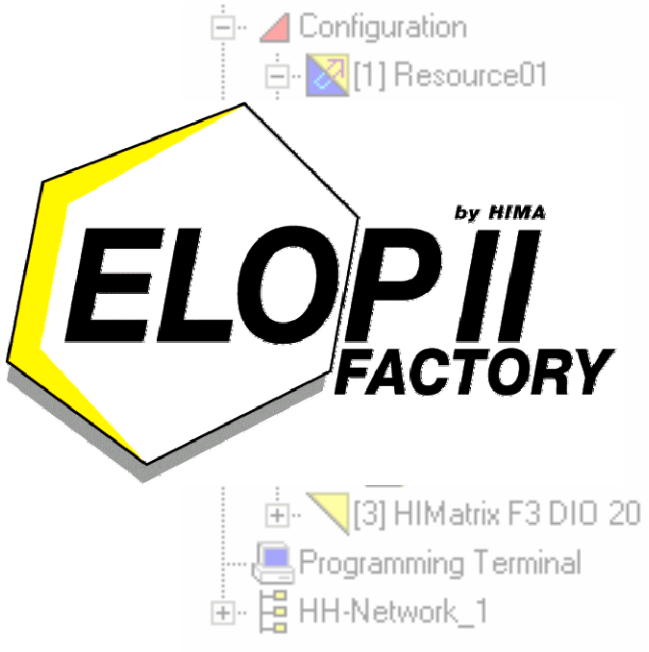
PC-based systems

ELOP II Factory

Project Management Version 4.1 build 6111
and Hardware Management-Version 7.56.10 or higher

First Steps



HIMA Paul Hildebrandt GmbH + Co KG
Industrial Automation

HI 800 006 CEA

For more information see the documentation on the CD-ROM and on our web site at www.hima.de.

More information can be requested from:

HIMA Paul Hildebrandt GmbH + Co KG
Postfach 12 61
68777 Brühl

Tel: +49 (6202) 709 0
Fax: +49 (6202) 709 107

E-mail: info@hima.com

Contents

1 General Overview

1.1 Shipping List

ELOP II Factory includes:

- This manual
  The "First Steps“ manual is a quick and easy start for learning ELOP II Factory. It includes an overview of most functions and step-by-step instructions on creating a project.
- A CD-ROM
  The CD-ROM contains the ELOP II Factory software, some auxiliary programs and the complete documentation for the current HIMatrix systems.
- A hardlock (dongle)
  The dongle administers the licence (protection against unauthorised use) of the protected ELOP II Factory software.

1.2 Information on this Manual

Users will find useful information in this manual to help them learn the most important functions of ELOP II Factory in training courses or in self-training.

Chapter 2 explains how to install ELOP II Factory. Chapters 3 to 6 give a general description of using and operating ELOP II Factory. Users without prior knowledge should read these sections thoroughly.

Chapter 7 contains a sample project that users with a basic knowledge of ELOP II Factory can use to learn project creation or to increase their skills.

The appendix in chapter 8 contains explanations of the technical terms used in the text, the index and the list of abbreviations.

1.3 Support

There are various options for finding answers to questions on operating the program or to report errors in the program and suggestions for improvement.

Frequently Asked Questions	Chapter in this book	Questions and answers on basic subjects
News, FAQs, Download	Our web site www.hima.com	New items, frequently asked questions, function blocks
Questions and suggestions	By e-mail: Support@hima.com Telephone: +49-(0)6202-709 185 Fax: +49-(0)6202-709 199	Between 9:00 a.m. and 5:00 p.m.

Note: *This manual is part of the working documentation for the ELOP II Factory seminars at HIMA. Because of the comprehensive nature of ELOP II Factory, only the most important functions of the program can be described here.*

We recommend attending a seminar for more information.

2 Installation

In this chapter:

- What is needed?
- Where is the program installed?
- How do I start the installation?
- Installation in a network
- Uninstall

ELOP II Factory is protected by a hardware lock. The hardlock module (dongle) must either be connected to the parallel or USB port.

Fig. 1: Hardlocks for printer port and USB-port

A driver must be installed on the computer so that the hardlock module can be addressed. To install the driver in Windows 2000/XP® the user must be logged as an administrator. If there is any doubt contact the system administrator.

2.1 What is needed?

In addition to the personal computer, the hardlock and the installation CD are required.

The computer hardware requirements are as follows:

	Minimum	**Recommended**
Processor	Intel Pentium III® 800 MHz	Intel Pentium IV® 3 GHz
RAM	256 MB	1024 MB
Graphic adapter	8MB XGA 1280x1024 TRUE-Colour	128MB XGA 1280x1024 TRUE-Colour
Hard disk	Min. 500MB for ELOP II Factory plus space for user program	
Operating system	Windows 2000® Professional with Service Pack 1 or higher Windows XP® up to Service Pack 2	

Table 1: Hardware requirements

If a printer is connected to the Hardlock, the printer should be switched on, because some printers have insufficient terminating resistance when they are switched off.

2.2 Where is the Software installed?

The software will be installed on the local hard disk.

With this version ELOP II Factory needs to be installed only once and it is accessible by all users who have at least Main user rights.

2.3 How is the Installation started?

ELOP II Factory and all other components are installed from the installation menu of the CD-ROM.

The CD-ROM also contains the complete documentation for the software and the HIMatrix system family as PDF files. Adobe Acrobat Reader©, which is required to read the PDF files, is also included.

Note: Make sure that you are logged in to the computer with Administrator rights when you install the hardlock driver.

2.3.1 Starting the Installation:

1. Insert the CD-ROM into the CD drive. The CD menu automatically opens. If the menu does not open automatically, open the root directory of the CD-ROM in Windows Explorer manually and double-click the file "setup.bat".
2. Select the desired language for guiding through the menu.
3. Select the Hardlock Software you want to install.

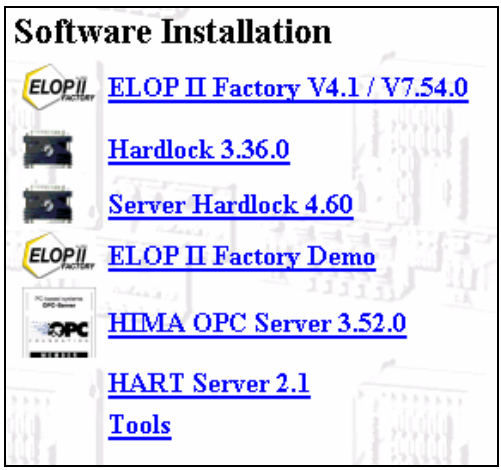


Fig. 2: Installing the Hardlock driver

4. Follow the installation prompts.
5. After installation of the hardlock driver, click on **ELOP II Factory V4.1 / 7.56.10** to install ELOP II Factory.
6. Select if you like to have desktop-icons and/or start menu entries. Minimum should be start menu entries. The icons you can add later.

After the successful installation the ELOP II Factory Control Center opens. You can start ELOP II Factory and auxiliary programs from here. Later you can open the Control Center and Project Management in the ELOP II Factory folder (see also chapter 7) from the Windows **Start** menu.

Fig. 3: ELOP II Factory Start menu

2.4 Uninstall

For older versions the complete uninstall must be carried out in three steps in the following sequence:

- ☐ Uninstall of the Hardware Management.
- ☐ Uninstall of the Project Management.
- ☐ Uninstall of the hardlock driver.

Note: *If ELOP II Factory is updated, Hardware Management and Project Management must be uninstalled first. It is not necessary to uninstall the hardlock driver.*
Restart the computer before running an update.

ELOP II Factory version 4.1 and newer can be uninstalled in a single step (see chapter 2.4.1).

2.4.1 Uninstall of the Hardware Management

☐ Open the Windows® Control Panel and double-click the "Add/Remove Programs" icon.

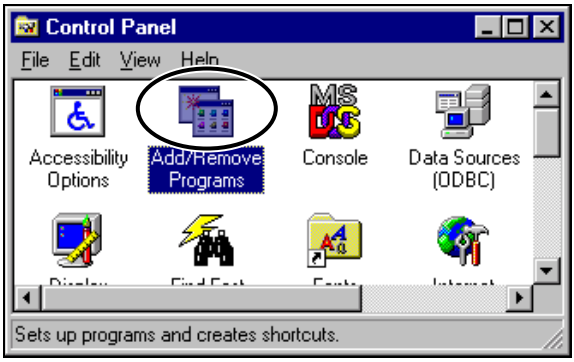


Fig. 4: Uninstalling software

☐ In the "Add/Remove" window click on "ELOP II Factory Hardware Management“ and then click on the **Add/Remove...** button.

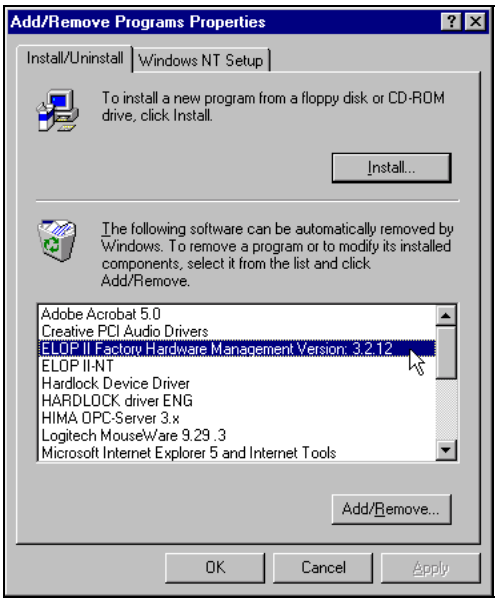


Fig. 5: Uninstalling Hardware Management

2.4.2 Uninstall of the Project Management

☐ Open the ELOP II Factory Control Centre and select **Uninstall**. Then select the program that you want to uninstall, depending on the security settings of your browser.

Fig. 6: Opening the Control Centre

☐ Select **Administration** and run the **Uninstall** function.

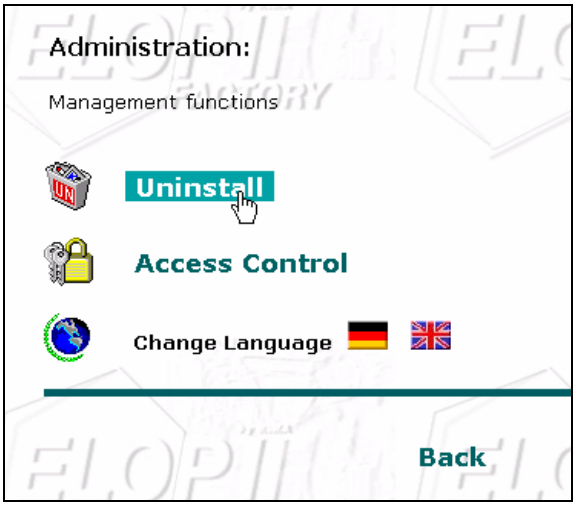


Fig. 7: Uninstalling Project Management

2.4.3 Uninstall of the Hardlock Driver

☐ Open the Windows® Control Panel and double-click the **Add/Remove Programs** icon (see chapter 2.4.1).

☐ In the "Add/Remove" window click on **Hardlock Device Driver** and then click on the **Add/Remove...** button.

3 Introduction to ELOP II Factory

In this chapter:

- Starting the program
- Components included in the ELOP II Factory interface
- Menu and title bar
- Toolbar and status bar
- Window layout, structure window and working area
- Error-State-Viewer

ELOP II Factory is a program with numerous functions, but the intuitive Windows-style user operation makes access easy.

3.1 Starting ELOP II Factory

Select the **Programs** folder from the Windows **Start** menu and then **ELOP II Factory**.

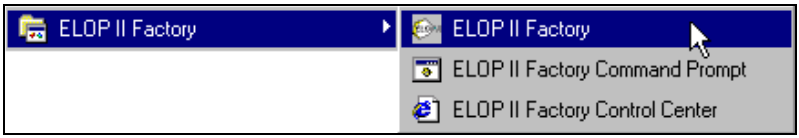


Fig. 8: ELOP II Factory Start menu

Alternatively you can also start the Project Management in the ELOP II Factory Control Centre, or double-click the  icon on the desktop.

ELOP II Factory comprises two windows:

1. **Project Management** for creating all user programs and for archiving or restoring projects.
2. **Hardware Management** for defining all hardware-specific data. Hardware Management is only opened when a project is created or opened.

3.2 Screen Layout of the Project Management

After starting ELOP II Factory the standard screen appears as shown in Fig. 9. The standard screen generally includes the following components:

1. Title bar
2. Structure window
3. Menu bar
4. Toolbar for Project Management
5. Working area
6. Toolbar for the Function Block Diagram editor (FBD editor)
7. Error-State-Viewer
8. Status bar with coordinate information of the function plan editor

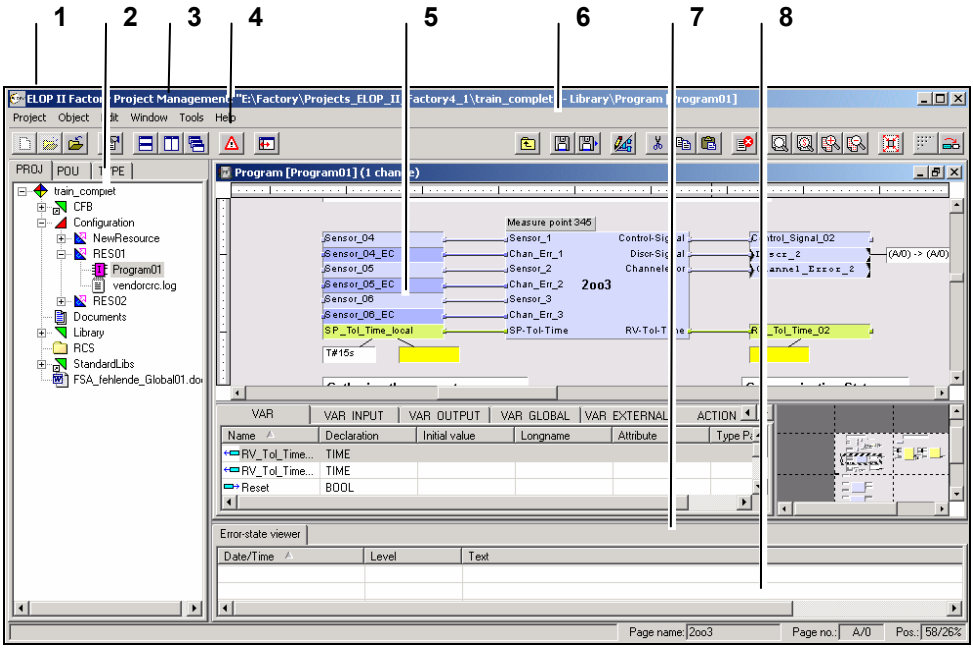


Fig. 9: ELOP II Factory standard screen

3.2.1 Title Bar

As well as the standard functions for minimising, maximising and closing the window, the title bar also includes information on the project and the edited function block.

1. Program
2. Open project
3. Open function block
4. Minimise, maximise or overlap, close

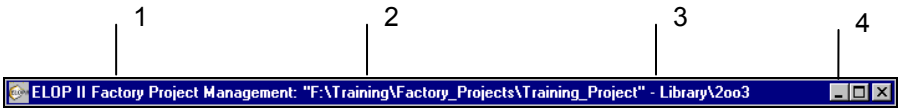


Fig. 10: Title bar

3.2.2 Menu Bar

The menu bar is used to operate ELOP II Factory. Most ELOP II Factory functions are available in the menu bar.

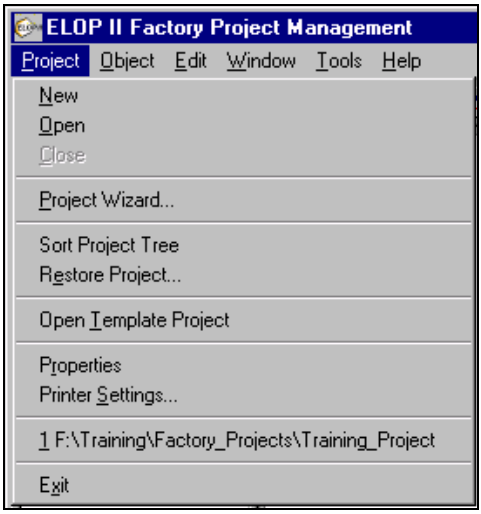


Fig. 11: The "Project" menu

3.2.3 Toolbar

The toolbar is displayed below the menu bar and divided into the two sections “Project Management” and “Function Block Diagram Editor”.

Fig. 12: Toolbar for the project

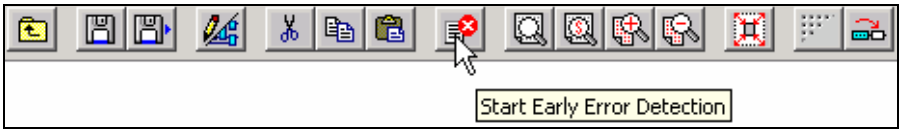


Fig. 13: Toolbar for the Function Block Diagram editor

Note: *When the mouse pointer is positioned briefly over a button, a Quick Info (short help text) is displayed.*

3.2.4 Status Bar

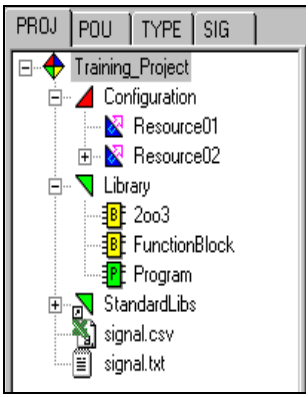
The status bar at the bottom of the window shows information and help texts on the Project Management and the Function Block Diagram editor and also the current pointer position.

Fig. 14: Status bar

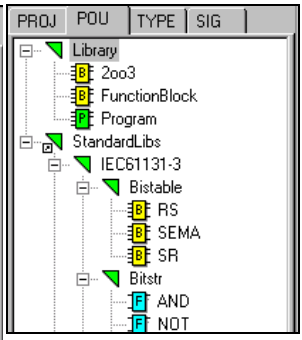
3.2.5 Structure Window

The structure window shows the hierarchical structure of the project. You can select one of three views with different degrees of detail.

Total project



All POUs[1]



All data types

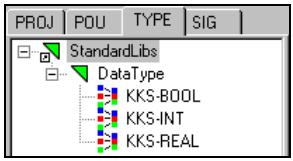


Fig. 15: Structure window

[1] **P**rogram **O**rganisation **U**nit

3.2.6 Context Menu for Objects

Right-click on an object in the structure window to open the context menu associated with that object. Select a command as usual by clicking with the left mouse button.

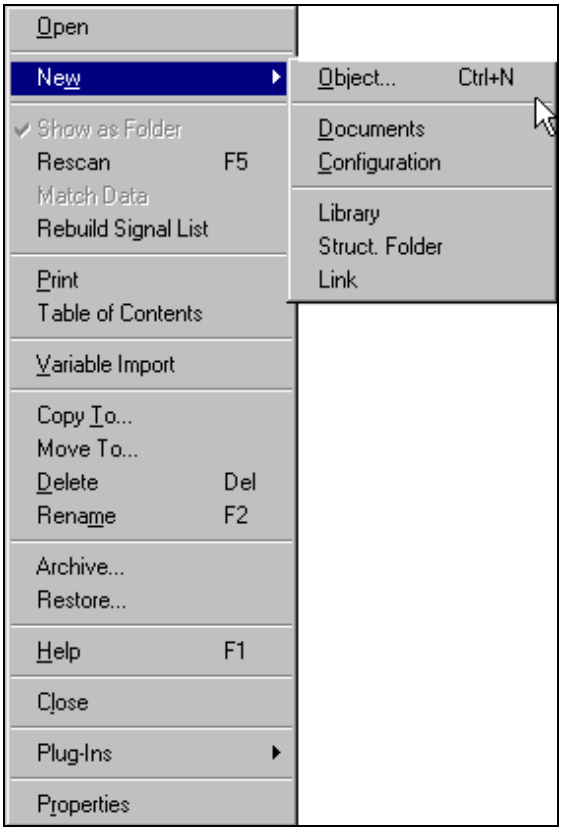


Fig. 16: «Project» context menu

3.2.7 Working Area

In the working area (see Fig. 9) data objects are edited with the

- Function Block Diagram editor and the
- Document editor.

3.2.8 Function Block Diagram Editor (FBD Editor)

With the FBD editor you create the function diagrams in the Function Block Diagram Language (FBD) or in the Sequential Function Chart Language (SFC).

The FBD editor (Fig. 17) comprises the following panes:

1. Drawing field
2. Variable declaration editor
3. Overview window
4. Interface declaration editor

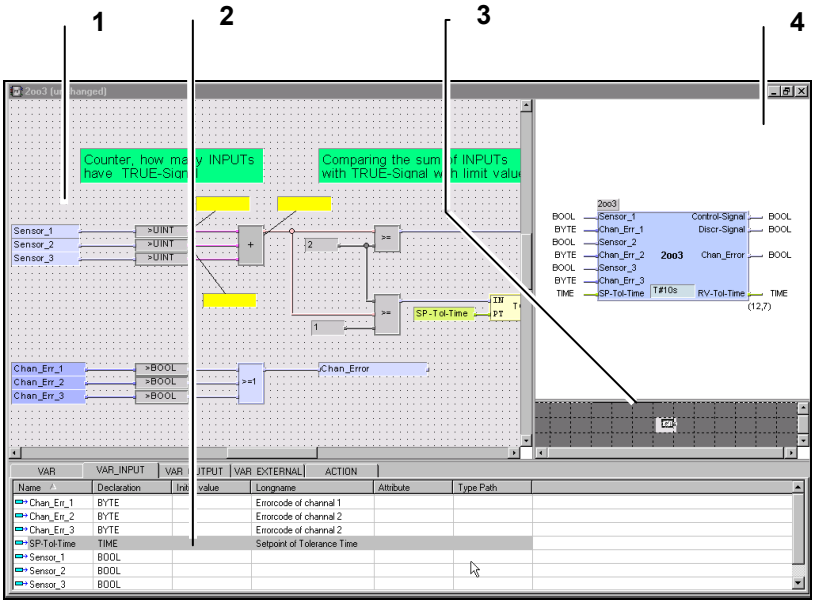
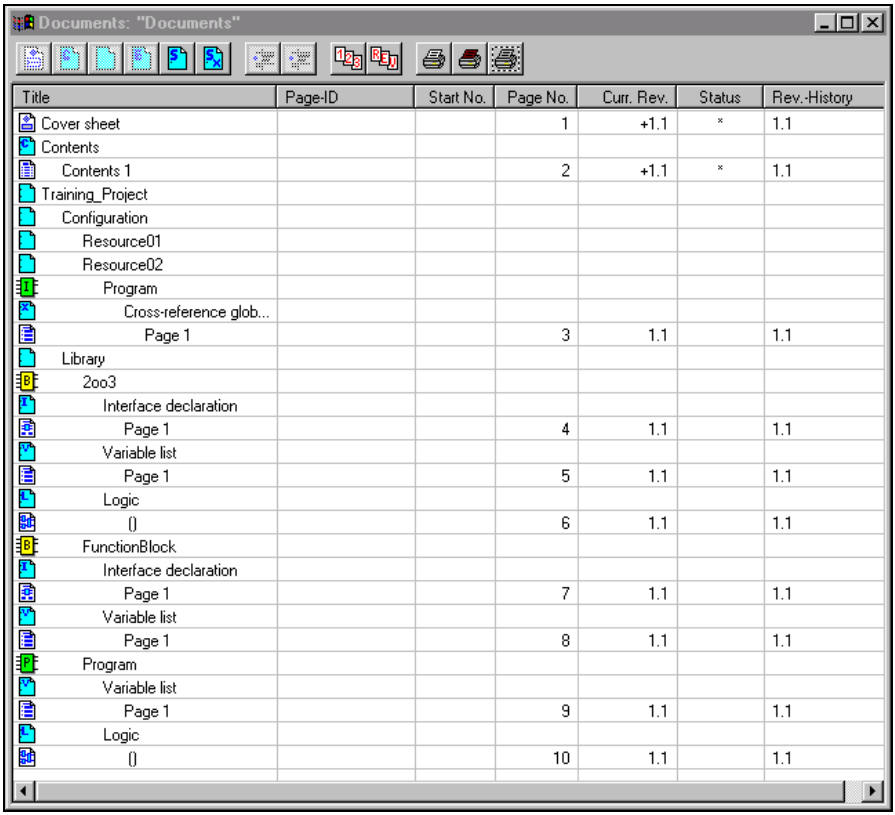


Fig. 17: Function Block Diagram editor

3.2.9 Document Editor

The document editor is used to create the documentation from the objects in the structure window. A common revision management can be used for all documents.



Title	Page-ID	Start No.	Page No.	Curr. Rev.	Status	Rev.-History
Cover sheet			1	+1.1	*	1.1
Contents						
Contents 1			2	+1.1	*	1.1
Training_Project						
Configuration						
Resource01						
Resource02						
Program						
Cross-reference glob...						
Page 1			3	1.1		1.1
Library						
2oo3						
Interface declaration						
Page 1			4	1.1		1.1
Variable list						
Page 1			5	1.1		1.1
Logic						
()			6	1.1		1.1
FunctionBlock						
Interface declaration						
Page 1			7	1.1		1.1
Variable list						
Page 1			8	1.1		1.1
Program						
Variable list						
Page 1			9	1.1		1.1
Logic						
()			10	1.1		1.1

Fig. 18: Document editor

3.2.10 Error-State-Viewer

The Error-State-Viewer is the central point for displaying error and status messages. The occurrence of a new message is indicated by a flashing icon in the Windows task bar.

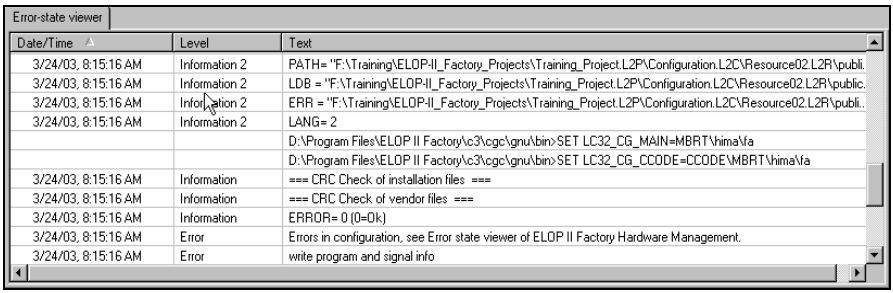


Fig. 19: Error-State-Viewer

3.2.11 Online Help

The online help contains detailed explanations for all functions in ELOP II Factory. You can use the index to find help on keywords quickly.

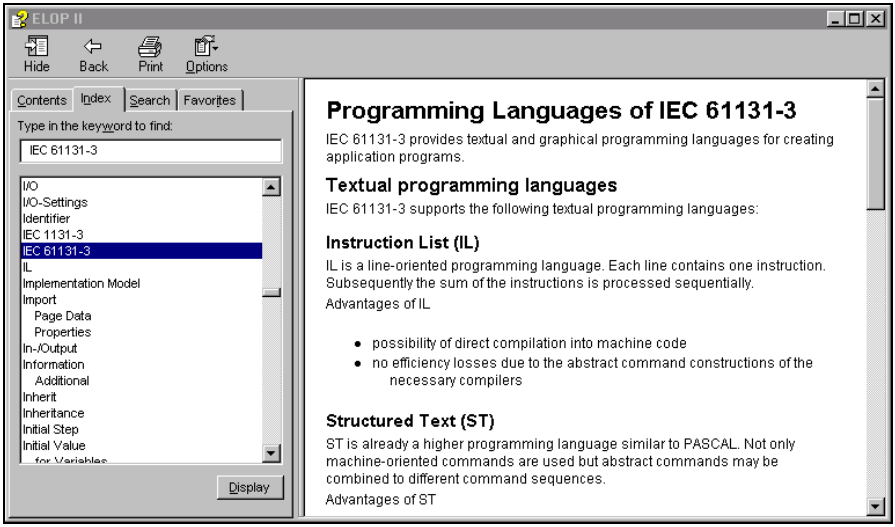


Fig. 20: Online help

3.3 Hardware Management Screen Division

1. Title bar
2. Menu bar
3. Project tree
4. Working area
5. Error-State-Viewer

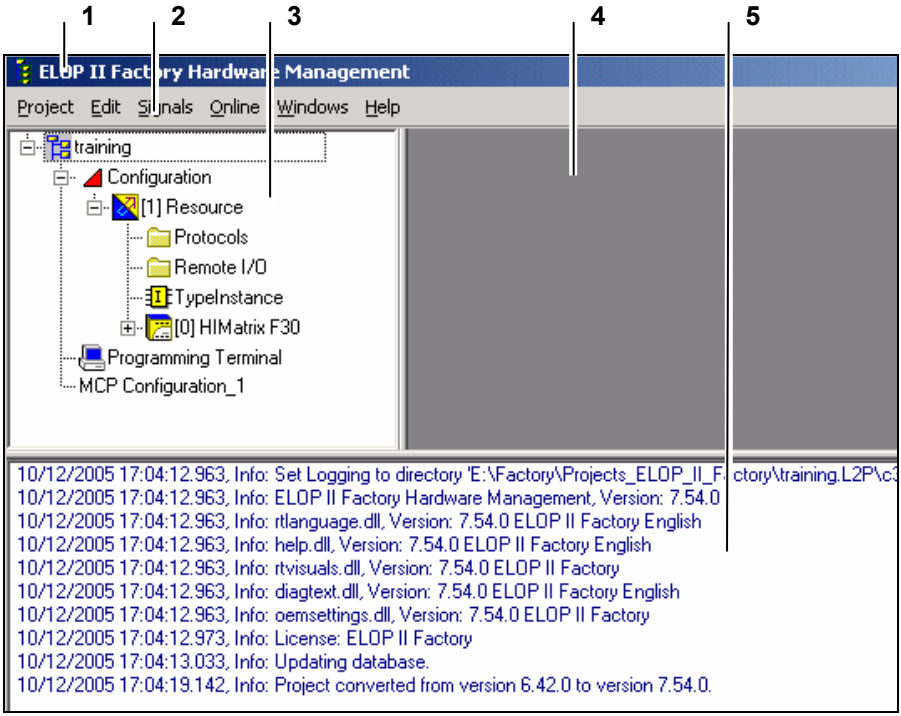


Fig. 21: Standard screen in the Hardware Management

The Hardware Management is controlled via the menu bar:

- ☐ Click on the menu to open it.
- ☐ Select the option that you want to execute. Click the left mouse button to execute the command.

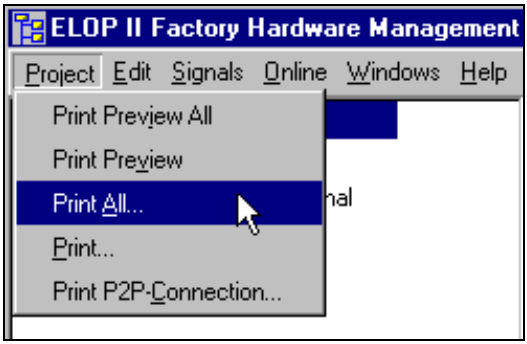


Fig. 22: «Project» menu

3.3.1 Structure Window

1 Project name
2 Configuration
3 Resource folder
4 Program instance
5 Communication protocols
6 Remote I/O folder
7 Remote I/O type
8 Components and modules
9 Resource type

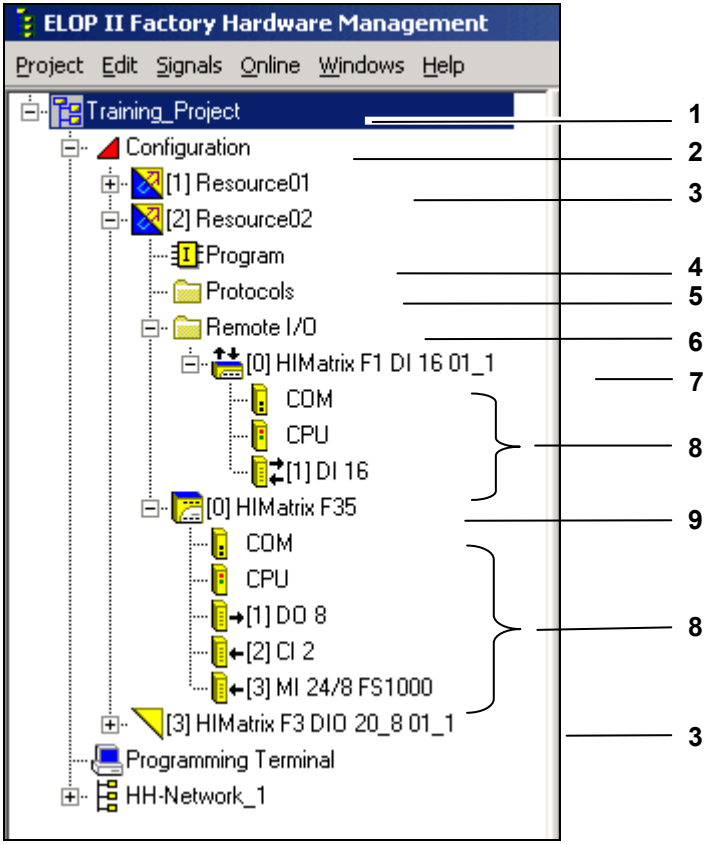


Fig. 23: Structure window

3.3.2 Context Menu

Right-click on an object in the structure window to open the context menu associated with that object. Select a command as usual by clicking with the left mouse button.

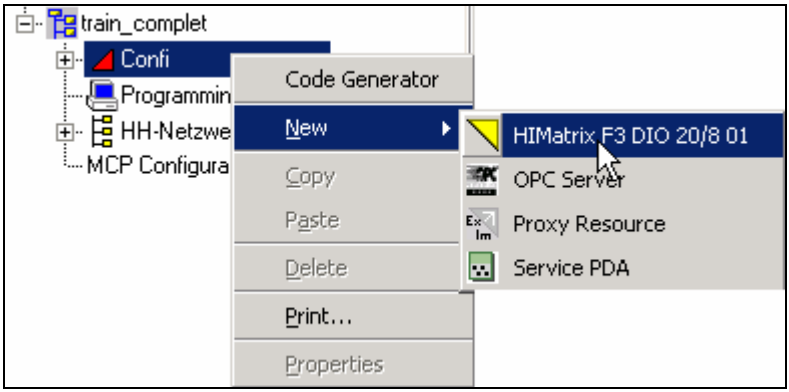
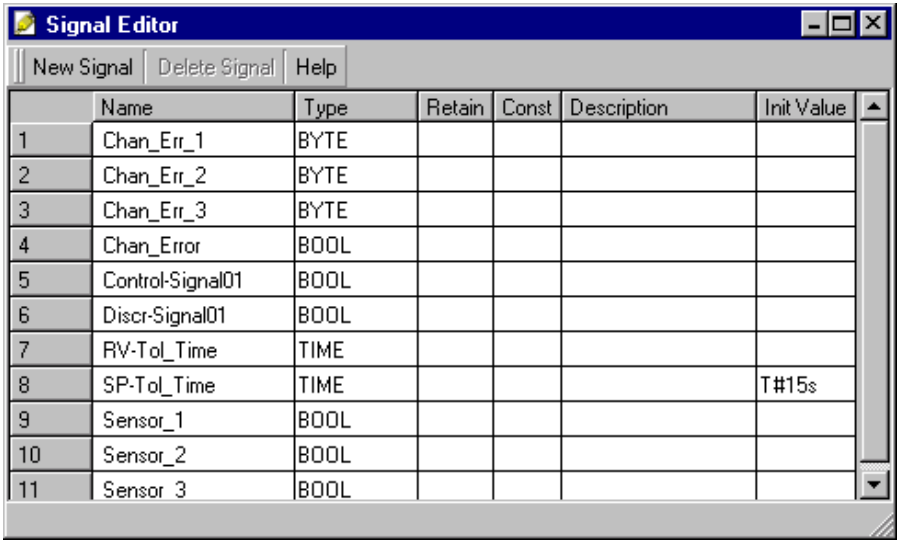


Fig. 24: Context menu

3.3.3 Signal Editor

The Signal Editor is selected from the **Signals** menu.

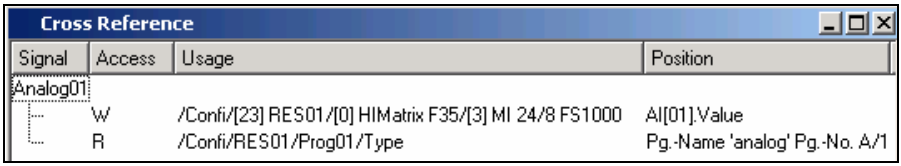


	Name	Type	Retain	Const	Description	Init Value
1	Chan_Err_1	BYTE				
2	Chan_Err_2	BYTE				
3	Chan_Err_3	BYTE				
4	Chan_Error	BOOL				
5	Control-Signal01	BOOL				
6	Discr-Signal01	BOOL				
7	RV-Tol_Time	TIME				
8	SP-Tol_Time	TIME				T#15s
9	Sensor_1	BOOL				
10	Sensor_2	BOOL				
11	Sensor_3	BOOL				

Fig. 25: Signal Editor

All variables that are to be transferred from one scope of validity (e.g. program) to another scope of validity (e.g. I/O level) must be given an assignment definition. This is done by using signals in the Signal Editor.

After the signal has been created in the Signal Editor, the signal is copied to the relevant areas with Drag & Drop. By this a cross reference is entered to the signal. You can check them by a right mouse click on the signal name and selecting cross reference.



Signal	Access	Usage	Position
Analog01			
	W	/Confi/[23] RES01/[0] HIMatrix F35/[3] MI 24/8 FS1000	AI[01].Value
	R	/Confi/RES01/Prog01/Type	Pg.-Name 'analog' Pg.-No. A/1

Fig. 26: Cross reference of signal «Analog01»

3.3.4 Online Help

The **Help, Contents** menu function shows information on all subjects relevant to the Hardware Management.

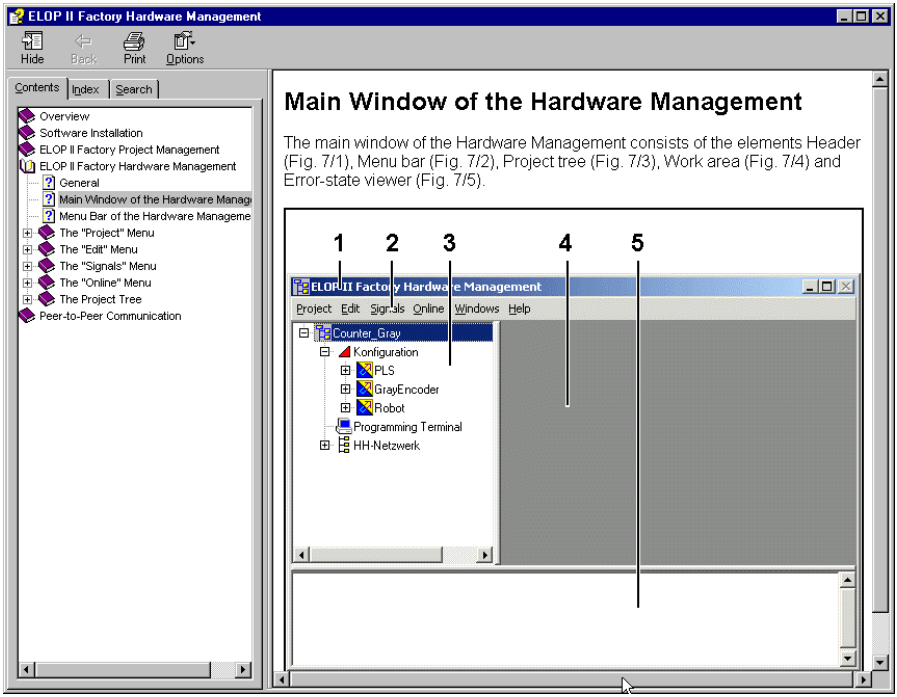


Fig. 27: Online help

4 The Objects in the Structure Window

In this chapter:

- Hierarchical structure of the objects in the structure window
- Meaning of the objects

All objects in the project (see also Fig. 15) are displayed in their hierarchical structure and managed in the structure window.

4.1 Project

The project  is the highest-order object. All other objects are created subordinate to a project. Only one project can be open at the same time in ELOP II Factory.

4.2 Library

The library  contains functions, function blocks and programs. They are also referred to as program organisation units (POU) in compliance with IEC 61131-3. ELOP II Factory contains standard libraries with predesigned functions. You can use these functions to create your own more complex functions, function blocks and programs (POU).

4.2.1 Program Type

A program type  contains all functions of an application. One program type can be assigned to multiple controllers for execution. Every controller is then executing a program instance of the program type declared in the library.

4.2.2 Function Block Type

The function block type  contains subfunctions of an application, comparable to a subroutine. The function block type can also be used to structure the program to correspond to the system layout. The function block type can save values temporarily in local variables. The output value depends on the input values and the temporarily stored values (typical example: flip-flop, timer).

The function block type can also be used to access external variables (global variables of a program).

4.2.3 Function

The function  contains basic functions of an application. In contrast to the function block type, a function cannot save states. The output value depends exclusively on the input values (typical example: AND, OR).

4.3 Configuration

The configuration  groups controllers into logical units, between which a communication connection can be set up.

4.3.1 Resource

The resource  is the term specified in IEC 61131-3 for a target system that executes the controller task, in this case the HIMatrix controller.
A resource is created within a configuration.

The above icon of a resource corresponds to the view in the Project Management. There are two variations in the Hardware Management:

1.  The opposite part to the resource created in the Project Management.
2.  Tagging of a Remote I/O, which can communicate with more than one resource. The Remote I/O is created in the Hardware Management and cannot contain logic.

Remote I/Os, which can communicate only with one resource (the so-called "parent" resource), is represented by the icon  . In the Project tree these Remote I/Os are located in the hierarchy below the parent resource in the "Remote I/O" folder. See also Fig. 23.

4.3.2 Program Instance, Type Instance

The program instance  is a reference to an existing program type in a library. The program is executed in this resource.

A type instance has a similar symbol  but it has no link to a program type in the library. The type instance has its exclusive program type in its own folder.

The instance in the Hardware Management has always a yellow symbol.

4.4 Documentation

The documentation  makes it possible to compile the objects for documentation simply by Drag & Drop. You have the option of using revision management for all included documents.

In this case the reference is the documentation of the logic (of the user program).

Note: *The hardware documentation is displayed and printed in the Hardware Management and can be selected as a function in the **Project** menu.*

5 Basic Features of ELOP II Factory

In this chapter:

- Project Management
- Creating functions
- Documentation management

This chapter covers the basic features of ELOP II Factory and helps you to understand the software tool. Chapter 7 contains practical examples.

5.1 Project Management

The Project Management is the organisation centre for working with ELOP II Factory. The project folder is shown in first position in the structure window (see also Fig. 15, left section).

You can

- open or close a project (with the menu bar or the button on the toolbar),
- archive a project (with the context menu) or
- restore a project (with the "Project“ menu).

You can make numerous presets for design purposes with the help of the template project, which can also be opened from the menu bar.

An open project is displayed in the structure window.

5.2 Function Block Libraries

A project can consist of any number of libraries with numerous function blocks. Libraries can be created in a project, a configuration, a resource or another library.
This allows you to structure the libraries to match the system.

The basic functions of IEC 61131-3 are gathered in the library "StandardLibs", which is automatically linked when a new project is created.

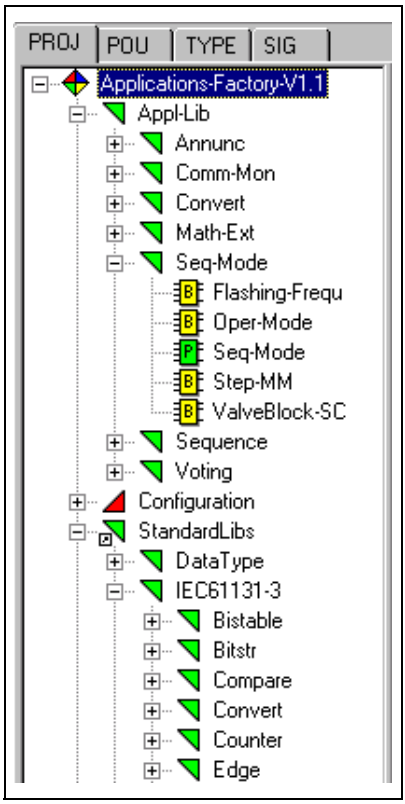


Fig. 28: Libraries in the Structure window

5.3 Function Block Diagram Editor

The Function Block Diagram editor is started automatically when a program organisation unit (function, function block or program) is opened.

The panes

- Drawing field
- Interface declaration editor
- Variable declaration editor
- Overview window

are displayed in a window within the working area of the Project Management as they were saved last in the project.

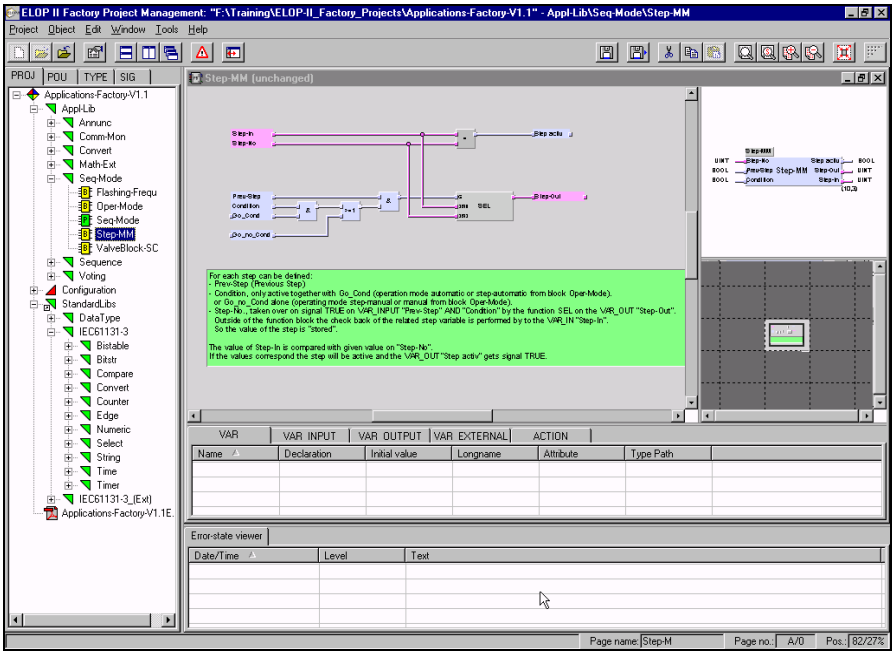


Fig. 29: Function Block Diagram editor in the Working area

Extra functions are provided for switching between the maximized working area and the general overview.

5.3.1 Maximising the Working Area

The structure window from Fig. 29 can be shown or hidden with the  button in the left side of the toolbar.

The same applies for the Error-State-Viewer and the  button. This allows the working area for the Function Block Diagram editor to be enlarged or reduced.

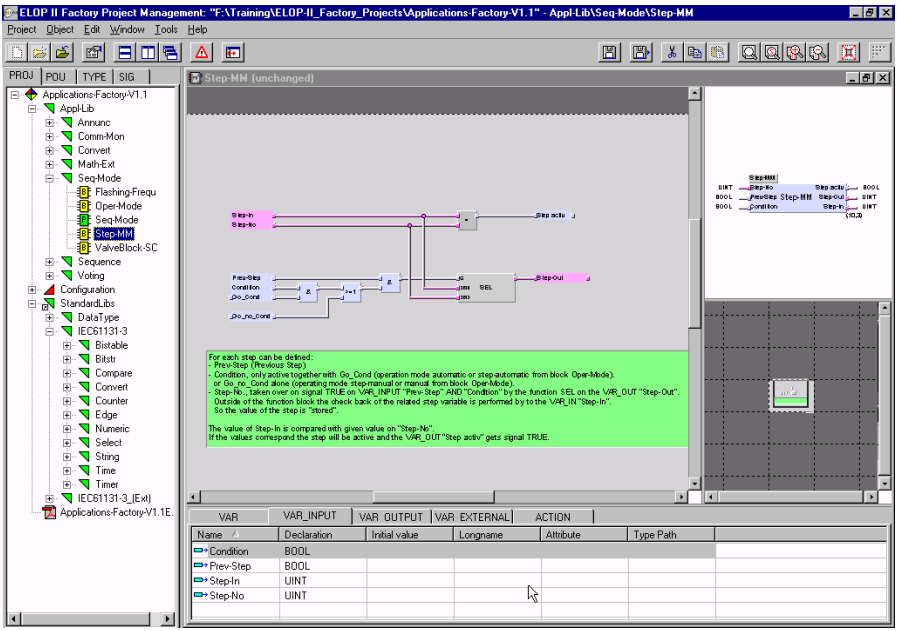


Fig. 30: FBD editor in the maximised Working area

Note: *The size of the working area can also be controlled by dragging the boundaries towards the variable declaration editor and towards the interface declaration editor.*

5.3.2 Maximising and Restoring Panes

A window is activated in the working area by clicking into the window (Fig. 30).

The active window can be maximised by clicking the  button.

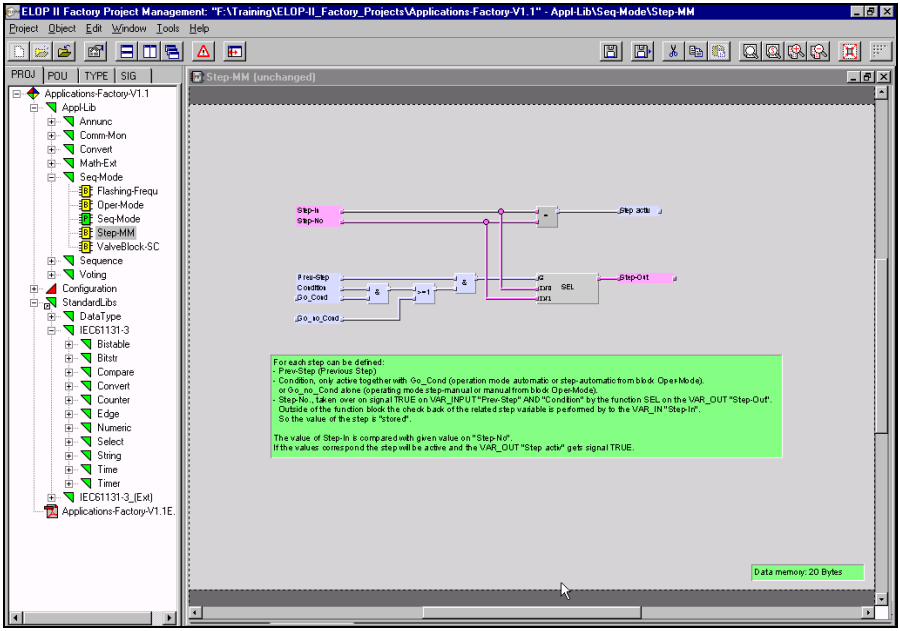


Fig. 31: «Drawing field» maximised

Click the button again to restore the areas of the Function Block Diagram editor to their original size.

5.4 Function Diagrams with centred Starting Point

The ELOP II Factory concept eliminates the need to subsequently insert individual pages later, because a plan of any size required is created. The position of a page is specified by coordinates. Columns are identified by capital letters and rows by numbers.

By default the first page has the coordinate A/0. As soon as an element is added to this page it becomes active.

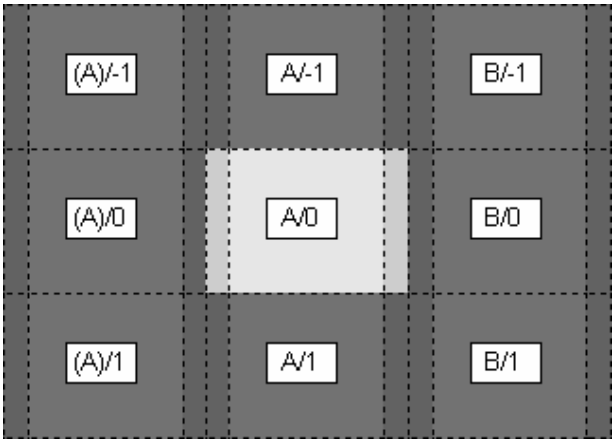


Fig. 32: Page numbering in the function diagram

Active pages are highlighted. When an element is placed on an adjacent page, this page also becomes active and is also highlighted. In this way, the function diagram can be extended in any desired direction.

Note: *If you need to insert a page between existing pages, you can move a page. To do this, select **Plug-Ins, Move page** in the context menu of the page.*
This function should only be used during the development of a project, but not with a user program that is already in use. Because pages that are "in the way" are moved, execution priorities may change.

The overview window shows the general view of the function plan. You can navigate between pages by clicking on one of the pages in the overview.

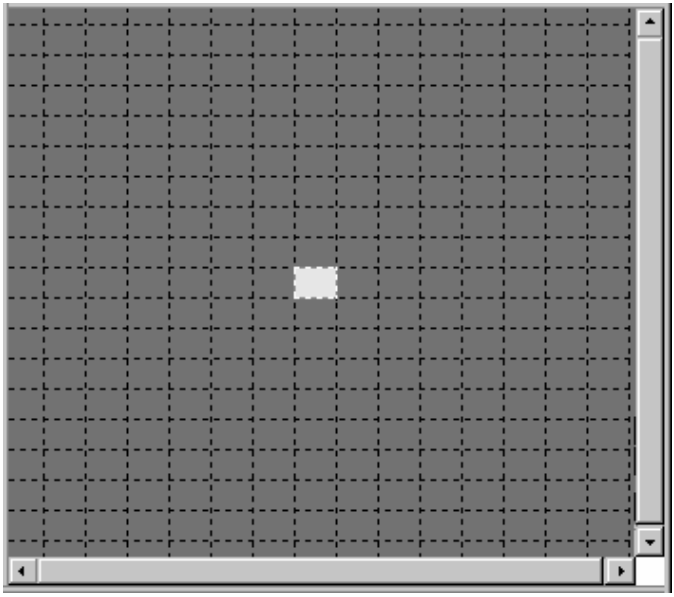


Fig. 33: Overview window

Note: *The "Page list" function is in the page context menu in the "Plug-Ins“ menu. You can also use this function to go to specific pages.*

5.4.1 Zoom

You can use the buttons  in the toolbar to enlarge or reduce the display in the drawing field and in the overview window.

5.5 Setting Drawing Field Properties

The following drawing field properties can be set:

- Specifying value field bars (recommendation: set the width of the value field bar to zero),
- Size of value fields, comment fields and connectors,
- Activate automatically placed attached comments (AC) or deactivate the automatically created AC.

5.6 Creating Logic

You can create a logic in the function diagram with these basic functions.

5.6.1 Dragging & Dropping Variables

Variables are created in the variable declaration editor. There is a distinction between

- (Local) variables,
- Input variables,
- Output variables and
- External variables, or at program level
- Global variables.

Click on a variable in the variable declaration editor and drag it to the desired position in the drawing field. A value field with the variable name is created. In the variable declaration editor an icon in front of the variable shows the type of use in the drawing field (see also chapter 7.3.1).

Note: You can also import variables from an external data source. See the Online Help for more information.

5.6.2 Dragging & Dropping Function Blocks

To use a function block select it from a library in the structure window and drag it to the desired position in the drawing field.

5.6.3 Connecting Elements

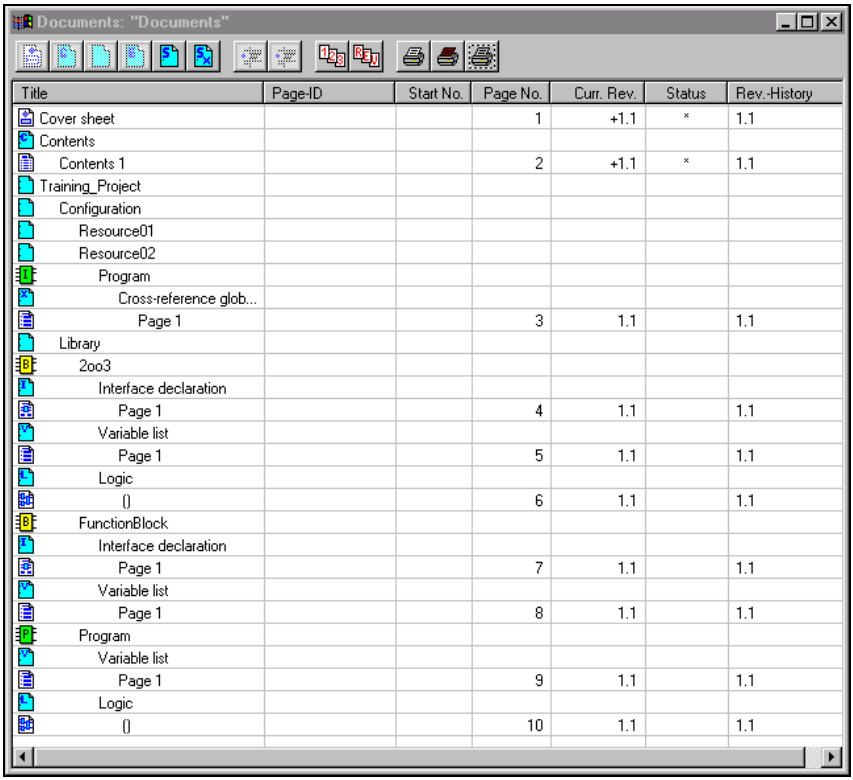
The elements placed in the drawing field can be connected by lines between their nodes.

5.7 Creating and Managing Documentation

ELOP II Factory provides document management for the software plans with a revision service in the Project Management.

This provides the option of creating different revisions of documents.

In addition to a complete printout with extensive revision service, document management also detects revisions of individual pages of the documents, allowing you to view the modified pages only (see also chapter 7.13.2).



Title	Page-ID	Start No.	Page No.	Curr. Rev.	Status	Rev.-History
Cover sheet			1	+1.1	*	1.1
Contents						
Contents 1			2	+1.1	*	1.1
Training_Project						
Configuration						
Resource01						
Resource02						
Program						
Cross-reference glob...						
Page 1			3	1.1		1.1
Library						
2oo3						
Interface declaration						
Page 1			4	1.1		1.1
Variable list						
Page 1			5	1.1		1.1
Logic						
()			6	1.1		1.1
FunctionBlock						
Interface declaration						
Page 1			7	1.1		1.1
Variable list						
Page 1			8	1.1		1.1
Program						
Variable list						
Page 1			9	1.1		1.1
Logic						
()			10	1.1		1.1

Fig. 34: Documentation management

Note: *To update changes in the logic and in the documentation, after opening click the "Update table of contents“ button*  *and then "Create revision“*  .

5.7.1 Templates for Printing Documents

DXF forms are used as print templates when printing documents. ELOP II Factory includes a standard set of forms for printing all objects.

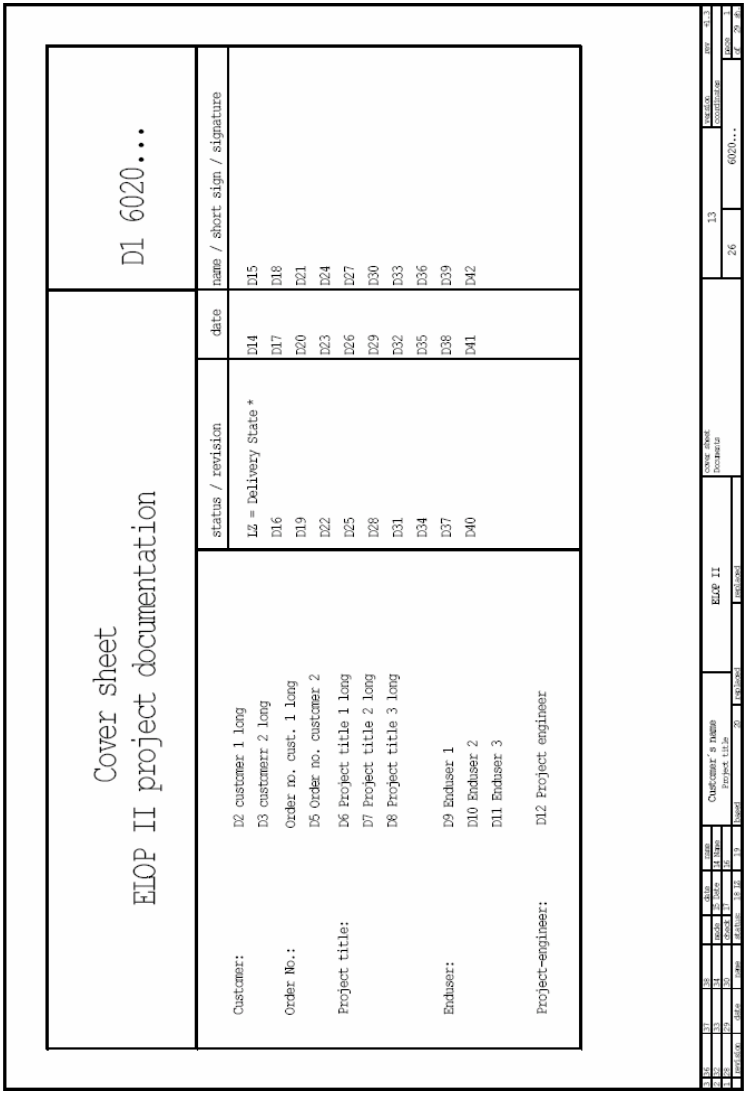


Fig. 35: DXF form for the cover sheet

You can change the default entries for the print templates and also individual fields in the DXF forms in the documentation object properties.

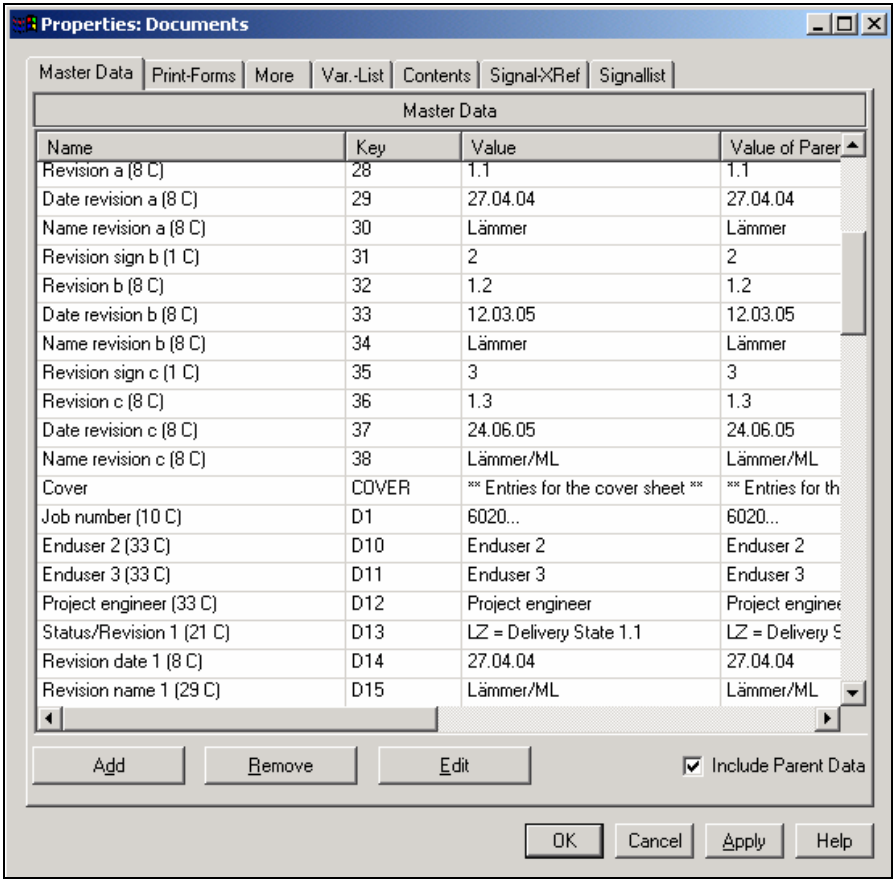


Fig. 36: Project definitions for printout

Note: *The hardware documentation can be printed out in the Hardware Management (see also chapter 7.15.2).*

5.8 Offline Simulation of Function Diagrams

You can use the Offline Simulation to check the created function diagrams on the PC for logical correctness without having to use a programmable electronic system (PES = PLC). The function plans are translated by ELOP II Factory and processed by the PC.

The Offline Simulation can only be executed on program instances within a resource.

The Offline Simulation displays an animated view of the function diagram. You can use online test fields (OLT fields) to display individual values at any position in the function plan. Additionally, the lines for Boolean values are shown in colour.

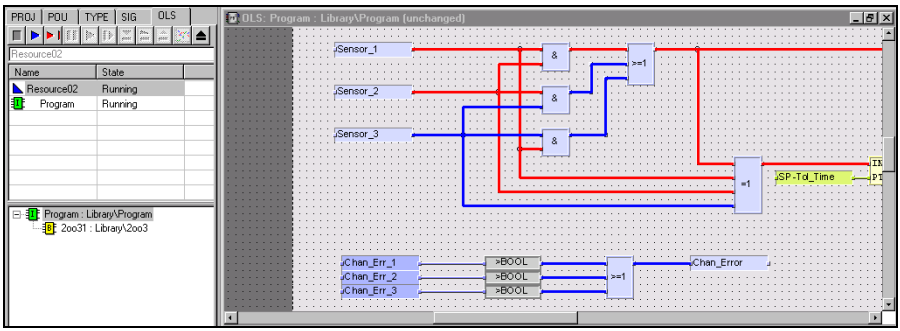


Fig. 37: Offline simulation of a function diagram

5.9 Online Test (Power Flow)

The Online Test in the Project Management is used to monitor all values of the variables and signals within the logic while the application is running.

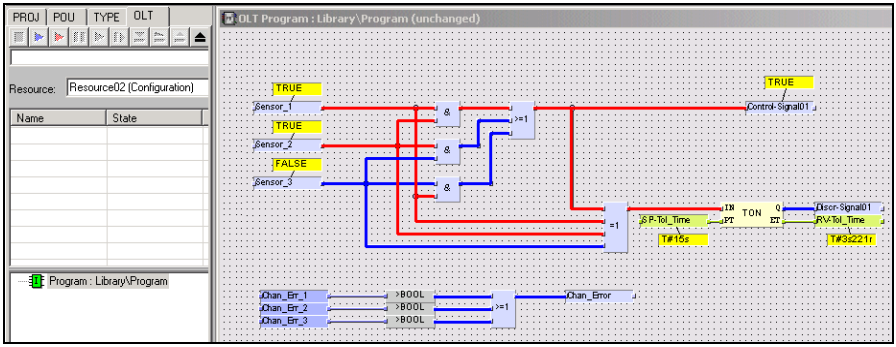


Fig. 38: Online Test

6 Working with Resource Types

In this chapter:

- Assignment of controller type and I/O modules
- Code generating
- Downloading to the controller
- Online test
- Hardware documentation

After the function diagram has been created and tested independently from the hardware, it is assigned to a specific resource type.

As defined by IEC 61131-3, a resource is a system that executes a program and serves the I/O level.

The type of system is selected from a list of all available resources (the resource types).

6.1 Creating a Resource

A resource is a device which can execute a logic program. Devices without this capability are called Remote I/Os.
An exception is the HIMatrix F3 DIO 20/8 01 which is handled like a resource although it is a Remote I/O.

- Resources (e.g. HIMatrix F30, F35, F60)
  Resources are created in the configuration of the Project Management. This resource is at first a so called “Unassigned Resource” because a special type is not assigned in this moment.
- HIMatrix F3DIO20/8 01
  This device is created in the configuration of the Hardware Management. The type is assigned simultaneously with the creation. Because this Remote I/O has the same communication capabilities as a resource, it is handled like a resource in the Hardware Management.
- Remote I/Os (e.g. HIMatrix F2, F3DIO20/8 02)
  These devices can only communicate safety-related with a parent resource. A Remote I/O is created in the Hardware Management in the “Remote I/O” folder of the parent resource.
  See also chapter 6.4.1.

Step 1: Creating a resource:

- ☐ Change to the Project Management.
- ☐ Open the context menu of the configuration in the Project Management.
- ☐ Click on **New, Resource**.

Step 2: Renaming the resource:

- ☐ In the Project Management click twice slowly on the name of the resource in the structure window.
  An entry field opens and the name can be changed.
- ☐ Or click on the resource with the right mouse button and select **Rename** in the context menu.

Step 3: Creating a HIMatrix F3 DIO 20/8 01:

- ☐ Change to the Hardware Management.
- ☐ Open the context menu of the configuration.
- ☐ Click **New** and select **HIMatrix F3 DIO 20/8**.

Step 4: Renaming the HIMatrix F3 DIO 20/8 01:

- ☐ In the Hardware Management right-click on the resource name and select **Properties** in the context menu.
- ☐ Enter a name in the "Name“ field.
- ☐ Enter a value greater than 0 in the "System ID [SRS]“ field.
- ☐ Confirm your input with **OK**.

Note: *You can also rename a Remote I/O the way described above. In this case however, keep the type of the remote I/O with the name.*
You can change all parameters except the resource type at any time.

For the creation of a Remote I/O see chapter 6.4.

6.2 Assigning a Program to a Resource

With creating a resource a type instance  is created simultaneously in the folder of the resource. This type instance can be used directly by a double-click on it and entering logic. Or you delete it and assign a program type, which is already existing in a library, to the resource. The instance which is created in that moment is called program instance  .

Assigning a program to a resource is carried out in the project management:

- ☐ Open the resource context menu.
- ☐ Select **New, Program Instance**.
- ☐ Select a program type from your library in the dialog box. The program type then appears as a program instance with the same name in the structure window in the resource.

6.3 Assigning a Resource Type

A resource type is assigned in the Hardware Management:

- ☐ Select the Hardware Management.
- ☐ Expand the configuration in the structure tree.
- ☐ Open the context menu of the resource and select **Properties**.
- ☐ Select the desired resource type from the "Type“ field.
- ☐ Enter a value greater than 0 in the "System ID [SRS]“ field and click **Apply**.

Note: The System ID (SRS = System-Rack-Slot) can be compared to a user number and must only be used once in the project.

Resource structure without assigned resource type

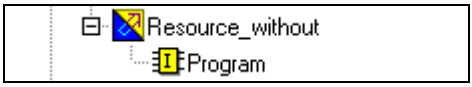


Fig. 39: Resource structure without assigned resource type

Resource structure with assigned resource type

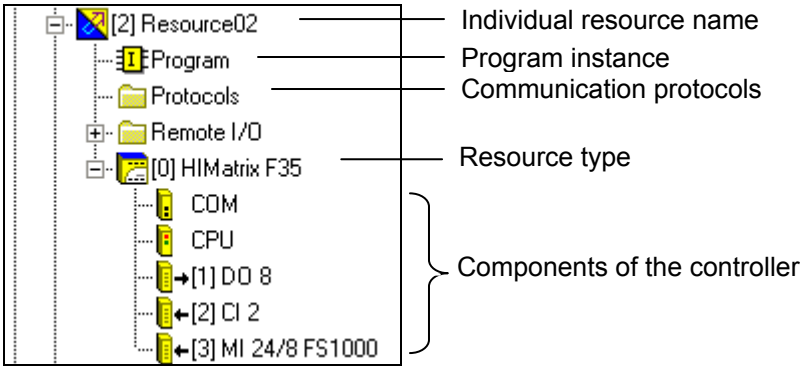


Fig. 40: Structure of an F35

6.4 I/O Module Assignment

Note: I/O modules have to be assigned in the Hardware Management.

6.4.1 Assigning Modules

Up to six I/O modules and maximum 64 Remote I/Os can be assigned to a modular system (e.g. HIMatrix F60). A compact system (e.g. F30, F35) can be extended with a maximum of 64 Remote I/Os.

To assign input/output modules proceed as follows:

- ☐ Expand the structure of the resource in the Hardware Management and open the context menu of the resource type with a right mouse click.
- ☐ Select **New** and then the desired I/O module.
- ☐ Double-click on the new module.
- ☐ Enter the correct slot number in the module properties (see also chapter 7.7.2).

To assign Remote I/Os proceed as follows:

- ☐ Expand the structure of the resource in the Hardware Management and open the context menu for the “Remote I/O” folder.
- ☐ Select **New** and then the desired Remote I/O.
- ☐ Open the context menu of the Remote I/O and select **Properties**.
- ☐ Set the “Rack ID” to a value greater than zero.

6.4.2 Assigning Signals to the I/O Channels

Signals that have been defined in the Signal Editor can be assigned to the hardware channels (see also chapter 7.8).

- ☐ Open the Signal Editor in the **Signals** menu.
- ☐ Open the context menu of a module and select **Connect Signals**.
- ☐ Position the two windows side by side.
- ☐ Drag signals from the Signal Editor and drop them on the desired channels.

See also chapters 7.8 and 7.8.3.

6.4.3 Assigning System Signals

System signals are signals that contain information on the status of the CPU or the I/O modules. They are also used for configuring I/O channels.

The procedure is identical with that for assigning I/O channels as described in chapter 6.4.2.

For the meaning of the system signals please see the controller manuals and system manuals (see also chapter 7.8.3).

6.5 Code Generator

The code generator translates the graphical inputs of the functions in the drawing field into code that can be executed in the Programmable Electronic System (PES) and generates a unique code version.

The code generator is started from the context menu of a resource. Depending on the resource type this is done in the Project Management or Hardware Management.

The result of the code generator is output in the Error-State-Viewer.

6.6 Control Panel

In the Control Panel a program is loaded into the controller, the controller is started, stopped, the communication settings can be changed etc.

Note: *Before the code generated by the code generator can be loaded into the controller, all IP addresses of the controllers must be correctly configured.*

- ☐ To start the Control Panel for a resource, right-click on the resource in the structure tree and select **Online, Control Panel** in the context menu (see also chapter 7.12).

6.7 Hardware Documentation

The documentation for the hardware layout and the hardware configuration is printed in the Hardware Management.
All documentation of all resources can be printed, or only the section covering the previously selected object.

7 Training Project

7.1 Creating a Project

Step 1: Start ELOP II Factory:

- ☐ In the Windows Start Menu click **Programs, ELOP II Factory, ELOP II Factory**.
- ☐ Or double-click the **ELOP II Factory** icon on your desktop.

Fig. 41: Starting ELOP II Factory

Step 2: Create a new project:

- ☐ Select **New** in the **Project** menu.

Fig. 42: Creating a new project

- ☐ Or click on the icon in the toolbar.

Step 3: Enter the project path and name:

☐ In the Structure tree select the directory in which you want to create the new project (Fig. 43, left). Enter a name for the new project in the "Object name“ field and click **OK**.

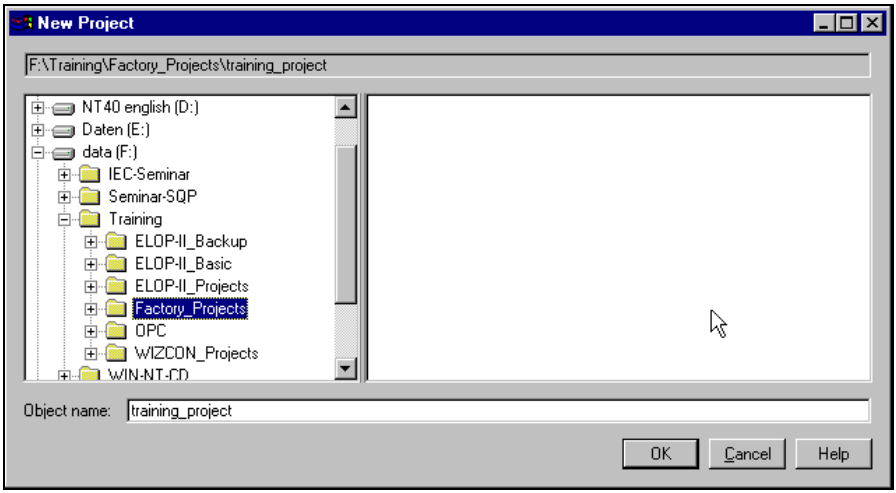


Fig. 43: Creating a new project

The new project is shown in the structure window of the Project Management. The project contains the standard library and the CFB-Library (Certified Function Blocks). In case CFB-Blocks are installed, they are listed here. With creating a project the configuration and one resource with a type instance is automatically done. The Hardware Management is opened at the same time.

Fig. 44: Project structure after creation

7.2 Creating a Library and generating Program Organisation Units (POU)

7.2.1 Creating a library

Step 1: Create a library as the first structure element. Later it will contain the POUs:

- ☐ Click on the project in the structure window.
- ☐ Open the context menu by clicking the right mouse button.
- ☐ Select **New, Library**.

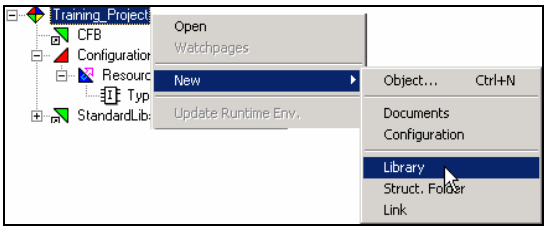


Fig. 45: Creating a library

A new library with the name "NewLib" is created in the project path.

Step 2: Change the name of the new library from "NewLib" to "Library“:

Either:

- ☐ Enter the new name immediately after creation of the library. The default name is already highlighted for renaming. Terminate the input with the "ENTER" key.

Or:

- ☐ Click twice slowly on "NewLib". The name will be highlighted for renaming after the second click. Enter the new name and terminate the input with the "ENTER" key.

Or:

- ☐ Open the context menu by clicking the right mouse button on "NewLib".
- ☐ Select **Rename** and change the name to "Library“.

7.2.2 Creating Program Organisation Units (POU)

Step 3: Create your own program organisation units (POU) in the "Library" library:

- ☐ Click on "Library" and open the context menu.
- ☐ Select **New, Function Block Type**.

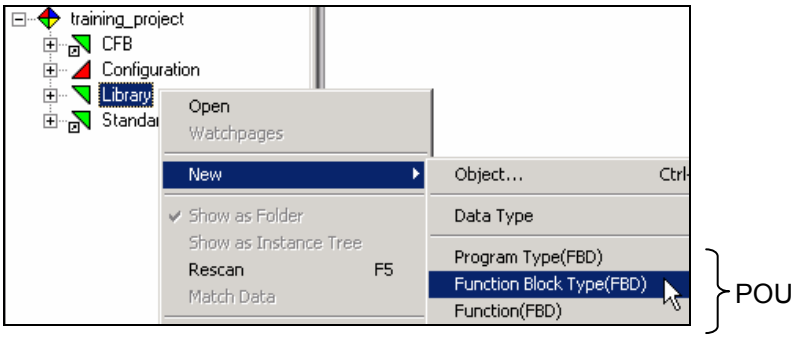


Fig. 46: Creating function-block type

- ☐ Click on the "**+**" in front of "Library".
  A new function block is added to the structure.
- ☐ Rename the function block type as described in step 2.
- ☐ Repeat step 3 and create a **Program Type** in the same way if you like to use a programm instance instead of type instance which is allready created in the resource. For the difference see chapters 4.2.1 and 4.3.2.
- ☐ Rename the program type as described in step 2.

This is what the project structure looks like after these steps:

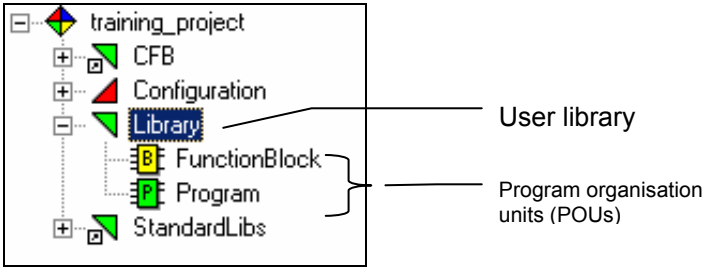


Fig. 47: Project structure after this exercise

7.3 Editing Function Blocks

Step 1: Open the function block:

☐ Open the function block (see Fig. 47) by double-clicking on the function block icon .
The FBD editor opens and displays the various panes in which definitions can be made.

7.3.1 Variable Declaration

Step 2: Select the variable type:

☐ Click on the tab with the desired variable type.
Select "VAR_INPUT" as shown in Fig. 48.

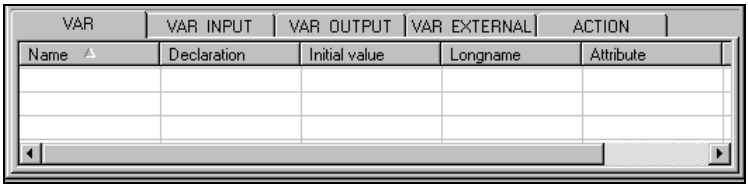


Fig. 48: Variable declaration editor

Note: *VAR_INPUT and VAR_OUTPUT are used as input and output variables from/to the next higher POU.*
The variable type VAR_EXTERNAL can only be used within a function block, but not in a function. A VAR_EXTERNAL is recognised everywhere a variable of the VAR_EXTERNAL or VAR_GLOBAL type is defined with the same spelling.
If a variable of the VAR_EXTERNAL type is to be linked with input or output channels, or if the value of the variables is to be displayed in the Force Editor, the variable must be created in the Signal Editor and dragged into the variable declaration of the function block (see also chapter 7.8).

Step 3: Input the variable data:

- ☐ Open the variable declaration dialog by double-clicking in a free area in the variable declaration editor.
- ☐ Overwrite the default “I1“ in the "Name" field with "Variable1".
- ☐ Select the data type in the "Declaration" field, e.g. BOOL.
- ☐ Specify the "Position" of the variables on the function block. In the example in Fig. 50 "Variable1" is displayed in the second position from top left.

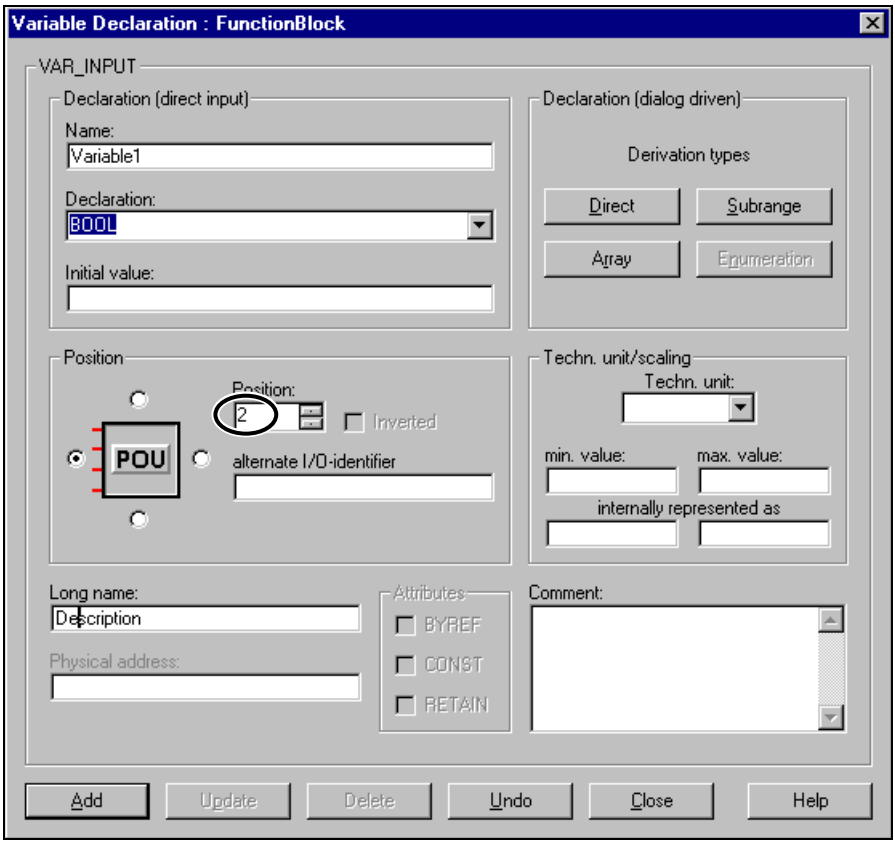


Fig. 49: Declaration of a variable within a function block

☐ Enter a descriptive name for "Variable1" in the "Long name“ field if required. The long name is an additional description of the variable.

☐ Click on **Add** to add the variable to the variable list.

The dialog window remains open and a new variable with the same name and the next number in the sequence is ready. The new variable has the same data type and is placed in the next position. The "Long name“ remains unchanged.

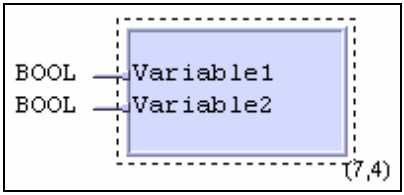


Fig. 50: Interface declaration

Note: VAR_INPUT and VAR_OUTPUT variable appear in the specified position in the interface declaration.

7.3.2 Specifying the Interface Declaration (Graphic View)

Step 1: Specify the properties of the function block in the interface declaration window:

☐ Right-click on the function block and select **Properties** in the context menu.

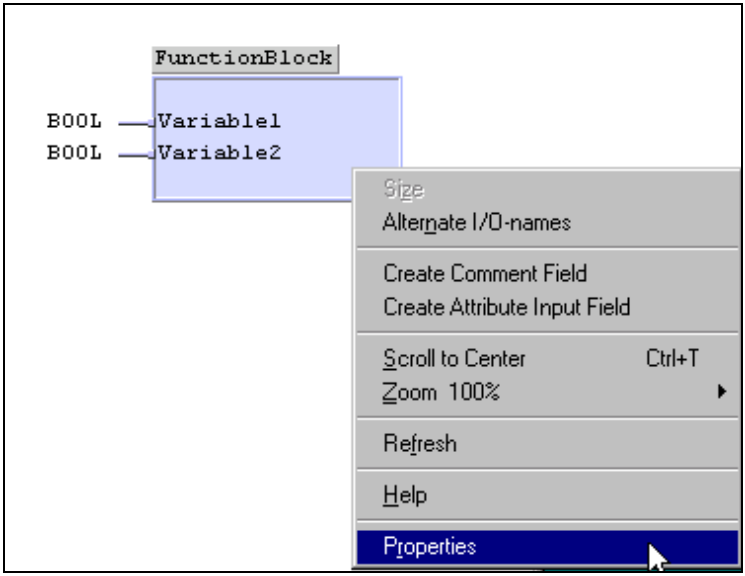


Fig. 51: Context menu of the function block (POU)

Step 2: Enter a POU identifier:

☐ Enter a name for the function block in the "POU text“ field. The POU text should be identical to the name of the function block in the library. If necessary, the font can also be changed by clicking the **Edit font...** button.

The input name appears in the centre of the function block (see Fig. 54).

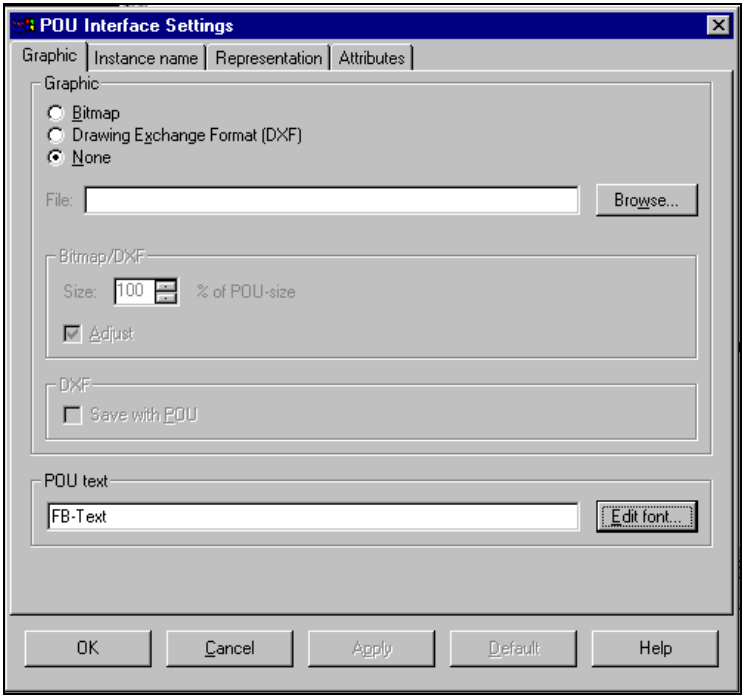


Fig. 52: Entering the POU text

Step 3: Specify an instance name:

- ☐ Select the **Instance name** tab.
- ☐ Enter a name in the "Instance name“ field and select the **Display** field.
- ☐ Set the **Font** and the **Alignment** if necessary. By default the instance name is shown at the top left of the block (see Fig. 54).

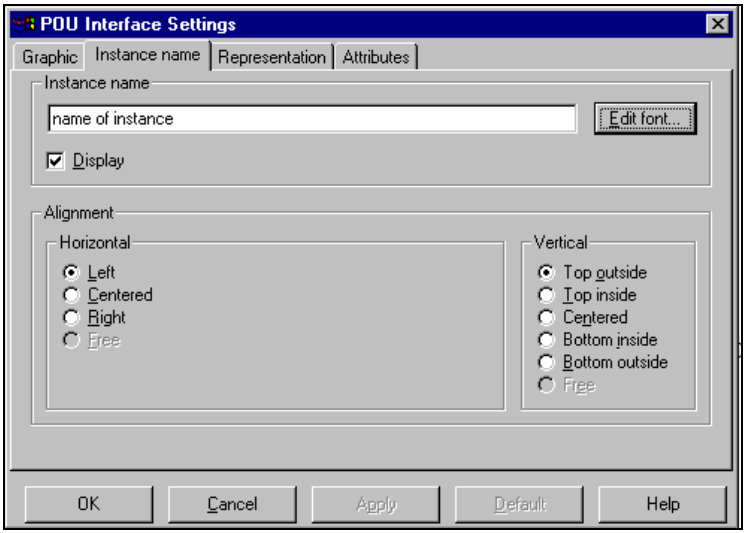


Fig. 53: Entering and formatting the instance name

Note: *The POU text describes the logic in the function block, e.g. 2 out of 3. The text remains identical for all instances of the function block.*
The instance name entered is a default, but can be changed and should define the function of this POU in the project, e.g. «2 out of 3 Temperature Monitoring Point 15».
If the name is not changed, a count index is appended to ensure that the name remains unique.

Step 4: Set the function block size:

☐ Place the mouse pointer on the bottom right corner of the POU. When a small black double-ended arrow appears, press and hold the left mouse button and change the size of the function block as desired.

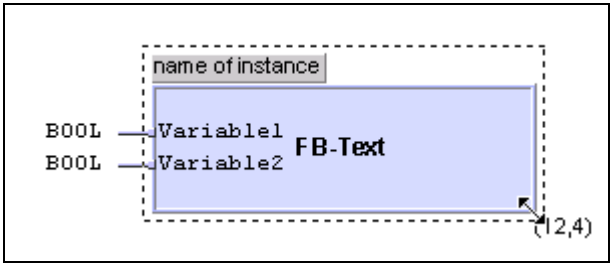


Fig. 54: Setting the POU size

7.3.3 Logic Input in the Drawing Field of the Function Block

Step 1: Drag & Drop logic elements from the libraries to the drawing field:

- ☐ Open the library **StandardLibs** in the structure window by clicking on the "**+**“ icon.
- ☐ Open the library **IEC61131-3** from the **StandardLibs** and then **Bitstr**.
- ☐ Click on **AND**, press and hold the left mouse button and drag the function block from the structure window into the drawing field.
  A preview of the object is displayed while dragging.
- ☐ After releasing the mouse button the block is placed at the mouse pointer position.

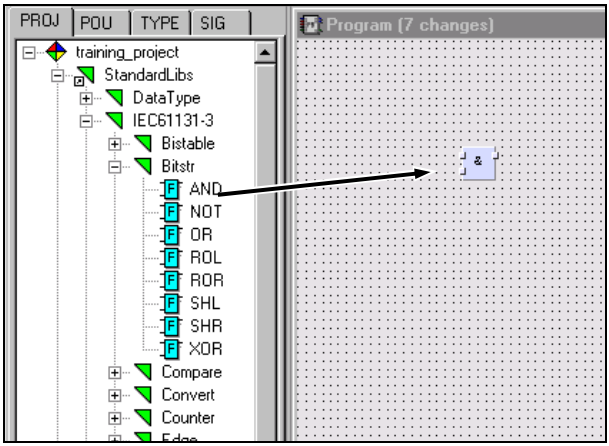


Fig. 55: Drag & Drop for function blocks

Note: *The procedure for inserting functions blocks from libraries with Drag & Drop is also valid for user libraries.*

Step 2: Complete the page data:

Because placing the **AND** function block is the first change to the contents of this page (see also chapter 5.4), the "Edit Page Data“ dialog window opens automatically.

☐ Enter a descriptive name for the page in the "Long name“ field.

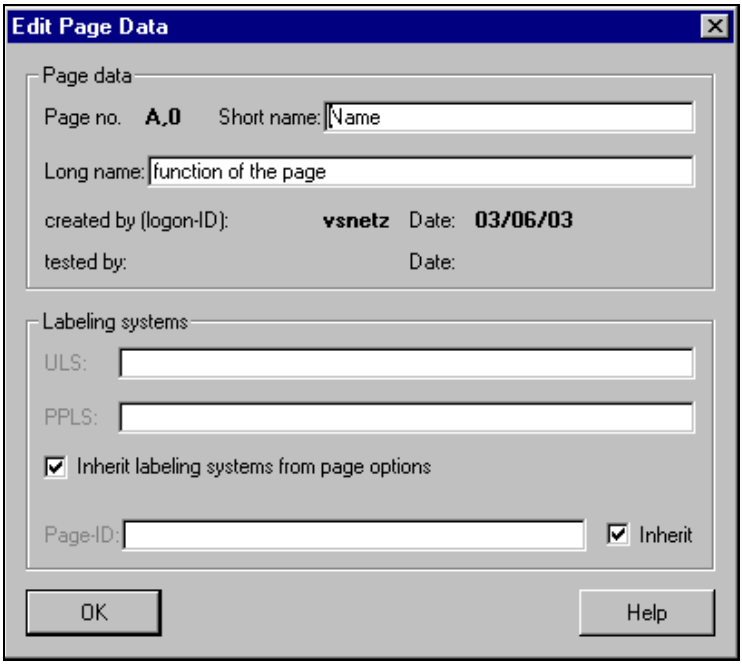


Fig. 56: Editing page data

Note: If required, disable automatic numbering of the "Short name“ under "Page data“ in the properties of the drawing field and assign your own short name.

Step 3: Extend the logic with additional function blocks:

- ☐ Copy additional function blocks from the libraries into the drawing field as described in **Step 1**.
- ☐ Duplicate identical function blocks by pressing and holding the **CTRL** key and dragging an existing function block to a different position in the drawing field with the mouse. Release the mouse button first or the block will only be moved.

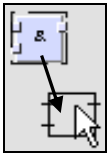


Fig. 57: Copying a function block

Note: *If function blocks are placed on top of each other during inserting or copying, the action is aborted with an acoustic warning. If the mouse pointer is at a prohibited position a "prohibitory sign" icon is displayed.*

Step 4: Toggle the grid on and zoom in:

- ☐ Toggle the grid.

Fig. 58: Button for the grid

- ☐ Zoom in to the section that you want to edit.

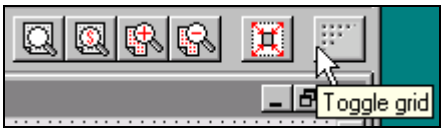


Fig. 59: Buttons for zoom and grid

Note: *The buttons on the right side refer to the drawing field of the open function block.*

Step 5: Add variables to your logic:

☐ Click on a variable name in the variable list, press and hold the left mouse button and drag the variable into the drawing field. A preview of the value field is displayed while dragging.

When the mouse button is released the variable is positioned and the variable name appears in the value field.

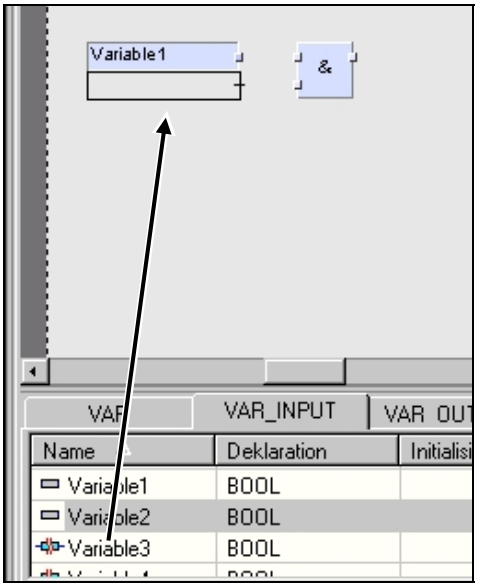


Fig. 60: Drag & Drop for variables

Note: *If an attached comment should be appended (not appended) to the variable, the automatic append in the drawing field properties can be enabled (disabled) under* ***Comment, Attached Comment, AC value field drawing field****.*

Step 6: Draw connecting lines between the variables (value fields) and function blocks:

- ☐ Place the mouse pointer on the connection point of a variable (= output).
- ☐ Press and hold the left mouse button and draw a line to the right.
- ☐ Draw the line to the input of another function block and release the mouse button.
  The result is a straight connecting line between the two connecting points.

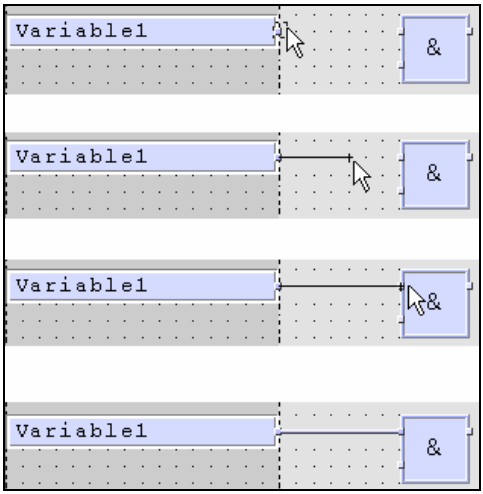


Fig. 61: Drawing a line

Note: *The colour of the line depends on the data type (BOOL, integer, real etc.).*
If the line is drawn too far into the function block or if the data type of the input does not match the data type of variable, a "prohibitory sign" icon is displayed.

7.3.4 Completing the Function Block

After completion of the logic and the interface declaration, the function block must be saved by clicking the diskette icon  .

Note: *Every new or modified function block should be tested offline to ensure early detection of incorrect behaviour (see also chapter 7.6.2).*

7.4 Editing a Program or Type Instance

☐ Open the program/type instance (see chapter 7.2.2 for creating) by double-clicking the  or  icon.
The FBD editor opens and displays various panes in which definitions can be made.

7.4.1 Variable Declaration

The procedure for variable declaration is identical to that for the declaration of function blocks (see chapter 7.3.1).

Note: *VAR, VAR_GLOBAL and VAR_EXTERNAL are used for the program type/type instance.*
Variables linked to input and output channels, communication variables or system variables must first be created in the Hardware Management with the Signal Editor and copied to the program variable declaration with Drag & Drop. They are automatically saved in the VAR_EXTERNAL register (see also chapter 7.8).

7.4.2 Specifying the Interface Declaration

This function does not apply to POUs of the program type. Because the program itself cannot be used in other POUs, there is no interface declaration.

7.4.3 Logic Input in the Drawing Field

The logic input procedure corresponds to the input of function blocks (see chapter 7.3.3).

7.5 Creating a Resource

With creating a project a configuration is automatically done including a resource with a type instance. If you need more than one resource they have to be added.

If you have an older version of ELOP II Factory the configuration was not automatically done and you have to do this before you can create a resource.
For this you open the context menu of the project by right mouse click on the project name an select **New, Configuration**.
Eventually you enter a new name for the configuration.

Step 1: Create a new resource inside the configuration:

☐ Right-click on the configuration in the structure window.

☐ From the context menu select **New, Resource**.

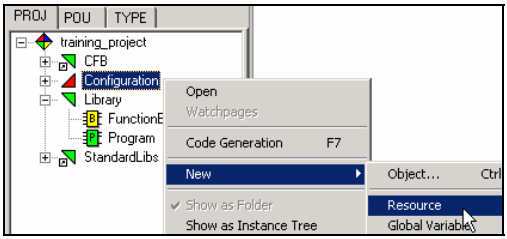


Fig. 62: Creating a resource

Note: *The resource is created as a "neutral resource" in the Project Management.*
The resource type is assigned later in the Hardware Management.

Step 2: Assign a program instance to the resource:

Normally a type instance is already assigned to the resource, which has its own program type. Usually this structure is used and you can enter the logic after double-clicking on the type instance.

If a program type should be assigned to the resource you have to delete at first the type instance and then create the program instance.

☐ Right-click on the resource in the structure window.

☐ From the context menu select **New, Program instance...**

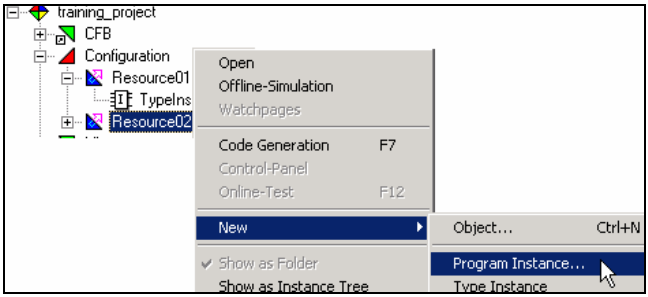


Fig. 63: Creating a new program instance

Step 3: Select the program that is to be processed in the resource:

☐ Select your program from the project library.
The program block is imported as a program instance.

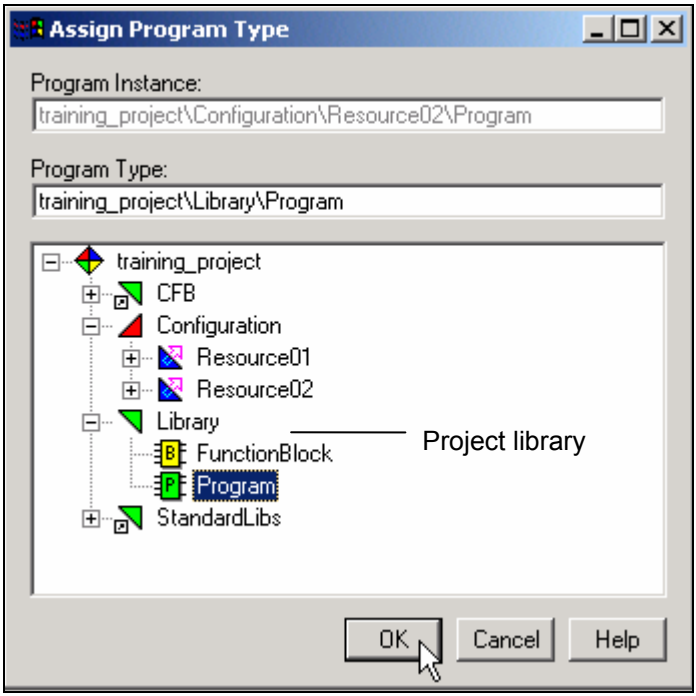


Fig. 64: Assigning a program type

This is what the project structure looks like after these steps:

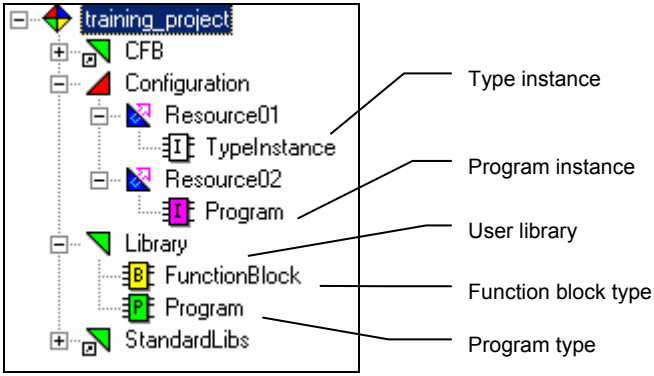


Fig. 65: Project structure

Note: *The program instance in the resource is only a reference to the program type in the user library which is run on the controller. The program type in the user library contains the "software", i.e. the logic and the variables.*

7.6 Offline Simulation

7.6.1 Offline Simulation of a Program

The behaviour of a function block or of a program is tested in the Offline Simulation without using the controller (hardware). This allows programming errors to be detected and corrected before commissioning.

A configuration must have been created in the Project Management. The configuration contains the resource in which the program that is to be tested is instanced (see also chapter 7.5).

Step 1: Invoke the Offline Simulation:

- ☐ Open the context menu of the resource by clicking the right mouse button.
- ☐ Select **Offline-Simulation**.

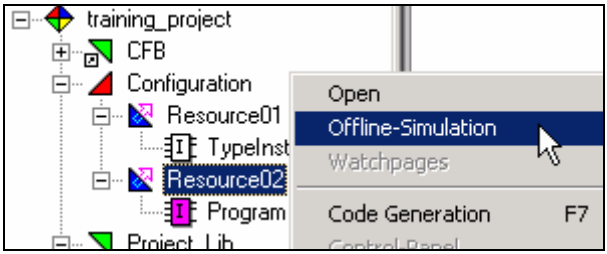


Fig. 66: Activating the Offline simulation

Note: *An additional tab "OLS“ (Offline Simulation) opens in the structure window.*

Step 2: Start the Offline Simulation:

Normally the Offline-Simulation is immediately in “RUN”. In case of an older version you can start it with the button Cold boot.

☐ Click on the button **Cold boot**  .

After starting the “State” changes from «Stopped» to «Running».

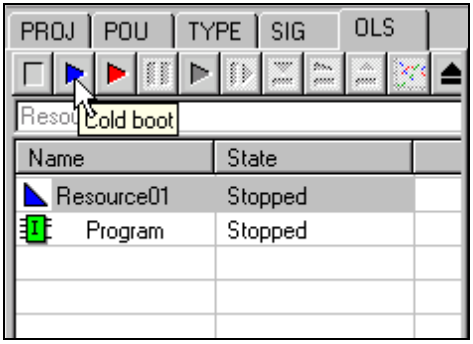


Fig. 67: Starting the Offline simulation

Step 3: Open the program in the Offline Simulation:

☐ Open the program by double-clicking on the program in the structure window (lower left pane).

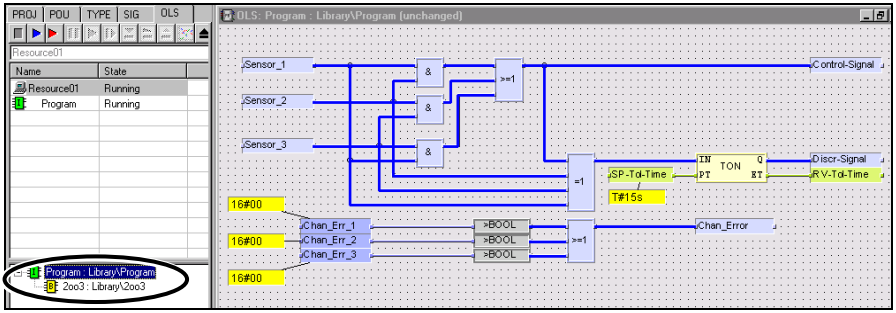


Fig. 68: Offline simulation of the program

Note: *You can open multiple function blocks simultaneously and trace the signals between the function blocks. The function blocks can also be opened directly in the logic in the drawing field.*

Step 4: Change the state of the signals and test the logic:

a) Change signal state with an Online Test field (OLT field):

- ☐ Click on a value field and hold down the mouse button.
- ☐ Drag the mouse from the value field and release the mouse button at a free position on the screen. A preview of the OLT field is displayed.
- ☐ Position the OLT field by clicking with the mouse.
- ☐ Change the signal state by double-clicking in the OLT field.

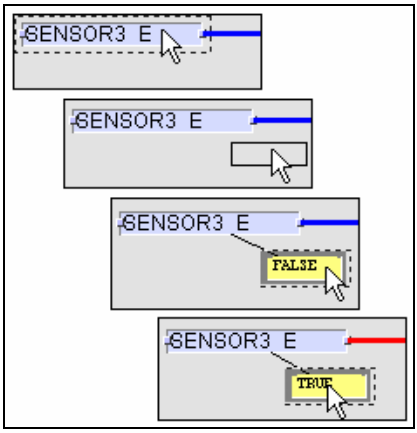


Fig. 69: Creating an OLT field

Note: *After adding OLT fields the prompt "Save changes?" appears when closing the function block.*
*Clicking **Yes** saves the OLT fields with the project. Clicking **No** deletes the created OLT fields.*
OLT fields can be created when creating the program via the context menu of the element.

b) Change signal state directly in the value field:

- ☐ Place the mouse pointer on the value field whose value you want to change.
- ☐ Press and hold the "ALT“ key. The variable state is displayed.
- ☐ Change the variable state by clicking with the mouse on the value field.
- ☐ Release the "ALT“ key. The variable name is shown again.

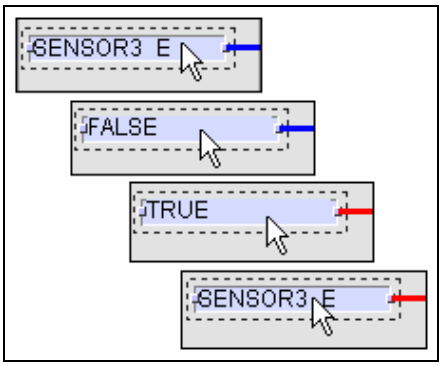


Fig. 70: Switching value field display with the ALT key

Note: *You can only change values which are not written by the logic.*

Step 5: Close the Offline Simulation:

☐ Click on **Close OLS** in the toolbar.

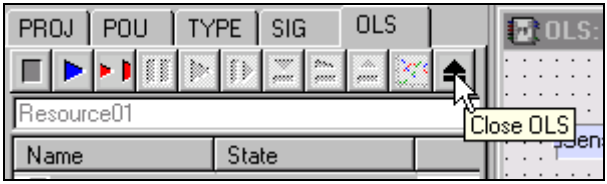


Fig. 71: Closing the Offline simulation

7.6.2 Offline Simulation of a user-defined Function Block

Step 1: Drag the function block that has to be tested into the program. Save the program:

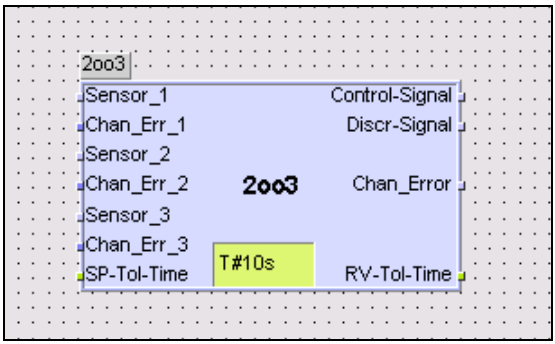


Fig. 72: Block without wiring

Note : If necessary, wire the required feedbacks (e.g. for group signals) to allow the block to be tested with the feedback function.

Step 2: Start the Offline Simulation as described in chapter 7.6.1:

☐ After opening the program, open the function block by double-clicking on the function block.

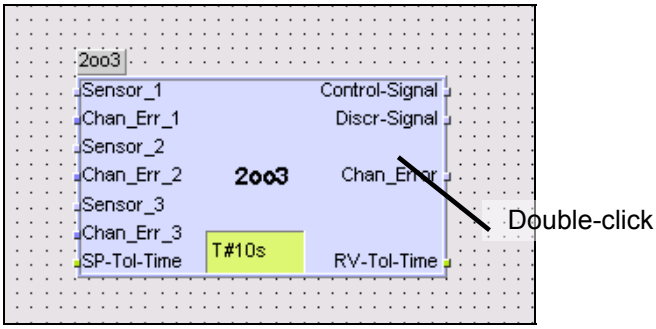


Fig. 73: Opening the function block by double-click

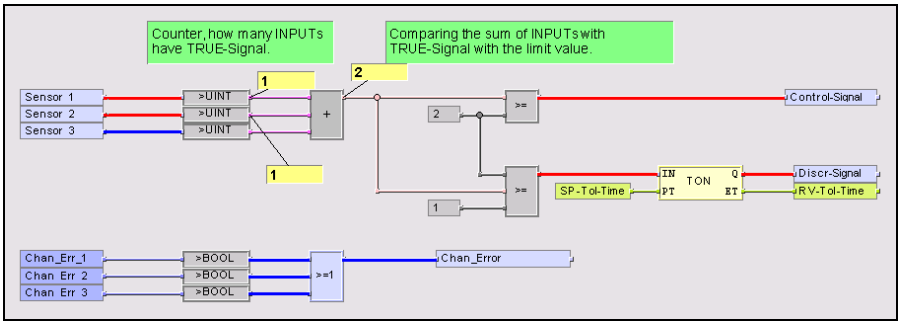


Fig. 74: Logic in the function block

☐ Test the behaviour of the function block by changing the VAR_INPUT values (possibly VAR_EXTERNAL) by clicking with the mouse while holding down the ALT key.

☐ Close the Offline Simulation again (see also chapter 7.6.1, Step 5).

7.7 Controller Hardware Assignment

All hardware-based settings must be made in the Hardware Management. The Hardware Management is displayed in a separate window, which is opened when a project is created or opened.

7.7.1 Assigning a Resource Type

Step 1: Open the properties of a resource:

- ☐ Right-click on the resource in the Hardware Management.
- ☐ In the context menu select **Properties**.

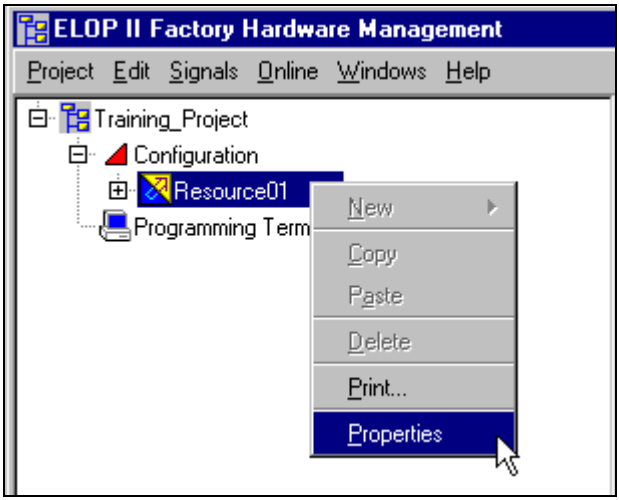


Fig. 75: Opening the resource properties

Step 2: Select a resource type:

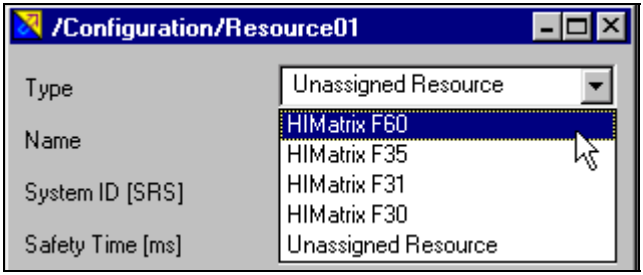


Fig. 76: Selecting a resource type

- ☐ Select a resource type in the "Type" dropdown menu.
- ☐ Enter a System ID in the "System ID [SRS]“ field and click **Apply**.

Note: *The System ID (SRS = system-rack-slot) is a user number and must only be used once in the project.*
Values from 1 to 65535 are possible.

Step 3: Edit the resource properties:

Note: *Only after resource "Type" and "System ID" have been set the other parameters are unlocked and can be modified.*

☐ Change the other parameters of the resource as required.

☐ Complete your inputs by pressing **OK**.

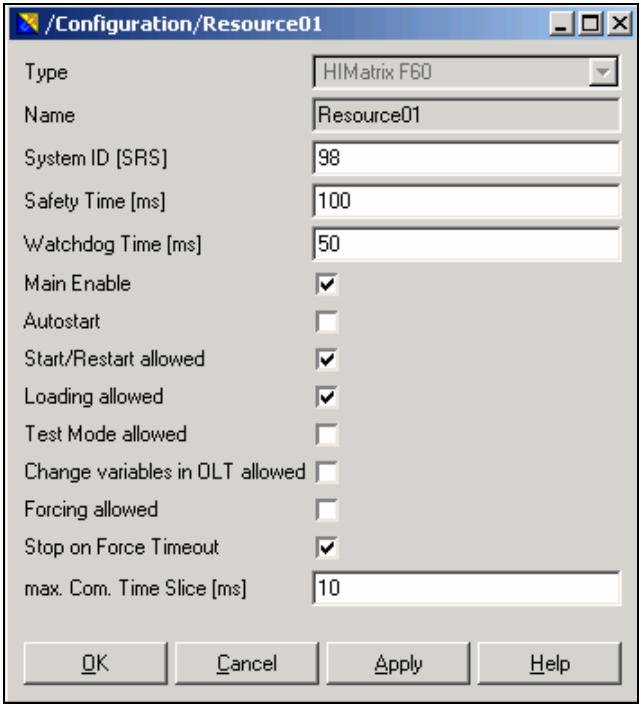


Fig. 77: System ID

Note: *The exact description of all properties available from the Online Help by clicking on the button Help in this window.*

Note: *With the property „Autostart" the user program will be started automatically direct after connecting with power supply and finishing the initialisation. If it will be started with cold or warm start depends on the properties of the program in the Hardware Management (see next chapter).*

7.7.2 Start-Settings of the Program

In the settings of the resource was decided if in principle the program should be started automatically or not. Here you decide in which way.

- ☐ Open the context menu of the program by right mouse click.
- ☐ Open the properties and select the wanted start version.

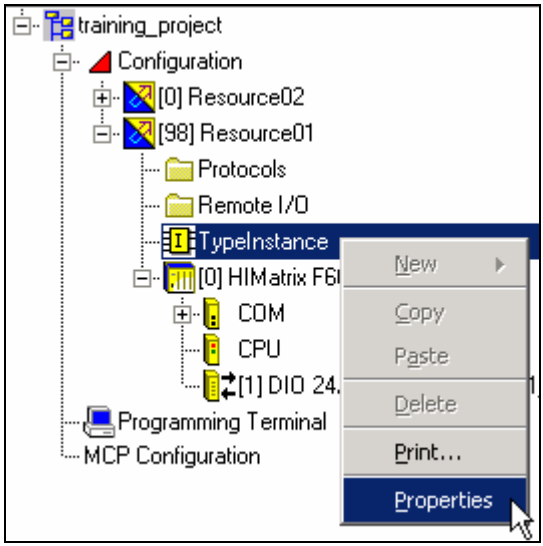


Fig. 78: Open the properties

The following ways of Autostart are possible:

Warmstart:
Signals with the attribute „Retain“ keep their buffered value.

Coldstart:
All Signals will be set to initial value.

Off:
No automatic start. The program has to be started from the PC.

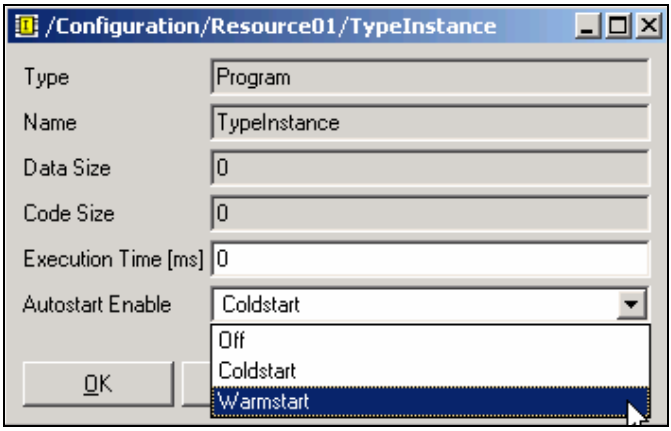


Abb. 79: Ways of Autostart

7.7.3 Settings in the Properties of the COM-Module

Communication between the Programming Terminal and the controllers is carried out over Ethernet. Communication over Ethernet uses the UDP/IP protocol. Therefore the user assigns an IP address to each controller in the network (in the **Properties** of the COM).

IP addresses are logical addresses and do not have fixed assignments to communication interfaces of controllers. Only the MAC address - as the physical address - is permanently assigned to the communication interface. The MAC address is programmed firmly with the production of the controller.

An IP address is a four-byte size dual number. Every byte is shown as a decimal number.

An IP address consists of the net ID, subnet ID and the node ID (node = participant, also host ID). The specification which part of the IP address contains the net ID plus subnet ID is defined in the Subnet Mask. See the example given below.

IP Address	192	168	0	25
	11000000	10101000	00000000	00011001
Subnet Mask	255	255	252	0
	11111111	11111111	11111100	00000000

- All bits of the IP address that are masked with "1" in the Subnet Mask belong to the network ID plus subnet ID.
- All bits of the IP address that are masked with "0" in the Subnet Mask belong to the node ID.

Mathematically the above example contains 2^{10} - 1 = 1023 possible participants. The values 0 and 255 are not allowed in the last byte.

Important: The net ID plus subnet ID within a configuration must be identical for all participants, unless gateways and routers are used.

If the Programming Terminal and the controllers are in their own, closed network, the network parameters can be freely defined.

Note: If the Programming Terminal and the controllers are part of a network that is also used by others, contact your network administrator for assignment of IP addresses.

Step 1: Define the properties of the COM module:

☐ Expand the resource or Remote I/O in the structure tree so that all modules are visible.

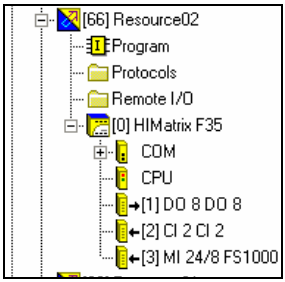


Fig. 80: Structure of a resource

☐ Select **Properties** in the context menu of the COM module.

☐ Set the "IP Address“.
Example: 192.168.0.60.

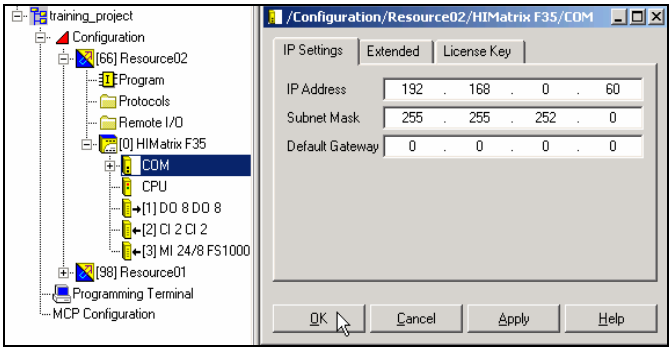


Fig. 81: Properties of the COM module

☐ Do not alter the default setting of the Subnet Mask in a closed network. Otherwise, configure the Subnet Mask as directed by the network administrator.

☐ If a standard gateway is not used, leave the address at "0.0.0.0“.

Note: The settings for the communication parameters have to be done for all resources and Remote I/Os.

7.7.4 Adding I/O Modules to a Modular Resource

Step 1: Select input/output modules:

- ☐ Click on the "**+**“ icon in front of the resource to open the structure of the resource.
- ☐ Right-click on the resource type.
- ☐ From the context menu select **New** and then choose a module.

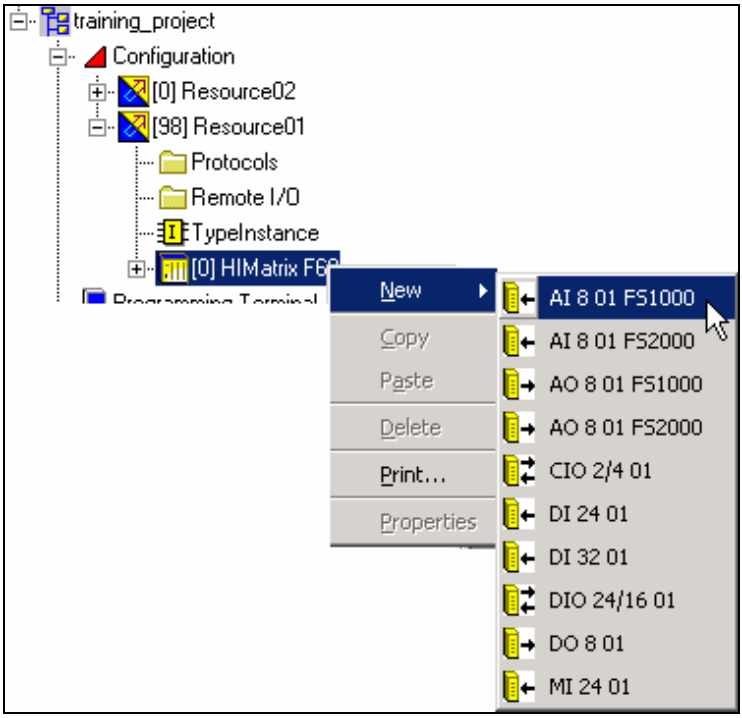


Fig. 82: Adding input/output modules

Step 2: Assign a slot to an input/output module:

☐ Open the context menu of the input/output module and click on **Properties**.

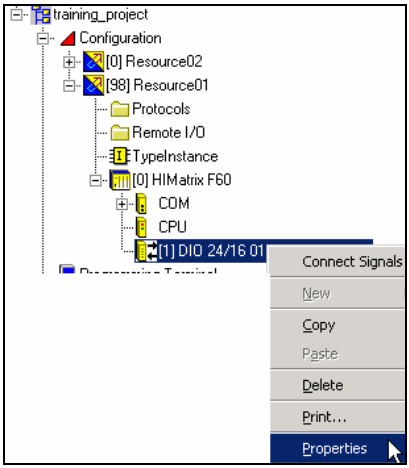


Fig. 83: Opening properties of the module

☐ Enter the correct slot number.

☐ Click **OK** to complete the input.

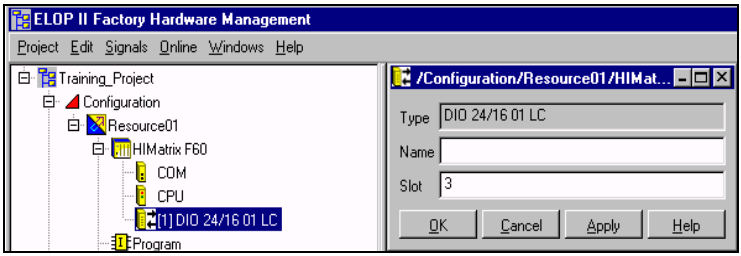


Fig. 84: Assigning slot number

Note: *A newly added module always has slot number 1. The slot number can have a value between 1 and 6 and must be unique within a resource.*

The slots are numbered in sequence from left to right. The power supply and the CPU do not have slot numbers, because their positions are factory preset.

7.7.5 Create a Remote I/O (RIO)

In the HIMatrix System there are two kinds of Remote I/Os:

1. HIMatrix F3 DIO 20/8 01
   Regarding communication via Safe**ethernet** the F3 DIO 20/8 01has the same capabilities like a resource and also has a System ID. Thus it is referred to as a resource.
2. Remote I/Os like HIMatrix F1, F2 or F3 DIO 20/8 02
   These Remote I/Os have limited communication capabilities and can only exchange data with their parent resource. Remote I/Os are part of a resource and are grouped in the “Remote I/O” directory of the structure tree. Remote I/Os must be configured with a Rack ID.

Step 1: Create a HIMatrix F3 DIO 20/8 01:

- ☐ Right-click on the “Configuration” in the Hardware Management.
- ☐ In the context menu select **New, HIMatrix F3 DIO 20/8.**

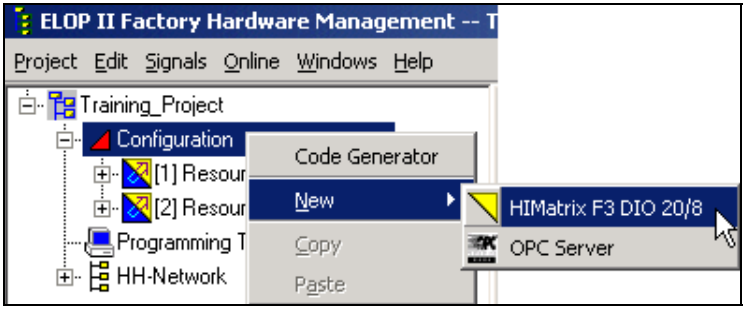


Fig. 85: Create a F3 DIO 20/8 01

Step 2: Rename the “F3 DIO 20/8 01” and assign a System ID:

- ☐ Right click on the resource and select **Properties** from the context menu.
- ☐ Enter a name in the field “Name”.
- ☐ Enter a System ID in the field “System ID [SRS]”.

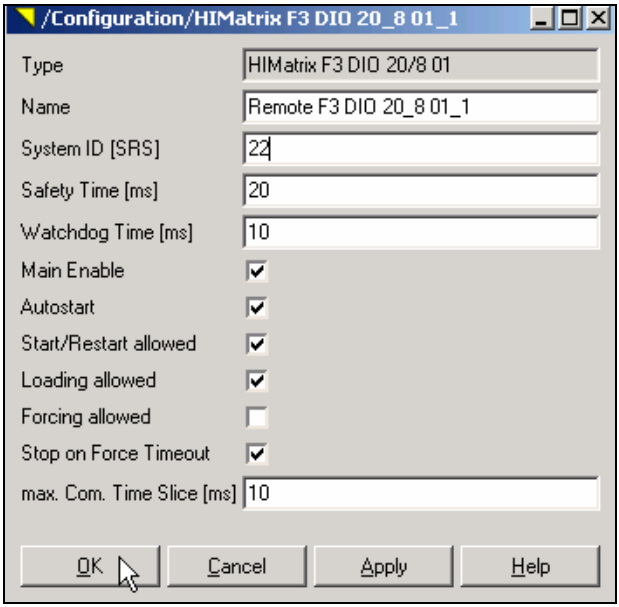


Fig. 86: Properties of F3 DIO 20/8 01

Step 3: Create a Remote I/O:

- ☐ In the Hardware Management open the folder of the parent resource.
- ☐ Open the context menu of the folder **Remote I/O**.
- ☐ Select **New** and the desired Remote I/O.

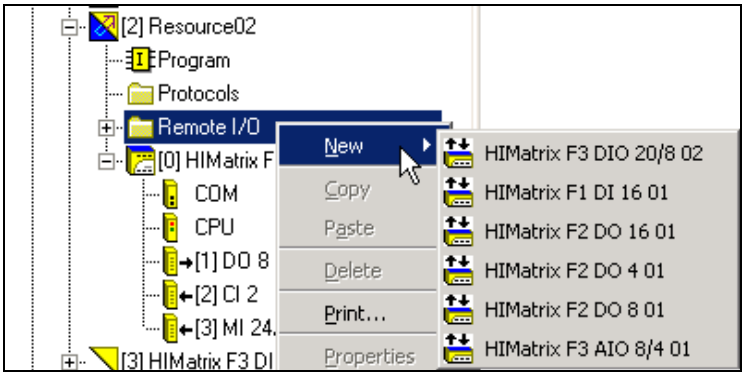


Fig. 87: Create a Remote I/O

Note: *If you have not created a HH-Network for Peer-to-Peer communication so far, or the parent resource has not been assigned to a Token Group, a dialog window is opened. Select a HH-Network and a Token Group or create new elements.*

Step 4: Assign a Rack ID to the Remote I/O:

☐ Open the context menu of Remote I/O and select **Properties**.

☐ Change the Rack ID [SRS] to a value > 0 and ≤ 511.

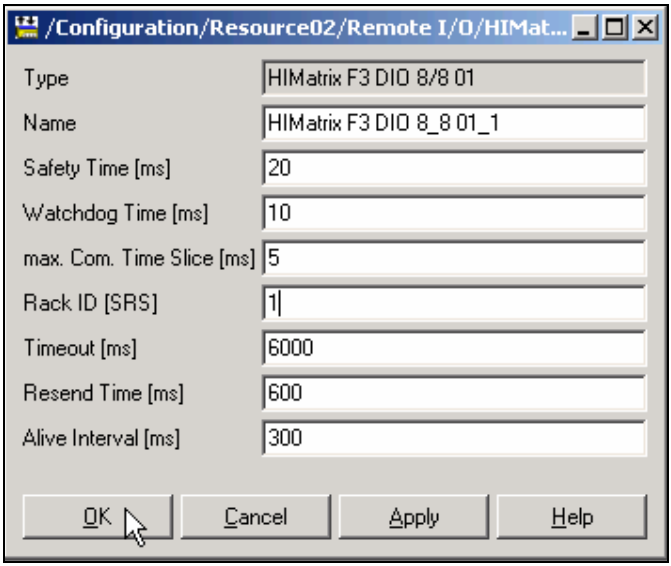


Fig. 88: Changing Rack ID

7.8 Signals

7.8.1 Definition of the difference between signal and variable:

A **Variable** is a placeholder for a value within the program logic. The memory cell address where the value is stored is symbolically addressed via the variable name.

Signals are used for data exchange between the individual components of a resource (e.g. user program, I/O channels) and safety-related and not safety-related data exchange with other resources. The signal comprises all assignment regulations for the data transfer defined with Drag & Drop.

If the value of a variable of the program instance is to be used in another area, a signal must be created in the Signal Editor of the Hardware Management. Then the signal is dragged into the variable declaration or the drawing field of the program and dropped there.

At this moment the program variable with the same name is transferred to the VAR_EXTERNAL tab or created there if it does not already exist. Finally, the signal has to be assigned to an input/output channel, a system signal or a communication partner.

If a variable that is linked to a signal is used in multiple function blocks as VAR_EXTERNAL, it must also be defined there in the same manner.
If you use a previously programmed function block that contains variables of the type VAR_EXTERNAL that have not yet been defined in the Signal Editor, you must define them now and assign the use of the signal by Drag & Drop.

Note: *Ideally the required signals are determined before starting the programming. These signals are created first in the Signal Editor and then defined in the program or function block with Drag & Drop.*
This also applies for all variables that do not acquire a hardware reference but whose value must be forced in the Force Editor during operation.

7.8.2 Defining Signals

Step 1: Open the Signal Editor:

☐ Click **Signals, Editor** in the menu bar.

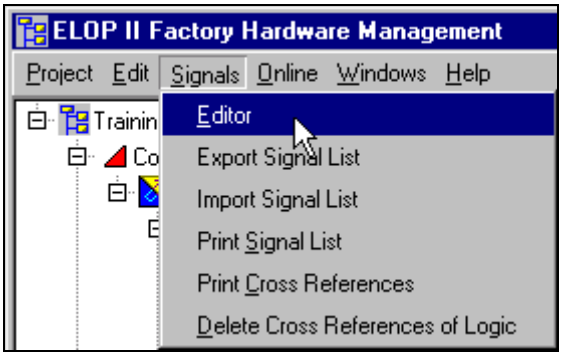


Fig. 89: Opening Signal editor

Fig. 90: Signal editor without signals

Step 2: Define the signal(s):

- ☐ Click the **New Signal** button.
  A new input line opens.
- ☐ Input data into the "Name“ and "Type“ boxes. These inputs are required.

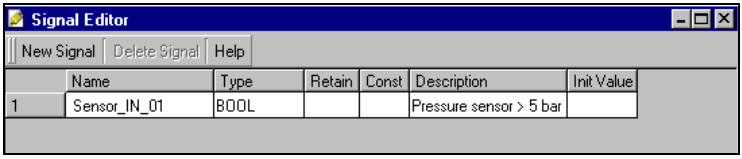


Fig. 91: Input of signal data

- ☐ "Retain“ or "Constant“ marks are set and cleared by double-clicking in the respective field.
- ☐ Take care about lower/upper case characters.
  The Signal Editor will not accept two signals with identical spelling.

Important: *Never enable "Retain“ and "Constant“ simultaneously. This will result in error messages during code generation.*

The user program must have read and write access to signals with the "Retain“ attribute. However, write access to signals with the "Constant“ attribute is not possible.

Step 3: Define the signal use:

The use of signals in function blocks or in the user program is defined with Drag & Drop. Proceed as follows:

- ☐ If there are applications open other than ELOP II Factory (e.g. e-mail program), minimise these applications.
- ☐ Click the Windows taskbar with the right mouse button. Select **Tile Windows Horizontally** in the context menu and arrange Project Management and Hardware Management one on top of the other (see Fig. 93).

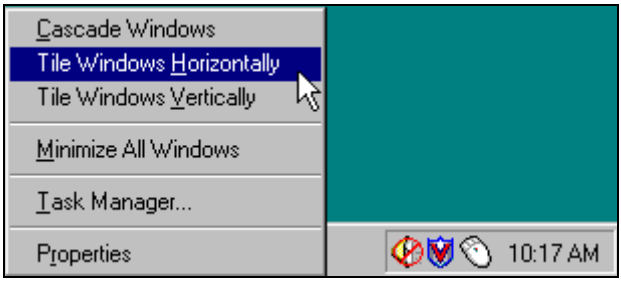


Fig. 92: Tile Windows horizontally

- ☐ Enlarge the program or function block drawing area pane so that you can see as much as possible (Fig. 93 top).
- ☐ Enlarge the signal list pane in the Signal Editor so that you can see as many signals as possible (Fig. 93 bottom).

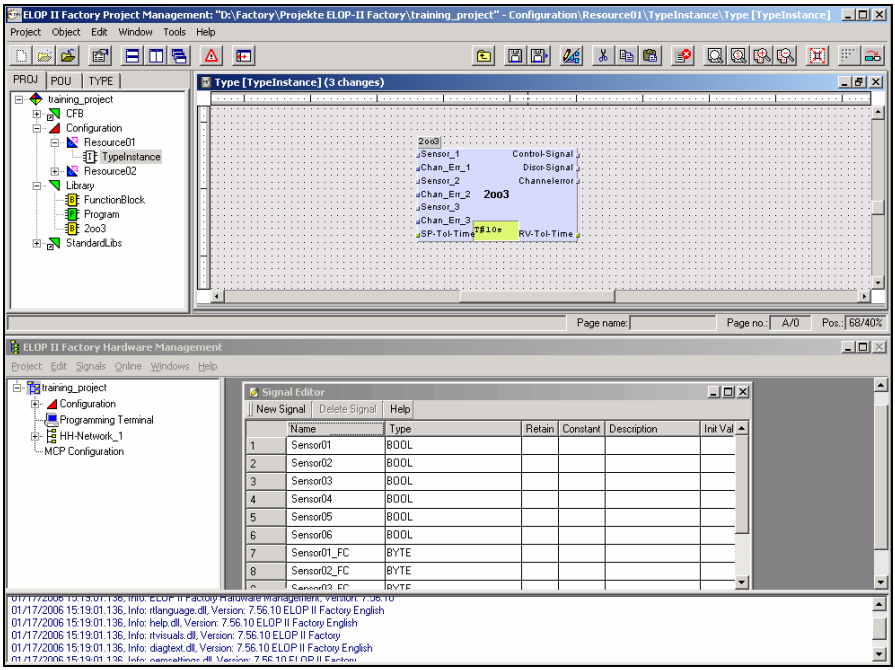


Fig. 93: Drawing area and Signal editor maximised

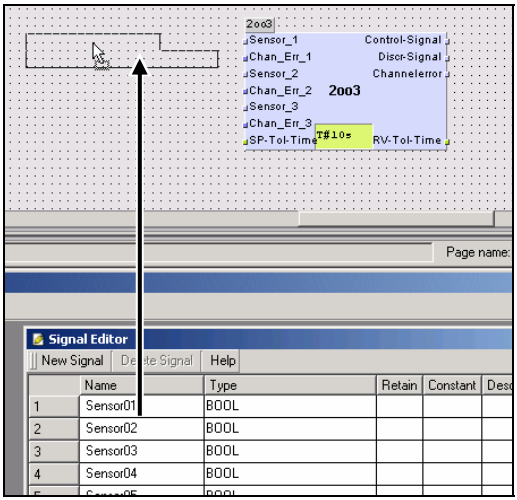


Fig. 94: Dragging & dropping signals into the logic

☐ Drag a signal from the Signal Editor directly to the logic or into the variable declaration. The signal is automatically created in the VAR_EXTERNAL tab.

Note: *If there is already a variable with identical spelling in the VAR_EXTERNAL tab, the variable is updated with the new data.*
If there is a variable with the same name in an other tab than VAR_EXTERNAL, the variable is transferred to the VAR_EXTERNAL tab after a prompt and it is updated with new data.

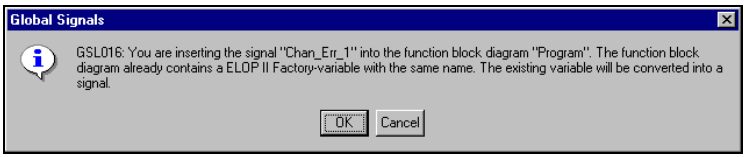


Fig. 95: Prompt before conversion of a variable into a signal

7.8.3 Assigning Signals to I/O Channels

Step 1: Open the window for signal assignment:

- ☐ Open the Signal Editor (see chapter 7.8.2).
- ☐ Right-click on an I/O module and select **Connect Signals** from the context menu.

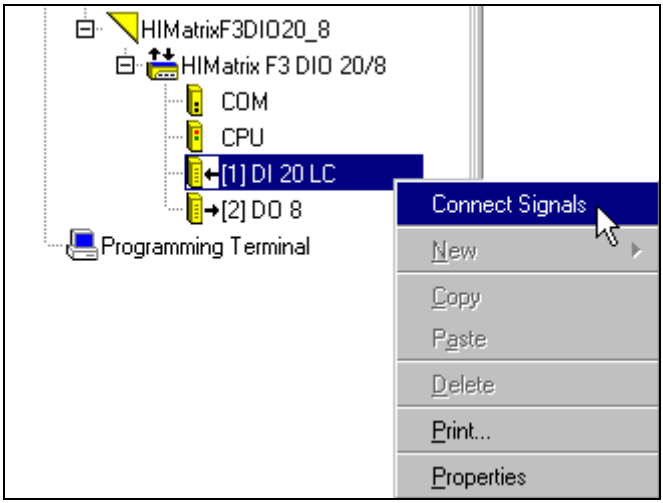


Fig. 96: Assigning signals to input/output channels

Step 2: Tile the windows:

- ☐ Tile the Signal Editor and the "Signal Connections" windows side by side.

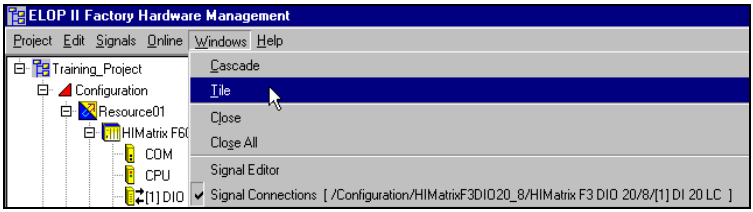


Fig. 97: Tiling windows

Step 3: Allocate the signal(s):

☐ In the Signal Editor click on a signal name and drag the signal into the "Signal" column of the “Signal Connections” window.

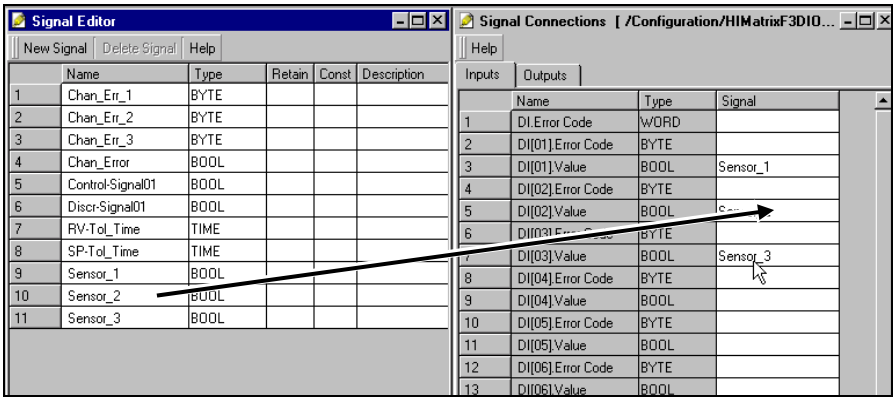


Fig. 98: Allocating signals

Note *It is possible to select several signals at the same time and Drag & Drop them en bloc into the “Signal Connections” window. In this case it is necessary that the signals in the Signal Editor are in the same ordering like the channels in the signal connections.*
In every window it is possible to influence the sorting by clicking in a column title.

Note: *In ELOP II Factory a module is mirrored by signal inputs and outputs, even if the module is only furnished with physical outputs. A module provides diagnostic signals about the module status and error codes about the channel status. The data direction tells on whether it is an input or output.*

Error codes for physical inputs and outputs are located in the "Inputs" tab, because these are input values for the user program.

Parameters are assigned in the "Outputs" tab, independently of whether it concerns parameters for physical inputs or outputs.

See also "Signals and Error Codes..." in the HIMatrix controller data sheets.

	Input module		Output module	
	Input	Output	Input	Output
Hardware signal from or to field	X	-	-	X
Error codes of channels or module	X	-	X	-
Setting parameters or configuring the channels	-	X	-	X

7.8.4 Assigning Signals to the System Signals

Similar to the chapter before the signal editor should be open and it has to be opened the signal connection of the CPU.

For the assignment itself it is to be done in the same way like for the I/O-channels in chapter 7.8.3.

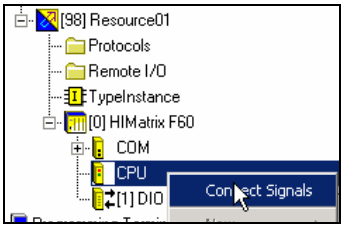


Fig. 99: Opening the system signal connection

See also the description of the system signals in the system manuals. Especially the inputs power supply state, remaining force time and temperature state should be monitored.

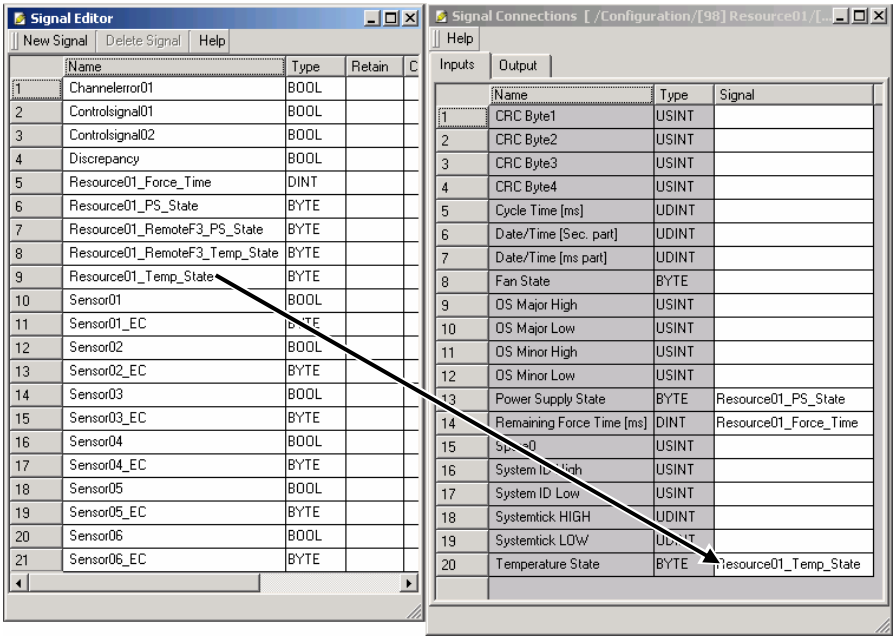


Fig. 100: CPU inputs

7.9 Communication with other HIMatrix Controllers

Communication with other controllers is primarily for the purpose of exchanging signals between the controllers.

7.9.1 Peer-to-Peer Communication (P2P Communication)

Peer-to-Peer communication is defined as communication between two nodes within the same network without requiring a communication master.

Peer-to-Peer communication is currently only possible between controllers of the HIMatrix family.

To enable two or more HIMatrix controllers to exchange signals with each other, a network must be set up first and the nodes (participants) must be defined in the network. Then the communication partners must be defined for every node. This is done in the Peer-to-Peer Editor of the particular resource. Communication signals are then defined for every Peer-to-Peer connection.

The configuration is carried out in the Hardware Management.

Step 1: Create network and Token Group:

☐ Select **New, HH-Network** in the “Project” context menu.

Fig. 101: Adding HH-Network

The newly created network is added to the project tree and already contains a Token Group.

Fig. 102: Token Group

☐ If necessary, change the name in the network context menu in **Properties**.

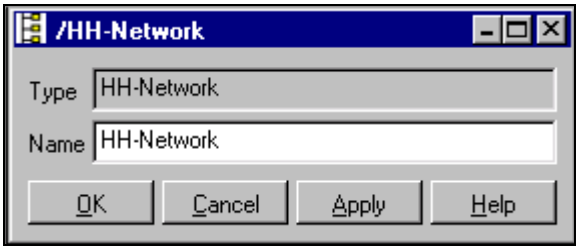


Fig. 103: Network properties

☐ Open the Token Group **Properties** in the context menu.

☐ Rename the Token Group in the "Name" field if required.

☐ Select the "Fast“ profile (default setting).

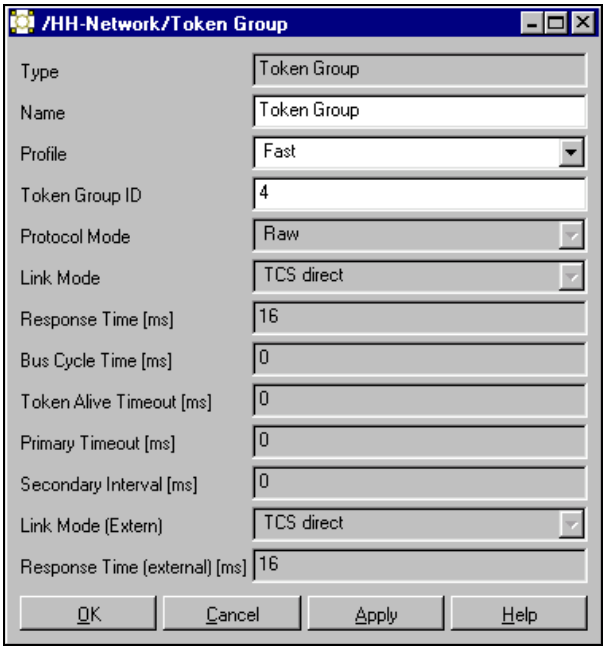


Fig. 104: Properties of the Token Group

Note: *"Fast“ is the profile normally utilized for networks in which switches are used.*
"Medium“ is the profile normally utilized for networks in which hubs are used.
The "None“ setting is required for manual network configuration. However manual network configuration is very complex due to the multiplicity of parameters and should be accomplished only by experienced users
If there are communication partners with different profiles in the same network, the communication partners need to be in different Token Groups within the network. These Token Groups need to have different Token Group IDs.

Step 2: Define participants in a Token Group:

☐ In the context menu of the Token Group select **Node Editor** to define the members of a Token Group.

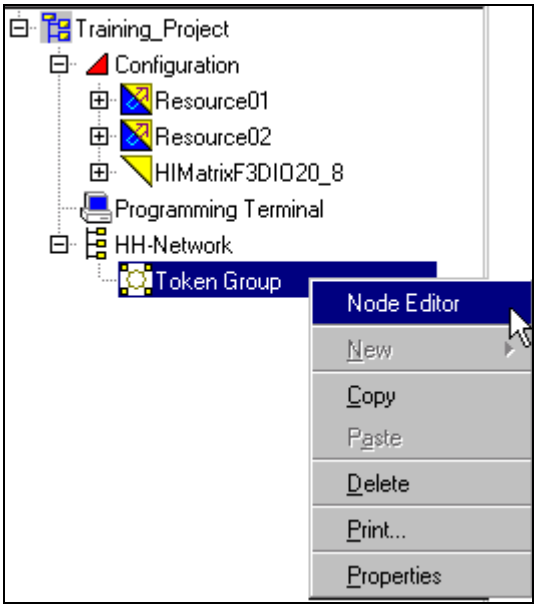


Fig. 105: Opening the Node Editor

☐ Drag the resources that belong to the selected Token Group from the structure tree to the node editor.

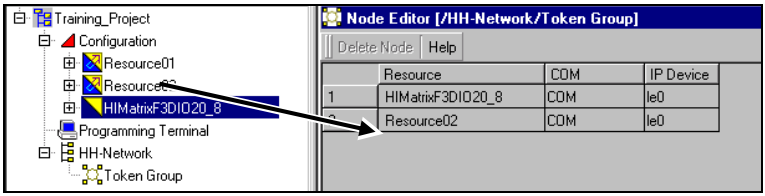


Fig. 106: Declaring participants of the Token Group

Step 3: Declare communication partners for a resource:

- ☐ Open the **Peer-to-Peer Editor** in the context menu of a resource.
- ☐ Drag resources from the structure tree to the Peer-to-Peer Editor with which the resource is to communicate.

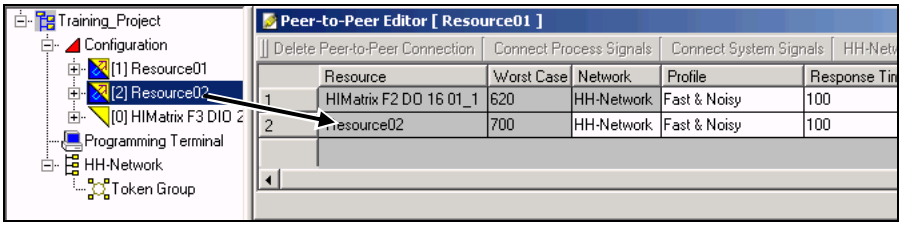


Fig. 107: Defining communication partners

- ☐ If not already existing, enter the network name in the "Network“ column (case-sensitive), or drag the network name from the structure tree to the "Network“ column.
- ☐ Define the desired profile in the "Profile“ column. The default profile is "Fast & Noisy“.

Note: *When you define a Peer-to-Peer connection for a resource, the return path required for the communication partner is automatically created. You only need to define one direction for a communication path.*

Note: *There is a detailed description of all profiles in the Online Help of the Hardware Management under "Peer-to-Peer Communication -> Peer-to-Peer Network Profiles“.*

Step 4: Assign Process Signals for Peer-to-Peer communication:

☐ Open the Peer-to-Peer Editor.

☐ In the Peer-to-Peer Editor click on the row number next to the name of a communication partner.
This highlights the row and activates the buttons in the toolbar.

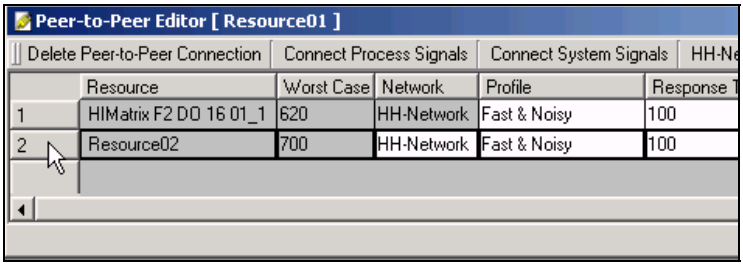


Fig. 108: Connecting process signals

☐ Click **Connect Process Signals** in the toolbar.

☐ Select the tab for the direction of communication.

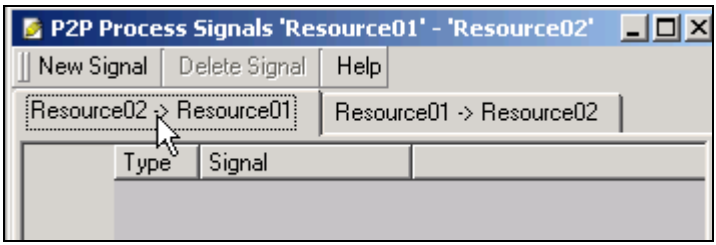


Fig. 109: Selecting direction of communication

☐ Open the Signal Editor with **Signals, Editor** and position it beside the "P2P Process Signals“ window.

☐ Position the cursor over a signal name in the Signal Editor and copy it into the "P2P Process Signals“ window with Drag & Drop.

Note: *It is possible to select several signals at a time and copy them into the “Peer-to-Peer Process Signals” window.*

Take care of the selected direction of transfer.

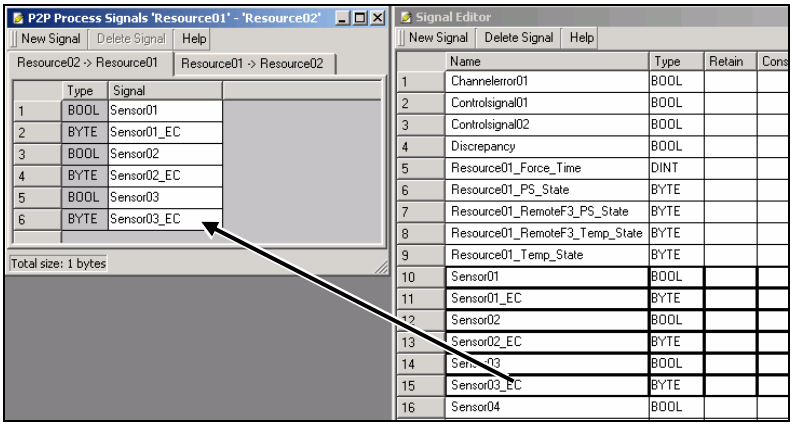


Fig. 110: Assigning Process Signals

The left side of Fig. 111 shows the signals that are sent from "Resource01“ to "Resource02“.

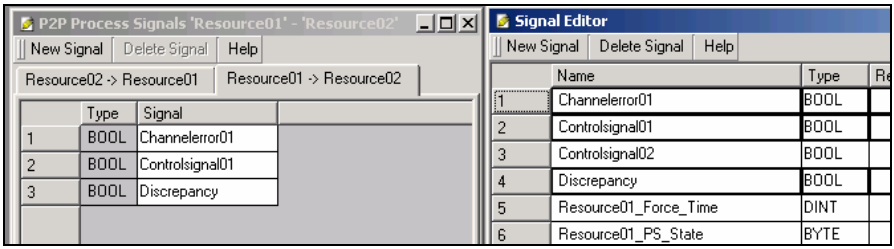


Fig. 111: "Resource01" export signals

The left side of Fig. 112 shows the signals that are sent from "Resource02“ to "Resource01“.

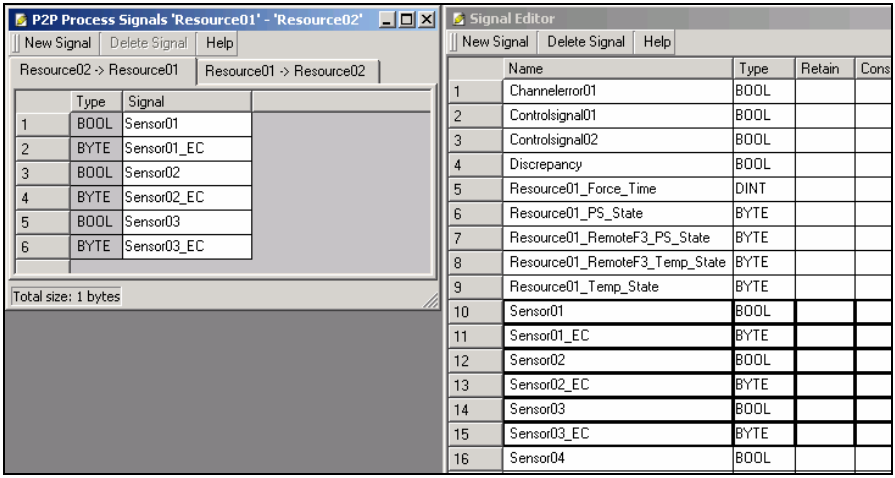


Fig. 112: "Resource01" import signals

Step 5: Assign System Signals for Peer-to-Peer communication:

System Signals of P2P-Communications are specific in point of view of that resource from where you opened the P2P-Editor. To make it complete you have to look for all P2P-Editors of all resources and for all individual connection.

- ☐ In the Peer-to-Peer Editor click on the row number next to the name of a communication partner.
  This highlights the row and activates the buttons in the toolbar.
- ☐ Click the button **Connect System Signals**.

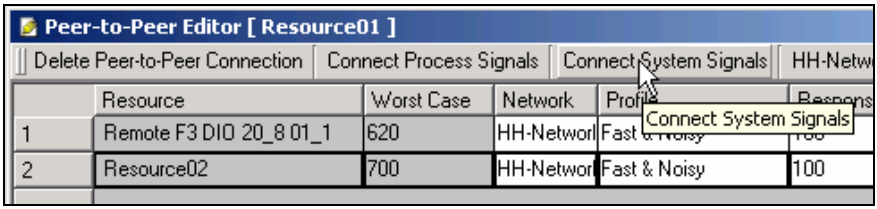


Fig. 113: Connect System Signals

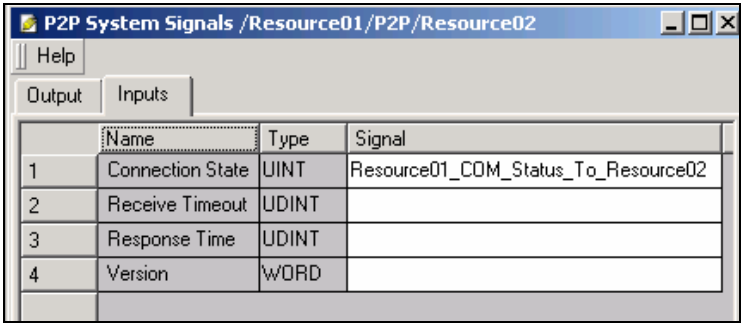


Fig. 114: Inputs of the System Signals of one connection

Note: *Please find the complete description in the System Manual in the chapter "Operating system", subchapter "System signals of a Peer to Peer connection".*

7.10 Code Generator

After completing the configuration and the user program, the resource configuration that can ultimately be loaded into the resource has to be generated.

- Regardless of the resource type every controller must be configured.
- Remote I/Os cannot contain a user program. In addition to the configuration, the input/output channels must be defined. The definition of the input/output channels is also a part of the resource configuration.
- Resources can have a user program. The user program is also a part of resource configuration.

Step 1: Start the Code Generator the first time:

a.) For resources:

- ☐ Open the Project Management.
- ☐ Open the context menu of a resource and select **Code Generation**.
  The code for the assigned Remote I/Os is automatically included.
- ☐ If you want to start the Code Generation for all resources together then open the context menu of the configuration and select **Code Generation**.

b.) For HIMatrix F3 DIO 20/8 01:

- ☐ Change to the Hardware Management and open the context menu of the HIMatrix F3 DIO 20/8 01 and select **Code Generation**.
- ☐ If you want to start the Code Generation for all HIMatrix F3 DIO 20/8 01 together then open the context menu of the configuration and select **Code Generation**.

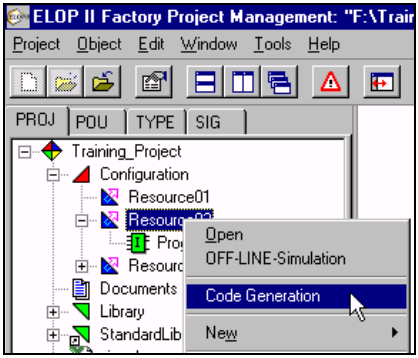


Fig. 115: Starting code generation for a resource

Note the messages in the Error-State-Viewer of the Project Management and the Hardware Management.
An abortion of the code generation can be caused by errors in the configuration and/or by error in the logic of the user program.

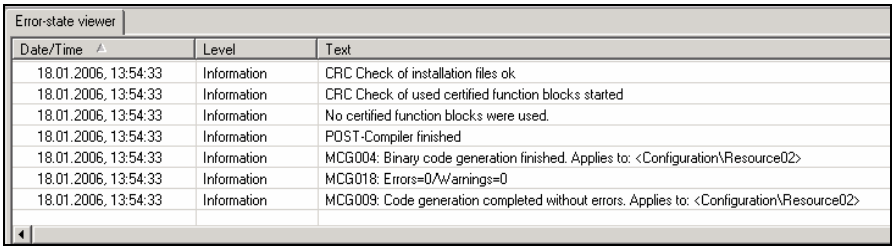


Fig. 116: Error-State-Viewers after code generation

Note: *In case of warnings also note the preceding messages.*

Step 2: Start the Code Generator a second time:

- ☐ If code generation was successful, start the code generator a second time.

For safety-related applications you must start the code generator twice and compare the checksums (CRCs) of the two generated code versions. Both code versions must be identical.
Thus errors are avoided, which can be caused by a non-safe standard PC.
See the Safety Manual for additional details.

Step 3: Check the two CRC-Values:

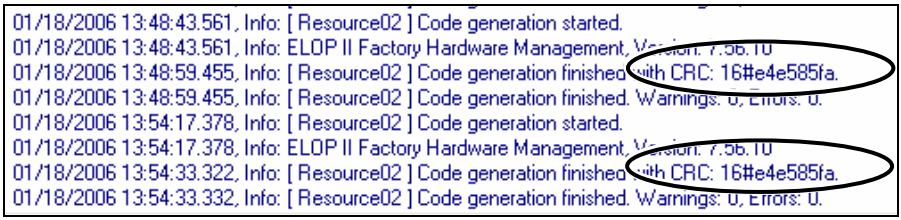


Fig. 117: Check Error-State-Viewer in the Hardware Management

In case of an older version of ELOP II Factory the CRC is not shown in the Error-State-Viewer. Then you have to get the data from the Window “About Configuration”

☐ In the Hardware Management open the context menu of the resource and select **About Configuration**.

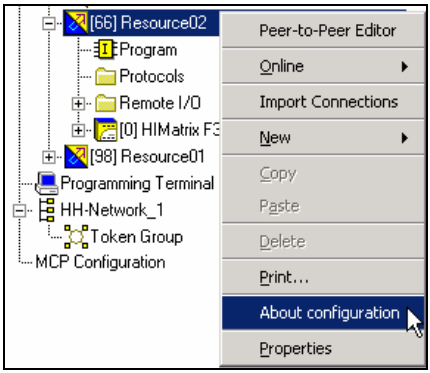


Fig. 118: Open “About configuration”

☐ Write down the checksums for "root.config“ in the column "CRC PADT“, or make a screen shot and copy it e.g. to a Word file.

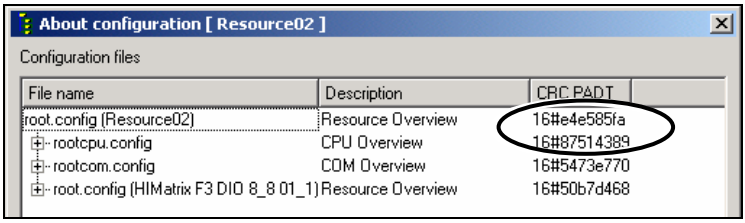


Fig. 119: Noting checksums

☐ Run the code generation again as described in **Step 1**.

☐ Open the "About configuration“ window.

☐ Compare the checksums of the second code generation with the previously noted checksums.
Only if the two check sums are identically it is allowed to load the code on the resource (see chapter 7.12).

7.11 Configuring the PC and the Controllers

So far all settings were done in the project, e.g. the IP addresses of the controllers.
But the controllers have to be parameterised so that they really work with these settings.

7.11.1 Parameterise the PC

Step 1: Set the IP address of the Programming Terminal (PC):

- ☐ Click on **Start** and select **Settings, Network and Dial up Connections, LAN**.
- ☐ Open the **Properties** for the TCP/IP protocol.
- ☐ Select "Use the following IP Address“.
- ☐ Input an IP address, which fits your network and is not used by a controller.
- ☐ Input a valid subnet mask in the field "Subnet Mask“.

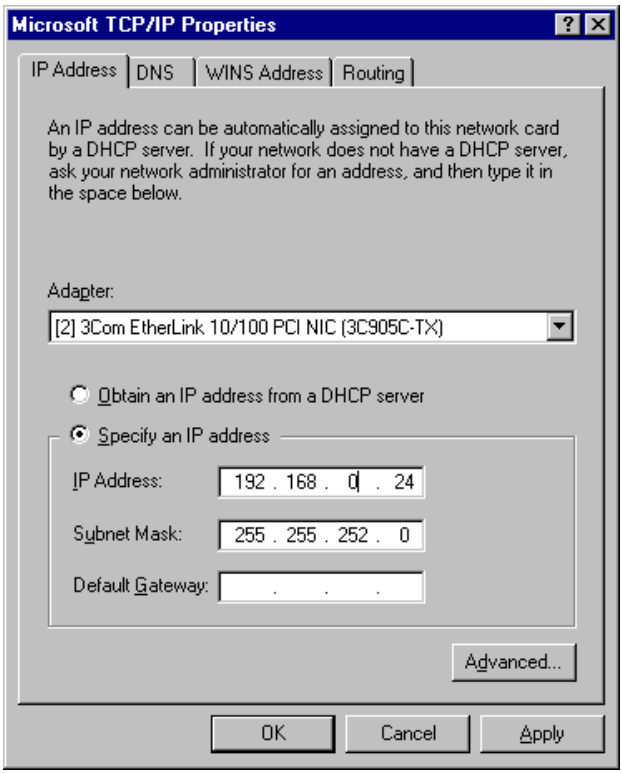


Fig. 120: Setting the IP address of the Programming Terminal

Note: *Usually Windows 2000 and Windows XP do not have to be restarted to active these settings.*

7.11.2 Configuring the Controller for Communication

The following description is valid for resources and Remote I/Os.

A new controller features the following factory settings (unless otherwise specified):	
IP address	192.168.0.99
Subnet Mask	255.255.252.0
System.Rack ID for resources	60000.0
System.Rack ID for Remote I/Os	60000.1
User name	Administrator
Password	Blank

Step 1: Connect the controller to a power supply:

- ☐ Disconnect all external connections (Ethernet cable, inputs and outputs).
- ☐ Connect the controller to a sufficiently rated power supply.
- ☐ After 30 seconds check if the controller is in RUN mode (RUN LED is continuously on), or if it remains in STOP-Mode (RUN LED is flashing, for the F60 the STOP LED is continuously on).

- If the controller is in STOP mode after the booting, continue with **Step 2**.
- If the controller is in RUN mode after the booting, it has to be stopped (see chapter 7.11.3).

Step 2: Configure the IP address:

- ☐ Connect the Ethernet port of the Programming Terminal to one of the Ethernet ports on the controller.
- ☐ Select the Hardware Management.
- ☐ Select **Online, Communication Settings** in the resource or Remote I/O context menu.

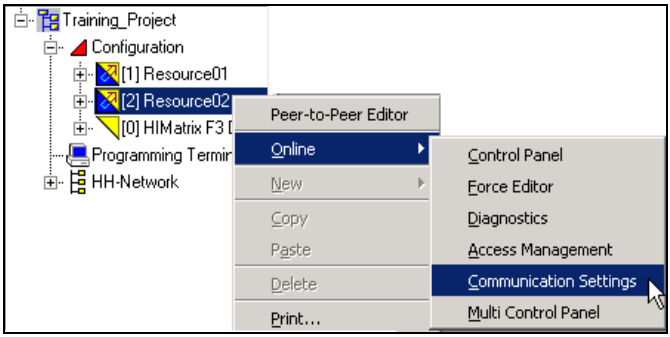


Fig. 121: Open «Communication Settings»

The window “Communication Settings” opens with the data of the project.

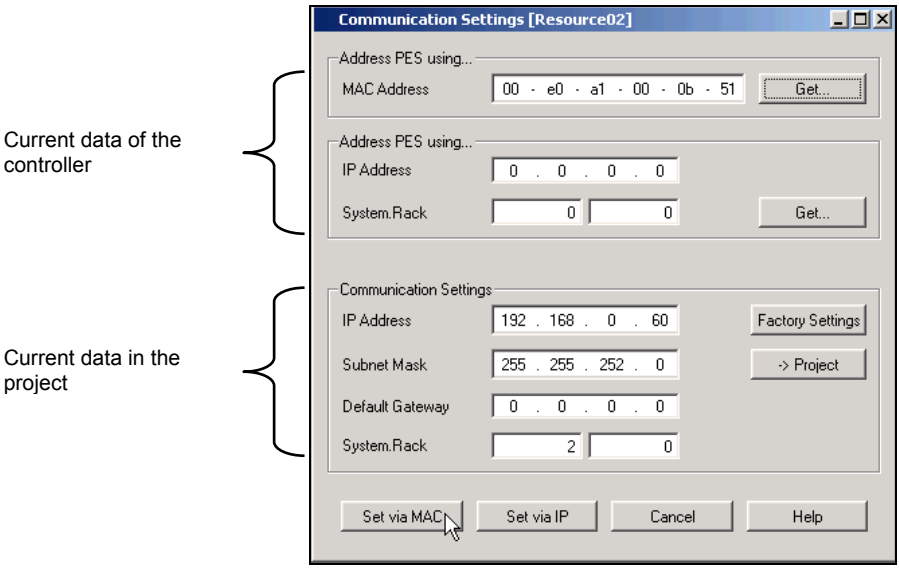


Fig. 122: Setting IP address and System.Rack ID via MAC address

☐ Enter the “MAC Address” of the controller. You find the MAC address sticker on the bottom side of the controller near the Ethernet connectors, or for the F60 directly on the CPU module.

☐ Click on **Set via MAC**.

Note: *If you know the current IP address and System.Rack ID of the controller, you also can change the settings by **Set via IP**.*

☐ Enter a user name with administrator rights and the corresponding password when you are asked for authentication. The factory setting is “Administrator” without a password.

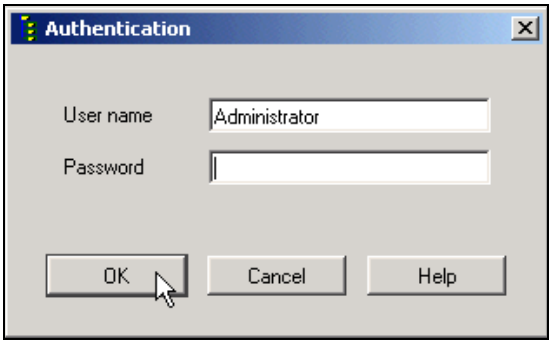


Fig. 123: Authentication

Note: *Watch the Error-State-Viewer for the “…successful” message on data change.*
If the entered user name is not accepted, check for the correct writing.
If the administrator access is not known, you must set the controller to the factory defaults. See chapter 7.11.4.

After setting the parameters for all resources and Remote I/Os you can download your application into the resources (see chapter 7.12).

7.11.3 Stopping a controller running an unknown Project

If a controller goes into mode RUN after the initialisation, you must stop the controller in order to configure the communication settings.

You can do this with the help of a dummy resource.

Step 1: Creating a dummy resource:

- ☐ Select the Hardware Management.
- ☐ In the context menu of the configuration select **New, HIMatrix F3 DIO 20/8 01**.

Note: *The resource type is not important for the configuration of the communication settings. This means that the chosen resource type need not to match the real type. You can configure several controllers one after the other with the same resource object.*

- ☐ Double click on **HIMatrix F3 DIO 20/8 01** to open the properties.
- ☐ Rename the **HIMatrix F3 DIO 20/8 01** to **Dummy** and enter any System ID.
- ☐ Confirm the changes with **OK**.

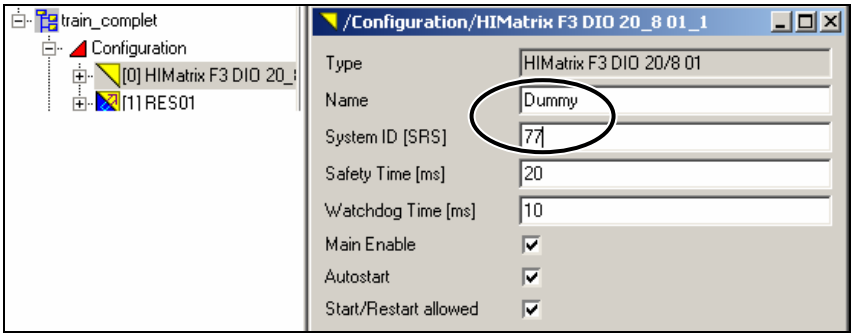


Fig. 124: Changing the properties of the dummy resource

Step 2: Connecting the controller:

☐ Connect the Ethernet port of the Programming Terminal to one of the Ethernet ports of the controller.

Step 3: Reading out the current communication parameter from the controller:

☐ Select **Online, Communication Settings** in the context menu of the dummy resource

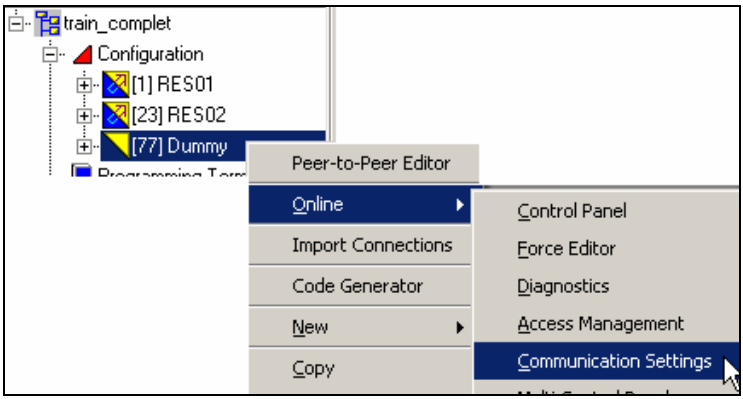


Fig. 125: Open «Communication Settings»

☐ Enter the “MAC Address” of the controller. You find the MAC address sticker on the bottom side of the controller near the Ethernet connectors, or for the F60 directly on the CPU module.

For manually entering the current controller data

Current data in the project

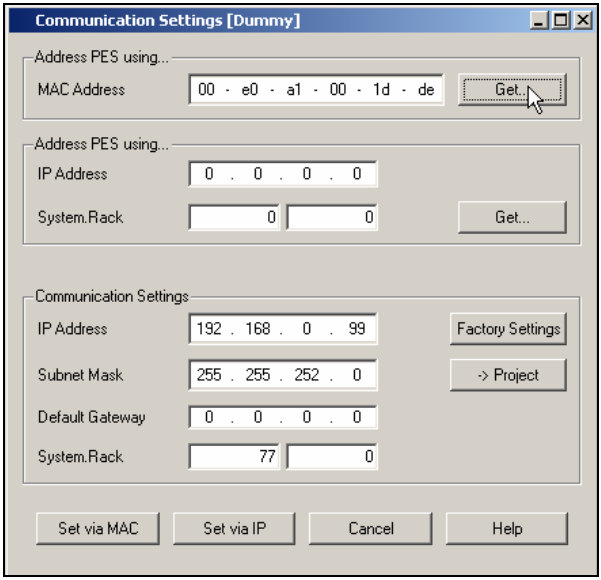


Fig. 126: Enter the MAC address

☐ Click on **Get...** to the right of the MAC address. The communication settings of the controller will be displayed in the group box “Communication Settings”

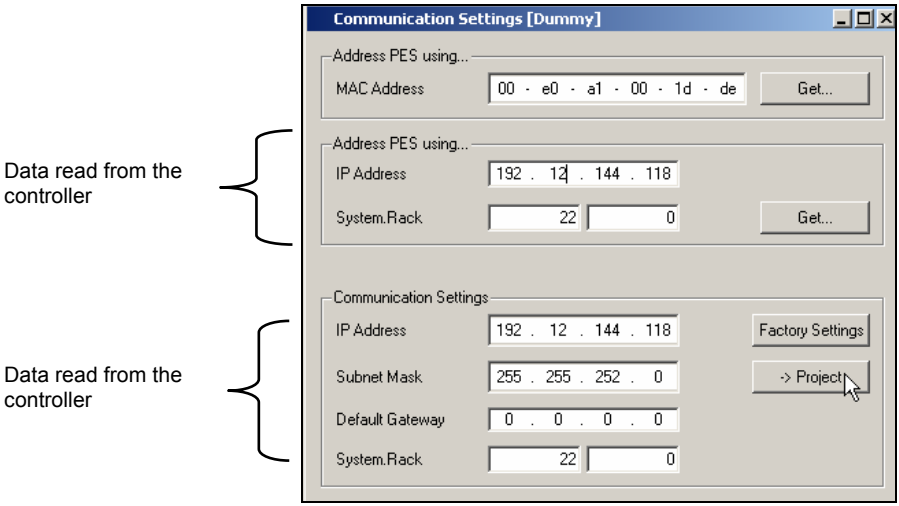


Fig. 127: Read data from the controller

☐ Click on the button **Project**.
The data read out from the controller are stored in the dummy resource.

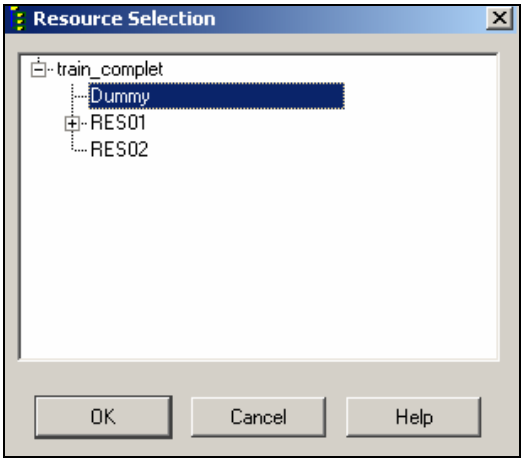


Fig. 128: Selection of the resource in the project

Step 4: Stopping the Controller by means of the Control Panel:

Note: *Leave the window "Communication Settings" open.*

☐ In the context menu of the dummy resource select **Online, Control Panel**.

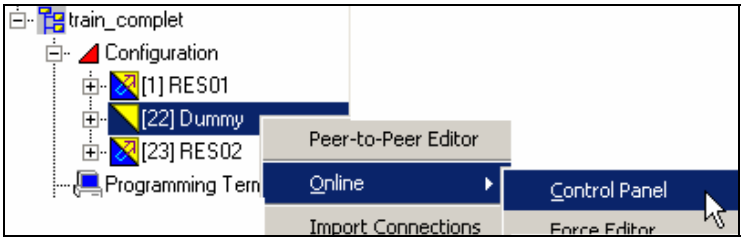


Fig. 129: Open the Control Panel for the dummy resource

☐ Log in with User name and Access type «Administrator». If a different user account is configured for administrator, log in with those data.

Note: *If the entered user name is not accepted, check for the correct writing.*
If the administrator access is not known, you must set the controller to the factory defaults. See chapter 7.11.4.

Fig. 130: Login dialog

- ☐ Click on **Stop** in the toolbar or select **Resource, Stop** from the menu.
- ☐ Confirm the question with **Yes**.

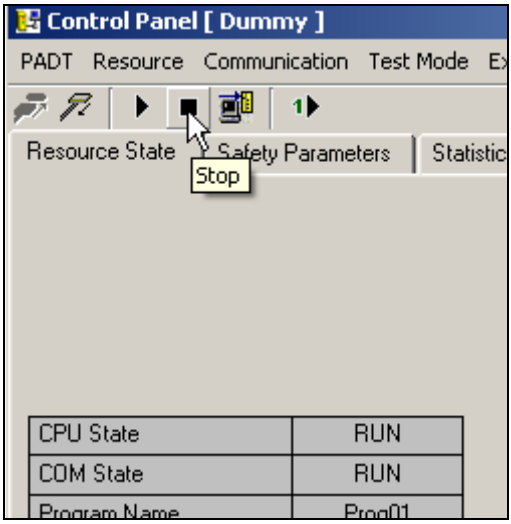


Fig. 131: Stopping the resource

Step 5: Changing the System ID and IP address according to the project data:

- ☐ Close the Control Panel.
- ☐ Click on the resource which actually represents your controller and whose parameter you want to set. If the Communication Settings window is still open the data of the selected resource are displayed in the **Communication Settings** group box.
- ☐ For the configuration of the IP address and System.Rack ID continue as described in chapter 7.11.2 starting from Fig. 122.
- ☐ Delete the dummy resource if it is no longer needed.

7.11.4 Activating Factory Defaults

Note: *Resetting the controller is necessary if the user name and password of the administrator is unknown.*
If it is only the IP address that does not match your network you can manage the connection with the help of a "route add" entry on your PC. Please ask your network administrator.

Step 1: Pressing the reset push button:

Attention: *The reset push button has to be handled very carefully.*

Note: *For the compact modules the push button is accessible through a small round hole on the upper side of the housing, approximately 4 - 5 cm from the left. For the F60 and F20 the hole is on the front of the housing.*

- ☐ Disconnect the power supply.
- ☐ Press the Reset push button with a thin pin made of insulating material.
- ☐ Reconnect the power supply while you keep the push button depressed for at least 20 seconds.

Note: *After the initialisation the controller remains in STOP mode. All settings are reset to factory defaults.*
For COM operating system version 10.42 and newer the controller will reject the download of a user program after an initialisation with operated reset push button.
Prior to downloading you first have to parameterise the communication settings and the access management on the controller and do a normal power off/on reboot.

Step 2: Changing the communication data according to the project data:

- ☐ Connect the Ethernet port of the Programming Terminal to one of the Ethernet ports on the controller.
- ☐ Select the Hardware Management.
- ☐ Select **Online, Communication Settings** in the resource context menu.

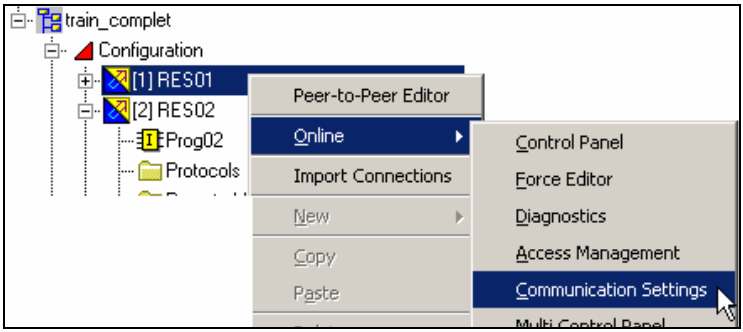


Fig. 132: Open the «Communication Settings»

The Communication Settings window displays the data according to the project settings.

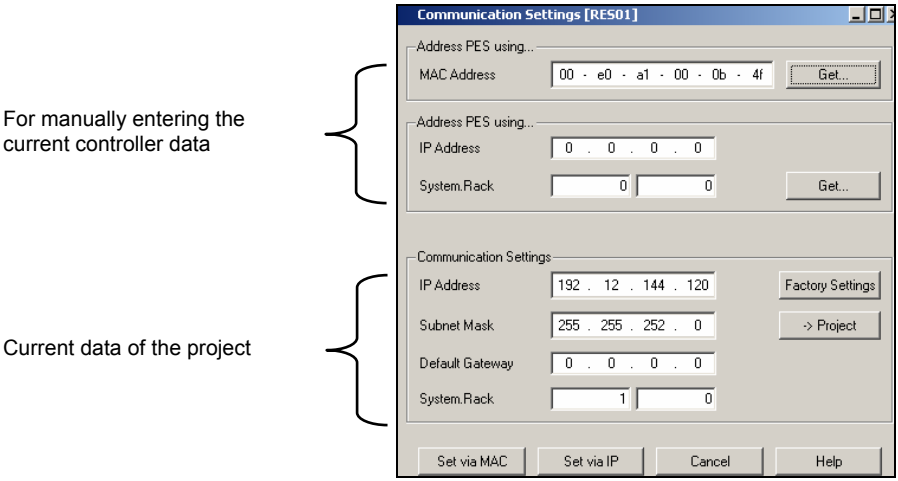


Fig. 133: Setting the Communication data via MAC address

- ☐ Enter the “MAC Address” of the controller. You find the MAC address sticker on the bottom side of the controller near the Ethernet connectors, or for the F60 directly on the CPU module.
- ☐ Click on **Set via MAC**.
- ☐ When you are asked for authentication, enter the User name **Administrator**. Note that the password remains empty.

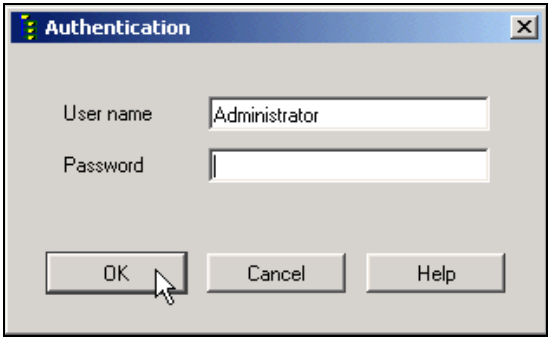


Fig. 134: Authentication

Note: *Watch the Error-State-Viewer for the “successful” message of the data change.*

Step 3: Set the Access Management to the default:

- ☐ Open the Access Management from the context menu of the resource.

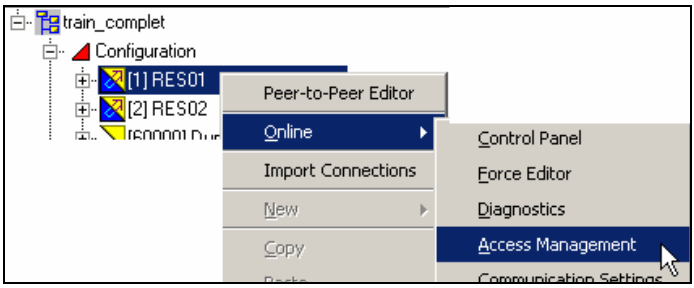


Fig. 135: Open the «Access Management»

☐ Click on **Connect** to start communication with the controller.

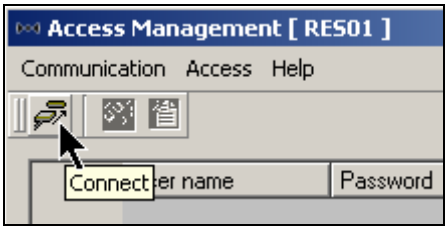


Fig. 136: Connect the controller

☐ Log in with the User name and Access type “Administrator” and leave the Password empty (shortcut CTRL + a).

☐ Click on **Default account** in the toolbar to erase all individual accounts and active the default.

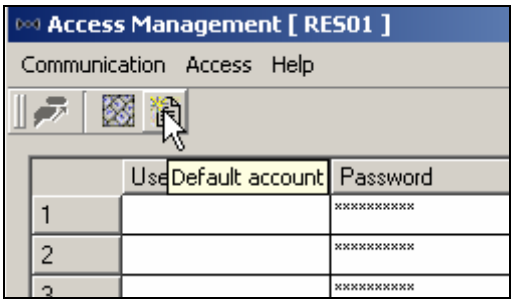


Fig. 137: Activating the default account

Step 4: Rebooting the controller:

☐ Switch off the power supply of the controller and after a short while switch it on again.

Now you can load your application program into the controller. See chapter 7.12.

7.12 Loading and Starting the Program (Resource Configuration)

Before the resource configurations can be loaded into the resources, the code for the resources must have been generated as described in chapter 7.10 and the Programming Terminal and the resources must have valid communication settings (see chapter 7.11).

Step 1: Open the Control Panel:

☐ Select **Online, Control Panel** in the resource context menu.

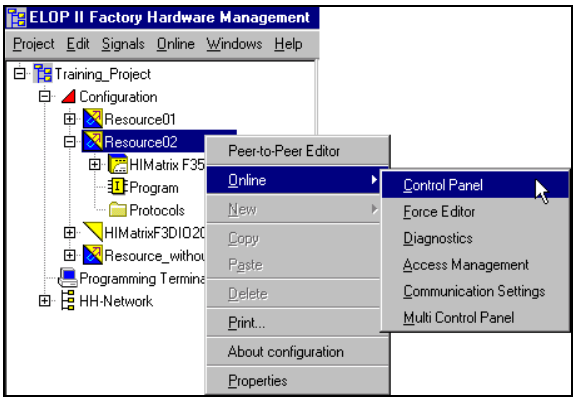


Fig. 138: Open the «Control Panel»

☐ Log in as «Administrator» for “User name” and “Access type” (shortcut: CTRL + A).

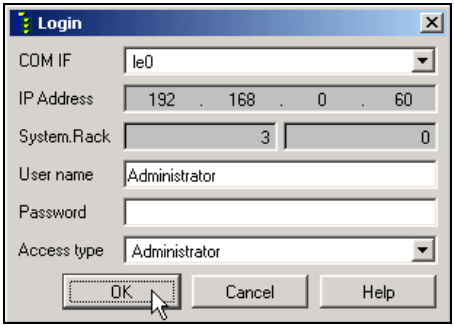


Fig. 139: Entering user name and access type

Step 2: Load the resource configuration:

☐ The controller must be in STOP mode. If necessary run **Resource, Stop**.

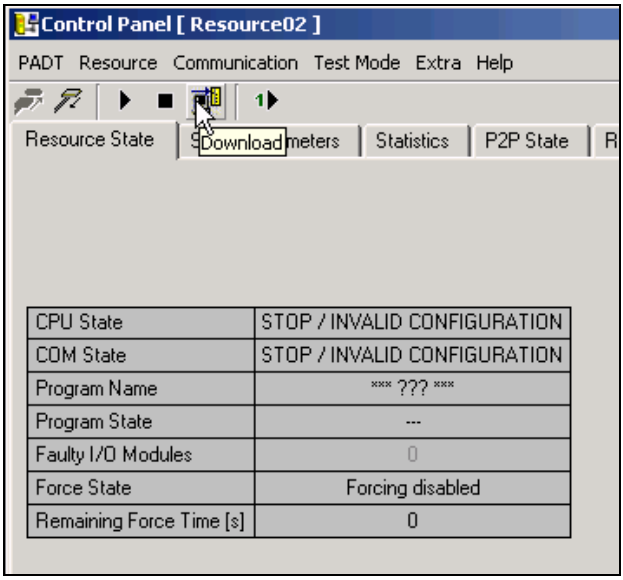


Fig. 140: Control Panel

☐ Click **Download** in the toolbar.
A prompt appears (see Fig. 141).

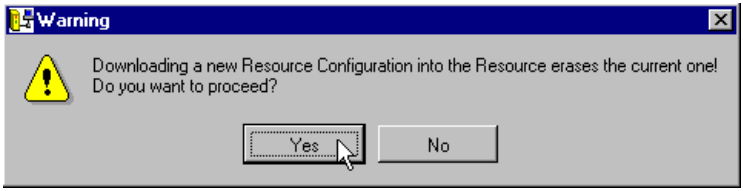


Fig. 141: Prompt before download

The download starts as soon as the prompt is acknowledged with **Yes**.

Step 3: Start the user program:

☐ Click on **Coldstart** in the toolbar.

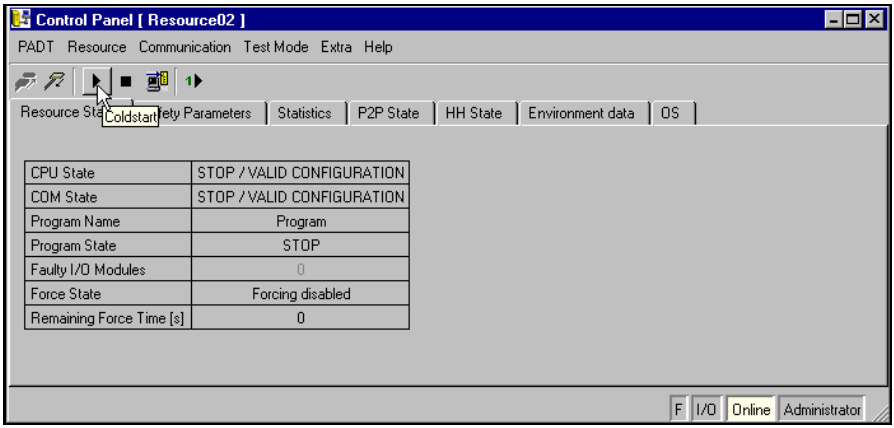


Fig. 142: Resource in STOP mode

After the coldstart "CPU State“, "COM State“ and "Program State“ change to RUN.

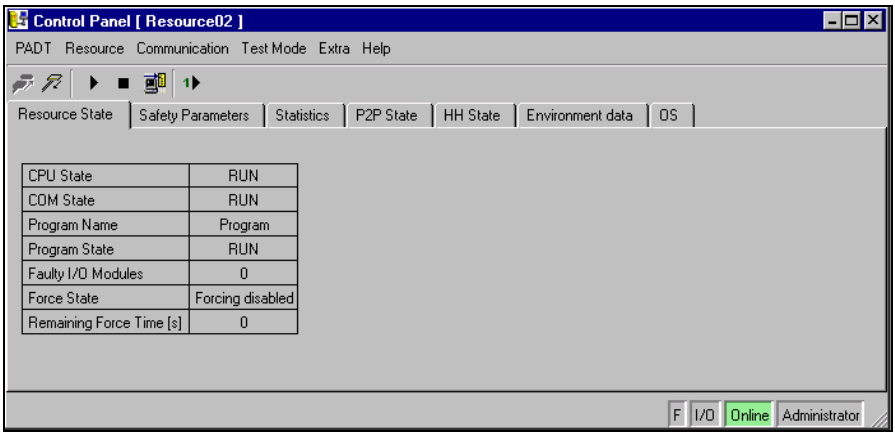


Fig. 143: Resource in RUN mode

Note: *The **Start**, **Stop** and **Download** functions can also be run from the "Resource" menu.*

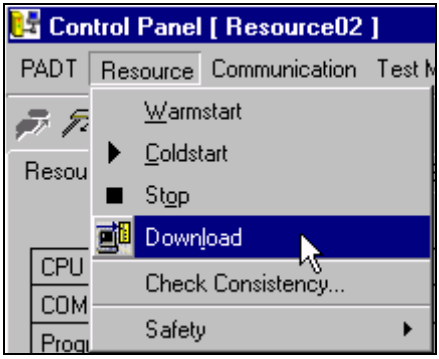


Fig. 144: «Resource» menu

A resource configuration which is loaded can not be uploaded back from the controller to the PC!
That is the reason why Archiving is very important. See therefore chapter 7.16.

7.13 The Force Editor

The Force Editor can just be used for monitoring the values of the signals or you can use it for forcing values to signals. The Force Editor is opened for an individual resource. All signals assigned to this resource are available in the Force Editor. The Force Editor is available in the menu Online of the Hardware Management.

Step 1: Open the Force Editor:

☐ Click **Online, Force Editor** in a “Resource” context menu.

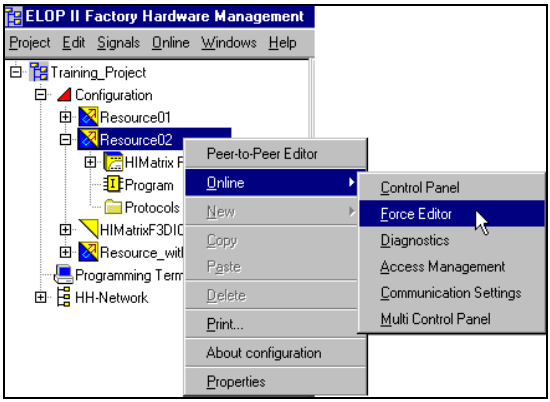


Fig. 145: Opening the Force Editor

☐ Login is required if there is no communication between the resource and the Programming Terminal. In case the Force Editor opens OFFLINE you can activate the Login by the menu function **Resource, Forcing Online**.

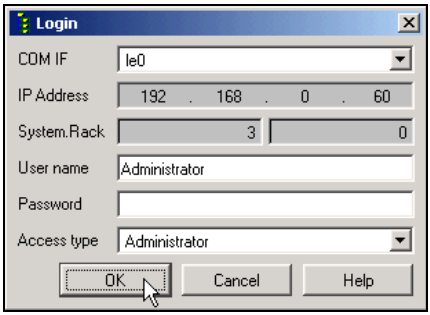


Fig. 146: Login dialog

Step 2: Reset the old Force settings:

Note: *Only execute the following step if Forcing is not active. Otherwise proceed as described in chapter 7.13.2.*

☐ Select the menu function **Resource, Clear Force Values on Resource**.

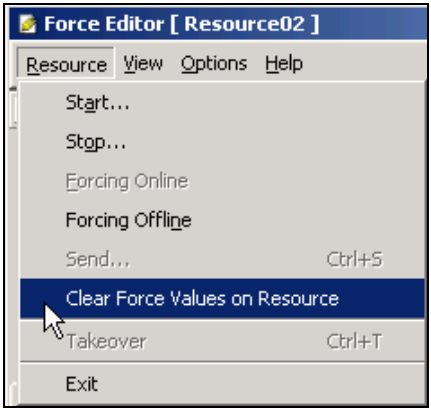


Fig. 147: Reset Force-Values

Step 3: Select signals for viewing in the Force Editor:

☐ Initially there are no signals in the Force Editor. Click **Configure** to open the "Select signals to view" window.

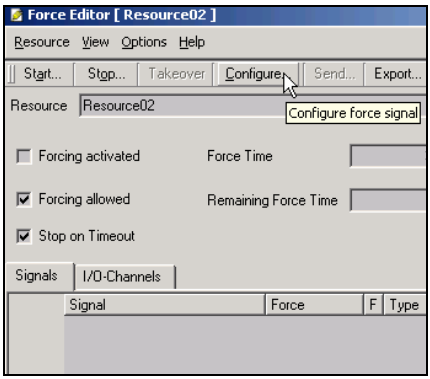


Fig. 148: Force Editor

In the window "Select signals to view“ you can choose signals, which you want to force or display in the Force Editor.

☐ Click the check boxes or **Select all**. Close with **OK**.

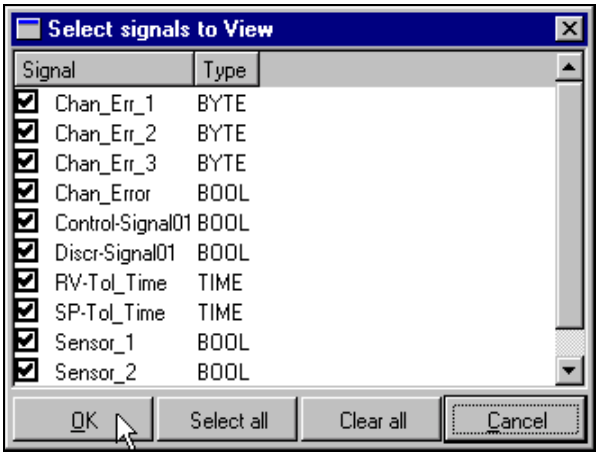


Fig. 149: Select signals

Step 4: Prepare for Forcing:

☐ Enter the desired Force value in the "Force“ column.

☐ In column "F“ specify whether the signal is to be forced (double-click).

	Signal	Force	F	Type	R-Value	R-Force	RF
1	Chan_Err_1	16#00		BYTE	16#00	16#00	
2	Chan_Err_2	16#00		BYTE	16#00	16#00	
3	Chan_Err_3	16#00		BYTE	16#00	16#00	
4	Chan_Error	FALSE		BOOL	FALSE	FALSE	
5	Control-Signal01	FALSE		BOOL	FALSE	FALSE	
6	Discr-Signal01	FALSE		BOOL	FALSE	FALSE	
7	RV-Tol_Time	T#0ms		TIME	T#0ms	T#0ms	
8	SP-Tol_Time	T#10s	✔	TIME	T#15s	T#0ms	
9	Sensor_1	1	✔	BOOL	FALSE	FALSE	
10	Sensor_2	FALSE		BOOL	FALSE	FALSE	
11	Sensor_3	FALSE		BOOL	FALSE	FALSE	

Fig. 150: Signal list in the Force Editor

Note: *Boolean values can be input as TRUE and FALSE or as "1“ and "0“.*
Note the unit in the case of time signals.

☐ The force values and the selection of the signals for forcing must be transferred to the controller. To do this click the **Send** button.

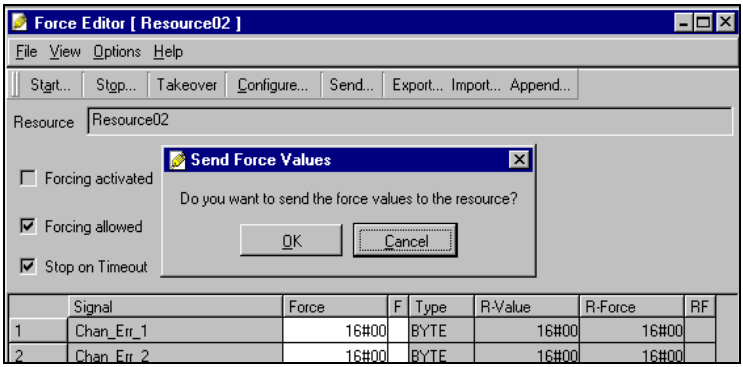


Fig. 151: Sending the force values

Overview of signals in the Force Editor after sending. Forcing is not activated yet!

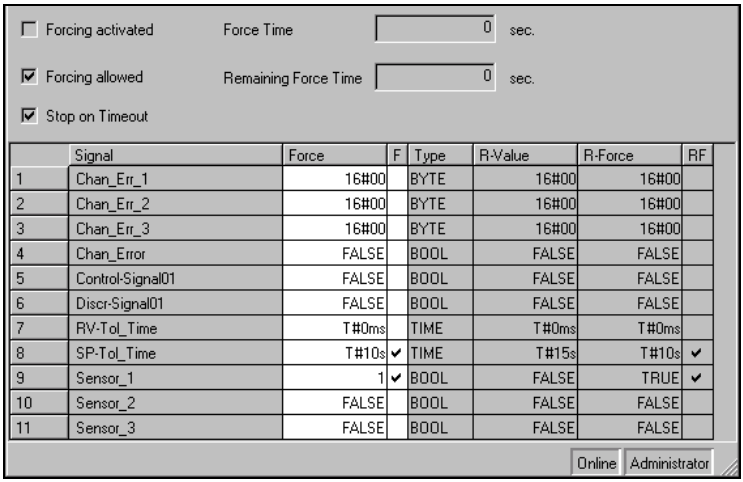


Fig. 152: List of signals for forcing

- The "R-Value“ column contains the value of the signal as derived from the process or the logic.
- The "R-Force“ column contains the value that replaces the R-Value during forcing.
- If the force value is active when forcing is started depends on whether there is a checkmark in the "RF" column.

Step 5: Start the Forcing:

☐ Pressing the **Start** button opens a prompt for the "Force Time“ in seconds.

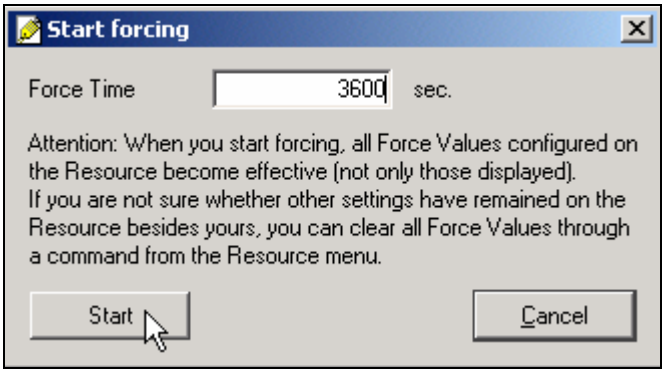


Fig. 153: Setting the Force time

Note: *On expiry of the Force Time the force value is replaced again by the R-Value. If "Stop on Timeout“ is activated in the resource properties, the controller returns to STOP after the forcing process.*

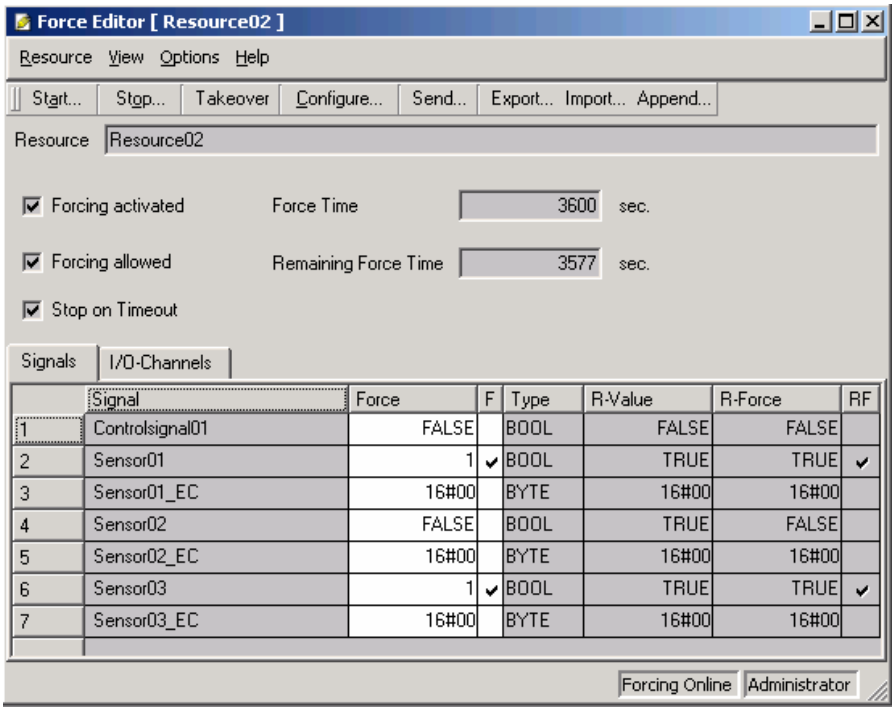


Fig. 154: Forcing activated

- Forcing can be stopped manually with the **Stop** button. In this case the controller remains in the RUN state, because the timeout was not reached.
- To be sure that all Force values are reset you should run the menu function **Clear Force Values on Resource**.

7.13.1 Saving and Loading Signal Selections

In order to keep track of the multiplicity of signals, a selection of signals can be stored and loaded again at a later time.

Step 1: Prepare the selection:

☐ Open the Force Editor and press **Configure** to select the signals that are to be saved.

Step 2: Save the selection:

☐ Press **Export** to save the selection under the desired name with the extension ".fdi“.

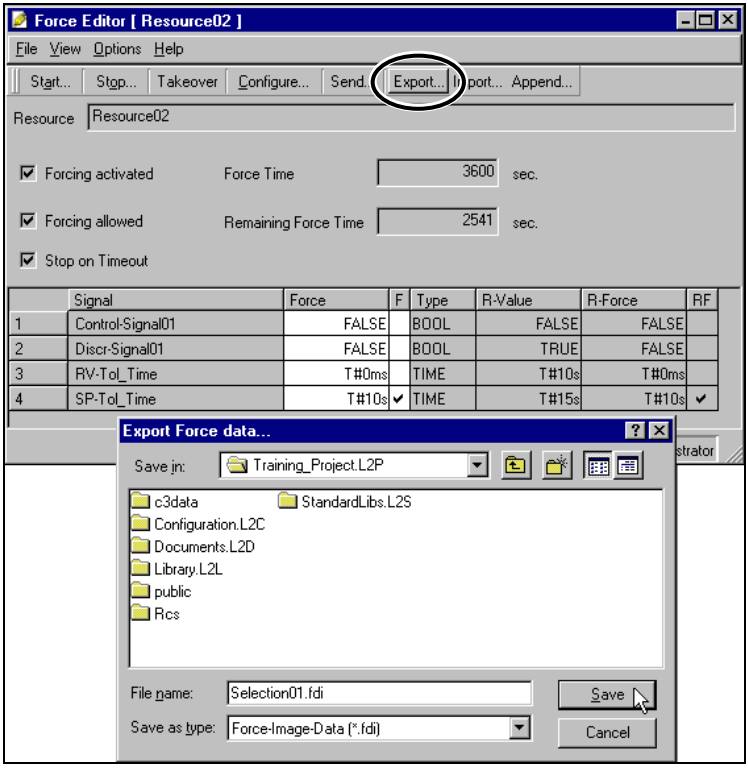


Fig. 155: Save signal selection (export)

Step 3: Import the saved signal selection:

- ☐ Press **Import** to open the "Import Force data...“ dialog box.
- ☐ Select the desired file and click **Open** or double-click the file directly.

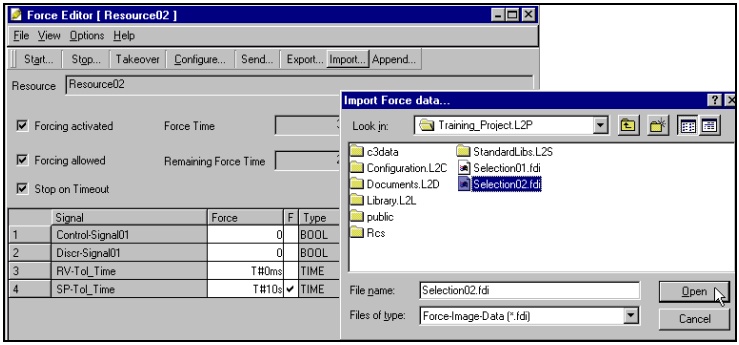


Fig. 156: Load signal selection (import)

Note: *Signals already existing in the Force Editor are replaced by the loaded selection.*

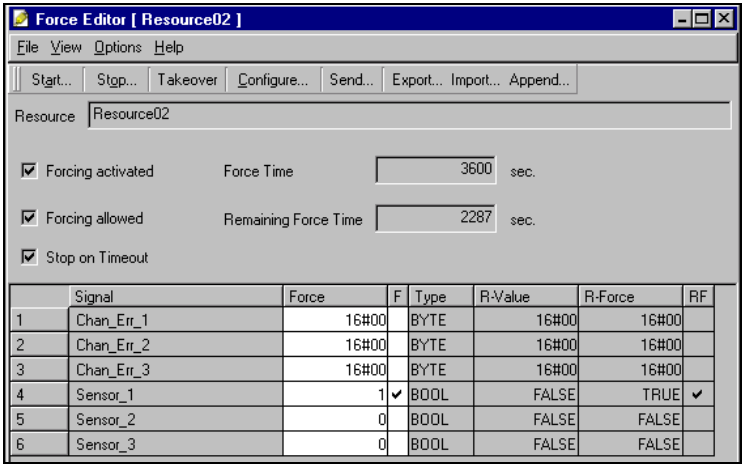


Fig. 157: Force Editor after loading the signal selection

Step 4: Append a saved signal selection:

- ☐ Press **Append** to open the "Append Force data...“ dialog box.
- ☐ Select the desired file and click **Open** or double-click the file directly.

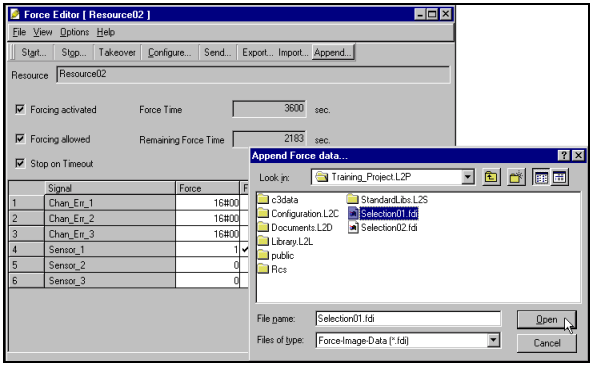


Fig. 158: Appending signal selection

The previous signals remain in the Force Editor and the newly loaded signals are appended to the signal list.

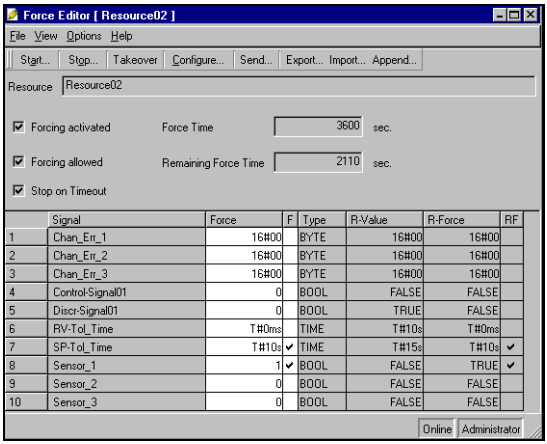


Fig. 159: Force Editor after appending the signal selection

Note: *With operating system ≥ 4 it is possible to display I/O channels in the Force Editor, which are not assigned to a signal. From the output only the diagnostic data are available.*

7.13.2 Starting Forcing of an already forced controller

If you go to a system for forcing and it is already forced, you should save the momentary Force Status.

Step 1: Open the Force Editor:

- ☐ Open the Force Editor as described in chapter 7.13.

Step 2: Save the momentary Force Status:

- ☐ Select **Configure, Select all** to mark all signals in the editor.
- ☐ Click **Takeover**, to get all force settings from the resource to the PC.

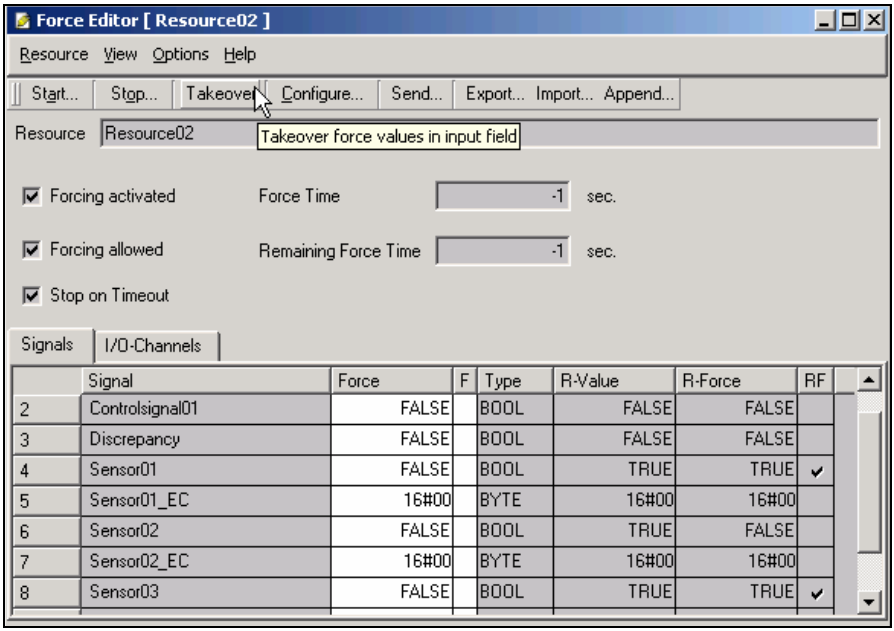
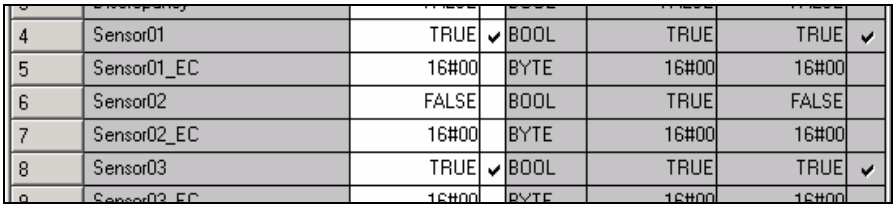


Fig. 160: Upload Force settings of the resource to the PC



4	Sensor01	TRUE	✔	BOOL	TRUE	TRUE	✔
5	Sensor01_EC	16#00		BYTE	16#00	16#00	
6	Sensor02	FALSE		BOOL	TRUE	FALSE	
7	Sensor02_EC	16#00		BYTE	16#00	16#00	
8	Sensor03	TRUE	✔	BOOL	TRUE	TRUE	✔

Fig. 161: Transferred values

- ☐ Save the settings with Export (Fig. 155 p.140).
- ☐ Now do the forcing like you want without resetting the already forced signals like in chapter 7.13 described.
- ☐ Before you leave the Force Editor import the saved Force Status with **Import** and send it to the controller. (Fig. 156 p.141)

7.14 Online Test (Power Flow)

The Online Test in the Project Management is used to monitor the values of the signals and variables within the logic plan while the resource is executing the application.

The Online Test is possible only if code is generated for all changes in the logic, the code is loaded into the controller and the controller is in RUN. The Control Panel of the resource must be open and communication must be established with the controller. The controller must have an operating system version ≥ 4.

Step 1: Open the Online Test:

- ☐ Expand the “Configuration” in the Project Management.
- ☐ Open the context menu of the “Resource” and click on **Online-Test**.

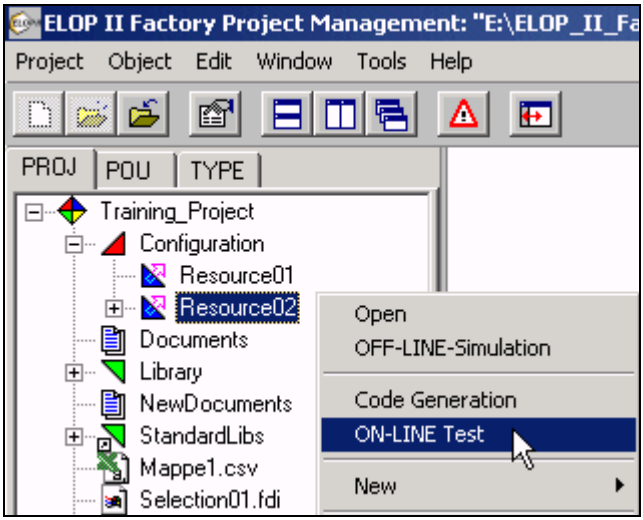


Fig. 162: Open ON-LINE Test

☐ Double-click on the program instance in the Online Test.

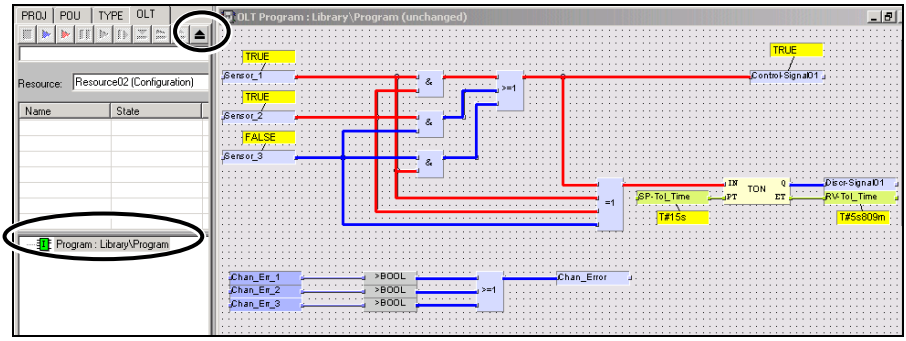


Fig. 163: Online Test

Note: *The values are displayed in Online test fields (OLT-Field). In the case of Boolean values the connection line has a colour code (blue = FALSE, red = TRUE). OLT-Fields can be added during the Online Test and saved in the project without any effect on the CRC-value by closing the Online Test.*

☐ Create a new OLT-Field with a right mouse click and select **Create OLT Field**.

☐ Use the «black arrow up» in the upper right corner of the left pane to exit the Online Test.

7.15 Documentation

7.15.1 Software Documentation

A printout of the function logic can be initialised and organised in a documentation object in the Project Management. All POUs (modules) are printed out with the documentation object. The hardware is documented separately in the Hardware Management.

Step 1: Create a new documentation object:

- ☐ Right-click on the project name.
- ☐ In the context menu select **New, Documents**.

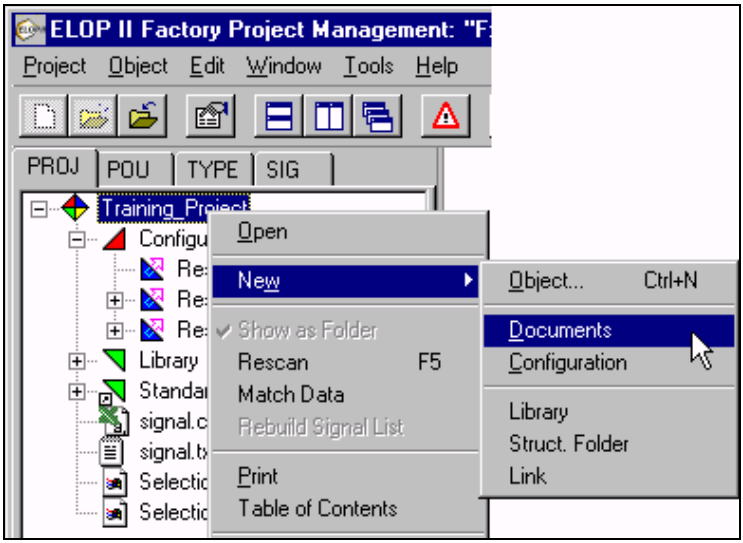


Fig. 164: Creating a documentation object

Step 2: Change the name of the new documentation:

☐ Slowly double-click on the name with the left mouse button. Change the name in the input field.

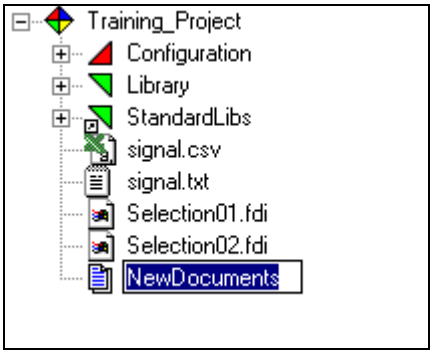


Fig. 165: Renaming document object

Step 3: Insert all data of your project into the documentation:

☐ Open the documentation by double-clicking it.

☐ Add the signal list from the Hardware Management.

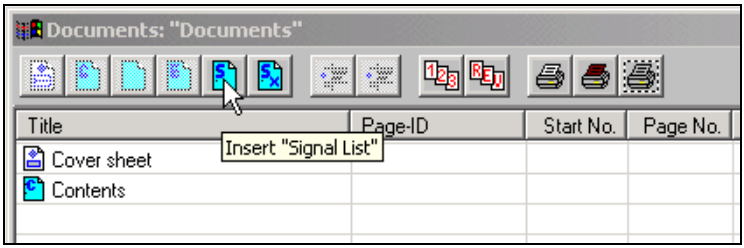


Fig. 166: Add signal list to the documentation

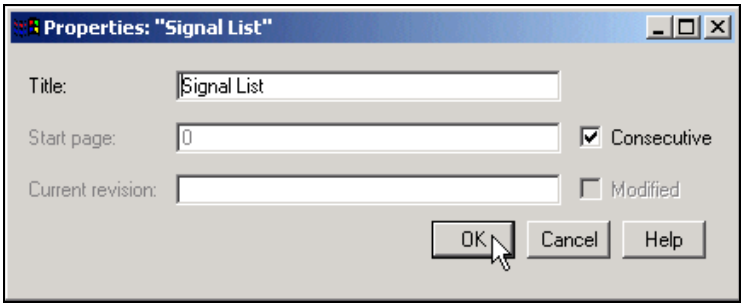


Fig. 167: Change name of the signal list

☐ Add the signal cross reference list from the Hardware Management.

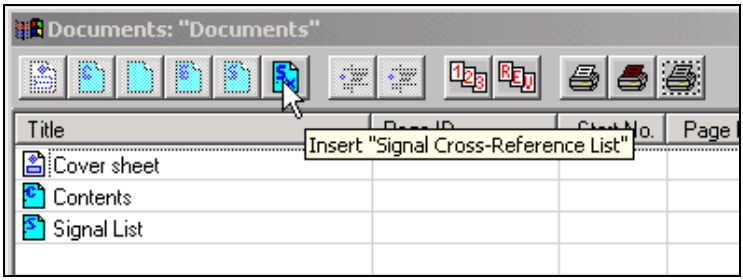


Fig. 168: Add signal cross reference list to the documentation

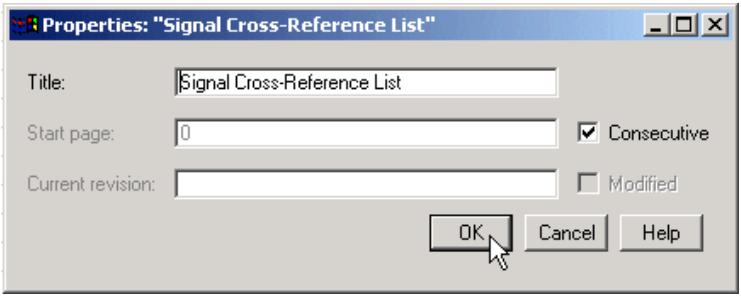


Fig. 169: Change name of the signal cross reference list

☐ Click on the “Project”, hold down the mouse button and drag the project into the documentation.
Now the documentation shows all elements that are contained in your project.

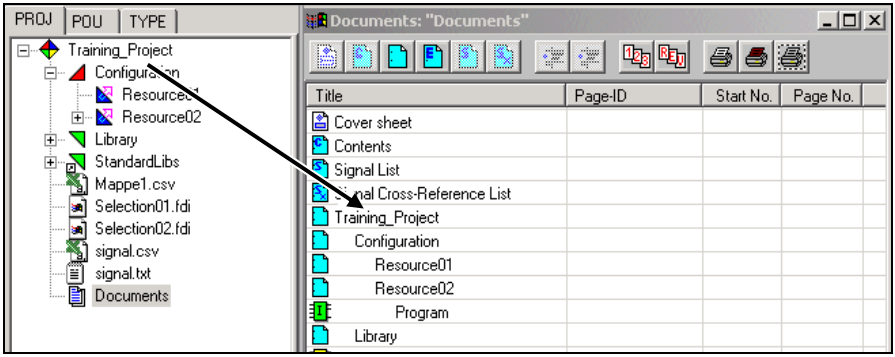


Fig. 170: Adding project to documentation

Note : You can also add single objects from your project to the documentation.

Step 4: Update the table of contents:

☐ Click **Update contents**  in the toolbar.

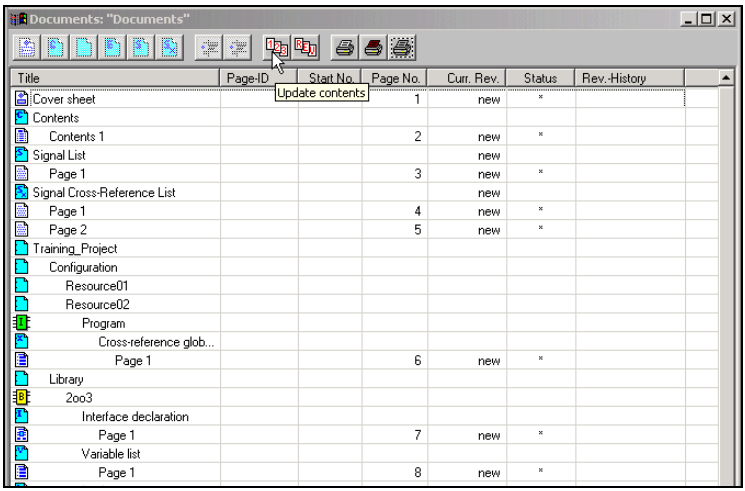


Fig. 171: Updating contents

After updating the contents the pages are listed with their pagination.
The sections in the documentation object correspond to the project folder and the library folders with the indents reflecting the hierarchy in the structure window.

Note : *You can change the order of the elements, or delete individual elements. Don't forget to update the table of contents.*

Step 5: Create a new revision status:

☐ Click **Create revision**  in the toolbar.

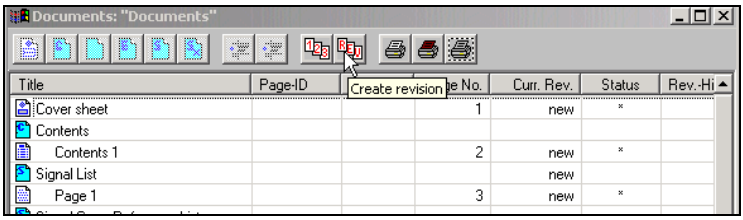


Fig. 172: Create new revision

☐ Enter a value for the revision status and click **Create**.

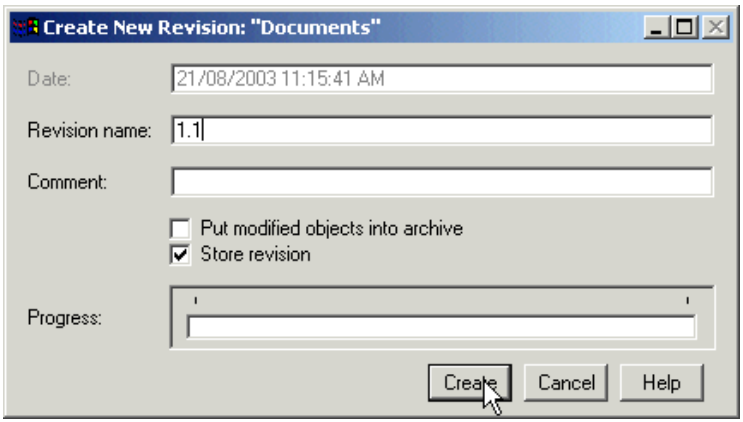


Fig. 173: Enter revision name (version)

Step 6: Update coversheet entries:

- ☐ Right click on the documentation and select **Properties** from the context menu.
- ☐ For the fields “Made by“, “Make date“, “Enduser 1“ and „Ordernumber“ enter values in the column “Value”. At least these fields should be filled in.

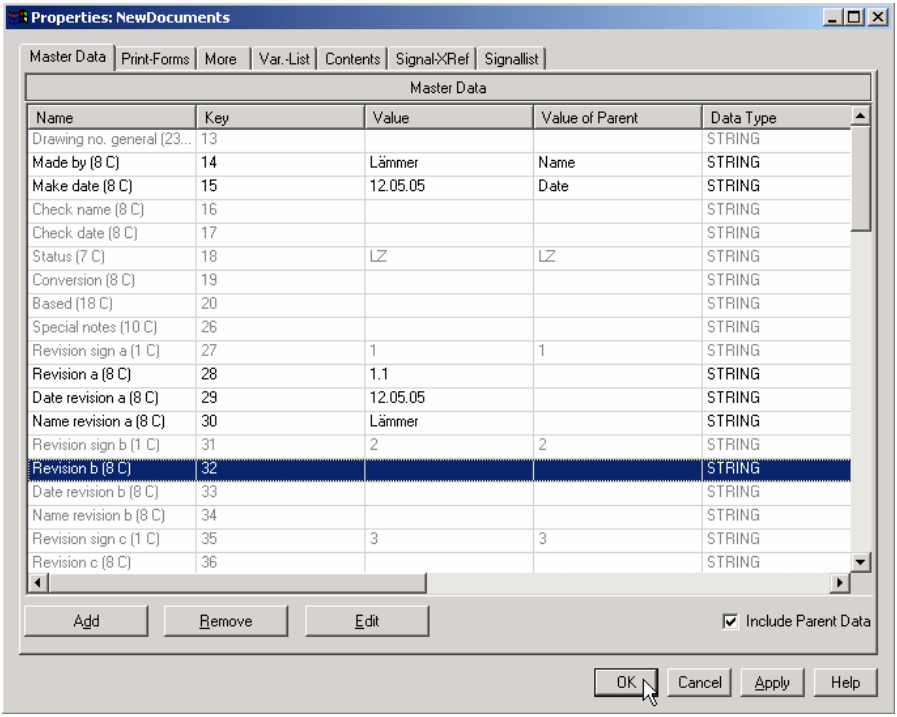


Fig. 174: Window for entering the cover sheet data

Note: *In case of changes please update the according fields.*

Step 7: Print the documentation:

☐ Start the print job for individual, all or modified pages from the context menu or with the buttons in the toolbar.

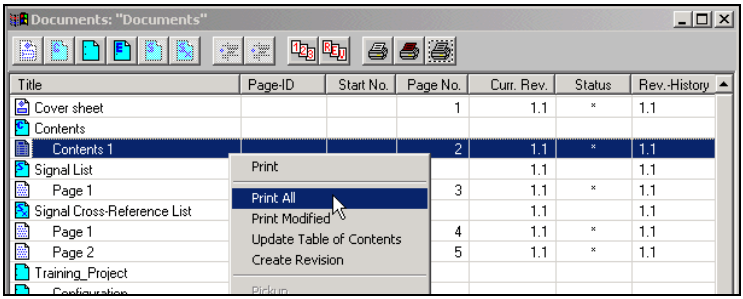


Fig. 175: Printing with the context menu

1. Print All
2. Print Modified
3. Print Selected Pages

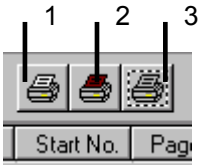


Fig. 176: Starting print

7.15.2 Hardware Documentation

The hardware documentation can be printed complete or only single elements can be printed.

7.15.2.1 Print Cross Reference List of Signals

A list of signals with their types of use is essential to allow easy tracking of the signals.

Step 1: Print the cross reference list:

☐ In the **Signals** menu click on **Print Cross References**.

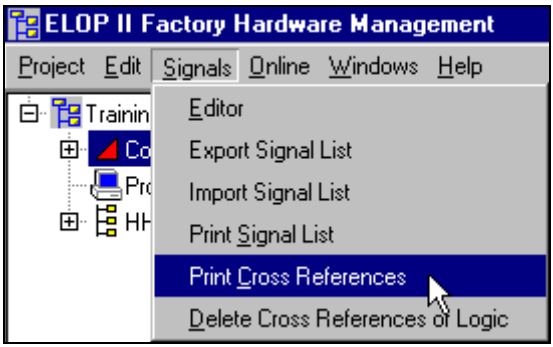


Fig. 177: Printing cross references

Note: *You can also print the signal list without cross references by selecting **Print Signal List** in the **Signals** menu.*

7.15.2.2 Print Resource Documentation

Note: *The resource documentation contains all hardware-relevant data, including I/O connections, Peer-to-Peer signals and other communication signals.*

Step 2: Print the resource documentation:

☐ Go to the Hardware Management.

☐ Select **Print...** in the context menu of a “Resource”.

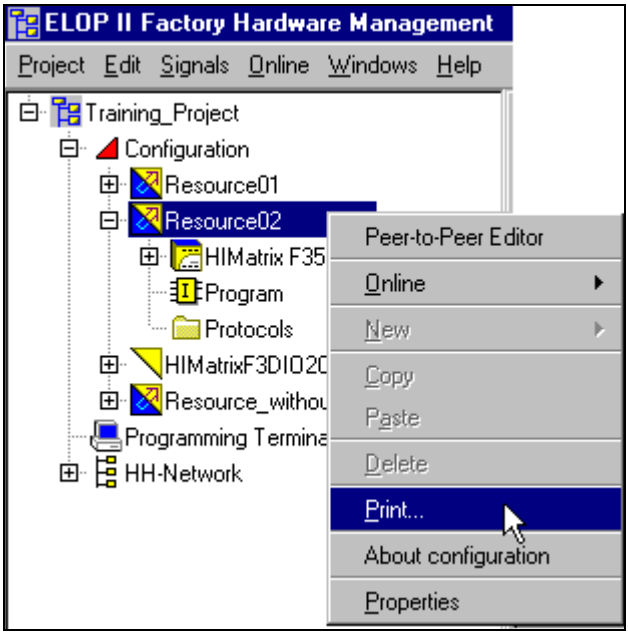


Fig. 178: Printing the resource documentation

Note: *You can start the print job for the hardware documentation for the resources altogether (see chapter 7.15.2.3).*

7.15.2.3 Print Hardware Management Documentation

Step 3: Print the Hardware Management documentation:

- ☐ In the Hardware Management click on **Project**.
- ☐ Select
  - **Print All...** to print the complete documentation,
  - **Print...** to print previously highlighted elements,
  - **Print Preview** to preview the documentation of a previously highlighted object,
  - **Print Preview All** for documentation of the complete project.

Fig. 179: Print Hardware Management documentation

7.16 Archiving

A project is archived in the Project Management.

A project should be archived when it has reached an important interim stage or if it has been downloaded into a controller.

A user program that has been downloaded into a controller cannot be uploaded into the Programming Terminal!
For this reason archiving is very important.

Step 1: Start the archiving:

- ☐ Right-click on the “Project”.
- ☐ In the context menu select **Archive...**

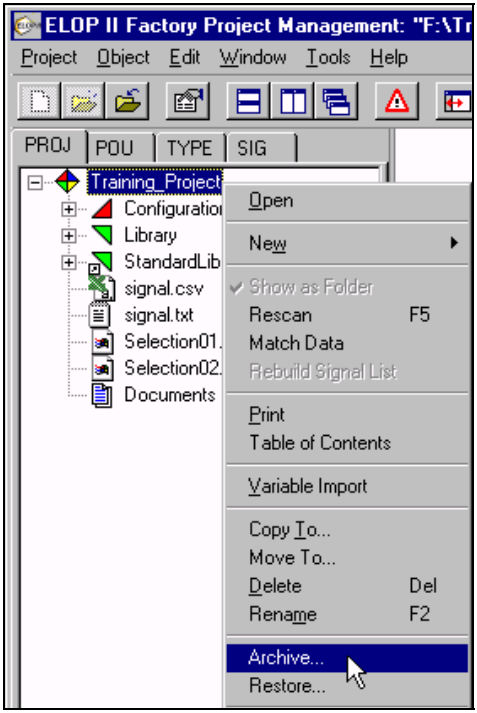


Fig. 180: Archiving a project

Step 2: Assign a name to the archive:

☐ In the “Archive” window (Fig. 181) specify the path where the project should be archived. In the "Target-file“ field enter a directory or press the **Browse...** button to select a directory in the dialog box (Fig. 182).

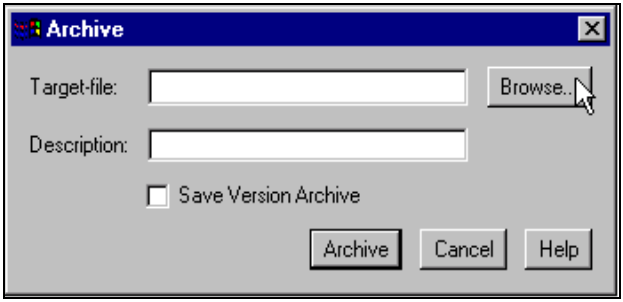


Fig. 181: Finding archive path

☐ Enter the name of the archive file in the "Object name“ field (without extension). It is recommended to include the date (Year_Month_Day) in the file name.

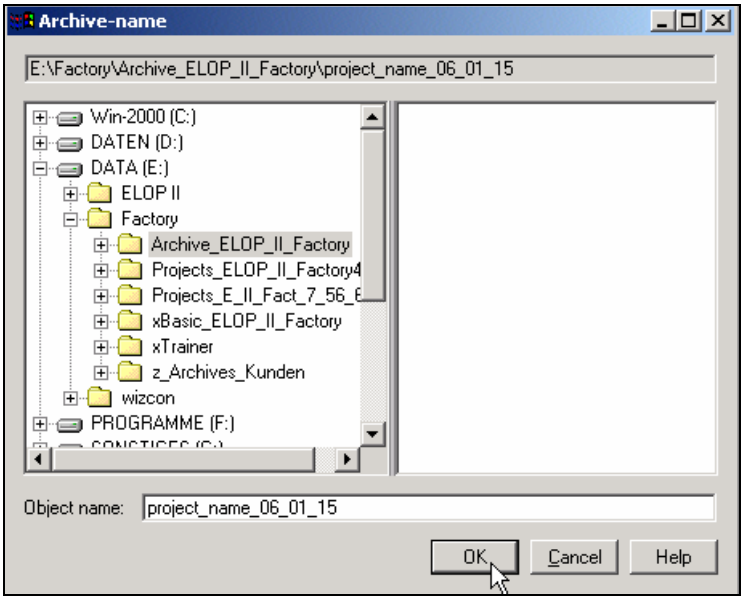


Fig. 182: Setting archive name

☐ Enter a comment for your project into the "Description" field.

☐ Click on **Archive**.

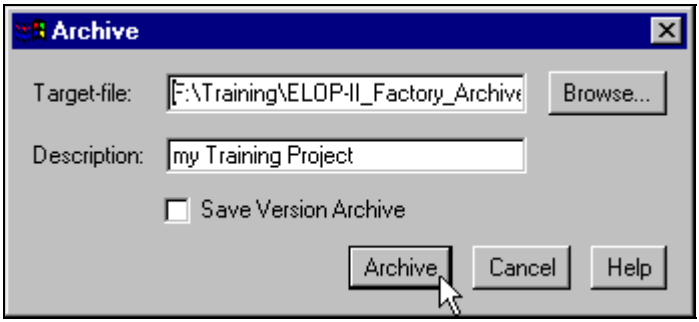


Fig. 183: Archive description

The project is archived in the specified directory and carries the extension ".L3P". The archive consists of three files.

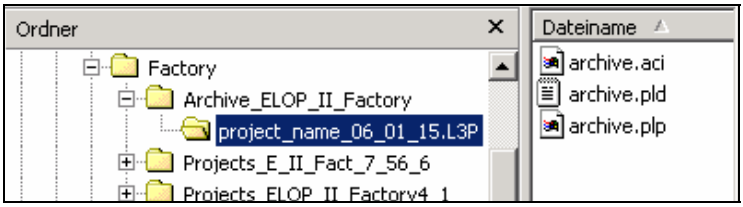


Fig. 184: Archived project

7.17 Restore

Note: *To be able to restore a project from an archive in ELOP II Factory, no other project must be open.*

Step 1: Start restoring a project:

☐ Click **Restore Project...** in the Project menu.

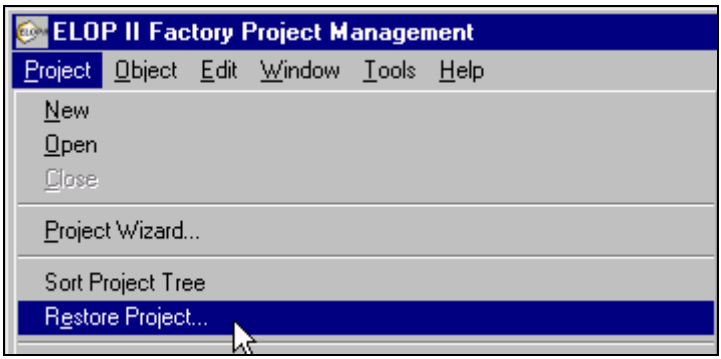


Fig. 185: Starting «Restore Project...»

Step 2: Select the project to be restored:

☐ In the “Restore” window specify the “Target-directory” where you want to restore the project. Use the **Browse...** button to select an existing directory.

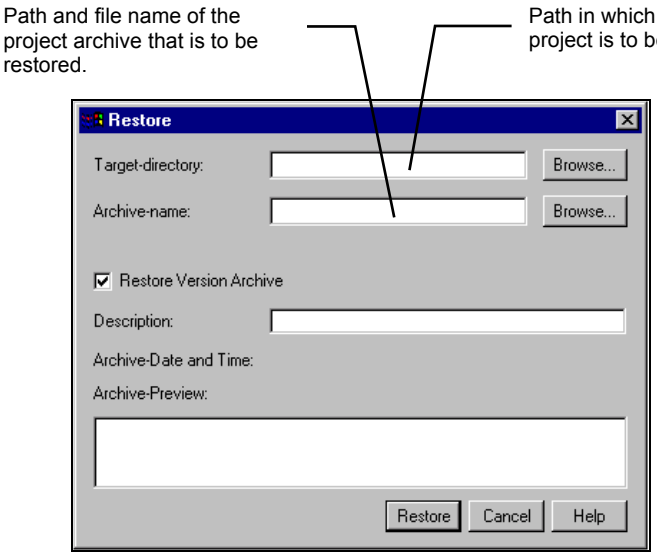


Fig. 186: Restoring archive

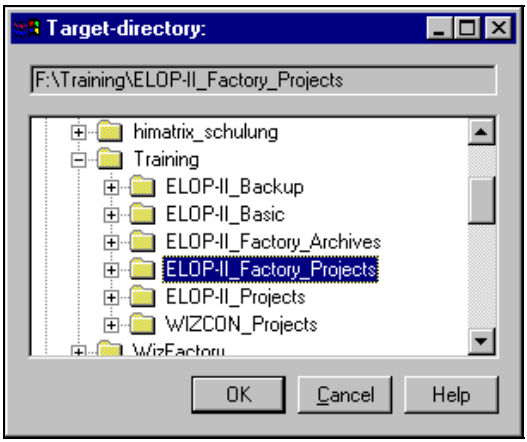


Fig. 187: Selecting target directory

- ☐ Click the **Browse...** button (for the archive name) to specify the path in a selection window (Fig. 188) by clicking with the mouse.
- ☐ Highlight the path for your archive in the left side of the dialog window. The right side only shows the files that are detected as archive.
- ☐ Highlight the desired archive.
- ☐ Click **OK** to finish the selection.

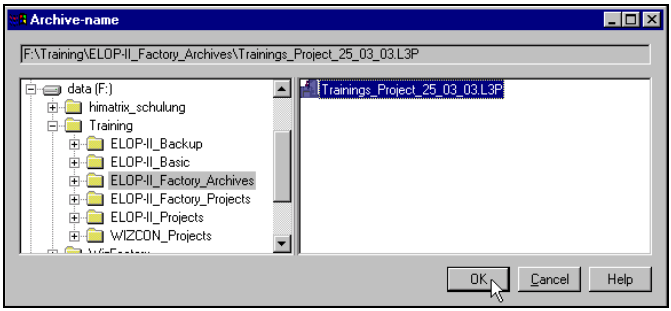


Fig. 188: Selecting archive

Step 3: Finish the restore:

- ☐ Click on the **Restore** button.

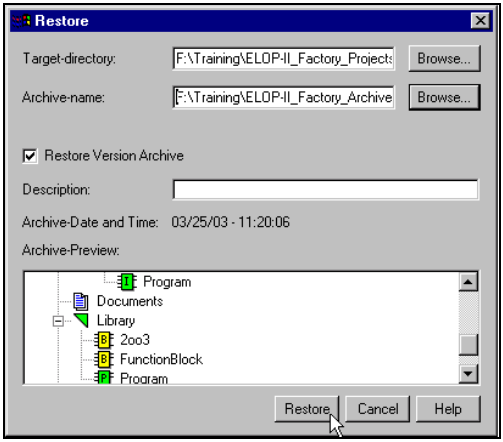


Fig. 189: Finishing restore

The project is displayed in the structure window after restoring is complete.

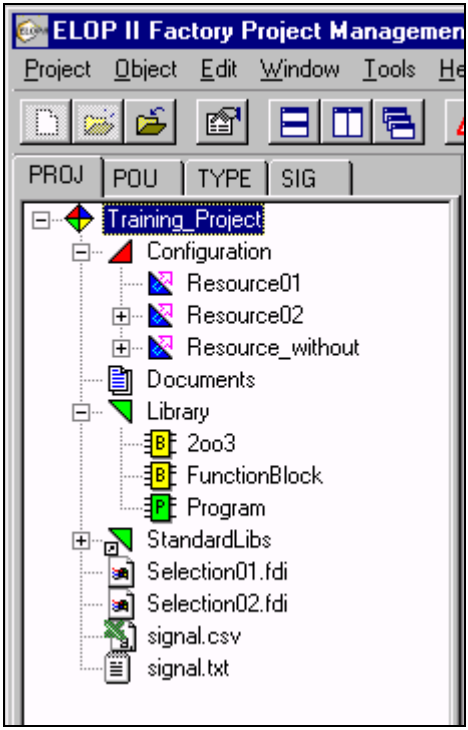


Fig. 190: Restored project

8 Appendix

8.1 Glossary

Block
Program organisation unit (POU), which is used and linked in the function block diagram editor. Blocks are in the standard library or in user-specific libraries.

Centred starting point
Method used by ELOP II Factory to display the function logic: The first page of the function plan is in the centre of a function plan of any desired theoretical size, which can be extended in all directions.

Drawing field
Area of the Function Block Diagram editor in which the logic is programmed.

Configuration
Term from the PES world. Resources that form a unit inside a project are saved in a configuration

Context menu
Menu that is displayed directly above the selected object when the right mouse button is pressed with the pointer on the object. The menu contains commands that can be applied to the object.

Data type
Defines the properties of the value range of a variable.

Document editor
Editor for collecting, structuring and printing program organisation units (POUs) and objects. Administers objects from the current project in an overall document.

Document management
Function integrated into the document editor with which various objects are collected to print these documents together and to integrate them into a common revision process.

Dongle
Relevant for full version, see also hardlock.

DXF

Drawing eXchange Format; data exchange format defined by the Autodesk company. Industry standard for exchanging drawings between different CAD systems.

Enable

Enables an ELOP II Factory function or a supplementary product in a hardlock using an individual signature that creates a serial number referring to every hardlock and transfers the number to it.

Error-State-Viewer

Area in the Project Management or Hardware Management in which the error and status messages from ELOP II Factory are output.

Focus

Navigation option in ELOP II Factory. The visible and displayed area in the Function Block Diagram editor can be centred on page view or the mouse pointer position. Used for fast navigation in the function logic.

Folder

Same meaning as directory. A folder can contain other folders and also file objects.

Font

Type and thickness of characters.

Format string

Element of an ELOP II Factory script language for documentation. It specifies a character string, the type and scope of comments or cross references and can include format instructions.

Function (FUN)

A program organisation unit (POU) of the FUNCTION type. In a function the initial states in every cycle are determined by the input states. (e.g. AND, OR).

Function block (FB)

A function block is a program organisation unit (POU) of the FUNCTION_BLOCK type. Function logic can be created in a function block. An FB can flag previous values (e.g. timer, flip-flop).

Function Block Diagram (FBD)

A programming language for describing networks with simultaneously operating Boolean, arithmetic and similar elements.

Function Block Diagram editor (FBD editor)
Editor used to create the logic in function blocks.

Hardlock
Hardware protection for the ELOP II Factory program package. A device that plugs into the parallel port of a computer. Required for operation of ELOP II Factory. It contains the access authorisation for product components of ELOP II Factory and supplementary products.

Hardlock driver
System software that enables communication with the hardlock.

Hardware Management
All hardware-based data and properties are processed here. Specifies resource types, defines signals and assigns the channels for the resources and specifies communication between the resources etc.

Toolbar
Bar with icons for fast access to commands.

Instance
Concrete use of a program organisation unit (POU) in a program. The program itself is also instanced for its use in a resource (see also program instance).

Interface declaration editor
Also block editor; area of the function block language editor in which the graphic view of a block is set.

IP address
Individual addressing of a PES or the Programming Terminal for communication

Link
Not a genuine data object but the definition of a path to an object (e.g. block library) that is not contained in the project or in this directory. A link to the default block library is automatically created in every project.

MAC address
Device-specific address assigned by the manufacturer and unique throughout the world. Used for initial communication with the device to allow project-specific settings to be made.

Maximise
Enlarges a window to its maximum size.

Menu bar
Horizontal bar that shows the names of all the menus.

Minimise
Shrinks a window to icon size.

Network
Significance in the IEC 61131-3 standard: all elements that are graphically linked.

Offline simulation
Program that enables the graphic test of the created program instance or program organisation unit: the logic is "animated“. This enables errors to be detected and corrected early.

Online
Functions that read data from the resources and download them into the resources. Downloads, starts and stops program. Tracks and forces signals etc.

Working area
Area in which the data object is edited with editors.

Overview window
Area of the Function Block Diagram editor that shows the logic pages of the Function Block Diagram editor in a minimised overview. In this window the focus can be placed on the position that shall be displayed in the drawing field of the editor with the mouse and keyboard. Used for fast and simple navigation.

Project
Folder object in which all other objects are contained. A project folder must be open to be able to work in the Project Management.

Project tree
Display of the structure in the form of a tree inside the project.

Project Management
1. The main ELOP II Factory program, which runs application-oriented. A project is created, managed, archived and restored with the Project Management.
2. Application window within which the project structure is shown and in which all editors are started with reference to logic design.

Program instance

A concrete use of a defined program type. A program instance executes the function that is specified in an associated program type in the controller of the resource.

Program type (PROG)

A program organisation unit (POU) of the PROGRAM type. The program type shows the highest level of a POU, i.e. it contains the complete logic, formed of functions and function blocks.

Quick Info

Short help text that is shown when the mouse pointer is positioned over a button.

Resource

Structuring element of IEC 61131-3, which corresponds to a central unit of the PES system. On Project Management side in a Resource object the program instance is created. On Hardware Management side the resource type is assigned and all other settings and assignments are made here.

Revision

Term from the ELOP II Factory document management. A revision is a tested or revised version of a document object referring to the total document. Different revisions can be created using the revision management.

Sequence Function Control SFC

A programming language for describing sequential and parallel processes in the function plan logic with time and event control (step chains).

Signal

A signal is used as an assignment specification between different areas of the complete controller. For example, the value of a variable in the program must be written to a hardware output. The assignment specification for transfer between the variables in the program and the hardware address correspond to the signal.

Signal editor

All signals are defined in the Signal Editor.

Status bar

Row at the bottom of the ELOP II Factory window that shows the status information.

Structure folder
Folder without special ELOP II Factory functions for structuring objects.

Structure window
Area consisting of multiple tabs and showing different views of the structure of the loaded project.

System ID (SRS)
The System ID (SRS = System-Rack-Slot) can be compared with the user number and can be used only once in the project. It can theoretically contain values from 1 to 65535.

Tab
Window element that shows the user associated information and selection options and simplifies navigation through different pages.

Template project
ELOP II Factory project that is installed with the program and contains the settings for a project. Every new project is created based on this template. The template project can be configured.

Title bar
Horizontal bar at the top of a window that shows the title of an applications including the objects being edited or the name of an open function.

Token Group
All HIMatrix controllers that exchange signals must be listed in Token Groups.

Variable
Designates a data memory that can store values that are specified by the data type and by settings during the variable declaration.

Variable declaration editor
Area of the Function Block Diagram editor in which the variables of the block are created and defined.

Variable import/export
ELOP II Factory function with which the variable lists from external files or databases (e.g. CSV files, Excel files, databases) can be imported into a project.

Zoom
Navigation option in ELOP II Factory. The visible and displayed are in the Function Block Diagram editor can be enlarged or reduced.

8.2 Index

8.3 Abbreviations

AC	**A**ttached **C**omment
CFG	**C**on**fig**uration (bus documentation)
CG	**C**ode **G**enerator
CONST	Constant
CRC	**C**yclic **R**edundancy **C**heck
CRF	**C**ross **Ref**erence (info on inputs and outputs)
CSV	Data format for import and export functions, ASCII format with comma "," separation character (**c**omma **s**eparated **v**alue)
D&D	**D**rag and **D**rop
DXF	Standard AutoCAD graphic format for print templates
ESV	Error-State-Viewer
FB	**F**unction **B**lock
FBD	**F**unction **B**lock **D**iagram
FBD	**F**unction **B**lock **L**anguage
FUN	**Fun**ction type block
GV	**G**lobal **V**ariable
HW	**H**ard**w**are
IL	**I**nstruction **L**ist
IO	**I**nput/**O**utput

OLS	**Offl**ine **S**imulation
OLT field	**On**l**i**ne **T**est Field
Peer-to-Peer	**P**eer-**to**-**P**eer communication
PADT	**P**rogramming **a**nd **D**ebugging **T**ool (PC)
PES	**P**rogrammable **E**lectronic **S**ystem (controller)
POU	**P**rogram **O**rganisation **U**nit (block)
RES	**Res**ource
RETAIN	Latching properties
Remote I/O	**R**emote **I**/**O**, controller with no user program
SL	**S**equence **L**anguage
SRS	**S**ystem-**R**ack-**S**lot

HIMA

...the safe decision

HIMA Paul Hildebrandt GmbH + Co KG

Industrial Automation

Postbox 1261 68777 Brühl

Telephone: (06202) 709-0 Telefax: (06202) 709-107

Email: info@hima.com Internet: www.hima.com

(0622)