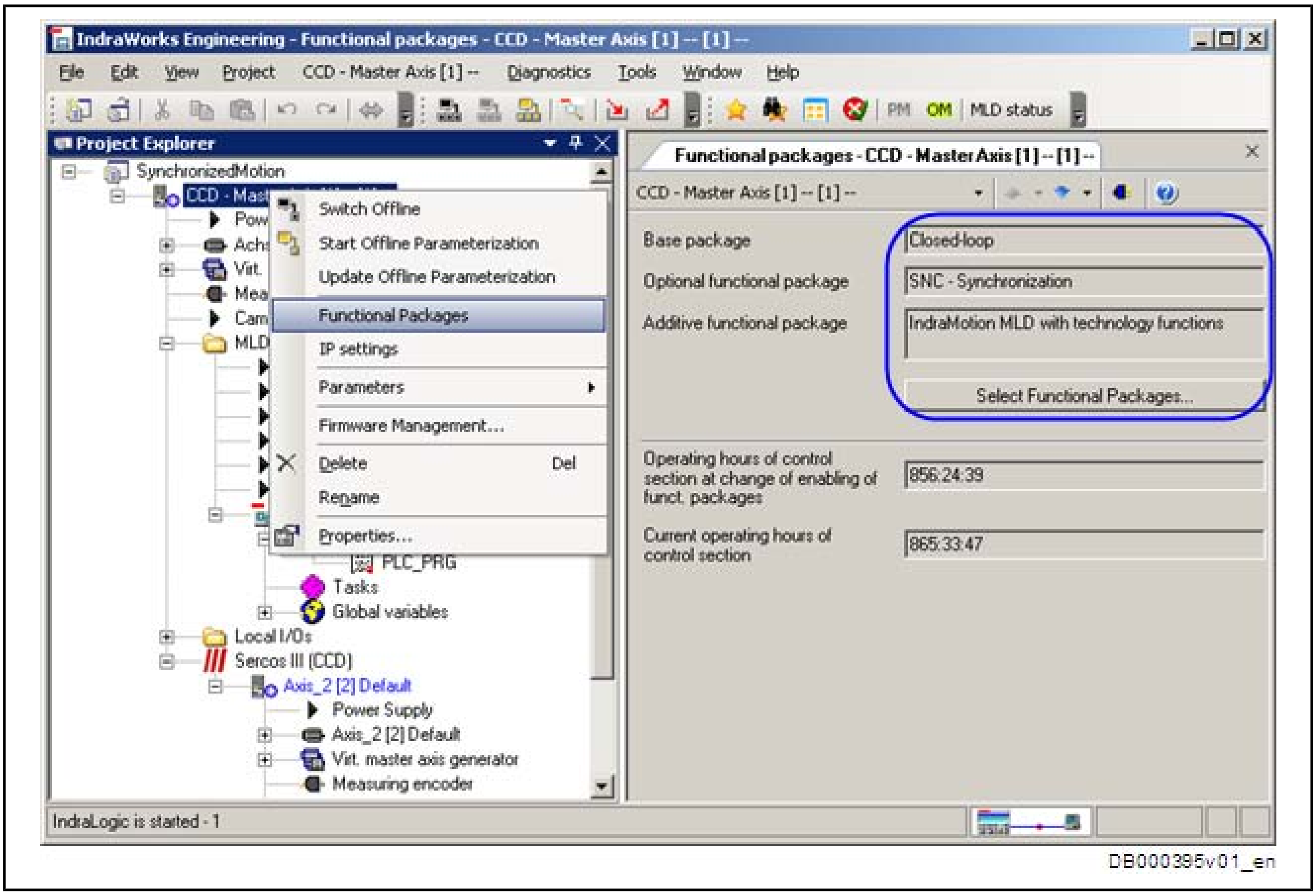


Fig.7-4: IndraWorks Dialog to Enable the Required Functional Packages for the CCD Master Axis

Scaling Settings

The screenshot below shows the IndraWorks dialog for setting the scaling.

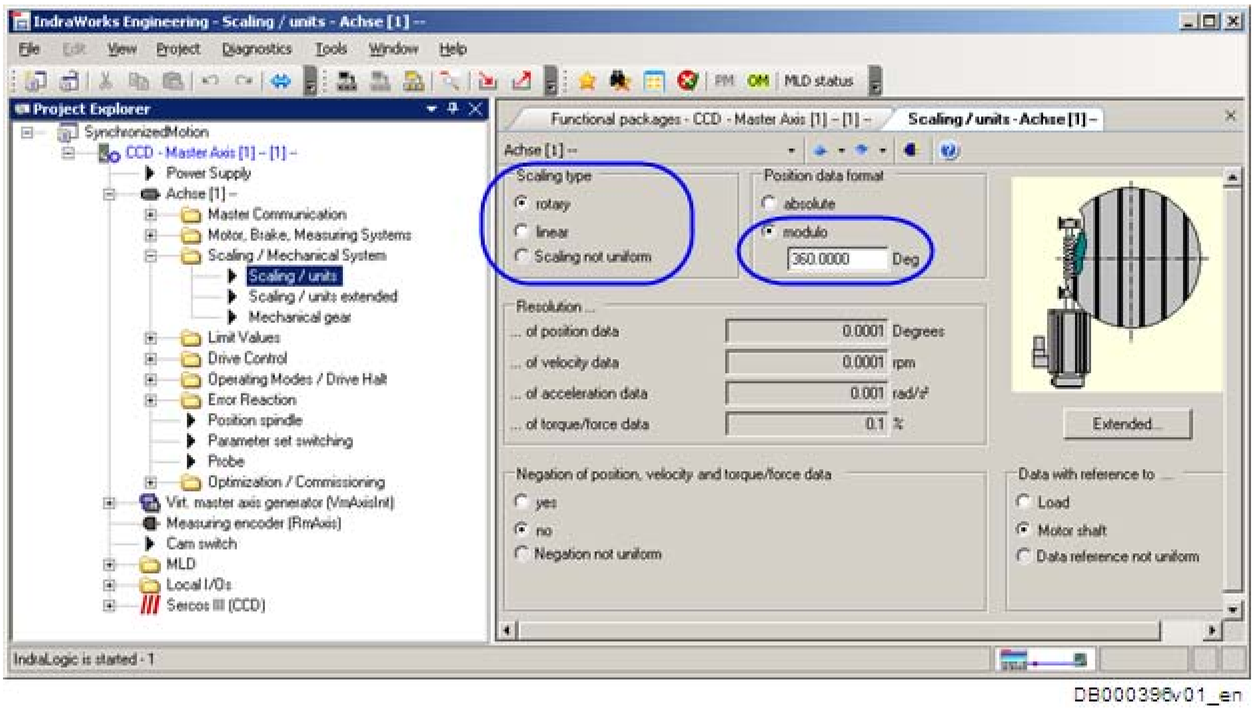


Fig.7-5: IndraWorks Dialog of the Scaling Settings for the CCD Master Axis

CCD Configuration

Via the CCD communication (SERCOS III), MLD in the CCD master axis can command the CCD master axis itself and up to 7 other CCD slave axes in the MLD-M system mode.

With the **MPx04 firmware**, you must carry out the required configuration as follows:

- In the Parameter "P-0-1601, CCD: Addresses of projected drives", enter the addresses of the CCD slaves projected in the CCD group. For the address of the CCD slave axis, see parameter "P-0-4025, Drive address of master communication" of the corresponding CCD slave axis (or drive display).
- Cross communication (CCD) is activated by selecting "Cross Communication Drive active" in the IndraWorks dialog (see figure below). In addition, set the option "MLD-M in CCD master". The field "Available slaves" lists the addresses from the parameter "P-0-1601". If not yet entered, apply them to the field "Projected slaves" and confirm with "Apply".

Synchronous Multi-Axis Motion With Virtual Master Axis

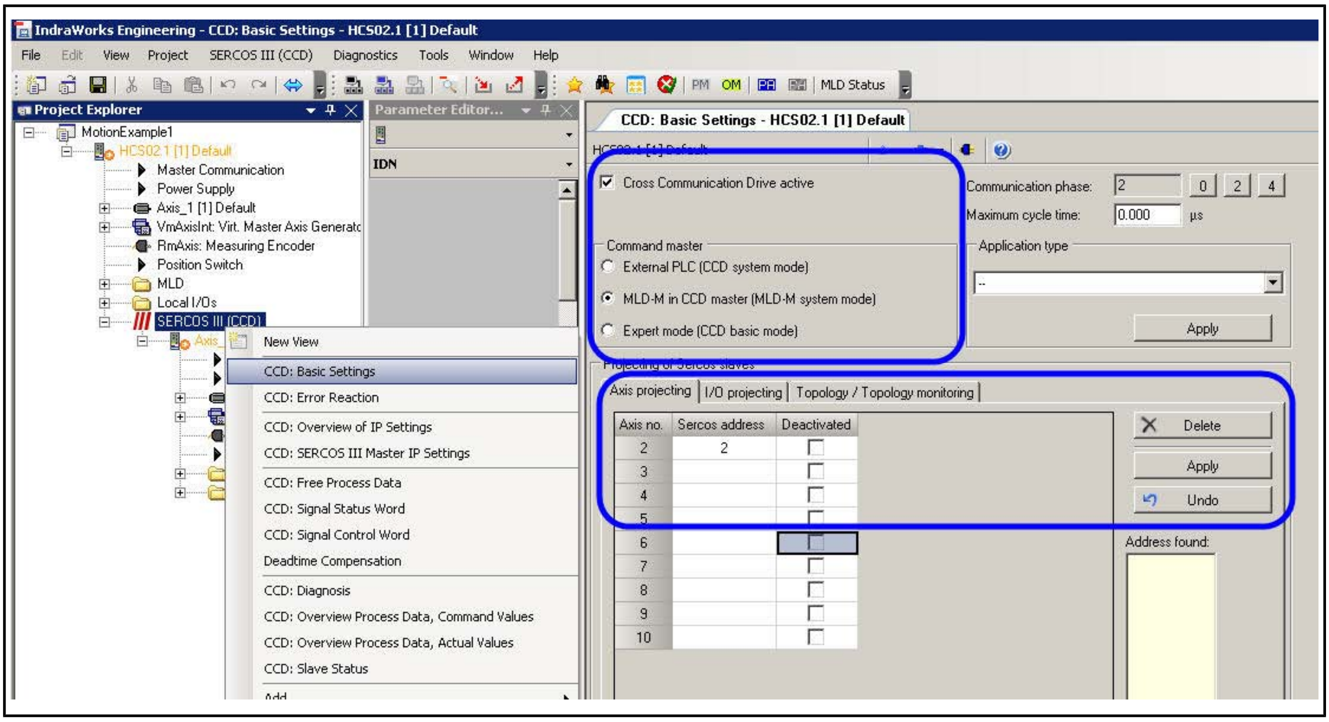


Fig.7-6: IndraWorks Dialog of the Basic CCD Settings With MPx04 Firmware

With the **MPx05 firmware**, you must carry out the required configuration via the corresponding window of the IndraWorks dialog (see screenshot):

1. In the field "Command master", select the option "MLD-M in CCD master".
2. Activate the CCD communication by ticking the check box "Cross Communication Drive active".

   → The automatic determination of the available slave address is carried out (function "remote address assignment") and the results are entered in the table "Projecting of Sercos slaves".

   If you would not like to make any changes, confirm the entries found with "Apply"!
3. The automatically determined addresses can be changed in the field "Proj. addr." and assigned to axes.

   With "Assign Projected Addresses to Slaves", confirm the changes you made manually.

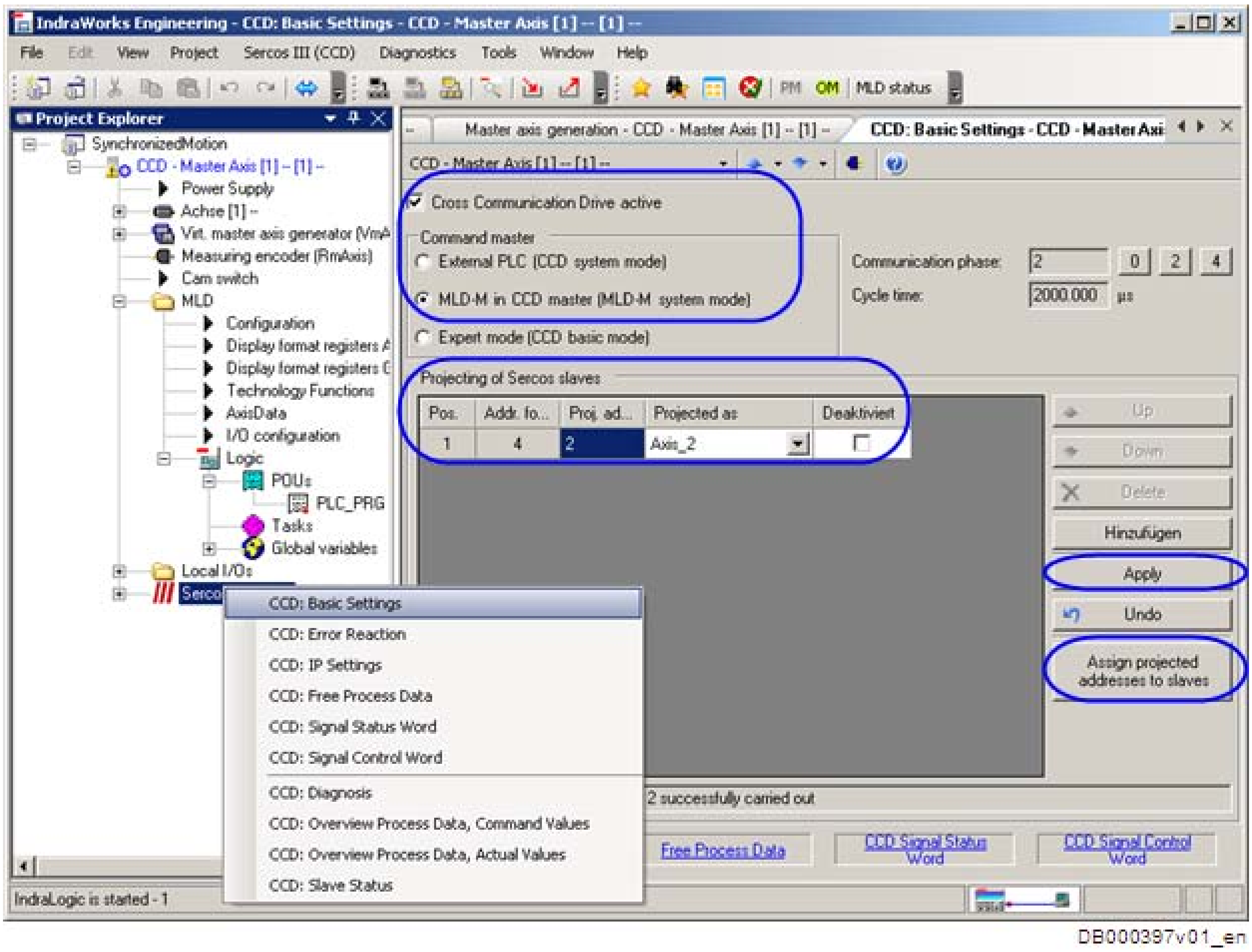


Fig.7-7: IndraWorks Dialog of the Basic CCD Settings With MPx05 Firmware

MLD Configuration

In the drive PLC, you have to select the option "permanent control" for the CCD master. As the axis data structure "AxisData" is used, you must activate it, too (see screenshot below).

Synchronous Multi-Axis Motion With Virtual Master Axis

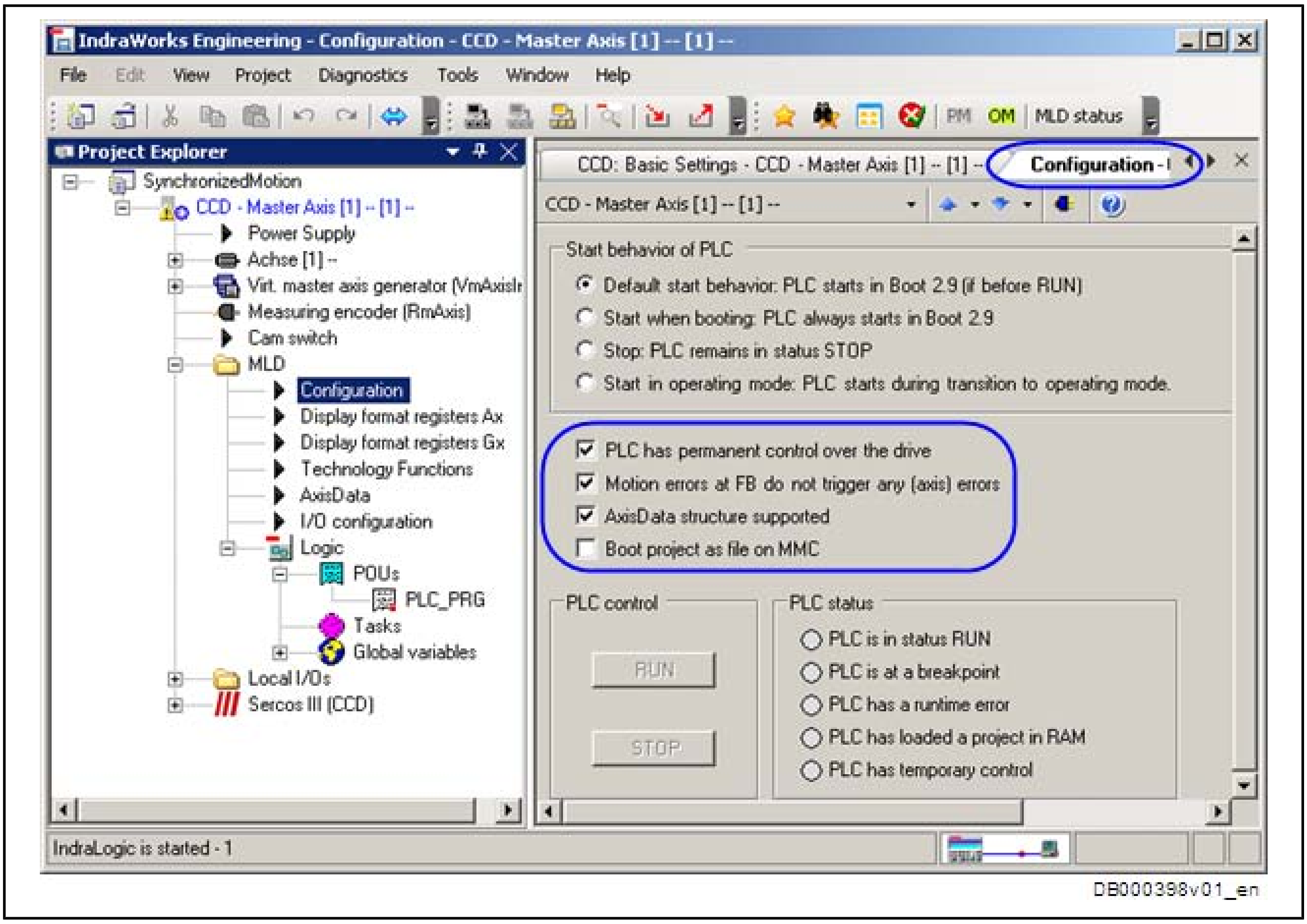


Fig.7-8: IndraWorks Dialog of MLD Configuration

Master Axis Generator/Master Axis Format Converter

The master axis position is derived from the value of the parameter "P-0-0758, Virtual master axis, actual position value" of the virtual axis in the CCD master; the real axes follow this master axis position (see figure "Configuring the Application Example"). The virtual axis can be moved or positioned with the same motion function blocks as the real axes (see also separate documentation "IndraMotion MLD – Library Description"). The virtual actual position value is available in the position data format (degrees, mm, inch), like the actual position values of the real axes. The format of the master axis, which the real drives are to follow, is 2^{20} increments per master axis revolution. Via the functionality "master axis format converter", which has been integrated in the IndraWorks dialog "Master Axis Generator", you can, among other things, convert the content of the parameter "P-0-0758, Virtual master axis, actual position value" to a master axis position. The converted value is available as "P-0-0761, Master axis position for slave axis".

After the action "load basic parameters", the parameter "P-0-0758, Virtual master axis, actual position value" has to be entered for the master axis format converter.

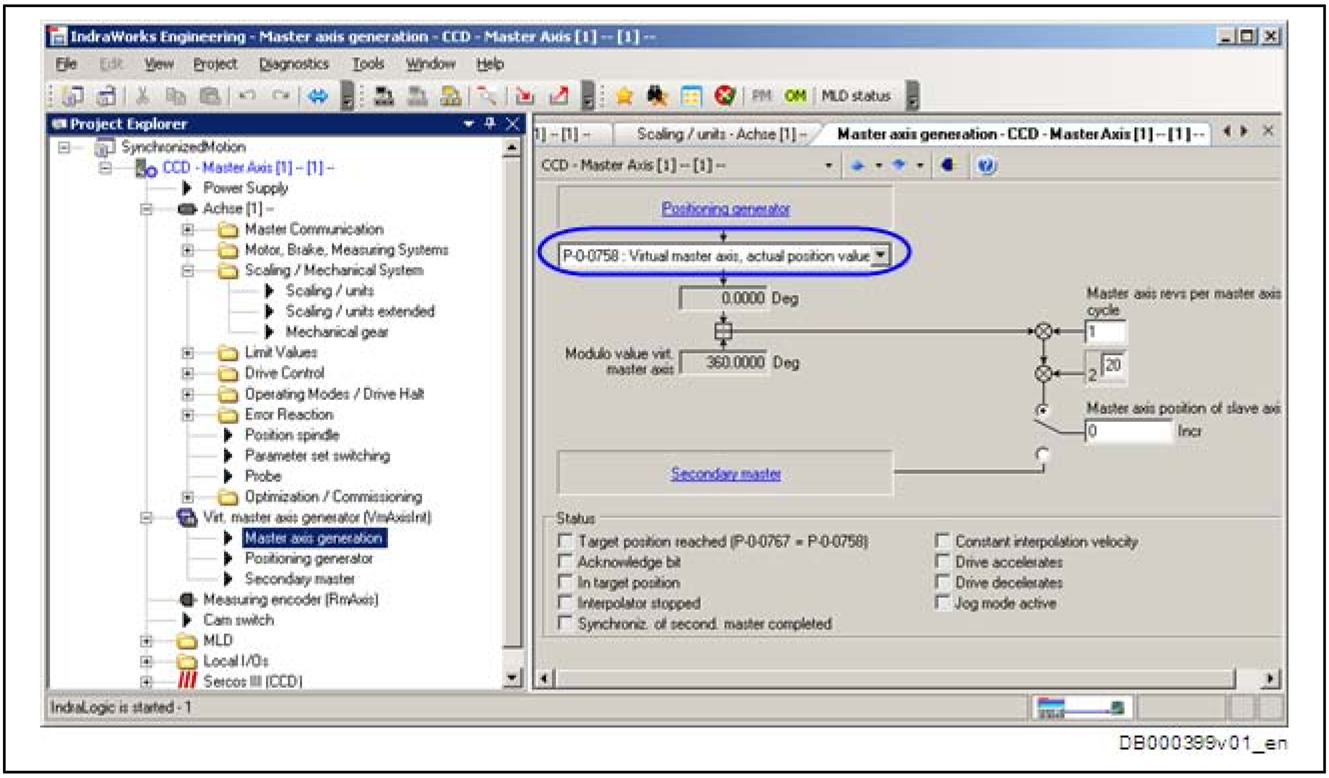


Fig.7-9: IndraWorks Dialog for Setting the Master Axis Generator

Signal Control Word In the IndraWorks dialog shown below, configure the signal control word for the CCD master axis.

Synchronous Multi-Axis Motion With Virtual Master Axis

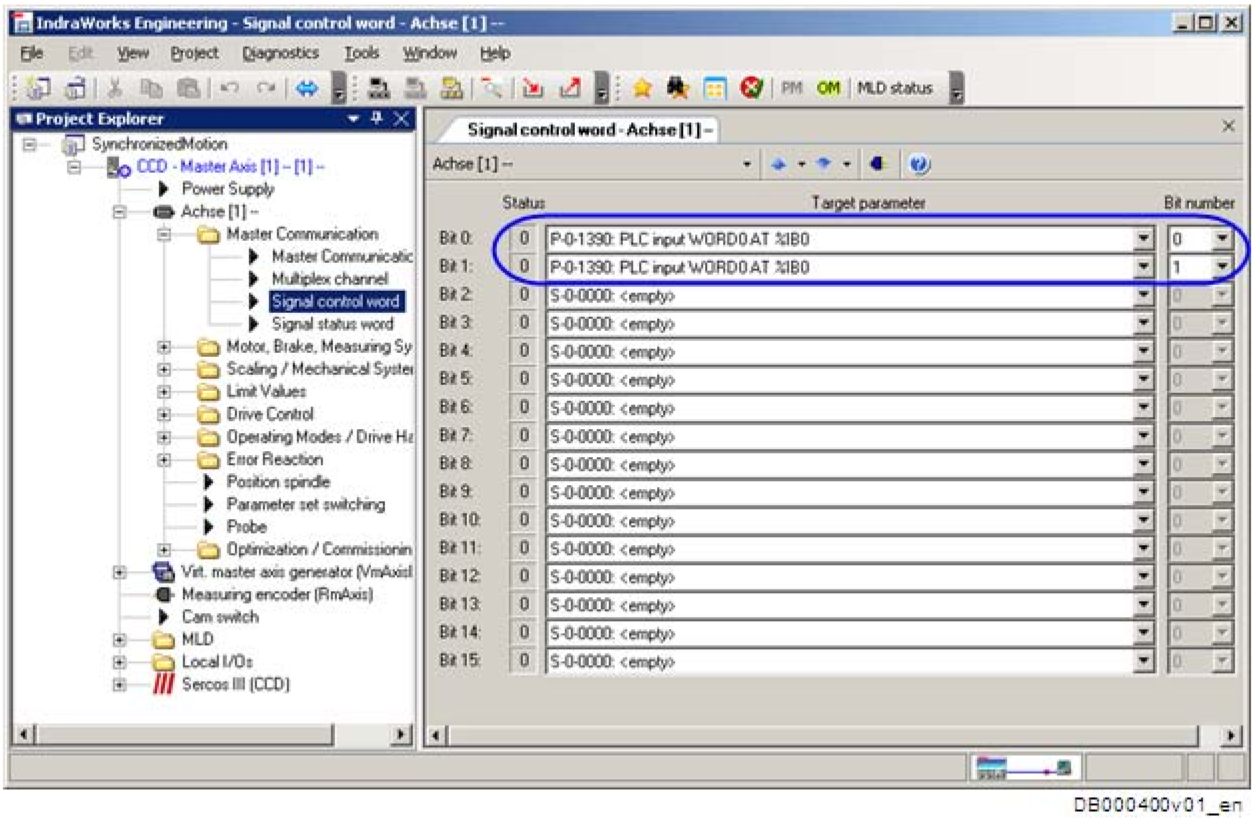


Fig.7-10: IndraWorks Dialog for Configuring the Signal Control Word

With this setting, the bits in the parameter "P-0-1390, PLC input WORD0 AT %IB0" have the following significance:

- **Bit 0** → Start of the application (variable "bAutomatic_i")
- **Bit 1** → Emergency stop switch (variable "bEstop_i")

Establishing Reference

Carry out "set absolute position" procedure or "homing" for the actual position value 1 (S-0-0051).

7.2.3 CCD Slave Axis

Again starting from the completion of the action "load basic parameters", make the following settings for the CCD slave axes.

Enabling of Functional Packages

This application example requires the base package "Closed-Loop" and the enabling of the optional functional package "Synchronization".

☞ It is only allowed to enable licensed functional packages!

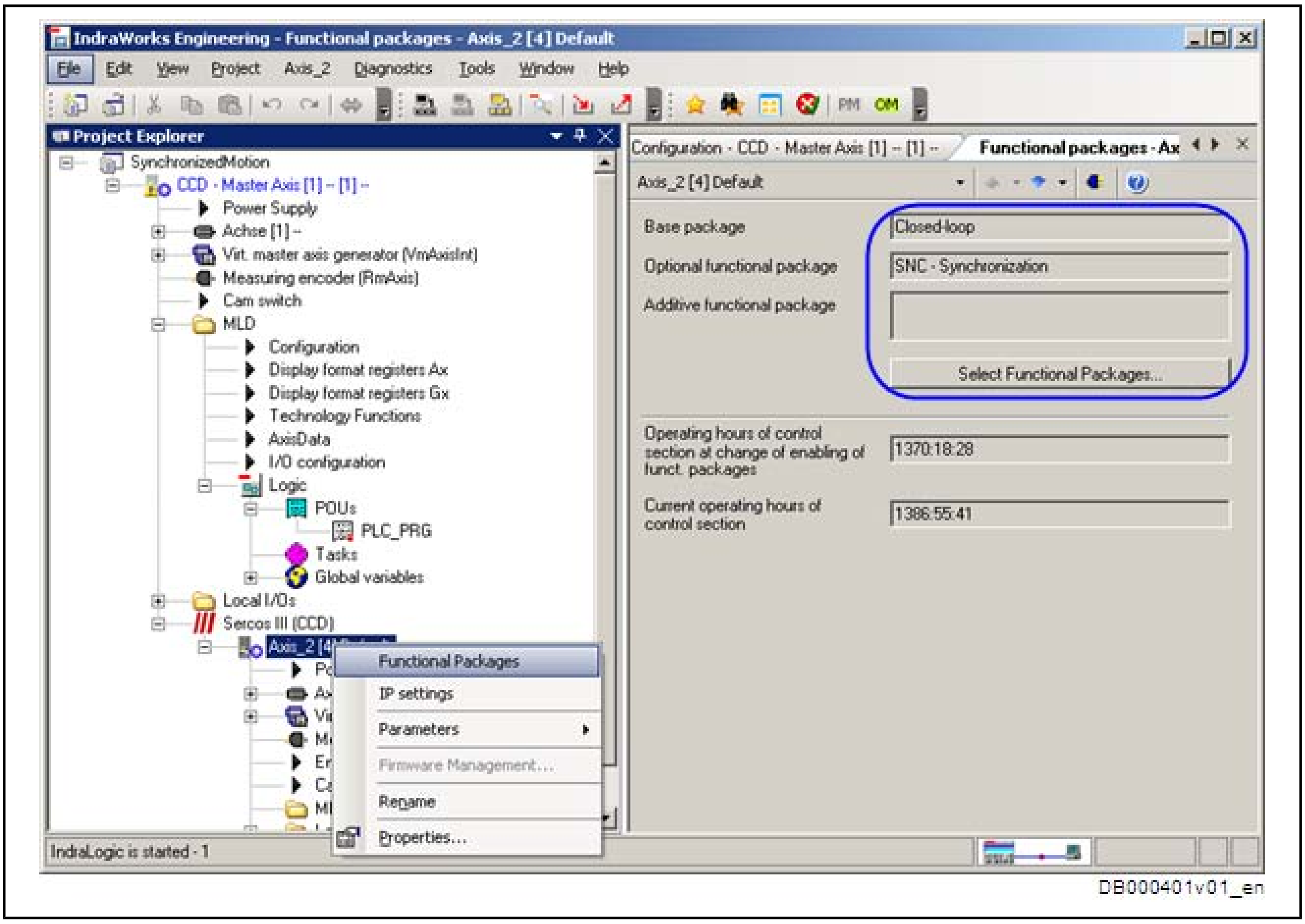


Fig.7-11: IndraWorks Dialog to Enable the Required Functional Packages for the CCD Slave Axis

Settings for Operation Mode "Electronic Motion Profile"

Material feed is to take place from the CCD slave axis which is operated in the "electronic motion profile" mode. In the CCD slave axis, make the settings shown in the IndraWorks dialog below for this application example and this operation mode.

Synchronous Multi-Axis Motion With Virtual Master Axis

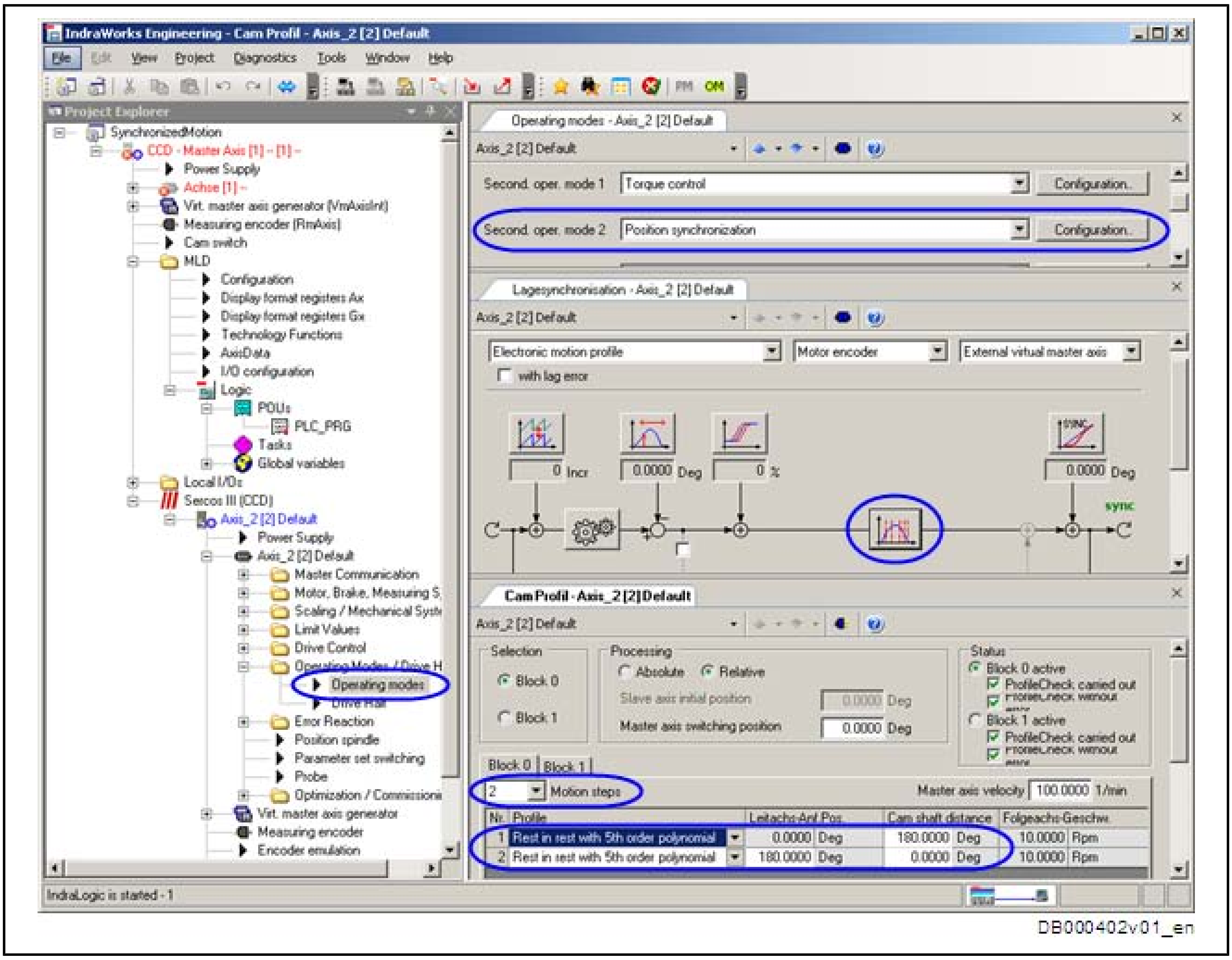


Fig.7-12: IndraWorks Dialog to set the Position Synchronization Mode "Electronic Motion Profile"

In the parameter "P-0-0703, Number of motion steps, set 0", you have defined 2 of 8 possible motion steps with these settings. Define the master axis range in which the corresponding motion step is contained via the parameter "P-0-0705, List of master axis initial positions, set 0". Observe that the motion step 1 always starts with the master axis initial position "0 degrees". The second motion step in this example is from 180 degrees to 360 degrees or 0 degrees. Define the master axis position with which the motion step is accessed by the parameter "P-0-0227, Cam table, access angle". Set the processing mode for each motion step via the parameter "P-0-0706, List of motion step modes, set 0". For this example, the mode "rest in rest via a 5th order polynomial" was selected for both motion steps; i.e. the drive is in standstill at the beginning and at the end. Set the distance for the corresponding motion step via the parameter "P-0-0707, List of distances, set 0". The feed motion in this example is realized in motion step 1. The distance value of step 1 defines the feed length and was set to 180 degrees. In the second motion step, motion is not to take place. That is why the distance was set to zero in this step. In addition, bit 10 (position data processing [electr. motion profile]) of the parameter "P-0-0088, Control word synchronization modes" was set to "1" (relative). In the case of relative processing, a motion step begins at the point where the previous motion step ended. All other settings are made in the program via the function block "MB_MotionProfile".

Scaling Settings

The scaling settings for the CCD slave axis are made as for the CCD master axis.

Establishing Reference

Carry out "set absolute position" procedure or "homing" for the actual position value 1 (S-0-0051).

Synchronous Multi-Axis Motion With Virtual Master Axis

7.3 Programming

The exemplary program has been divided into several code sections which are illustrated in simplified form in the figure below.

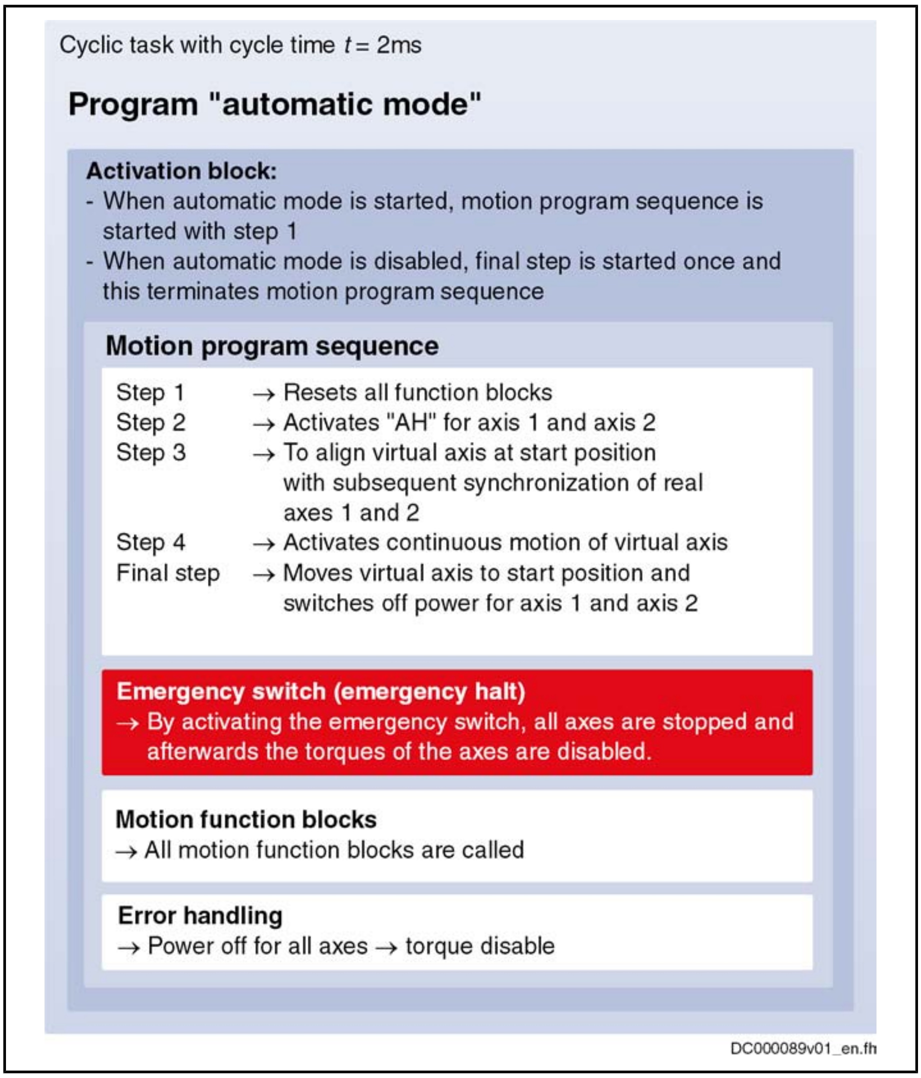


Fig.7-13: Program Structure

The following motion function blocks were used for realizing the application task.

MC_MoveAbsolute

Declaration	fbMoveAbsVmAxis: MC_MoveAbsolute;
Task	Positions the virtual axis of the master axis generator.
Axis	VmAxisInt: Virtual axis of the master axis generator

MC_MoveVelocity

Declaration	fbMoveVelocityVmAxis: MC_MoveVelocity;
Task	Moves the virtual axis of the master axis generator with constant velocity (production velocity).
Axis	VmAxisInt: Virtual axis of the master axis generator

MB_GearInPos

Declaration	fbGearInPosAxis1: MB_GearInPos;
Task	Activation and parameterization of the operation mode "phase synchronization" in the CCD master axis.
Slave	Axis 1 (with MLD-M, always corresponds to master axis; with MLD-S, always corresponds to local axis)
Master	VmAxisExt (specification of master axis via "P-0-0053, Master axis position")

MB_MotionProfile

Declaration	fbMotionProfilAxis2: MB_MotionProfile;
Task	Activation and parameterization of the operation mode "electronic motion profile" in the CCD slave axis.
Slave	Axis2 (corresponds to CCD slave axis [first CCD slave axis in CCD axis group])
Master	VmAxisExt (specification of master axis via "P-0-0053, Master axis position")

Why "VmAxisExt"?

We might ask ourselves, why apply the master axis "VmAxisExt" instead of "VmAxisInt" to the "Master" input of the synchronous motion function blocks"fbGearInPosAxis1" and "fbMotionProfilAxis2".

Justification: The internal virtual master axis "VmAxisInt" always refers to the local axis. Each drive has its own internal virtual master axis (VmAxisInt). It is our objective, however, that all drives follow the same master axis. For this purpose, the internal virtual master axis generated in the CCD master axis must be transmitted to the CCD slave axes. With activated MLD-M system mode, this is automatically configured by the drive. Due to the transmission of the master axis position to the CCD slave axes, there is a dead time of one CCD cycle between the internal master axis (VmAxisInt) of the CCD master axis and the secondary master (VmAxisExt) effective in the CCD slave axes. To compensate for this dead time, the CCD master axis also has a virtual CCD slave axis. The internal virtual master axis (VmAxisInt), too, is transmitted to its own virtual CCD slave axis. The transmitting internal virtual master axis thereby is also available to the CCD master axis itself as an external virtual master axis (VmAxisExt). The resulting dead time is the same as with the real CCD slaves. Thereby, the same master axis is available to all drives in the form of "VmAxisExt".

The following paragraphs will not describe all parts of the program in detail, but only those ones which are responsible for the actual sequence of motion.

Program section 2.3: Calling motion function blocks

Synchronous Multi-Axis Motion With Virtual Master Axis

```
(* ****************************************************************************************
(* Code Section 2.3    -> call all motion function blocks
   (* ************************************************************************************
fbMoveAbsVmAxis(        Execute                 := bExeAbsVmAxis,
                        Position                := rVmInitPos,
                        Velocity                := rVmAbsVel ,
                        Acceleration            := rVmAbsAcc,
                        Deceleration            := rVmAbsDec,
                        Axis                    := VmAxisInt ,
                        Done=> , Active=> , CommandAborted=> , Error=> , ErrorID=> ,ErrorIdent=> );

fbMoveVelocityVmAxis(   Execute                 := bExeVelVmAxis,
                        Velocity                := rVmVel,
                        Acceleration            := rVmVelAcc,
                        Deceleration            := rVmVelDes,
                        Axis                    := VmAxisInt,
                        InVelocity=> , Active=> ,CommandAborted=> ,Error=> ,ErrorID=> , ErrorIdent=> );

fbGearInPosAxis1(       Execute                 := bExeGearInPos,
                        RatioNumerator          := uiGearOutAxis1,
                        RatioDenominator        := uiGearInAxis1,
                        MasterFineadjust        := rFineAdjustAxis1 ,
                        SyncMode                := enSyncDirectAxis1 ,
                        StartMode               := enSyncModeAxis1 ,
                        Master                  := VmAxisExt ,
                        Slave                   :=Axis1 ,
                        InSync=> , Active=> , CommandAborted=> ,Error=> , ErrorID=> , ErrorIdent=> );

fbMotionProfilAxis2(    Execute                 := bExeMoProfil,
                        RatioNumerator          := uiGearOutAxis2,
                        RatioDenominator        := uiGearInAxis2,
                        MasterFineadjust        := rFineAdjustAxis2,
                        SetSelection            := uiSetSelection ,
                        RelativePositioning     := bRelPositioning,
                        SyncMode                := enSyncDirectAxis2,
                        StartMode               := enSyncModeAxis2 ,
                        Master                  := VmAxisExt,
                        Slave                   := Axis2,
                        InSync=> , Active=> ,CommandAborted=> ,Error=> ,ErrorID=> ,ErrorIdent=> ,ActiveSet=> );
```

DB000403v01_nn

Fig.7-14: Code Section 2.3

```
(* 2.0 variables of virtuell Axis *)
(* 2.1 input variables of fbMoveAbsVmAxis *)
bExeAbsVmAxis           : BOOL := FALSE;    (* start FB with rising edge                                   *)
rVmInitPos              : REAL := 0.0;      (* P-0-0766, Virtual master axis, positioning command value   *)
                                            (* -> start position                                           *)
rVmAbsVel               : REAL := 50.0;     (* P-0-0770, Virtual master axis, positioning velocity        *)
rVmAbsAcc               : REAL := 10.0;     (* P-0-0771, Virtual master axis, positioning acceleration    *)
rVmAbsDec               : REAL := 10.0;     (* P-0-0772, Virtual master axis, positioning deceleration    *)

(* 2.2 input variables of fbMoveVelocityVmAxis *)
bExeVelVmAxis           : BOOL := FALSE;    (* start FB with rising edge                                   *)
rVmVel                  : REAL := 50;       (* P-0-0770, Virtual master axis, positioning velocity        *)
rVmVelAcc               : REAL := 10;       (* P-0-0771, Virtual master axis, positioning acceleration    *)
rVmVelDes               : REAL := 10;       (* P-0-0772, Virtual master axis, positioning deceleration    *)

(* 3.0 variables of Axis1   *)
(* 3.1 input variables of fbGearInPosAxis1 *)
bExeGearInPos           : BOOL := FALSE;    (* start wirth rising edge                                     *)
uiGearOutAxis1          : UINT := 1;        (* P-0-0157, Master drive gear output revolutions             *)
uiGearInAxis1           : UINT := 1;        (* P-0-0156, Master drive gear input revolutions              *)
rFineAdjustAxis1        : REAL := 0;        (* P-0-0083, Gear ratio fine adjust                            *)
enSyncDirectAxis1       : MC_SYNC_DIRECTION := CATCH_UP  ;  (* P-0-0154, Synchronization direction  -> pos. Direction*)
enSyncModeAxis1         : MC_START_MODE := ABSOLUTE;        (* P-0-0155,  Synchronization mode Bit1 := 0          *)

(* 4.0 variables of Axis2   *)
(* 4.1 input variables of MotionProfilAxis2 *)
bExeMoProfil            : BOOL := FALSE;    (* start wirth rising edge                                     *)
uiGearOutAxis2          : UINT := 1;        (* P-0-0157, Master drive gear output revolutions             *)
uiGearInAxis2           : UINT := 1;        (* P-0-0156, Master drive gear input revolutions              *)
rFineAdjustAxis2        : REAL := 0;        (* P-0-0083, Gear ratio fine adjust                            *)
uiSetSelection          : UINT := 0;        (* P-0-0088, Control word for synchronous operation modes     *)
                                            (*   -> Bit9 = 0  ->  activate set 0                           *)
bRelPositioning         : BOOL := TRUE;     (* P-0-0088, Control word for synchronous operation modes     *)
                                                        (*  -> Bit10 = 1  -> position data processing is relative  *)
enSyncDirectAxis2       : MC_SYNC_DIRECTION := CATCH_UP  ;  (* P-0-0154, Synchronization direction  -> pos. Direction*)
enSyncModeAxis2         : MC_START_MODE := RELATIVE;        (* P-0-0155,  Synchronization mode Bit1 := 1          *)
```

DB000404v01_nn

Fig.7-15: Declaration of Input Variables for Motion Function Blocks

In this program section, all motion function blocks which are used are called once per task cycle with the condition specified for the corresponding Execute input.

☞ The significance of all inputs/outputs of all function blocks is described in the separate documentation "IndraMotion MLD – Library Description".

Program section 2.1: Motion Control (case instruction)

With this program section, control of the motion program takes place.

Step 1 (initStep): Initial state

Step 1 carries out the reset or initialization of the motion program. All used function blocks are initialized by direct call with "Execute" or "Enable = FALSE", or by setting the corresponding activation variable (variable which is connected to the "Execute" or "Enable" input) to FALSE. Furthermore, the virtual master axis is deactivated, its current actual velocity (P-0-0759) is set to zero and the command value processing mode (P-0-0769) of the virtual master axis is set to positive direction of rotation; i.e. the virtual master axis (VmAxisInt) can only be moved in positive direction. Thereafter, switching to step 2 takes place.

Synchronous Multi-Axis Motion With Virtual Master Axis

```
(* ************************************************************************************************* *)
(* Code Section 2.1    -> case instruction
(* ************************************************************************************************* *)
CASE iStateAutomatic OF
iInitStep: (* Code Section 2.1 Step1     -> init step  -> initialize of all motion functions and variables
    fbE_StopAxis1(  Execute := FALSE , Deceleration := rHaltAccAxis1_gb,Axis := Axis1 ,
                    Done=> , Active=> , CommandAborted=> , Error=> , ErrorID=> , ErrorIdent=> );    (* Reset MC_Stop of axis1 *)

    fbE_StopAxis2(  Execute := FALSE , Deceleration := rHaltAccAxis2_gb,Axis := Axis2 ,
                    Done=> , Active=> , CommandAborted=> , Error=> , ErrorID=> , ErrorIdent=> );    (* Reset MC_Stop of axis2 *)

    fbE_StopVmAxis(  Execute :=FALSE , Deceleration := rHaltAccAxis1_gb, Axis := VmAxisInt ,
                     Done=> , Active=> ,CommandAborted=> , Error=> , ErrorID=> , ErrorIdent=> );    (* Reset MC_Stop of virtual axis *)

    bEstopInternal  := FALSE;        (* reset internal emergency switch *)

    fbPowerAxis1(Enable:= FALSE, Axis:=Axis1 ,Status=> , Error=> ,ErrorID=> ,ErrorIdent=> );        (* Power off Axis1 (torque free)*)
    fbPowerAxis2(Enable:= FALSE, Axis:=Axis2 ,Status=> , Error=> ,ErrorID=> ,ErrorIdent=> );        (* Power off Axis2 (torque free)*)

    bPow1Error_gb           := FALSE;          (* variable for visualization *)
    enPow1ErrorTable_gb     := NO_TABLE_USED;  (* variable for visualization *)
    dwPow1ErrorIdent_gb     := 0;              (* variable for visualization *)
    bPow2Error_gb           := FALSE;          (* variable for visualization *)
    enPow2ErrorTable_gb     := NO_TABLE_USED;  (* variable for visualization *)
    dwPow2ErrorIdent_gb     := 0;              (* variable for visualization *)

    DV_P_0_0917 := 0;              (* virtual axis is disabled *)
    DV_P_0_0759 := 0;              (* set P-0-0759, Virtual master axis, actual velocity value to 0 rpm *)
    DV_P_0_0769 := 0;              (* set P-0-0769, Virtual master axis, command value mode to 0 *)
                                   (* Only positive direction of rotation is allowed for positioning *)


    bExeVelVmAxis   := FALSE;      (* set input Execute of fbMoveVelocityVmAxis: MC_MoveVelocity to FALSE *)
    bExeAbsVmAxis   := FALSE;      (* set input Execute of fbMoveAbsVmAxis:      MC_MoveAbsolute to FALSE *)
    bExeGearInPos   := FALSE;      (* set input Execute of fbGearInPosAxis1:     MB_GearInPost to FALSE *)
    bExeMoProfil    := FALSE;      (* set input Execute of fbMotionProfilAxis2 : MB_MotionProfile to FALSE *)

    iStateAutomatic := iPowerStep;  (* in the next task cycle go to case iPowerStep *)
```

DB000405v01_nn

Fig.7-16: *Code Section 2.1.1*

Step 2 (iPowerStep): AH (drive)

In step 2, drive enable is set for both real axes. This is done for each axis via the function block "MC_Power". Both axes then are in the drive state "Drive Halt" (AH). When both axes are in the state "AH", switching to step 3 takes place. The state is determined by evaluating the status output of the corresponding function block "MC_Power".

```
iPowerStep: (* Code Section 2.1 Step2     -> power step  -> to activate AH for all real axis *)
    fbPowerAxis1(Enable:= TRUE, Axis:=Axis1 ,Status=> , Error=> ,ErrorID=> ,ErrorIdent=> ); (* Power on Axis1  (AH > torque )*)
    fbPowerAxis2(Enable:= TRUE, Axis:=Axis2 ,Status=> , Error=> ,ErrorID=> ,ErrorIdent=> ); (* Power on Axis2  (AH -> torque)*)

    IF fbPowerAxis1.Status AND fbPowerAxis2.Status THEN  (* if the state of Axis 1 and Axis 2 AH *)
        iStateAutomatic := iToJoggle;                    (* yes -> in the next task cycle go to case iToJoggle *)
    END_IF
```

DC000406v01_nn

Fig.7-17: *Code Section 2.1.2*

Step 3 (iToJoggle): Aligning the installation automatically

```
iToJoggle:  (* Code Section 2.1 Step3 -> to joggle the machine *)
    DV_P_0_0917 := 1;             (* activate function virtual axis *)
    bExeGearInPos := TRUE;        (* set input Execute of fbGearInPosAxis1 *)
    bExeAbsVmAxis := TRUE;        (* set input Execute of fbMoveAbsVmAxis *)

    IF fbMoveAbsVmAxis.Done AND fbGearInPosAxis1.InSync THEN
        IF fbMotionProfilAxis2.InSync THEN
            bExeAbsVmAxis    := FALSE;                (* reset input Execute of fbMoveAbsVmAxis *)
            iStateAutomatic :=  iContinuousMotion ;   (* in the next task cycle go to case iContinuousMotion *)
        ELSE
            bExeMoProfil     := TRUE;                 (* set input Execute of MotionProfilAxis2 to TRUE *)
        END_IF;
    END_IF
```

DB000407v01_nn

Fig.7-18: *Code Section 2.1.3*

To avoid collision of punching machine and material when the operation modes are activated, the two axes are aligned with one another. For this purpose, the virtual master axis generator is activated by writing the value "1" to the direct variable "DV_P_0_0917". Afterwards, the operation mode "phase synchronization" is activated for the punching drive via the function block "fbGearInPosAxis1" by setting the variable "bExeGearInPos". Simultaneously, the positioning of the virtual master axis is started via the function block "fbMoveAbsVmAxis" by setting the variable "bExeAbsVmAxis" to the initial position of the master axis for material feed (rVmInitPos: REAL: = 0.0;). When the function block "fbGearInPosAxis1", via the "InSync" output, signals that the punching drive has synchronized and the function block "fbMoveAbsVmAxis" additionally signals via the "Done" output that the initial position of the master axis for material feed has been reached, the operation mode "electronic motion profile" is activated for the roll drive via the function block "fbMotionProfilAxis2" by setting the variable "bExeMoProfil". When this function block, too, signals via the "InSync" output that it has synchronized, switching to step 4 takes place.

Step 4 (iContinuousMotion): Continuous operation

```
iContinuousMotion:  (* Code Section 2.1 Step4 -> continuous motion *)
    bExeVelVmAxis := TRUE;                          (* set input Execute of fbMoveVelocityVmAxis *)
                                                                                DB000408v01_nn
```

Fig.7-19: Code Section 2.1.4

After both axes have been aligned with one another, the actual application operation can start. For application operation, the virtual master axis is only moved continuously. For this purpose, the function block "fbMoveVelocityVmAxis" is activated by setting the variable "bExeVelVmAxis".

Synchronous Multi-Axis Motion With Virtual Master Axis

7.4 Commissioning and Testing

The exemplary program has been designed in such a way that it can be tested as an independent program.

→ Create a new IndraWorks project.

Commissioning sequence of this application example in the drive:

1. Parameterize/configure drives as described.
2. Afterwards, execute "Restore" task in IndraWorks project.

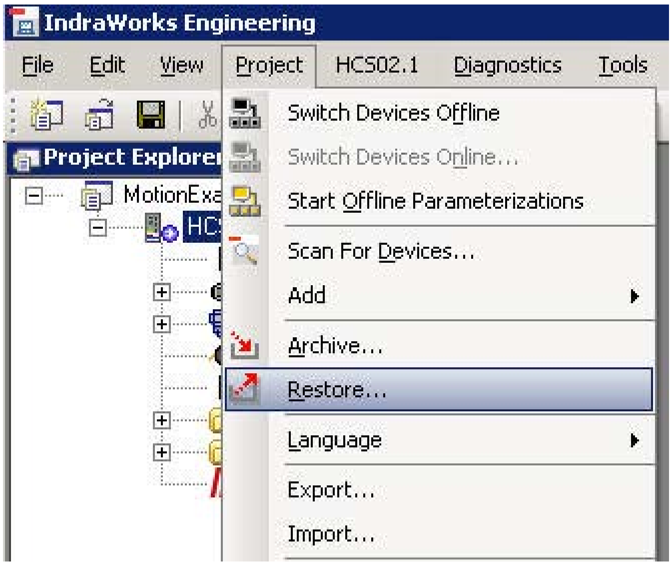


Fig.7-20: Restoring an IndraWorks Archive

3. Connect to drive by going online.
4. Switch both axes to operating mode (OM); clear error message.
5. Switch power on → drives then are in state "AB".
6. Establish absolute position data reference for both axes.
7. Open exemplary project via IndraLogic branch by double-clicking "PLC_PRG":

Fig.7-21: Part of IndraWorks Structure Tree

8. Go to **IndraLogic**.
9. In IndraLogic menu, check the following settings and correct them, if necessary:

   → Project → Options → Build → tick check box "Replace constants"
10. Start PLC via "F5".

11. Compile program and then load it to drive.
12. Set bit 0 of signal control word via switch (via this bit, the so-called "automatic mode" is started or disabled).

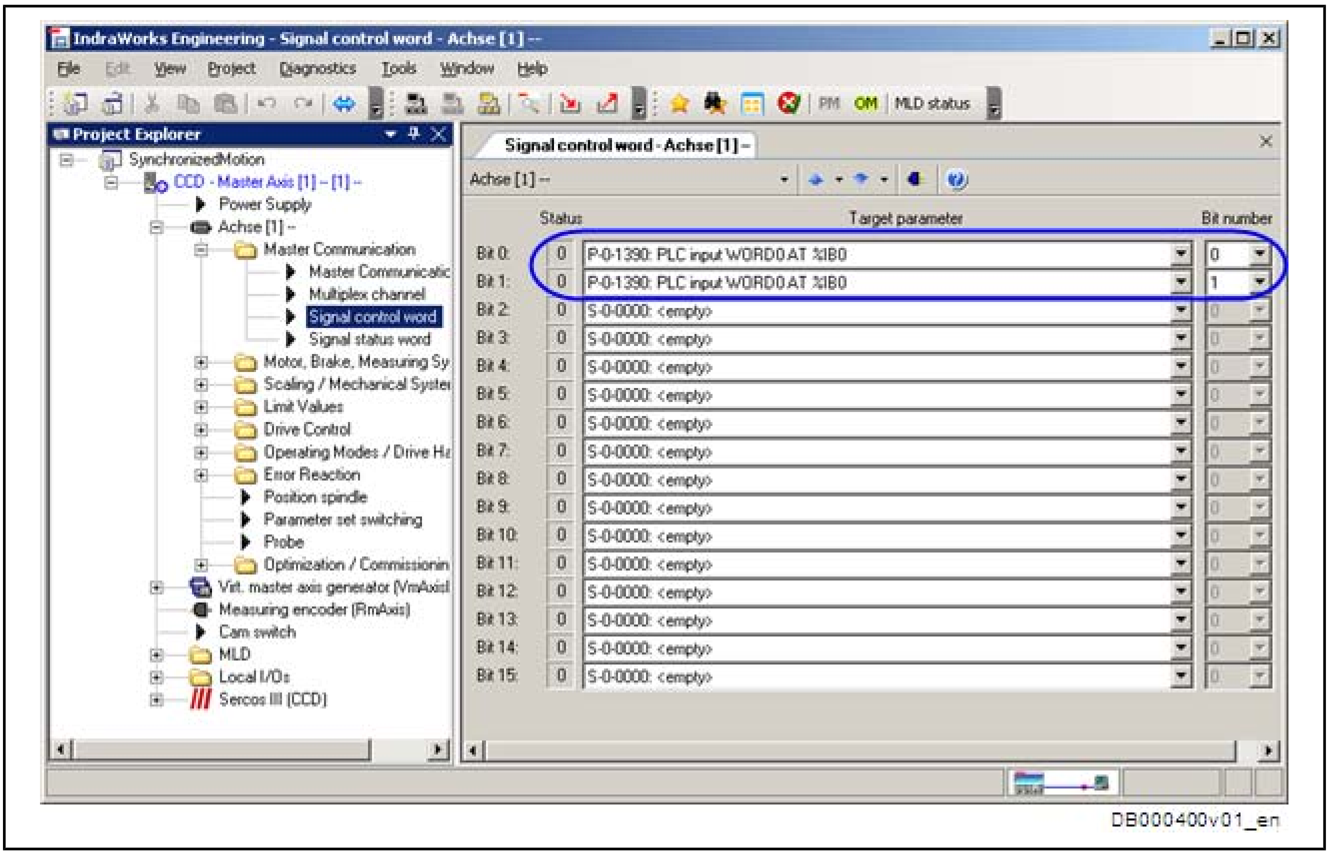


Fig.7-22: IndraWorks Dialog: Configuring the Signal Control Word

13. **Emergency stop switch:** The emergency stop switch has been assigned to bit 1 of the signal control word. When the emergency stop switch is set, the axes are stopped with the deceleration that has been set. Afterwards, drive enable is removed for both axes. After the emergency stop switch was reset, the automatic mode has to be started again via bit 0 of the signal control word.
14. If one or both axes signal an error, this error message must be "manually" cleared. After you have fixed the cause of the error and cleared the error message, restart the automatic mode as described!

7.5 Notes on Programming and Parameterization for Other Relevant Types of Master Axis Linking

7.5.1 General Information

Independent of the application example, the following paragraphs show other relevant variants of master axis linking and its realization in the MLD-M system mode.

With minor adjustments of this application example 1, these variants of synchronous multi-axis motion can be realized.

7.5.2 Real Axis in CCD Slave Moves Synchronously to Real Axis in CCD Master

Example

There are applications in which axes are to carry out the same motion as the real CCD master axis. At this point, it is possible to extend the example; a second axis (CCD slave axis), for example, is to move in a phase synchronous way to the CCD master axis. The CCD slave axis is operated in the operation mode "phase synchronization". The principle of this operation mode is that the real drive, after it has synchronized to the master axis, moves in a phase synchronous way to the master axis. This master axis ("P-0-0053, Master axis position") is generated from the parameter "P-0-0434, Position command value of controller" of the CCD master axis. The value of P-0-0434, which is available in the position data format, must be converted to the master axis format. This is done via the master axis format converter in the CCD master (see also IndraWorks dialog "Master Axis Generator" and Functional Description "Master Axis Generator: Master Axis Format Converter" in the drive documentation). The result is contained in the parameter "P-0-0761, Master axis position for slave axis". This value must be transmitted to the slave axis. Due to the CCD transmission and the subsequent command value processing in the CCD slave, a position offset would occur between the CCD master and the CCD slave. To compensate this, the master axis position must be pre-controlled. This is done via an extrapolator. The output of the extrapolator is written to the parameter "P-0-0053, Master axis position" and is available to all axes in the CCD group at the same time with the same value. For the configuration, see the table in the following section.

Parameterization

The parameterization of example 1 has to be extended as follows:

Master axis format converter	CCD dead time compensation (extrapolator)	
P-0-0916, Master axis format converter signal selection	P-0-1616, CCD: Extrapolated cmd value signal selection	P-0-1617, CCD: Number of extrapolation steps
P-0-0434	P-0-0761	2

☞ When the CCD master axis has been scaled in absolute form, the CCD slave axis, too, must be scaled in absolute form and the parameter "P-0-0750, , Master axis revolutions per master axis cycle" must have the value "0" for all axes.

Programming

The operation mode "phase synchronization" is activated via the function block "MB_GearInPos" for the CCD slave axis.

MB_GearInPos

Declaration	fbGearInPosAxis2: MB_GearInPos;
Task	Activation and parameterization of the operation mode "phase synchronization" in the CCD slave axis.
Slave	Axis2 (corresponds to CCD slave axis [first CCD slave axis in CCD axis group])
Master	VmAxisExt (specification of master axis via "P-0-0053, Master axis position")

Real Axis in CCD Master and CCD Slave Move Synchronously to Measuring Encoder Position in CCD Master

Example

Two axes are to move phase-synchronously to the measuring encoder position of the CCD master axis.

Parameterization

Parameterization must be made in the CCD master axis.

Master axis format converter	CCD dead time compensation (extrapolator)	
P-0-0916, Master axis format converter signal selection	P-0-1616, CCD: Extrapolated cmd value signal selection	P-0-1617, CCD: Number of extrapolation steps
P-0-0052	P-0-0761	2

☞ The parameter "P-0-0750, Master axis revolutions per master axis cycle" for all axes has to be set to the value of the parameter "P-0-0765, Modulo factor measuring encoder" of the CCD master axis.

Programming

The operation mode "phase synchronization" is activated via the function block "MB_GearInPos".

MB_GearInPos

Declaration	fbGearInPosAxis1: MB_GearInPos;
Task	Activation and parameterization of the operation mode "phase synchronization" in the CCD master axis.
Slave	Axis 1 (with MLD-M, always corresponds to CCD master axis and with MLD-S, always corresponds to local axis)
Master	VmAxisExt (specification of master axis in the parameter "P-0-0053, Master axis position")

MB_GearInPos

Declaration	fbGearInPosAxis2: MB_GearInPos;
Task	Activation and parameterization of the operation mode "phase synchronization" in the CCD slave axis.
Slave	Axis2 (corresponds to CCD slave axis → first CCD slave axis in CCD axis group)
Master	VmAxisExt (specification of master axis in the parameter "P-0-0053, Master axis position")

7.5.3 Real Axis in CCD Master and CCD Slave Move Synchronously to Measuring Encoder Position in CCD Slave

Example

Two axes are to move phase-synchronously to the measuring encoder position of the CCD slave axis.

Parameterization

Synchronous Multi-Axis Motion With Virtual Master Axis

To solve this task, it is first necessary to transmit the measuring encoder position of the CCD slave axis to the CCD master axis. For this purpose, make the following parameter setting in the CCD dialog of IndraWorks:

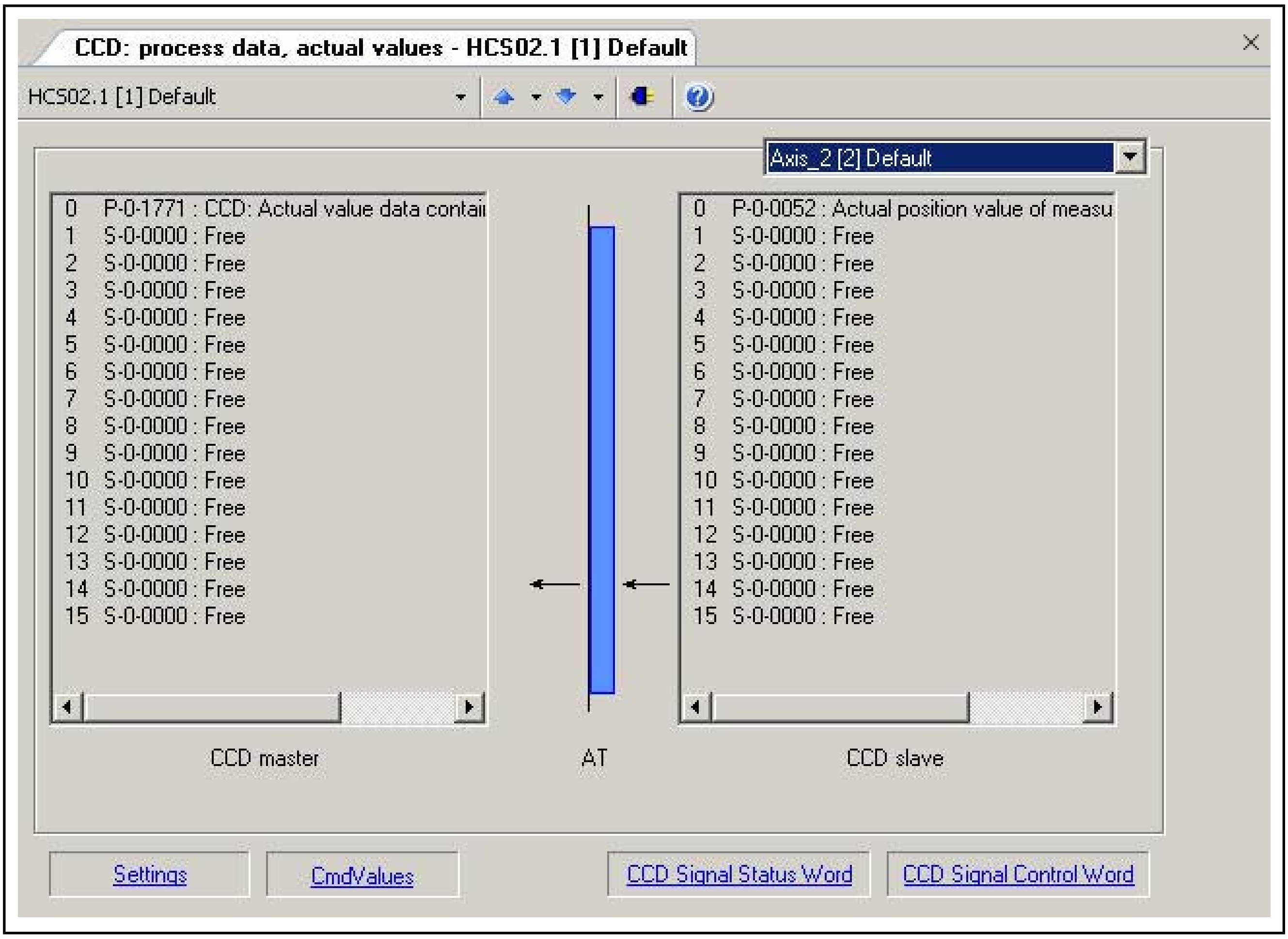


Fig.7-23: IndraWorks Dialog: CCD: Process Data, Actual Values

Afterwards, make the following parameterization in the CCD master axis:

Master axis format converter	**CCD dead time compensation (extrapolator)**	
P-0-0916, Master axis format converter signal selection	P-0-1616, CCD: Extrapolated cmd value signal selection	P-0-1617, CCD: Number of extrapolation steps
P-0-177x	P-0-0761	3

☞ The parameter "P-0-0750, Master axis revolutions per master axis cycle" for all axes has to be set to the value of the parameter "P-0-0765, Modulo factor measuring encoder" of the CCD slave axis.

Programming

See example "Real Axis in CCD Master and CCD Slave Move Synchronously to Measuring Encoder Position in CCD Master"

7.5.4 Position Command Value Linking (Gantry Axis)

Real Axis in CCD Slave Moves Synchronously to CCD Master (Gantry Group)

Example

Gantry axes are used very often to realize synchronous motion between 2 axes by means of position command value linking.

Parameterization

CCD slave axis

In the CCD slave axis, the operation mode "drive-controlled position control" (0x0305) has to be configured in the parameter "S-0-0287, Secondary operation mode 7". As the secondary operation mode 7, because MLD-M of the CCD master axis automatically configures the other operation modes.

As you want to move synchronously to the CCD master axis, you must set the parameter "P-0-0187, Position command processing mode" to the value "0"; i.e. the position command value (S-0-0047) is processed within one NC cycle (S-0-1001).

CCD master axis

The principle of Gantry linking is such that the value of the parameter "P-0-0434, Position command value of controller" of the CCD master axis is transmitted as the position command value (S-0-0047) to the CCD slave axis.

However, it is not possible to do this directly, as the transmission dead time between CCD master axis and CCD slave axis must be compensated. It is also necessary to compensate the processing time of the command value (S-0-0047) in the CCD slave axis. The dead time compensation is realized by means of the CCD extrapolator.

Make the following parameter setting in the CCD master axis:

CCD dead time compensation (extrapolator)	
P-0-1616, CCD: Extrapolated cmd value signal selection	P-0-1617, CCD: Number of extrapolation steps
P-0-0434	2

After the extrapolator has been parameterized, the output of the extrapolator (corresponds to the value of the parameter "P-0-1618, CCD: Extrapolated command value") must be configured as the value for the parameter "S-0-0047, Position command value" in the MDT for the CCD slave (see CCD dialog).

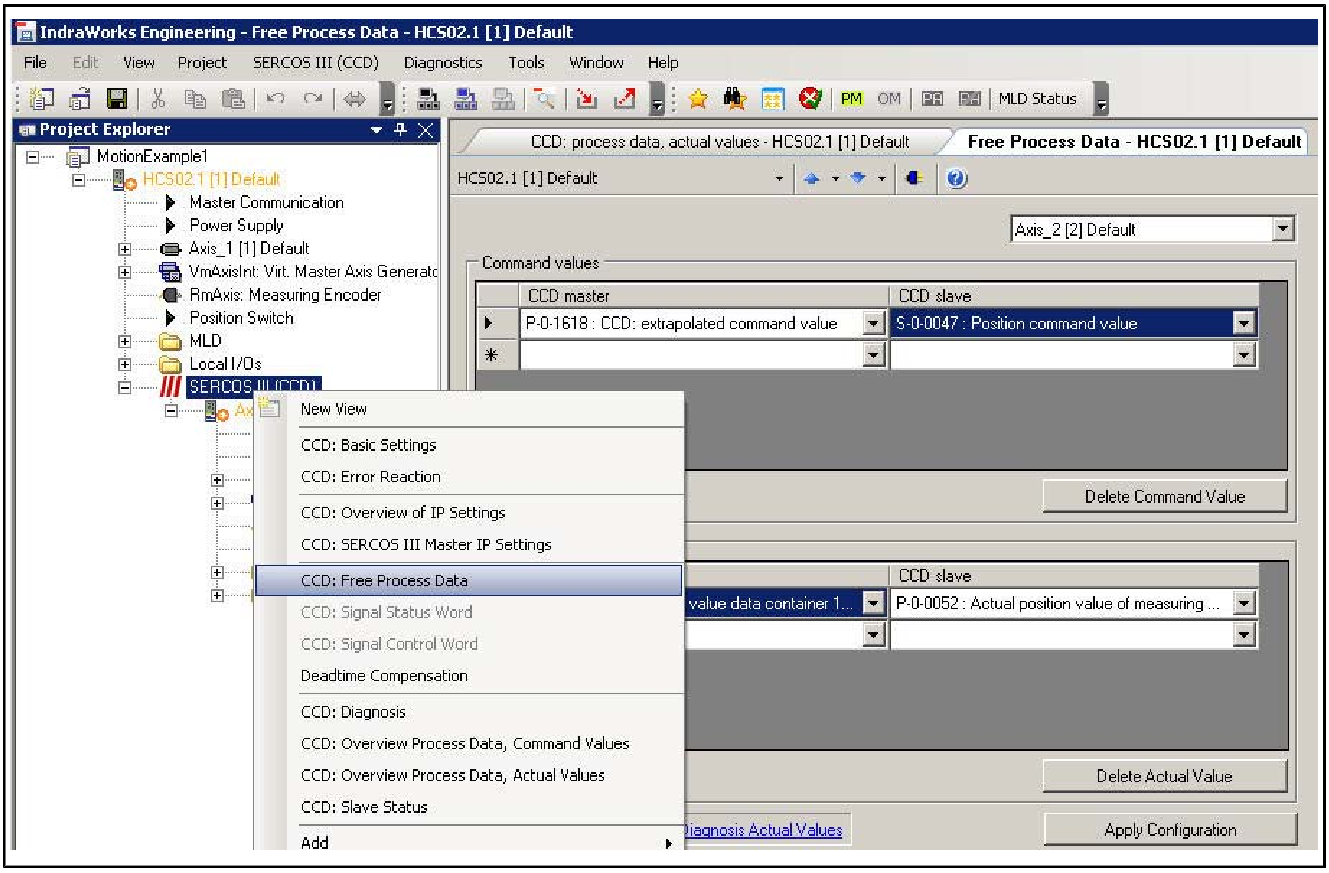


Fig.7-24: IndraWorks Dialog: CCD: Process Data, Command Values

Programming

The secondary operation mode 7 ("drive-controlled position control") in the CCD slave axis can be activated via the function block "MX_SetOpMode".

Declaration

fbSetOpModeAxis2: MX_SetOpMode;

Coding

SetOpModeAxis2

(NewOpMode: = 7, → (* activate secondary operation mode 7 *)

Execute: = TRUE, → (* activate the function block *)

Axis: = Axis2); → (* first CCD slave axis *)

8 Vibration Damping With Superimposed Process Loop (Process Control With Intelligent Servo Axis)

8.1 Task Definition – Application Description

8.1.1 Task Definition

Using an external sensor, active vibration damping (process control) for low resonance frequencies is to be implemented by means of "IndraMotion MLD".

The objective is to minimize the vibration behavior of the axis and, consequently, to achieve higher contour precision of the axis. In order that the normal positioning process is not inhibited, the process loop is a cascade superimposed to the drive control loop structure. That is to say, the command value must continue taking effect in the respective operation mode and a command value of the process loop is simply added to it.

☞ For this purpose, the MLD-S library "DRIVE_LIB_01V02.lib" makes available a comprehensive PID loop function block!

8.1.2 Functional Overview/Concept

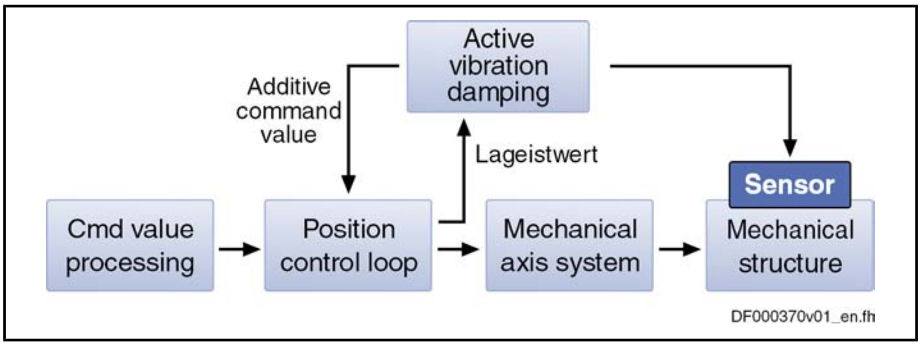


Fig.8-1: Functional Concept

Detecting the Actual Value (Acceleration)

The output signal of an acceleration sensor ist to be read in via an analog input at the IndraDrive control section. This sensor is mounted at a point of the mechanical system which is susceptible to vibration. After the sensor signal has been adjusted, it is used as the actual value for process control.

☞ It is our objective to reduce the vibration occurring at this point in order to achieve a better dynamic response and higher production quality.

Point at Which the Process Loop Takes Effect

The resulting actuating variable of the control loop is added to the command value. This does not affect the command value input of the external PLC, i.e. the process loop is superimposed to the drive control loop structure. Thus, an external PLC always has Motion Control over the drive.

For the point at which the command value takes effect in the drive cascade structure, the following additive command values are available:

- P-0-0059, Additive position command value, controller
- S-0-0037, Additive velocity command value
- S-0-0081, Additive torque/force command value

The paragraphs below describe the steps required to solve the task and show how to create a simple program. The developed program, however, does not contain any error handling.

Vibration Damping With Superimposed Process Loop (Process Control With Intelligent Servo Axis)

8.2 Requirements/Settings

Drive Configuration

Starting from the basic parameters, you have to make some fundamental settings for the application example "process control for vibration damping".

Connecting the Sensor

The number and functions of the analog inputs differ according to the type and configuration of control section.

For more detailed information on the different control sections, see the hardware documentation for the control sections "Rexroth IndraDrive - Control Sections for Drive Controllers" (Project Planning Manual). For the example described below, we use a control section of the CSH01.1C type. See the Project Planning Manual of this control section for the pin and connector assignment.

Function			Con-nection	Factory setting	Nominal data	Technical data
digital outputs	1	A8	X32.6		24 V / 0,5 A	digital outputs
		A9	X32.7			
		A10	X32.8			
		A11	X32.9			
analog inputs	voltage input	EAn+	X32.4		+/-10 V typ. 2 kOhm	analog inputs type 4
		EAn-	X32.5			
analog outputs	voltage output	An1	X32.1		5 V / 1mA	analog outputs type 2
		An2	X32.2			
	reference potential for analog voltage output	A_GND	X32.3			
voltage supply of digital inputs/outputs	voltage supply of digital inputs/outputs	+24V	X31.8			DC 19...30 V; max. 1,1 A
		0V	X31.9			
serial interface	RS232		X2			serial interface
Optional functions	allowed options see configuration table					see corresponding optional module

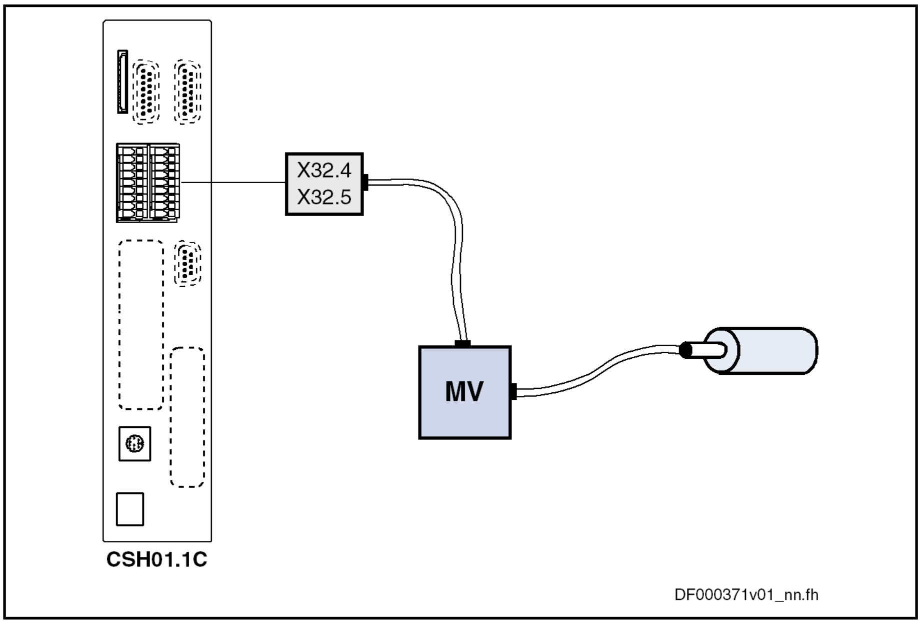
DB000385v01_en

Fig.8-2: Pin Assignment of a CSH01.1C Control Section

The amplified sensor signal is connected to the corresponding connection. The input voltage range of the control sections is +/-10 V. The amplification of the measuring amplifier is selected such that the biggest possible window of the input voltage range is used. Due to the vibration properties of the mechanical system in this example, we have selected a sensor with the following characteristic values:

- Sensitivity 10 mV/g
- Evaluation range up to 50g

The resulting amplification factor is fa = 20 and the amplified voltage signal of the sensor is +/-10 V.



MV Measuring amplifier (MV = "Messverstärker")

Fig.8-3: Connection of a CSH01.1C Control Section

Configuring the Analog Input

To allow processing the sensor signal in the MLD as a feedback, the value of the analog input must be mapped to a PLC register.

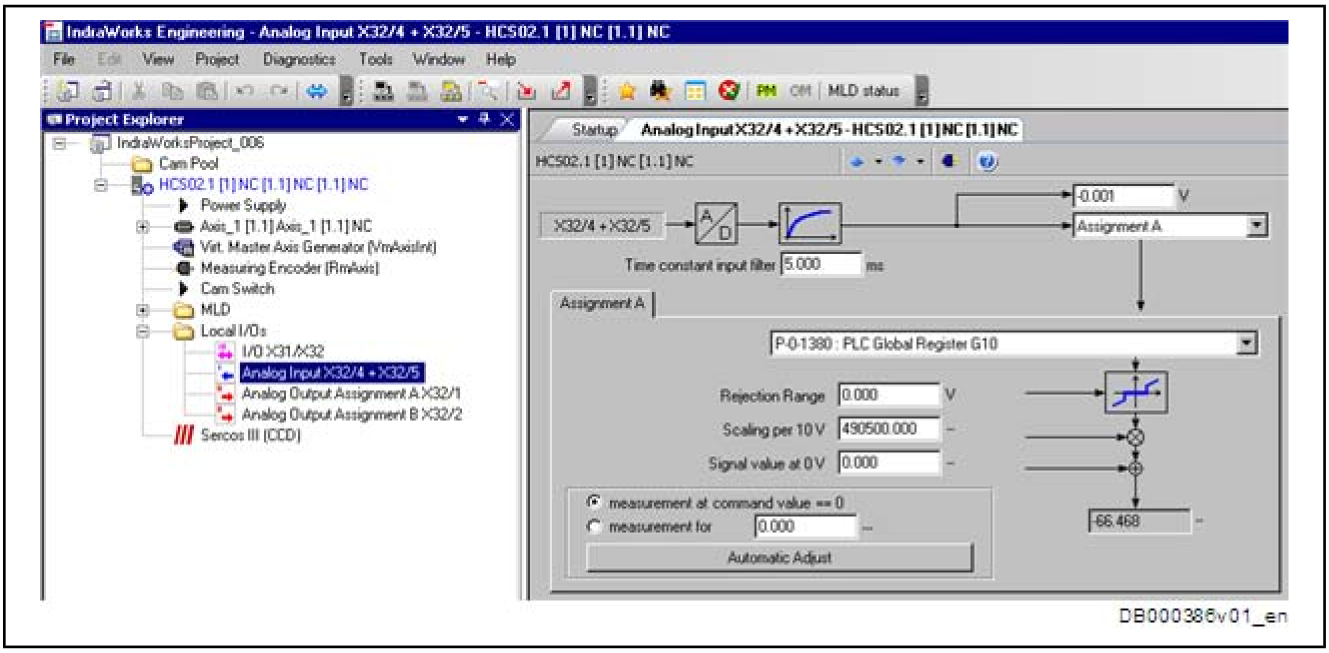


Fig.8-4: IndraWorks Dialog for Configuring the Analog Input of the Control Section

Settings in the dialog:

- Time constant input filter → The input signal is filtered via a PT1 low pass. At this point, the user can parameterize a limit frequency for the filter.
- Assignment type → Assignment A
- Target parameter → Select any unassigned PLC register as the target parameter. In the described example, the parameter P-0-1380 is to be used (the display format of the register can be set in the parameter P-0-1386).

- Rejection Range → Set the rejection range to 0 V.
- Scaling per 10 V → This value depends on the sensor, measuring amplifier and the parameterized units. The paragraph below describes how to derive the value.
- Signal value at 0 V → Select the value such that 0 V is displayed in the target parameter at an acceleration of "0g". The value differs from zero when the applied signal has an offset.

How to derive the "Scaling per 10 V"

The derivation is described for our application by the example of a system with linear scaling (position in mm, velocity in mm/min) which uses the above-mentioned sensor. It therefore makes sense to display the acceleration value of the sensor in the parameter P-0-1380 in mm/s².

Measuring amplifier output:

$$\frac{10\ \text{mV}}{g} \times 20 = \frac{200\ \text{mV}}{g}$$

Acceleration value; scaled in mm/s^2:

$$1 \times g = 9{,}81\ \frac{\text{m}}{\text{s}^2} = 9810\ \frac{\text{mm}}{\text{s}^2}$$

$$\frac{200\ \text{mV}}{9810\ \frac{\text{mm}}{\text{s}^2}} = 2{,}039 \times 10^{-5}\ \frac{\text{V}}{\frac{\text{mm}}{\text{s}^2}}$$

Voltage output:

$$0{,}00002039\ \text{V} = 1\ \frac{\text{mm}}{\text{s}^2}$$

$$10\ \text{V} = x\ \frac{\text{mm}}{\text{s}^2}$$

$$0{,}00002039\ \text{V} \times \frac{10}{0{,}00002039} = 1\ \frac{\text{mm}}{\text{s}^2} \times \frac{10}{0{,}00002039}$$

$$10\,\text{V} \approx 490500\ \frac{\text{mm}}{\text{s}^2}$$

The result is:

$$\frac{10\,V}{490500\,\frac{mm}{s^2}}$$

Using the value "490 500" in the dialog window "Scaling per 10 V" you achieve that the acceleration value is displayed in the desired unit "mm/s^2".

IndraLogic

At the beginning of the project creation, the MLD-S library "DRIVE_LIB_01V02.lib" is included in the IndraLogic library manager by right-clicking the library list.

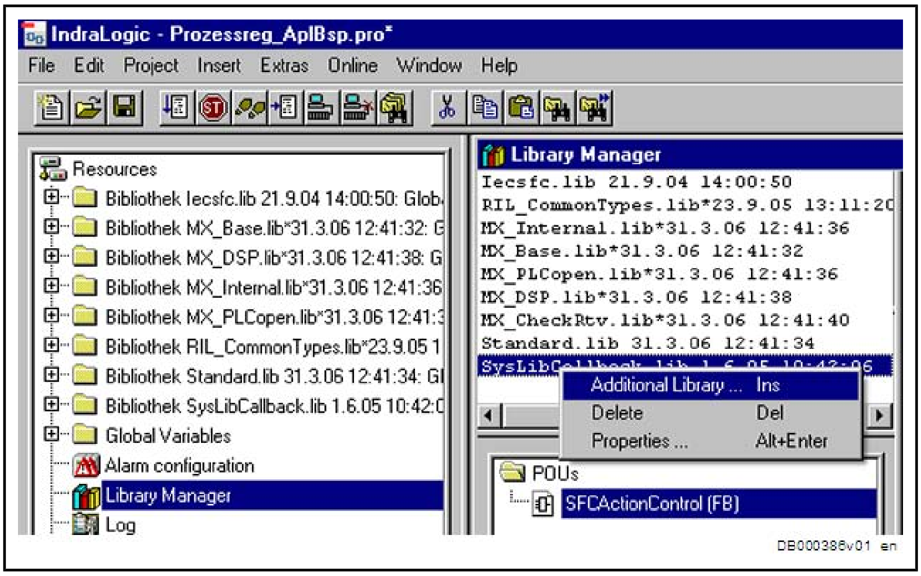


Fig.8-5: Including the MLD-S Library in IndraLogic

This library contains the function block "MX_PID_Regler" on which process control is based.

Restrictions

For vibration damping, the process control is superimposed to the drive control loop structure which means that "Motion Control" still is in the external PLC. When the actuating variable of the process control is an additive command value for one of the inner control loops, it actually is an interference to the outer loop. Due to the higher dynamic response of the inner control loops, it can be of advantage, however, to preset an additional velocity command value, although the axis is in position control.

Process control is only as good as the feedback signal it is based on; i.e. proceed with diligence and accuracy when installing the sensor and adjusting the signal.

Vibration Damping With Superimposed Process Loop (Process Control With Intelligent Servo Axis)

8.3 Programming

8.3.1 System Structure

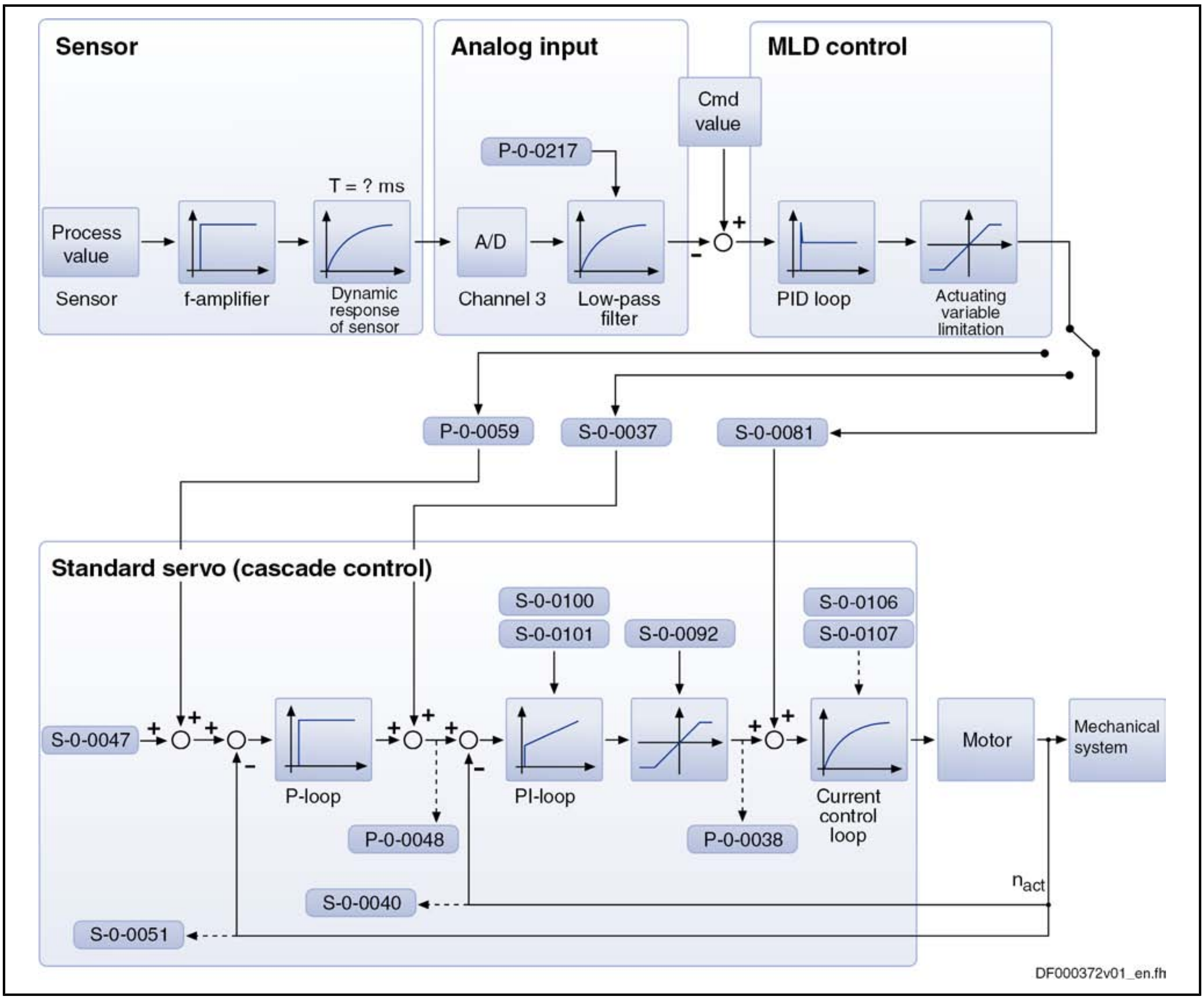


Fig.8-6: System Structure

In our example, we use an axis which is operated in position control. The actuating variable of the process loop, which is to suppress/compensate unwanted vibration, is an additive command value to the outermost control loop. In this case, the parameter "P-0-0059, Additive position command value, controller" consequentially is the target parameter of the actuating variable. In this case, the command value for process control would be the twofold derived position profile which is preset by the external PLC. With infinite stiffness, this would simultaneously be the ideal acceleration profile at the sensor's point of installation.

8.3.2 Funktion Block "MX_PID_Regler"

Interface Description

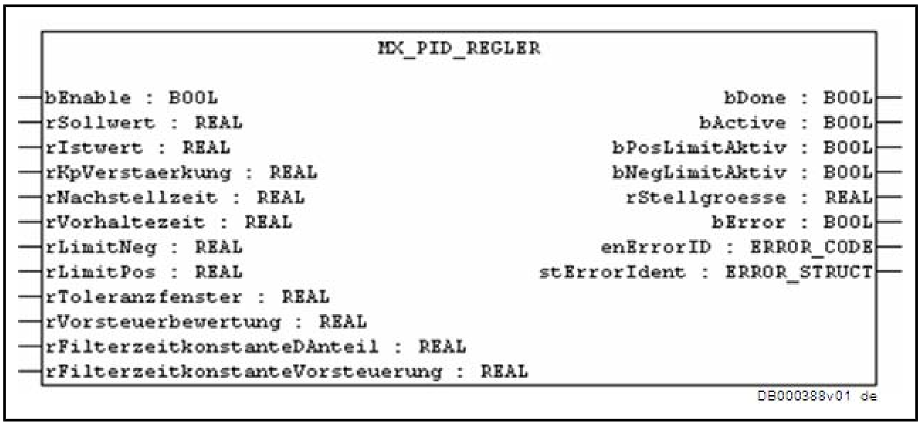


Fig.8-7: Interfaces of the Function Block "MX_PID_Regler"

	Name	Type	Description
VAR_INPUT	bEnable	BOOL	Sets control enable
	rSollwert	REAL	Command value of the control variable
	rIstwert	REAL	Actual value of the control variable
	rKpVerstaerkung	REAL	P-gain of the PID loop
	rNachstellzeit	REAL	Integral action time (I-component of the PID loop) in ms
	rVorhaltezeit	REAL	Derivative action time (D-component of the PID loop) in ms
	rLimitNeg	REAL	Negative limitation of the controller output
	rLimitPos	REAL	Positive limitation of the controller output
	rToleranzfenster	REAL	Tolerance window for "bDone" message
	rVorsteuerbewertung	REAL	Feedforward from command value difference
	rFilterzeitkonstante D-Anteil	REAL	Filter time constant for PT1 filter in D-component (in ms)
	rFilterzeitkonstante Vorsteuerung	REAL	Filter time constant for PT1 filter in feedforward (in ms)
VAR_OUTPUT	bDone	BOOL	Actuating variable > tolerance window
	bActive	BOOL	Controller active
	bPosLimitActive	BOOL	Positive limitation active
	bNegLimitActive	BOOL	Negative limitation active
	rStellgroesse	REAL	Actuating variable at controller output
	bError	BOOL	Error in function block
	enErrorID	INT (ENUM)	Rough error information (only when "bError = TRUE")
	StErrorIdent	ERROR_ STRUCT	Detailed error information (only when "bError = TRUE")

Fig.8-8: Interface Description of the Function Block

Functional Description

With the function block "MX_PID_Regler", the control of internal and external values has to be carried out via a process loop superimposed to drive control.

The function block "MX_PID_Regler" generally is a PID loop with feedforward function and PT1 filters for feedforward and the D-component.

By means of the tolerance window, you can set a range in which the output gets the state "Done True" when the actuating variable is smaller than the tolerance window.

Switching the controller off by "Enable = False" sets all internal variables and the actuating variable to zero.

Error Handling

In the case of error, this function block generates a detailed diagnosis. For a precise description of the error code which is output (Additional1, Additional2), see the table below.

ErrorID	Additional1	Additional2	Description
INPUT_RANGE_ERROR	16#0C01	16#1	Preset negative limitation is greater than positive limit
	16#0C01	16#2	Kp-gain smaller than "0"
	16#0C01	16#3	Integral action time smaller than sampling time or smaller than "0"
	16#0C01	16#4	Derivative action time smaller than sampling time or smaller than "0"
	16#0C01	16#5	Filter time constant D-component smaller than sampling time or smaller than "0"
	16#0C01	16#6	Filter time constant feedforward smaller than sampling time or smaller than "0"

Fig.8-9: ERROR_TABLE

8.3.3 Accessing Drive Parameters

The control is based on process data which can be generated from the drive via parameter access. The screenshots below show the function blocks "Read_cyclic()" and "Write_cyclic()". They illustrate how to have read access and write access to the drive parameters.

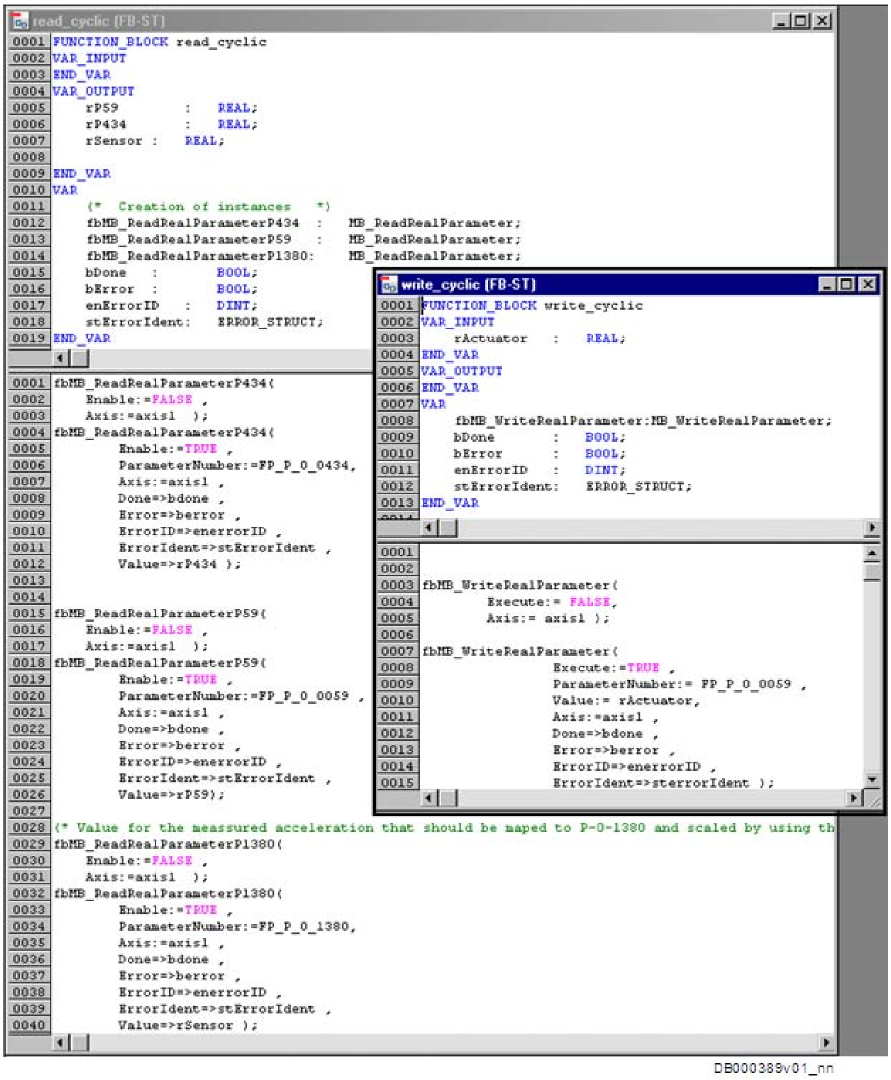


Fig.8-10: Read Access and Write Access

These parameters are of particular interest:

- P-0-0434, Position command value of controller (READ)
- P-0-0059, Additive position command value, controller (READ and WRITE)
- P-0-1380, PLC Global Register G10 (READ) → scaled sensor signal

8.3.4 Generating the Command Value Characteristic

The generation of the command value characteristic takes the parameters "P-0-0434, Position command value of controller" and "P-0-0059, Additive position command value, controller" into account. From the values of these two parameters, IndraMotion MLD determines an acceleration command value characteristic which is only preset by the control unit, i.e. without process control taking effect.

```
FUNCTION_BLOCK FB_CommandValue
VAR_INPUT
    rSamplingTime: REAL;
    rP434: REAL;
    rP59: REAL;
END_VAR
VAR_OUTPUT
    rCommand: REAL;
END_VAR
VAR
    rPos_Command_old: REAL;
    rPos_Command_cur: REAL;
    rVel_Command_old: REAL;
    rVel_Command_cur: REAL;
    rPos_diff: REAL;
END_VAR
```

```
(*  Generation of command value  *)

rPos_Command_old := rPos_Command_cur ;
rPos_Command_cur := rP434 - rP59 ;
rVel_Command_old := rVel_Command_cur;
rPos_diff:=(rPos_Command_cur - rPos_Command_old) ;
(*  Generation and Convertion Velocity command: [mm/ms] to [mm/min]*)
rVel_Command_cur:=(rPos_diff)*60/rSamplingTime*1000;
(*  Generation and Convertion Acceleration command: [mm/min²] to [mm/s²] *)
rCommand:=(rVel_Command_cur - rVel_Command_old)/60/(rSamplingTime/1000);
```

DB000390v01 nn

Fig.8-11: Generating the Acceleration Command Value

The output variable "rCommand" represents the acceleration command value which was derived by the position input of the external PLC.

8.3.5 Overall Structure of Process Control

Process control is activated by setting, through the Boolean "bStart", the Enable input of the instance of "MX_PID_Regler" to TRUE (both user-defined function blocks and standard function blocks must be created in the declaration section as instances). For this case, the controller synthesis has shown that the compensation of the disturbance, without any permanent deviation of drive control, is only possible via "P-0-0059, Additive position command value, controller" or "S-0-0037, Additive velocity command value", when you use a process controller without an integral component. In the following paragraphs, a mere P-controller will be implemented. For this reason, the resulting PLC program structure is as follows:

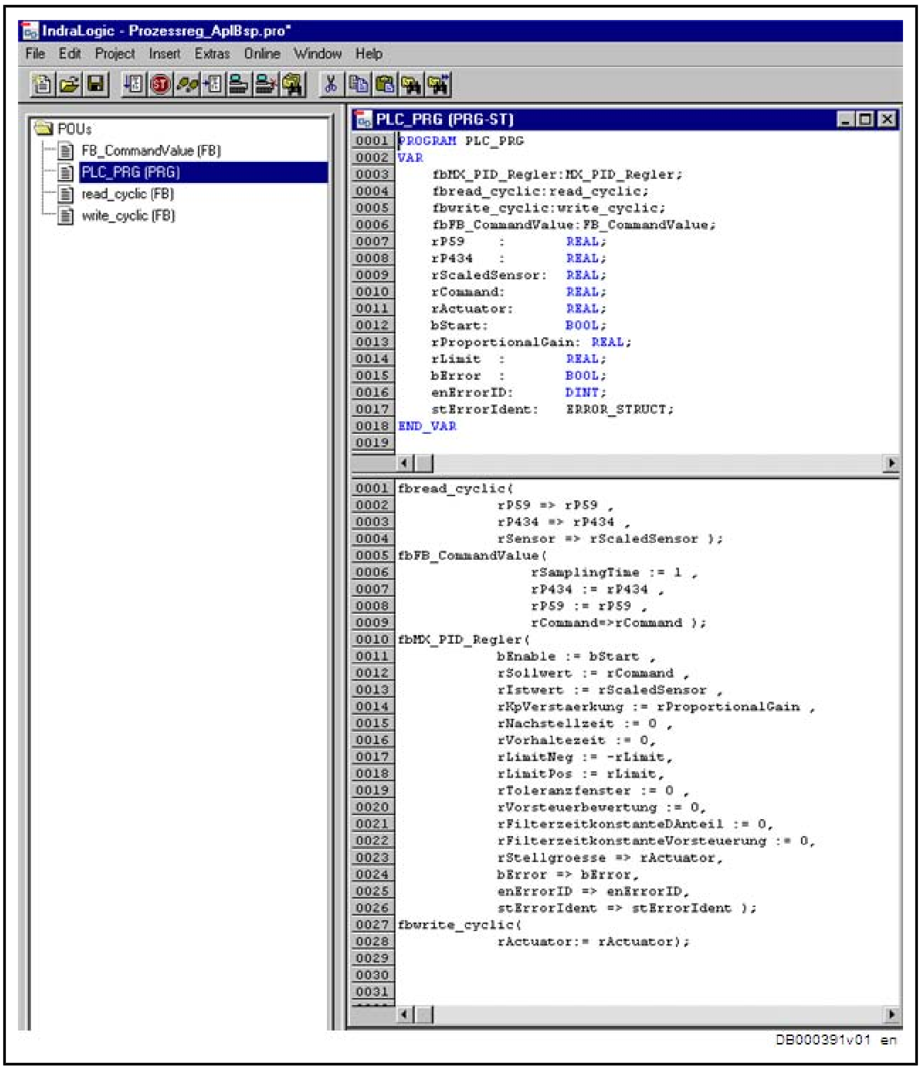


Fig.8-12: PLC Program

In the "PLC_PRG" program, the instances of the user-defined function blocks "read_cyclic", "FB_CommandValue" and "write_cyclic", as well as the instance of the function block "MX_PID_Regler" from the "DRIVE_LIB_01V02.lib" library are called. It is now only necessary to append the program to a cyclic task to be defined. The minimum interval time of the cyclic task depends on the control section type and the configured performance setting.

Vibration Damping With Superimposed Process Loop (Process Control With Intelligent Servo Axis)

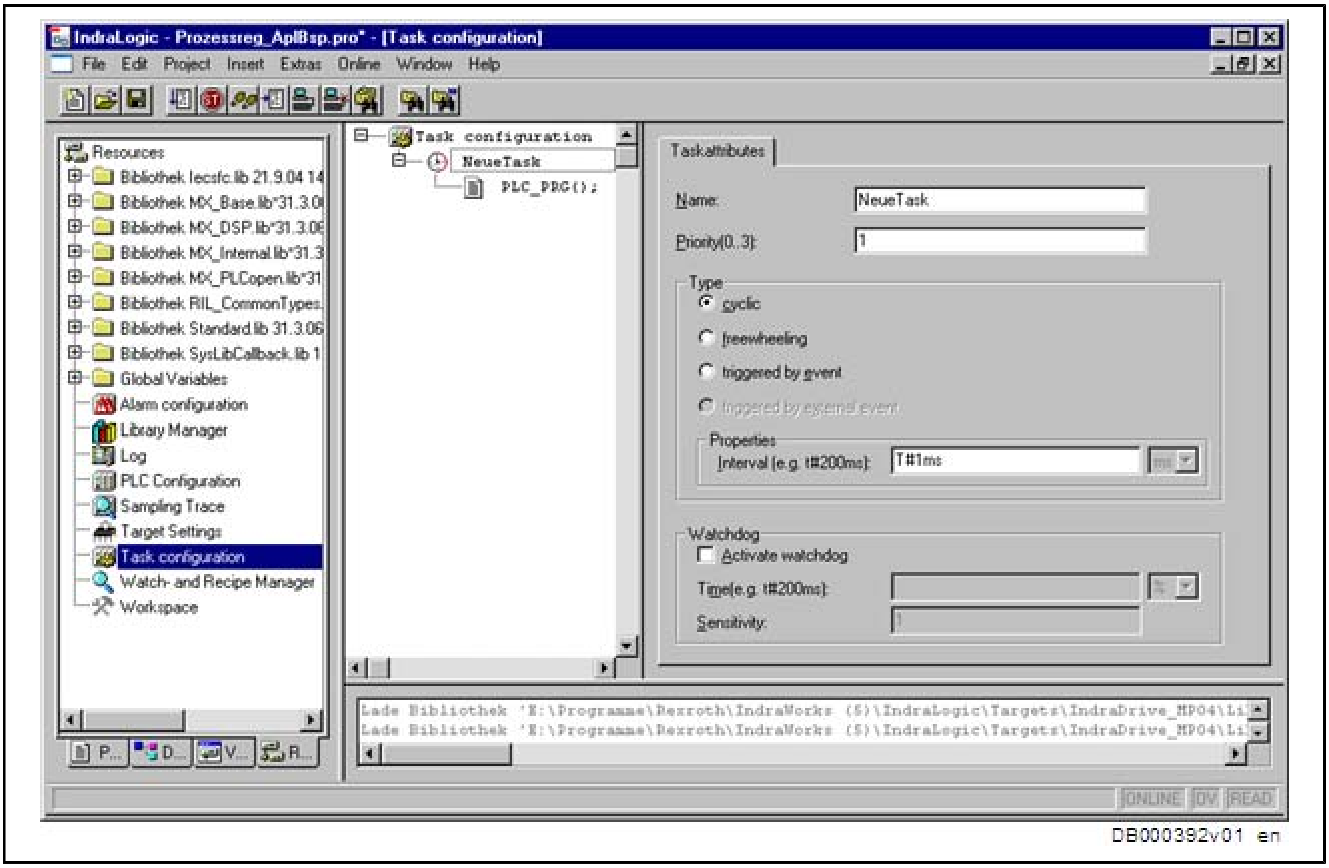


Fig.8-13: Task Configuration

Via the IndraLogic menu under "Online → Login", the task configuration can be loaded to the PLC with the program call and started with "Online → Start".

8.3.6 Visualization and Diagnostics

The results can be visualized by means of the drive-internal oscilloscope function. The user interface of this function can be found in the IndraWorks menu under "Diagnosis → Start Oscilloscope".

8.4 Commissioning and Testing

How to Proceed

Follow the steps below to commission the vibration damping:

- Connect the (amplified) sensor signal to the IndraDrive control section
- Adjust the sensor signal by means of the IndraWorks dialog of the analog input
- Embed the "DRIVE_LIB_01V02.lib " library and program MLD for the desired type of control

After you have carried out the steps listed above as described in the previous chapters, it is now necessary that you adjust or optimize the superimposed process controller. When setting the control loop proportional gain, take the orientation of the axis acceleration with regard to the coordinate orientation of the sensor into consideration in order not to cause positive feedback by the control loop. In our case, the coordinate systems have been equally oriented; therefore, it is necessary to set a positive proportional gain.

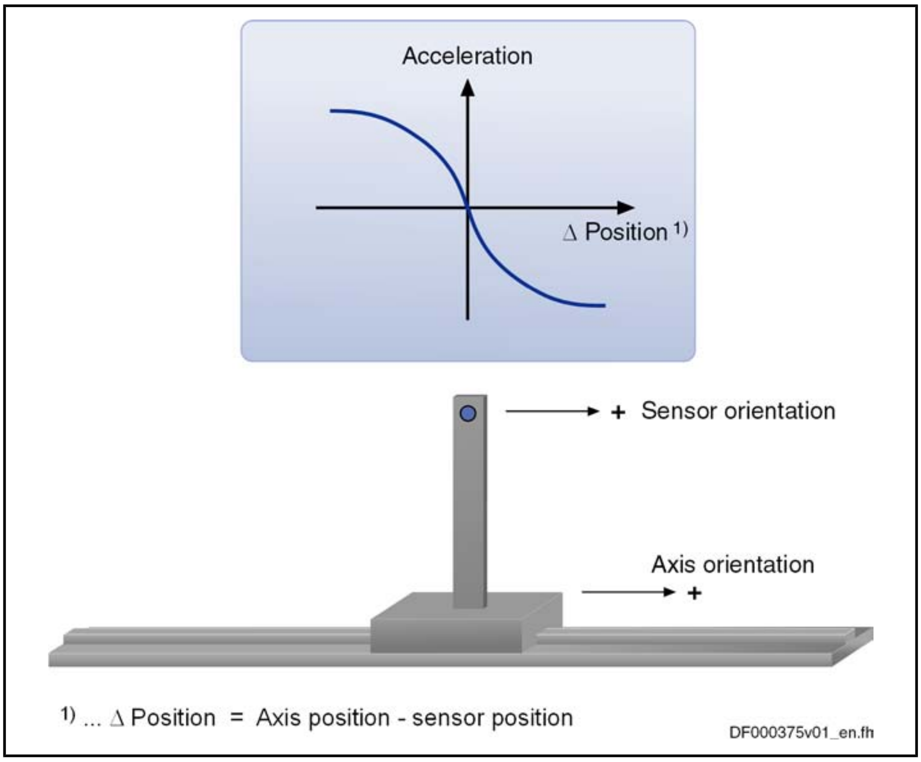
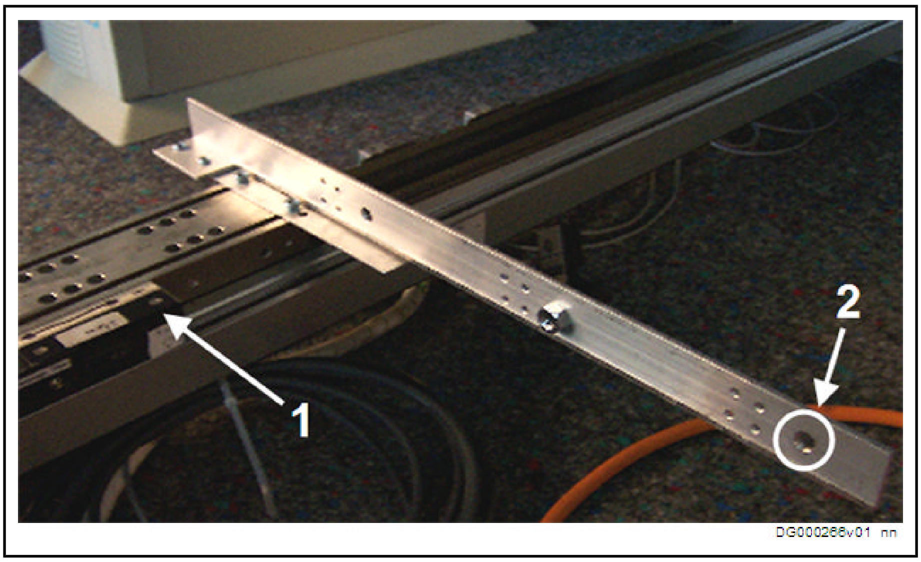


Fig.8-14: Coordinate Orientation

With the pictured setup, the process control for vibration damping was implemented and the compensation of the parasitic vibration recorded by means of the oscilloscope function at the place of installation of the acceleration sensor.



1 Linear slide
2 Sensor
Fig.8-15: Mechanical Model

8.5 Visualization and Diagnostics

The oscilloscope recording below shows the acceleration command value and the actual acceleration value. It additionally shows the actuating variable of the process loop ("P-0-0059, Additive position command value, controller") and the actual position value of the slide. Triggering took place at the point of time when process control had been switched on.

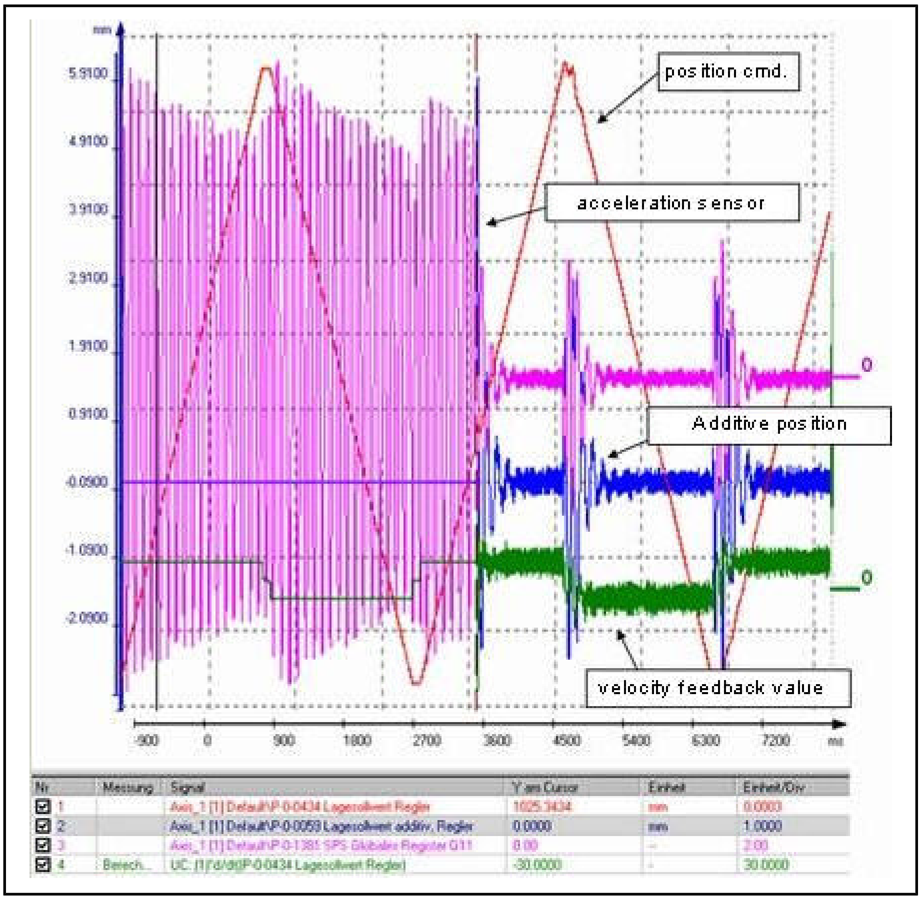
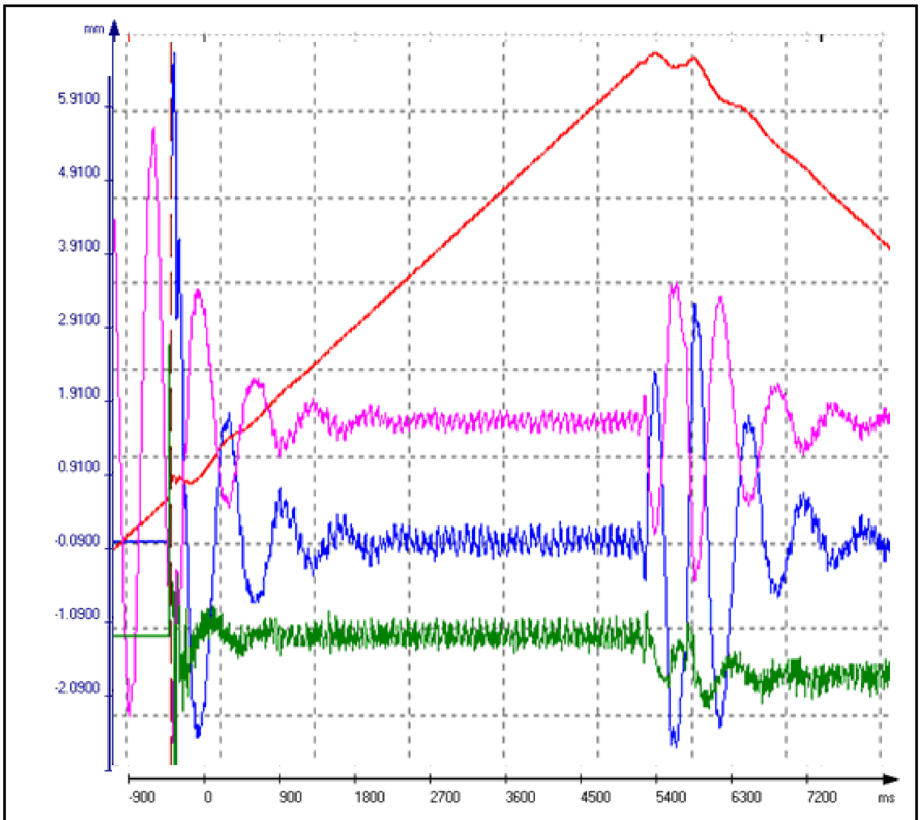


Fig.8-16: Signal Sequence After Process Control Switched On



Nr	Messung	Signal	Y am Cursor	Einheit	Einheit/Div
☑ 1		Axis_1 [1] Default\P-0-0434 Lagesollwert Regler	1025.3434	mm	0.0003
☑ 2		Axis_1 [1] Default\P-0-0059 Lagesollwert additiv, Regler	0.0000	mm	1.0000
☑ 3		Axis_1 [1] Default\P-0-1381 SPS Globales Register G11	0.00	--	2.00
☑ 4	Berech...	UC: [1]*d/dt([P-0-0434 Lagesollwert Regler)	-30.0000	-	30.0000

DB000394v01_de

Fig.8-17: Detailed View of the Signal Sequence

9 Service and Support

Our service helpdesk at our headquarters in Lohr, Germany and our worldwide service will assist you with all kinds of enquiries. You can reach us **around the clock - even on weekend and on holidays**.

	Helpdesk	Service Hotline Worldwide
Phone	+49 (0) 9352 40 50 60	Outwith Germany please contact our sales/service office in your area first. For hotline numbers refer to the sales office addresses on the Internet.
Fax	+49 (0) 9352 40 49 41	
E-mail	service.svc@boschrexroth.de	
Internet	http://www.boschrexroth.com You will also find additional notes regarding service, maintenance (e.g. delivery addresses) and training.	

Preparing Information

For quick and efficient help please have the following information ready:

- Detailed description of the fault and the circumstances
- Information on the type plate of the affected products, especially type codes and serial numbers
- Your phone, fax numbers and e-mail address so we can contact you in case of questions.

Index

Index

Notes

Rexroth
Bosch Group

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Tel. +49 (0)93 52-40-0
Fax +49 (0)93 52-48 85
www.boschrexroth.com/electrics

R911324034

Printed in Germany
DOK-INDRV*-MLD-APPLI**-AW02-EN-P