The Basics of ASiM

TM410

Perfection in Automation

www.br-automation.com

Requirements

Training modules:	TM210 – The Basics of Automation Studio
	TM400 – The Basics of Drive Technology
Software:	Automation Studio 3 or higher
Hardware:	B&R drive system (controller, network, ACOPOS, B&R motor)

Table of contents

1. INTRODUCTION

The B&R drive concept is **fully integrated in Automation Studio**. As a result, a complete and simple project configuration is possible as well as the implementation of control and positioning tasks.

In this training module we will cover the individual steps needed to create a drive configuration in an Automation Studio project. We will get to know the various tools and interfaces used to manage and configure the individual components.

Fig. 1: Automation Studio with integrated drive concept

It will also be important for us to take a look at the interaction of the individual components.

Diagnostic tools provide an efficient environment for tests and implementation.

1.1 Training guide objectives

Participants will learn the procedure for creating a drive configuration in Automation Studio.

Participants will also become familiar with the main components of the B&R drive concept and will know the communication relationships.

You will be able to use the diagnostics tool NC-Test to send commands to the ACOPOS and simulate specific tests.

You will learn how to use NC-Trace to record specific values for analyzing positioning procedures.

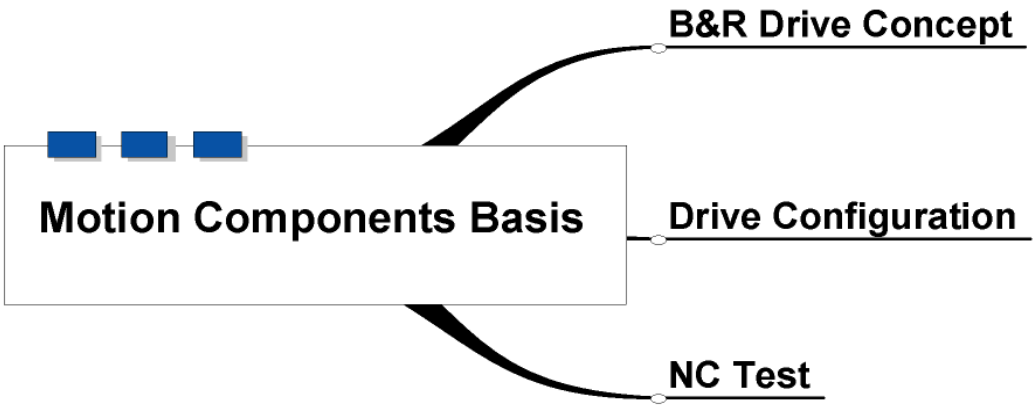


Fig. 2: Overview

2. B&R DRIVE CONCEPT

In the following section we will take a look at the **system architecture of the B&R drive solution**. Our goal is to gain a basic understanding of the procedures and relationships in this system. This knowledge will also be useful later on when we cover how to operate the ACOPOS servo drives.

Which components make up the B&R drive solution?

A "physical" (= hardware-based) representation of the system configuration looks something like this:

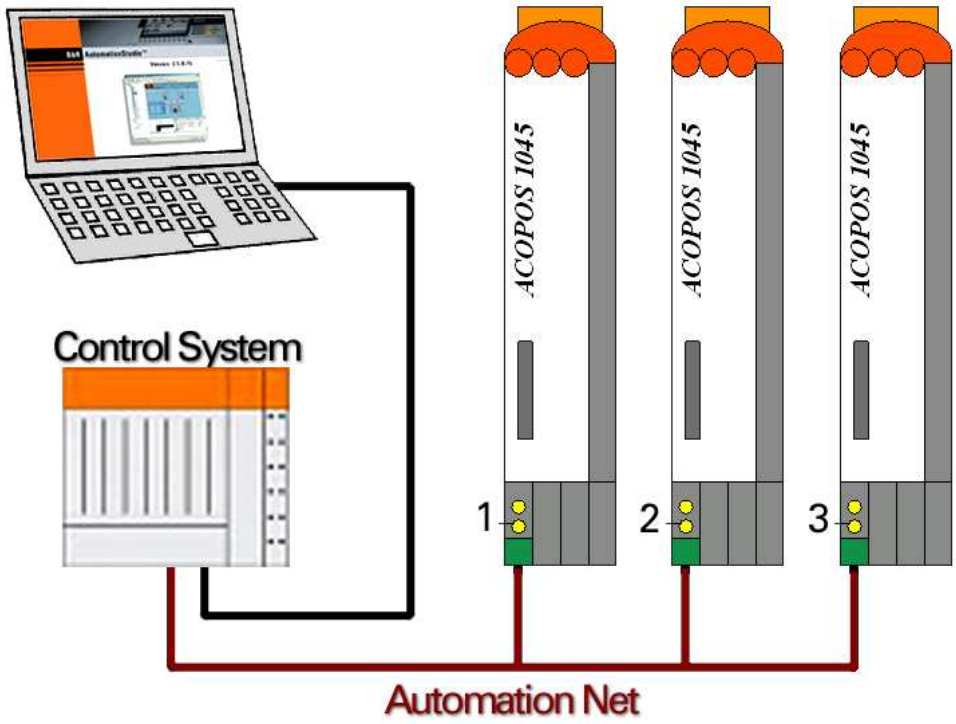


Fig. 3: "Physical" configuration, hardware

The system includes one or more **ACOPOS servo drives** that are connected with a **B&R controller** via a specific **network** (CAN or POWERLINK).

Configuration takes place using **B&R Automation Studio**. Various tools are available to do this, which can be divided into the following **main categories**:

- **Hardware configuration**, because the right components must be correctly applied to the different configurations. These settings are generally made using Automation Studio.
- **Software generation**, for controlling the drive and implementing the positioning sequences using an application program. For the B&R drive solution, the software can be generated on different user interfaces to meet the user's requirements (PLCopen Motion Control library, ACOPOS: ACP10 software).
- **Diagnostic and test utilities**, are also provided to the user in **Automation Studio** with specific setups (NC Test) and functions. For example, certain processes (e.g. positioning) can be accurately adjusted or values can be recorded for analyses.

The components for a drive configuration were shown in the figure above. To understand how these components work together, we will now take a brief look at the individual components and will then combine them in a common communication concept.

2.1 ACOPOS servo drives

A servo drive has a large number of parameters for making specific settings.
Configuring parameters is nothing more than defining settings.

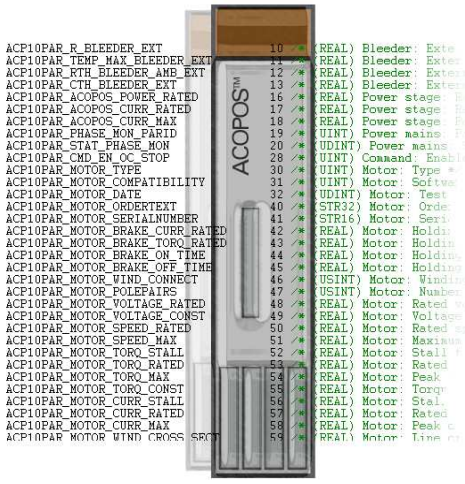


Fig. 4: "The ACOPOS has a large number of parameters"

Essentially, these parameters can be divided into two groups:

Parameters for the hardware configuration

The ACOPOS must be set up for the connected hardware:

- Characteristics of the motor and holding brake (if present)
- Insert cards on the ACOPOS (encoder and network cards)

Parameters for positioning sequences

The ACOPOS has parameters that are used to control and to check positioning sequences:

- Movement parameters, commands and status values
- Monitors

Note:

Configuration of drive motors and encoder systems is supported by a wizard in Automation Studio. Fully pre-defined parameter sets can be used with B&R motors.

The ACOPOS servo drive has an **internal logic**. This internal logic is called the **NC operating system**.
It manages the many parameters on the ACOPOS and contains all of the components used for positioning.

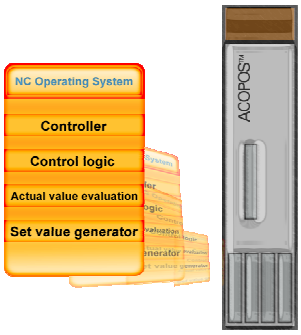


Fig. 5: NC operating system on the ACOPOS

Note:

The various servo drive types in the ACOPOS family have different power elements (control power). The control logic is completely identical. This means that the configuration of different ACOPOS types (1022, 1045, etc.) is the same for the user.

Note:

What does "**NC**" stand for?
The prefix "NC" **(Numerical Control)** is used in Automation Studio for all software components that are used on the PLC to operate Motion Hardware (ACOPOS, etc.). These components are handled using Motion Components in Automation Studio.

2.2 Control and communication

As explained earlier, there are many parameters available for operating a drive on the ACOPOS. A clearly arranged user interface was developed to put all of these parameters into perspective; the **NC software**.

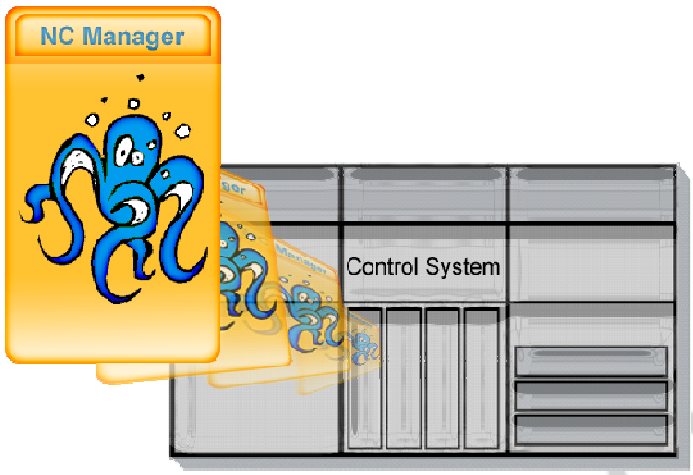


Fig. 6: NC Manager on the controller

The **NC Manager** is the **link** between the **user application** and the servo drive's **NC operating system**.
An **operating structure** (NC data structure) is provided for configurations and for operating the respective drive object.
When using the NC Manager, commands are sent using simple functions. **Further communication to the NC operating system** for executing the respective actions is then handled by the NC manager.

This means that the user does not have to tediously configure the numerous parameters in the servo drive. The user is provided with a clearly structured user interface and, as we will see later, simple functions for sending commands.

How is data exchanged?

Coordinated data transfer is required to execute the functions of a drive (value initializations, positioning movements, etc.).
Relevant information is exchanged **between the controller (NC manager) and the servo drive (NC operating system) via a network**.

The B&R drive solution uses both network types **ETHERNET POWERLINK** and **CAN** (**C**ontroller **A**rea **N**etwork).

Two types of data are transferred:

- **Parameter data and commands** are sent from the controller to the servo drive.
- **Status data** is sent from the servo drive to controller.

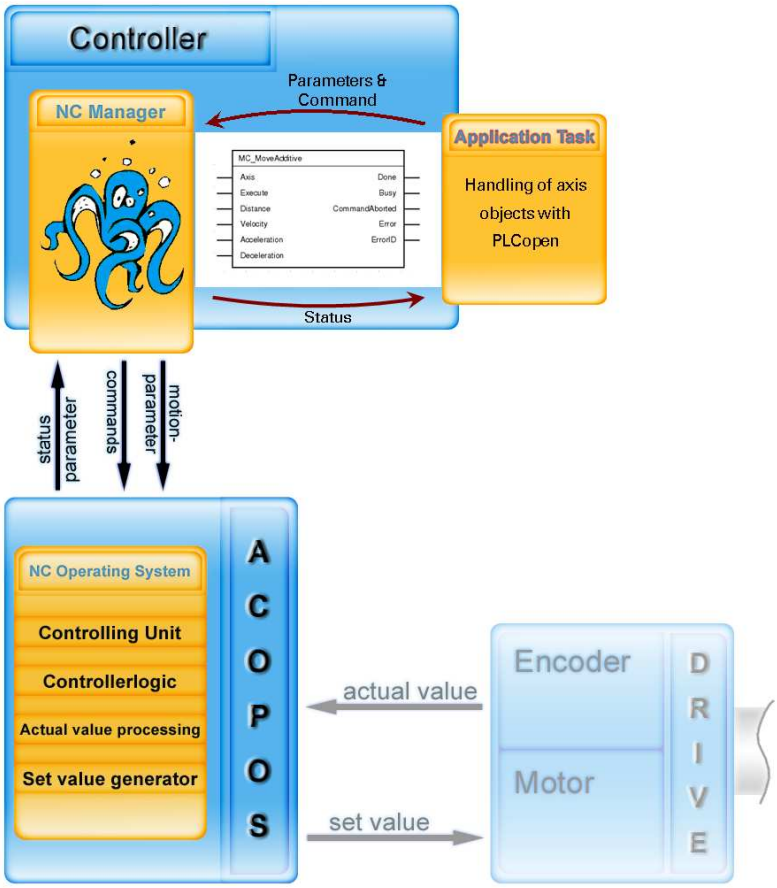


Fig. 7: Communication between NC Manager and ACOPOS

Note:

For example, an absolute target position can be specified as a positioning parameter. A corresponding command ("absolute positioning") is then used to start the action. The servo drive executes the desired action with the corresponding parameters. There is always information about the current status of the drive on the controller for monitoring the process (e.g. query: "drive at target position?").

2.2.1 Operating possibilities

The user has two possibilities for operating a drive:

- Standard **PLCopen motion control function blocks**
- **ACP10 software** (B&R-specific interface)

ACP10 software (B&R specific interface):

This operating mode offers a structured and well-organized interface for controlling drives.
All of the drive data (parameters and status values) is made available in a **data structure** (NC structure). This provides the user with clearly arranged access to all of the important settings for the drive.
Commands are sent to the drive via **NC actions**. The drive then returns **status data** indicating whether the action has been executed correctly.

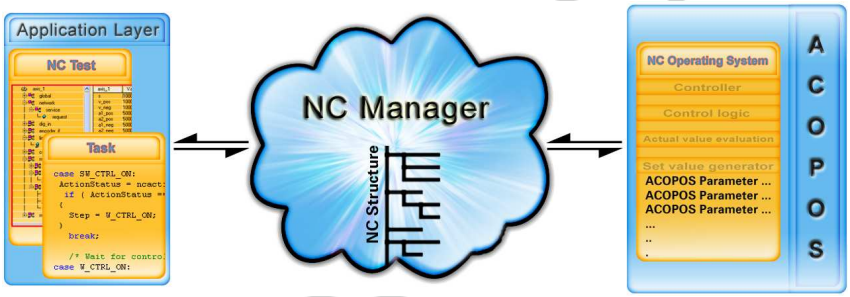
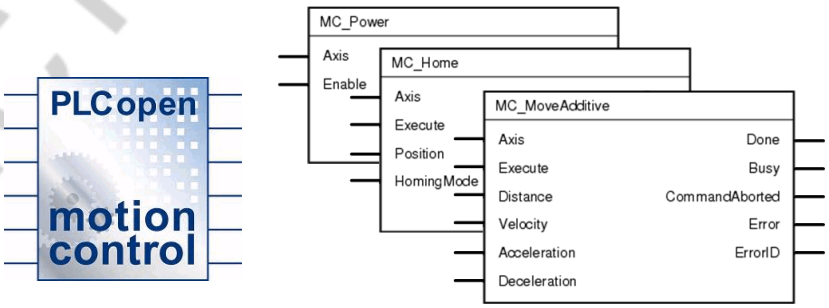


Fig. 8: Drive operation with ACP10 software, data structure

Standardized PLCopen motion control function blocks:

The **ACP10_MC library** provides function blocks for drive control. The function blocks are organized in clear categories. Each function block offers the setting of all parameters which are relevant for an action (e.g. "relative movement") and returns the respective status information.



Drive operation via PLCopen Motion Control:

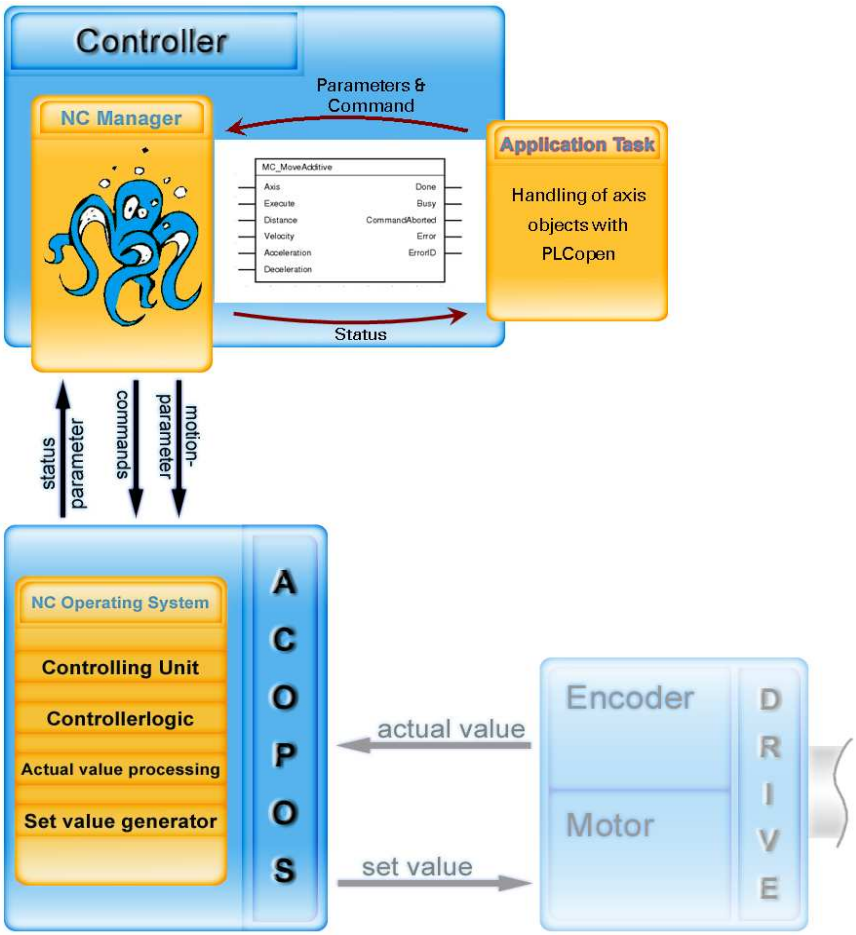
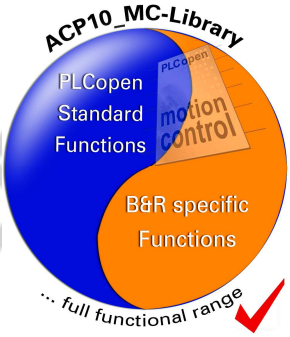


Fig. 9: Drive operation via PLCopen Motion Control

In addition to the function blocks that are defined for drive control in the PLCopen Motion Control Standard, B&R also offers **specific function blocks** for special ACOPOS functions (e.g.: cam profile automat, etc.).

These function blocks are operated the same way as the standard functions.
This enables full utilization of the entire ACOPOS range of functions.



Detailed information about operating the ACP10_MC function blocks is provided in the training modules **"TM440 Motion Control Basic Functions"** and **"TM441 Motion Control Multi-axis Functions"**.

2.3 Software concept

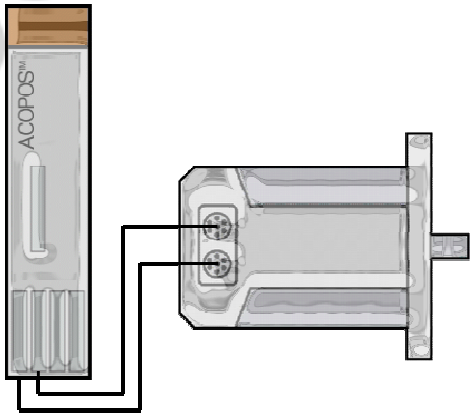
The object-oriented approach is based on the **NC objects**.
These software objects represent "real objects" on the controller. Every NC object has an **operating structure**.

There are a few different NC objects:

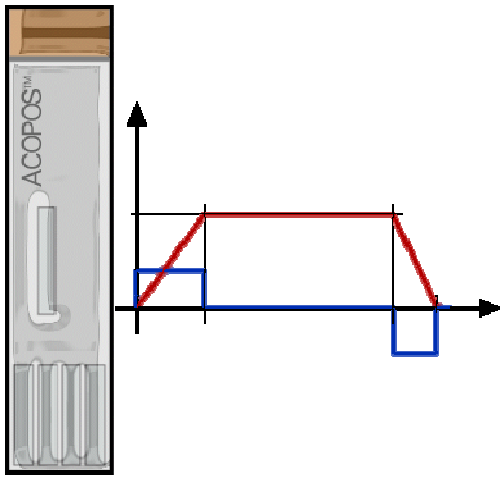
- Real axis
- Virtual axis
- CNC system

The **NC Manager** manages these NC objects.
It sends the **commands that have been made together with the defined parameters** from the operating structure to the corresponding ACOPOS servo drive, which the object was assigned to. It also ensures that the corresponding status values are returned to the structure of the respective object.

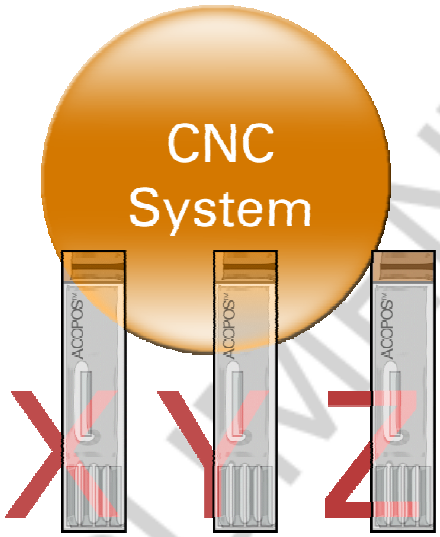
→ **Real axis:** This NC object is used to operate a real drive with motor and position encoder. All components needed for this are provided in the operating structure of the NC object.



→ **Virtual axis:** In addition to the real axes, the ACOPOS also offers the option of operating a virtual drive. This drive works solely as a type of "set value generator" (i.e. generates values for position and speed) and is operated identically to a real axis. This can be used for various applications like linking a real axis to the position of a virtual axis.



→ **CNC System**: A CNC system is an application, in which multiple axes are linked with each other and perform a common, defined movement. The CNC system handles the management and coordination of the individual CNC axes.



The operating structure of NC objects differs according to the different areas of application where they are used. For example, a virtual axis requires fewer operating parameters than a real axis because it does not have encoder or control parameters.

3. DRIVE CONFIGURATION

The following section will deal with adding a drive to a project and configuring the settings. We will also take a closer look at the tools used in Automation Studio for managing the drive objects.

> **Note:**
>
> In automation technology, a drive object is often referred to as "axis object" or "axis".

3.1 Inserting an axis

A hardware configuration with the following structure must be integrated in Automation Studio:

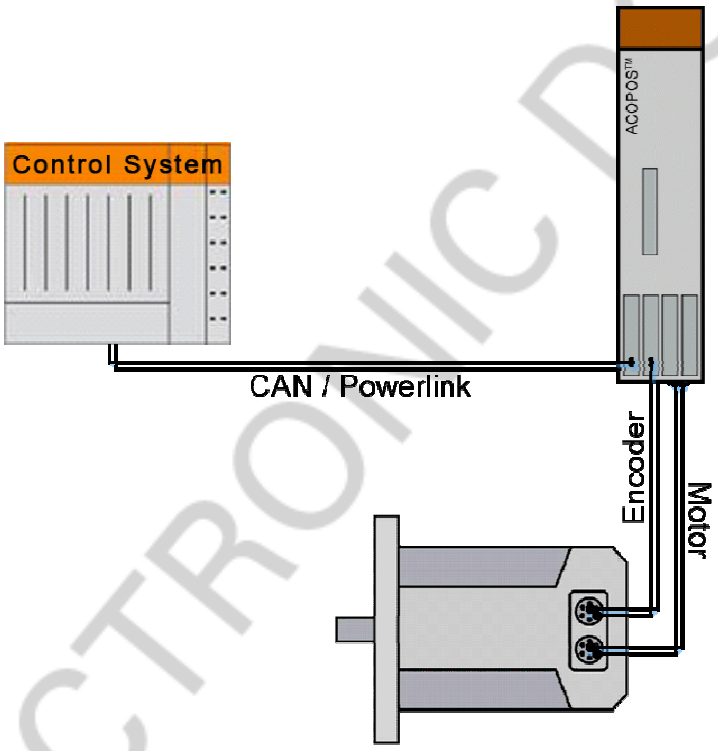


Fig. 10: Drive configuration

An ACOPOS can be added by **right-clicking** on the **respective interface card** (**CAN or POWERLINK**). This is shown here using a CAN interface.

The desired interface for inserting the ACOPOS must first be selected and opened.

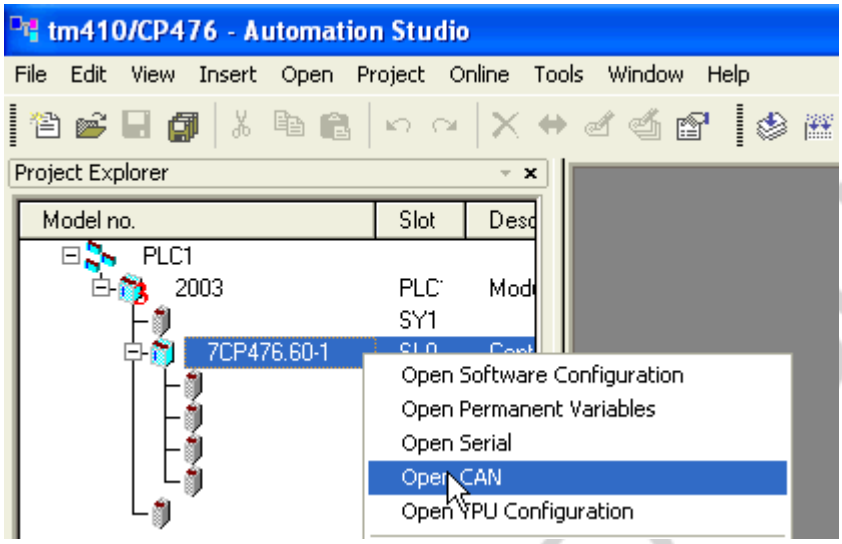


Fig. 11: Opening the desired interface

Right-clicking on the desired interface and selecting **Insert** opens the dialog box for selecting the supported components for this interface.

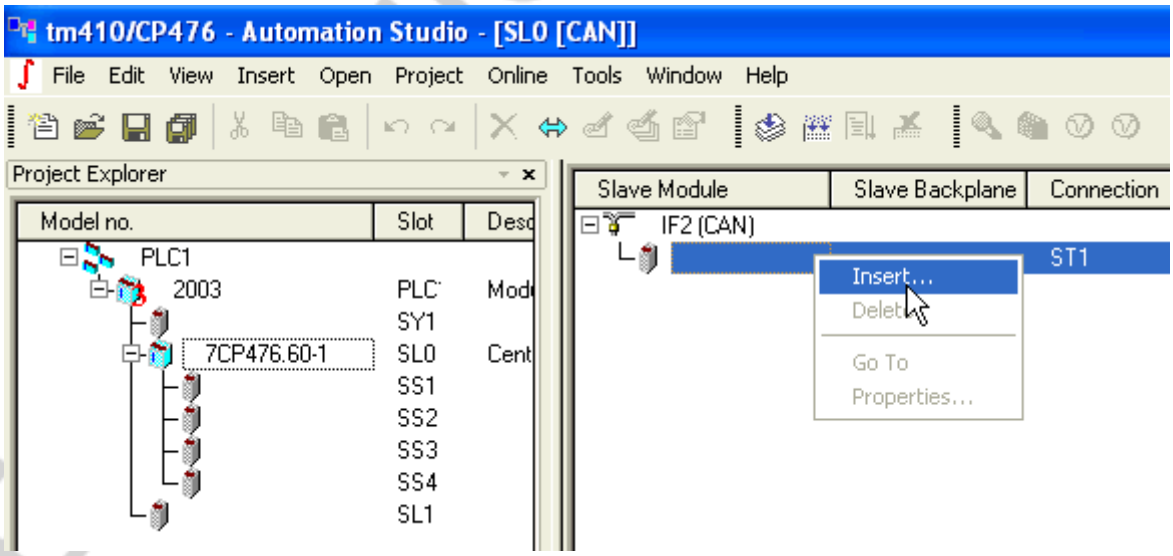


Fig. 12: Inserting hardware

The corresponding ACOPOS servo drive is selected from this dialog box.

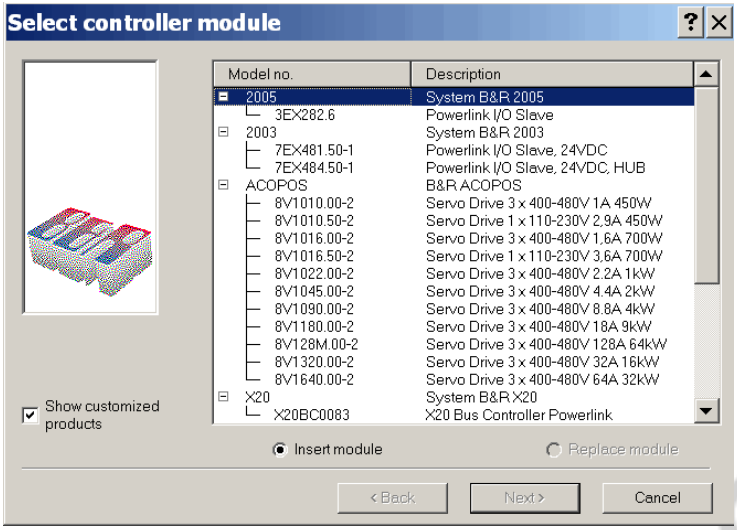


Fig. 13: Selecting the hardware components

The node number is specified in the next step.
The node number is important for uniquely identifying the respective ACOPOS on the bus. Therefore, the node number set on the interface card must be entered:

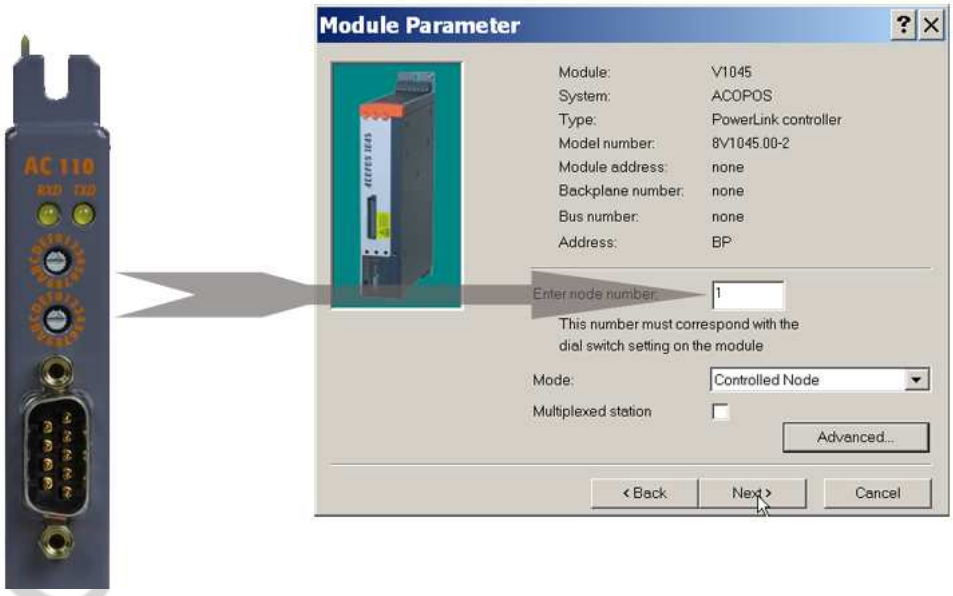


Fig. 14: Setting the node number

Note:

The node number on the ACOPOS interface cards should be set with **hexadecimal**, but specified in Automation Studio as **decimal**.

The next dialog box is used to specify the first insert card on the ACOPOS (in this case, CAN insert card):

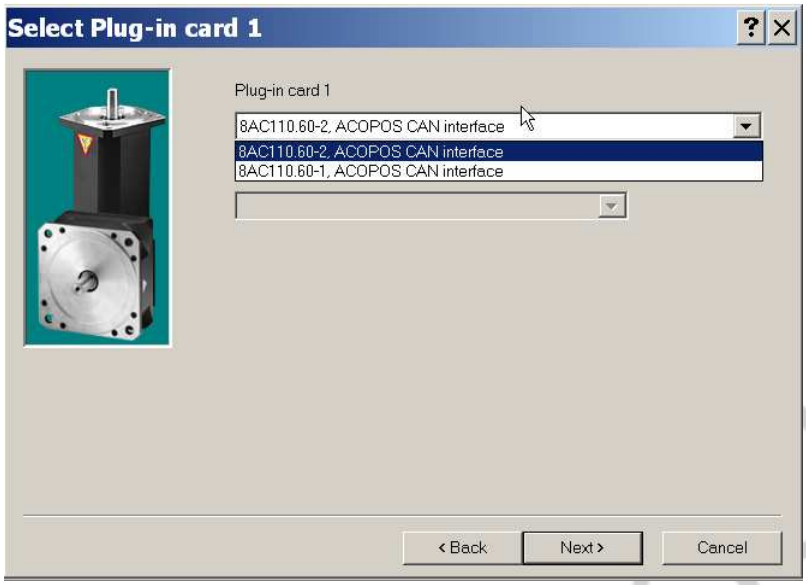


Fig. 15:Exact definition of the CAN insert card on the ACOPOS

Clicking on **Next** will continue with the wizard.

The second insert card on the ACOPOS is specified in this dialog box. We will select the EnDat interface.

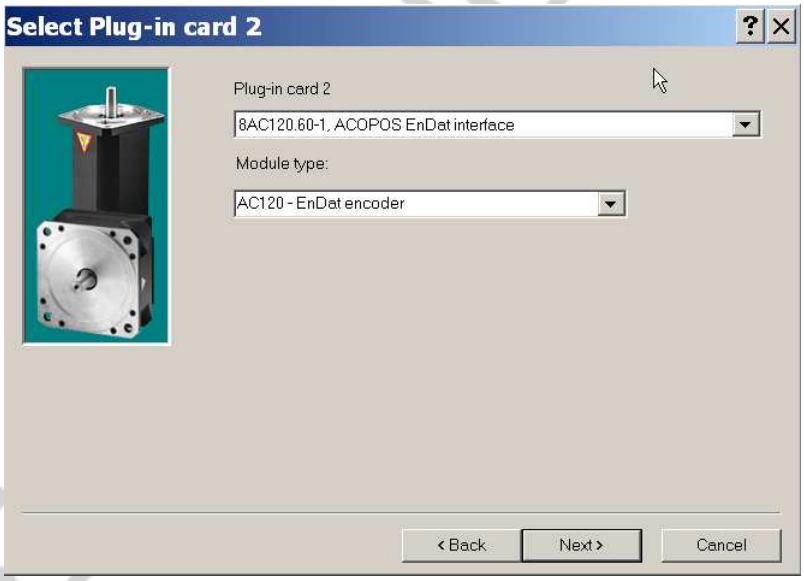


Fig. 16: Definition of second insert card on the ACOPOS

The next dialog box is opened by clicking **Next**.

In addition to determining the intended use based on CNC or standard positioning tasks, the following dialog box can also be used to determine whether **PLCopen** should be used for programming. In this case, the required **ACP10_MC library** is automatically imported to the project.

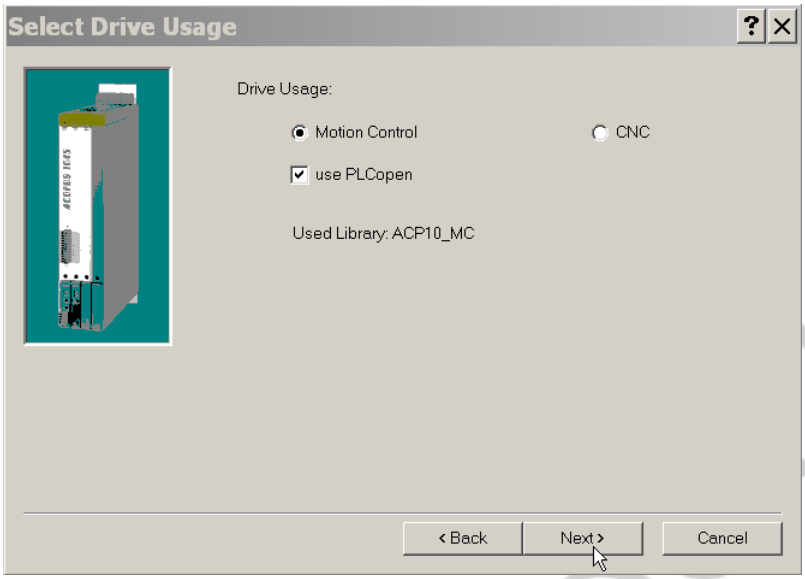


Fig. 17: ACOPOS intended usage

Clicking on **Next** will continue with the wizard.

The **motor** must still be determined in the following step. Certain aspects of the subsequent approach will vary depending on the selected motor.

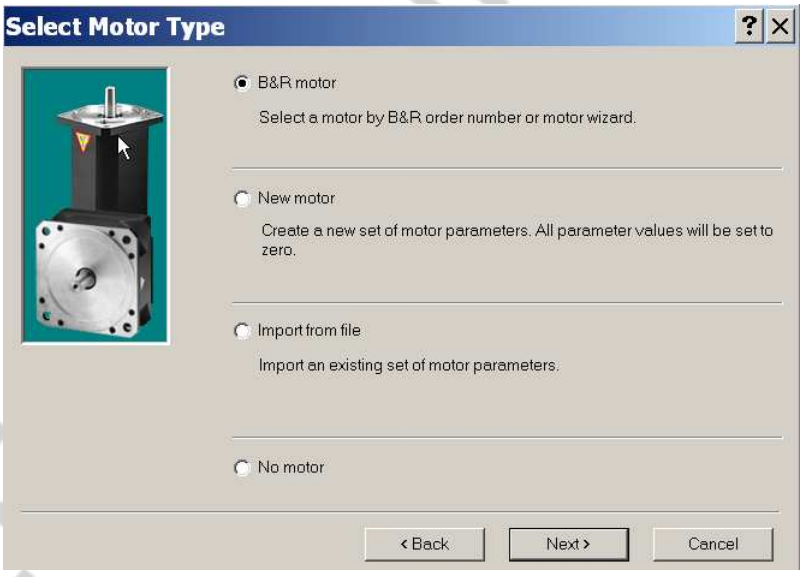


Fig. 18: Dialog box for determining the motor

Clicking on **Next** will continue with the wizard.

Possibilities for defining the motor:

- When using a **B&R motor with EnDat encoder**, definition of the motor or configuration of the motor's electrical characteristics for the ACOPOS can be skipped by selecting **Finish**.

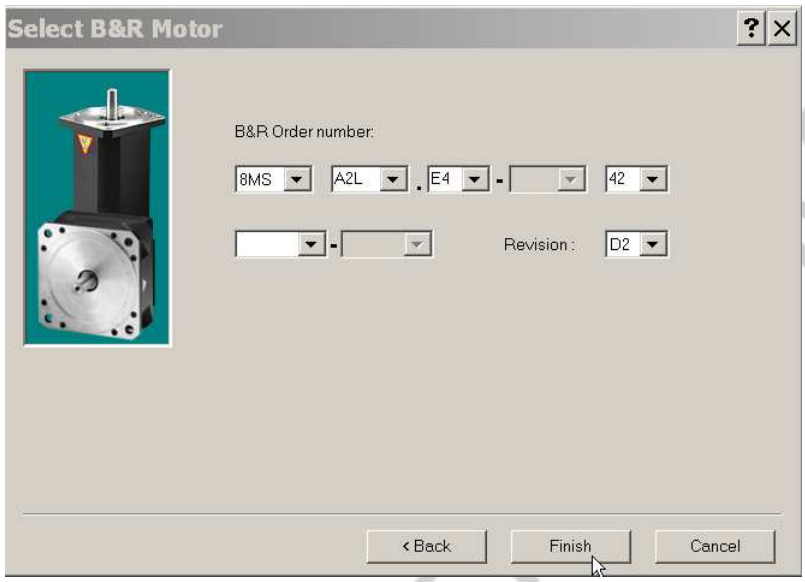


Fig. 19: Selecting the respective B&R motor

EnDat encoders have what is called an **embedded parameter chip**. All relevant motor parameters are stored in the EnDat encoder's electronic memory. The values required for defining the motor are automatically read by the ACOPOS during power-up.

- The motor definition cannot be skipped **when using a B&R motor together with another encoder system** (without embedded parameter chip). In this case, the exact motor name must be entered in the dialog box shown above using the **B&R model number** (type number shown on the type plate of the B&R motor), before the hardware configuration can be completed by pressing **Finish**.

  The pull-down menus should be set to the appropriate values.

  Automation Studio contains data for all of the motors in the B&R product palette.

- When **using a motor from another manufacturer** (with an encoder system without embedded parameter chip), the motor type must be defined first (synchronous/induction) and then the required parameters (motor data sheet, dimensions). Detailed information on this topic is covered in the training module **"TM460 Starting up Motors"**.

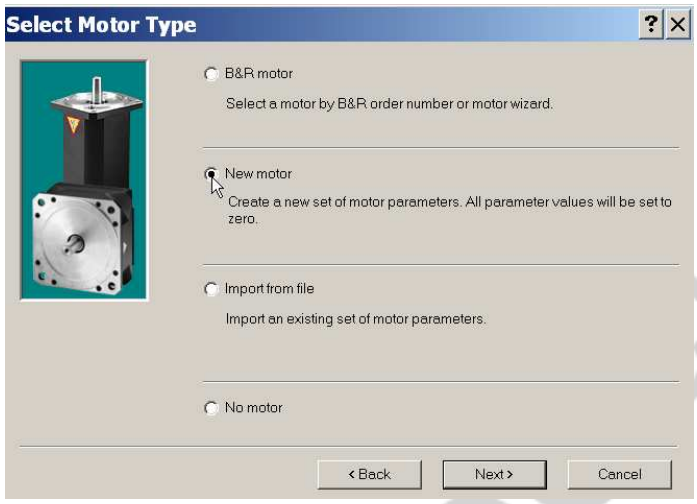


Fig. 20: Selecting a motor from another manufacturer

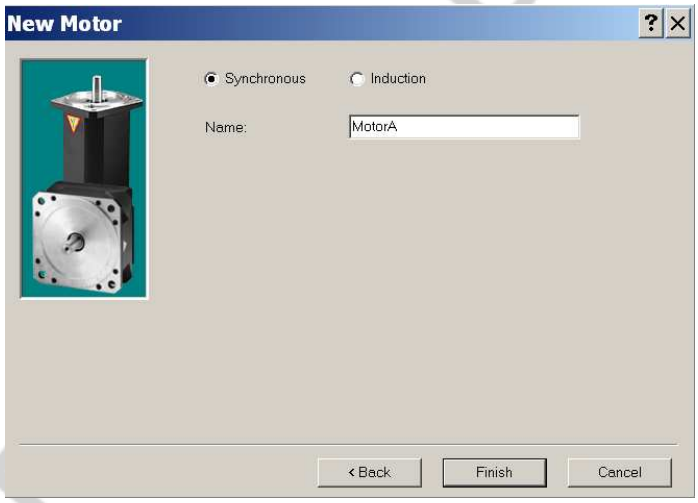


Fig. 21: Differentiating between synchronous and induction motors

After being completed, the hardware configuration is applied to the project together with the parameters from the motor definition.

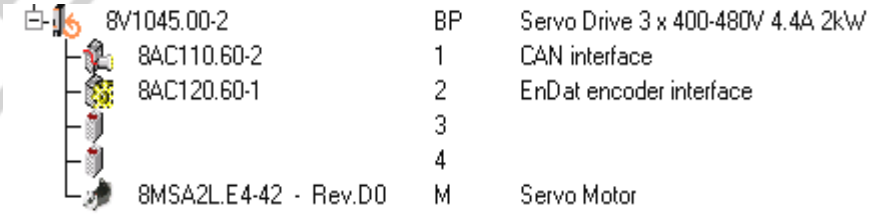


Fig. 22: Inserted ACOPOS in the hardware tree

Note:

When an Ethernet POWERLINK connection is used, Automation Studio checks the defined cycle time. ACOPOS requires this time value to be a **multiple of 400µs**:

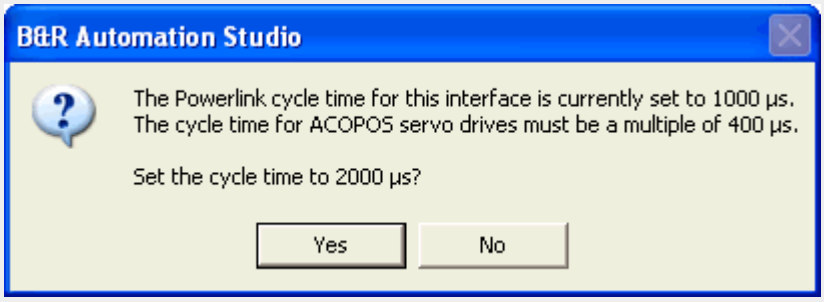


Fig. 23: Automatic definition of the Ethernet Powerlink cycle time

When using the POWERLINK connection, the **POWERLINK cycle should be used as timer for the task class system**. This setting can be made in the **CPU settings** (select the tab **Timing –→System Timer**).

3.1.1 ACOPOS hardware configuration:

It is also possible to expand or alter the hardware configuration of an ACOPOS servo drive. The following options are available:

Inserting ACOPOS insert cards:

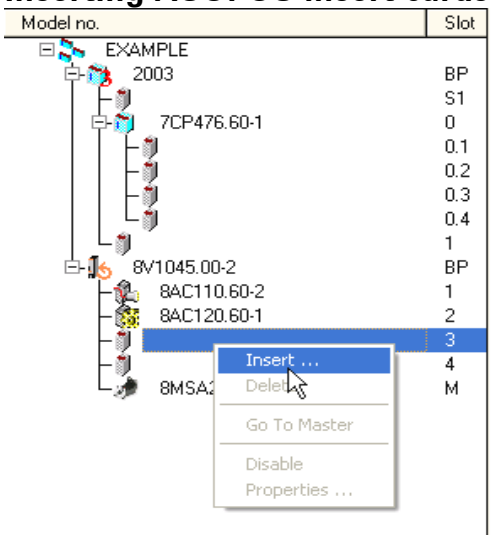


Fig. 24: Inserting an ACOPOS module

e.g.

- EnDat interface
- Resolver interface
- etc. ...

Adding a motor:

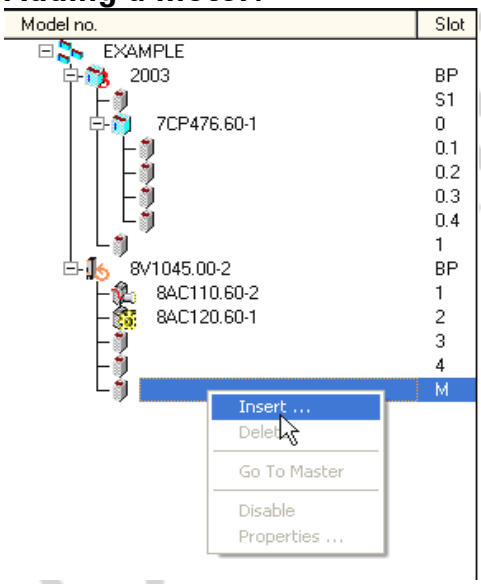


Fig. 25: Inserting a motor

3.1.2 Software objects

All necessary **software components** are automatically added to the project when inserting the first ACOPOS for specific NC software:

Type / name	Description
System objects	
acp10man	NC manager, operating system expansion (controller)
acp10sys	NC operating system for the ACOPOS (the user can also decide to store this on the controller)
acp10cfg	Configuration for the NC software
Libraries	"Operating the NC objects"
acp10man	Data types for the NC structure
acp10par	Constants for ACOPOS parameters
ncglobal	NC functions and constants for NC command codes and NC parameter values.

Note:

If, for example, an ACOPOS is added to a POWERLINK interface for the ACP10 software, then the **"powerlnk" library** is also added if it does not already exist in the project. It is required for the operation of the POWERLINK network.

3.2 INIT parameter module

An **INIT parameter module** (= NC software object) can be added to the project for each axis type (real axis / virtual axis) to define **fixed settings (initial settings or defaults) for basic parameters** (e.g.: maximum acceleration, controller parameters, etc.).

3.2.1 Inserting an NC software object in Automation Studio

A new object can be added by **right-clicking** on the folder with the project name in the **Logical View** and selecting **Add Object**.

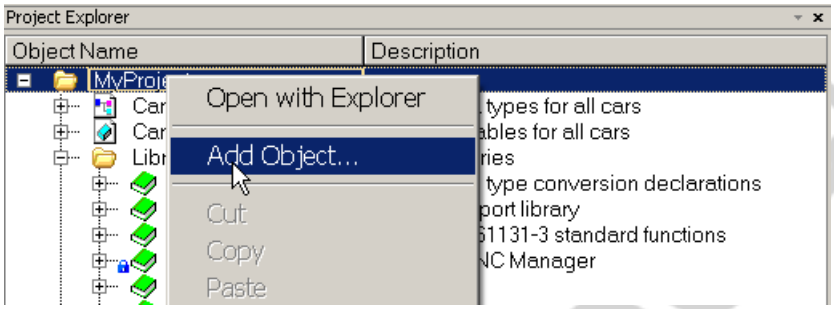


Fig. 26: Inserting an NC software object

The corresponding category **Motion** is selected under **Categories**. A new INIT parameter module can then be selected from the right half of the dialog window by choosing **New NC INIT Parameter Axis**.

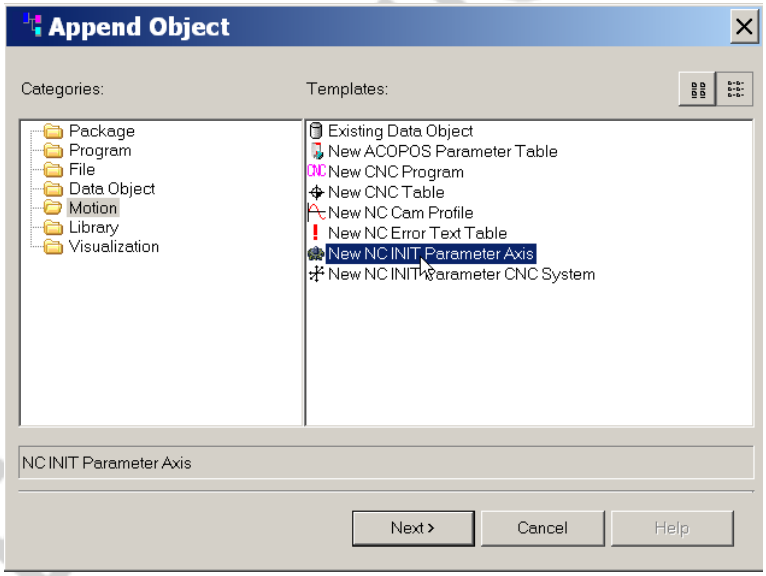


Fig. 27: Selecting a new INIT parameter module

This dialog box can be used to enter a **name** and a **description** for the new INIT parameter module.

To insert an INIT parameter module for a real axis, we will choose the **subtype ACP10:Axis**.

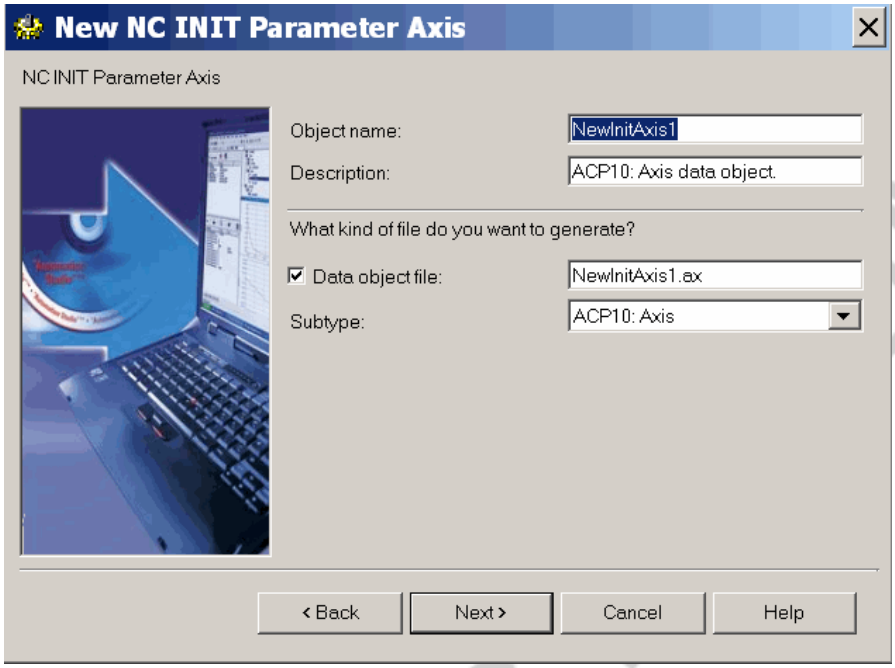


Fig. 28: Entering a name and description for the INIT parameter module

We will assign the INIT parameter module **to the active CPU**.

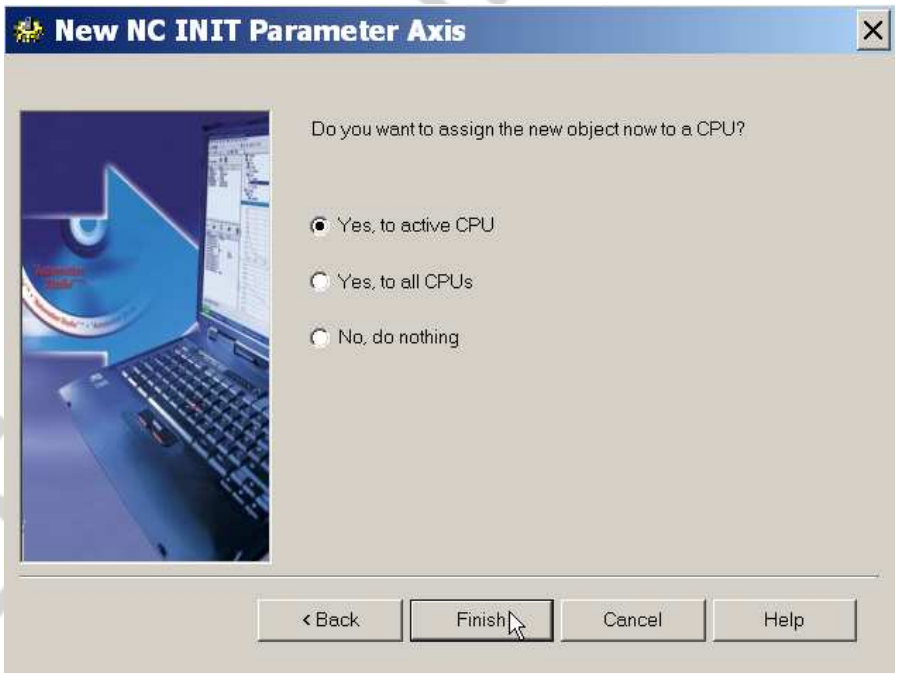


Fig. 29: Assigning an inserted INIT parameter module to the active CPU

After finishing this dialog box, the INIT parameter module can be opened and edited by **double-clicking** on the name of the corresponding module in the tree structure of the the **Logical view**.

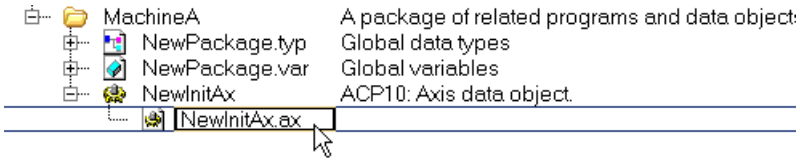


Fig. 30: Opening an INIT parameter module for editing

The following image shows the structure of the INIT parameter module. All basis axis parameters, such as parameters for a basis movement are "pre-defined".

Name	Value	Unit	Description
ACP10AXIS_typ			
dig_in			Digital Inputs
encoder_if			Encoder Interface
limit			Limit value
controller			Controller
move			Movement
stop			Stop Movement
parameter[0]			Parameter record
parameter[1]			Parameter record
parameter[2]			Parameter record
parameter[3]			Parameter record
homing			Homing procedure
basis			Basis movements
parameter			Parameters
v_pos	10000	Units/s	Speed in positive direction
v_neg	10000	Units/s	Speed in negative direction
a1_pos	50000	Units/s²	Acceleration in positive direction
a2_pos	50000	Units/s²	Deceleration in positive direction
a1_neg	50000	Units/s²	Acceleration in negative direction
a2_neg	50000	Units/s²	Deceleration in negative direction
message			Messages (errors, warnings)

Fig. 31: Structure of the INIT parameter module

The values set in this structure are then applied to the data structure (NC structure) of the NC object during each system startup and can be initialized on the ACOPOS (i.e. enabled for ACOPOS operation) using a command ("Global Init").

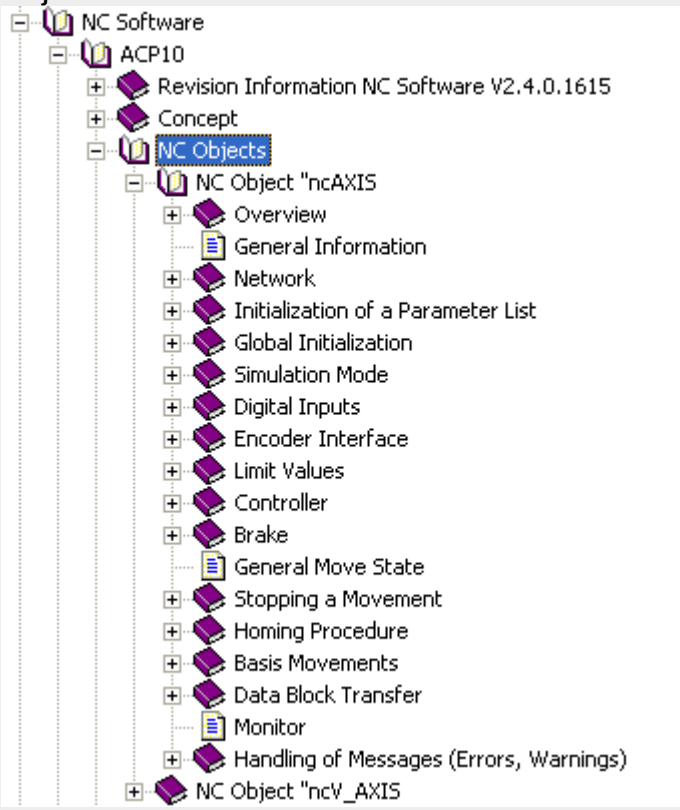
Note:

If changes should be made to the INIT parameter module in Automation Studio, it must be saved in the project. The changed INIT parameter module values are not valid on the ACOPOS until the project has been transferred (with the **"Global Init" command**).

Notes:

Assignment of an INIT parameter module to an NC object (real axis, virtual axis, etc.) is variable. For example, a drive (NC object "real axis") can be assigned another INIT parameter module using a corresponding dialog box.

Detailed information about all subgroups and parameters for the NC objects can be found in the Automation Studio online help.



Task: Inserting an axis

Use the method described above to insert an INIT parameter module for a real axis in your project.

3.3 Settings for the NC software

Assigning INIT parameter modules to axes:

Double-clicking on the ACOPOS in the hardware tree will open an **NC Mapping Table** where the INIT parameter modules can be assigned for the axes. The INIT parameter modules that have already been created in the project are offered.
A new module is created by entering a new name.

Because an ACOPOS has a real axis as well as a virtual axis, an INIT parameter module can be assigned here for both axes.

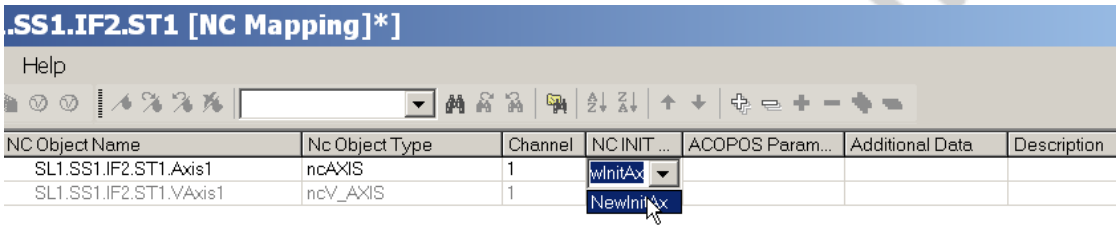


Fig. 32:Assigning an INIT parameter module for the real and virtual axes in the axis mapping table

NC software version being used:

The NC software version being used can be determined via the library version of the **acp10man** library in the Logical View.

The NC Software version can be seen by **right-clicking** on the **acp10man library** in the **Logical View** and viewing the **Properties** on the **Details** tab.

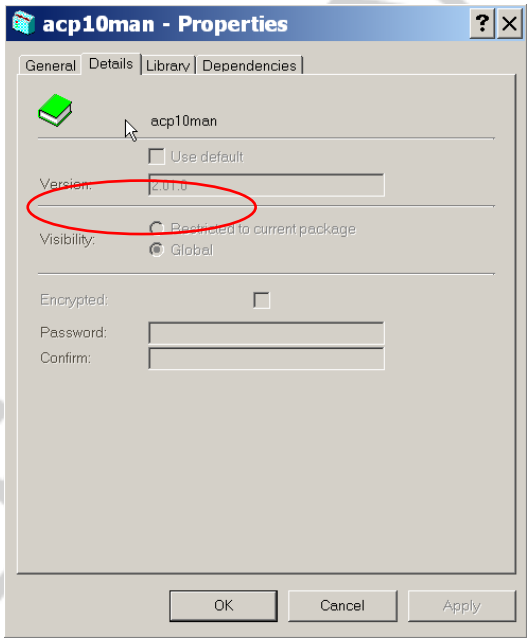


Fig. 33: NC software version being used

The NC software version can be changed by inserting a different version of the **acp10man library**.

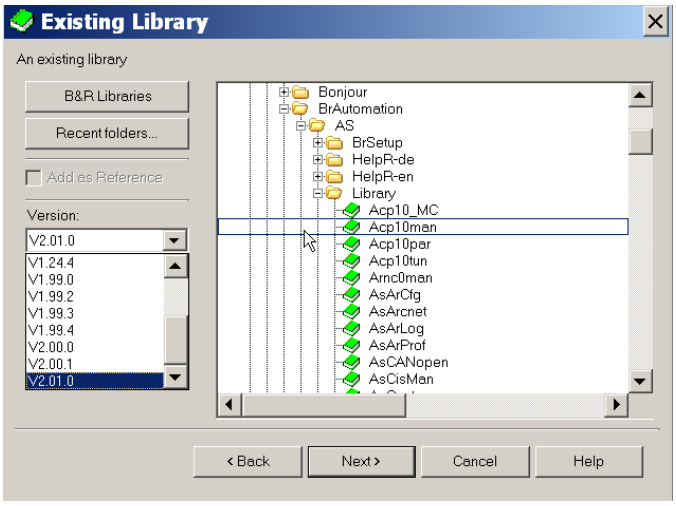


Fig. 34: Changing the NC software version

Automatic NC operating system version update:

- When using SG4 controllers, the operating system required for the ACOPOS devices is stored on the controller by default.
  When transferring the project, the ACOPOS operating system is transferred to and stored on the controller. During ACOPOS startup, the NC manager automatically transfers the ACOPOS operating system stored on the controller to all ACOPOS servo drives that contain a different version.
- In order to save memory space, this is not the default on SG3 controllers and therefore must be enabled if desired.
- Automatic version update for the NC operating system can be enabled by **right-clicking** on the **(SG3) PLC** in the **Configuration View** and going to **Properties** on the **Motion** tab.

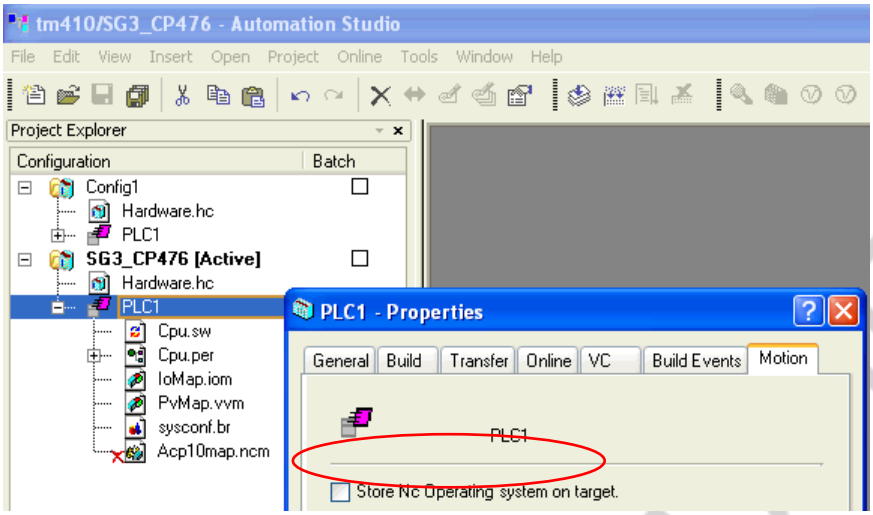


Fig. 35: Enabling automatic version update for the NC operating system

During the project transfer, the data segments of the ACOPOS operating system transferred to the controller are transferred to the

NC Manager on the controller. The NC manager then continuously passes these data segments on to all ACOPOS servo drives that contain a different operating system version.

Because the ACOPOS operating system is never stored on the controller and Automation Studio has no direct access to the ACOPOS servo drives for version-checking, it would also be transferred to the controller during the project transfer, if all ACOPOS servo drives already contained the correct version.

NC configuration:

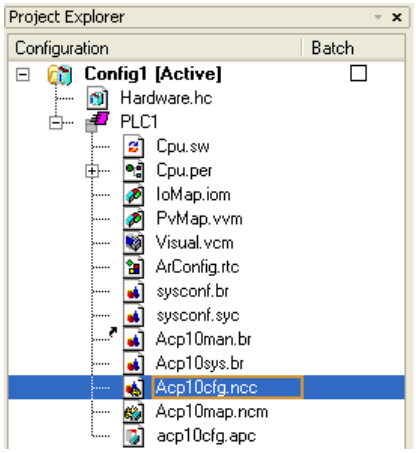


Fig. 36: Configuration view

The following table is opened in the right side of the window by **double-clicking** on the file **acp10cfg.ncc** in the **Configuration View**:

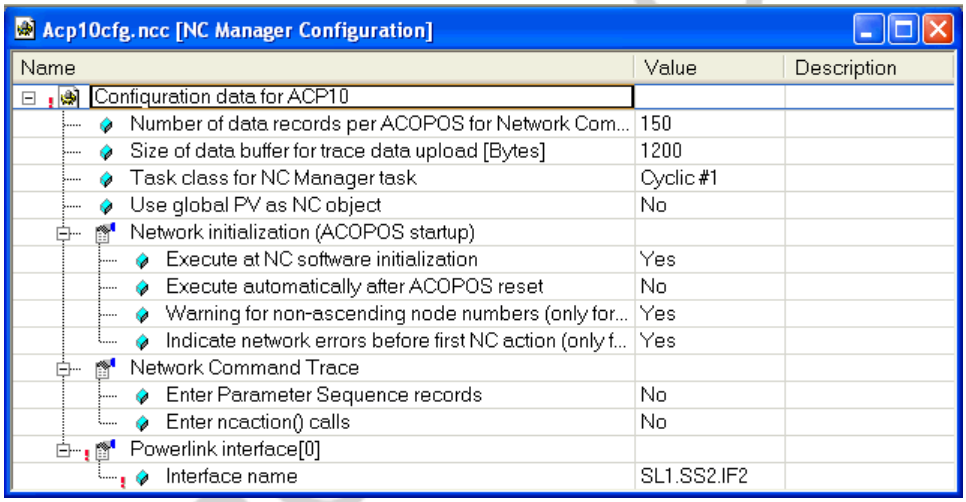
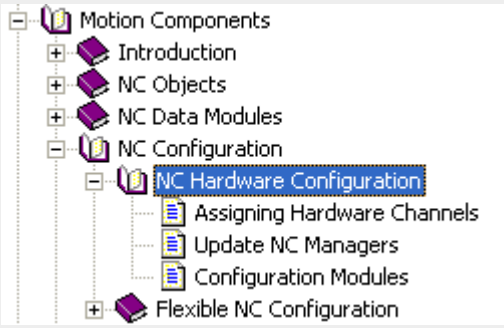


Fig. 37: NC Manager configuration

It is usually not necessary to change the default settings. Therefore, changes should only be made by advanced users!

Note:

Detailed information about the NC configuration settings can be found in the Automation Studio online help.



Task: Project download

Download the project with the new software components. If necessary, the system starts the transfer of the NC operating system to the ACOPOS.

3.4 Flexible NC configuration

The flexible NC configuration offers the possibility to define all of the NC software objects and properties required for an axis in a **NC mapping table**.

For example, **one INIT parameter module can be used for multiple axes**. This is why the rule of unique assignment between INIT parameter modules and axis objects was cancelled and the NC concept was expanded based on **INIT parameter modules without fixed hardware assignment**.

The following three changes must be made in the project to use this flexible type of management:

- **Redefine the NC objects and their hardware assignment in an NC mapping table**

- **Disable the ACOPOS module in the hardware tree**

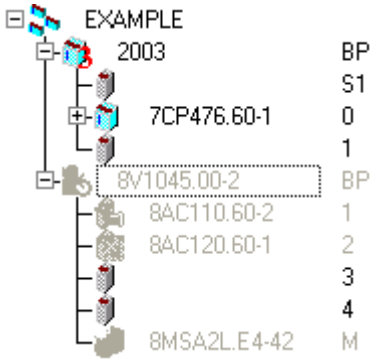


Fig. 38: Disable the ACOPOS module in the hardware tree

The ACOPOS module must then be disabled in the hardware tree to correctly set the definitions in the mappg table:

Note:

If the ACOPOS module is not deactivated in the hardware tree, a conflict occurs if the same NC objects (real and/or virtual axis) have also been entered in the deployment table ("double definition"). We recommend entering and managing all objects in the deployment table from now on.

The dialog boxes for the NC configuration are still offered via the module entry in the hardware tree (deactivated element).

3.4.1 Inserting an NC mapping table

A new object can be added by **right-clicking** on the PLC in the **Configuration View** and selecting **Add Object**.

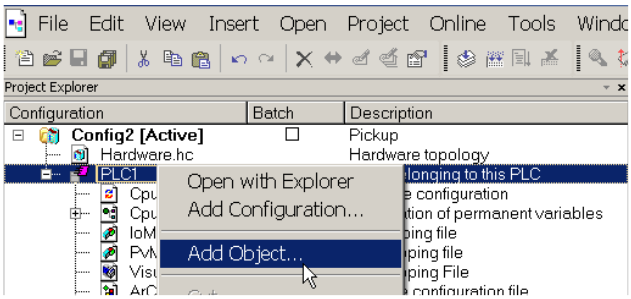


Fig. 39: Adding a new object – NC Mapping Table

Select **New NC Mapping Table** from the **Motion** category.

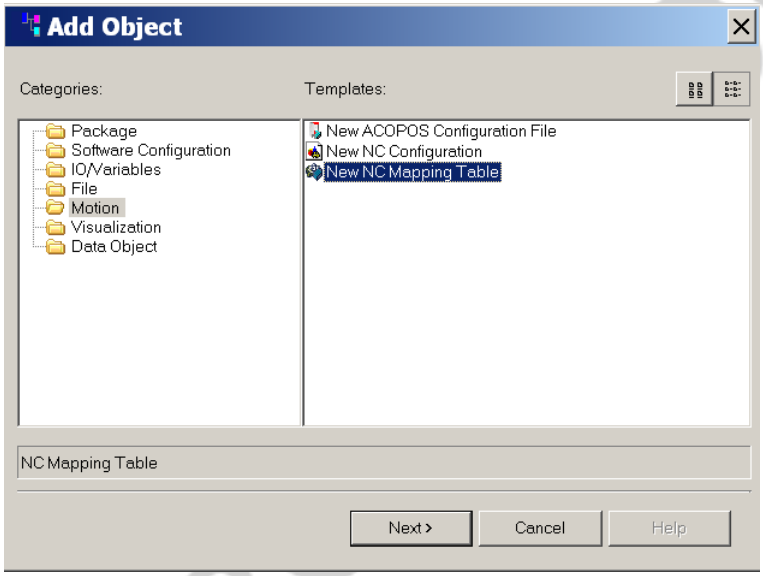


Fig. 40: Inserting a new mapping table

Clicking on **Next** will continue to the next dialog box.

A **name** for the NC Mapping Table is entered in the top field of this dialog box. Additionally, a **description** can also be added. We will select the **subtype ACP10: Mapping Table.**

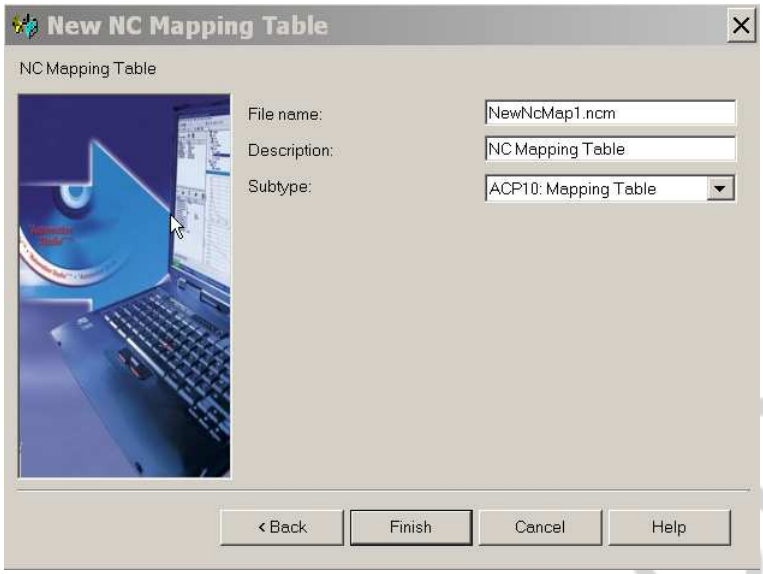


Fig. 41: New NC Mapping Table

After selecting **Finish**, the procedure for inserting the Mapping Table is complete.

The **NC objects and their hardware assignment can now be re-defined** in this table.

The Mapping Table can be be opened by **double-clicking** on the corresponding file name.

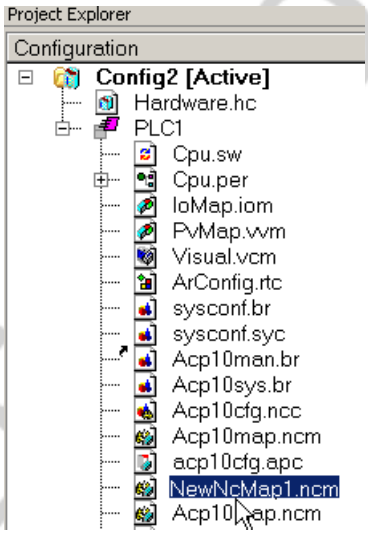


Fig. 42: Opening the mapping table

The following fields are provided to configure the **hardware-software assignment** for the individual NC objects (each in one line):

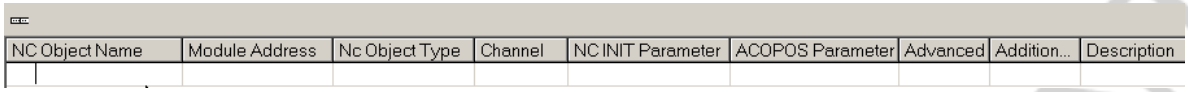
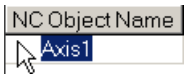


Fig. 43: New NC mapping table

What is defined in these fields?

- **"NC Object Name"**:

  A name that is unique within the entire project is defined in the "NC Object Name" with a maximum of 32 characters. This name will be required later for accessing the NC software object in an application program in the NC function **ncaccess()** (see the more detailed modules **TM440** and **TM441**).

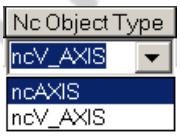
- **"Module Address"**

  A selection list is offered for "Model Address", in which all network interfaces are displayed. At least one of the NC hardware objects (e.g. ACOPOS) is connected to a network interface.

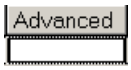
  The **x** at the end of the address string must be replaced by the node number of the corresponding ACOPOS.

- **"NC Object Type"**

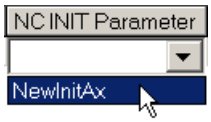
  The type of NC object is defined here. As discussed earlier, in addition to the operation of real axes (ncAXIS), every ACOPOS also offers the possibility of operating a virtual axis (ncV_AXIS). Both object types can be used simultaneously on the same ACOPOS module.

- **"Channel"**



For ACOPOSmulti: determines the axis on the servo drive in the case of double-axis modules.

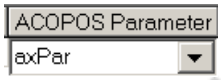
- **"NC INIT Parameter"** (→optional)



Definition of an INIT parameter module. The settings from this module are transferred to the corresponding parameter in the user data of the respective NC software object during the NC software initialization.

All INIT parameter modules in the project are offered for the specified NC object type. An INIT parameter module can be specified in NC mapping tables for as many NC software objects as needed under the column "NC INIT Parameter".

- **"ACOPOS Parameter"** (→optional)



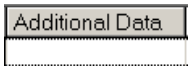
"ACOPOS Parameter" is used to define an ACOPOS parameter table whose parameters should be transferred to the corresponding ACOPOS unit during the NC software initialization (more information about the ACOPOS Parameter Table is available in section "3.5. ACOPOS Parameter Table").

All ACOPOS parameter tables in the project are offered for selection. An ACOPOS parameter table can be specified in NC mapping tables for as many NC software objects as needed under the column "ACOPOS Parameter".

- **"Advanced"**

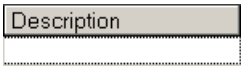
All advanced network settings can be made here. When using the ETHERNET POWERLINK network, the parameters e.g. for the Powerlink property "Multiplexed", can be defined here.

- **"Additional Data"** (→optional)



Additional data for an NC software object can be directly entered in XML syntax in the "Additional Data" column (e.g. for advanced functions).

- **"Description"** (→optional)



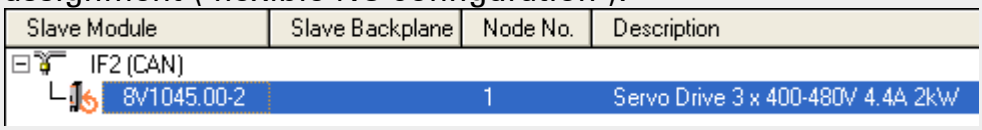
This area provides a place for notes and comments.

These entries are used to assign the hardware components (ACOPOS modules in the network) to corresponding software components (NC object types, INIT parameter modules, etc.).

Note

A selection list is offered for the "Network Interface" field in the mapping table where all network interfaces with **at least one** connected ACOPOS are offered.

It is necessary to insert an ACOPOS for the corresponding interface as shown in section "3.1. Inserting an Axis" before switching to the flexible assignment ("flexible NC configuration").



This type of management offers the following advantages:

The NC objects are configured and managed in a clear and structured manner in a table and can be easily changed when necessary. This brings the following advantages for the user:

- Flexibility
- More clarity and structure, especially for larger projects.
- Multiple configuration modules (mapping tables) allowed
- Flexible assignment of the INIT parameter modules and ACOPOS parameter tables for all NC objects
- Use of symbolic names ("NC Object Name") for simplified access when working in the application program.

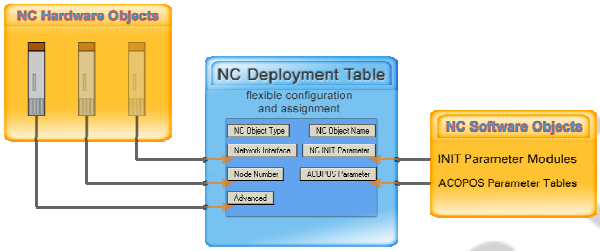


Fig. 44: Flexible NC configuration

Note:

In the more detailed modules **(TM440** and **TM441)**, we will get to see the extraordinary advantages of assigning symbolic names for accessing an NC object. In the application task, the NC object can be accessed using this name. Changes to the configuration (interface, node number, etc.) do not have to be taken into consideration in the program.

Note:

Additional information about the characteristics and settings for flexible NC configuration can be found in the Automation Studio online help.

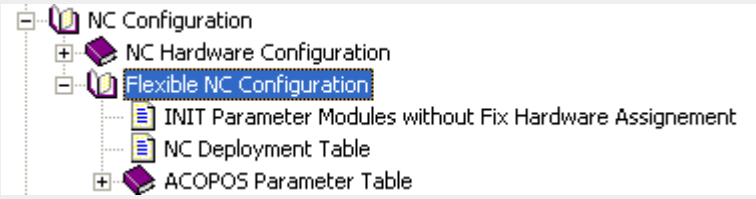


Fig. 45: Automation Studio Online Help

3.5 ACOPOS parameter table

Parameters on the ACOPOS can be addressed directly. In the earlier sections we saw that the ACOPOS servo drive has a large number of parameters. Parameters are provided on the ACOPOS to define configuration-specific data (motor characteristics, encoder, etc.), as well as to define commands (positioning movements, etc.).

Note:

The NC software provides a clear, object-oriented user interface, as we have already seen in the example for the INIT parameter module (main objects, subgroups, etc.). The NC Manager provides "help" defining the configuration and sending commands. Use of the standard PLCopen Motion Control function blocks is also supported.

In an **ACOPOS parameter table**, a **group of ACOPOS parameters can be put together** and transferred to the ACOPOS.

When doing this, the parameters are uniquely identified via a parameter ID e.g.:

Constant-name	Value / ID	Description
ACP10PAR_ICTRL_KV	223	Current controller: Proportional gain
ACP10PAR_ENCOD_SSI_BITS	238	Encoder1: SSI number of data bits.
....		

3.5.1 Creating an ACOPOS parameter table

A new object can be added by **right-clicking** on the folder with the project name in the **Logical View** and selecting **Add Object**.

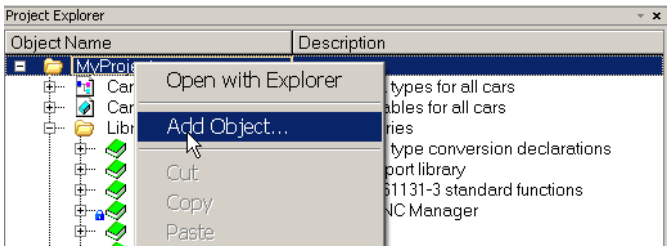


Fig. 46: Inserting an NC software object

The corresponding category **Motion** is selected under **Categories**. A new ACOPOS parameter table can be then be selected from the right side of the dialog box by choosing **New ACOPOS Parameter Table**.

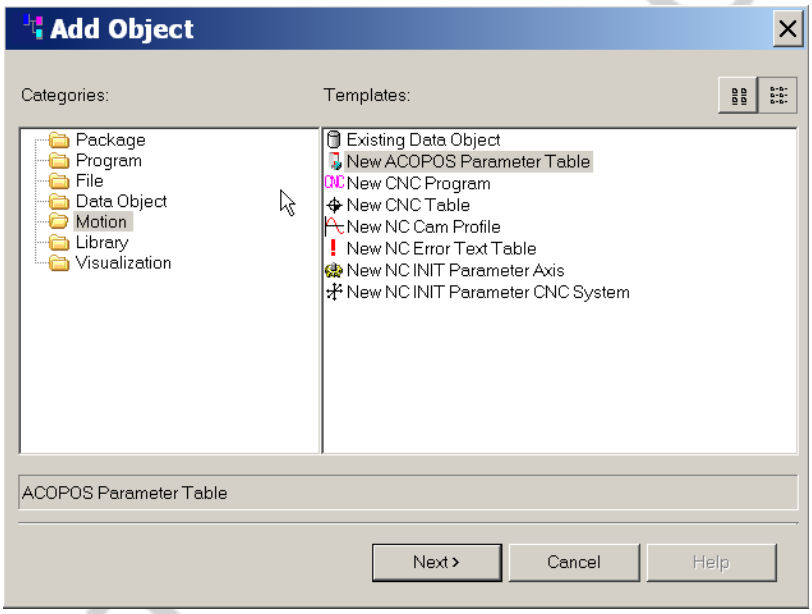


Fig. 47: Inserting an ACOPOS parameter table

A new dialog box is opened after confirming your selection with **Next**.

This dialog box is used to enter a **name** and a **description** for the new ACOPOS parameter table.

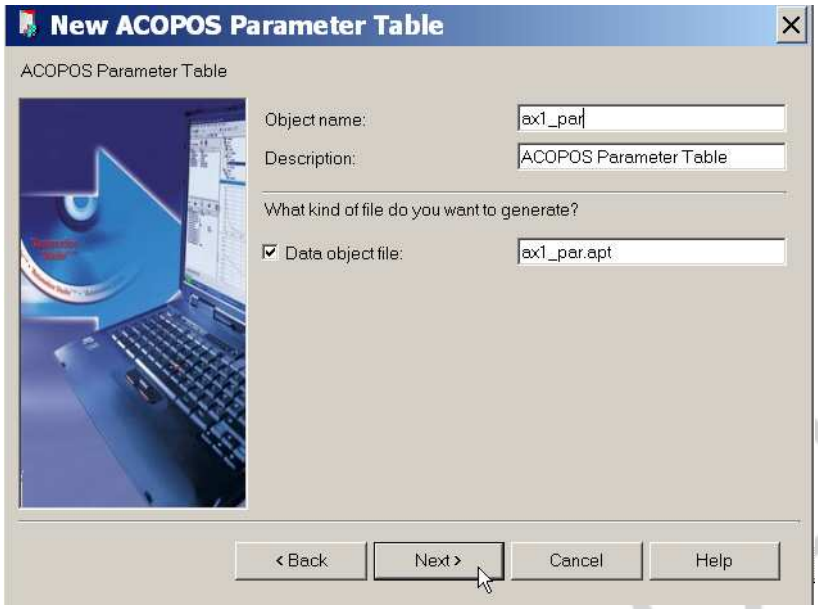


Fig. 48: Name and description of an ACOPOS parameter table

We will assign the new ACOPOS parameter table **to the active CPU**.

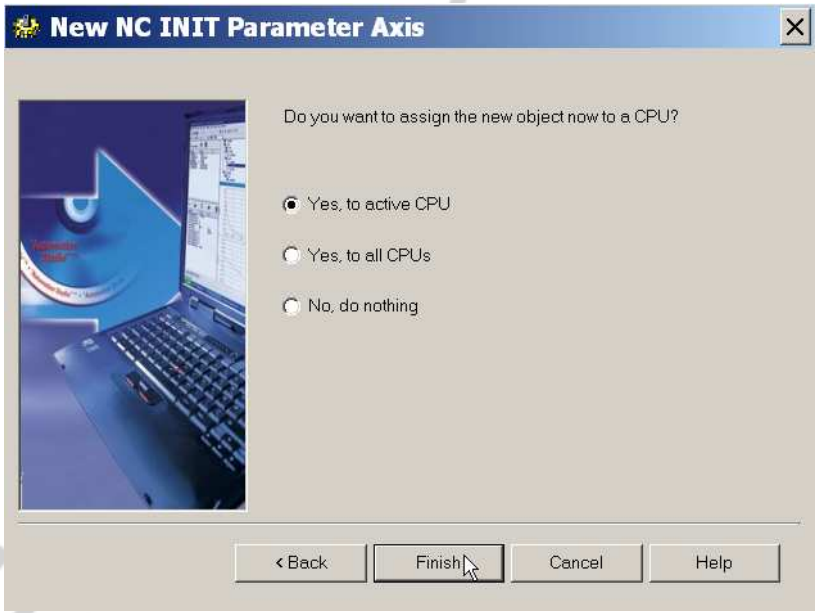


Fig. 49: Assigning an inserted INIT parameter module to the active CPU

After the ACOPOS parameter table has been added successfully, it can be opened and edited by double-clicking on the corresponding name in the tree structure of the the **Logical view**.

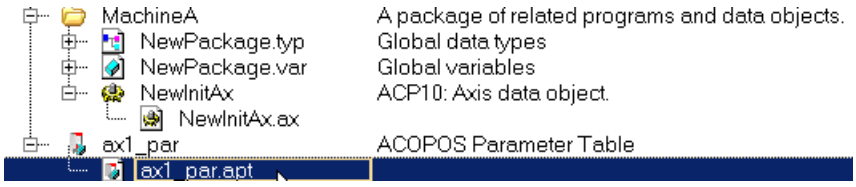


Fig. 50: Opening the ACOPOS parameter table

3.5.2 Inserting individual parameters

Individual parameters and parameter groups can be added to the blank table by **right-clicking**:

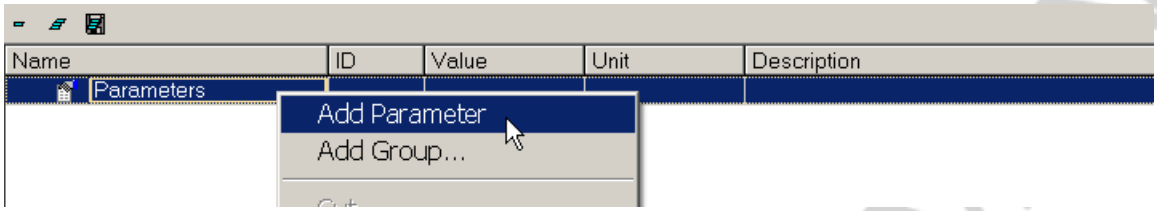


Fig. 51: Appending parameters

A new parameter is added to the list by selecting **Add Parameter**.

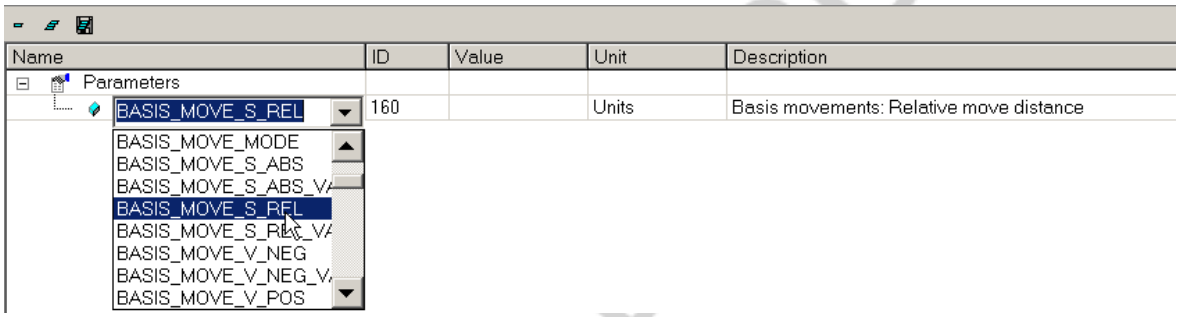


Fig. 52: Selecting a parameter

All ACOPOS parameters are displayed for selection in a **pull-down menu** under the **Name** column. However, the parameter can also be entered using the parameter ID (if known by the user). The parameter is automatically detected after entering the respective **Par ID number** in the **ID** field. The value for the selected parameter is assigned in the **Value** field.

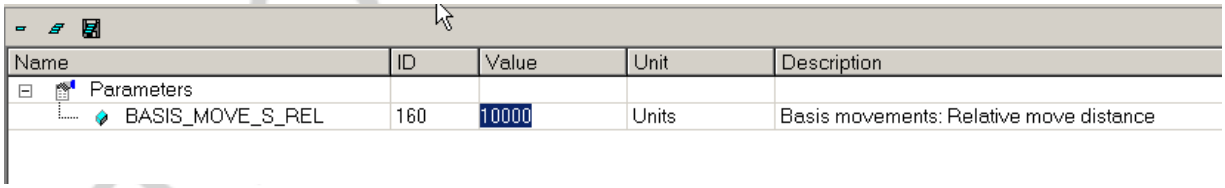


Fig. 53: Entering parameter values

Note:

A listing of all ACOPOS parameters with information about there purpose and application can be found in the Automation Studio online help.

3.5.3 Inserting a parameter group

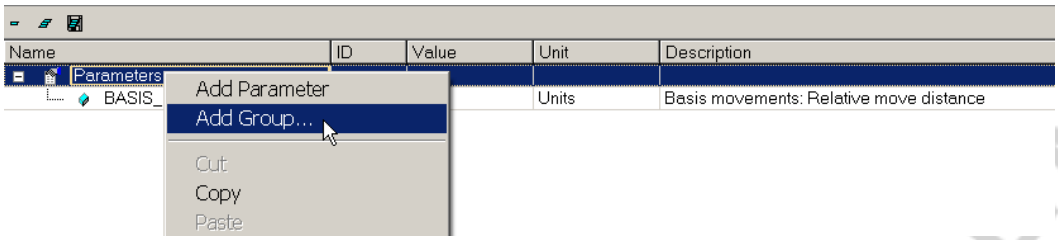


Fig. 54: Appending a parameter group

The following selection dialog box is opened via the **Shortcut menu** with the item **Add Group**:

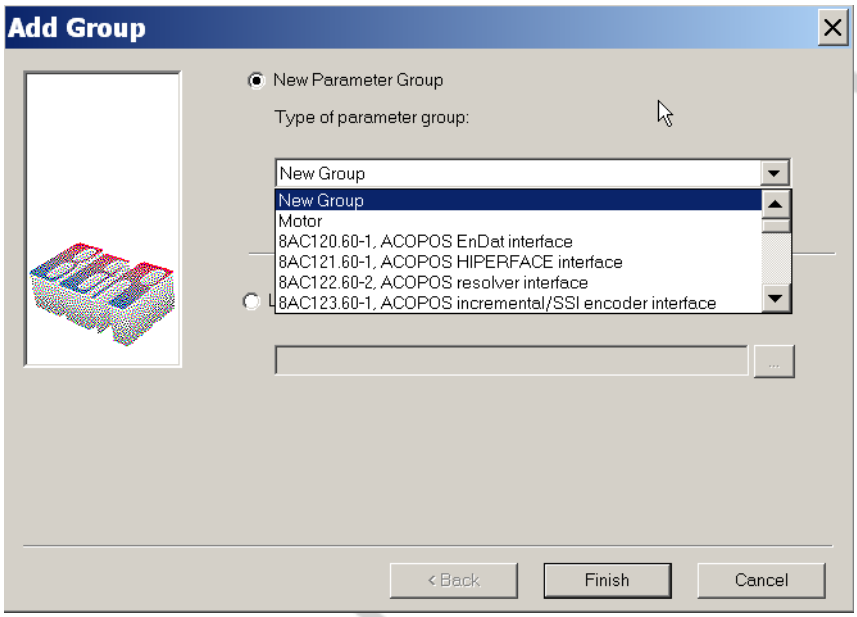
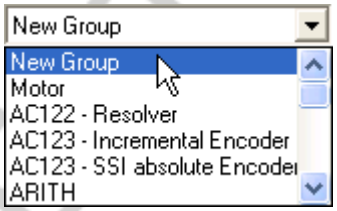


Fig. 55: Selecting the group

Here it is also possible to add a parameter group from an existing file.

Different groups are provided for selection in a **pull-down menu** by choosing the option **New Parameter Group**".



A new, empty sub group is added to the parameter table by selecting **New Group** (similar to a folder). Individual parameters and parameter groups can be added to this "folder".

Note:

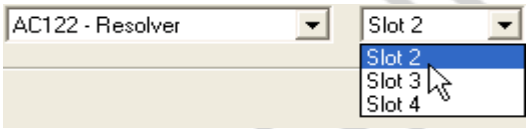
This possibility can be put to good use for structuring large parameter lists.

Furthermore, predefined parameter groups are also contained in this selection. This includes:

- Grouping of **motor parameters** (characteristics) for definition for the ACOPOS
- Grouping of **plug-in cards** (e.g. encoder cards) for definition for the ACOPOS
- Grouping for **SPT functions (S**mart **P**rocess **T**echnology functions, see **TM447)**

When selecting the "Motor" group, a wizard is opened for choosing the motor being used (see section **"3.1. Inserting an axis").** The motor is now defined as usual. After completing this procedure, all parameters required for defining the motor are entered in the parameter list with the corresponding values (characteristics).

The slot number on the ACOPOS must be defined **when selecting a group for a plug-in card**:



After completing this procedure, the parameters required for defining the plug-in motor are entered in the parameter list.

This makes it possible to create any kind of parameter grouping needed for the ACOPOS.

3.5.4 Downloading an ACOPOS parameter table

In principle, there are two ways to **download a parameter table** and **initialize the parameter values** (order starting from top to bottom) on the ACOPOS:

- by downloading the parameter table during runtime from the application task (more about this in the training module **TM440**)
- by assigning the parameter table in the NC mapping table. The parameters from the table are transferred to and initialized on the ACOPOS during system startup.

Note:

Initializing parameters means that the corresponding parameter values become active on the ACOPOS drive. Initializing parameters and transferring data to the ACOPOS do not always happen together. There are other methods of transferring data where it is possible to initialize the values on the ACOPOS at a later time.

3.5.5 Hardware configuration with an ACOPOS parameter table

Parameters for ACOPOS hardware (e.g. motors and ACOPOS plug-in cards) are defined in the NC hardware configuration while managing the ACOPOS servo drives in the hardware tree. These parameters are then assigned directly to a module in the hardware tree.

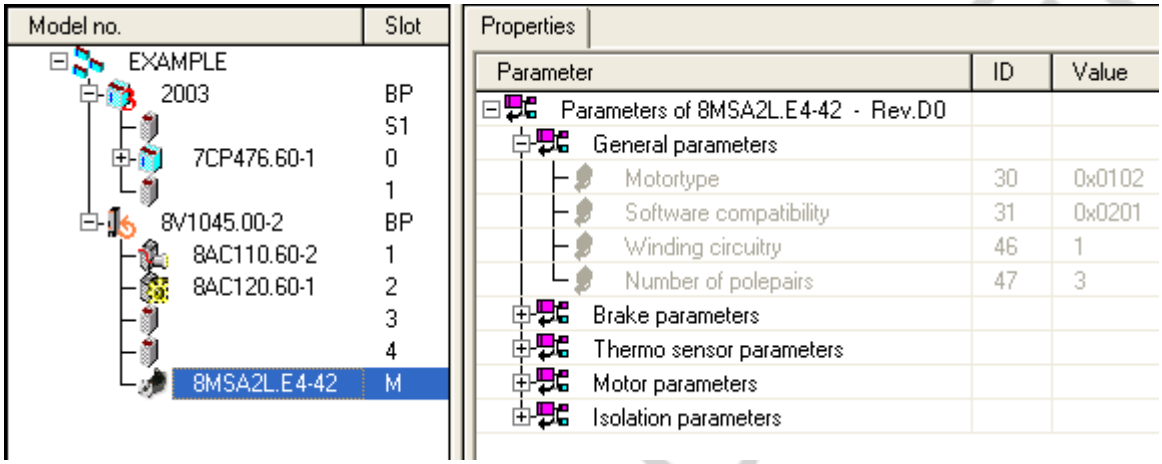


Fig. 56: Hardware parameter

This type of assignment is no longer valid when managing the drive objects with the **flexible NC configuration** (NC mapping table).
In this case, the hardware configuration must be made using an ACOPOS parameter table.

Therefore, predefined parameter groups (e.g. **"motor"** or **"AC122 – Resolver"**, see above) are provided which can be added to an ACOPOS parameter table for ACOPOS hardware.

Note:

When using an **EnDat encoder system**, the hardware configuration remains **unnecessary**. The EnDat encoder has an embedded parameter chip and automatically provides the ACOPOS with the parameters for the hardware configuration (motor, encoder).

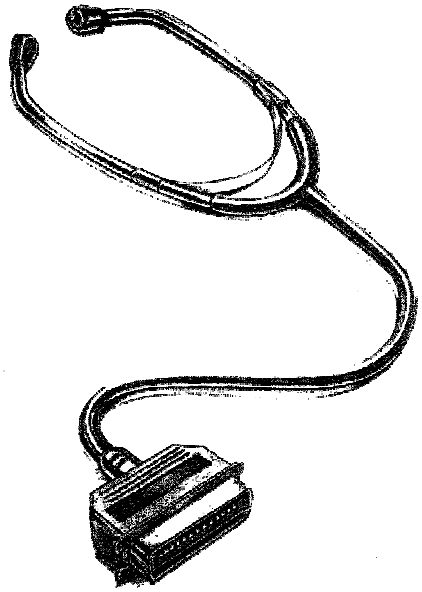
Task: Flexible NC configuration

Switch handling of the drive objects in your project to flexible NC configuration. Follow the steps listed earlier:

- Motion project settings: The "Hardware definition for the NC init parameter objects" must be disabled.
- Create an NC deployment table, configure the drive objects (NC objects)
- Disable the ACOPOS module in the hardware tree
- If an **EnDat encoder system is not used**: Create an ACOPOS parameter table and add the definition parameters for the motor and encoder (groups) in this table. Set the assignment to the corresponding NC object in the NC deployment table

4. NC TEST

Diagnostics and analysis tools are fully integrated in Automation Studio. These tools can meet all demands and allow the drive axes to be started quickly and efficiently.



Possibilities:

- Sending commands
- Access to the drive object's parameters
- Displaying status parameters
- Recording real-time data

Operation of the individual components in the NC Test is clearly structured and intuitive.

Note:

The **requirements** for NC Test are the same as for all other online tools. That means Automation Studio must be connected to the controller and all necessary software objects must be correctly transferred to the controller.

4.1 Opening the NC Test

The mapping table is opened by **double-clicking** on the ACOPOS in the hardware tree.
The NC Test can be opened by **right-clicking** on the corresponding axis object in the mapping table and selecting **Test**.

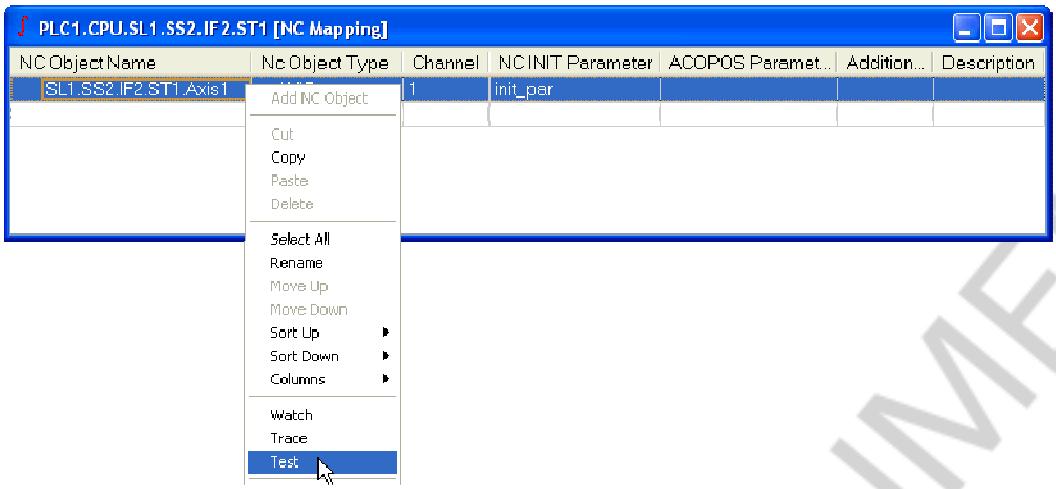


Fig. 57: Opening the NC Test

Note:

By sending commands in the NC Test, the axis can now receive commands from both the application and the NC Test.

The following dialog box appears when opening the Test window:

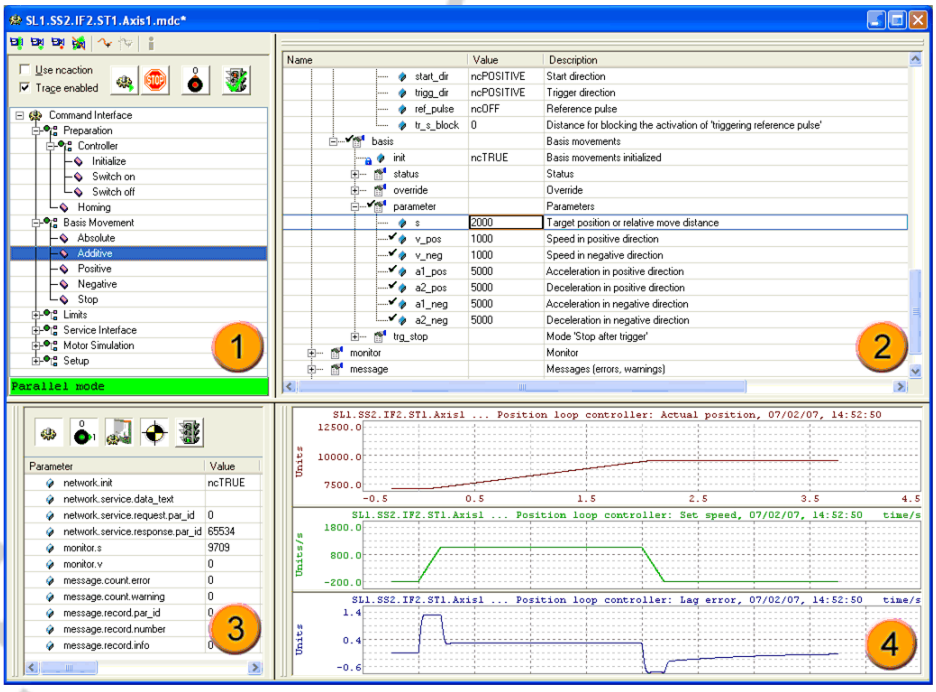


Fig. 58: NC Test mode

The user can choose whether the NC Test should operate the axis **parallel** to the application or **exclusively**. The application commands are not forwarded to the axis in exclusive mode.

4.2 Test window structure

The NC Test is divided into four main fields:

- Command interface (1)
- Parameters for commands (2)
- NC watch (3)
- NC trace (4)

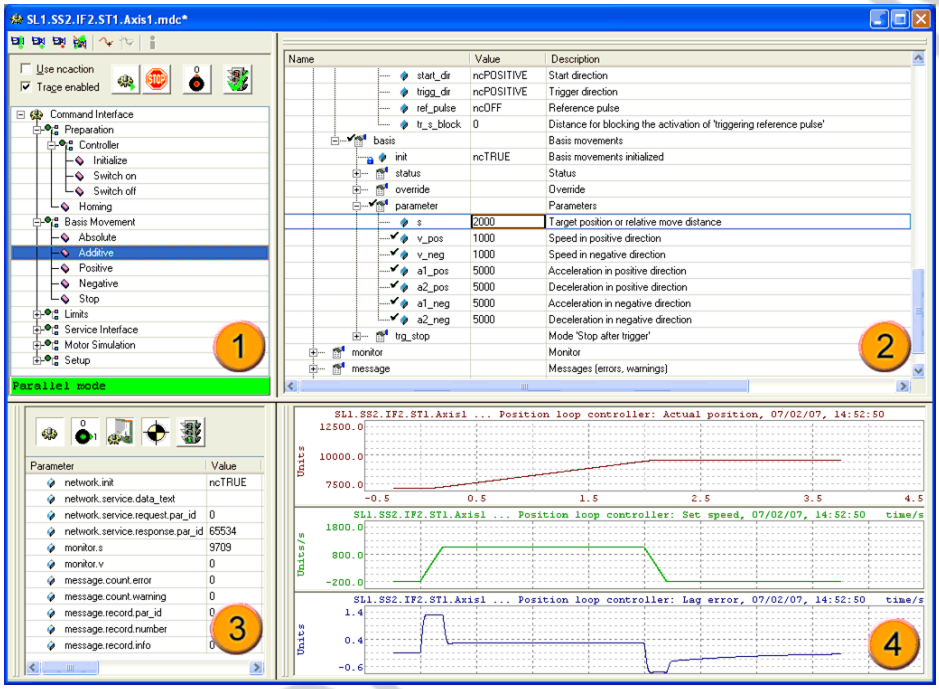


Fig. 59: NC Test window

When closing the NC Test during an active movement, the user can choose whether the movement should also be stopped:

Fig. 60: Movement stop when closing NC Test

4.3 Command interface

The upper left part of the window in NC Test is used for sending commands to the axis (movements, homing, etc.).

There are two different views available:

- **Command view with the structure:**

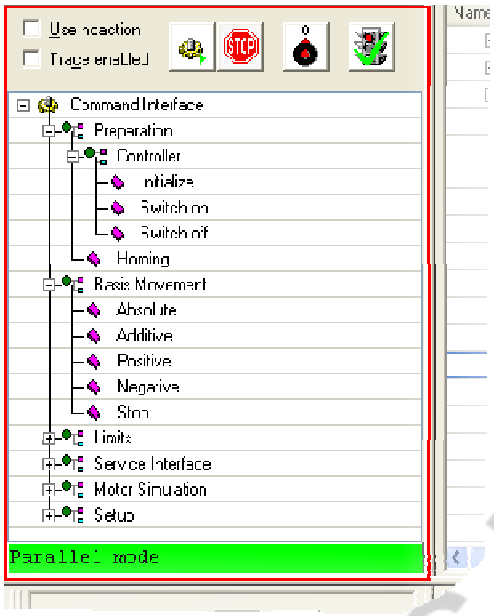
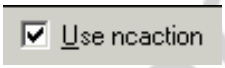


Fig. 61: Commands in the structure view

- **Command view with the NC actions:**
  Select the checkbox **"Use ncaction"** to keep this view.



Commands can be sent in this view via NC actions (ACP10 software user interface).

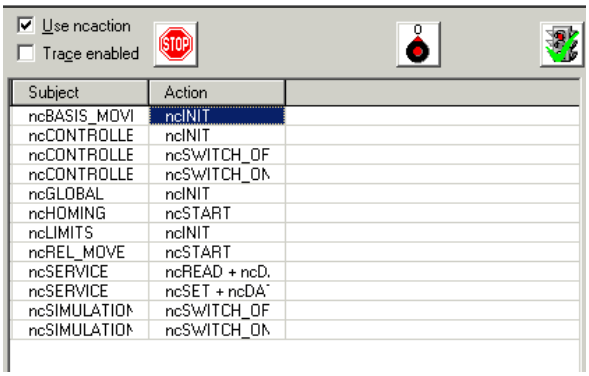


Fig. 62: Commands via the NC actions view

Additional NC actions can be inserted to this list by **right-clicking** and selecting **Insert** (or by using the **"Ins" key** or via the **Menu)**.

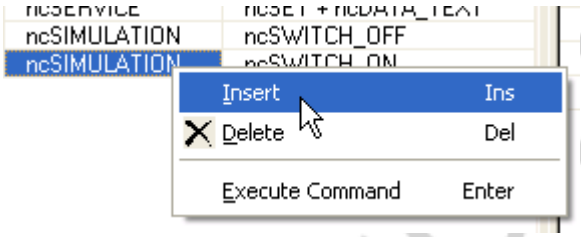


Fig. 63: Inserting additional NC actions

The actions, which are sorted in the respective subgroups, can be selected in the subsequent dialog box for inserting.

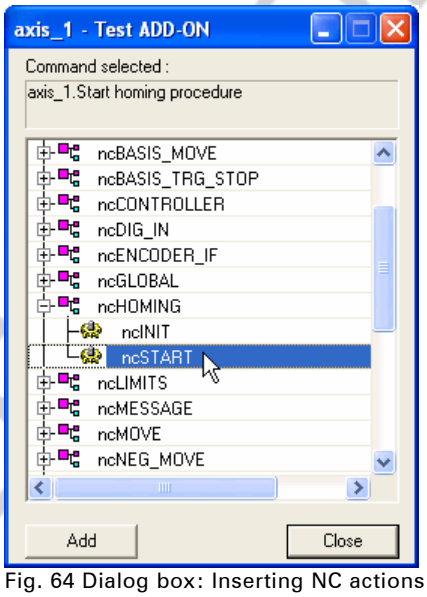


Fig. 64 Dialog box: Inserting NC actions

How are commands made in the command interface?

- Enter
- Double-click
- Button 

There are a few additional buttons for the most important commands so that they can be executed quickly:

Button	Description
 or F3	Stop movement
	The controller on the drive will be switched off
	Acknowledging axis errors

Note:

If the **NC Actions view** is used in the command interface, then the command **ncGLOBAL→ ncINIT** for initializing all parameters must always be made by the user. In the **Structure view**, the parameters are initialized automatically before executing the first command.

4.4 Parameters for commands

The respective parameters from the operating structure, which belong to the specific commands, are provided here.

When a command is selected, the corresponding parameter group is automatically displayed on the right side of the window. These parameters can be adjusted as needed. When executing a command, the corresponding parameters of the respective command are automatically applied.

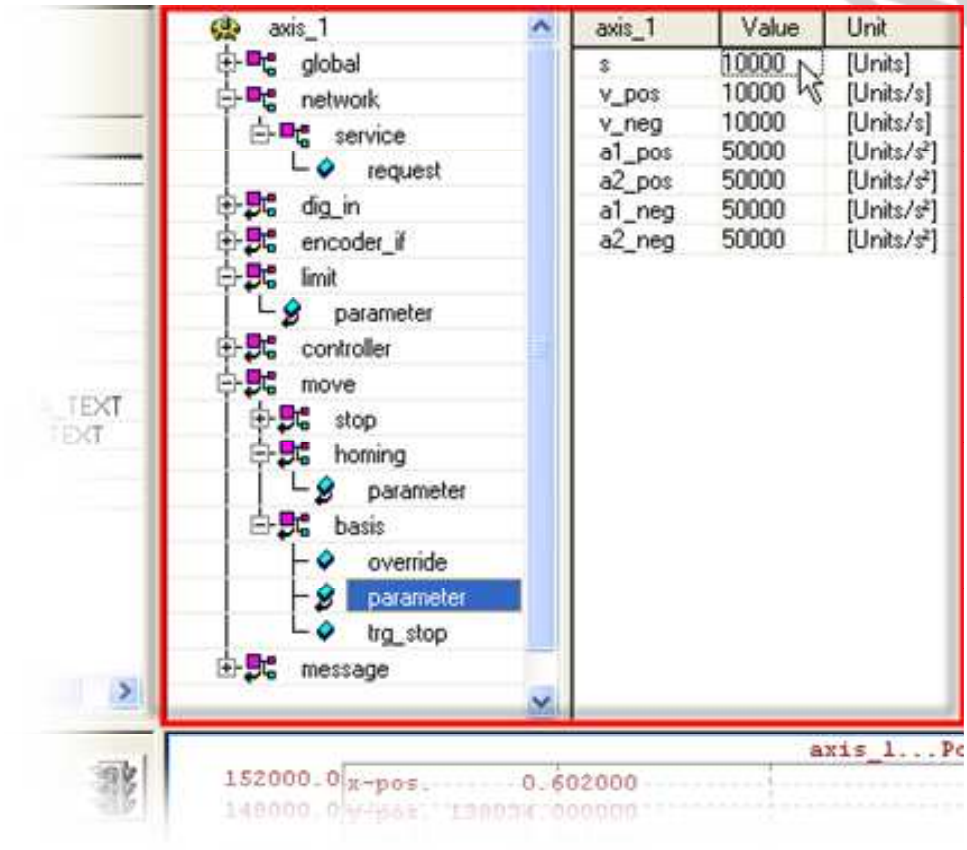


Fig. 65: Object parameter window

Note:

The parameters changed during the test are valid only until the test is finished. After the test is complete, the parameters from the application and from the init parameter module are used on the controller again.

Note:

This structure also contains the parameters from the INIT parameter module. If values are entered here that are different than the ones from the INIT parameter module (also when changed by an application task), then this is indicated by a black checkmark at the far left in this area:

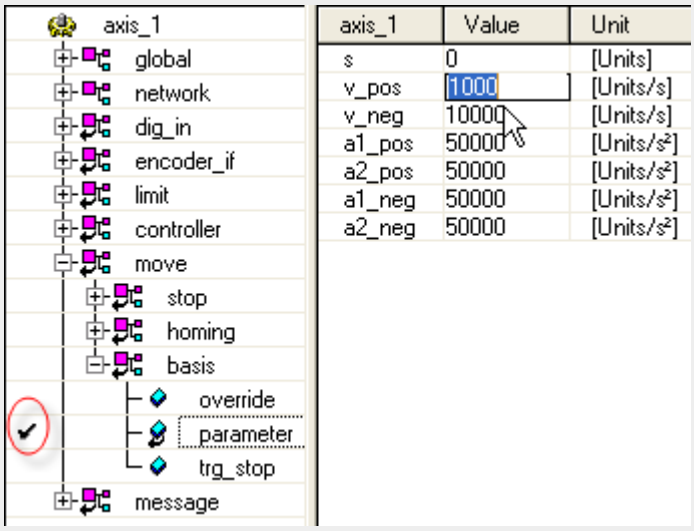
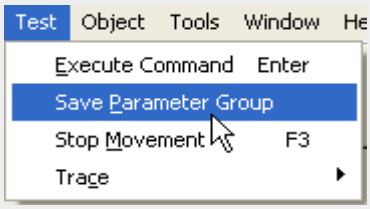
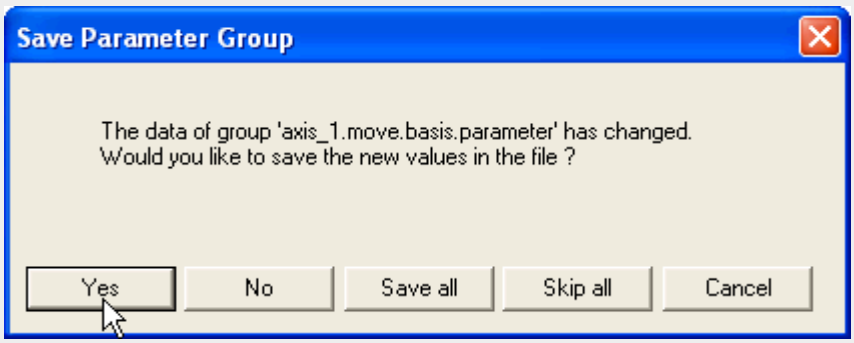


Fig. 66: Changes to initialization parameters

If initial parameters have been changed, then they can be transferred to the INIT parameter module (individually for each group of parameters) using the menu command **Test** / **Save Parameter Group**.



Otherwise a query appears when closing the NC test, asking to accept or reject the changes to the initial parameters in the INIT parameter module together or individually.



4.5 Status values in NC Watch

The NC Watch shows the user the current status of an NC object.

The NC Watch window can be divided into three parts:

- Object states (icons above)
- Object data (below)
- Error text (below)

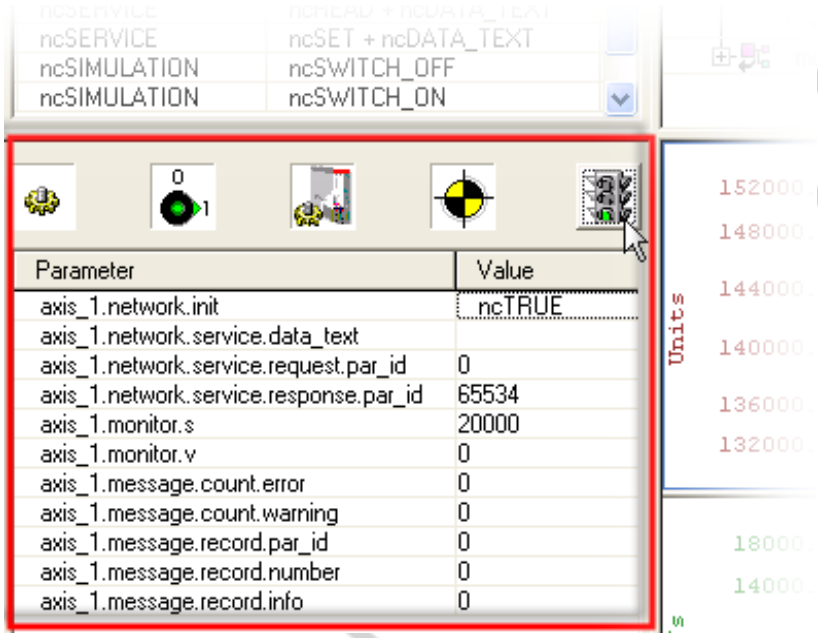


Fig. 67: NC Watch window

Note:

This tool can also be opened outside of the NC Test (same procedure as opening the NC Test, **Selection: Watch**).

4.5.1 Object states

Icons are used in the bar at the top to display the most important NC object states.

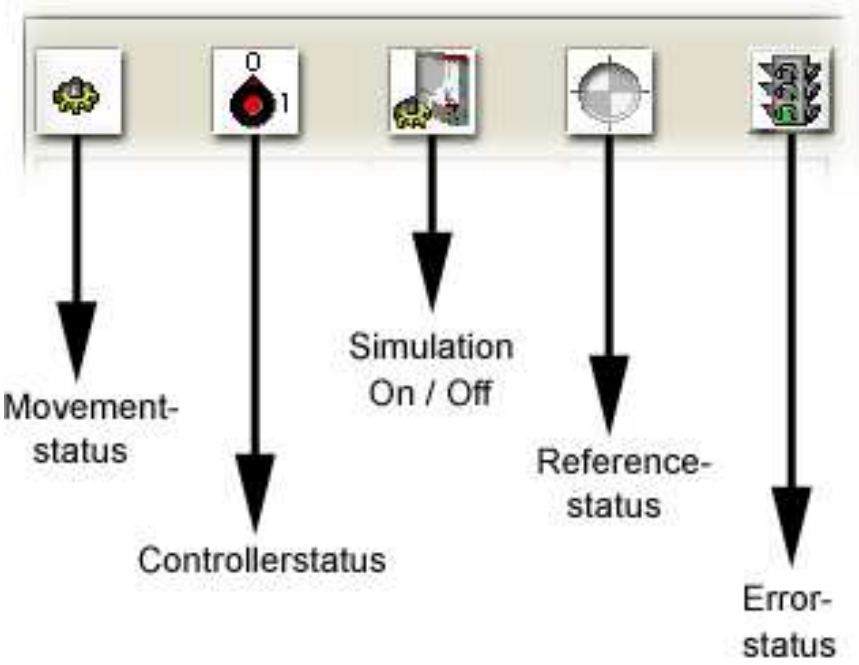


Fig. 68: Watch: Status icons

This gives the user a quick and useful overview of the current NC object status.

Note:

A connection error for the NC object on the controller is illustrated using the following pictogram:

In this case, check the physical connection and the relevant settings.

4.5.2 Object data

The first time the NC watch is started, this section displays a few elements from the operating structure that are used frequently. The user has the option to adapt the selection of the displayed parameter.

Additional elements can be added to or removed from the list using a dialog box as usual.

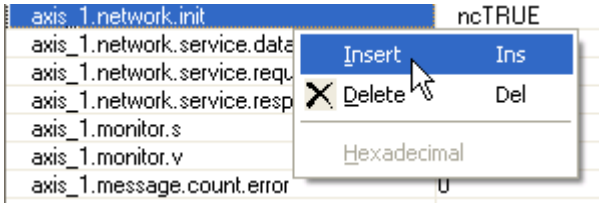


Fig. 69: Inserting status values

The desired elements (to the right) are selected in the corresponding dialog box from the NC object operating structure subgroups:

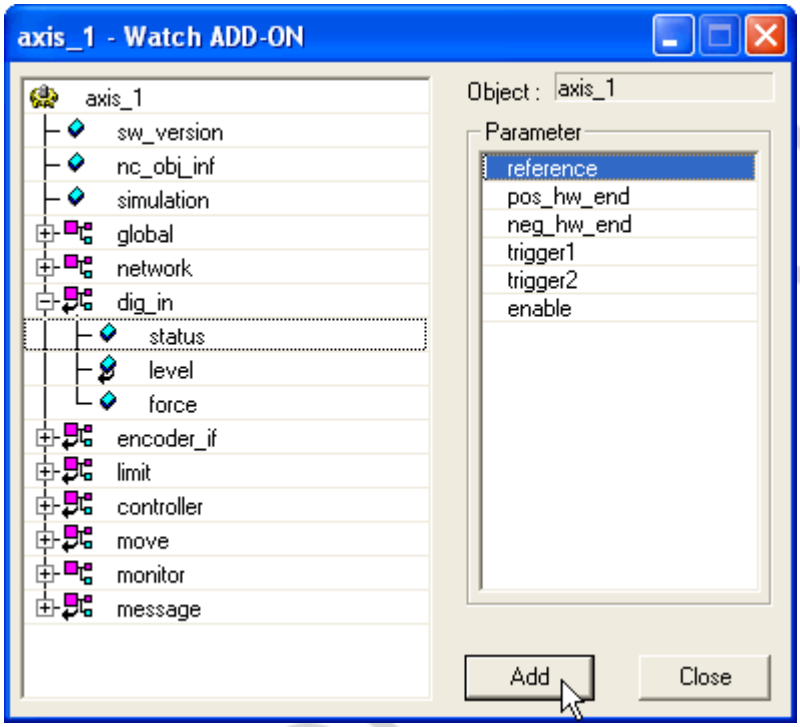


Fig. 70: Dialog box: Inserting watch parameters

The selected entries are then added to the NC Watch. The values can also be displayed in hexadecimal form.

Fig. 71: Settings for the desired number system

The list of the parameters to be displayed from the parameter window is saved (without the parameter values) when closing the NC Watch window. This list is restored the next time the NC Watch is opened.

4.5.3 Error text

The traffic light button on the far right can be used to switch between status data and error text for display in the lower part of the NC Watch window.

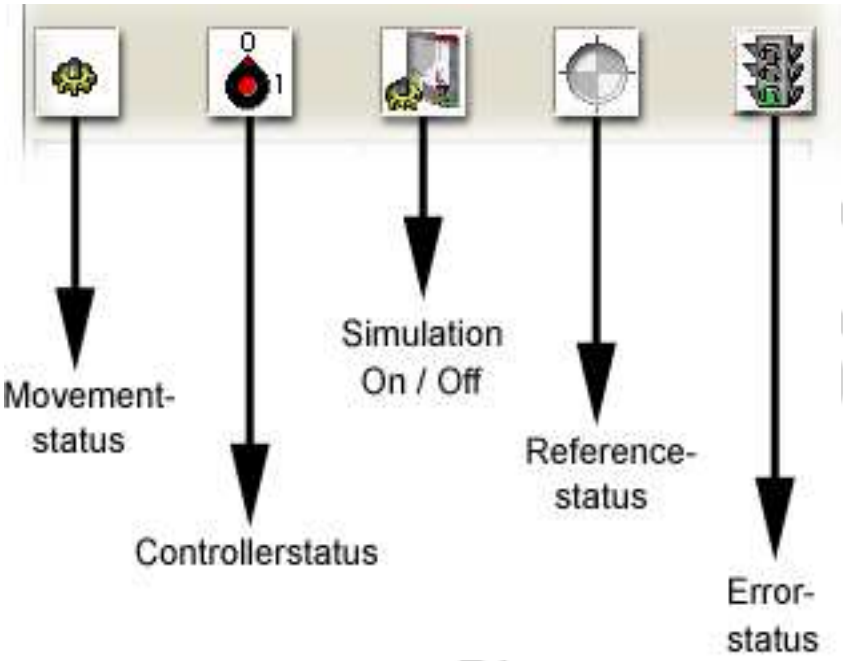


Fig. 72: Watch: Status icons

No error on axis:  Error on axis: 

If the NC object is in an error state, the corresponding error description for the current error is displayed here:

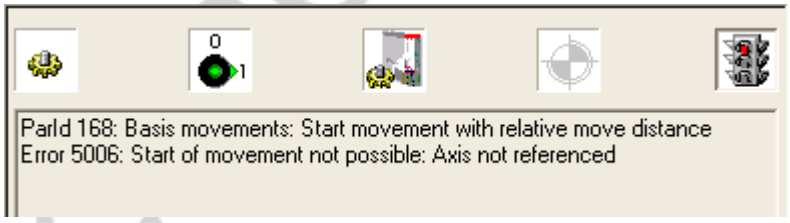


Fig. 73: NC Watch: Error text display

The present errors can be acknowledged (starting with the first occurring error) using the  button.

Multiple errors caused simultaneously by an action can be acknowledged in order.

Task: Execute basis movements in NC Test

Open the NC test for your drive. Use the object parameter to set the logical level of all digital inputs to "ACTIVE HIGH". This ensures that a positioning movement will not be blocked by inputs that are not connected (e.g. hardware limit switch active). To accept the settings, carry out the action **"ncDIG_E→ncINIT"**.

Procedure (with the use of the command structure):

- Check if the **green LED** on the ACOPOS is on ("ready")
  Switch on controller: "Preparation – Controller – Switch On"
  (Detailed information about setting the drive controller can be found in the training module **TM450 ACOPOS Control Concept and Controller Settings**)
- Execute **Homing procedure**: "Preparation - Homing"

  The drive is now ready for movement actions.
- **Move with relative traverse distance**: "Basis Movement - Absolute"
- **Move with absolute target position**: "Basis Movement - Additive"
- **Positive movement at constant speed**: "Basis Movement - Positive"
- **Negative movement at constant speed**: "Basis Movement - Negative"

The corresponding **basis movement parameters** are offered automatically and can be adjusted as needed. Parameters are automatically initialized and applied before the movement is started.

Procedure (with the use of NC actions):

- Carry out **global initialization** "ncGLOBAL, ncINIT". In this way, the NC Manager transfers all parameters required for basic operation from the operating structure to the ACOPOS and they take effect.
- Check if the **green LED** on the ACOPOS is on ("ready")
- **Switch on controller**: "ncCONTROLLER, ncSWITCH_ON"
- Execute **Homing procedure**: "ncREFERENCE, ncSTART"

  The drive is now ready for movement actions.
- **Move with relative traverse distance**: "ncREL_MOVE, ncSTART"
- **Move with absolute target position**: "ncABS_MOVE, ncSTART"
- **Positive movement at constant speed**: "ncPOS_MOVE, ncSTART"
- **Negative movement at constant speed: "ncNEG_MOVE, ncSTART"**

Note:

A connection terminal for **digital inputs** is located on the front of the ACOPOS module.

Fig. 76 ACOPOS

- Reference switch
- Positive hardware limit switch
- Negative hardware limit switch
- Trigger input 1
- Trigger input 2

The "Active level" (**"ncACTIVE_HI"** / **"ncACTIVE_LO"**) and the additional function **"+ ncQUICKSTOP"** can be configured for each input.
If an input which was configured with **"+ncQUICKSTOP"** has been activated, the drive is slowed down to standstill, with the preset movement parameters.

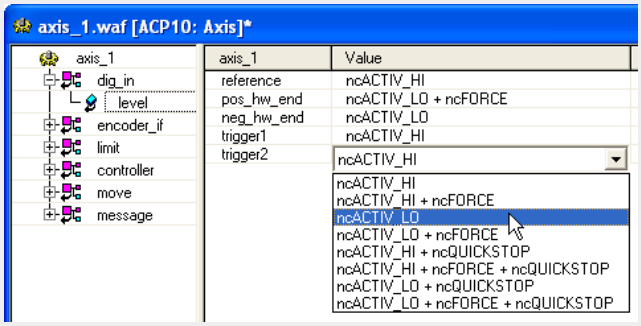


Fig. 76: Forcing digital inputs

By also using **"+ncFORCE"**, the respective input can be set ("forced") to a certain value, independent of the actual physical input value, using the command ("ncDIG_E, ncFORCE").

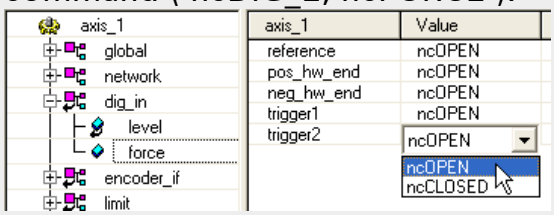


Fig. 76: Force level setting

The initialization (= transfer of settings from the operating structure to the parameters on the ACOPOS and thereby their functionality) is performed using the NC action **"ncDIG_IN, ncINIT"**.

4.6 NC Trace

The Trace functions provide very useful tools for testing the behavior of the ACOPOS and control program.

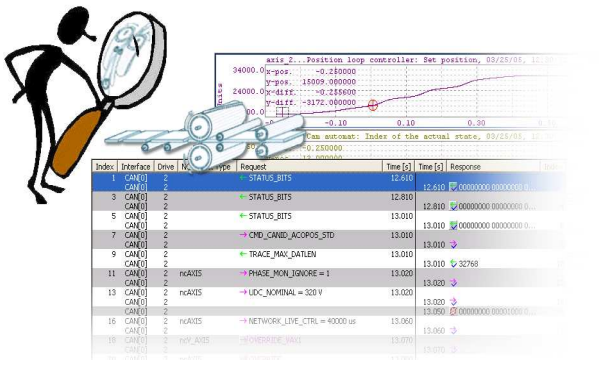


Fig. 77: NC Trace as analysis tool

4.6.1 Cyclic trace

The trace function allows **real-time data to be recorded** directly on the ACOPOS device. This data is loaded to Automation Studio from the ACOPOS device via the controller where it is then evaluated.

> **Note:**
>
> "Cyclic" means that a certain data set is registered in a defined sampling interval (cycle).

The Trace offers the following functions:

- Full-text search for parameters
- Intuitive and easy configuration
- Automatic calculation of the sampling time
- Automatic calculation of the maximum Trace duration
- Record buffer capacity (fill level indicator)
- Multi-axis trace (synchronous recording of multiple axes)

The following image displays a recording of measurement results in NC Trace:

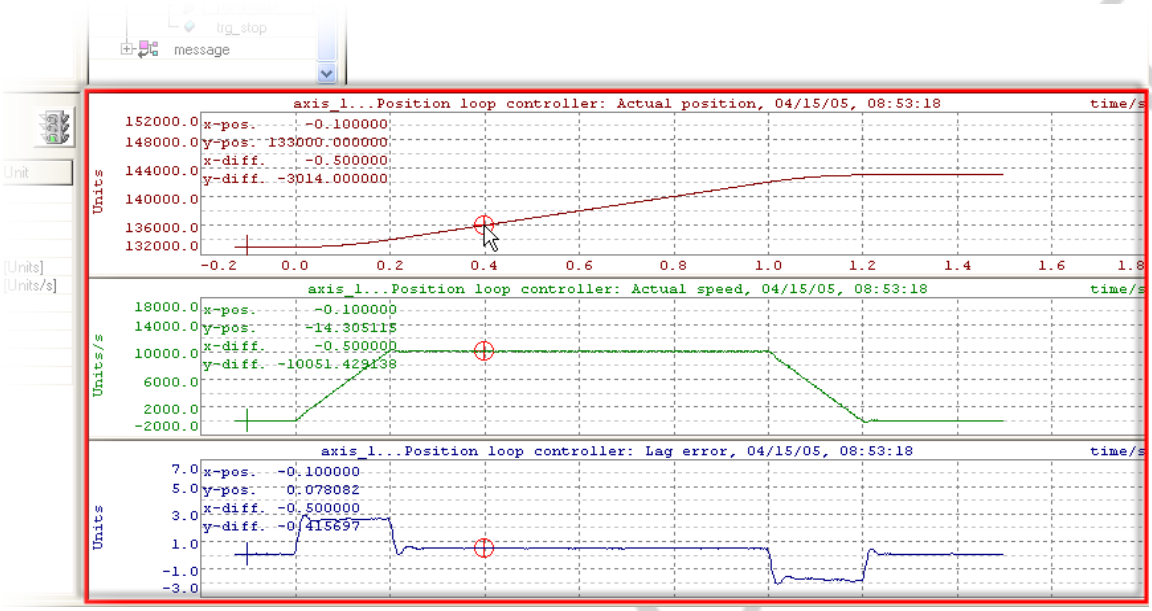


Fig. 78: Example for a parameter trace

Different measurement tools make it possible to accurately analyze the data.

Note:

Like NC Watch, the Trace function can be operated independent of the NC test by opening it directly for the NC object.

A few application examples for the cyclic trace:

- Setting the control parameters
- Checking the motor load
- Checking the load on the ACOPOS
- Checking the positioning sequence
- Troubleshooting

  Etc.

Configuration of the cyclic trace

The dialog box for configuring how the values are recorded is opened by **right-clicking** in the **Trace window** and selecting **Configuration**.

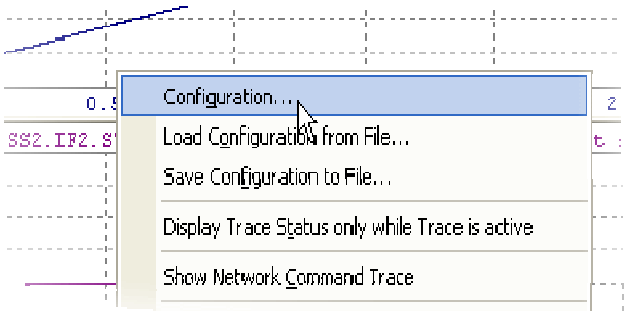


Fig. 79: Opening trace settings

The Trace configuration is divided into two areas separated by tabs:

- Data Points
- Timing

Data point configuration:

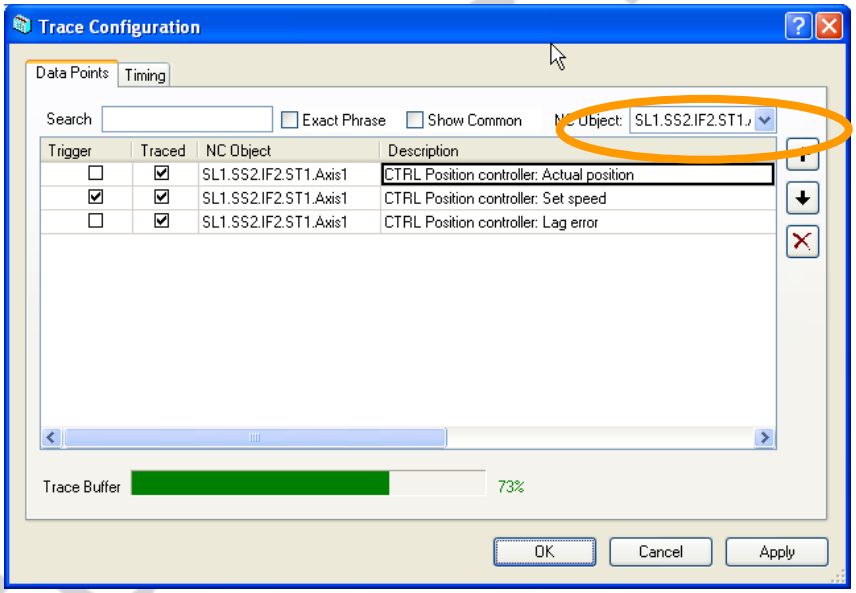


Fig. 80: Trace - Data Points

NC Object:
Axis whose parameters should be recorded. Either the name (if present) or the address is recorded.

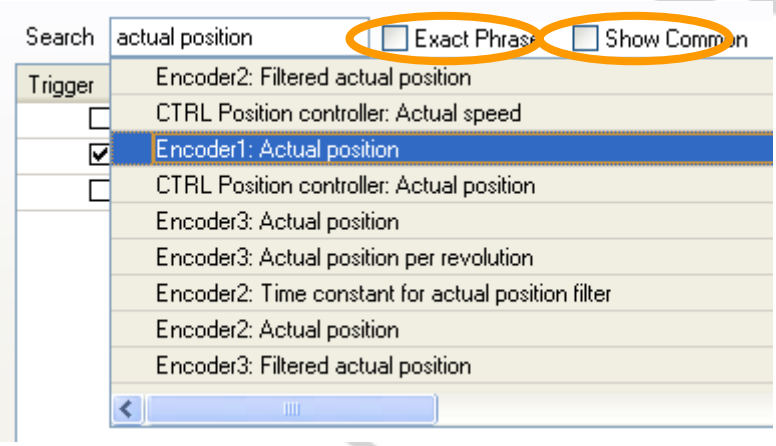
Note:

A multi-axis trace is automatically started if parameters from different NC objects (axes) are recorded.

Data points are selected via full-text search:

- The search is non-case-sensitive
- Every single word is linked with AND
- Searches are based on descriptions, constants and Par IDs

The search results are expanded after entering the search criteria in the **Search** field:



The focus is changed to the list using **Enter** or the **"Down arrow"** key. The list is closed using the **Esc** key. Use the mouse or double-click to select an entry for adding to the list of data points to be recorded.

Checkbox 1: Exact Phrase:
In this case, the entire search expression is searched for (not the individual words).

Checkbox 2: Show Common:
Displays standard basic parameters.

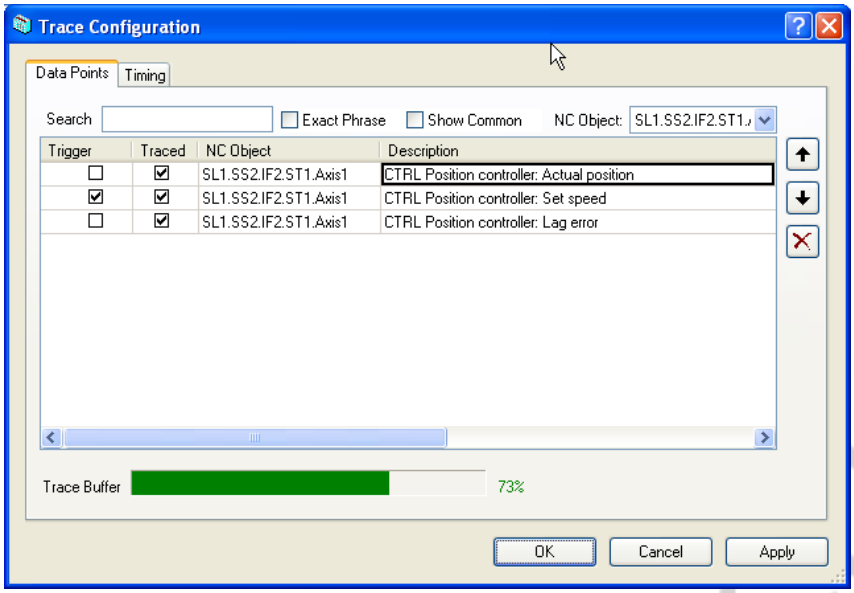


Fig. 81: Trace - Data Points

Trigger column
Determines the parameters whose value should be triggered

Traced column
Determines the parameters that should be recorded. This allows you to acquire parameters from a (quickly accessible) pool without having to load a new configuration.

Note:

Recording more data points within a certain amount of time fills up the Trace buffer more quickly. The fill level indication provides information about the current capacity. The buffer is colored red if there is not enough space for recording the configured amount of data. In this case, the effective recording duration is automatically shortened.

Timing :

The timing for the trace including trigger is defined in the "**Timing**" tab:

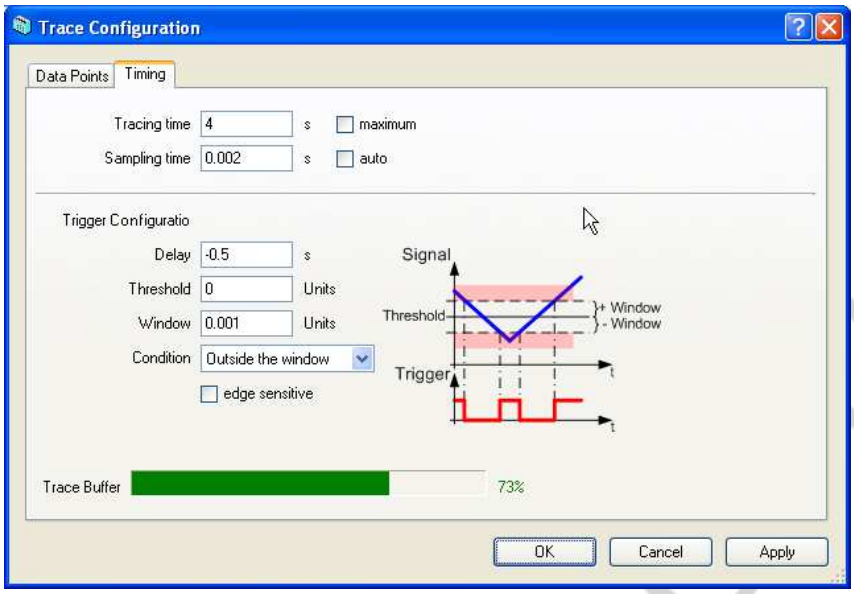


Fig. 82: Trace timing

Tracing time
Used to determine the recording duration. This time is accordingly reduced if the trace buffer becomes full due to the amount of data for the selected settings (many parameters, high sampling rate). The user is notified of this via a message window.
This setting can also be calculated automatically by selecting the "**maximum**" checkbox.

Sampling time
The time interval at which the current parameter values are added to the buffer. This setting also directly affects the amount of data for the record: **higher sampling rate → more memory required → potentially shortened trace duration ...**

Trigger configuration:

- **Delay**
  A delay time can be set here for the actual start of recording after the trigger event occurs:
  This setting can be automatically calculated. The calculation is made by attempting to fully load the Trace buffer with minimum sampling time.

- **Threshold**
  If a parameter is needed to trigger the configured trigger event, this threshold value determines the value that must be exceeded for the trigger to happen.
- **Trigger window**
  Specifies a window for certain trigger events. The middle of the window is the threshold value, and the window extends above and below the configured value.
- **Condition**
  Specifies a certain condition that acts as the trigger when it occurs.

Sample configurations:

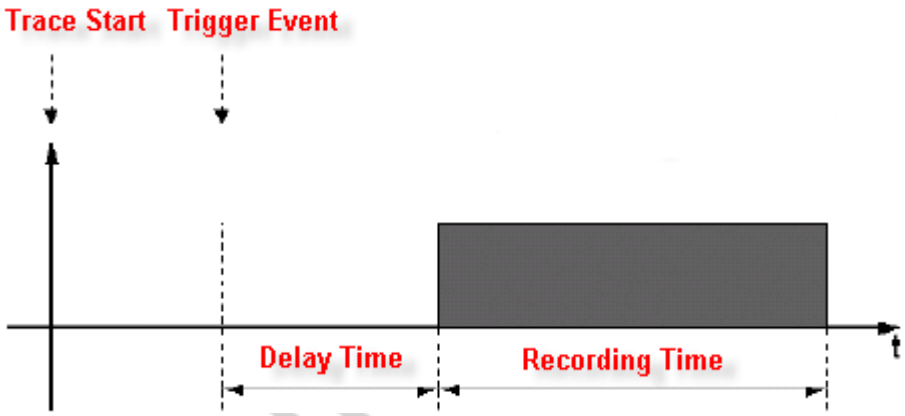


Fig. 83: Positive trigger delay

This is shown in the above diagram for a positive delay time. However, it is also possible to record with negative trigger delay:

Fig. 84: Negative trigger delay

As illustrated in the image above, this can also be used to record values for a specific amount of time **before** the trigger event occurs.

Note:

This option could, for example, be used when searching for errors. The error criteria to be searched for (e.g. "lag error larger than 10 units") must be set as trigger event. Using a negative trigger delay, the behavior of relevant parameters can be monitored before this condition occurs.

Options for starting the recording

The trace can now be started using one of the following methods:

- **Start trace as configured** via **Trace/Activate** or the  button on the toolbar. Recording is then started after the set trigger event has occurred.

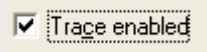
- **Start trace immediately** via **Trace/Start** or the  button on the toolbar. The set trigger event is replaced by **ncOFF** when using this method to start the trace. This means that the trace starts recording immediately.

Evaluating the Trace results

After the recording is finished, the data is transferred to Automation Studio via the controller and the course of the parameter values is displayed in charts placed above one another.

Note:

The "Trace enabled" checkbox can be used to link each individual action with the start of the trace recording.



This setting is made separately in the action field for each action. If it is selected, the corresponding Trace (with trigger condition) is automatically enabled before starting the action.

Chart properties:

Different settings for displaying the value curves can be defined and changed in the "Chart Properties".

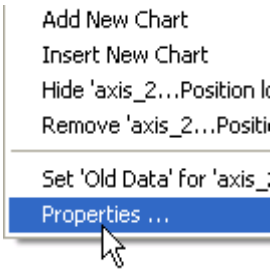


Fig. 85: Opening the chart properties

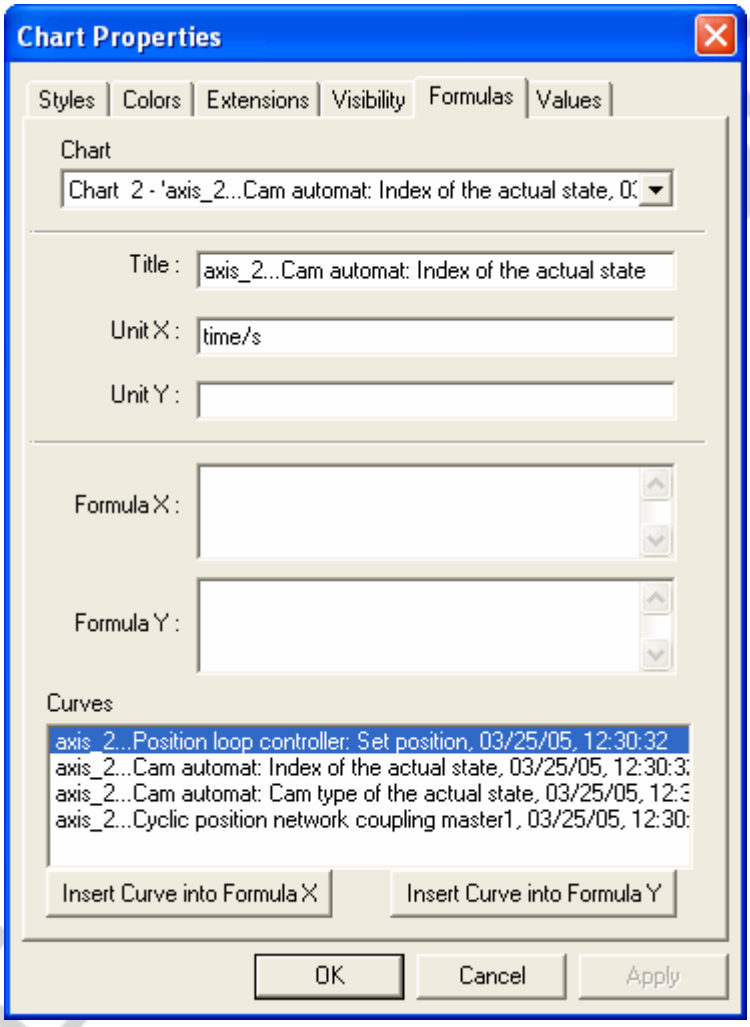
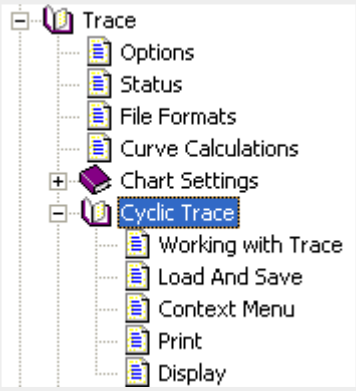


Fig. 86: Trace chart properties

It is also possible to create custom charts. This is one way to perform e.g. analyses of mathematic calculations in a separate chart.

Note:

Detailed information about operating the cyclic Trace can be found in the Automation Studio online help.



Task: Cyclic Trace operation

With your drive, carry out a relative movement and record the following values:

- Actual position
- Actual speed
- Set position
- Lag error

Select the Trace parameters so that the entire movement process is recorded.

- Use the actual position in the diagram and the measurement tool to determine the traverse distance.
- Calculate the lag error using the "set position – actual position" in a different diagram.

4.6.2 Network Command Trace

The network command trace makes it easy to record communication **between the NC manager and the ACOPOS**.

This function is very helpful for checking the order in which the commands were given to the controller by the tasks. This makes it possible to quickly discover any errors in the sequence.

The command trace is an expansion of the NC Trace and is therefore also opened in the Trace environment.

The network command Trace is opened by **right-clicking** on the NC Trace interface and by using the command **Show Network Command Trace**. This method can also be used to switch back to the cyclic Trace.

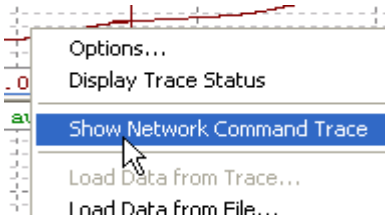


Fig. 87: Opening Network Command Trace

The Network Command Trace is constantly active. Depending on the defined size for the recording buffer (dialog box: **NC Properties**), a certain number of entries can be called up. The records from the trace are loaded by **right-clicking** and selecting **Load Data from Trace**.

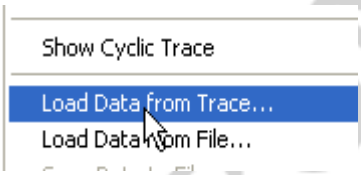


Fig. 88: Loading the records

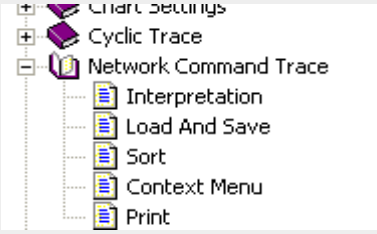
The Network Command Trace shows a list of entries. These provide information about the parameters sent from the NC manager to the ACOPOS and about the corresponding response from the ACOPOS module.

Index	Interface	Drive	NC Object Type	Request	Time [s]	Time [s]	Response	Index
1	CAN[0]	2		← STATUS_BITS	12.610			
	CAN[0]	2				12.610	00000000 00000000 0...	2
3	CAN[0]	2		← STATUS_BITS	12.810			
	CAN[0]	2				12.810	00000000 00000000 0...	4
5	CAN[0]	2		← STATUS_BITS	13.010			
	CAN[0]	2				13.010	00000000 00000000 0...	6
7	CAN[0]	2		→ CMD_CANID_ACOPOS_STD	13.010			
	CAN[0]	2				13.010	→	8
9	CAN[0]	2		← TRACE_MAX_DATLEN	13.010			
	CAN[0]	2				13.010	← 32768	10
11	CAN[0]	2	ncAXIS	→ PHASE_MON_IGNORE = 1	13.020			
	CAN[0]	2				13.020	→	12
13	CAN[0]	2	ncAXIS	→ UDC_NOMINAL = 320 V	13.020			
	CAN[0]	2				13.020	→	14
	CAN[0]	2				13.050	00000000 00001000 0...	15
16	CAN[0]	2	ncAXIS	→ NETWORK_LIVE_CTRL = 40000 us	13.060			
	CAN[0]	2				13.060	→	17
18	CAN[0]	2	ncV_AXIS	→ OVERRIDE_VAX1	13.070			
	CAN[0]	2				13.070	→	19
20	CAN[0]	2	ncAXIS	→ OVERRIDE	13.080			
	CAN[0]	2				13.080	→	21

Fig. 89: Network Command Trace

Note:

Detailed information about the Trace network command settings can be found in the Automation Studio online help.



For CNC applications, an additional tool is available to analyze the command traffic, the DPR Command Trace. Detailed information about this can be found in the **TM470 CNC** training module.

4.7 Other services

4.7.1 Service interface

The NC software makes it possible to also specifically access individual parameters on the ACOPOS. The "Service interface" is used to do this.

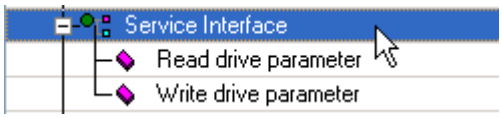


Fig. 90: Service interface

The desired parameter ID is entered in the **"request"** element under **"par_id"**:

Fig. 91: Entering the desired parameter ID

The **parameter can now be read out** directly using the corresponding command.

When **writing a parameter**, the desired value must be entered in the "**service**" element under the **"data_text"** entry:

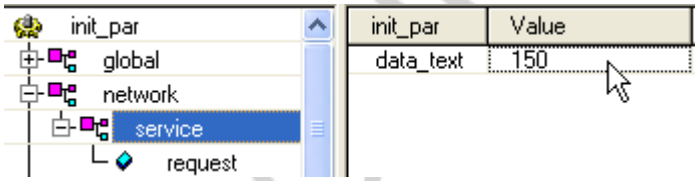
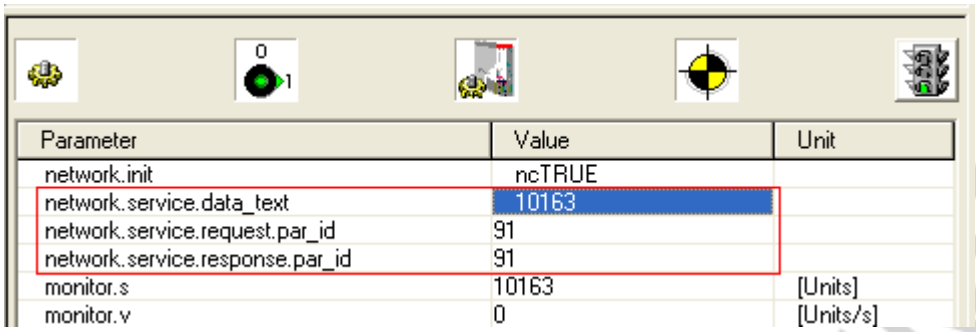


Fig. 92: Entering the desired value for the Par ID

The command for writing/reading the parameter can then be sent.

Response about the action is provided in the **NC Watch**. The relevant parameters can be displayed here:



After a read or write action has been successfully completed, the response ParID ("...service.response.par_id") is compared with the request ParID ("...service.request.par_id").

Note:

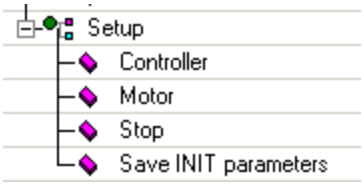
For a faulty service operation (e.g. attempting to write to a "ReadOnly" parameter), the response ParID is set to 65535 ($FFFF_{hex}$). A corresponding error message is also generated in NC Watch.

Task: Operating the service interface

Try reading various ACOPOS parameters using the service interface in the NC test.

- Par ID 91: "Encoder1: Actual position"
- Par ID 380: "Power stage: Heatsink temperature sensor: Temp."
- Par ID 298: "CTRL DC bus: Voltage"

4.7.2 Setup



The following services are intended to provide some assistance when setting up the drive system:

- Automatic setting of the drive controller
- Automatic determination of the motor characteristics (IM)

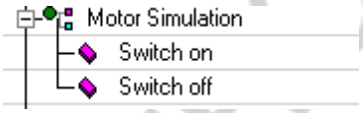
Note:

Up-to-date information about using this function can be found in the Automation Studio online help.

4.7.3 Motor simulation

The ACOPOS can also be operated without a motor in this mode. Motor behavior is simulated. This makes it possible to run through almost any procedure (movements, etc.).

This mode can very easily be enabled or disabled in the command window:



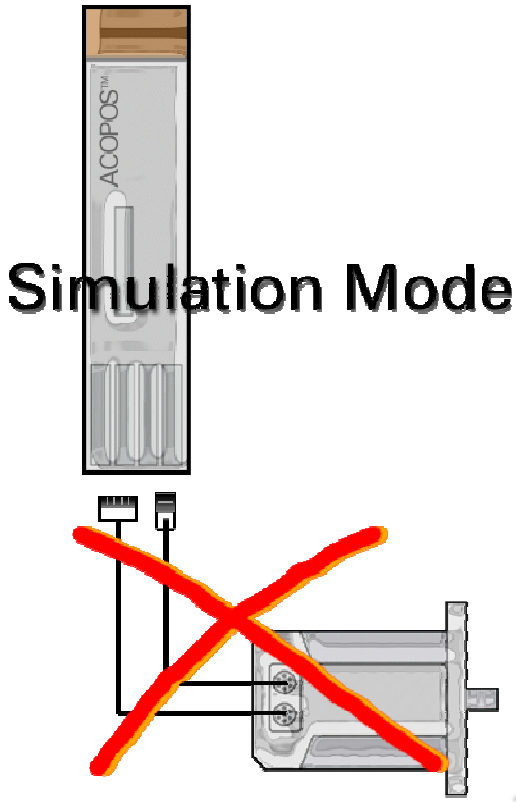


Fig. 93: Simulation Mode

In this mode, you can test the parts of the application program with the ACOPOS servo drive that are hardware independent without the hardware components such as motor or encoder being connected.

Task: Motor simulation

Activate motor simulation and test the different movement actions in this mode.

5. SUMMARY

Drive objects are managed centrally in Automation Studio.

The NC manager uses the controller to communicate with the NC operating system on the ACOPOS over an entire network.

The NC Manager is the contact point for controlling the drives. It handles the transfer of parameters and commands to the ACOPOS as well as the status information updates for the user.

Fig. 94: Automation Studio with integrated drive concept

Diagnostic tools provide an efficient environment for tests and implementation.

This considerably reduces engineering expenditure for software as well as for start-up and maintenance.

Notes:

Overview of training modules

TM200 – B&R Company Presentation **
TM201 – B&R Product Spectrum **
TM210 – The Basics of Automation Studio
TM211 – Automation Studio Online Communication
TM212 – Automation Target **
TM213 – Automation Runtime
TM220 – The Service Technician on the Job
TM223 – Automation Studio Diagnostics
TM230 – Structured Software Generation
TM240 – Ladder Diagram (LAD)
TM241 – Function Block Diagram (FBD)
TM246 – Structured Text (ST)
TM247 – Automation Basic (AB)
TM248 – ANSI C
TM250 – Memory Management and Data Storage
TM260 – Automation Studio Libraries I
TM261 – Closed Loop Control with LOOPCONR

TM400 – The Basics of Motion Control
TM410 – The Basics of ASiM
TM440 – ASiM Basic Functions
TM441 – ASiM Multi-Axis Functions
TM445 – ACOPOS ACP10 Software
TM446 - ACOPOS Smart Process Technology
TM450 – ACOPOS Control Concept and Adjustment
TM460 – Starting up Motors

TM500 – The Basics of Integrated Safety Technology
TM510 – ASiST SafeDESIGNER

TM600 – The Basics of Visualization
TM610 – The Basics of ASiV
TM630 – Visualization Programming Guide
TM640 – ASiV Alarm System
TM650 – ASiV Internationalization
TM660 – ASiV Remote
TM670 – ASiV Advanced

TM700 – Automation Net PVI
TM710 – PVI Communication
TM711 – PVI DLL Programming
TM712 – PVIServices
TM730 – PVI OPC

TM800 – APROL System Concept
TM810 – APROL Setup, Configuration and Recovery
TM811 – APROL Runtime System
TM812 – APROL Operator Management
TM813 – APROL XML Queries and Audit Trail
TM830 – APROL Project Engineering
TM840 – APROL Parameter Management and Recipes
TM850 – APROL Controller Configuration and INA
TM860 – APROL Library Engineering
TM865 – APROL Library Guide Book
TM870 – APROL Python Programming
TM890 – The Basics of LINUX

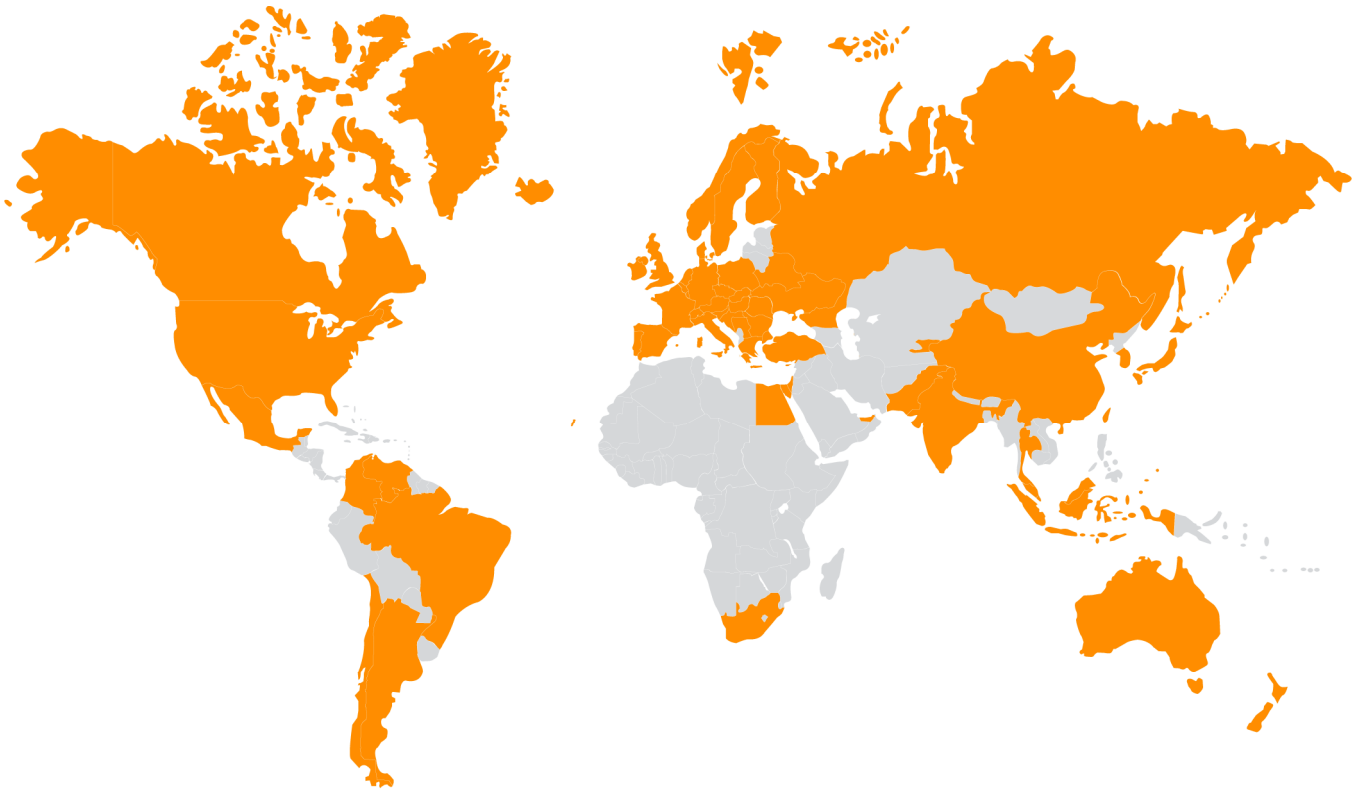
**) see Product Catalog

KONZERNZENTRALE

Bernecker + Rainer Industrie-Elektronik Ges.m.b.H.
B&R Strasse 1
5142 Eggelsberg
Austria
Tel.: +43 (0) 77 48 / 65 86 - 0
Fax: +43 (0) 77 48 / 65 86 - 26
info@br-automation.com
www.br-automation.com

TM410TRE.30-ENG 0907

Immer in Ihrer Nähe - 140 Büros in über 55 Ländern - **www.br-automation.com/contact**



Australia • Argentina • Austria • Belarus • Belgium • Brazil • Bulgaria • Canada • Chile • China • Colombia • Croatia • Cyprus
Czech Republic • Denmark • Egypt • Emirates • Finland • France • Germany • Greece • Hungary • India • Indonesia
Ireland • Israel • Italy • Japan • Korea • Luxemburg • Kyrgyzstan • Malaysia • Mexico • The Netherlands • New Zealand
Norway • Pakistan • Poland • Portugal • Romania • Russia • Serbia • Singapore • Slovakia • Slovenia • South Africa
Spain • Sweden • Switzerland • Taiwan • Thailand • Turkey • Ukraine • United Kingdom • USA • Venezuela • Vietnam