XPSMFWIN
Software Manual

07/2007

33003788.02

a brand of Schneider Electric

Telemecanique

Table of Contents

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of this symbol to a Danger or Warning safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER
DANGER indicates an imminently hazardous situation, which, if not avoided, **will result** in death or serious injury.

WARNING
WARNING indicates a potentially hazardous situation, which, if not avoided, **can result** in death, serious injury, or equipment damage.

CAUTION
CAUTION indicates a potentially hazardous situation, which, if not avoided, **can result** in injury or equipment damage.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

About the Book

At a Glance

Document Scope

This manual is a quick and easy start for learning XPSMFWIN. It includes an overview of all functions and step-by-step instructions on creating a project.

Validity Note

This documentation is valid for XPSMFWIN under Microsoft Windows XP service pack or Windows 2000 professional with service pack 1 or higher.

Schneider electric assumes no responsibility for any errors that may appear in this document. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

The data and illustrations found in this documentation are not binding. We reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be construed as a commitment by Schneider Electric.

Related Documents

Title of Documentation	Reference Number
Safety Suite V2 Installation Instruction	33003529
XPSMF•• Manuals	

You can download these technical publications and other technical information from our website at *www.telemecanique.com*.

Product Related Warnings

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When controllers are used for applications with technical safety requirements, please follow the relevant instructions.

WARNING

UNINTENDED EQUIPMENT OPERATION

Failure to use Schneider Electric software or Schneider Electric approved software with Schneider Electric hardware products may result in injury, harm, or improper operation.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

User Comments

We welcome your comments about this document. You can reach us by e-mail at techpub@schneider-electric.com

General Overview

1

At a Glance

Overview

This chapter gives a general overview of the XPSMFWIN Software Manual.

What's in this Chapter?

This chapter contains the following topics:

Shipping List

XPSMFWIN Package

The SSV1XPSMFWIN packaging includes:

Content	Description
This manual	The manual is a quick and easy start for learning XPSMFWIN. It includes an overview of all functions and step-by-step instructions on creating a project.
CD-ROM	The CD-ROM contains the Safety Suite V1 software with XPSMFWIN, some auxiliary programs and the complete documentation for the current Preventa systems.
USB Hardlock (Dongle)	The dongle administers the licence (protection against unauthorised use) of the protected XPSMFWIN software.

Note: Within this document all references to the ELOP II Factory Software are equivalent to the XPSMFWIN Software.

Note: In this document the Preventa XPSMF** Safety PLC are referred to by only the end of the product code. For instance XPSMF3022 will be named F30.

References of Safety PLCs and Remote I/O Modules

Compact Safety PLCs

Schneider Electric Reference	Reference in Software
XPSMF3022	HiMatrix F30
XPSMF31222	HiMatrix F31
XPSMF3502	HiMatrix F35
XPSMF3522	
XPSMF3542	
XPSMF4000	XPSMF40
XPSMF4002	
XPSMF4020	
XPSMF4022	
XPSMF4040	
XPSMF4042	

Modular Safety PLCs

Schneider Electric Reference	Reference in Software	Remark
XPSMFCPU22	XPSMF60	modular safety PLC
XPSMFAI801	AI801 FSI000 AI801 FS2000	I/O card
XPSMFAO801	AO801 FSI000 AO801 FS2000	I/O card
XPSMFCIO2401	CIO2/401	I/O card
XPSMFDI3201	DI3201	I/O card
XPSMFDIO241601	DIO24/1601	I/O card
XPSMFDO801	DO801	I/O card
XPSMFDI2401	DI2401	I/O card

Remote I/O Modules

Schneider Electric Reference	Reference in Software
XPSMF1DI1601	HiMatrix F1 DI 1601
XPSMF2DO401	HiMatrix F2 DO 401
XPSMF2DO801	HiMatrix F2 DO 801
XPSMF2DO1601	HiMatrix F2 DO 1601
XPSMF2DO1602	HiMatrix F2 DO 1602
XPSMF3DIO8801	HiMatrix F3 DIO 8/801
XPSMF3DIO16801	HiMatrix F3 DIO 16/801
XPSMF3DIO20802	HiMatrix F3 DIO 20/802
XPSMF3AIO8401	HiMatrix F3 AIO 8/401

Information on this Manual

Overview

You will find useful information in this manual to help you to learn the most important functions of XPSMFWIN.

General Structure

The manual is structured in such a way to help you to familiarize yourself with the software with a minimal amount of effort.

The general structure of the manual is:
- safety relevant information
- use of manual
- contents of manual
- structure of programming environment
- project management - working area
- hardware management - working area
- step-by-step project configuration for new users
- step-by-step project configuration for advanced users
- additional features for programming areas

How to Program Safety

2

At a Glance

Overview

This chapter describes how to program safety.

What's in this Chapter?

This chapter contains the following topics:

Introduction

Overview

Systems comprised of electrical and/or electronic components have been used for many years to perform safety functions in most application sectors. Computer-based systems (generically referred to as programmable electronic systems (PESs) are being used in all application sectors to perform non-safety functions and, increasingly, to perform safety functions.

If computer system technology is to be effectively and safely exploited, it is essential that those responsible for making decisions have sufficient guidance on the safety aspects on which to make those decisions.

Programming Safety

To understand the procedure of programming safety related systems entails knowing the EN/IEC 61508 standard. Within this section a basic overview is outlined from the EN/IEC 61508-3 standard. For more information regarding each process please refer to *EN/IEC 61508*.

Any software forming a part of a safety-related system, or used to develop a safety-related system is termed safety-related software.

Safety-related software includes:
- operating systems
- system software
- software in communication networks
- human-computer interface functions
- support tools
- firmware
- application programs

Application programs include high level programs, low level programs and special purpose programs in limited variability languages (*see 3.2.7 of IEC 61508-4*).

Requirements

Overview

The functional safety planning shall define the strategy for the software procurement, development, integration, verification, validation and modification to the extent required by the safety integrity level of the Electrical/Electronic/ Programmable Electronic (E/E/PE) safety-related system.

Software configuration management should regard the aspects described in the following.

Software Changes

Software configuration management should apply administrative and technical controls throughout the software safety lifecycle, in order to manage software changes and thus ensure that the specified requirements for software safety continue to be satisfied.

Software Safety Integrity

Software configuration management should guarantee that all necessary operations have been carried out to demonstrate that the required software safety integrity has been achieved.

Configuration Items

Software configuration management should maintain accurately and with unique identification all configuration items which are necessary to maintain the integrity of the E/E/PE safety-related system.

Configuration items include at least the following:
- safety analysis and requirements
- software specifications and design documents
- software source code modules
- test plans and results
- pre-existing software components and packages which are to be incorporated into E/E/PE safety-related system
- all tools and development environments which are used to create or test, or carry out any action on, the software of the E/E/PE safety-related system

Change-Control Procedures

Software configuration management should apply change-control procedures to
- prevent unauthorized modifications,
- document modification requests,
- analyse the impact of a proposed modification,
- approve or reject the request,
- document the details of, and the authorization for, all approved development,
- to document the (partial) integration testing which justifies the baseline,
- guarantee the composition of, and the building of, all software baselines (including the rebuilding of earlier baselines).

Note: Management decision and authority is needed to guide and enforce the use of administrative and technical controls.

Documentation

Software configuration management should document the following information to permit a subsequent audit:
- configuration status
- release status
- the justification for and approval of all modifications
- the details of the modification

Master Copies

Software configuration management should formally document the release of safety-related software. Master copies of the software and all associated documentation should be kept to permit maintenance and modification throughout the operational lifetime of the release software.

References

Please refer to the EN/IEC 65108 for a detailed understanding ensuring that you use the XPSMFWIN software package according to the Safety Management procedures.

Installation

3

What is installed from Safety Suite?

Overview

Safety Suite V1or V2 installs the following components on your PC:

- XPSMFWIN
- USB hardlock driver
- certified function blocks
- safety PLC user documentation

Along with the Safety Suite V1or V2 the following complete software package are delivered:

- ASISWIN
- XPSMCWIN
- P.A.D. (protected area design)

Introduction to XPSMFWIN

4

At a Glance

Overview

This chapter describes how to start the XPSMFWIN software and the overall layout of XPSMFWIN programming environment.

What's in this Chapter?

This chapter contains the following topics:

Starting XPSMFWIN

Overview

XPSMFWIN is a program with numerous functions, but the intuitive Windows style user operation makes the software simple to use.

Start Program

Step	Action	Result
1	Insert USB hardlock onto PC	-
2	Select **Start** → **Program** → **Schneider Electric** → **Safety Suite**.	safety suite opens
3	Select **XPSMFWIN** from the Safety Suite list of programs	XPSMFWIN comprises two windows. ● **Project Management** for creating all user programs and for archiving or restoring projects. ● **Hardware Management** for defining all hardware-specific data. **Hardware Management** is only opened when a project is created new or opened.
4	Use project wizard to create a new project or open an existing project	-

General Layout of Programming Environment

Overview

The programming environment is composed of 2 windows.
- Project Management
- Hardware Management

Project Management

The **Project Management** window is used for:
- creating resources
- creating user programs
- creating user function blocks
- offline simulation
- online testing
- creating and managing user documentation
- archiving and restoring projects
- code generating

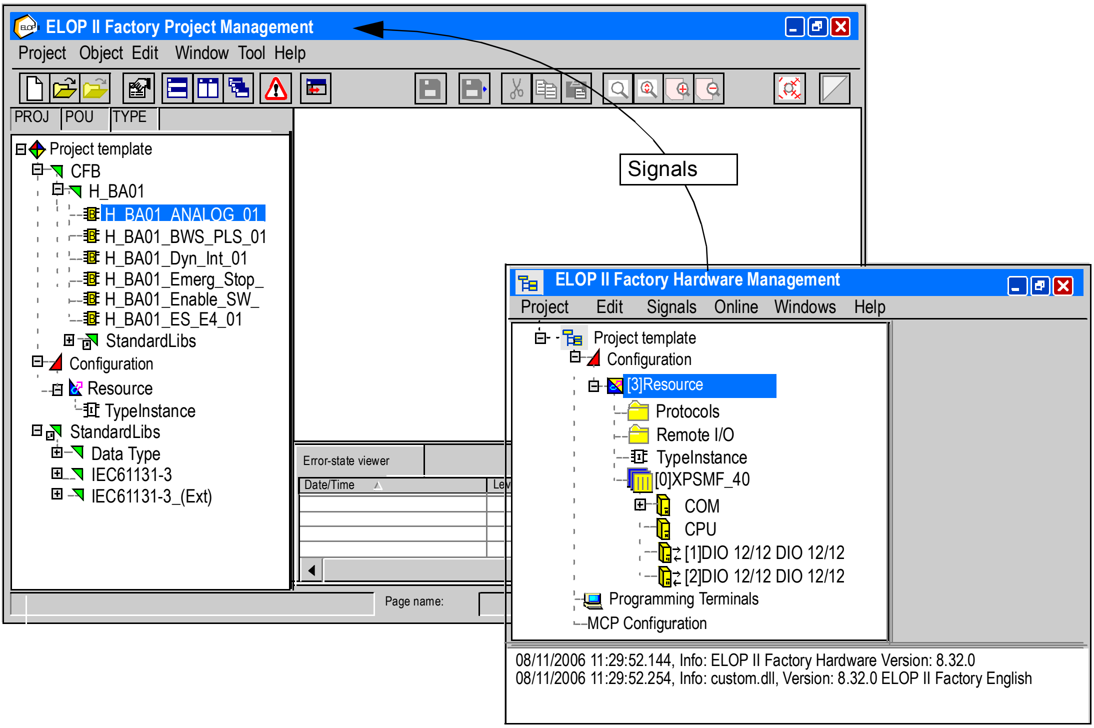
Hardware Management

The **Hardware Management** window is used for:
- viewing program size
- assigning resources PLC types
- assigning I/O modules to resources
- defining hardware inputs and outputs
- setting network parameters
- assigning non-safety protocols e.g. Modbus
- assigning signals
- defining diagnostics
- online control panel
- defining user parameters (user administration)
- downloading to hardware

Note: Within the programming environment please take note that short cuts e.g `Ctrl+a` only work in lower case letters.

Presentation

The following diagram provides a main overview of the 2 windows and how they work together.



Project Management

5

At a Glance

Overview

This chapter provides information on the Project Management.

What's in this Chapter?

This chapter contains the following sections:

5.1 Introduction

Project Management

Overview

The **Project Management** is the organization center for working with XPSMFWIN.

The project folder is shown in first position in the structure window (see also diagram *Structure Window (Project Management), p. 40*, 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).
- create resources
- create user programs
- create user function blocks
- simulate offline
- test online (must be connected to safety PLC)
- create and manage user documentation
- generate code of a project

An open project is displayed in the structure window.

Template Project

You can make numerous presets for design purposes with the help of the template project, which can be opened from the menu bar (when all projects are closed).

5.2 Screen Layout of the Project Management

At a Glance

Overview

This section describes the screen layout of the Project Management.

What's in this Section?

This section contains the following topics:

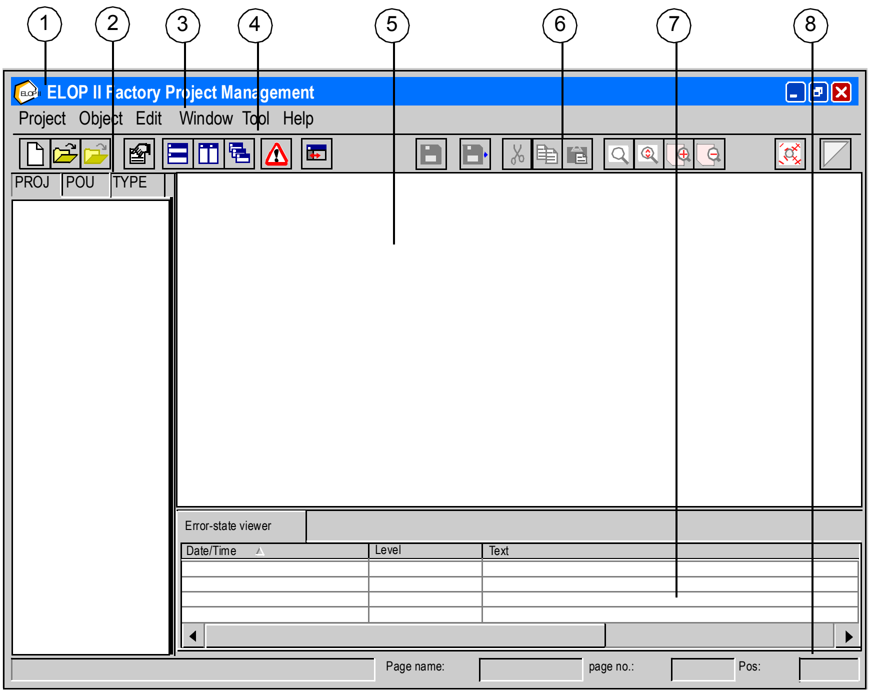
Screen Layout (Project Management)

Overview

After XPSMFWIN is started the screen layout appears as shown in the diagram below.

Representation

The screen layout (project management) includes the following components:



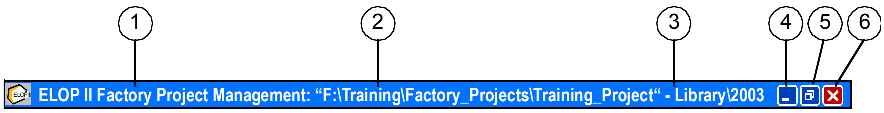
1 Title Bar
2 Structure Window
3 Menu Bar
4 Toolbar for Project Management
5 Working Area
6 Toolbar for the Function Block Diagram Editor (FBD Editor)
7 Error State Viewer
8 Status Bar with Coordinate Information of the Function Plan Editor

Title Bar (Project Management)

Overview

As well as the standard functions for minimizing, maximizing and closing the window, the title bar also includes information on the project.

Representation



1 Program (ELOP II Factory)
2 Open Project
3 Open Function Block
4 Minimize
5 Maximize or Overlap
6 Close

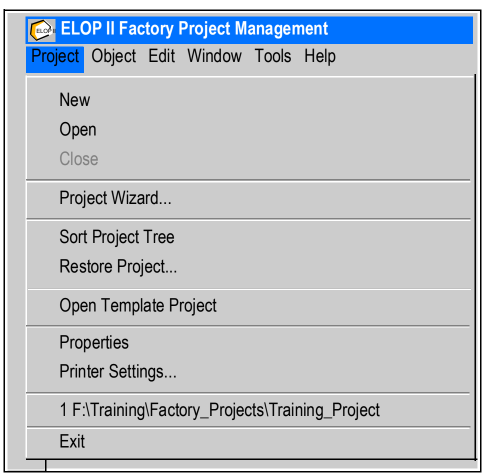
Menu Bar (Project Management)

Overview

The menu bar provides additional user options. Most XPSMFWIN functions are available in the menu bar.

Representation

Project menu (project management)



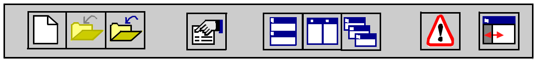
Toolbar (Project Management)

Overview

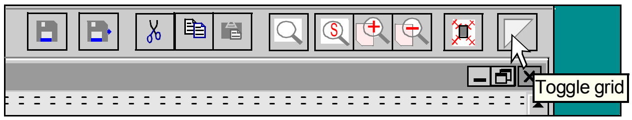
The toolbar is displayed below the menu bar and divided into the two sections **Project Management** and **Function Block Diagram Editor**.

Representation

Toolbar for the project (project management)



Toolbar for the function block diagram editor (project management)



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

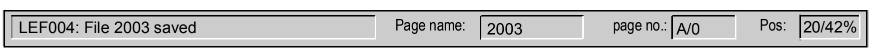
Status Bar (Project Management)

Overview

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.

Representation

Status bar



Error State Viewer (Project Management)

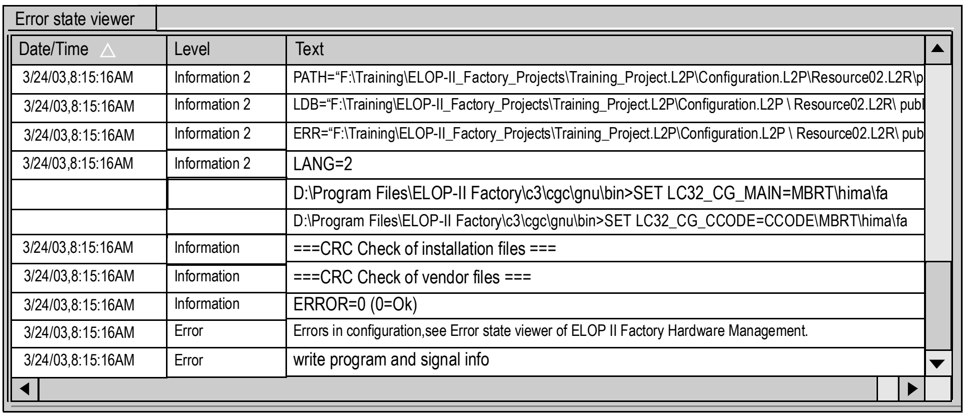
Overview

The error state viewer is the central point for displaying error and status messages.

The occurrence of a new message is signalled by a flashing  icon in the Windows task bar.

The meaning of program messages can be found within the online help from within the Project Management window under **Help Topics** → **List and References** → **Program Messages** → **Occurence of Program Messages**.

Representation



Error state viewer

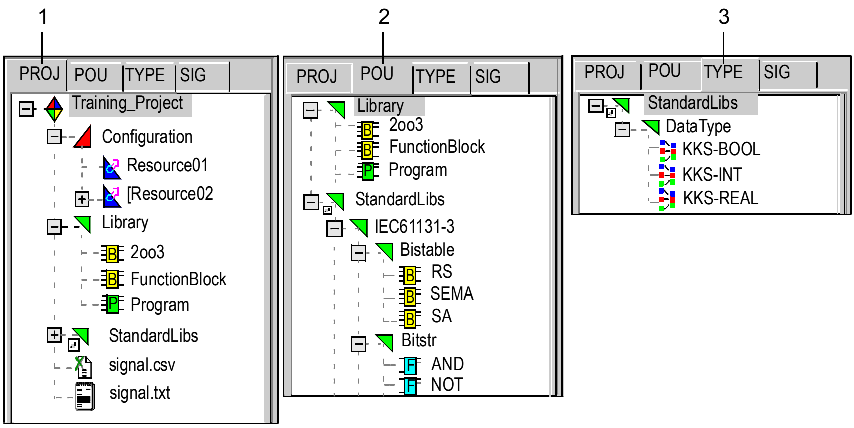
Date/Time	Level	Text
3/24/03,8:15:16AM	Information 2	PATH="F:\Training\ELOP-II_Factory_Projects\Training_Project.L2P\Configuration.L2P\Resource02.L2R\p
3/24/03,8:15:16AM	Information 2	LDB="F:\Training\ELOP-II_Factory_Projects\Training_Project.L2P\Configuration.L2P \ Resource02.L2R\ publ
3/24/03,8:15:16AM	Information 2	ERR="F:\Training\ELOP-II_Factory_Projects\Training_Project.L2P\Configuration.L2P \ Resource02.L2R\ pub
3/24/03,8:15:16AM	Information 2	LANG=2
		D:\Program Files\ELOP-II Factory\c3\cgc\gnu\bin>SET LC32_CG_MAIN=MBRT\hima\fa
		D:\Program Files\ELOP-II Factory\c3\cgc\gnu\bin>SET LC32_CG_CCODE=CCODE\MBRT\hima\fa
3/24/03,8:15:16AM	Information	===CRC Check of installation files ===
3/24/03,8:15:16AM	Information	===CRC Check of vendor files ===
3/24/03,8:15:16AM	Information	ERROR=0 (0=Ok)
3/24/03,8:15:16AM	Error	Errors in configuration,see Error state viewer of ELOP II Factory Hardware Management.
3/24/03,8:15:16AM	Error	write program and signal info

Structure Window (Project Management)

Overview

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

Representation



1 Complete Project
2 All POUs, Program Organization Unit (Libraries)
3 All Data Types

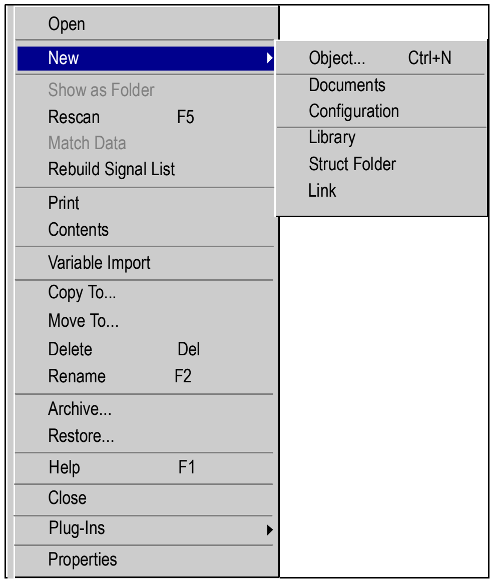
Context Menu for Objects (Project Management)

Overview

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.

Representation

Context menu (project management)



Working Area (Project Management)

Overview

In the working area (see *Screen Layout (Project Management), p. 36*) data objects are edited using:

- function block diagram editor
- document editor

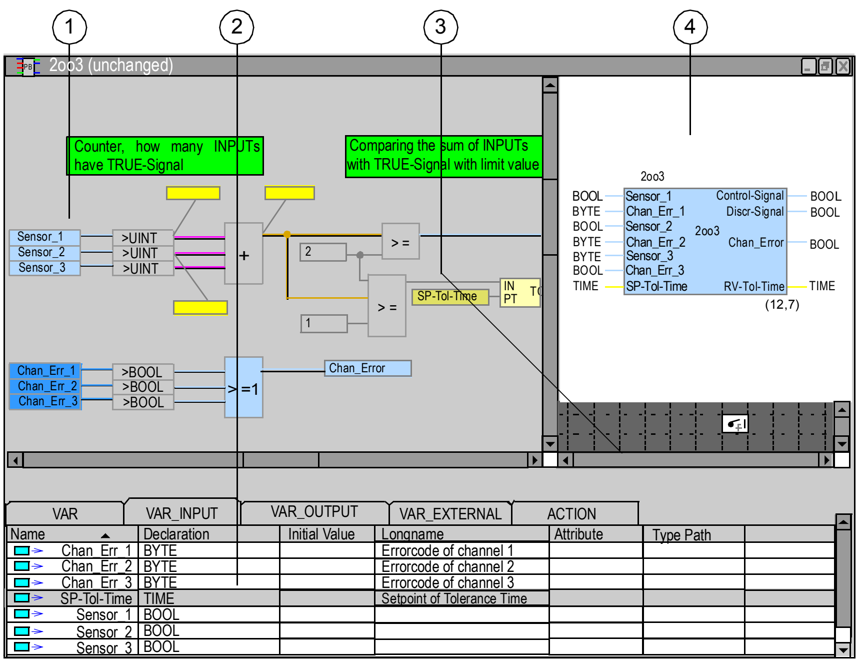
Function Block Diagram (FBD) Editor (Project Management)

Overview

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

Representation

The FBD editor comprises the following panes:



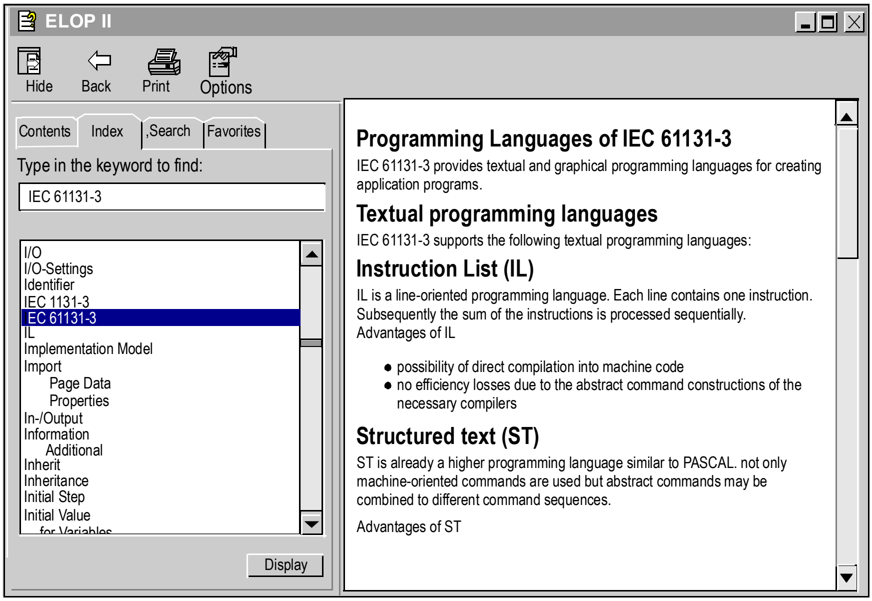
1 Drawing Field
2 Variable Declaration Editor
3 Overview Window
4 Interface Declaration Editor

Online Help (Project Management)

Overview

The online help contains detailed explanations for all functions in XPSMFWIN.

Representation



5.3 Objects in the Structure Window (Project Management)

At a Glance

Overview

This section describes the objects in the project management structure window.

What's in this Section?

This section contains the following topics:

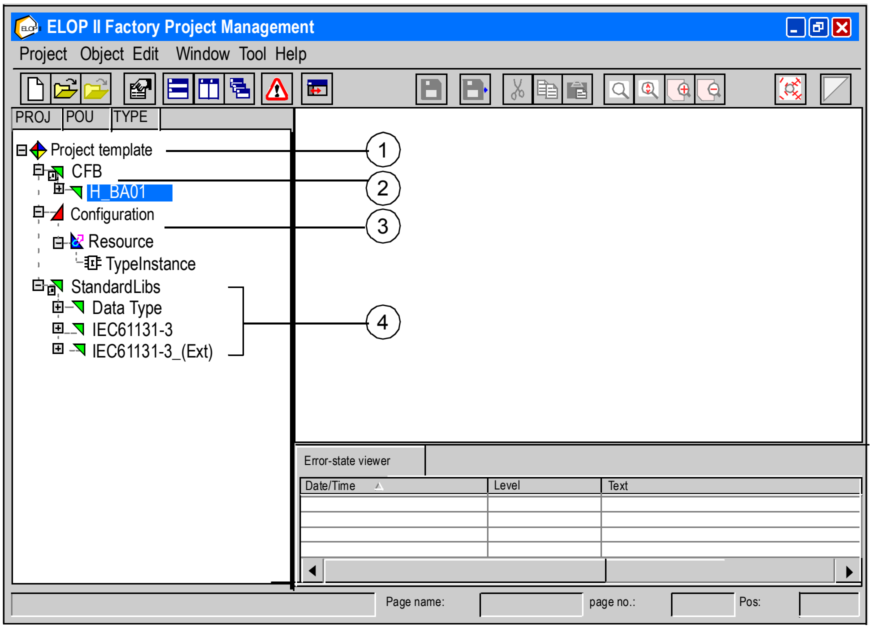
Structure Window, General

Overview

All objects in a project are displayed in their hierarchical structure and managed in the structure window.

Presentation

After creating a new project the **Project Management** window automatically is displayed containing the base template for starting programming.



1 Project Name
2 Certified Function Block Library
3 Configuration with a Resource
4 Logical Library Containing All IEC 61131-3 Certified Functions

Structure Window, Symbols (Project Management)

Overview

The project  is the highest-order object.
All other objects are created subordinate to a project.

Note: Only one project can be open at the same time in XPSMFWIN.

Symbols

Symbols used in the structure window of the **Project Management** window

Symbol	Description
	Project Highest order object, all other objects are created within the project. Only one project can be opened at any time.
	Configuration Groups the safety PLCs into logical units, enabling communication connections to be established.
	Resource A target system which executes a controller task, this is created within a configuration.
	Program Instance Refers to an existing program type in a library. The program is executed in this resource.
	Type Instance Contains no link to the program type but can contain logic.
	Library Contains functions, function blocks and programs. Also known as Program Organization Units (POU) in compliance to IEC 61131-3.
	Program Type Contains all the functions of an application. A program type can be assigned to multiple controllers. Each controller executes a program instance of the program type.
	Function Block Type Contains the subfunctions of an application similar to a subroutine. It can be used to structure the program to correspond to the system layout and can store values temporarily.

Library (Project Management)

Overview

The library contains functions, function blocks and programs.

They are also referred to as program organization units (POU) in compliance with IEC 61131-3. XPSMFWIN contains standard libraries with prefabricated functions.

You can use these functions to create your own more complex functions, function blocks and programs (POU).

CFB Library

The CFB library contains 14 Certified Function Blocks for various safety functions. These function blocks can be used alone or with other CFBs and logic such as AND, OR etc.

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.

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).

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).

5.4 Block Libraries (Project Management)

Structure of the Block Libraries (Project Management)

Overview

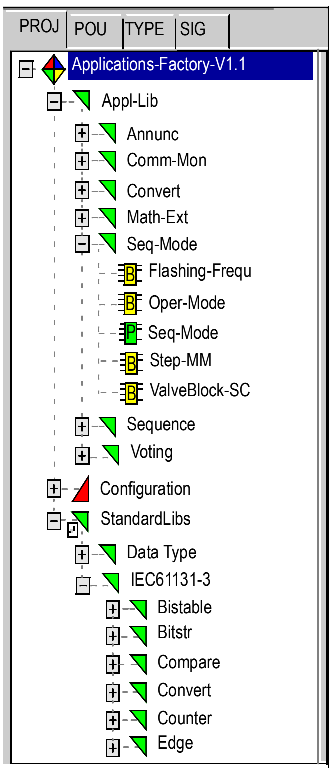
A project can consist of any number of personal libraries with numerous building blocks.

Personal 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.

Representation

Libraries in the structure window



5.5 Function Block Diagram Editor

At a Glance

Overview

This section describes the Function Block Diagram Editor.

What's in this Section?

This section contains the following topics:

General

Overview

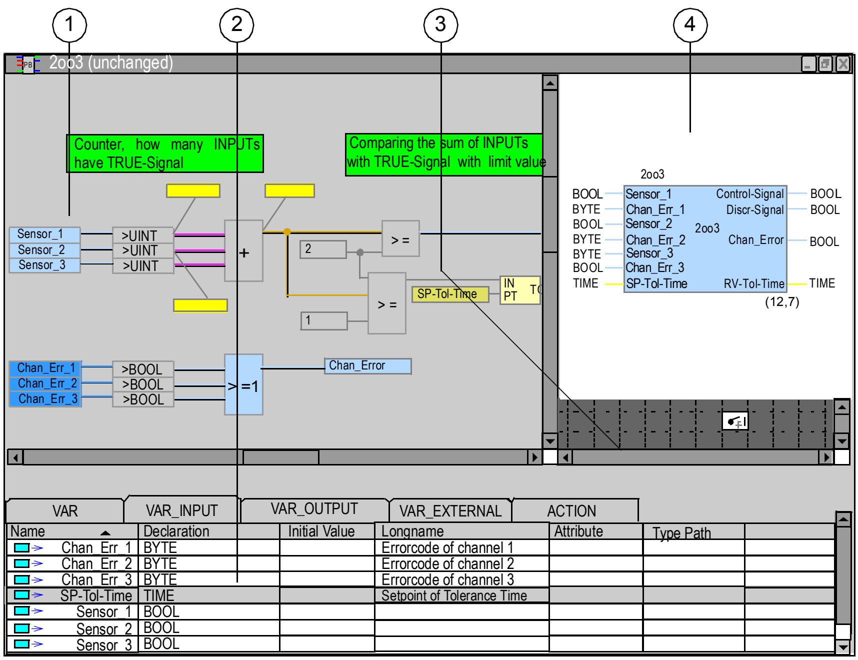
The Function Block Diagram editor is started automatically when a program organization unit (function, function block or program) is opened.

The following panes are displayed in a window within the working area of the **Project Management**:

- drawing field
- variable declaration editor
- overview window
- interface declaration editor

Representation

The FBD editor comprises the following panes:



1 Drawing Field
2 Variable Declaration Editor
3 Overview Window
4 Interface Declaration Editor

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

Maximizing the Working Area

Overview

The structure window (see *Structure Window, General, p. 46*) can be shown or hidden with the following button in the left side of the toolbar:

The same applies for the Error State Viewer and the following button:

This allows the working area for the Function Block Diagram editor to be enlarged or reduced.

The active window can be maximized by clicking the  button.

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

5.6 Function Diagrams

At a Glance

Overview

This section describes the Function Diagrams.

What's in this Section?

This section contains the following topics:

Function Diagrams with Centered Starting Point

Overview

The XPSMFWIN concept eliminates the need to subsequently insert individual pages later, because a plan of any size required is created.

Position of a Page

The position of a page is specified by coordinates. Columns are identified by capital letters and rows by numbers.

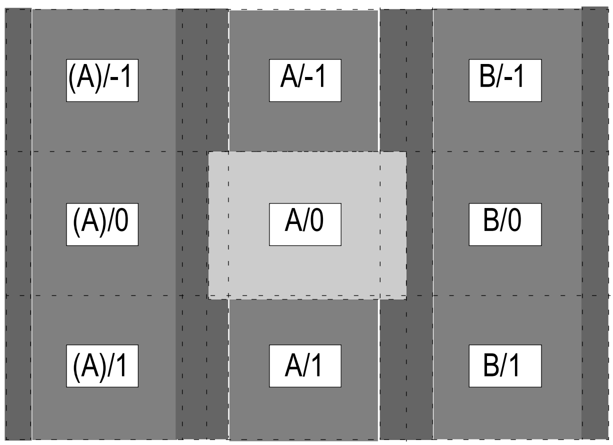
By default the first page has the coordinate A/0. As soon as an element is added to this page it becomes active.

Active Pages

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.

Representation

Page numbering in the function diagram

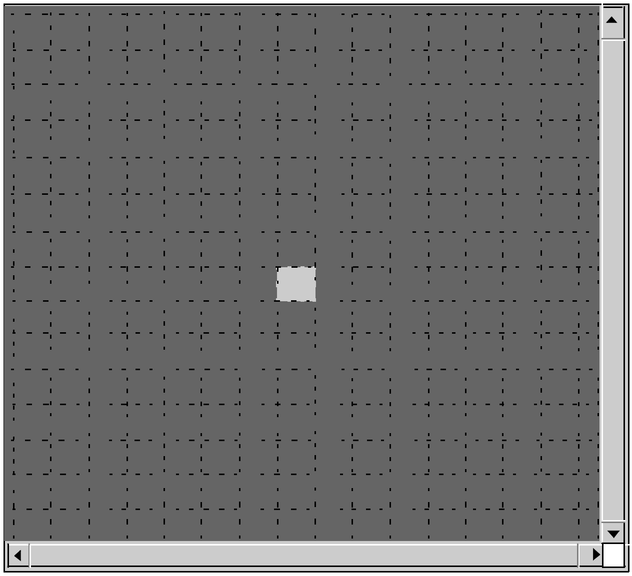


Note: If you need to insert a page between existing pages, you can move a page. To do this right-click and select **Plug-Ins** → **Move page** in the context menu of the page.This function should only be used during the development of a project, but not with a user program that is already in use. The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

Overview Window

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.

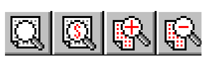
Overview window



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.

Zoom

Overview

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

5.7 Drawing Field Properties

At a Glance

Overview

This section describes the drawing field properties.

What's in this Section?

This section contains the following topics:

Drawing Field Properties

Overview

To edit the drawing field properties right-click in the working area and select **Properties** from the context menu.

Settings

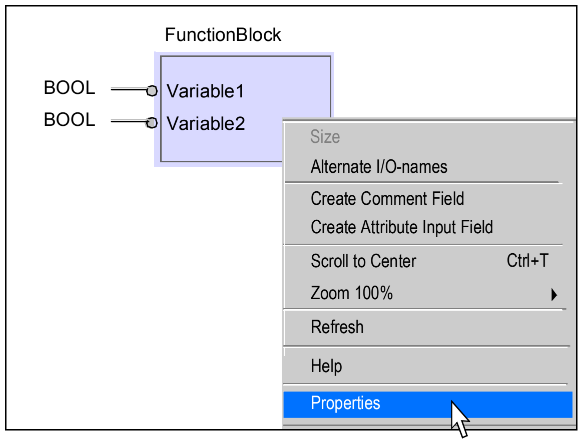
Within the drawing field properties you can change the settings of the following:

- grid
- page dimension
- color of drawing field and objects
- comment field position
- connector type
- value field
- page naming and numbering
- page changes
- SFC step intervals
- other general values

Specifying the Interface Declaration (Graphic View)

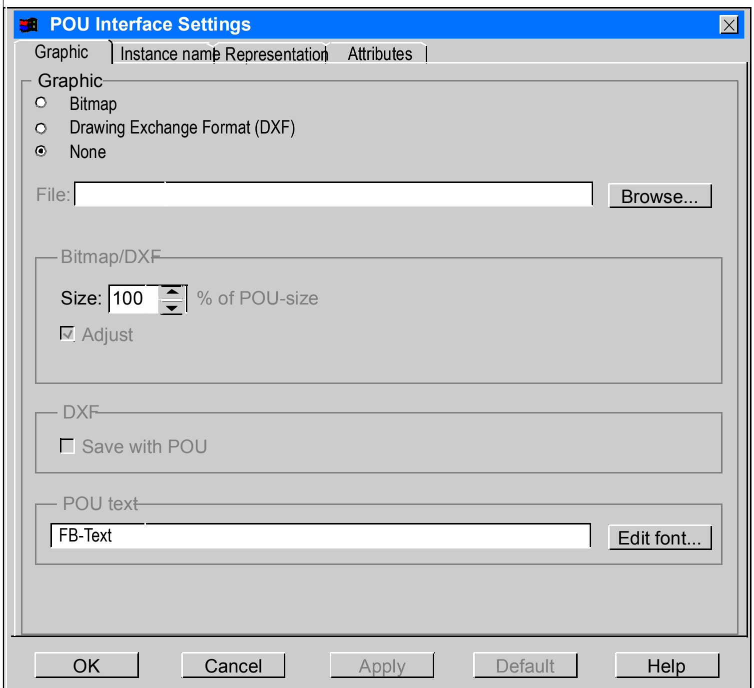
Step 1: Context Menu of the Function Block (POU)

Specify the properties of the function block in the Interface Declaration Window

Step	Action
1	Right-click on the function block and select **Properties** in the context menu. 

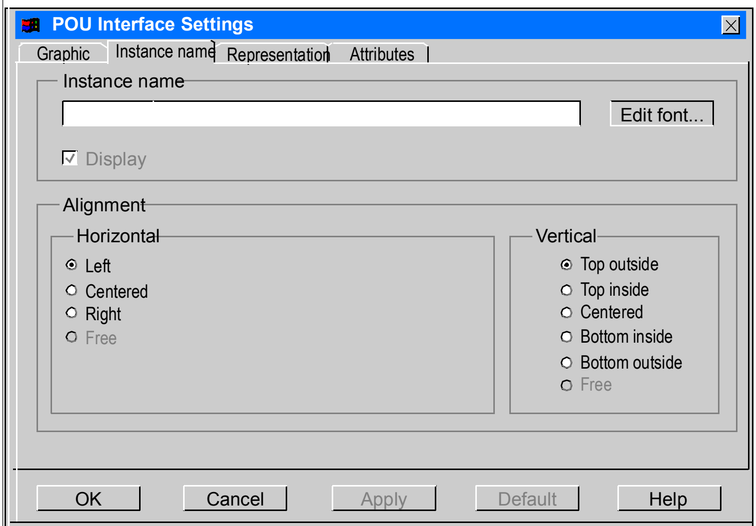
Step 2: Enter a POU Identifier

Entering the POU Text

Step	Action
1	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. **Result**: The input name appears in the centre of the function block (*see diagram step 4*). 

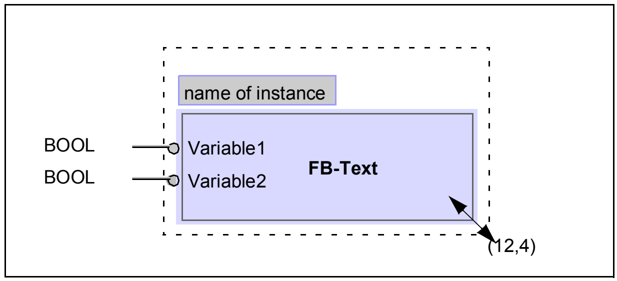
Step 3: Specify an Instance Name

Entering and formatting the instance name

Step	Action
1	Select the **Instance name** tab.
2	Enter a name in the **Instance name** field and select the **Display** field.
3	Set the **Font** and the **Alignment** if necessary. By default the instance name is shown at the top left of the block (*see diagram step 4*).  **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: Setting the Function Block Size

Setting the POU size

Step	Action
1	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. 

5.8 Creating Logic

At a Glance

Overview

This section describes creating logic in the function diagram with basic functions.

What's in this Section?

This section contains the following topics:

The Difference Between Signals and Variables

Signals

Signals are used to connect to the I/O of the hardware device and then to the program logic.

Signals are used for data exchange between the individual components of a resource (e.g. user program, I/O channels).

A variable which needs to be used in another area of a program such as communication via Modbus or a physical IO connection must be defined within the signal editor in the Hardware Management.

The signals are listed in the **VAR_EXTERNAL** tab, when transferred over to the project management screen.

Variables

A variable is a placeholder for a value within the program logic.

Therefore the name of the variable assigns a specific address in the memory where the value is stored and accessed.

Any variable requiring monitoring should be assigned as a signal.

Internal/External Interface

Note: Signals are used when a variable requires connection to the external environment (hardware I/O, Modbus/Profibus transfer etc.).
Variables should be used when no external interface is required.

Example

Requirement	Signal/Variable
A reset push button requires I/O connection	A signal is used.
A counter requires internal time input.	A variable is used.

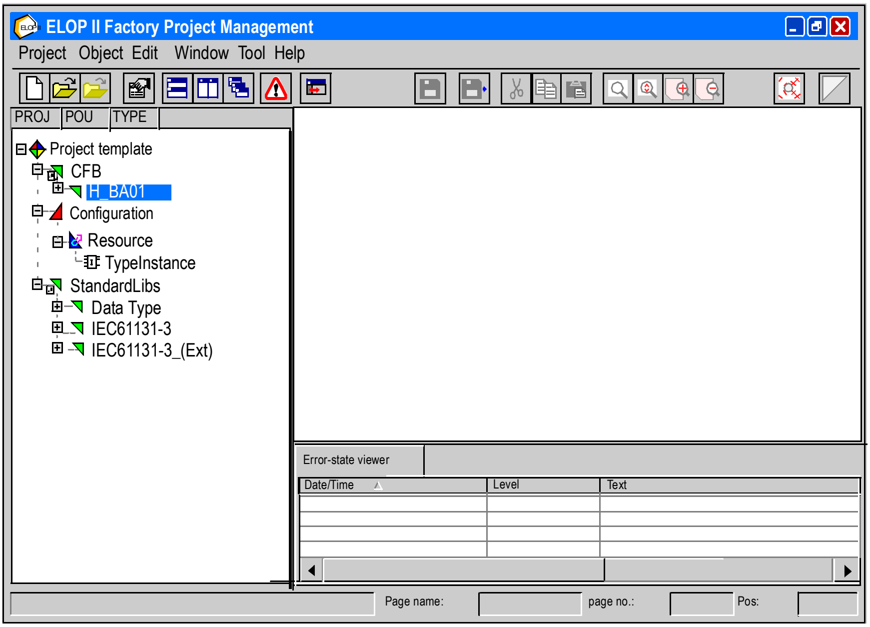
Types of Variables

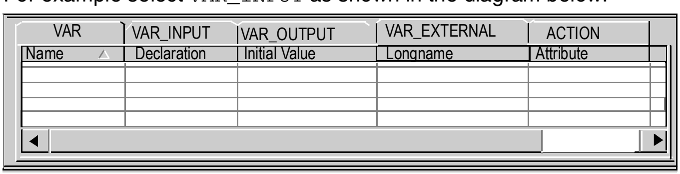
The 5 types of variables are used in the following ways.

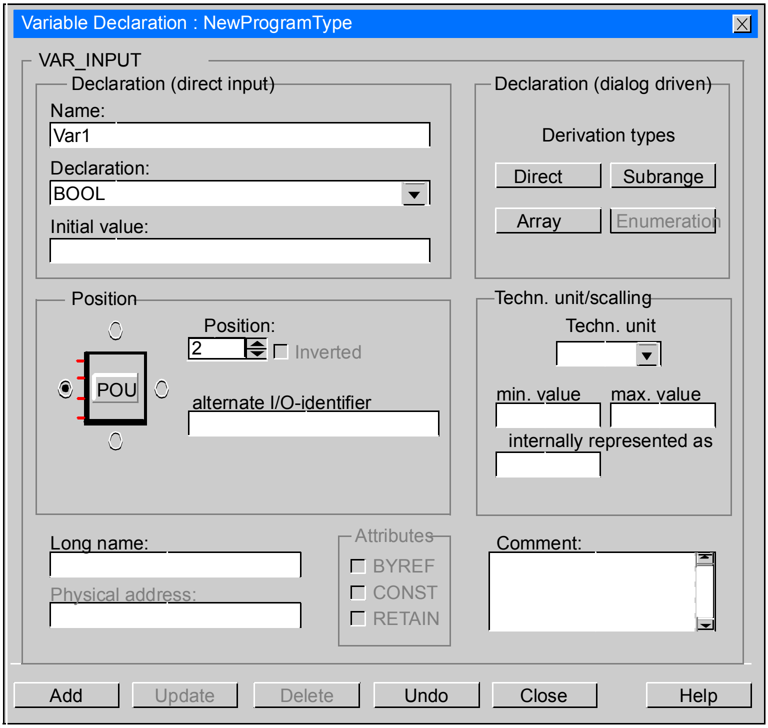
Type of Variable	Use
`VAR` standard internal variable	A standard internal variable is used within either a function block or program type. It can be used as an internal input or output.
`VAR_INPUT` input variable	An input variable is used within a function block. It is used as an input channel in a function block from a higher POU.
`VAR_OUTPUT` output variable	An output variable is used within a function block. It is used as an output channel in a function block to a higher POU.
`VAR_GLOBAL` global variable	A global variable has the same name as another variable and can be used repetitively.
`VAR_EXTERNAL` external variable	An external variable is used to store signals when transferred from the hardware management **Signal Editor**.

Creating a Variable

Procedure Creating a variable using the variable declaration editor.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>To create a variable double-click on the TypeInstance in the default Resource folder.


Result:The Function Block Diagram Editor will open. The bottom pane from the Function Block Diagram Editor contains the Variable Declaration Editor.</td></tr>
<tr><td>2</td><td>In the Variable Declaration Editor click on the tab with the desired variable type.
For example select <code>VAR_INPUT</code> as shown in the diagram below.

</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>Open the Variable Declaration dialog by double-clicking the next empty field in the Name column.
Result:


</td></tr>
<tr><td>4</td><td>The Variable Declaration dialog allows you to name the variable, to select the type, to provide an initial value and various other attributes.</td></tr>
</table>

Drag-And-Drop of Variables

Overview

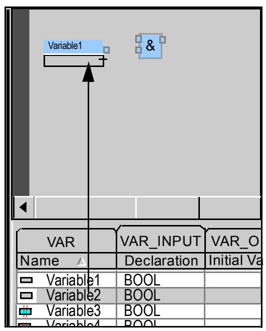
Variables are created in the variable declaration editor. There is a distinction between

- (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.

Representation

Drag-and-drop of variables into the drawing field



Note: You can also import variables from an external data source. See the Online Help for more information or see section "Additional features for programming areas" of this manual.

Drag-And-Drop of Function Blocks

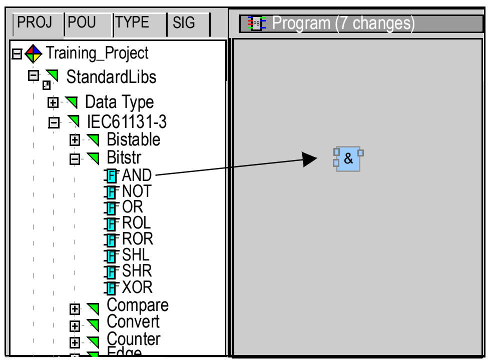
Overview

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.

Drag a Function Block to Drawing Field

Drag-and-drop logic elements from the libraries to the drawing field

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Open the library StandardLibs in the structure window by clicking on the + icon.</td></tr>
<tr><td>2</td><td>Open the library IEC61131-3 from the StandardLibs and then Bitstr.</td></tr>
<tr><td>3</td><td>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.</td></tr>
<tr><td>4</td><td>After releasing the mouse button the block is placed at the mouse pointer position.

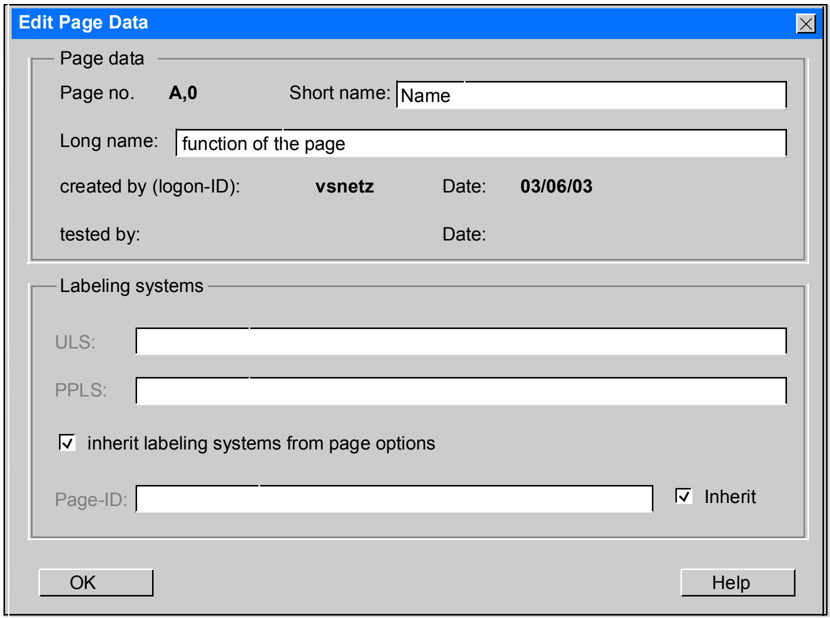


Note: The procedure for inserting functions blocks from libraries by drag-and-drop is also valid for user libraries.</td></tr>
</table>

Edit Page Data

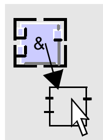
Because placing the `AND` function block is the first change to the contents of this page (see also *Function Diagrams with Centered Starting Point, p. 56*), the **Edit Page Data** dialog window opens automatically.

Enter a descriptive name for the page in the **Long name** field.



> **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.

Extend the Logic with Additional Function Blocks

Step	Action
1	Copy additional function blocks from the libraries into the drawing field as described in *Drag a Function Block to Drawing Field, p. 71*.
2	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  **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 **no parking** icon is displayed

Connecting Elements

Overview

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

The type of the line connection is determined by the type of the source node.

The two nodes must be of the same data type (e.g. `BOOL`, `BYTE`, `INT`).

Connecting

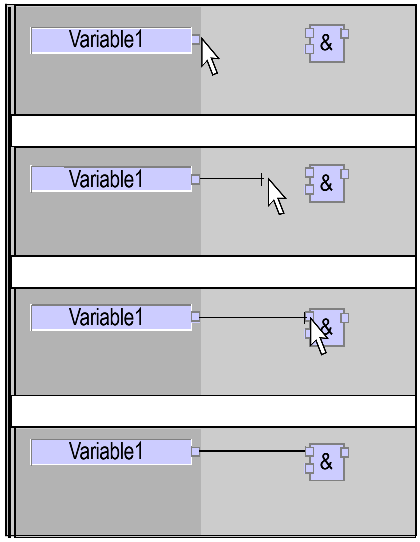
To create a connection, click on an output node and connect it to an input node by drawing a connecting line with the left mouse button held down.

You can also create nodes by connecting an existing line with another input or output.

> **Note:** XPSMFWIN automatically routes connecting lines, but they can be changed manually with the mouse if required. To do this, press and hold the `SHIFT` key and the left mouse button simultaneously. Drag the connecting line to the new position. Release the mouse button and then the `SHIFT` key.

Draw a Line

Draw connecting lines between the variables (value fields) and function blocks.

Step	Action
1	Place the mouse pointer on the connection point of a variable (= output).
2	Press and hold the left mouse button and draw a line to the right.
3	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.  **Note**: The color 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 **no parking** icon is displayed.

5.9 Creating and Managing Documentation

At a Glance

Overview

This section describes creating and managing documentation for the software plans.

What's in this Section?

This section contains the following topics:

General

Overview

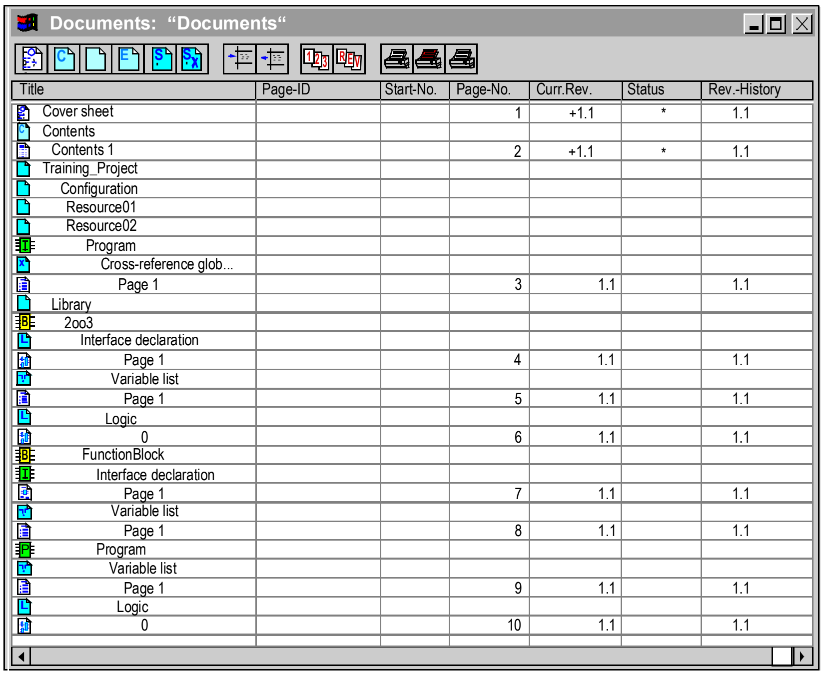
XPSMFWIN provides document management for the software plans with a revision service in the Project Management.

Modified Pages Only

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.

Representation

Documentation management



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						
0			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						
0			10	1.1		1.1

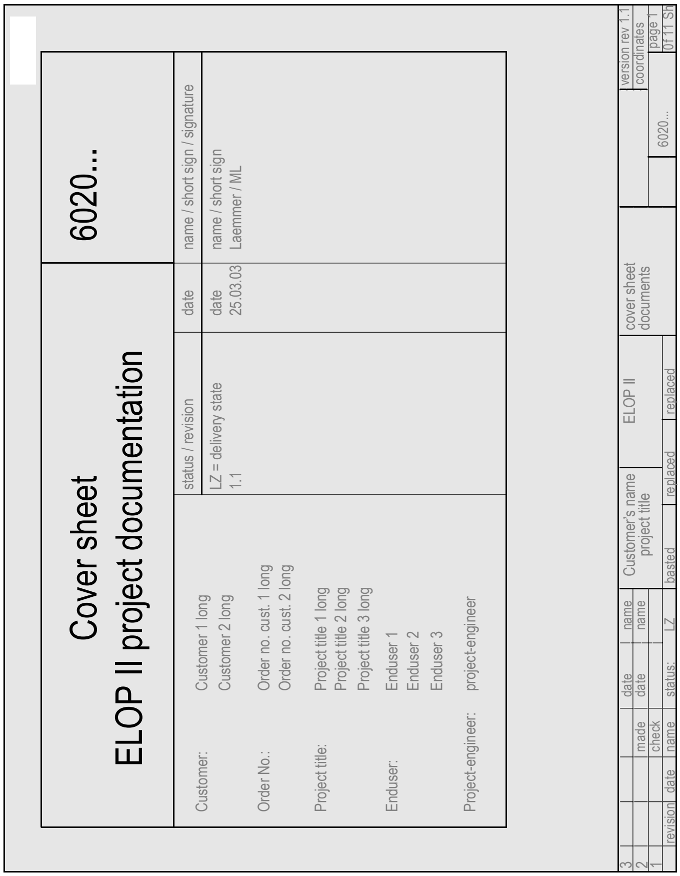
Templates for Printing Documents

Overview

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

Representation

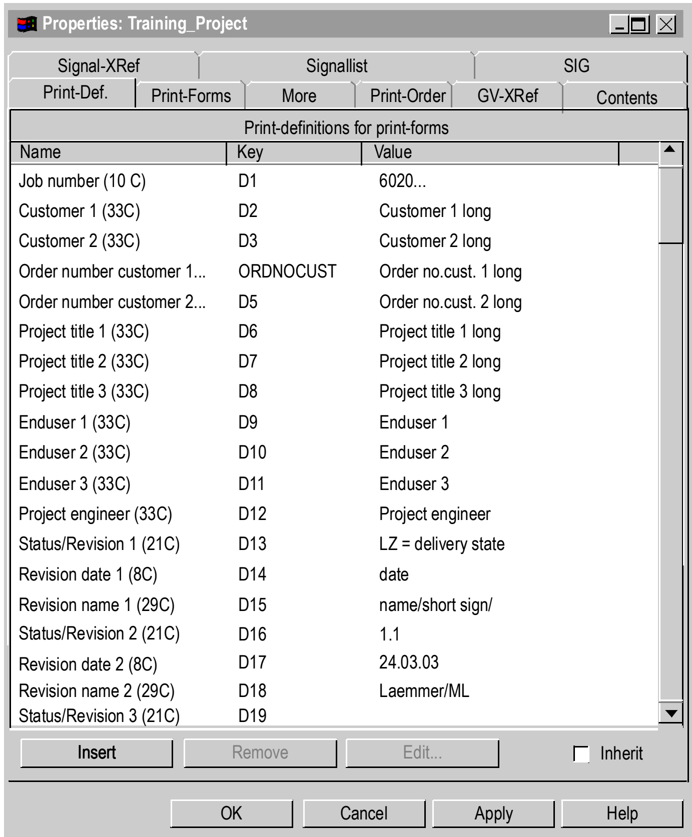
DXF form for the cover sheet



Object Properties

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

Project definitions for printout



Note: The hardware documentation must be printed out in the Hardware Management (see *Hardware Documentation, p. 88*).

Software Documentation

Overview

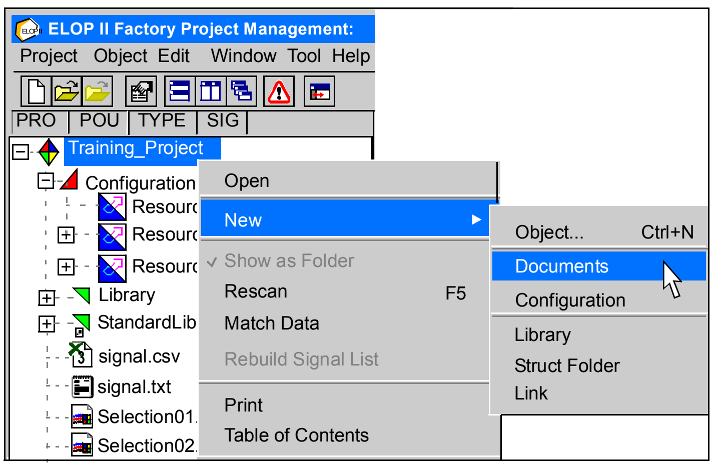
A printout of the function logic can be initialized and organized in a documentation object in the Project Management.

All POUs (modules) are printed out with the documentation object.

The hardware is documented separately in the Hardware Management.

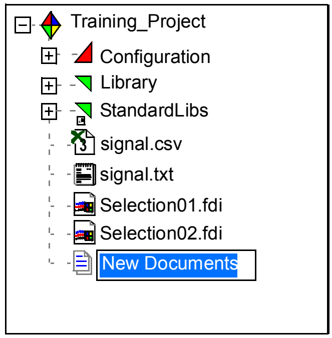
Step 1: New Documentation Object

Create a new documentation object

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Right-click on the project name.</td></tr>
<tr><td>2</td><td>In the context menu select New → Documents.
Creating a documentation object.

</td></tr>
</table>

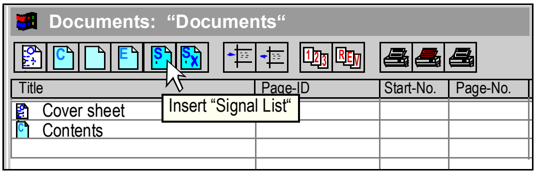
Step 2: Change the Name

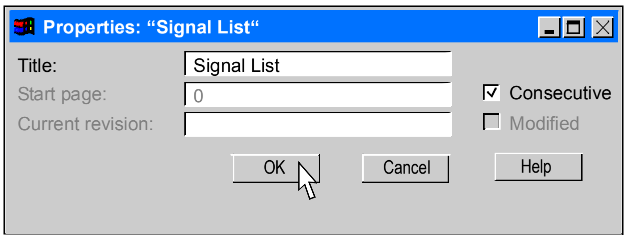
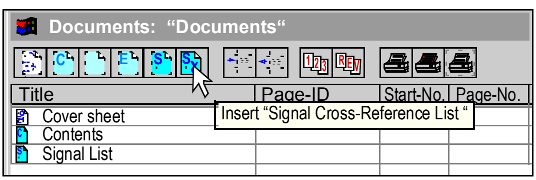
Change the name of the new documentation

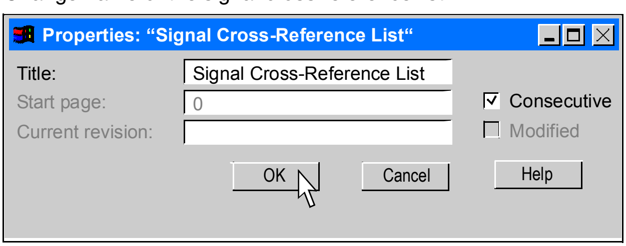
Step	Action
1	Click twice slowly on the name with the left mouse button. Change the name in the input field. Renaming document object 

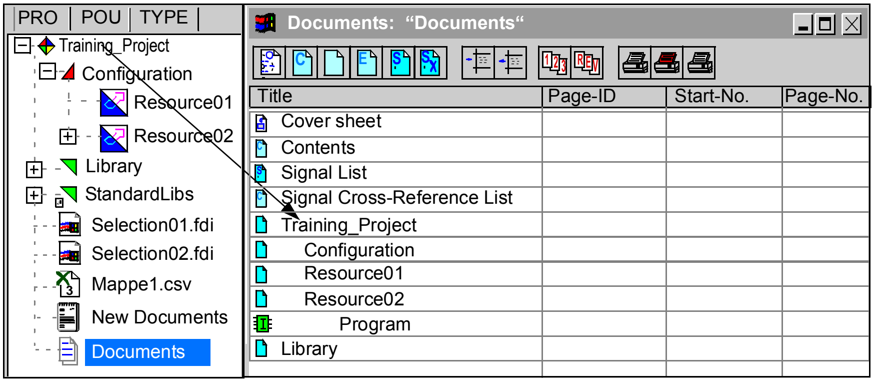
Step 3: Insert Data of Project

Insert all data of your project into the documentation

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Open the documentation by double-clicking it.</td></tr>
<tr><td>2</td><td>Add the signal list from the Hardware Management.
Add signal list to the documentation


Change name of the signal list

</td></tr>
<tr><td>3</td><td>Add the signal cross reference list from the Hardware Management.
Add signal cross reference list to the documentation.


Change name of the signal cross reference list.

</td></tr>
</table>

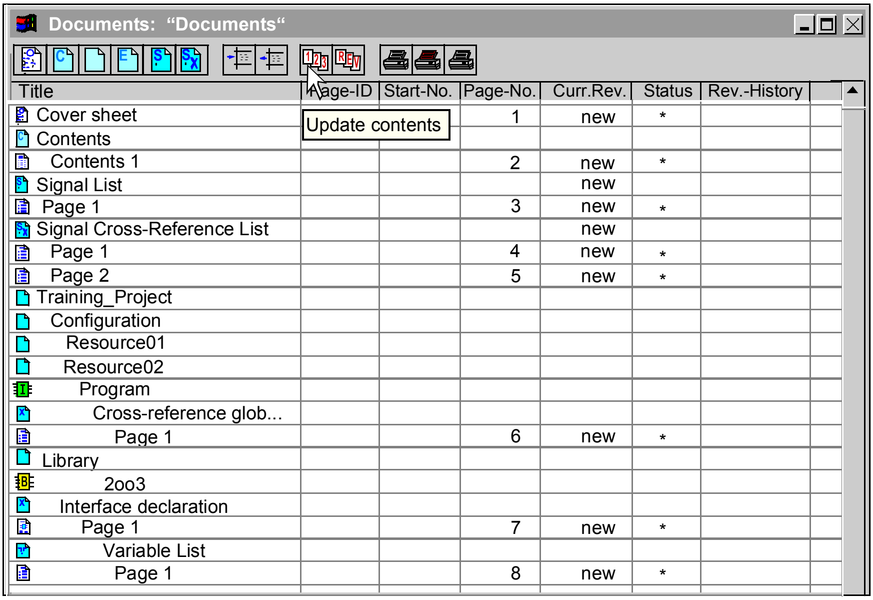
Step	Action
4	Click on the project, hold down the mouse button and drag the project into the documentation. **Result**: Now the documentation shows all elements that are contained in your project. Adding project to documentation.  **Note**: You can also add single objects from your project to the documentation.

Step 4: Update the Contents

Update the table of contents

Step	Action
1	Click on the **Update contents** button .

Updating contents

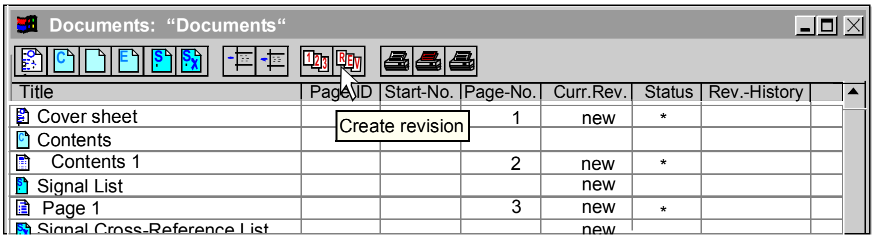
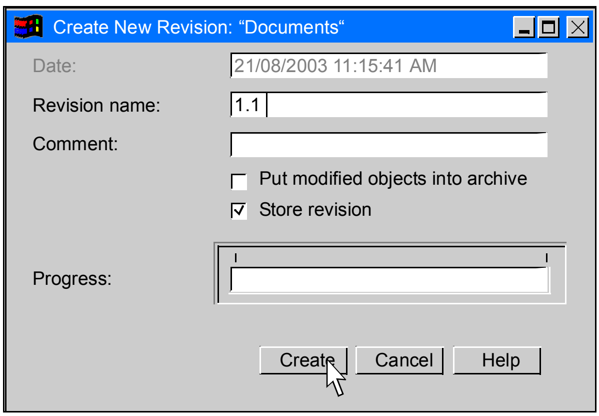


Title	Page-ID	Start-No.	Page-No.	Curr.Rev.	Status	Rev.-History
Cover sheet			1	new	*	
Contents						
Contents 1			2	new	*	
Signal List				new		
Page 1			3	new	*	
Signal Cross-Reference List				new		
Page 1			4	new	*	
Page 2			5	new	*	
Training_Project						
Configuration						
Resource01						
Resource02						
Program						
Cross-reference glob...						
Page 1			6	new	*	
Library						
2oo3						
Interface declaration						
Page 1			7	new	*	
Variable List						
Page 1			8	new	*	

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.

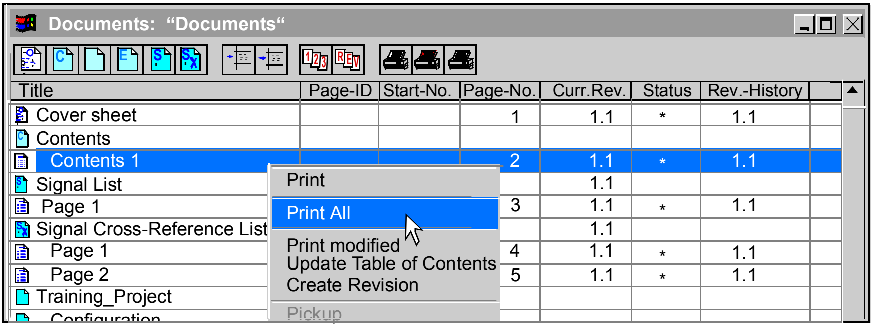
Step 5: Create New Revision

Create a new revision status

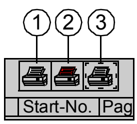
Step	Action
1	Click on the **Create revision** button  . 
2	Enter a value for the revision status and click on the button **Create**. Enter revision name (version) 

Step 6: Print the Documentation

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>
Start the print job for individual, all or modified pages from the context menu or with the buttons in the toolbar.
Printing with the context menu



Starting print



1 Print All
2 Print Modified
3 Print Selected Pages
</td></tr>
</table>

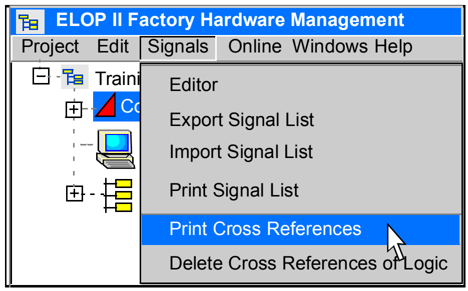
Hardware Documentation

Overview

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

Step 1: Print Cross Reference List

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

Step	Action
1	In the **Signals** menu click on **Print Cross References**. Printing cross references  **Note**: You can also print the signal list without cross references by selecting **Print Signal List** in the **Signals** menu.

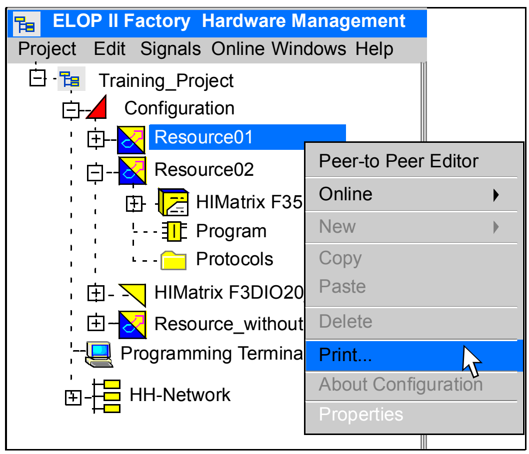
Print Resource Documentation

Overview

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

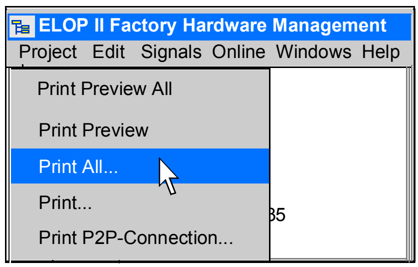
Step 1: Print Resource Documentation

Print resource documentation

Step	Action
1	Go to the Hardware Management.
2	Select **Print...** in the context menu of a Resource. Printing the resource documentation  **Note**: You can start the print job for the hardware documentation for the resources altogether (see *Print Hardware Management Documentation, p. 90*).

Print Hardware Management Documentation

Step 1: Print Documentation

Step	Action
1	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. ● **Print P2P-Connection** to print all Peer-to-Peer connections. 

5.10 Offline Simulation of Function Diagrams (OLS)

At a Glance

Overview

This section describes the Offline Simulation of Function Diagrams.

What's in this Section?

This section contains the following topics:

Check Function Diagrams Without a PLC Connected

Overview

You can use the offline simulation to check the created function diagrams on the PC for logical correctness without having to use a programmable logic controller (PLC).

The function plans are translated by XPSMFWIN and processed by the PC.

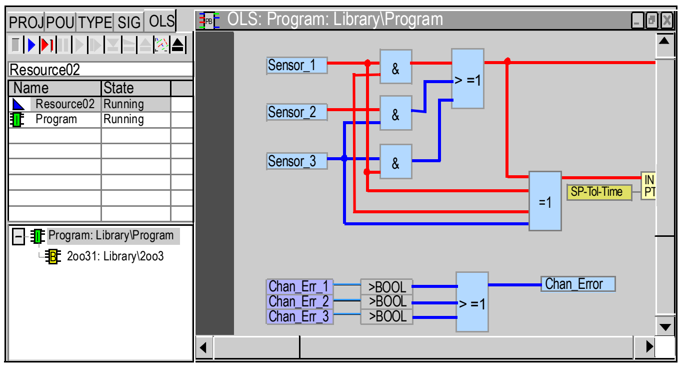
The offline simulation can only be executed on program instances within a resource.

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 color.

Representation

Offline simulation of a function diagram



Offline Simulation of a Program

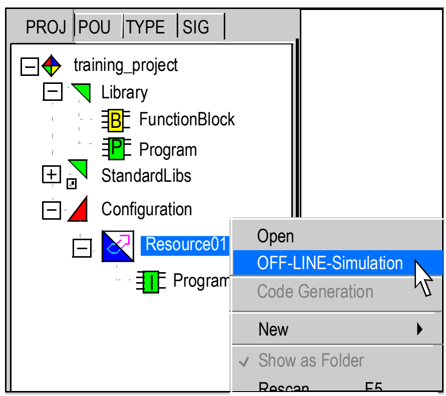
Overview

The behavior of a function block or of a program is tested in the offline simulation without using the PLC (hardware). This allows programming errors to be detected and corrected before commissioning.

Configuration

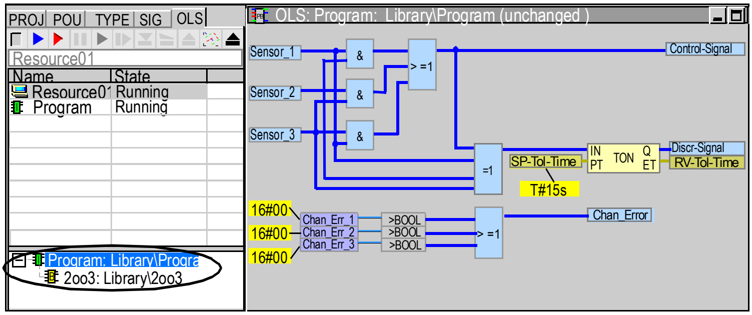
The configuration contains the resource in which the program that is to be tested is instanced.

Step 1: Activating the Offline Simulation

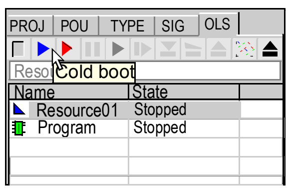
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Open the context menu by clicking the right mouse button on the Resouce which contains the users’ program to be tested.</td></tr>
<tr><td>2</td><td>Select OFF-LINE Simulation.


Note: An additional tab OLS (Offline Simulation) opens in the structure window.</td></tr>
</table>

Step 2: Open the Offline Simulation

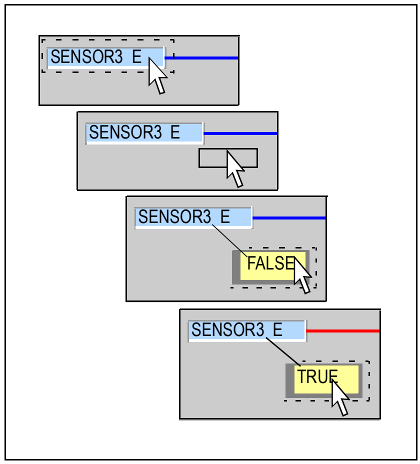
Step	Action
1	Open the program by double-clicking on the program in the structure window (lower left pane).  **Note**: You can open multiple function blocks simultaneously and trace the signals between the function blocks. The function blocks can also be opened directly in the logic in the drawing field.

Step 3: Starting the Offline Simulation

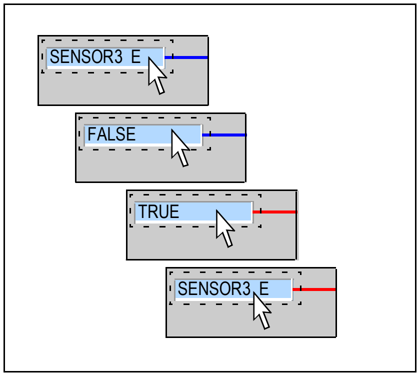
Step	Action
1	Click on the **Cold boot** button. 

Step 4: Change the State of the Signals

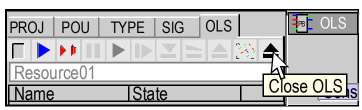
a) Change signal state with an online test field (OLT field)

Step	Action
1	Left click on a variable/signal and hold down the mouse button.
2	Drag the mouse from the variable/signal and release the mouse button at a free position on the screen. A OLT field is displayed.
3	Position the OLT field by clicking with the mouse.
4	Change the signal state by double-clicking in the OLT field. 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:

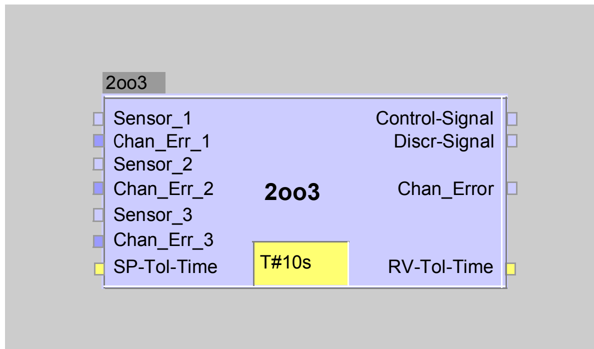
Step	Action
1	Place the mouse pointer on the value field whose value you want to change.
2	Press and hold the **ALT** key. The variable state is displayed.
3	Change the variable state by clicking with the mouse on the value field.
4	Release the **ALT** key. The variable name is shown again. Switching value field display with the **ALT** key  **Note**: You can only change values that are not written by the logic.

Step 5: Close the Offline Simulation

Step	Action
1	Click on **Close OLS** in the toolbar. 

Offline Simulation of a User-Defined Function Block

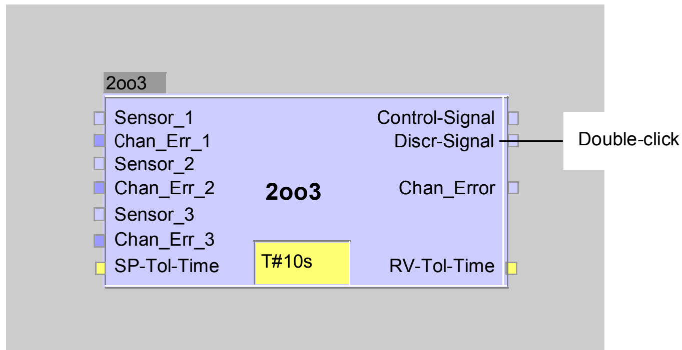
Step 1: Block Without Wiring

Step	Action
1	Drag the function block that is to be tested into the program.
2	Save the program  **Note**: If necessary, wire the required feedbacks (e.g. for group signals) to allow the block to be tested with the feedback function.

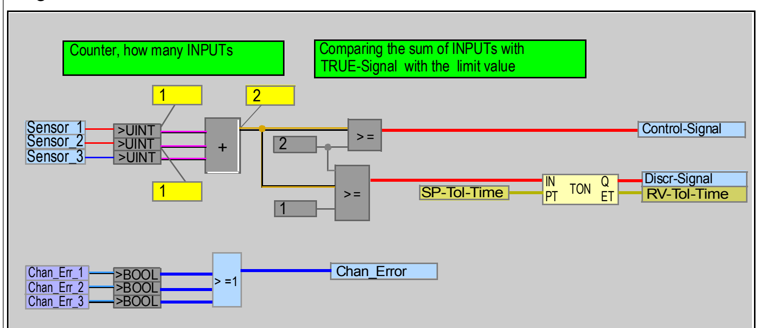
Step 2: Opening the Function Block

Start the offline simulation as described in *Offline Simulation of a Program, p. 93*.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>After opening the program, open the function block by double-clicking on the function block.



Logic in the function block


</td></tr>
<tr><td>2</td><td>Test the behavior of the function block by changing the <code>VAR_INPUT</code> values (possibly <code>VAR_EXTERNAL</code>) by clicking with the mouse while holding down the ALT key.</td></tr>
<tr><td>3</td><td>Close the offline simulation again (see also Offline Simulation of a Program, p. 93, Step 5).</td></tr>
</table>

5.11 Online Test (Power Flow)

Monitoring Values of Variables and Signals

Overview

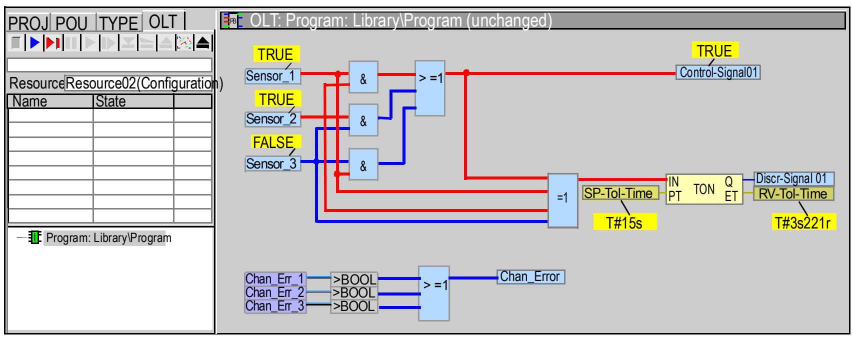
The online test in the Project Management is used to monitor all values of the variables and signals within the logic while the application is running.

The online test allows you to check the connection flow ensuring that the overall program logic functionality is correct during the functional test phase of a project.

To run the online test (OLS) the programs must be attached to the resource (PLC).

Representation

Online test

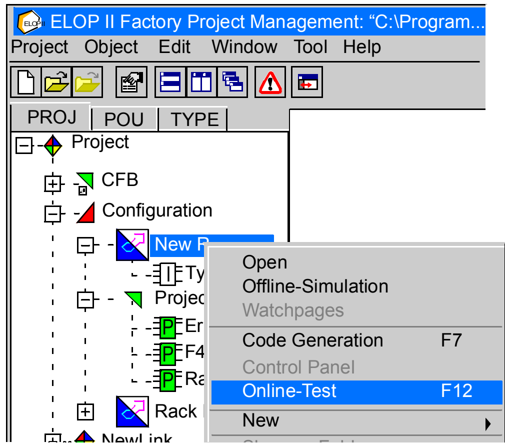
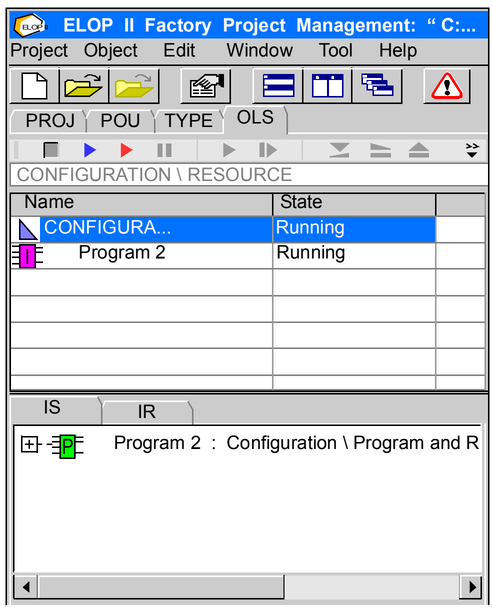


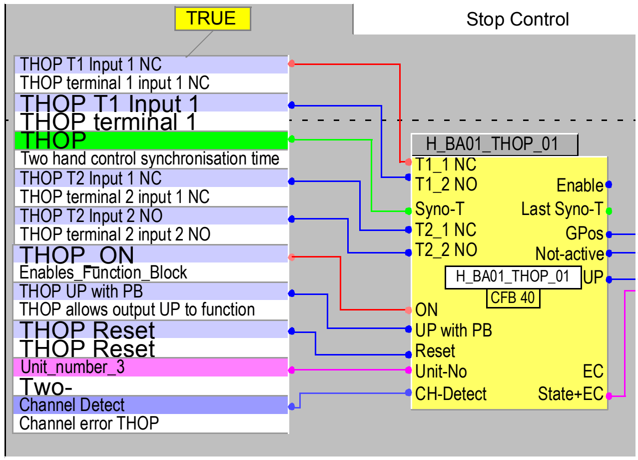
Note: The PLC must be physically connected to the programming terminal (either directly or via a network) with the PLC's online control panel **online**.

Requirement

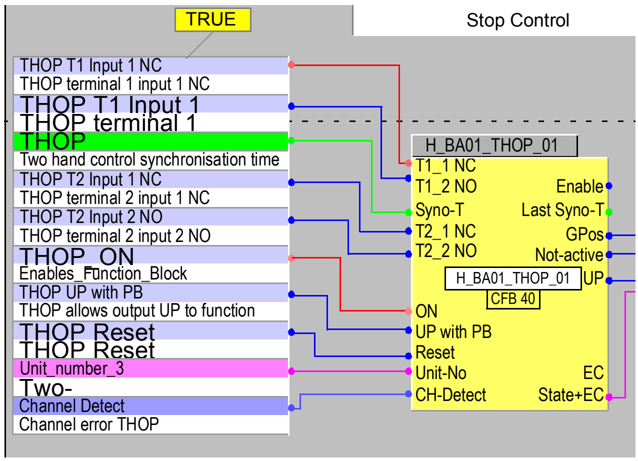
The safety PLC must be connected to the programming terminal, and online with the last version of the program downloaded and in the run state.

Procedure

Step	Action
1	Right-click on the **NewResource** under **Configuration** and select **Online-Test** (F12)  **Result:** 
2	Add a new tab to the project tree.

Step	Action
3	Double-click on the program that you want to test. **Result:** The logic will open with all states 
4	To add OLT field keys (shown in yellow), click on the signal or variable and drag to a free space in the working area. **Result:** The OLT fields show the current state of the signal or variable. **Note:** Values of physical input signals connected to hardware inputs, outputs or non-safety related communication inputs and outputs can only be changed using the **Force Editor** from within the **Hardware Management** working area. For detailed information see *Signal Monitoring with the Force Editor, p. 269*.

Step 2: Adding OLT field keys

Step	Action
1	Add a new tab to the project tree.
2	Double-click on the program that you want to test. **Result:** The logic will open with all states 
3	To add OLT field keys (shown in yellow), click on the signal or variable and drag to a free space in the working area. **Result:** The OLT fields show the current state of the signal or variable. **Note:** Values of physical input signals connected to Hardware inputs, outputs or non-safety related communication inputs and outputs can only be changed using the Force Editor from within the Hardware management working area. See section 6.9 for assistance with the Force editor

5.12 Creating a Resource

Creating and Renaming a Resource

General

See *General Working With Resource Types, p. 151*.

Resource / Remote I/O

A resource is a device which can execute a logic program.

Devices without this capability are called remote I/Os.

Resource

A resource (e.g. HIMatrix F30, F31, F35, XPSMF40, XPSMF60) is created in the Project Management.

As a default a resource is provided within the configuration folder of the project management screen.

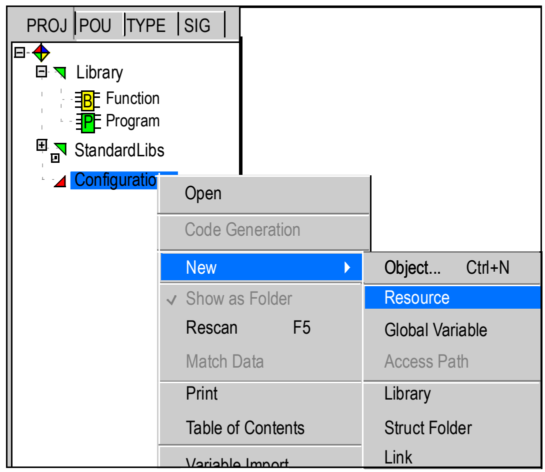
> **Note:** To define the physical properties of the Resource such as product type HIMatrix F30, F31, F35, XPSMF40, XPSMF60 you must use the **Hardware Management** window and define the type from the **Properties** tab of the resource. Therefore as the program is independent of Hardware it simplifies the programming structure.

Remote I/O

A remote I/O ((e.g. HIMatrix F2DO 1602, F3DIO20/8 02, etc. ...) is created in the Hardware Management in the **Remote I/O** folder of the parent resource.

These devices only communicate safety-related information with a parent resource.

Step 1: Creating a Resource

Step	Action
1	Open the Project Management.
2	With the right mouse button click on the **Configuration** in the Project Management and open the context menu.
3	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.
  Result: 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.
- Or click on the resource with the left mouse button and then press F2.

Setting Resource Properties

The properties of a resource are set in the Hardware Management. See *Setting the Properties of a Resource, p. 154*.

Creation of a Remote I/O

For the creation of a Remote I/O see *Assigning Remote I/O, p. 167*

5.13 Creating a Personal Library

Creating a Personal Program and Function Block Library

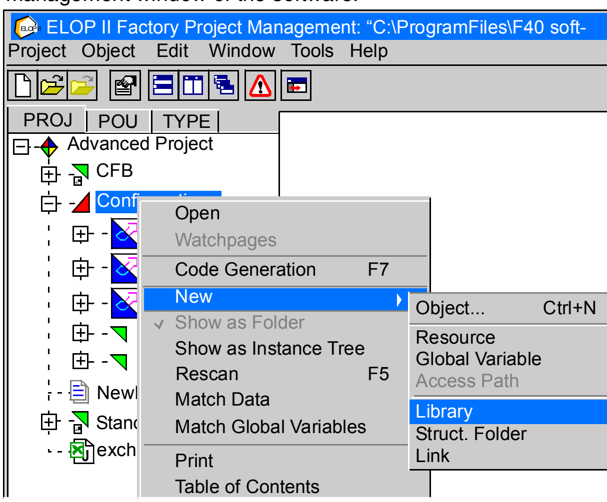
Overview

To organize all personal programs, functions and function blocks, create a personal library.

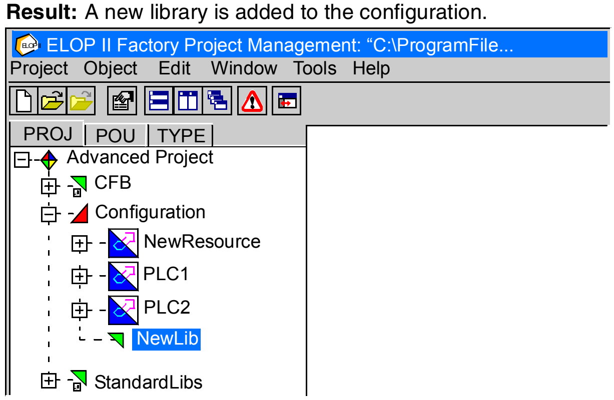
Creating a New Library

Create as many libraries as required by repeating the following procedure:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Right-click on the Configuration → New → Library within the project management window of the software.



Result: A new library is added to the configuration.


</td></tr>
</table>

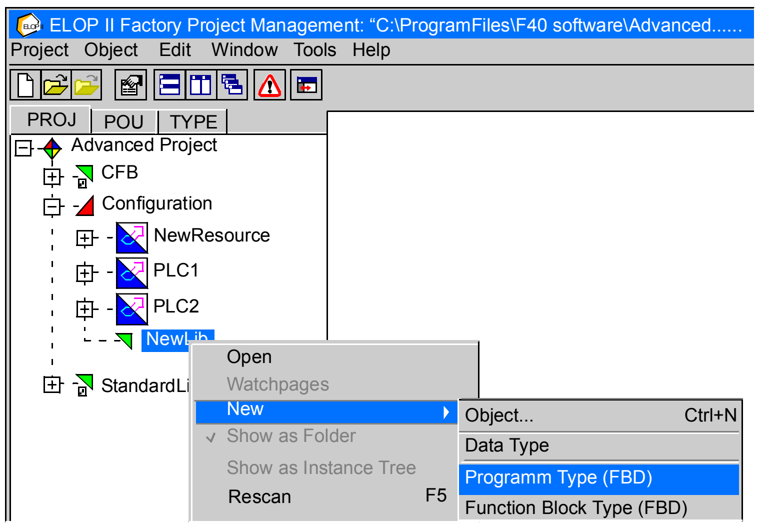
Renaming a Library

The new library can be renamed as needed in the following way:

Step	Action
1	Select the library that you want to rename.
2	Press function key F2 and rename the library.

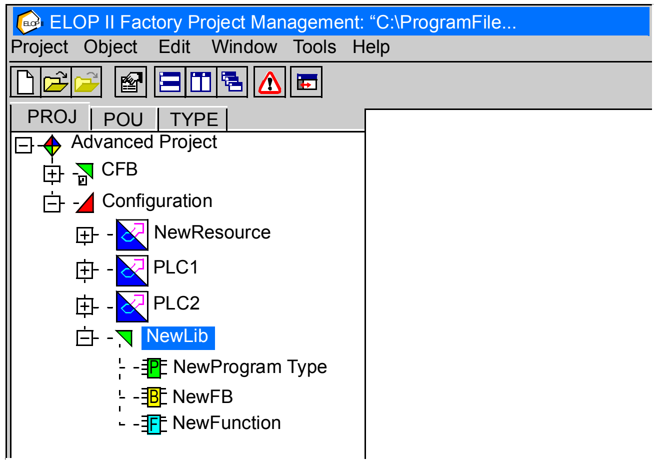
Adding a Program, Function or Function Block

To add new elements right-click on **NewLib** and select
- New → Program Type (FBD)
- New → Function Block Type (FBD)
- New → Function (FBD)



Creating Logic

To create a logic within the **Program Type**, **Function Block Type** or **Function** proceed as follows:

Step	Action
1	Double-click on the icon from your library. 
2	Enter variables and logic (CFB's, etc.) into the required drawing areas.

5.14 Assigning a Program to a Resource

Assign Program Type to Resource

Overview

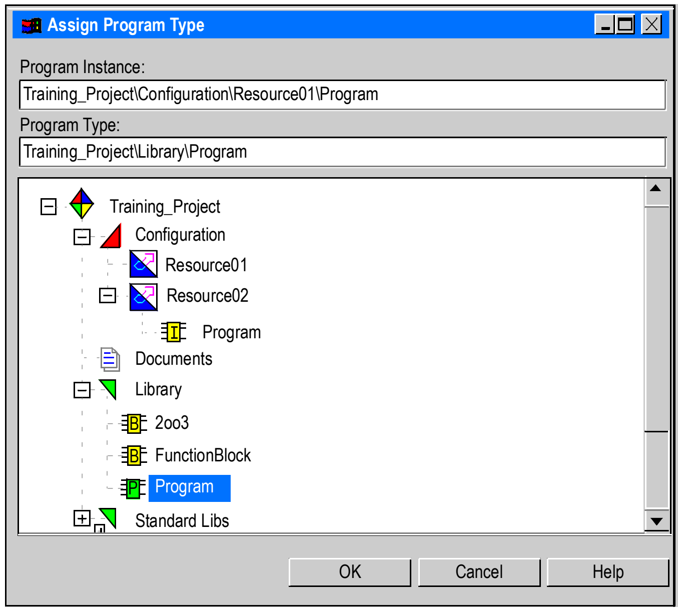
Before a resource can execute a program, a program type must be assigned to the resource.

A program is assigned to a resource in the Project Management.

Note: As a default a type instance is assigned to each resource.
It is possible to program within the type instance or to delete the type instance and program within a program instance.

To attach a program instance you must first delete the type instance from the resource.

Program Type of a Resource

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Open the resource context menu by right-clicking with the mouse and select New → Program Instance.</td></tr>
<tr><td>2</td><td>Select a program type from your library in the dialog box.
Result: The program type then appears as a program instance with the same name in the structure window in the resource.

</td></tr>
</table>

Reasons for using program library or type instance.

Reasons	Type Instance	Program
Can be used for creating program (including use of Function Block, etc.)	+	+
Reduces overall project size as only the required program is assigned	+	-
Able to link same program to many PLC’s	-	+
Able to use the link function to use the programs in other projects	-	+
Possible loss of program when Resource is manually deleted	+	-
Complete project simplicity	-	+

+ applicable

- not applicable

5.15 Code Generator

At a Glance

Overview

This section describes the code generator.

What's in this Section?

This section contains the following topics:

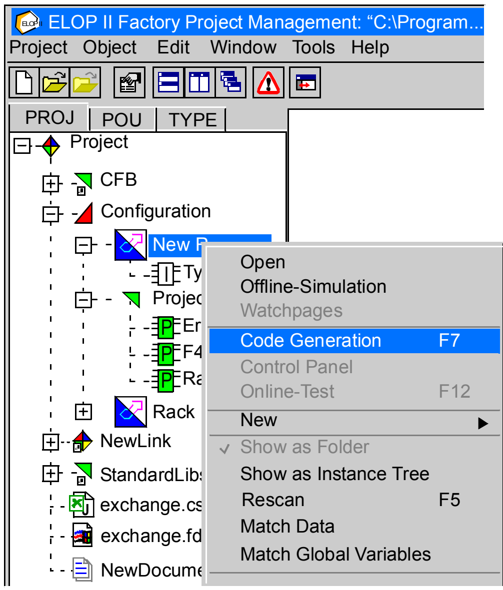
Code Generator

Overview

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.

Starting the Code Generator

The code generator is started from the context menu of the configuration or of a resource within the **Project Management**.



Note: Do not use the code generator in the Hardware Management as it has no effect. It was used for a specific remote I/O module which is no longer available.

Result of the Code Generator

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

Note: When code is generated on a resource, only the selected resource will execute the code generation.
When code is generated on a configuration, the complete application will execute the code generation

Program Messages

The meaning of program messages can be found within the online help from within the Project Management window: **Help Topics** → **Lists and References** → **Program Messages** → **Occurrence of Program Messages**.

Error-State Viewer

Note the messages in the error-state viewer of the Project Management and the Hardware Management. An interruption of the code generation can be caused by errors in the configuration and/or by error in the logic of the user program.

Error-state viewers after code generation

Error-state viewer

Date/Time	Level	Text
3/24/03,12:43:41 PM	Information 2	ERROR=0 (0=ok)
3/24/03,12:43:41 PM	Information 2	==========END:of POST-compiler messages ==========
		D:\Program Files\ELOP II Factory\BIN>LcErrLev 100 0
3/24/03,12:43:41 PM	Information 2	MCG004:Binary code generation finished for <Resource02>
3/24/03,12:43:41 PM	Information 2	MCG019:Error=0/Warnings=1
3/24/03,12:43:41 PM	Information 2	MCG009:Error-free code generated for <F:\Training\ELOP II_Factory_Projects\Training_Project L2P\Configu

```
3/24/2003,12:34:32.728, Info: ( Resource02 ) Code generation started.
3/24/2003,12:34:41.230, Warning: ( Resource02 ) / Configuration 'Resource02/Program: Used signal 'SP-Tol_Time' has init value but no source.
3/24/2003,12:34:41.681, Info: ( Resource02 ) code generation finished. Warnings: 1, Errors: 0.
```

Note: In the event of warnings also note the preceding messages. In the above example the signal SP-Tol_Time was created in the Signal Editor and dragged into the program.
The system normally expects a source for the value of the signal. An input channel can be used as a source, or the data exchange with another resource.
In the example only the option of displaying or modifying the value of the signal in the Force Editor was intended.

Error-Free Code

Note: For safety-related applications the code generator must be carried out twice and must be downloaded two times.
The checksums (CRCs) of the two generated code versions must be compared. The code is guaranteed to be error-free only if the checksums are identical.

Cyclic Redundancy Check

Overview

The CRC also known as the checksum is a unique value which is created during code generation.

The CRC is a 4 Byte value which is dependant on the Logical configuration, Hardware used, IO configuration, non-safety protocol connection and the communication parameters of each PLC and Remote IO within the configuration.

Configuration Changes

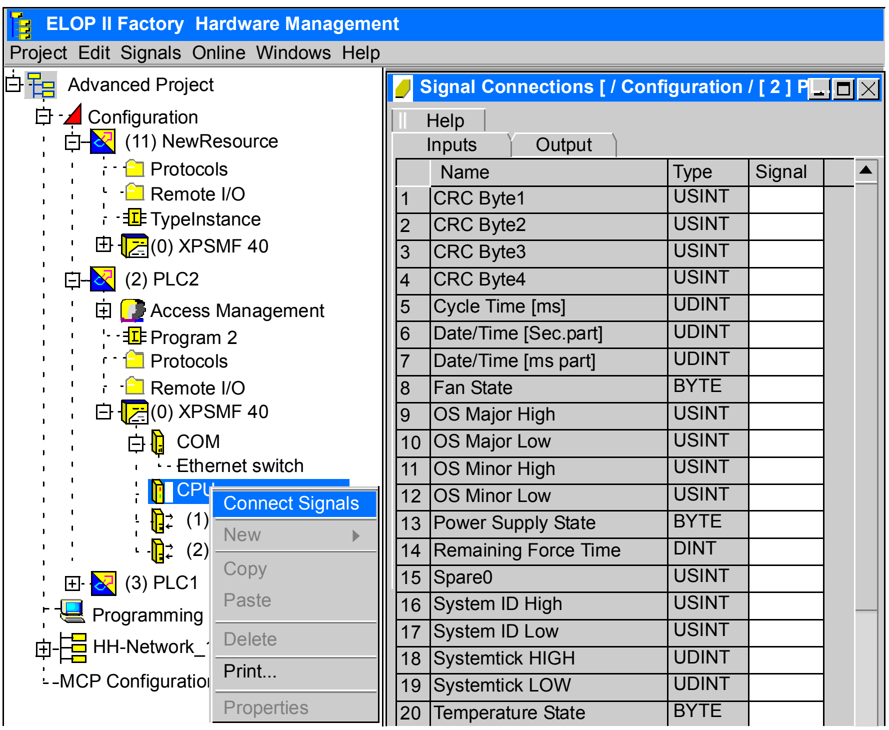
Any changes to the configuration will change the CRC values.

Therefore, if a change is made to any of the parameters within the software, code generation must be carried out again (two times to ensure safety) before a download is possible.

Monitoring CRC Values

The CRC values can be monitored within each safety PLC and Remote IO by assigning signals to each of the 4 Bytes located within each CPU of the PLC and Remote IO.

Signal connection is managed within the Hardware management window of the Software.



5.16 Checksums

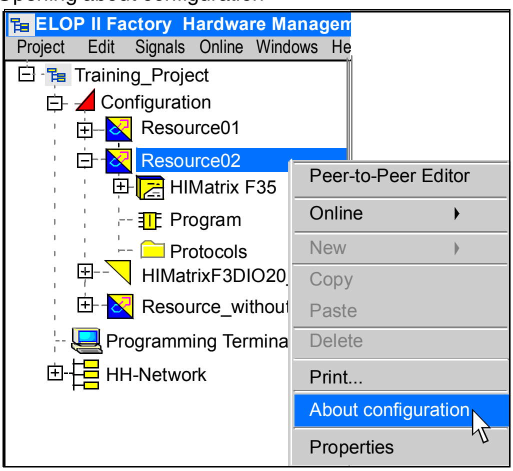
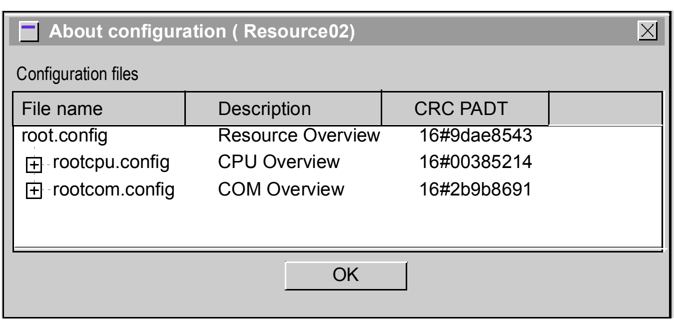
Checksums

Overview

After code generation is completed the checksums of two generated code versions must be compared to guarantee the correctness of the code.

Noting Checksums

If code generation was successful, write down the generated checksums. Proceed as follows:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select About configuration in the resource context menu of the Hardware Management.
Opening about configuration

</td></tr>
<tr><td>2</td><td>Write down the checksums for root.config in the column CRC PADT.
Noting checksums

</td></tr>
</table>

Regenerating the Code

⚠ WARNING
UNINTENDED EQUIPMENT OPERATION
For safety-related applications the code generator must be started twice and the checksums (CRCs) of the two generated code versions must be compared.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

The code is only error-free if the checksums are identical. See the safety manual for additional details.

Generate the code for the resource again

Step	Action
1	Run the code generation again.
2	Open the **About configuration** window as described in Step 4.
3	Compare the checksums of the second code generation with the previously noted checksums. **Remark**: The code must only be loaded into the resource if the checksums are identical. **Note**: By generating the code for a resource, the assigned remote I/Os are included automatically. The checksums of the remote I/Os contribute to the *root.config* CRC of the parent resource.

5.17 Archiving

Archiving a Project

Overview

A project is archived using the **Project Management** working area.

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

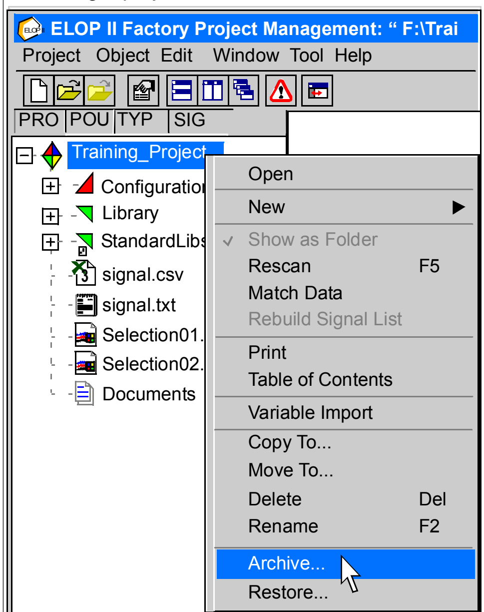
CAUTION

LOSS OF DATA

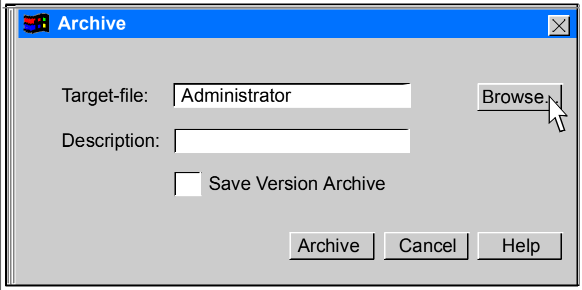
For reasons of security and safety a user program that has been downloaded into a PLC cannot be uploaded into the Programming Terminal! For this reason archiving is important. Make sure that you archive your program before downloading.

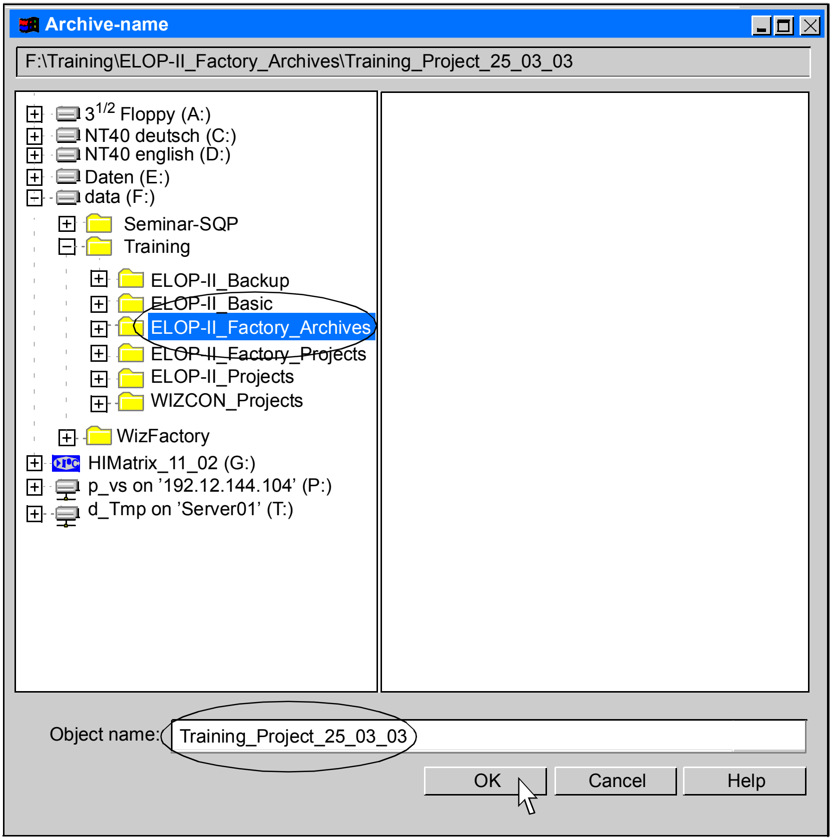
Failure to follow these instructions can result in injury or equipment damage.

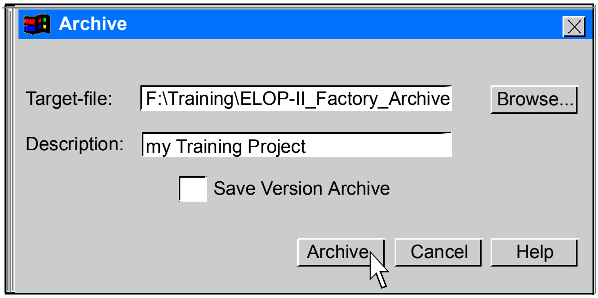
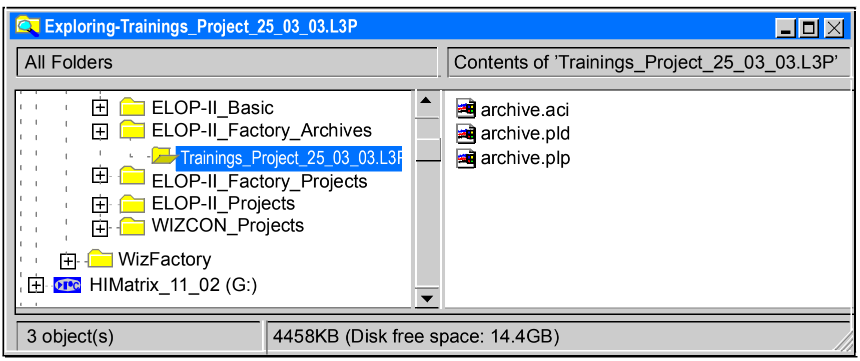
Step 1: Start the Archiving

Step	Action
1	Right-click on the **Project**.
2	In the context menu select **Archive...** Archiving a project 

Step 2: Assign a Name to the Archive

Step	Action
1	In the **Archive** window (Diagram below) 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 (following diagram). Finding archive path 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Enter the name of the archive file in the Object name field (without extension).
Setting archive name

</td></tr>
<tr><td>3</td><td>Enter a comment for your project into the Description field.</td></tr>
</table>

Step	Action
4	Click on the **Archive** button. Archive description  **Result**: The project is archived in the specified directory and carries the extension **.L3P**. The archive consists of three files. Archived project 

5.18 Restoring

Restoring a Project

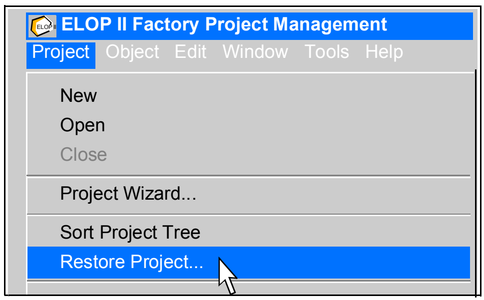
Requirement

To be able to restore a project from an archive in XPSMFWIN, no other project must be open.

Step 1: Start Restoring

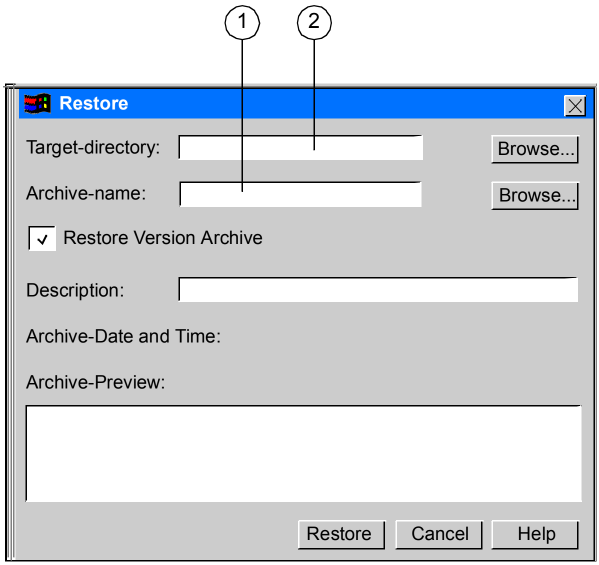
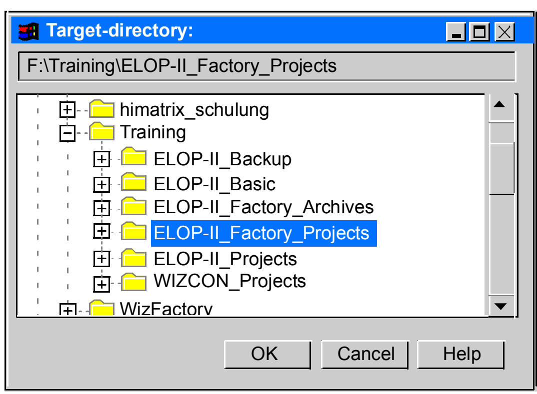
Start restoring a project

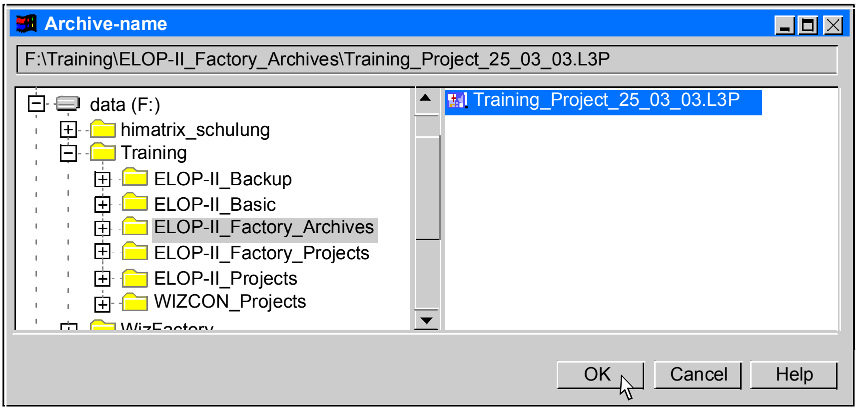
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Click Restore Project... in the Project menu.
Starting Restore Project...



Note: Project restoration can also be managed using the Project Wizard.</td></tr>
</table>

Step 2: Select the Project Select the project to be restored

Step	Action
1	In the **Restore** window specify the **Target-directory** where you want to restore the project. Use the **Browse...** button to select an existing directory. Restoring archive  **1** Path and file name of the project archive that is to be restored. **2** Path in which the restored project is to be saved. Selecting target-directory 
2	Click the **Browse...** button (for the archive name) to specify the path in a selection window (*Diagram below*) by clicking with the mouse.

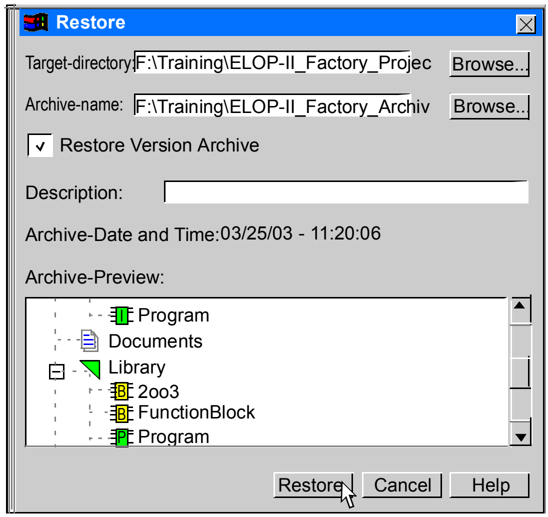
Step	Action
3	Highlight the path for your archive in the left side of the dialog window. The right side only shows the files that are detected as archive.
4	Highlight the desired archive.
5	Click **OK** to finish the selection. Selecting archive 

Step 3: Finishing Restore

Finish the restoring

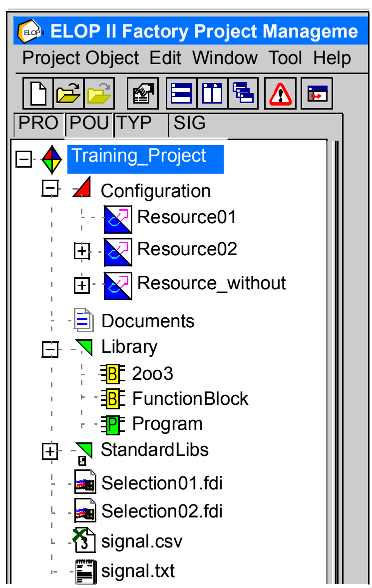
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>

Click on the **Restore** button.



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

Restored project



</td></tr>
</table>

Hardware Management

6

At a Glance

Overview

This chapter provides information on the Hardware Management.

What's in this Chapter?

This chapter contains the following sections:

6.1 Introduction

Hardware Management

Overview

The **Hardware Management** is the organization centre for creating all hardware related assignments.

The **Hardware Management** window is used for:
- viewing program size
- assigning resources PLC types (e.g. XPSMF40, XPSMF60 etc.)
- assigning Remote I/O modules to resources
- defining hardware inputs and outputs
- setting network parameters
- assigning non-safety protocols e.g. Modbus
- assigning signals
- defining diagnostics
- connecting to the online control panel
- defining user parameters
- downloading to hardware

6.2 Screen Layout of the Hardware Management

At a Glance

Overview

This section describes the screen layout of the Hardware Management.

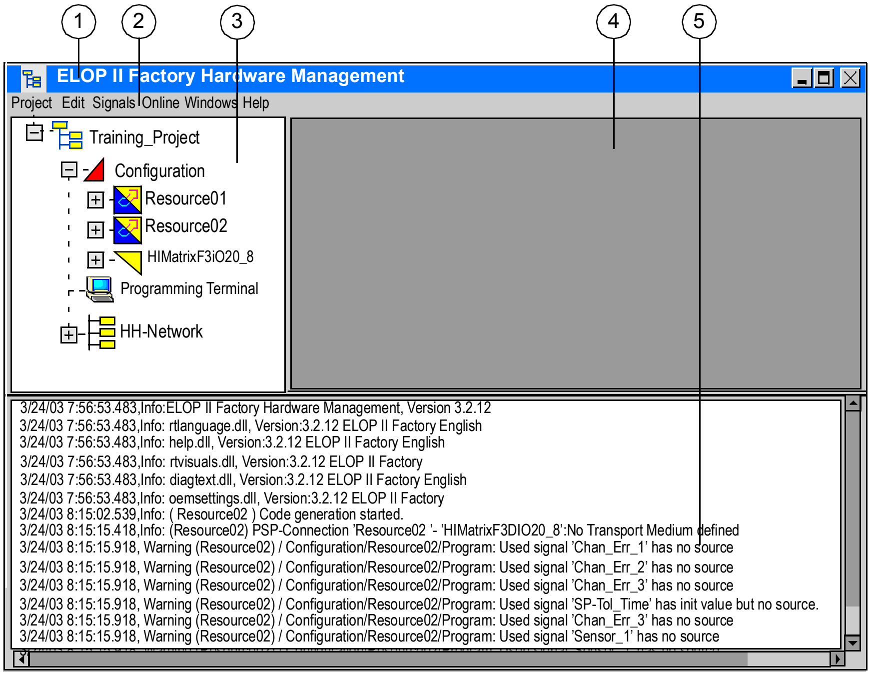
What's in this Section?

This section contains the following topics:

Screen Layout (Hardware Management)

Representation

The screen layout (hardware management) includes the following components:



1 Title Bar
2 Menu Bar
3 Structure Window
4 Working Area
5 Error State Viewer

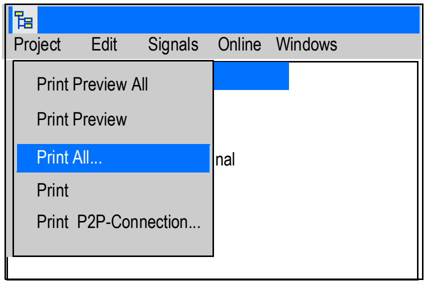
Hardware Management Menu Bar

The Hardware Management menu bar provides additional functionality:

Step	Action
1	Click on the menu to open it.
2	Select the command you want to execute.
3	Click the left mouse button. **Result:** The command is executed.

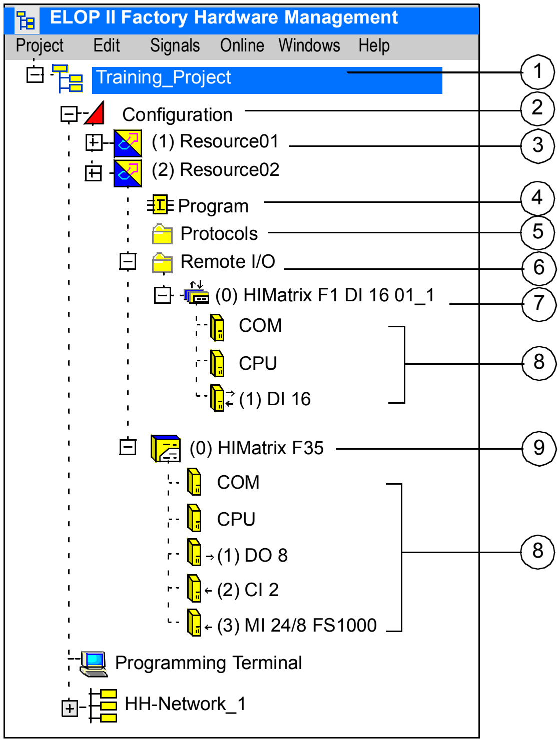
Example

Project menu



Structure Window (Hardware Management)

Representation



Elements of the structure window

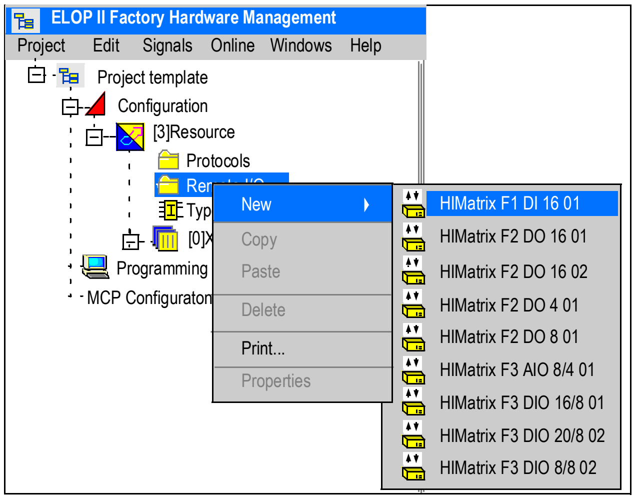
No.	Element	Symbols	Description
1	project name		name of the current project
2	configuration		provides all hardware information
3	resource folder		provides hardware information of a specific resource
4	program instance		attached program, provides size of application
5	communication protocols		Modbus/Profibus DP connection
6	remote I/O folder		allows user to add remote I/O devices to parent resource
7	remote I/O type		specific remote I/O module
8	components and modules		communication parameters (Ethernet) physical I/O connections
9	Resource Type		type of resource e.g. XPSMF40, XPSMF60 etc.

Context Menu (Hardware Management)

Overview

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.

Representation



Signal Editor (Hardware Management)

Overview

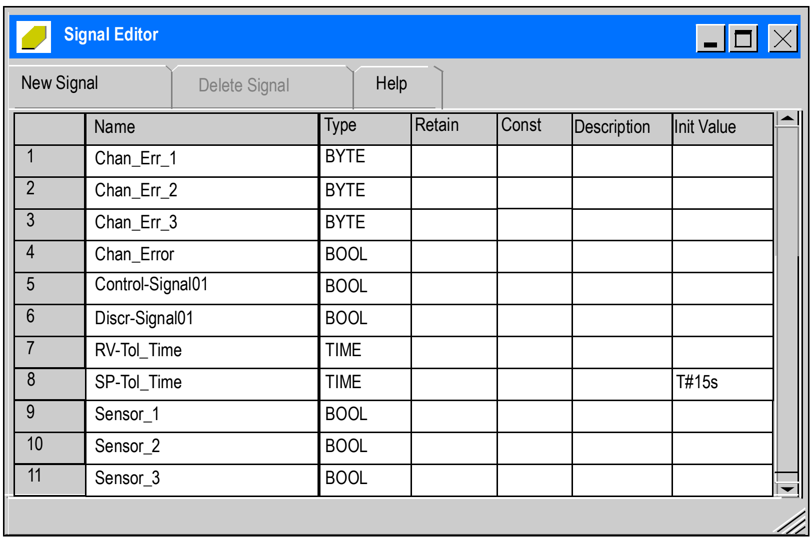
All variables that are to be transferred from one scope of validity (e.g. program) to another scope of validity (e.g. I/O level) must be given an assignment definition. This is done by using signals in the signal editor.

After the signal has been created in the signal editor, the signal is copied to the relevant areas by drag-and-drop.

> **Note:** All variables which require connection to physical I/O or to the external environment (e.g. via Modbus TCP/IP) must be configured as signals.
> All variables which do not require an external connection to I/O or to an external environment should be configured as variables.

Representation

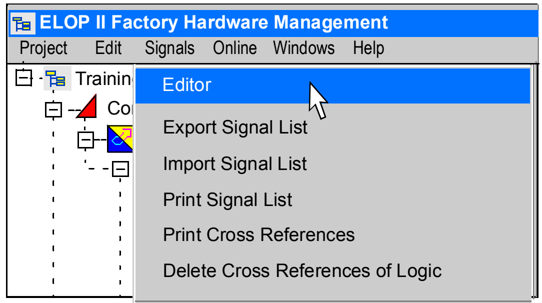
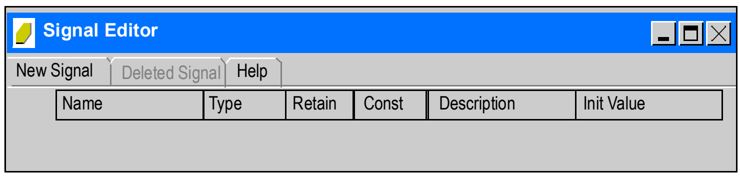
The **Signal Editor** is selected from the **Signals** menu.



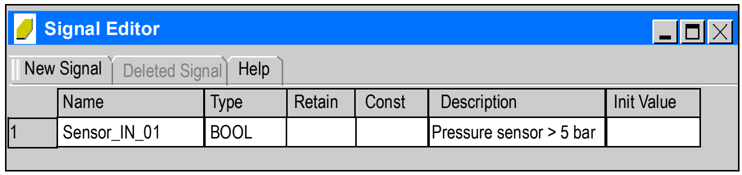
	Name	Type	Retain	Const	Description	Init Value
1	Chan_Err_1	BYTE				
2	Chan_Err_2	BYTE				
3	Chan_Err_3	BYTE				
4	Chan_Error	BOOL				
5	Control-Signal01	BOOL				
6	Discr-Signal01	BOOL				
7	RV-Tol_Time	TIME				
8	SP-Tol_Time	TIME				T#15s
9	Sensor_1	BOOL				
10	Sensor_2	BOOL				
11	Sensor_3	BOOL				

> **Note:** All signals can be imported from an Exel file, or see section *Additional Features for Programming Areas , p. 443* of this manual.

Step 1: Open the Signal Editor

Step	Action
1	Click on **Signals -> Editor** in the menu bar. Opening Signal Editor  Signal Editor without signals 

Step 2: Define Signal

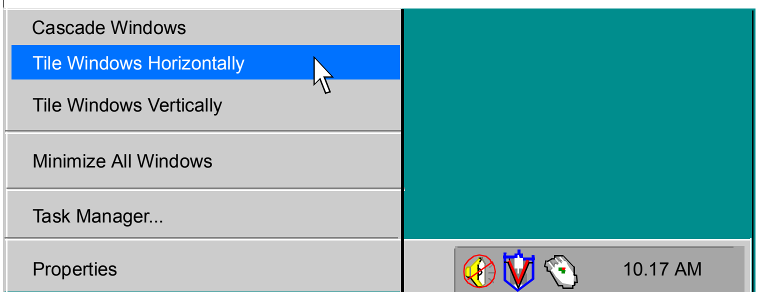
Step	Action
1	Click the **New Signal** button. **Result**: A new input line opens.
2	Input data into the **Name** and **Type** boxes. These inputs are required. 
3	**Retain** or **Constant** marks are set by a double-click into the field or cleared by another double-click.

Note: Take care about lower/upper case characters. The Signal Editor will not accept two signals with identical spelling.

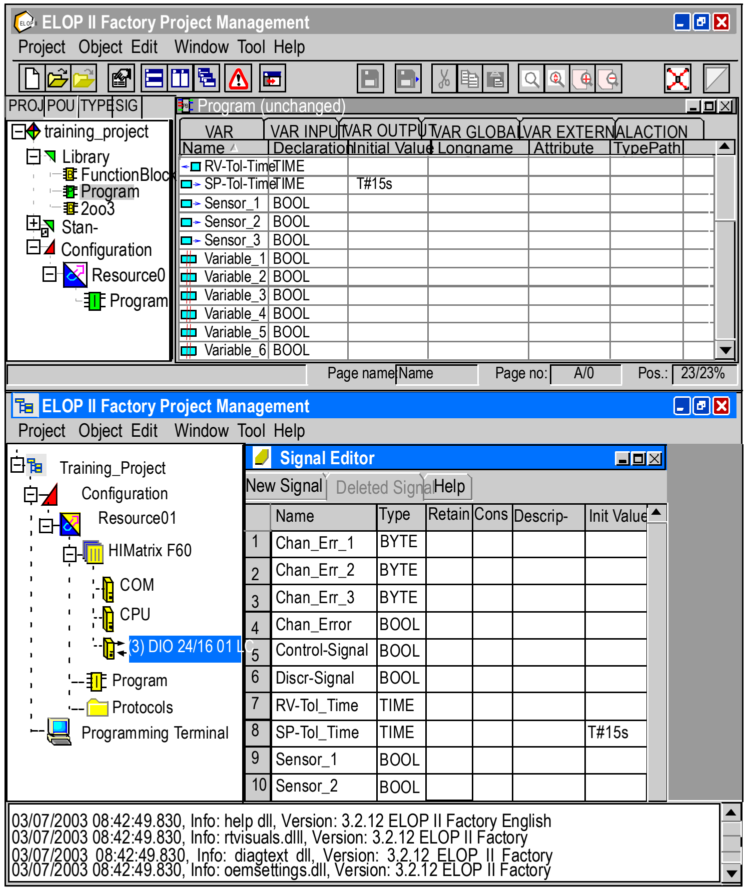
Note: Never enable **Retain** and **Constant** simultaneously. This will result in error messages during code generation.The user program must have read and write access to signals with the **Retain** attribute. However, write access to signals with the **Constant** attribute is not possible.

Step 3: Define Signal Use

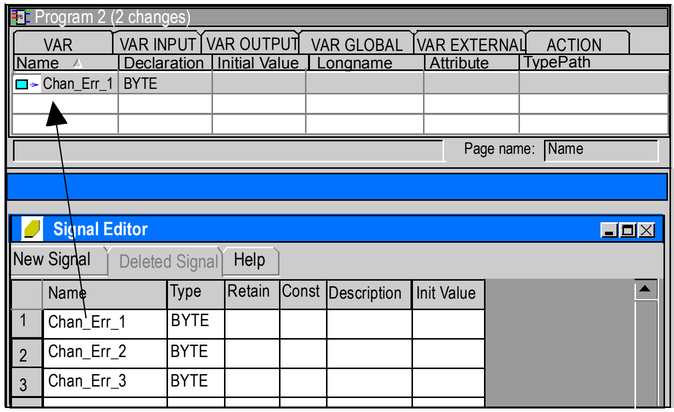
The use of signals in function blocks or in the user program is defined by drag-and-drop. Proceed as follows:

Step	Action
1	If you have applications open other than XPSMFWIN (e.g. e-mail program), minimize these applications.
2	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 diagram: Variable declaration and Signal Editor maximized*). Tile windows horizontally 
3	Enlarge the program or function block variable declaration pane so you can see as many variables as possible (*see diagram below, top*).

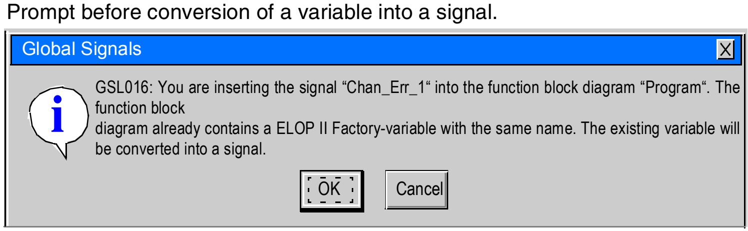
Step	Action
4	Enlarge the signal list pane in the Signal Editor so you can see as many signals as possible (*see diagram below, bottom*). Variable declaration and Signal Editor maximized



<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>5</td><td>
Dragging-and-dropping signals into the variable declaration.



Drag a signal from the Signal Editor to the variable declaration or directly to the logic. The signal is automatically created in the `VAR_EXTERNAL` tab (*see diagram above*).
??? Alternatively you can open the drawing area as wide as possible and drag the signals directly into the drawing area. ???
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.
Prompt before conversion of a variable into a signal.



</td></tr>
</table>

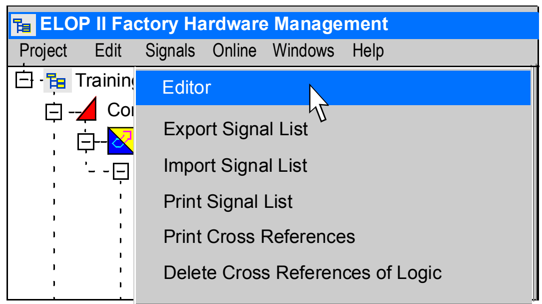
Defining Signals

Overview

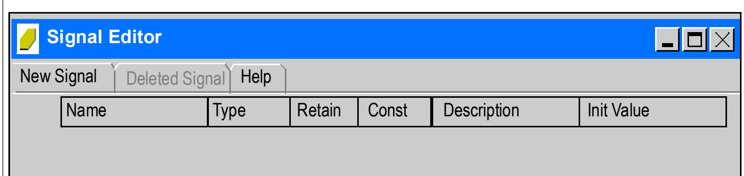
The **Signal Editor** is used to define signals.

Step 1: Open the Signal Editor

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Click on **Signals -> Editor** in the menu bar.
Opening Signal Editor

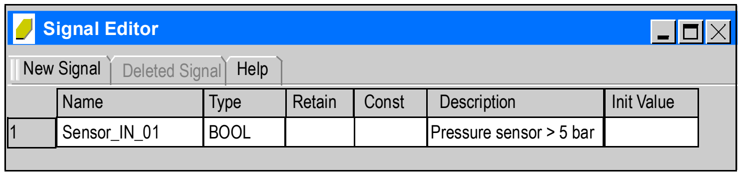


Signal Editor without signals



</td></tr>
</table>

Step 2: Define Signal

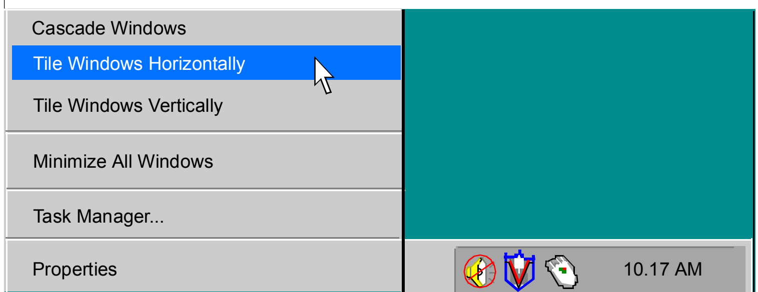
Step	Action
1	Click the **New Signal** button. **Result**: A new input line opens.
2	Input data into the **Name** and **Type** boxes. These inputs are required. 
3	**Retain** or **Constant** marks are set by a double-click into the field or cleared by another double-click.

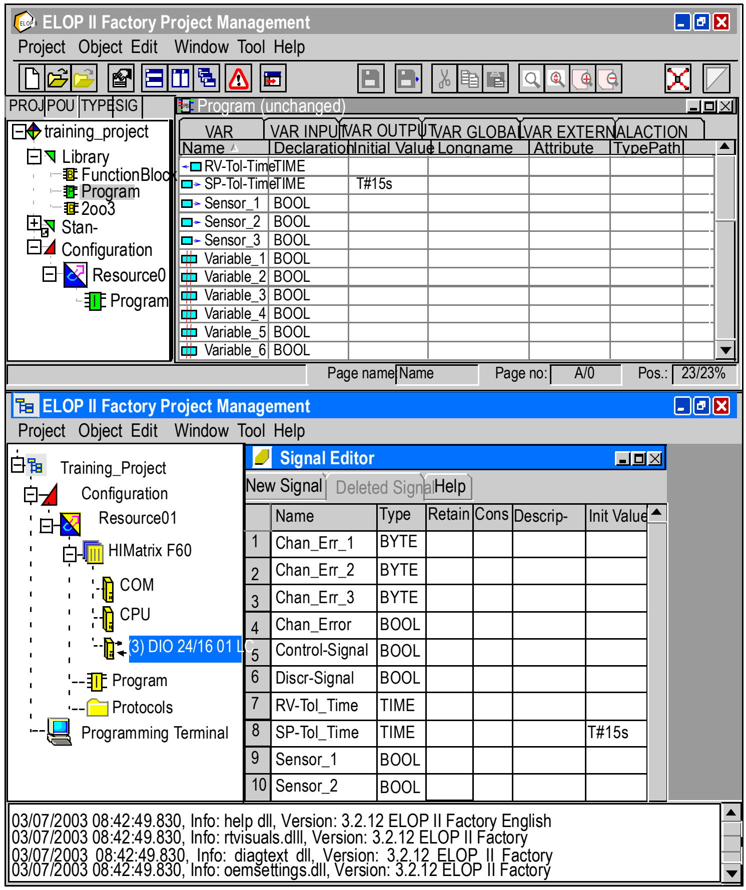
Note: Take care about lower/upper case characters. The Signal Editor will not accept two signals with identical spelling.

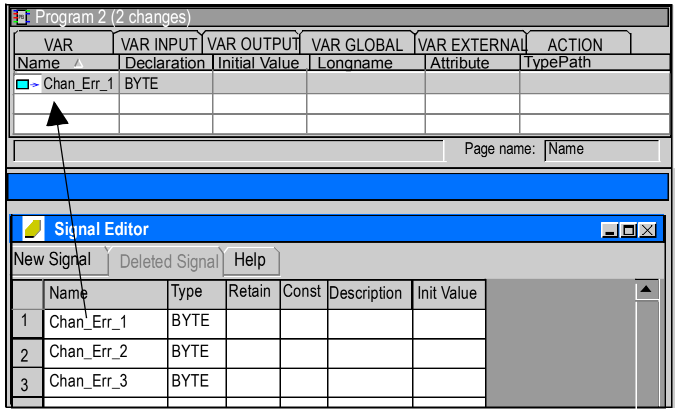
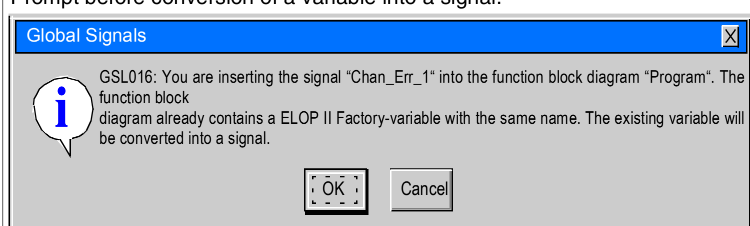
Note: Never enable **Retain** and **Constant** simultaneously. This will result in error messages during code generation.The user program must have read and write access to signals with the **Retain** attribute. However, write access to signals with the **Constant** attribute is not possible.

Step 3: Define Signal Use

The use of signals in function blocks or in the user program is defined by drag-and-drop. Proceed as follows:

Step	Action
1	If you have applications open other than XPSMFWIN (e.g. e-mail program), minimize these applications.
2	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 diagram: Variable declaration and Signal Editor maximized*). Tile windows horizontally 
3	Enlarge the program or function block variable declaration pane so you can see as many variables as possible (*see diagram below, top*).

Step	Action
4	Enlarge the signal list pane in the Signal Editor so you can see as many signals as possible (*see diagram below, bottom*). Variable declaration and Signal Editor maximized 

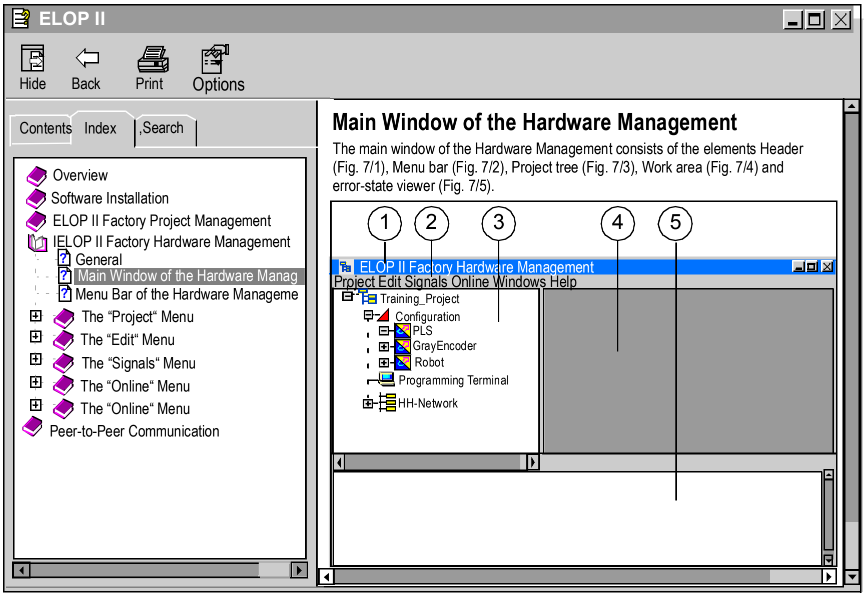
Step	Action
5	Dragging-and-dropping signals into the variable declaration.  Drag a signal from the Signal Editor to the variable declaration or directly to the logic. The signal is automatically created in the `VAR_EXTERNAL` tab (*see diagram above*). Alternatively you can open the drawing area as wide as possible and drag the signals directly into the drawing area. **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. Prompt before conversion of a variable into a signal. 

Online Help (Hardware Management)

Overview

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

Representation



6.3 Objects in the Structure Window (Hardware Management)

At a Glance

Overview

This section describes the objects in the hardware management structure window.

What's in this Section?

This section contains the following topics:

Structure Window, General

Overview

All objects in a project are displayed in their hierarchical structure (see *Structure Window (Hardware Management), p. 134*) and managed in the structure window.

Configuration (Hardware Management)

Overview

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

Resource

The resource is the term specified in IEC 61131-3 for a target system that executes the controller task, in this case the safety PLC (XPSMF40, XPSMF60, HIMatrix F30, F31, F35).

A resource is created within a configuration.

The above icon of a resource corresponds to the opposite part of the resource created in the Project Management .

Remote I/Os can communicate only with one resource (**parent** resource) represented by the icon. In the structure window these remote I/Os are located in the hierarchy below the parent resource in the **Remote I/O** folder.

See *Structure Window (Hardware Management), p. 134*.

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 the same symbol but it has no link to a program type.

The type instance includes the logic.

6.4 General Working With Resource Types

General Working With Resource Types

Overview

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

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).

Management

The different features of a resource are managed either in the
- Project Management or in the
- Hardware Management.

Project Management

Handle the following features in the **Project Management**:
- creating resources
- setting resource name
- creating a program for the resource
- assigning a program to the resource

Hardware Management

Handle the following features in the **Hardware Management**:
- setting resource properties, e.g. type, safety time, etc.
- adding remote I/O
- setting communication parameters
- setting I/O parameters
- downloading to hardware

6.5 Properties of a Resource

At a Glance

Overview

This section describes how to set the properties of a resource.

What's in this Section?

This section contains the following topics:

Setting the Properties of a Resource

Creating a Resource

A resource (e.g. HIMatrix F30, F31, F35, XPSMF40, XPSMF60) is created in the **Project Management**. See *Creating and Renaming a Resource, p. 103*.

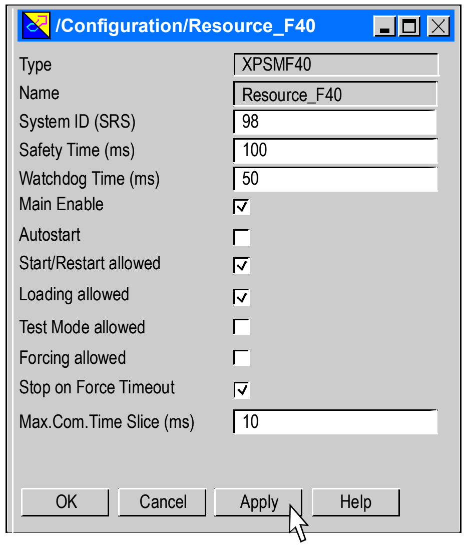
The resource (PLC) type is assigned within the Hardware management window.

Change Resource Type

Note: Once defined, to change the resource type, you must delete the resource and create a new resource.
Ensure that when programming on a type instance, the program is copied to a program before deleting.

Setting the Properties

The resource properties are set in the **Hardware Management**.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>In the Hardware Management right-click on the resource name and select Properties in the context menu.</td></tr>
<tr><td>2</td><td>Enter the resource type using the Type box, e.g. XPSMF40, XPSMF60, F30, F31 etc.</td></tr>
<tr><td>3</td><td>Enter a value greater than 1 in the System ID [SRS] field.</td></tr>
<tr><td>4</td><td>Confirm your input with Apply.


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.</td></tr>
</table>

SRS (System Rack Slot) Value

> **Note:** The SRS of the programming terminal is **1**.The SRS is used to structure the hardware into a modular internal architecture.
> The following System Rack Slot (SRS) names refer to the following safety processing components:
> System = Resource
> Rack = Remote I/O
> Slot = Modular PLC I/O Cards

Use values from 2 to 65,535 for resources and from 1 to 511 for remote I/Os.

This value is only for program related functionality placing the PLC or remote I/O device in the correct hierarchical order.

The SRS value is a user defined value.

Assigning SRS Value to Safety PLC

Defining your SRS is simple to do.

The assignment for a resource (Safety PLC) is done by simply providing a number within the range of 2...65535.

Once you have given the resource a SRS within the **Properties** window of the resource (see *Setting the Properties, p. 155*) the value is added before the resource name in the **Hardware Management** structure tree.

> **Note:** This SRS value cannot be used again on another resource.
> Example:
> You have 2 resources. If you assign the first resource with a SRS of 2, then the second resource can have any value above the value of 2.

Assigning SRS Value to Remote I/O

The SRS assignment with remote I/O modules is managed in the same way as the SRS assignment with the Safety PLC, however this value is independent to the resource value.

Remote I/O modules can have any value between 1 and 511.

The link between the resource and remote I/O modules is managed by two factors:
- IP address, and
- SRS.

This can bee seen within the communication settings section of this manual (see *Setting IP Address to a PLC or Remote I/O, p. 183* for setting the communication of resources and remote I/O).

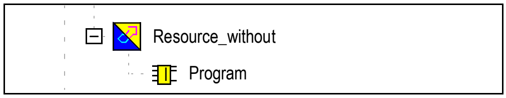
Properties

Property	Description
Type	resource type; model number such as XPSMF40
Name	resource name; can only be altered in the **Project Management** window
System ID (SRS)	SRS (system rack slot) is a user defined value. The value allocates a position within the system, therefore the SRS value can only be used once within a project. Any value between 2 to 655535 for resources and 1...511 for remote I/O is acceptable. This value is completely user defined. Once a value is used it cannot be used on another resource or remote I/O in the same project.
Safety Time (ms)	The maximum time a PLC may require in order to react to the change of an input signal with an output signal and within which a PLC must react to an error. The default setting for a Resource is 50 ms and remote I/O is 20 ms and can be adjusted by the user within the range of 20 to 50,000 ms, depending on the users´ safety requirements. It must ba a minimum of 2*watchdog time
Watchdog Time (ms)	The maximum time which a resource is allowed to execute one program cycle. If the watchdog time is exceeded (execution of a program cycle takes too long), the resource goes into ERROR STOP. The following boundary values apply to the watchdog time (WDZ): • WDZ ≥ 10 ms • WDZ ≤ 0.5 * safety time • maximum WDZ = 5,000 ms The default setting for a resource is 25 ms and for a remote I/O is 10 ms (0.5 x safety time) and can be adjusted by the user depending on the users' safety requirements.
Main Enable	The setting of this CPU switch affects the function of the other CPU switches. If **Main Enable** is switched off, the settings of the other CPU switches cannot be changed whilst the user program is being processed (PLC in RUN).
Autostart	Activates the automatic start-up of the user program after turning on the power, when activated there are two possible further options to select (within the properties menu of the program instance in **Hardware Management**): • **Cold Start** On start-up all variables and signals are reset to their initial values. • **Warm Start** On start-up the signals with the **Retain** attribute keep their current value.
Start/Restart allowed	If **start/restart allowed** is switched on, a user program can be started manually using the control panel. If **start/restart allowed** is switched off, the user program in the resource cannot be manually started from the PC. If this is the case, the user program can only be started if **Autostart** is activated and the resource is switched on or rebooted.
Loading allowed	If deactivated, no (new) user program can be loaded into the PLC. It should be deactivated if the user program (loaded into the PLC) is not to be overwritten.

Property	Description
Test Mode allowed	If switched on, the program instance can be switched to test mode. In test mode, the execution of the program instance is first stopped so that it can be executed manually in single step mode. **Note:** Switching the PLC into test mode during normal operation of a system is not permitted.
Online Test allowed	This is a function of the project management and is used to monitor the values of all variables and signals of a logic within the function block diagram editor while the PLC is running. If **Online Test allowed** is switched off, you cannot actively affect the behavior of the logic by use of online test fields. The online test can also be invoked (display only) if **Online Test allowed** is switched off.
Forcing allowed	Enables the use of forcing signals to user defined values. This mode should only be used during the testing phase.
Stop on Force Timeout	When forcing has completed the cycle, it will stop the program from running.
max.Com.Time Slice (ms)	Is a parameter required for the peer-to-peer communication. The maximum communication time slice is the time reserved for each CPU cycle to process all existing communication tasks. The communication time slice is dependant on the number of devices on the network.

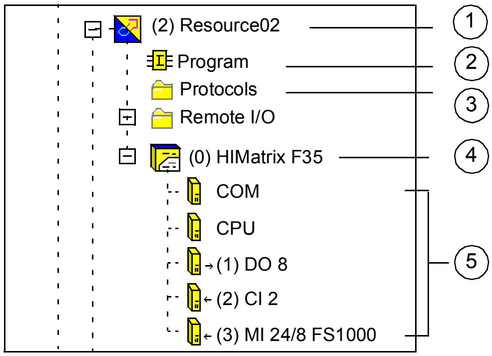
Structure without a Defined Resource Type

Resource structure without defined resource type



Structure with a Defined Resource Type

Structure of an F35



Legend

No.	Description
1	Individual Resource Name
2	Program Instance
3	Communication Protocols
4	Resource Type
5	Components of the Controller Ethernet Communications ● CPU Signal Information ● Input/Output Signal Connection

Program Property Autostart Enable

Autostart Property

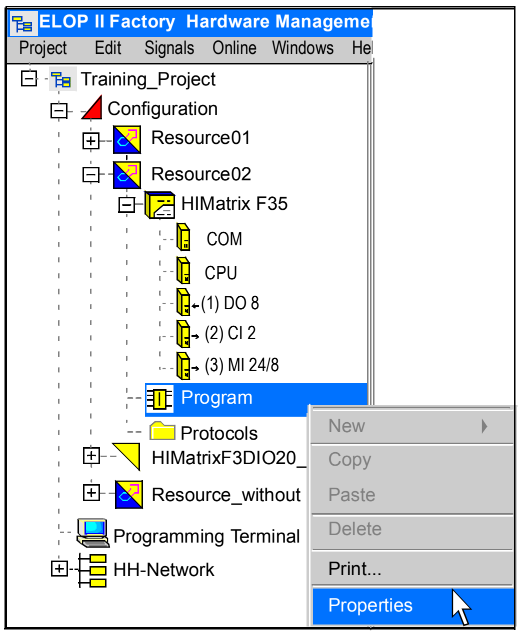
The property **Autostart Enable** activates the automatic start-up of the user program after switching on the power.

Coldstart/ Warmstart

Whether and how the user program starts (coldstart or warmstart) is defined in the properties of the program instance in the Hardware Management.

Program Instance Properties

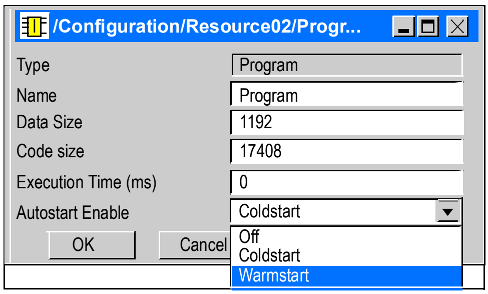
To open the properties dialog box proceed as follows:

Step	Action
1	In the resource right-click **Program** and select **Properties**. 

Settings

The following settings for **Autostart Enable** are possible:

Warmstart	Signals with the **Retain** attribute keep their current value.
Coldstart	All variables and signals are reset to their initial values.
Off	No automatic start. The user program must be started from the programming terminal.



6.6 Signals

Signals

Overview

Definition of the difference between signal and variable.

Signals

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 by drag-and-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.

Variable

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

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-and-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 by drag-and-drop. This also applies for all variables that do not acquire a hardware reference but whose value must be monitored in the Force Editor during operation, because the value must leave the area of the program at this moment.

6.7 I/O Module Assignment

At a Glance

Overview

This section describes how to assign I/O modules in the Hardware Management.

What's in this Section?

This section contains the following topics:

Assigning Modules to the XPSMF60 Modular Safety PLC

Overview

I/O modules for the modular XPSMF60 safety PLC are assigned in the **Hardware Management**.

> **Note:** When assigning the XPSMF60 type, the rack housing power supply and CPU do not need to be defined. Only the I/O cards types and slot numbers need to be defined.

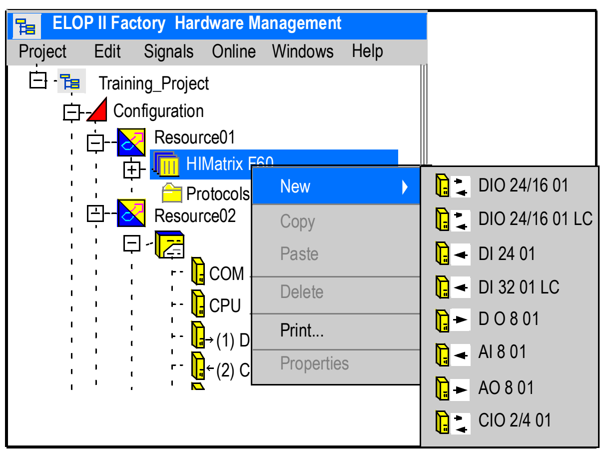
Maximum Number of Modules

Up to 6 I/O modules and maximum 64 remote I/Os can be assigned to a modular system (e.g. XPSMF60).

A compact system (e.g. HIMatrix F30, F31, F35, and XPSMF40) can be extended with a maximum of 64 remote I/Os and they require no I/O module assignments as they are predefined.

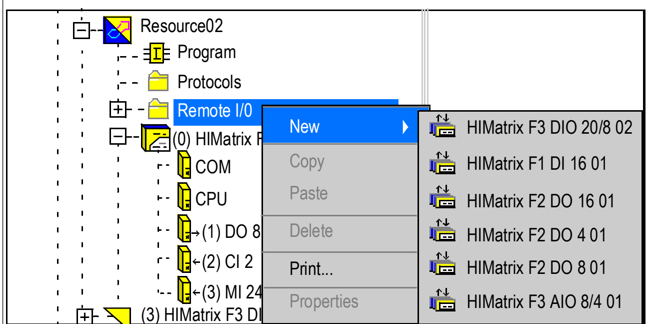
Assigning I/O Modules

To assign input/output modules proceed as follows (only for XPSMF60):

Step	Action
1	Expand the structure of the resource in the Hardware Management and open the context menu of the resource type with a right mouse click.
2	Select **New** and then the desired I/O module. 
3	Double-click on the new module.
4	Enter the correct slot number in the modules' properties.

Assigning Remote I/O

To assign remote I/Os proceed as follows (for compact and modular PLCs):

Step	Action
1	Expand the structure of the resource in the Hardware Management and open the context menu of the directory **Remote I/O** with a right mouse click.
2	Select **New** and then the desired **Remote I/O**. 
3	Open the context menu of the **Remote I/O** and select **Properties**.
4	Set the **Rack ID** to a value greater than zero.

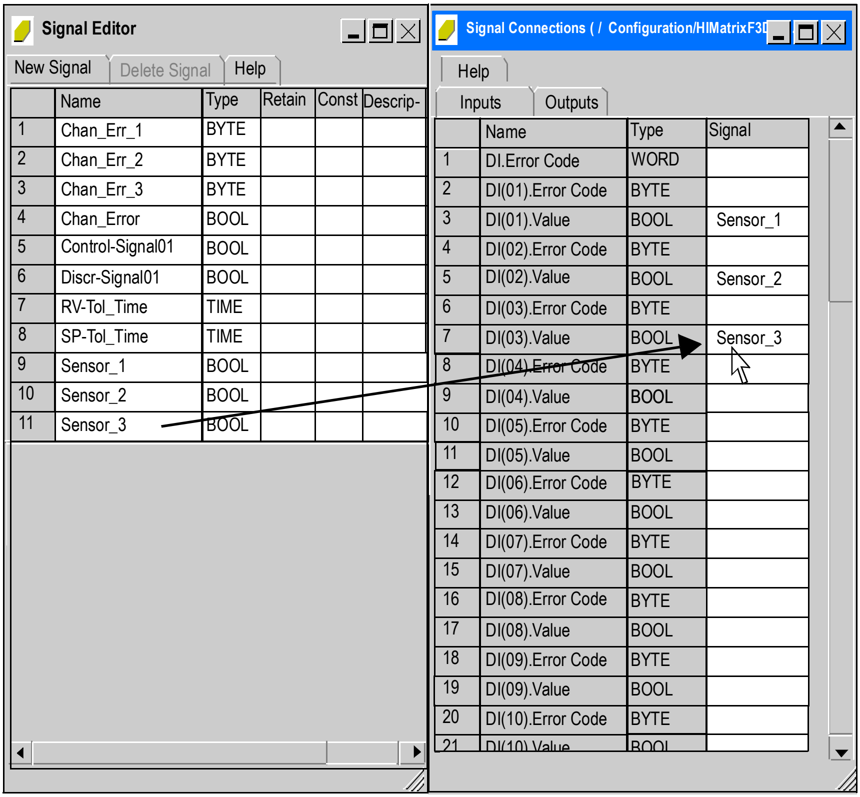
Assigning Signals to the I/O Channels

Overview

Signals that have been defined in the Signal Editor can be assigned to the hardware input/output channels.

Drag-And-Drop of Signals to Inputs/Outputs

Step	Action
1	Open the Signal Editor in the **Signals** menu.
2	Open the context menu of a module with a right mouse click and select **Connect Signals**.
3	Position the two windows side by side.

Step	Action
4	Drag signals from the Signal Editor and drop them on the desired channels.  **Note**: It is possible to select several signals at the same time and drag-and-drop them together 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. **Note**: In XPSMFWIN 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.

Input Module/ Output Module

	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

Assigning System Signals (Resource/Remote I/O)

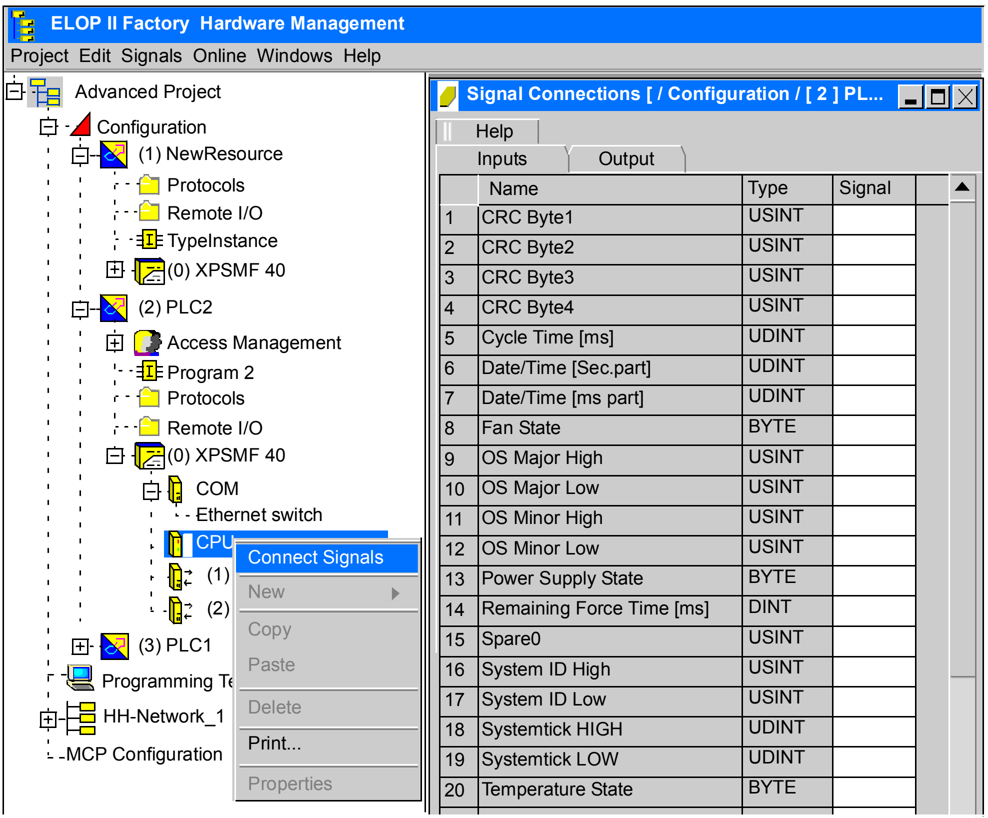
Overview

System signals are signals that contain information on the status of the CPU of the resource and the remote I/O modules.

The procedure is identical with that for assigning I/O channels as described in *Assigning Signals to the I/O Channels, p. 168*

Input System Signals

Input system signals



The types of input system signals are used in the following ways.

Name	Type	Description
CRC Byte	USINT	Cyclic redundancy check signals, stored from within PLC, are used to monitor any changes. Can be sent via Modbus to standard PLC for diagnostics.
Cycle Time (ms)	UDINT	CPU cycle time
Date/Time (Sec.part)	UDINT	useful for including in programs
Fan State	BYTE	only on XPSMF60 modular safety PLC 2 states: ● 0 = fan running (normal operation) ● 1 = error (must be replaced)
OS Major High/Low	USINT	operating system details
Power Supply State	BYTE	Can be monitored to ensure no loss in power. 2 states: ● 0 = Power supply okay ● 1 = Problem with supply
Remaining Force Time (ms)	DINT	Used only during testing phase. Can be used in program.
Temperature State	BYTE	temperature state monitoring of PLC Used to ensure PLC is operating within correct parameters. 4 states: ● 0 = normal ≈ < 40° C ambient T ● 1 = high temperature ≈ > 40°C, < 50°C ambient T possible additional cooling required ● 2 = error in reading temperature ● 3 = very high temperature ≈ > 50°C ambient T additional cooling required

Types of Output System Signals

Name	Type	Description
Emergency Stop 1 (2, 3, 4)	BOOL	Can be used to stop PLC in the event of a failure, e.g. monitoring CRC values via a standard PLC. When values are not equal, the standard PLC can send a signal to stop safety PLC.
Relay Contact 1 (2, 3, 4)	BOOL	Used only within XPSMF60 (XPSMFCPU22) for monitoring relay state on power supply.

Line Control Setup

Overview

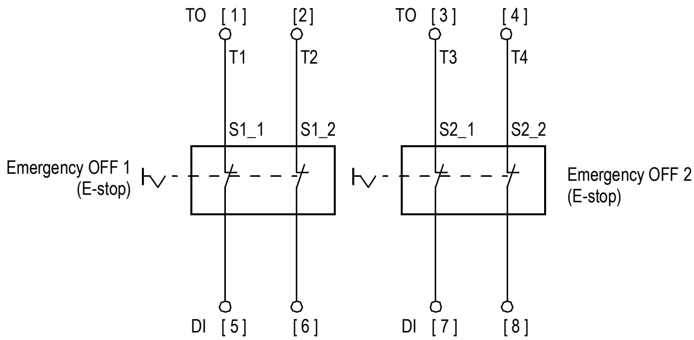
Line control signal assignment is used for short circuit and line break monitoring.

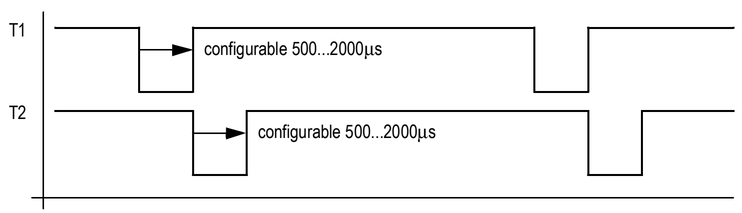
To reach SIL 3 according to EN/IEC 61508, Category 4 according to EN 954-1 you must use this feature.

Connection

Line control outputs should be connected to the digital inputs of the same system.

The following diagram shows connection of digital outputs and digital inputs.

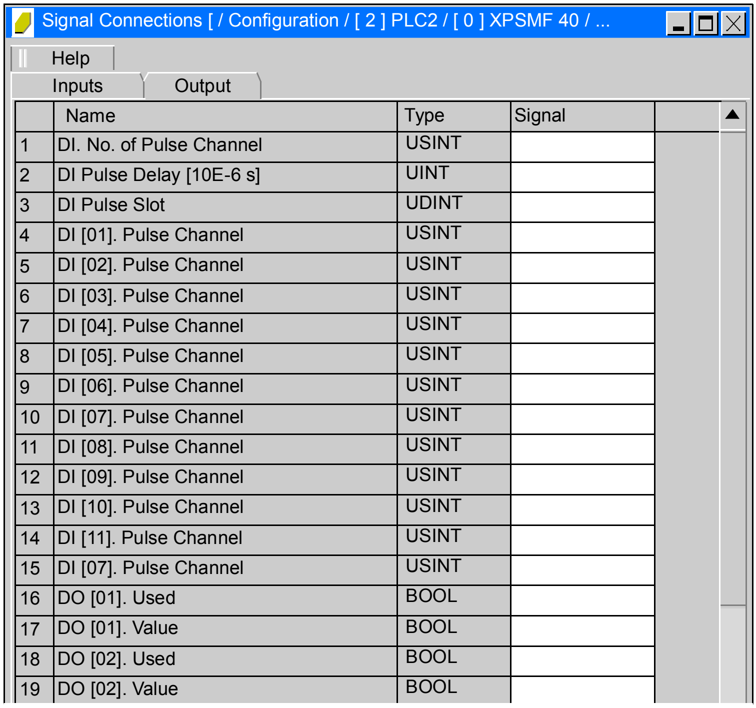




TO Line Control Output
DI Digital Input

Requirements

Click on the **Output** tab from the **(1) DIO 12/12** of the XPSMF40 safety PLC.


Signal Connections [/ Configuration / [2] PLC2 / [0] XPSMF 40 / ...

Help

Inputs | Output

	Name	Type	Signal
1	DI. No. of Pulse Channel	USINT	
2	DI Pulse Delay [10E-6 s]	UINT	
3	DI Pulse Slot	UDINT	
4	DI [01]. Pulse Channel	USINT	
5	DI [02]. Pulse Channel	USINT	
6	DI [03]. Pulse Channel	USINT	
7	DI [04]. Pulse Channel	USINT	
8	DI [05]. Pulse Channel	USINT	
9	DI [06]. Pulse Channel	USINT	
10	DI [07]. Pulse Channel	USINT	
11	DI [08]. Pulse Channel	USINT	
12	DI [09]. Pulse Channel	USINT	
13	DI [10]. Pulse Channel	USINT	
14	DI [11]. Pulse Channel	USINT	
15	DI [07]. Pulse Channel	USINT	
16	DO [01]. Used	BOOL	
17	DO [01]. Value	BOOL	
18	DO [02]. Used	BOOL	
19	DO [02]. Value	BOOL	

The first 3 signals are constants:

Signal	**Description**
DI No. of Pulse Channel	number of digital inputs which use line control
DI Pulse Delay (10E-6 s)	between 500 and 2000 microseconds dependant on the length of cable (normal 500 microseconds)
DI Pulse Slot	where the line control output is originating from slot 1, 2 or 3.

Identify what you require:

Signal	Requirement
DI No. of Pulse Channel	Count the number of inputs which require line control signals. In your example you will use only 2 for **EStop 3 Input 1** and **EStop 3 Input 2**.
DI Pulse Delay (10E-6 s)	This is typically found by experience, however within a typical application the pulse delay should be anywhere between 500 and 1500 microseconds. In the example above you will use 1000 microseconds.
DI Pulse Slot	Use the following table to help you identify which line control outputs slot you are using depending on hardware. For your example use slot 1.

Hardware/Slot

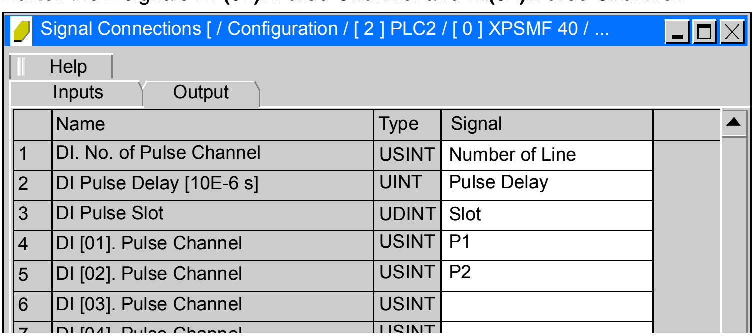
Line control outputs slot depending on hardware

Hardware	Slot
XPSMF40**	for terminals 1-12: slot 1 for terminals 13-24: slot 2
XPSMF3022	outputs configurable as line control: slot 2
XPSMF31222	outputs configurable as line control: slot 2
modular PLC	varies depending on location of I/O card XPSMFDIO241601
XPSMF1DI1601 (remote I/O)	slot 1
XPSMF3DIO8801 (remote I/O)	slot 3
XPSMF3DIO16801 (remote I/O)	first line control output slot 1 second line control output slot 2
XPSMF3DIO20802 (remote I/O)	slot 2

Activating Line Control

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Create 3 new signals within the hardware management Signal Editor.
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Retain</th><th>Const.</th><th>Description</th><th>Init Value</th></tr>
<tr><td>56</td><td>Number of Line control inputs</td><td>USINT</td><td></td><td>✓</td><td>Number of Line control inputs F40</td><td>2</td></tr>
<tr><td>57</td><td>Pulse delay</td><td>UINT</td><td></td><td>✓</td><td>Pulse delay time</td><td>1000</td></tr>
<tr><td>58</td><td>Slot</td><td>UDINT</td><td></td><td>✓</td><td>area where the pulse is originating</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>On the Output tab from the (1) DIO 12/12 of the XPSMF40 safety PLC drag and drop from the Signal Editor the signals into the appropriate I/O slots:
Signal Connections [/ Configuration / [2] PLC2 / [0] XPSMF 40 / ...
Help
Inputs | Output
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Signal</th></tr>
<tr><td>1</td><td>DI. No. of Pulse Channel</td><td>USINT</td><td>Number of Line</td></tr>
<tr><td>2</td><td>DI Pulse Delay [10E-6 s]</td><td>UINT</td><td>Pulse Delay</td></tr>
<tr><td>3</td><td>DI Pulse Slot</td><td>UDINT</td><td>Slot</td></tr>
<tr><td>4</td><td>DI [01]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>5</td><td>DI [02]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>6</td><td>DI [03]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>7</td><td>DI [04]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>8</td><td>DI [05]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>9</td><td>DI [06]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>10</td><td>DI [07]. Pulse Channel</td><td>UDINT</td><td></td></tr>
<tr><td>11</td><td>DI [08]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>12</td><td>DI [09]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>13</td><td>DI [10]. Pulse Channel</td><td>USINT</td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>To define different pulses for each of the line control inputs, you need to define new signals within the Signal Editor.
The convention should be as follows:
● Pulse1: USINT value 1
● Pulse2: USINT value 2
● Pulse3: USINT value 3
● Pulse4: USINT value 4
● Pulse5: USINT value 5
● Pulse6: USINT value 6
● Pulse7: USINT value 7
● Pulse8: USINT value 8

The name is user defined, however the values must be incremental according to the number of line control outputs available from the safety PLC or remote I/O.
Within the XPSMF40 safety there are 2 sets of 4 line control outputs.
Therefore numbering can only be from 1 to 4 on each of the line control terminals, e.g.
● (1) DIO 1/12: line control signal values 1 to 4
● (2) DIO 13/24: line control signal values 1 to 4</td></tr>
<tr><td>4</td><td>In the example above, you only require 2 different line control signals, therefore create the following 2 new signals:
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Retain</th><th>Const.</th><th>Description</th><th>Init Value</th></tr>
<tr><td>59</td><td>P1</td><td>USINT</td><td></td><td>✓</td><td>Pulse signal 1</td><td>1</td></tr>
<tr><td>60</td><td>P2</td><td>USINT</td><td></td><td>✓</td><td>Pulse signal 2</td><td>2</td></tr>
</table></td></tr>
<tr><td>5</td><td>On the output tab from the (1) DIO 12/12 of the XPSMF40 safety PLC drag and drop from the Signal Editor the 2 signals DI (01). Pulse Channel and DI(02).Pulse Channel.


This now links pulse value 1 (P1) to EStop 3 Input 1 and pulse value 2 (P2) to EStop 3 Input 2.
Note: In this case the 2 pulses must be of different values as you are supplying the input of 2 emergency stop contacts of the same emergency stop control. In order to detect any short circuit condition, line break or wrongly wired connection you must use 2 different signals.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>To activate the line control outputs, you need to turn on 2 of the 4 line control outputs.
These outputs are listed within the Output tab from the (1) DIO 12/12 of the XPSMF40 safety PLC with the following names:
TO(01).Value
TO(02).Value
TO(03).Value
TO(04).Value</td></tr>
<tr><td>7</td><td>You will use only TO(01) and TO(02).
TO(01) provides the pulse for P1
TO(02) provides the pulse for P2
To activate the line control use the signal ON from the Signal Editor and drag this into T0(01).Value and TO(02).Value slots.<table><tr><th></th><th>Name</th><th>Type</th><th>Signal</th></tr><tr><td>40</td><td>TO [01]. Value</td><td>BOOL</td><td>ON</td></tr><tr><td>41</td><td>TO [02]. Value</td><td>BOOL</td><td>ON</td></tr><tr><td>42</td><td>TO [03]. Value</td><td>BOOL</td><td></td></tr><tr><td>43</td><td>TO [04]. Value</td><td>BOOL</td><td></td></tr></table>Once done, the line control monitoring has been set up.The line control monitoring is hardware dependant, and therefore does not require any settings within the Project Management area of the software.</td></tr>
<tr><td>8</td><td>Finish assigning all the remaining signals onto the hardware I/O and close any open panes within the software.</td></tr>
</table>

6.8 Communication Settings

At a Glance

Overview

This section describes the communication between the programming terminal and the PLC.

What's in this Section?

This section contains the following topics:

Communication between Programming Terminal and PES

Overview

Communication between the programming terminal and the PLCs is carried out over Ethernet. Communication over Ethernet uses the TCP/IP protocol.

IP Address

Therefore you have to assign an IP address to each PLC in the network (in the **Properties** of the COM). IP addresses are logical addresses and do not have fixed assignments to communication interfaces of PLCs. Only the MAC address - as physical address - is permanently assigned to the communication interface. The MAC address is programmed firmly with the production of the PLC.

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

Content of the IP Address

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 value 0 is not allowed.

⚠ CAUTION

LOSS OF COMMUNICATION

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

Failure to follow these instructions can result in injury or equipment damage.

Network Parameters

If the programming terminal and the PLCs are in their own, closed network, the network parameters can be freely defined.

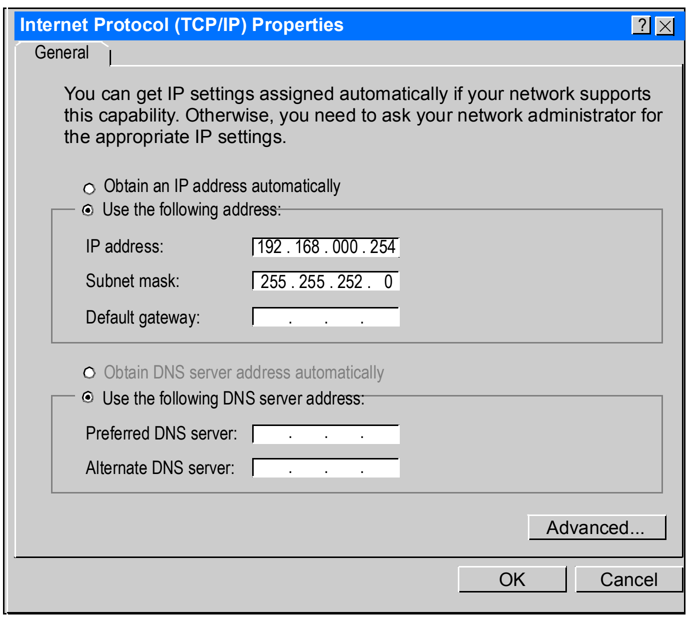
We recommend leaving the default setting of the Subnet Mask unchanged at `255.255.252.0` and setting the IP address to conform with the layout `192.168.0.xxx`, where `xxx` represents the resource node.

> **Note:** If the programming terminal and the PLCs are part of a network that is also used by others, contact your network administrator for assignment of IP addresses.

Configuring the Programming Terminal for Communication

Setting the IP Address

Set the IP address of the programming terminal (PC)

Step	Action
1	Open the Control Panel of the PC with **Start -> Settings -> Control Panel**.
2	Double-click **Network and Dial-Up Connections** (Windows 2000).
3	Open the properties for the TCP/IP protocol.
4	If the Programming Terminal has more than one network card, make the correct setting in the **Network card** field.
5	Input the IP address as specified in the project into the **IP Address** field of the programming terminal. **Note**: Note that the IP address of the programming terminal must be in the same address range as the IP addresses for the resources. It is recommended that the IP address be structured in the form 192.168.0.xxx.
6	Input a valid subnet mask in the **Subnet Mask** field. **Note**: Note that the subnet mask of the programming terminal must be in the same address range as the Subnet Mask for the resources. We recommend setting the Subnet Mask to 255.255.252.0.
7	The **Default Gateway** field should not contain any entries. Setting the IP address of the programming terminal. 

Setting IP Address to a PLC or Remote I/O

Disable Firewalls

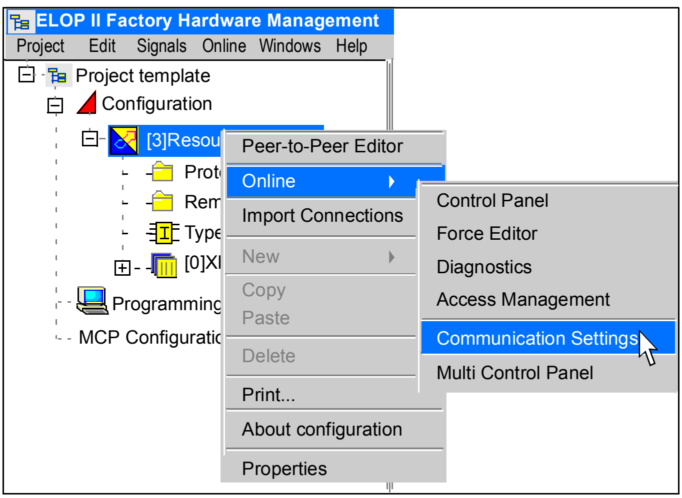
When setting the communication of the PLCs or remote I/O you must ensure that all firewalls are disabled.

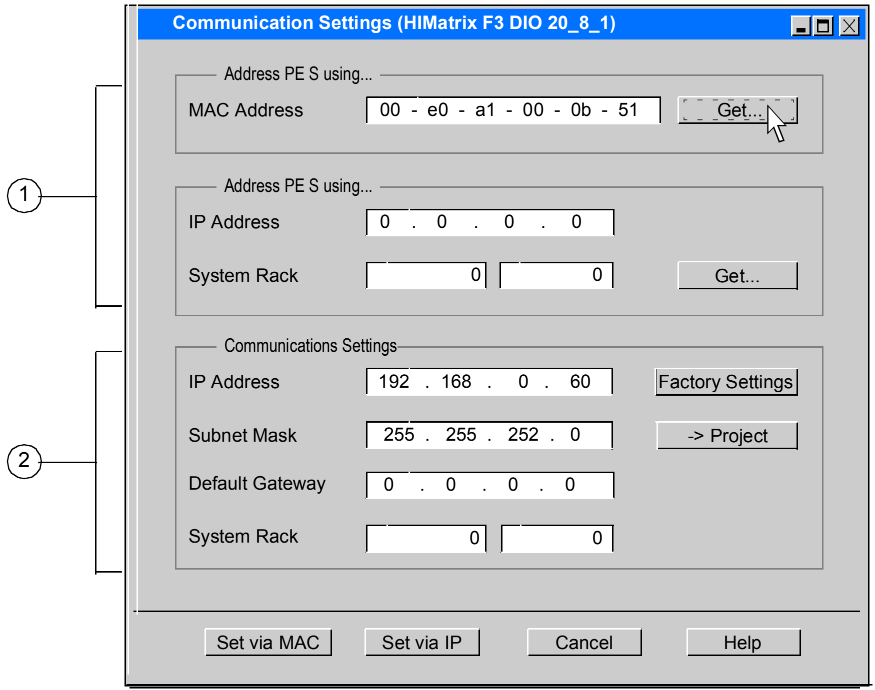
Typical firewalls are:

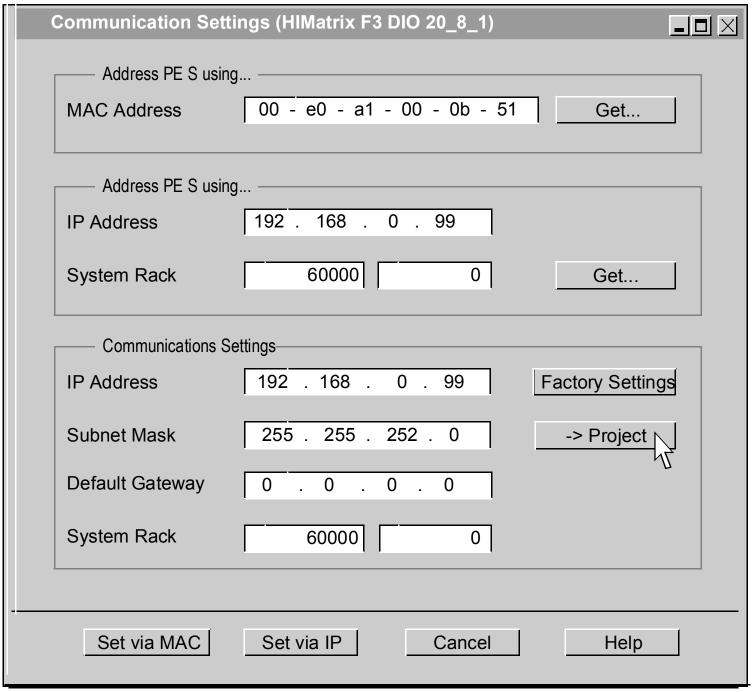
- Microsoft windows firewalls
- VPN network firewalls e.g. BlackICE

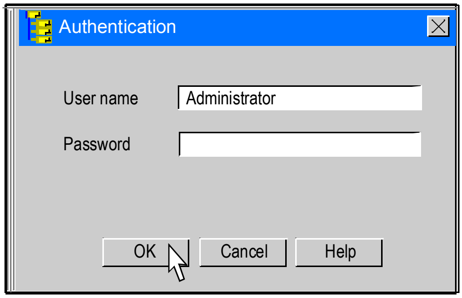
On PC start up, normally all firewalls are automatically enabled.

Set the IP Address

Step	Action
1	Connect the PLC or remote I/O via Ethernet to the programming terminal.
2	Select **Online** → **Communication Settings** in the **Resource** context menu. Opening Communication Settings 

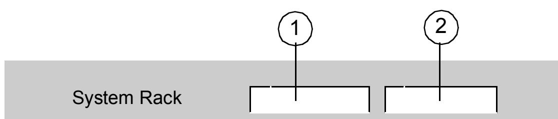
Step	Action
3	Enter the MAC address of the PLC in the **MAC Address** field in the **Communication Settings** window. You will find the MAC address on the bottom side of the PLC near the Ethernet connectors, or for the XPSMF60 (XPSMFCPU22) directly on the CPU module. Click on the **Get** button.  **1** Addressing area of PLC or Remote I/O **2** Area for Assigning IP Address of the PLC or Remote I/O

Step	Action
4	**Result**: The **Communication Settings** from PLC or remote I/O are displayed. ● IP address and subnet mask ● system rack values (SRS), defined in the properties of the PLC or remote I/O  **Note:** If you receive a timeout occurred error message **1.** check your connection to the network is okay **2.** check if any Firewalls are active
5	Enter the required IP Address/Subnet Mask/Default Gateway and System Rack. (see *System Rack Definition, p. 186*
6	Click on the -> Set via MAC to save the communication settings to the PLC

Step	Action
7	Set the new IP address to the PLC by entering the default setting **Administrator** without a password.  **Note**: See the message in the Error-State-Viewer for successful data change. After setting the parameters for all resources and remote I/Os you can download your application into the resources. **Note:** You must code generate after setting the IP configuration to download your configuration to your PLC.
8	Click on the **Project** button and set the communication properties to the Resource or Remote I/O Module.

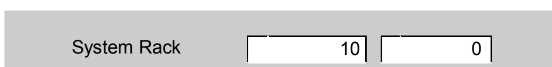
System Rack Definition

For the System Rack the definition is as follows:

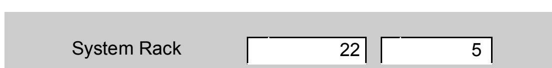


1 SRS of PLC/Resource
2 SRS of Remote I/O

Example 1: When assigning the System Rack of a Resource with SRS = 10, set the following value:



Example 2: When assigning a remote I/O with parent Resource (PLC) SRS = 22 and Remote I/O SRS = 5, set the following value:



Peer-to-Peer Communication (P2P Communication)

Overview

Peer-to-Peer (P2P) communication is used to exchange data between 2 or more PLCs in a network with several participants.

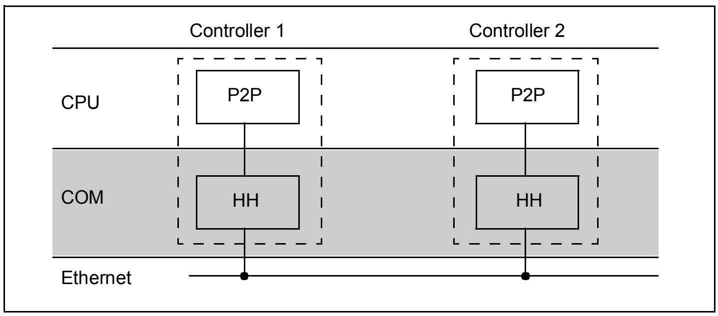
All PLCs are peer-to-peer capable and can be interconnected without any restrictions.

Communication Media

The PLCs are usually connected via Ethernet. However, by using so-called gateways, other communication media can be used, for example, telephone, radio relay or fibre optics.

Ethernet Interfaces

The PLCs are each fitted with at least two 100 BaseT Ethernet interfaces.



HH Protocol

The HH protocol is integrated into the operating system of the COM communication module and interacts with the Ethernet interfaces (see above figure).

The HH protocol is based on the UDP/IP and IEEE 802.3 standards and is responsible for a collision-free data transfer via Ethernet in various network topologies.

Peer-to-Peer Protocol

The peer-to-peer protocol is integrated into the safety-related operating system of the CPU. The CPU uses the COM communication module as a deterministic transfer channel.

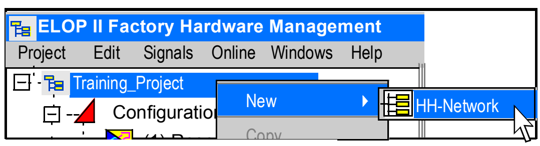
The peer-to-peer protocol is mainly responsible for:

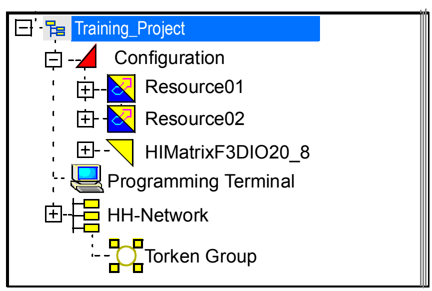
- communication between the central modules of the controllers, including the automatic establishment of a connection
- extended diagnostics
- all safety-relevant properties to ensure correct exchange of data

As the above block diagram shows, both the HH protocol and the peer-to-peer protocol are vital with respect to safe communication via Ethernet.

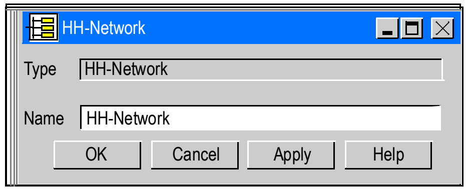
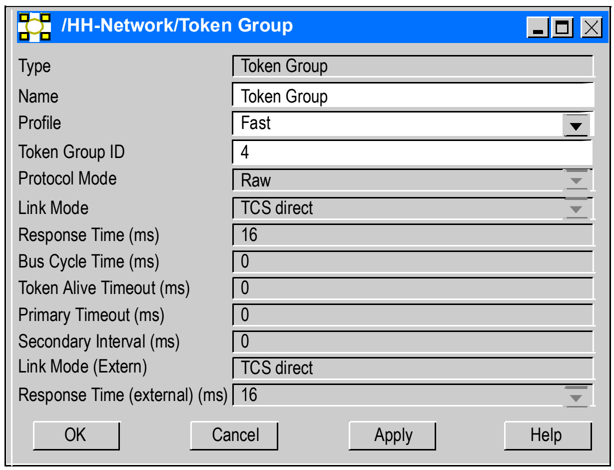
Note: The peer-to-peer protocol is safety-related according to DIN V 19250 (AK6), EN/IEC 61508 (SIL 3) and EN 954-1 (CAT 4).

Step 1: Create Network and Token Group

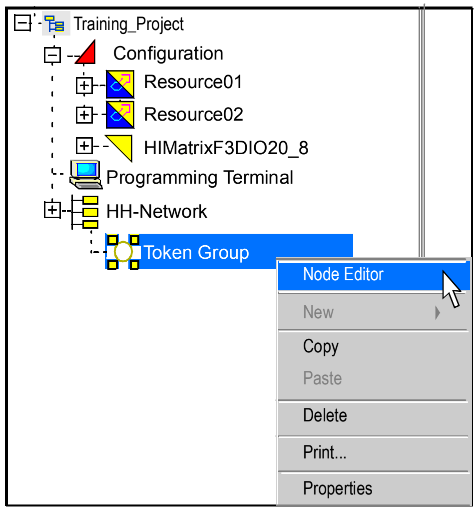
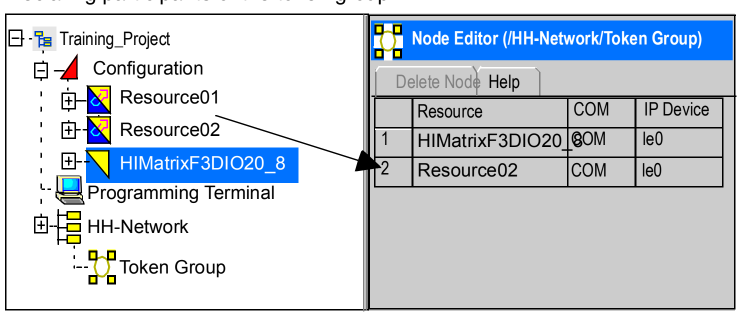
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select New → HH-Network in the Project context menu.


Result: The newly created network is added to the project tree and already contains a token group.


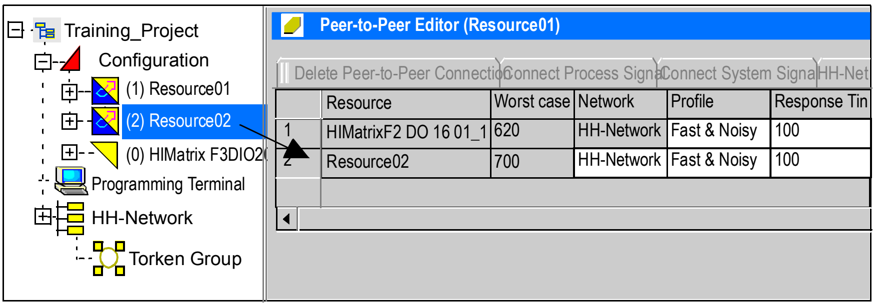
</td></tr>
</table>

Step	Action
2	If necessary, change the name in the network context menu in **Properties**. Network properties 
3	Open the token group **Properties** in the context menu.
4	Rename the token group in the **Name** field if required.
5	Select the **Fast** profile (default setting). 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. ● None setting requires detailed knowledge of the parameters and should only be used for testing. 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

Step	Action
1	In the context menu of the **Token Group** select **Node Editor** to define the members of a token group. 
2	Drag the resources that belong to the selected token group from the structure tree to the Node Editor. Declaring participants of the token group 

Step 3: Declaring Partners for a Resource

Step	Action
1	Open the **Peer-to-Peer Editor** in the context menu of a resource.
2	Drag resources from the structure tree to the Peer-to-Peer Editor with which the resource is to communicate. 
3	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.
4	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.

6.9 Peer-To-Peer Editor

At a Glance

Overview

This section describes the **Peer-to-Peer Editor**.

What's in this Section?

This section contains the following topics:

Introduction

Overview

The following provides details of the settings within the **Peer-to-Peer Editor**.

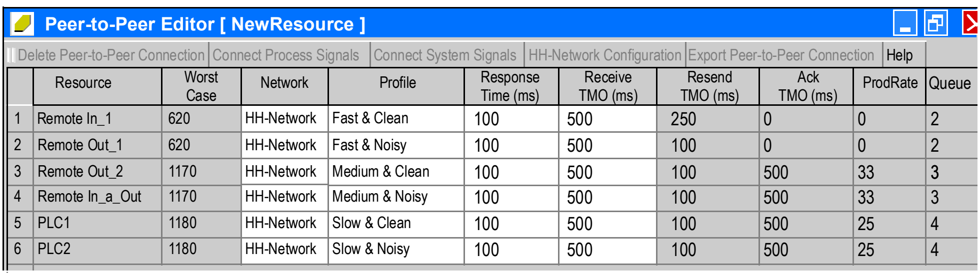
Peer-to-peer communication is used every time you establish a safe communication between either 1 or more safety PLCs or 1 or more remote I/O modules.

Therefore, please take note of the following details, as it will simplify the overall system setup.

Note: In most cases the default values are sufficient within an application. However, when very fast response times or different communication mediums are used (such as serial, or wireless, etc.) then the settings must be adjusted according to your application.

Representation

The following diagram shows a typical **Peer-to-Peer Editor** view.



Peer-to-Peer Editor [NewResource]

Delete Peer-to-Peer Connection | Connect Process Signals | Connect System Signals | HH-Network Configuration | Export Peer-to-Peer Connection | Help

	Resource	Worst Case	Network	Profile	Response Time (ms)	Receive TMO (ms)	Resend TMO (ms)	Ack TMO (ms)	ProdRate	Queue
1	Remote In_1	620	HH-Network	Fast & Clean	100	500	250	0	0	2
2	Remote Out_1	620	HH-Network	Fast & Noisy	100	500	100	0	0	2
3	Remote Out_2	1170	HH-Network	Medium & Clean	100	500	100	500	33	3
4	Remote In_a_Out	1170	HH-Network	Medium & Noisy	100	500	100	500	33	3
5	PLC1	1180	HH-Network	Slow & Clean	100	500	100	500	25	4
6	PLC2	1180	HH-Network	Slow & Noisy	100	500	100	500	25	4

The title bar of the **Peer-to-Peer Editor** shows the name of the PLC to which the other PLCs or remote I/Os shown in the main window are connected.

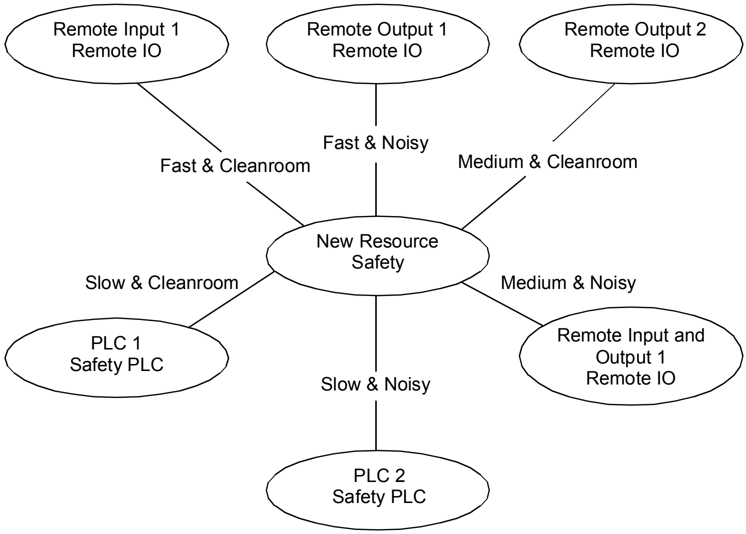
Connections

The PLC **NewResource** is connected to the following remote I/Os:
- Remote Input 1
- Remote Output 1
- Remote output 2
- Remote Input and output 1

In addition it is connected to the following safety PLCs:
- PLC1
- PLC2

Overall Communication

The image below only shows an overall communication and has no relation to the network architectures.

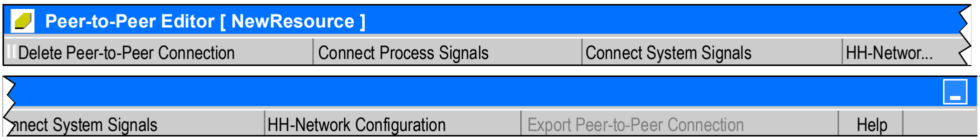


Menus

The **Peer-to-Peer Editor** has a number of options which are user defined, and others which are defined according to the network communications profile.

The following sections provide detailed information regarding each of the functions available via the menus.

Main menu



To activate items shown in black above select a number next to each of the PLCs or remote I/Os.

1	Remote Input 1
2	Remote Output 1
3	Remote Output 2
4	Remote Input and Output
5	PLC1
6	PLC2

Note: Some of the above functions are only available with PLCs.

Main Menu, Delete Peer-To-Peer Connection

Overview

This option removes the peer-to-peer connection between the selected PLC and the parent PLC communication partner (in this case **NewResource**).

This can also be done by selecting the relevant line number next to the resource name and selecting the delete button from your keyboard.

Note: The delete function only works with PLCs and not with remote I/O modules, as the remote I/O modules must always have a peer-to-peer communication with their parent resource.

Main Menu, Connect Process Signals

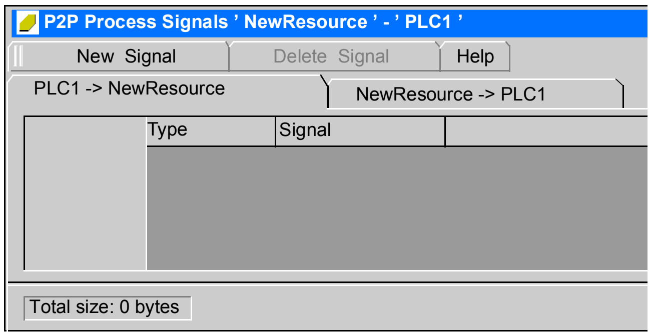
Overview

Connect Process Signals is used to read and write input data from/to PLCs.

Remote I/Os automatically send and receive data to and from the parent resource and therefore when selecting a remote I/O module this feature is not available.

Representation

The following window opens when selecting this menu:



The window shows two different tabs:
- PLC1 → NewResource
- NewResource → PLC1

Possibilities

This menu enables the following possibilities.

Areas where 2 or more PLCs are used to control areas (or zones) which need to have input data from a specific function such as the input states of an emergency stop function.

The PLCs in different areas may or may not have the same program used within each PLC.
- When the same program is used the PLC will know what the input values are for, and therefore it is possible to directly connect the outputs signals on the PLCs or its' respective remote I/O.
- When a separate program is used within each PLC, the input values must be again fed into the logic of the second program, and used in the same way as in the first program (however according to application requirements).

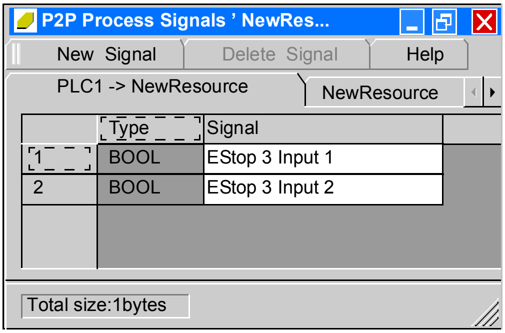
Connection of Input Signals

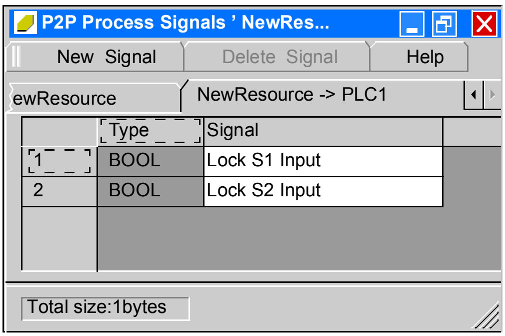
Connection of input signals to send and receive data

Step	Action
1	To transfer data between 2 PLCs open the **Signal Editor** from the **Hardware Management** menu bar and select **Connect Process Signals**.
2	Select the input signals which you wish to transfer between 2 PLCs from the **Signal Editor** and drag them into the respective tabs.

3	For this example transfer the following input signals:
	● **PLC1 → NewResource**
	  ● EStop 3 Input 1
	  ● EStop 3 Input 2
	Note: The input signals are from the **PLC1** program and are managed by inputs coming from **PLC1** or its' associated remote I/O.
	● **NewResource → PLC1**
	  ● Lock S1 input
	  ● Lock S2 input
	Note: The input signals are from the **NewResource** program and are managed by inputs coming from **NewResource** or its' associated remote I/O).

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>Drag and drop the respective signals to achieve the following result:





Note: The order of the signals is irrelevant within each window, and no addressing settings are required (e.g. M0) to transfer the data.
Note: This feature only allows input signals to be connected and will cause a failure in either the code generation phase or during downloading if output signals are used.</td></tr>
</table>

Main Menu, Connect System Signals

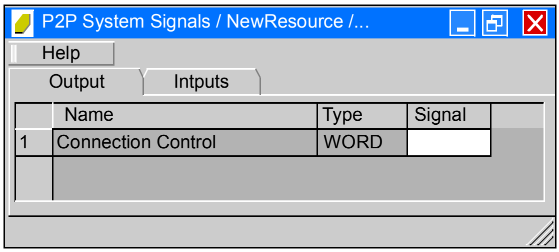
Overview

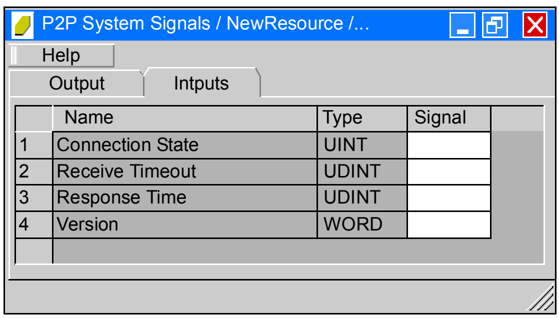
Connect System Signals is used to provide diagnostic data from the connection between 2 PLCs or a PLC and remote I/O, and in addition provides an option to control a network connection.

This feature is open for both PLCs and remote I/O.

Representation

The following window opens when selecting this menu:





The window shows two different tabs:
- Output
- Inputs

Possibilities for Output Tab

The output tab allows you to control how the peer-to-peer communication is established.

There are 4 different modes available:

Mode	Description
Mode 1: Auto Connect	This is the default setting, if no signal is attached to the connection control, or if a signal with a word value of 0000 is provided then the communication will automatically run.
Mode 2: Toggle Mode 0	After communication is lost the toggle mode allows the user to re-establish communication by changing the toggle mode value via the application program. Setting **Toggle Mode 0** requires a word value of 0100. When **Toggle Mode 0** is configured and communication is lost (`connection state=CLOSED`), a reconnect is only carried out after **Toggle Mode 1** has been enabled by the application program.
Mode 3: Toggle Mode 1	After communication is lost the toggle mode allows the user to re-establish communication by changing the toggle mode value via the application program. Setting **Toggle Mode 1** requires a word value of 0101. When **Toggle Mode 1** is configured and communication is lost (`connection state=CLOSED`), a reconnect is only carried out after **Toggle Mode 0** has been enabled by the application program
Mode 4: Disabled	Peer-to-peer communication is switched off, there is no attempt to connect. To disable the connection use a word value of 8000.

Possibilities for Input Tab

The input tab allows you to view diagnostic data from the peer-to-peer communication.

This data can be used within the application (e.g. for alarms) and also transferred to a non-safety related automation device (e.g. graphical user panel such as a Magelis terminal, or a standard PLC such as a Premium PLC).

There are 4 possible connections available:

Connection	Description
Connection State	Enables the status of the communication between 2 controllers to be evaluated in the user program. The connection state can have the following 3 UINT values: • 0 = Closed: no attempt to connect • 1 = Trying to connect • 2 = Connected
Receive Timeout (ReceiveTMO)	Is the Monitoring time on the PLC in which a correct response must be received from its' communication partner either a second PLC or remote I/O. The receive timeout also applies in the reverse direction. If no correct response is received from the communication partner within the receive timeout, the safety-related communication is closed and all the signals imported via the communication channel are reset to the initial values configured by the user. The following requirement must be met for a network in which potential data collisions could occur: **Receive Timeout** =2***ResponseTime** (minimum) If this requirement is • fulfilled, the loss of at least 1 data packet can be tolerated without the peer-to-peer connection being dropped. • not fulfilled, the availability of a peer-to-peer connection can only be guaranteed in a network that is free of collisions and faults. However, this does not affect the safety of the CPU. **Note:** The maximum permitted value for **Receive Timeout** depends on the application process and is set in the **Peer-to-Peer Editor** together with the maximum expected **ResponseTime** and the network profile (this will be explained further within this section of the manual).
Response Time	Measured response time for a data packet from PLC1 -> PLC2 -> PLC1, based on the network hardware, CPU cycle time and the peer-to-peer profile.
Version	The version is the value of the check sum (CRC) for the peer-to-peer configuration between 2 PLCs. The check sums (CRC) in both PLC must be the same to be able to establish communications. To use any of the above features create new signals with the respective data types within the **Signal Editor** and drag the signals to either the inputs or output tab according to your needs.

Main Menu, HH-Network-Configuration

Overview

The **HH-Network-Configuration** provides the status of the link between the PLC and its' selected communication partner (PLC or remote I/O).

The configuration is set up automatically when setting up a peer-to-peer communication between PLCs or when assigning a remote I/O to a resource.

Submenu Bar of the Peer-to-Peer Editor

Representation

The submenu bar of the **Peer-to-Peer Editor** provides the following submenus:

Resource	Worst Case	Network	Profile	ResponseTime (ms)	ReceiveTMO (ms)	ResendTMO (ms)	AckTMO (ms)	ProdRate	Queue

The submenu bar provides header information regarding the properties described in the following sections.

Submenu, Resource

Overview

The **Resource** is the name of the PLC or remote I/O which you have assigned within the **Project Management** for PLCs and within the properties window of a remote I/O.

Submenu, Worst Case

Overview

The **Worst Case** reaction time T_R is the time between the occurrence of a change in a physical input signal on PLC_1 and the change of the corresponding physical output signal on PLC_2 or remote I/O_2.

> **Note:** T_R is dependent on the application requirements and must be agreed with the approving boards. The worst case reaction time can be seen in the **Worst Case** column in the **Peer-to-Peer Editor**.

Formula

The worst case reaction time is calculated as follows:

$T_R \leq t_1 + t_2 + t_3 + t_4$

Parameter	Description
T_R	worst case response time
T_1	equal to 2 * **Watchdog Time** of PLC_1 **Note:** Longest possible time until the user program has read and processed input signals in PLC_1 and made the data available for transmission.
T_2	additional transmit delays in PLC_1 • when the **ProdRate** is 0, T_2 = 0 ms • when **ProdRate** is not equal to 0 then T_2 = **ReceiveTMO** + **Watchdog Time** (PLC_1)
T_3	equal to **ReceiveTMO** maximum age of a data packet when it is received by PLC_1
T_4	equal to 2 * **Watchdog Time** of PLC_2 **Note:** Longest possible time until the user program has read and processed input signals in PLC_2 and made the data available for transmission.

Reduce the Worst Case Response Time

Communication is normally established without any problems using the previous network settings in the HH protocol and peer-to-peer protocol.

However, by further optimizing these settings, a more even network loading and faster data exchange can be achieved.

> **Note:** If there is no real need to reduce the worst case reaction time, do not carry out any changes to the **Watchdog Time** and the **ReceiveTMO**. Only the **ResponseTime** should be optimized. A high **Watchdog Time** or **ReceiveTMO** does not impair performance. An optimized **ResponseTime**, on the other hand, increases availability.

Before starting the optimizing process, leave the application to run for several hours.

We recommend that as many different operating conditions as possible are tested.

This prevents faults occurring after the optimization process due to time factors that have not been taken into account.

Procedure

Reducing the worst case response time and optimizing network conditions should be managed in the following ways:

Step	Action
1	The **Watchdog Time** for each PLC must be evaluated. To manage this is only possible after the complete program is completed (including all non-safety related protocol creation, etc.) Download to each communication partner (if remote I/O then only download to the PLC), and run the system with default watchdog settings.
2	From the online control panel view for each PLC the maximum cycle time for each of the PLCs/remote I/O and add a value of 1 to 2 ms to each maximum cycle time to obtain the **Watchdog Time**. **Note:** If the **Watchdog Time** falls below the cycle time (even for 1 cycle) the PLC will move into a failure stop mode therefore do not set it below the maximum cycle time.
3	Enter the new **Watchdog Time** within the properties window of each PLC or remote I/O.
4	Code generate and download the configurations again, to update the new settings to the PLCs.
5	Ensure within the **Peer-to-Peer Editor** that the value under **ProdRate** is 0. **Note:** The **ProdRate** is dependant on the minimum response time when using network profiles medium and slow. Using a typical switched Ethernet allows the fast profile to be used which will always give a **ProdRate** value of 0. When using the medium or slow profile adjust the minimum response time in the **Peer-to-Peer Editor** to **ResponseTime/4**.
6	**ReceiveTMO** is equal to or greater than 2*__ResponseTime__. To reduce the **ReceiveTMO** run the completed application on the PLCs and (over a period of several hours) examine the maximum response time. From this value it is possible to enter a new **ReceiveTMO** value (2*__ResponseTime__).

Submenu, Network

Overview

Network represents the name of the HH-Network shared by the resources.

> **Note:** The name of the HH-Network is automatically set when a resource is added to the Peer-to-Peer Editor, but can also later be entered manually, if a HH-Network has not been specified yet.

HH-Network Setting

The HH-Network setting should be created according to the communication path between 2 PLCs or a group of PLCs.

For example, you have to connect a PLC to 1 or more remote I/Os which have a communication path which is a normal switched Ethernet network.

Then all of the nodes within this segment which have the same network conditions (even when not directly communicating via the same path) can be assigned within the same network profile (HH-Network).

When a PLC is located within an area possessing different network conditions such as using routers, or using a serial communication network then a new network profile (HH-Network) should be created.

Submenu, Profile

Overview

Within the **Peer-to-Peer Editor** it is possible to select the closest communications profile between 2 PLCs or a PLC and remote I/O.

There are 6 profiles available.

Fast and Cleanroom

Use for the fastest data throughput for
- applications requiring fast data transmission
- applications requiring a worst case reaction time as low as possible

Item	Setting
minimum requirements for the Ethernet network	• Fast: 100 Mbit technology (100 Base TX) • Switched: Full duplex LAN switches only • Clean: Fault-free network Data loss due to network overload, external influences or network manipulation should be avoided. **Note:** The network can be shared by other participants if sufficient transport capacity is available.
characteristics of the communication path	• Minimum delays • ResponseTime ≥ ReceiveTMO (otherwise ERROR))
variables	Manually set in the **Peer-to-Peer Editor**: • ResponseTime • ReceiveTMO • Watchdog time
compatible HH-network profile	• Fast

peer-to-peer parameters set	• QueueLen = 2
	• Communication time slice large enough to enable all protocol stacks to be processed in one CPU cycle
	• if ReceiveTMO ≤ 2 * WDZ then
	  • ResendTMO = ReceiveTMO / 2 or
	  • ResendTMO = ResponseTime, whichever is greater
	• if ReceiveTMO < 2 * WDZ then
	  • ResendTMO = ReceiveTMO·
	• AckTMO = 0
	• ProdRate = 0

Fast and Noisy

Use for the fastest data throughput
- for applications requiring fast data transmission
- for applications requiring a worst case reaction time as low as possible
- if loss of individual data packets can be corrected

Item	Setting
minimum requirements for the Ethernet network	• Fast: • 100 Mbit technology (100 Base TX) for HH-Network profile I (Fast) • 10 Mbit technology (10 Base T) for HH-Network profile II (Medium) • Switched: • Fast Ethernet full duplex LAN switches for HH-Network profile I (Fast) • 10 Mbit hubs for HH-Network profile II (Medium) • Noisy: • Low probability of loss of data packets.Time for ≥ 1 resend
characteristics of the communication path	• Minimum delays • ResponseTime ≤ ReceiveTMO / 2 (otherwise ERROR)
variables	Manually set in the **Peer-to-Peer Editor**: • ResponseTime • ReceiveTMO • Watchdog time
compatible HH-network profile	• Fast • Medium (≤ 10 controllers per Token Group)
peer-to-peer parameters set	• QueueLen = 2 • Communication time slice large enough to enable all protocol stacks to be processed in one CPU cycle • if ReceiveTMO ≥ 2 * WDZ then ResendTMO = ResponseTime (≥ 1 resend possible) • if ReceiveTMO < 2 * WDZ then ERROR • AckTMO = 0 • ProdRate = 0

Medium and Cleanroom

Use for moderate data throughput

- for applications that only require moderately fast data transmission
- suitable for applications in which the worst case reaction time is not a critical factor
- suitable for Virtual Private Networks (VPN), in which the exchange of data, when safety devices are present (firewalls, encryption) is slow but free of errors

Note: The medium and noisy profile should normally be used instead of medium and cleanroom.

Item	Setting

minimum requirements for the Ethernet network	• Medium or fast:
	  • 10 Mbit (10 Base T) or
	  • 100 Mbit technology (100 Base TX) or
	  • network with both 10 Mbit and 100 Mbit components LAN switches required
	• Clean:
	  • Fault-free network
	  Data loss due to network overload, external influences or network manipulation should be avoided.
	  • Time for ≥ 0 resends

characteristics of the communication path	• Moderate delays • ResponseTime ≤ ReceiveTMO (otherwise ERROR)
variables	Manually set in the **Peer-to-Peer Editor**: • ResponseTime • ReceiveTMO • Watchdog time
compatible HH-network profile	• Fast
peer-to-peer parameters set	• QueueLen = 3 • Communication time slice large enough to enable all protocol stacks to be processed in one CPU cycle. • if ReceiveTMO ≥ 2 * WDZ then ResendTMO = ResponseTime (≥ 0 resends possible) • if ReceiveTMO < 2 * WDZ then ResendTMO = ReceiveTMO • AckTMO = ReceiveTMO or AckTMO = AckTMOMax, whichever is smaller • ProdRate = ResponseTime / 4

Medium and Noisy

Use for moderate data throughput
- for applications that only require moderately fast data transmission
- suitable for applications in which the worst case reaction time is not a critical factor
- if loss of individual data packets can be corrected

Item	Setting

minimum requirements for the Ethernet network	• Medium or fast:
	  • 10 Mbit (10 Base T) or
	  • 100 Mbit technology (100 Base TX)
	  • or network with both 10 Mbit and 100 Mbit components Hubs can be used
	• Noisy
	  • Low probability of loss of data packets Time for ≥ 1 resend

characteristics of the communication path	• Moderate delays • ResponseTime ≤ ReceiveTMO / 2 (otherwise ERROR)
variables	Manually set in the **Peer-to-Peer Editor**: • ResponseTime • ReceiveTMO Manually set in the resource properties: • Watchdog time
compatible HH-network profile	• Medium • Fast
peer-to-peer parameters set	• QueueLen = 3 • Communication time slice large enough to enable all protocol stacks to be processed in one CPU cycle. • if ReceiveTMO ≥ 2 * WDZ then ResendTMO = ResponseTime (≥ 1 resend possible) • if ReceiveTMO < 2 * WDZ then ERROR • AckTMO = ReceiveTMO or AckTMO = AckTMOMax, whichever is smaller • ProdRate = ResponseTime / 4

Slow and Cleanroom

Use for slow data throughput
- for applications that only require slow data transmission to PLCs (which could be a considerable distance away), and where the conditions of the communication path cannot be predicted by the user.

Note: The slow and noisy profile should normally be used instead of slow and cleanroom.

Item	Setting
network requirements	• Slow: Mainly for data transfer via ISDN, dedicated line or radio relay. • Clean: Fault-free network. Data loss due to network overload, external influences or network manipulation should be avoided. Time for = 0 resends.
characteristics of the communication path	• Moderate to long delays • ResponseTime = ReceiveTMO (otherwise ERROR)
variables	Manually set in the **Peer-to-Peer Editor**: • ResponseTime • ReceiveTMO • N: Number of communication partners Manually set in the resource properties: • Watchdog time
compatible HH-network profile	• Medium • Fast
peer-to-peer parameters set	• QueueLen = 4 • Communication time slice large enough to enable all protocol stacks to be processed in one CPU cycle. • if ReceiveTMO = 2 * WDZ then ResendTMO = ResponseTime (= 0 resends possible) • if ReceiveTMO < 2 * WDZ then ResendTMO = ReceiveTMO • AckTMO = ReceiveTMO or AckTMO = AckTMOMax, whichever is smaller • ProdRate = ResponseTime / 4

Slow and Noisy

Use for slow data throughput

- for applications that only require slow data transmission to controllers (which could be a considerable distance away)
- mainly for data transfer via poor telephone lines or noisy radio relays

Item	Setting
network requirements	• Slow: data transfer via telephone, satellite, radio, etc. • Noisy: Low probability of loss of data packets Time for ≥ 1 resend
characteristics of the communication path	• Moderate to long delays • ResponseTime ≤ ReceiveTMO / 2 (otherwise ERROR)
variables	Manually set in the **Peer-to-Peer Editor**: • ResponseTime • ReceiveTMO
compatible HH-network profile	• Medium • Fast
peer-to-peer parameters set	• QueueLen = 4 • Communication time slice large enough to enable all protocol stacks to be processed in one CPU cycle. • if ReceiveTMO ≥ 2 * WDZ then ResendTMO = ResponseTime (≥ 1 resend possible) • if ReceiveTMO < 2 * WDZ then ERROR· • AckTMO = ReceiveTMO or AckTMO = AckTMOMax, whichever is smaller • ProdRate = ResponseTime / 4

HH-Network Profiles

In addition to these 6 network profiles there are also 3 additional HH-network profiles:

- fast
- medium
- none

These profiles are directly linked to the 6 network profiles and should be used according to requirements provided in the 6 network profiles.

For the HH-network profiles please refer to *HH-Network, p. 219*.

Submenu, Response Time (ms)

Overview

ResponseTime is the time which elapses before the sender of the message receives acknowledgement from the recipient.

The response time is not a parameter that can be freely configured; it is a result of the physical conditions of the transmission path and of the configuration of the network protocol.

Test Run

As the response time influences the speed of the data transfer, a test run is recommended to examine the actual response time.

P2P State Tab

In the **P2P State** tab on the **Control Panel**, the last, average, minimum and maximum response times are displayed.

The maximum value from the control panel should be used within the **Peer-to-Peer Editor**.

Submenu, ReceiveTMO (ms)

Overview

The **ReceiveTMO** (receive timeout) is the monitoring time on the PLC in which a correct response must be received from its' communication partner either a second PLC or remote I/O.

The receive timeout also applies in the reverse direction.

If no correct response is received from the communication partner within the **ReceiveTMO**, the safety-related communication is closed and all the signals imported via the communication channel are reset to the initial values configured by the user.

Requirement

The following requirement must be met for a network in which potential data collisions could occur:
- ReceiveTMO = 2 * **ResponseTime** (minimum)

If this requirement is...	Then...
fulfilled	the loss of at least 1 data packet can be tolerated without the peer-to-peer connection being dropped.
not fulfilled	the availability of a peer-to-peer connection can only be guaranteed in a network that is free of collisions and faults. However, this does not affect the safety of the CPU.

Note: The maximum permitted value for **ReceiveTMO** depends on the application process and is set in the **Peer-to-Peer Editor** together with the maximum expected **ResponseTime** and the network profile (this will be explained further within this section of the manual).

Submenu, ResendTMO (ms)

Overview

Monitoring time interval on PLC_1 within which PLC_2 must acknowledge the receipt of a data packet, otherwise the data packet is resent.

You can configure **ResendTMO**.

Submenu, AckTMO (ms)

Overview

AckTMO is the maximum amount of time within which the CPU must acknowledge the receipt of a data packet.

AckTMO cannot be entered manually. It is set in the **Peer-to-Peer Editor** together with the profile selection.

Fast Network

For a fast network, **AckTMO** is 0, i.e. the receipt of a data packet is acknowledged immediately.

Slow Network

On a slow network, (e.g. telephone modem line) **AckTMO** is greater than 0.

If this is the case, the system will attempt to transfer the acknowledgement together with process data in order to reduce the network load by avoiding addressing/ security blocks.

Submenu, ProdRate

Overview

The **ProdRate** is the smallest time interval between 2 data packets.

The aim of **ProdRate** is to limit the number of data packets to a figure that will not overload a (slow) communications channel.

This ensures even loading of the transfer medium and avoids the receipt of obsolete data on the receiver side.

Production Rate

A production rate of 0 indicates that data packets are transferred with each cycle of the user program.

Note: A value of 0 should always be used.

Submenu, QueueLen

Overview

QueueLen describes the number of data packets that can be transmitted without having to wait for their acknowledgement.

The value is dependent on the transfer capacity of the network and any possible network delays.

QueueLen cannot be entered manually; it is set in the **Peer-to-Peer Editor** together with the profile selection.

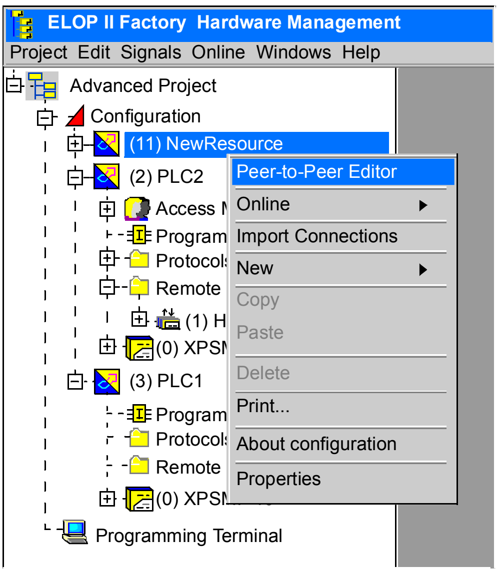
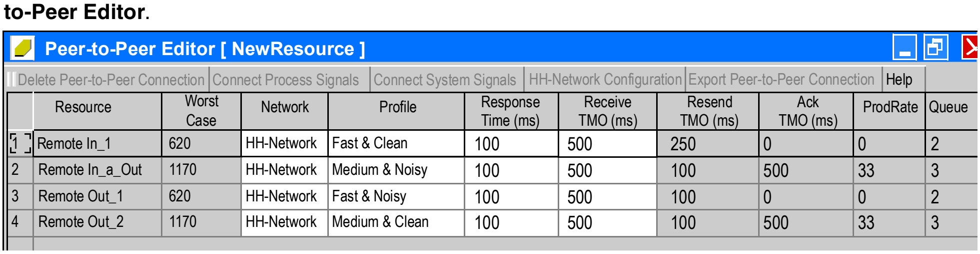
Connection of Communication Partners Within the Peer-to-Peer Editor

Overview

The connection of communication partners within the **Peer-to-Peer Editor** is created by following the steps outlined below.

Note: To follow this configuration you must have at least 2 PLCs within your configuration or a PLC and remote I/O.

Procedure

Step	Action
1	From within the **Hardware Management** window right-click on the resource and select **Peer-to-Peer Editor**.  Any remote I/Os present within the selected PLC configuration are automatically found within the **Peer-to-Peer Editor**. 

	Resource	Worst Case	Network	Profile	Response Time (ms)	Receive TMO (ms)	Resend TMO (ms)	Ack TMO (ms)	ProdRate	Queue
1	Remote In_1	620	HH-Network	Fast & Clean	100	500	250	0	0	2
2	Remote In_a_Out	1170	HH-Network	Medium & Noisy	100	500	100	500	33	3
3	Remote Out_1	620	HH-Network	Fast & Noisy	100	500	100	0	0	2
4	Remote Out_2	1170	HH-Network	Medium & Clean	100	500	100	500	33	3

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>To create a peer-to-peer communication between 2 resources is carried out by left-clicking on the second resource (PLC) and hold and drag to the Peer-to-Peer Editor of the first PLC.
The PLC2 is now connected to the first PLC (NewResource) within the Peer-to-Peer Editor.
Peer-to-Peer Editor [NewResource]
Delete Peer-to-Peer Connection | Connect Process Signals | Connect System Signals | HH-Network Configuration | Export Peer-to-Peer Connection | Help
<table>
<tr><th></th><th>Resource</th><th>Worst Case</th><th>Network</th><th>Profile</th><th>Response Time (ms)</th><th>Receive TMO (ms)</th><th>Resend TMO (ms)</th><th>Ack TMO (ms)</th><th>ProdRate</th><th>Queue</th></tr>
<tr><td>1</td><td>PLC2</td><td>640</td><td>HH-Network</td><td>Fast & Noisy</td><td>100</td><td>500</td><td>100</td><td>0</td><td>0</td><td>2</td></tr>
<tr><td>2</td><td>Remote In_1</td><td>620</td><td>HH-Network</td><td>Fast & Clean</td><td>100</td><td>500</td><td>250</td><td>0</td><td>0</td><td>2</td></tr>
<tr><td>3</td><td>Remote In_a_Out</td><td>1170</td><td>HH-Network</td><td>Medium & Noisy</td><td>100</td><td>500</td><td>100</td><td>500</td><td>33</td><td>3</td></tr>
<tr><td>4</td><td>Remote Out_1</td><td>620</td><td>HH-Network</td><td>Fast & Noisy</td><td>100</td><td>500</td><td>100</td><td>0</td><td>0</td><td>2</td></tr>
<tr><td>5</td><td>Remote Out_2</td><td>1170</td><td>HH-Network</td><td>Medium & Clean</td><td>100</td><td>500</td><td>100</td><td>500</td><td>33</td><td>3</td></tr>
</table>
Note: If you open the Peer-to-Peer Editor of the second PLC (PLC2) you will see PLC1 is within the Peer-to-Peer Editor and therefore you can see you must only define the connection once.</td></tr>
<tr><td>3</td><td>Repeat this procedure for all peer-to-peer communications you wish to set up between your PLCs within your program.
Note: It is not possible to create a peer-to-peer communication of a remote I/O associated from a different resource (PLC). If you wish to read data from a remote I/O associated with a second resource (PLC), you must manage this by selecting the resource (PLC) the remote I/O belongs to.</td></tr>
</table>

HH-Network

Overview

The HH-network connection must be created according to the physical conditions of the network (e.g. Ethernet or serial network, noisy etc..).

The HH-network contain only the profile settings and ensure that within a specific group of PLCs and/or remote I/Os they communicate together in the most optimized way.

HH-Network Set Up

The HH-network is automatically set up when creating the first remote I/O within the PLC configuration.

However according to your application additional HH-network groups can be set up by right-clicking on the overall project and selecting **New** → **HH-Network**.

HH-Network Profiles

There are 3 HH-network profiles that can be adjusted according to the 6 peer-to-peer communications profiles (see *Submenu, Profile, p. 207*).

Profile 1

Profile 1 (fast) is the standard profile for the setting of the HH-network and is recommended for all network topologies that exclusively use LAN switches.

Newer, powerful networks are usually already fitted with full duplex LAN switches.

If a new network has to be configured, full duplex LAN switches should be used instead of hubs.

So that **Profile 1** (fast) can be used, the Ethernet network must meet the following minimum requirements:

Item	Setting
minimum requirements for the Ethernet network	• 10 Mbit or 100 Mbit technology • Use of full duplex (recommended) LAN switches. • Data loss due to network overload, external influences or network manipulation should be avoided as this can lead to the loss of data. **Note:** The network can be shared by other participants if sufficient transfer capacity is available.

Profile 1 (fast) is suitable for

- applications that require a high data throughput or fast data exchange, as well as for applications with slow communication paths (e.g. telephone modem).
  This is achieved by setting an appropriate **ResponseTime** and **ReceiveTMO** in the **Peer-to-Peer Editor**.
  However, it should be noted that **ResponseTime** and **ReceiveTMO** have an effect on the **Worst Case Reaction Time**.
- communication between 2 or more **Token Groups**. In this case, **Profile 1** (fast) should be set for all token groups.

> **Note:** As token passing is switched off in the fast profile, a second controller is not required for token exchange. In this case, a token group can be generated with just 1 controller. The single controller can, however, communicate with other token groups with several controllers.

If an existing, older network (which is partly or exclusively based on hubs) is used for communication between the controllers, **Profile 2** (medium) should be used. Otherwise, collisions of data packets in the network will cause data losses that must be corrected by sending the data packets several times. This can slow down communications considerably.

Profile 2

Profile 2 (medium) is recommended for networks that are partly or exclusively based on hubs. In contrast to switches, hubs are not able to coordinate network access, a task which therefore has to be carried out by token passing.

In the case of **Profile 2** (medium), communication within a token group as well as to external token groups is controlled by token passing, which means all participating token groups must use the same profile.

The fact that only the controller holding the token can send data packets means the exchange of data is free of collisions and therefore deterministic. The data is, however, not exchanged as quickly as with **Profile 1** (fast).

To use **Profile 2** (medium), the Ethernet network must meet the following minimum requirements:

Item	Setting
minimum requirements for the Ethernet network	• 10 Mbit or 100 Mbit technology • Use of hubs or switches. • Data loss due to network overload, external influences or network manipulation should be avoided as this can lead to the loss of data. **Note:** With the exception of the controllers, no other participants are allowed to operate in a hub-based network.

Note: Only 1 programming terminal is allowed online at any one time. Programming terminals increase the amount of data in the network, but do not take part in token passing.

Profile 3

The **Profile None** allows you to adjust all values for the profile.

Note: Only experienced people should use this profile.

6.10 Control Panel, General

Control Panel, General

Overview

The **Control Panel** is the most important tool with regards to communication with a controller.

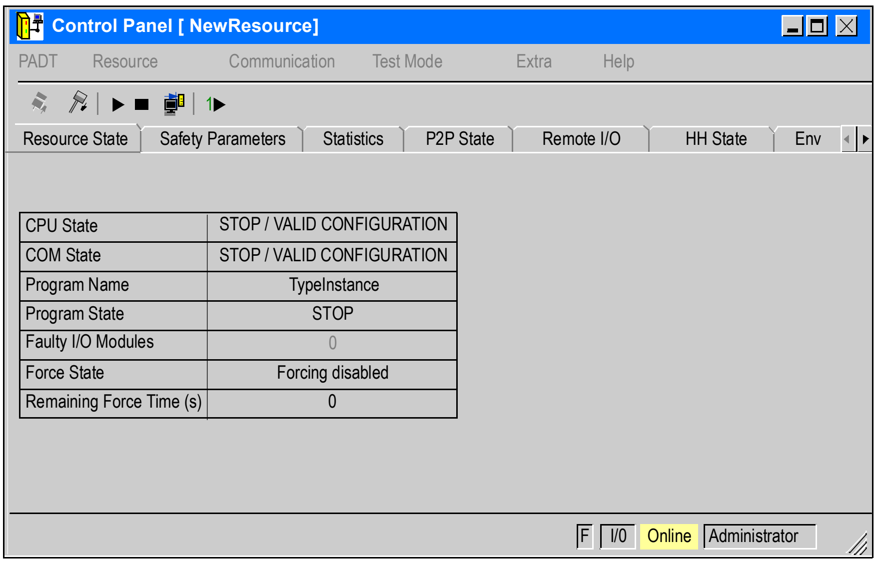
Within the **Control Panel** the user can:
- starting/stopping PLCs
- downloading
- communication statistics
- operating parameters
- operating statistics
- etc.

Note: Before the code generated by the code generator can be loaded into the PLC, all IP addresses must be correctly configured in the **Hardware Management**.

Control Panel Tabs

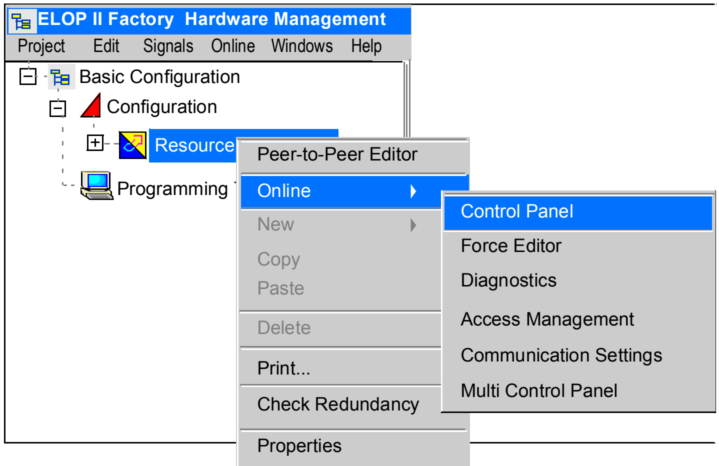
The **Control Panel** has its own menu and button bar and individual windows can be viewed by selecting the tabs:

- Resource State
- Safety Parameters
- Statistics
- P2P State (Peer to Peer
- Remote I/O
- HH State
- Environmental Data
- OS (Operating system)
- IP Settings
- Licence



Start the Control Panel

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.



Multi Control Panel

Note: For the **Multi Control Panel** please see *Run, p. 430*.

6.11 Control Panel, Menu Bar

At a Glance

Overview

This section describes the objects in the menu bar of the **Control Panel**.

What's in this Section?

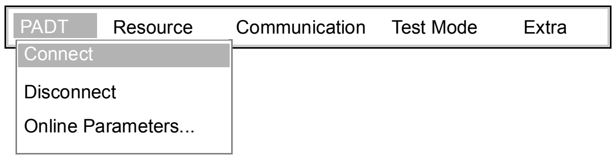
This section contains the following topics:

Menu Bar, PADT

Overview

The submenu **PADT** enables you to connect or disconnect from the PLC and view the **Online Parameters**.

Representation



Connect

Connect enables the PC and the PLC to communicate together.

Disconnect

Disconnect closes the connection between the PC and PLC.

Online Parameters

Online Parameters allows the user to adapt the response time for the communication between the PC and PLC.

Online Parameters includes the following options:

- Refresh Rate
- Service Timeout
- Request Delay

Online Parameters, Refresh Rate

The refresh rate is the interval in milliseconds in which the data displayed by the PC is read by the controller and updated on the screen. The value in brackets is the previously set (old) value.

Note: The refresh rate must be set so that neither the communication path not the PC is overloaded

If the programming terminal is connected to a controller via a slow modem line, a low refresh rate (= longer time) must be set, than would be the case with a fast Ethernet connection. The same applies if several programming terminals are communicating with the same controller simultaneously.

If the refresh rate is set too high in a fast network (= shorter time), increased system loading of the programming terminal and possibly delayed responses to keyboard inputs can occur

Online Parameters, Service Timeout

Service Timeout is the time that the PC waits for the response of a service. The value in brackets is the previously set (old) value.

The PC communicates with a resource via so-called services. If a service has not been executed by the resource, the PC waits for the service timeout before it signals the error to the user.

With slow communication channels, the service timeout must be selected in such a way that the resource has sufficiently time to execute the service. Otherwise, the requested service is acknowledged with an error message by the PC.

Note: The value must be = 500 ms.

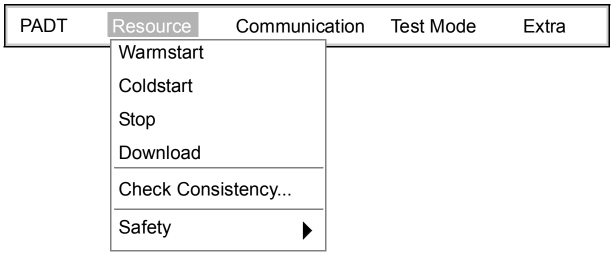
Online Parameters, Request Delay

If the PC requests several services from the resource, an interval can be set between the individual services in order to reduce the network load.

Normally it is not necessary to manually change the default value.

Menu Bar, Resource

Representation



Warmstart

Warmstart enables the user program in the resource to be started by the PC from the retained values provided.

A warmstart can also be carried out automatically if the **AutoStart Enable** switch in the resource properties is activated and **Warmstart** has been selected as the **AutoStart** mode from within the properties of the program instance.

Coldstart

Coldstart enables the user program in the resource to be started by the PC from the initial values provided.

A coldstart can also be carried out automatically if the **AutoStart Enable** switch in the resource properties is activated and **Coldstart** has been selected as the **AutoStart** mode from within the properties of the program instance.

Stop

Stops the execution of the user program on the PLC.

WARNING

UNINTENDED EQUIPMENT OPERATION

Evaluate operational state of all physical outputs before starting or stopping the program.
Stopping the user program resets all physical outputs on the resource to 0.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Download

Download loads a configuration and (if available) user program into the resource. To do this, the resource must be stopped and the code generation must have been successfully ended.

CAUTION

IRREPARABLE PROGRAM DAMAGE

Save the configuration/user program before downloading a new configuration/user program.
If a configuration/user program is already present in the PLC it will be overwritten by a download.

Failure to follow these instructions can result in injury or equipment damage.

Check Consistency

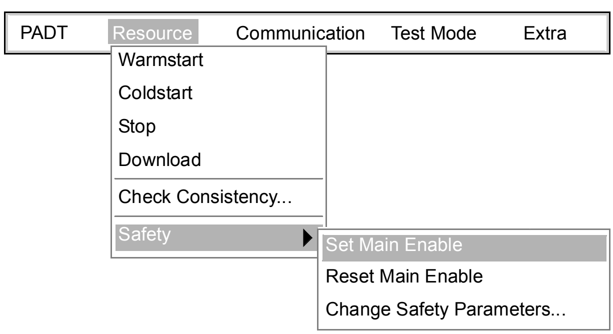
Check Consistency checks if the configuration currently loaded in the programming terminal is identical to the configuration loaded in the resource. The check is carried out using the checksums (CRCs).

Note: If the consistency check shows different checksums, certain functions (e.g. **Forcing** from the PC) cannot be carried out.

Safety

Safety includes the following options:

- Set Main Enable
- Reset Main Enable
- Change Safety Parameters



Safety, Set Main Enable

Note: To be able to set **Main Enable**, the resource must be in STOP mode.

The (safety switch) **Main Enable** can be set or reset in the CPU. Only when **Main Enable** is set, changes to the safety parameters can be carried out.

The **Main Enable** switch is set if the function is selected and the confirmation message that follows is confirmed with Yes. The setting is saved in the properties of the resource.

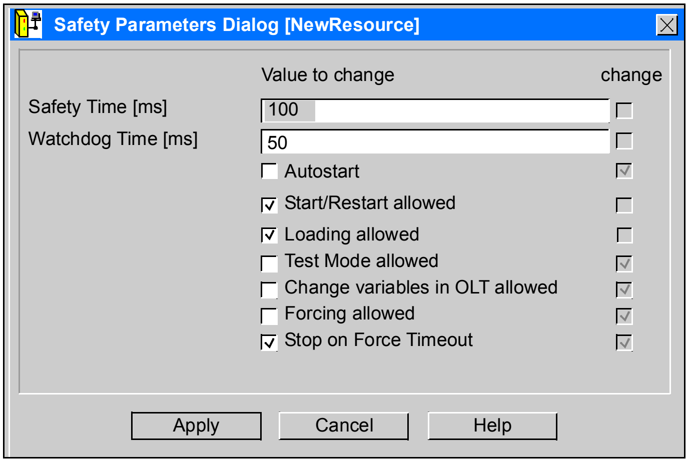
Note: Changes to the safety switch are immediately carried out in the resource and automatically saved in the project.

Safety, Reset Main Enable

Note: The resource does not have to be stopped to reset the **Main Enable**.

The **Main Enable** is reset if the function is selected and the confirmation message that follows is confirmed with Yes.

Safety, Change Safety Parameters



The configuration of a resource is a part of the project and is normally defined in the properties of the resource during the creation of the project.

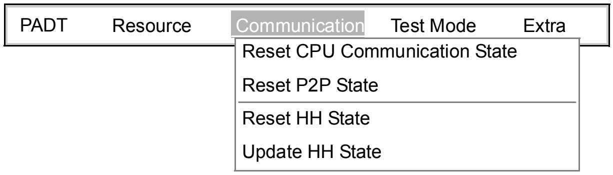
However, the safety parameters can be changed online if **Main Enable** has been set. Parameters that will change after the **Apply** button is pressed are marked with a tick in the **Change** column.

As soon as the new settings become active in the CPU, they are also saved in the properties of the resource in the project.

> **Note:** Detailed information regarding the settings of the safety parameters can be found in the *Safety Manual* and in the topic *Properties, p. 157* of this manual.

Menu Bar, Communication

Representation



Reset CPU Communication State

The **Reset CPU Communication State** function resets the values of the **Communication Time Slice** in the **Statistics** tab. The reset function is particularly useful if the maximum and minimum values have to be re-calculated.

Reset P2P State

Reset P2P State resets the values of the response time in the **P2P State** tab. The reset function is particularly useful if the maximum and minimum values have to be re-calculated.

Reset HH State

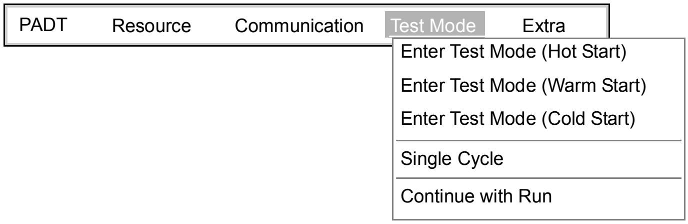
The **Reset HH State** function resets the values of the Token Cycle Time and the response time in the **HH State** tab. The reset function is particularly useful if the maximum and minimum values have to be re-calculated.

Update HH State

To keep the network load at a minimum, the COM communication module does not update the **HH State** automatically. A user with administrator access can manually update the screen display using the **Update HH State** command.

Menu Bar, Test Mode

Representation



⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use Test Mode during normal operation.
Limit use of Test Mode to system commissioning and de-bugging.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Enter Test mode (Hot Start)

Enter Test Mode (Hot Start) halts the execution of the user program, with all signal states and physical outputs retaining their values and settings. Using the **Single Cycle** command, the user program can then resume in single step mode.

Note: The test mode can only be switched on if **Test Mode allowed** has been previously activated in the properties of the resource.

Enter Test mode (Warm Start)

Enter Test Mode (Warm Start) halts the execution of the user program, with the signals declared as **Retain** retaining their values, and all other signals of the user program reset.

Note: The test mode can only be switched on if **Test Mode allowed** has been previously activated in the properties of the resource.

Enter Test mode (Cold Start)

Enter Test Mode (Cold Start) halts the execution of the user program, with «all» signals of the user program being reset. Using the **Single Cycle** command, the user program can then be started in single step mode.

> **Note:** The test mode can only be switched on if **Test Mode allowed** has been previously activated in the properties of the resource.

Single Cycle

When the user program is in test mode, **Single Cycle** instructs the CPU to execute the user program a single time. The command must be re-executed for each cycle.

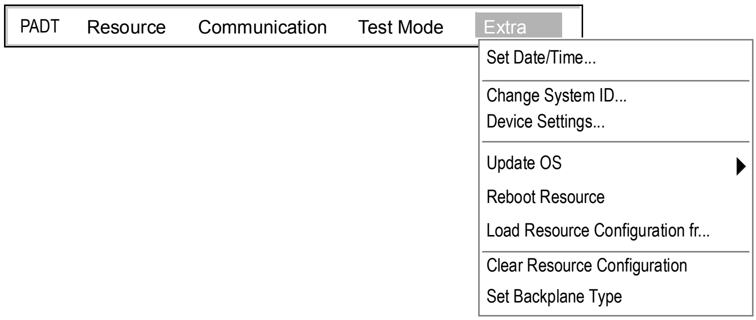
> **Note:** The Test Mode can only be switched on if **Test Mode allowed** has been previously activated in the properties of the resource.

Continue with Run

Continue with Run ends the test mode and the user program continues to run in normal operation with the current signal states.

Menu Bar, Extras

Representation



Set Date/Time

The clock integrated in the PLC can be set using **Set Date/Time**. Date and time are particularly important when evaluating the long term and short term diagnostics. When setting the date and time, the setting in the programming terminal is used as standard. If this is not suitable, a user-defined setting can also be carried out.

Note: The date and time can only be changed if the CPU **Main Enable** switch has been set before.

Change System ID

Change System ID is used to change the SRS (System Rack ID) online. Before the System ID can be changed, any user program being executed in the controller must be stopped. After entering the new System ID, it is saved in the controller using the **OK** button and acknowledged by means of a message.

Note: The configuration must be code generated again to ensure the ID has been updated within the configuration.

Device Settings

The **Device Settings** function allows the administrator to reset the Ethernet network parameters. To do this, the controller must be in STOP state.

Note: The configuration must be code generated again to ensure the IP address has been updated within the configuration.

Update OS

Is not used.

Reboot Resource

If a PLC is in ERROR STOP state (red LED lights up), it can only be reset by means of the **Reboot Resource** function.

Load Resource configuration from Flash

Reads the configuration of the PLC from the COM Flash File System of the PLC and transfers the data into the CPU and COM. This is comparable to a restore.

Used if the PLC goes into ERROR STOP due to a data error in the non-volatile memory of the CPU (NVRAM), the PLC must be rebooted using the command **Reboot Resource**. The controller then goes into STOP/INVALID CONFIGURATION state due to the data error in the NVRAM. Use this feature to transfer the backup data from the Flash to the CPU.

Clear Resource Configuration

Clear Resource Configuration deletes the configuration. However, the buffered memory for long and short term diagnostic, the date and time settings, the System ID and the IP Address all remain unchanged.

Set Backplane Type

Do not use this command.

6.12 Sub Window Tabs

At a Glance

Overview

This section describes the sub window tabs of the **Control Panel**.

What's in this Section?

This section contains the following topics:

Sub Window Tabs, Resource State

Representation

CPU State	STOP / VALID CONFIGURATION
COM State	STOP / VALID CONFIGURATION
Program Name	TypeInstance
Program State	STOP
Faulty I/O Modules	0
Force State	Forcing disabled
Remaining Force Time (s)	0

CPU State

The **CPU State** is information provided on the current **CPU State** of the PLC. The possible states are:

- Run
- Stop/Valid Configuration
- Stop/Invalid Configuration
- Stop/Loading OS
- Error Stop

COM State

The **COM State** is information provided on the current Communication State of the PLC. The possible states are:

- Run
- Stop/Valid Configuration
- Stop/Invalid Configuration
- Stop/Loading OS

Program Name

Name of the program instance currently loaded in the resource (PLC).

Program State

The **program state** is the current state of the program instance. The possible states are:

- Run
- Stop
- Halt
- Missing
- Error Stop

Faulty IO Modules

The **Fault IO modules** status provides the number of faulty or incorrectly configured input/output modules. A module is also deemed faulty if it has been incorrectly configured.

Force State

Force State provides the status of Forcing. Possible States are:

- Forcing inactive
- Forcing activated
- Forcing prepared

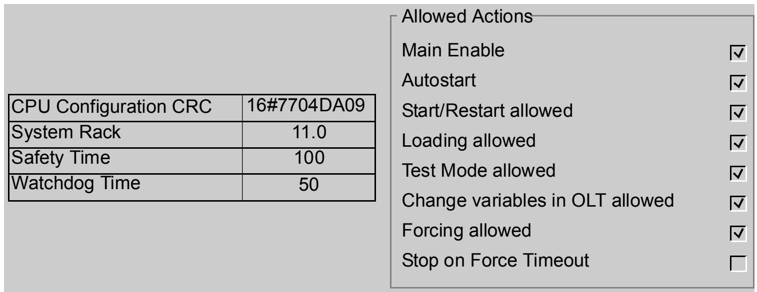
Remaining Force Time

Remaining Force time is the time in seconds until the **Force state** is reset. Depending on the state of the **Stop on Force Timeout** switch, the user program is either halted or continues to run with the latest process values.

The value is 0 if forcing is not active or is disabled.

Sub Window Tabs, Safety Parameters

Representation



CPU Configuration CRC

The **CPU Configuration CRC** is a hexadecimal representation of the checksum which results from the configuration of the CPU after the compilation of the project. See section Code Generator for more information.

System Rack

The **System Rack** provides the system ID of the PLC (SRS).

Safety Time

The **Safety time** is a time specified by the user in the resource properties which the user may need (maximum) in order to react to the change of a physical input signal with an output signal, and in addition is the maximum time which the PLC has to react to an internal failure/error.

Watchdog Time

The **Watchdog time** is the maximum time in milliseconds allowed for processing a routine cycle.

Allowed Actions

The current states of the CPU switches are displayed in the right half of the window, as well as how they were configured and saved in the project.

Sub WIndow Tabs, Statistics

Representation

Date/Time	04/09/2000 06:58:58

	last	avg.	min.	max.
Cycle Time	4	3	1	5
Com.Time Slice [ms]	0	0	0	0
Number of Time Slices	0	0	0	0

Date/Time

Date/Time display of date and time of the PLC. The settings can be changed using the **Extra** → **Set Date/Time** menu.

Cycle Time (ms)

The **Cycle Time** displays the last, average, minimum and maximum cycle time.

The cycle time is the time required by the CPU to execute the user program once completely.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Make sure that the cycle time is smaller than the watchdog time.
If the cycle time is not smaller than the watchdog time, the PLC or Remote I/O will move into an error stop condition.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Com. Time Slice (ms)

The **Communication Time Slice** is the time which was required (or which is required in average), to process all communication tasks within a CPU cycle. The communication time slice is part of the cycle time.

Number of Time Slices

Number of time slices which are/were required to process all communication tasks.

Note: This value should always be at 1! This means that the communication time slice calculated was sufficiently large to avoid requiring several CPU cycles to complete all communication tasks.

Sub Window Tabs, P2P State (Peer to Peer)

Representation

Resource	System.Rack	State	RspT last [ms]	RspT avg. [ms]	RspT min. [ms]	RspT max.[ms]	MsgNo	AckMsgNo
test	10.0	Closed	0	0	4294967295	0	0	0

Data Seq	Opens	Resends	BadMsgs	EarlyMsgs	ReceiveTMO [ms]	ResendTMO [ms]	AckTMO [ms]	CurKeVer
0	0	0	0	0	500	100	0	16#000020

Resource Name of other Safety PLC(s) with which the PLC currently being viewed (see header of the Control Panel) is communicating.

Syst. ID The System ID is the unambiguous identification of a resource in a network. The System ID is part of the SRS (System, Rack, and Slot) and can be between 2 and 65535.

State State provides the current status of communication. There are 3 different states:

- Connected
- Trying to open
- Closed, no attempt to connect

RspT (last, avg., min, max) Measured response time for a data packet from PES1 -> PES2 -> PES1, based on the network hardware, CPU cycle time and the Peer-to-Peer profile.

MsgNo The message number is provided by a 32 bit resolution counter for all data packets which have been sent to the PLC.

AckMsgNo The acknowledgement number is the number of the received data packets, the receipt of which the PLC must acknowledge.

DataSeq The data sequencing is provided by a 16 bit resolution counter for data packets which contain process data. The difference between the message number and data sequence is that the data sequence data packets only contain protocol information for the processing of the Peer-to-Peer communication.

Opens Number of successfully completed connects to a PLC. A number greater than 1 indicates that the communication to a controller failed temporarily and was automatically re-established.

Resends

The number of resends is provided by a 32 bit resolution counter for data packets which were resent because acknowledgement was not received within the **ResendTMO** (repeat time exceeded).

BadMsgs

The number of bad messages is provided by a 32 bit resolution counter for received data packets which were either damaged or which were unexpected. A damaged data packet is, for example, one with an incorrect sender or a faulty checksum. An unexpected data packet is, for example, an **Open** command, although the controllers were already connected

EarlyMsgs

The number of early messages is provided by a 32 bit resolution counter for data packets which were not received in the correct order. If a data packet is lost on the network due to faults and is not received, a gap appears and the next data packet is received too early.

ReceiveTMO

Value specified by the user for the exceeded receive time (Receive Timeout).

ResendTMO

Value specified by the user for the exceeded resend time (Resend Timeout).

AckTMO

Value specified by the user for the exceeded acknowledgement time (Acknowledge Timeout).

CurKeVer

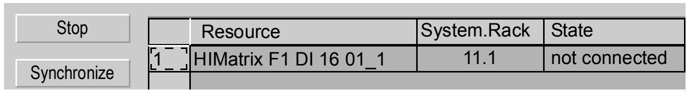
Checksum for the Peer-to-Peer configuration. Identical to the Peer-to-Peer system signal version (current version of the communication end point).

NewKeVer

Reserved for future applications.

Sub Window Tabs, Remote IO

Representation



STOP

The **Stop** button is used to stop the Remote I/O(s) selected in the **Remote I/O** table.

Step	Action
1	In the **Remote I/O** table click on a line number to select a Remote I/O. **Note:** You can select multiple Remote I/Os by holding down the CTRL key while clicking on line numbers, or by pressing the SHIFT key to mark the first and the last Remote I/O.
2	Click the **Stop** button.
3	Confirm the safety request with **OK**.

Synchronize

The **Synchronize** button is used to start a Remote I/O and to synchronize it with the parent resource. To synchronize the Remote I/O click on a line number within the table to select a Remote I/O. **Note:** In order to use **Synchronize**, the parent resource must be in **RUN**

Step	Action
1	In the **Remote I/O** table click on a line number to select a Remote I/O.
2	Click the **Synchronize** button.
3	Confirm the synchronize request with **OK**.

Remote IO table

The **Remote I/O** table displays the following information:

- Name of the Remote I/O
- System Rack ID of the Remote I/O
- Status of the Remote I/O:
  - Run
  - Stop/Valid configuration
  - Stop/Invalid configuration
  - Error Stop
  - Not connected

Sub WIndow Tabs, HH State

Representation

	last	avg.	min.	max.
Bus Cycle Time [ms]	0	0	0	0

Resource	Link ID	State	RspT last	RspT avg.	RspT min.	RspT min.	LinkMode	TokenGro...
HIMatrix F1 DI 16	16#000B0081	--	0	0	0	0	TCS direct	214748364

Bus Cycle Time

The **Bus Cycle Time** is the maximum permitted time in milliseconds for a Token Cycle. Within the **Bus Cycle Time**, a resource expects the Token which it passed on to be returned.

The **Bus Cycle Time** is dependent on the number of PLCs/Remote IO in a Token Group and can be found in the **HH Status** of the **Control Panel**.

If Token Passing is switched off, the **Bus Cycle Time** is 0.

Resource

The name of the Remote IO or PLC

LinkID

The Network ID of the Remote IO or PLC

State

Communication state

RspT (last, avg., min., max.)

If **Link mode** is TCS direct (Token Passing OFF), the response time (RspT) is the time for a data packet from PES1 -> PES2 -> PES1, depending on the network hardware and topology. This parameter cannot be set by the user.

If **Link mode** is TCS TOKCYC (Token Passing ON), the response time is part of the Token Cycle Time.

Link Mode

Value	Description
TCS direct (Token Passing is OFF)	Safety-related data is sent as soon as it is formatted and is ready for transmission. Access to the network is coordinated through the network hardware (LAN switches).
TCS TOKCYC (Token Passing is ON)	Used in connection with the "normal" protocol mode. Safety-related data is only sent if the resource receives the Token which is in circulation. Access to the network is coordinated through the software.

Token Group ID

Token Group ID is the identification of the Token Group.

Sub Window Tabs, Environmental Data

Overview

The **Environment Data** tab displays status messages for the temperature, power supply, ventilator and error relay in hexadecimal form.

Representation

Temperatur State	16#00
Power Supply State	16#00
Fan State	16#FF
Relay State	16#00

Temperature State

Provides the current operating temperature level of the PLC. The following are the possible values:

- 00 Normal approximate temperature: T < 40°C
- 01 High approximate temperature: 40°C > T < 50°C
- 02 Error with measurement
- 03 Very High approximate temperature: T > 50°C

Power Supply State

The **Power Supply State** provides the power supply operating level to the PLC. The following are the possible values:

- 00 Normal
- 01 Problem with supply

Fan State (only available for the Modular Safety PLC)

The **Fan State** provides the Fan operating status for the Modular PLC. The following are the possible values:

- 00 Normal
- 01 Error (must be replaced)
- FF Not available (Normal for Compact PLCs and Remote IO)

Relay State (only available for the Modular Safety PLC)

Provides details of the relay contact state on the Modular Safety PLC.

Sub Window Tabs, OS (Operating System)

Representation

Serial No	98220065100090048001

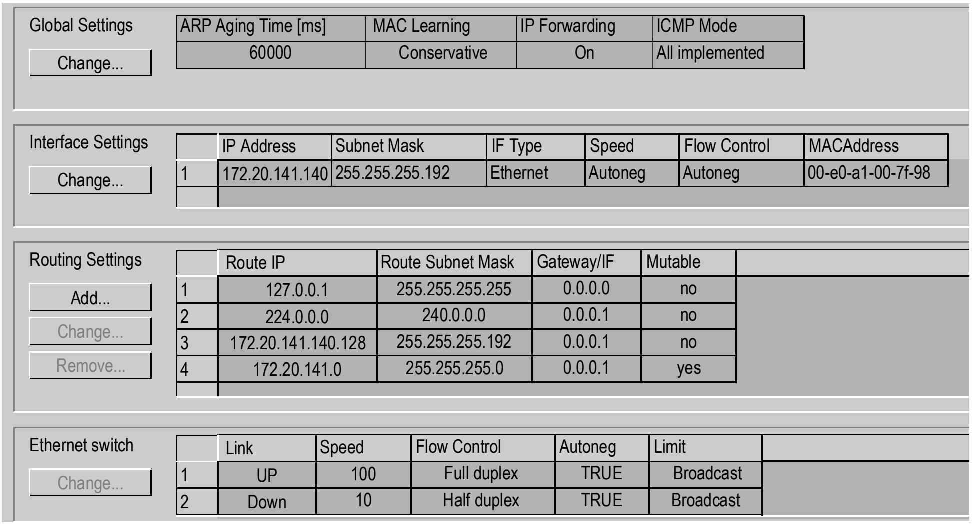
	Version	CRC
CPU OS	6.34	16#00730254
CPU OS Loader	6.12	16#AEAF61FB
CPU Boot Loader	4.56	16#E6D68E32
COM OS	11.24	16#F7C4B331
COM OS Loader	11.2	16#58C67B70
COM BOOT Loader	11.2	16#CED61556

Description

Element Name	Description
Serial Number	Serial number of the communication module
Serial number of the communication module	Version and checksum (Cyclic Redundancy Check, CRC) of the CPU operating system
CPU OS Loader	Version and checksum of the CPU operating system loader
CPU Boot Loader	Version and checksum of the CPU boot loader
COM OS	Version and checksum of the COM operating system
COM OS Loader	Version and checksum of the COM operating system loader
COM Boot Loader	Version and checksum of the COM boot loader

Sub Window Tabs, IP Settings

Representation



ARP Aging Time [ms]	MAC Learning	IP Forwarding	ICMP Mode
60000	Conservative	On	All implemented

	IP Address	Subnet Mask	IF Type	Speed	Flow Control	MACAddress
1	172.20.141.140	255.255.255.192	Ethernet	Autoneg	Autoneg	00-e0-a1-00-7f-98

	Route IP	Route Subnet Mask	Gateway/IF	Mutable
1	127.0.0.1	255.255.255.255	0.0.0.0	no
2	224.0.0.0	240.0.0.0	0.0.0.1	no
3	172.20.141.140.128	255.255.255.192	0.0.0.1	no
4	172.20.141.0	255.255.255.0	0.0.0.1	yes

	Link	Speed	Flow Control	Autoneg	Limit
1	UP	100	Full duplex	TRUE	Broadcast
2	Down	10	Half duplex	TRUE	Broadcast

Global Settings

Allows you to change the following:

- ARP Aging Time (ms)
- MAC Learning
- IP Forwarding
- ICMP Mode

Global Settings, ARP Aging Time (ms)

A PLC stores the MAC addresses of its communication partners in the ARP cache.

If a message from the communication partner arrives within 1x - 2x ARP aging time, the MAC address remains in the ARP cache.

If a message from the communication partner does not arrive within 1x - 2x ARP aging time, the MAC address is removed from the ARP cache.

The contents of the ARP cache cannot be read out by the user.

The range of valid values is 1000 ms to 3600000 ms. The default value is 20000 ms.The typical value for the ARP aging time in a local network is 5000 ms to 20000 ms.

Note: The ARP aging time must be adapted (increased) to the additional delays of forward and return paths when using routers/gateways.
If the ARP aging time is too small, the MAC address of the communication partner is removed from the ARP cache and communication is terminated.

Global Settings, MAC Learning

With **MAC Learning** and **ARP Aging Time (ms)** the user sets, how quickly a MAC address is to be learned.

There are the following settings:

Setting	Description
Conservative	If MAC addresses of communication partners are already in the ARP cache, these entries are locked for the duration of at least 1 x ARP aging time to maximally 2 x ARP aging time and cannot be replaced by other MAC addresses. Thus it is guaranteed that packets cannot be - intentionally or unintentionally - rerouted to irregular network participant. If a network is extended by a PLC, **Conservative** does not apply: The new controller is recognized immediately and the MAC address is stored in the ARP cache of the communication partners. Note: If a PLC is exchanged, for which ARP Aging time = 1 hour and MAC Learning = Conservative, the communication partner learns the new MAC address the earliest after 1 hour to maximally 2 hours. In this time communication with the exchanged controller is impossible.
Tolerant	With the receipt of a message the IP Address in the message is compared with the data in the ARP cache and the stored MAC address in the ARP cache is immediately overwritten with the MAC address from the message. Use the **Tolerant** setting, if you use routers with redundant interfaces and the availability of communication is more important than authorized access on the PLC.

Global Settings, IP Forwarding

P-Forwarding enables a PLC to work as a router and forward messages of other nodes.

Setting	Description
ON	IP-Forwarding is enabled
OFF	IP-Forwarding is disabled

Note: Do not use IP forwarding on Remote I/O.

Global Settings, ICMP Mode

ICMP is often considered part of the IP layer. It communicates error messages and other conditions that require attention. ICMP messages are usually acted on by either the IP layer or the higher layer protocol (TCP or UDP). Some ICMP messages cause errors to be returned to user processes.

ICMP Mode	Ping	ICMP error messages	Description
Don't answer	Off	Off	All ICMP commands are switched off. This makes the Remote I/O safe against sabotage via the network.
Echo response	On	Off	If **Echo response** is switched on, the node responds to a Ping command. As a result it can be determined whether a node is reachable. Safety is still high.
Host unreachable	Off	On	Not of importance for the user. Only for tests with the manufacturer.
All implemented	On	On	All ICMP commands are switched on. Thus a more exact fault-tracing of network errors is obtained.

Note: For resources the ICMP mode is constantly set to «All implemented». Only Remote I/Os the settings can be changed

Interface Settings

Allows you to change the following:

- IP Address
- Subnet Mask

Interface Settings, IP Address

The **IP Address** is the address under which the PLC is logged on in the network. This enables the controller to be clearly identified in the network.

The **IP Address** can be assigned to the PLC according to user-requirements and is defined in the COM properties of the controller and saved in the project.

An **IP Address** is a four byte binary number and consists of so called octets, for example, 192.12.144.65. The **IP Address** is comprised of the network address, the subnet address and the host address.The host address, also called Host ID, identifies the individual controller.

Interface Settings, Subnet Mask

The **Subnet Mask** is a 32 bit address mask which is used to subdivide the **IP Address** into the network address and the host address.

> **Note:** Any changes to these settings should be made within the project as the program must be recompiled and downloaded again.

Routing Settings

Routing allows data exchange between PLCs in different subnets. In the **Routing Settings**, a maximum of eight user-defined routes can be entered. The buttons **Add**, **Change** and **Remove** are used to add, change and remove routing entries.

Routing Settings, Route IP and Route Subnet Mask

Route IP and **Route Subnet Mask** define, for which target address range a routing entry is applied.

The target addresses ranges are: 1.0.0.0...223.255.255.255 (Class A, B, C).

Routing Settings, Gateway/IF

Node, over to which the packets are sent to the target address range.

Typical values:

- 0.0.0.1 Fixed designator for the Ethernet interface of a PLC
- 192.168.2.251 Example for the IP Address of a Gateway

Routing Settings, Changeable

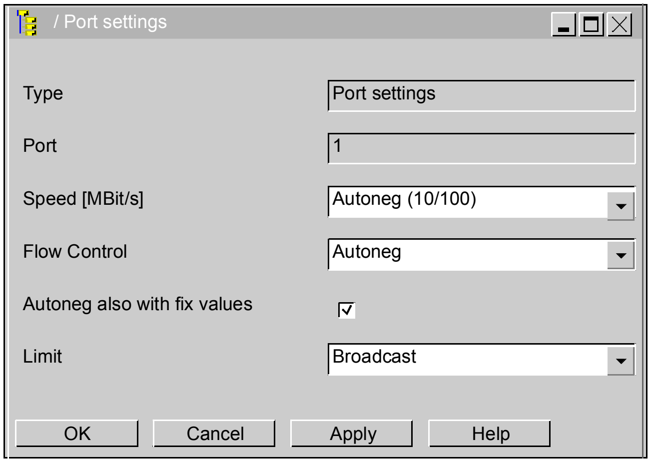
The eight user-defined routes are alterable. In addition to the user-defined entries not alterable routes displayed, e.g.

- 127.0.0.1 Loop Back Interface
- 192.168.4.0 Subnet Route to the subnet in which the PLC is located
- 192.168.4.1 Host Routing to the PC

Ethernet Switch

Enables the user to adjust the Ethernet communications settings for each of the switched Ethernet ports on the PLC.

The settings should be adjusted in the **Hardware Management** within the COM of each PLC.



Sub WIndow Tabs, Modbus Master

Licenses

To use the Safety PLC as a **Modbus Master** (TCP/IP or Serial) to control non-safety related devices such as non-safety related Remote I/O modules you must purchase either or both of the following licenses:

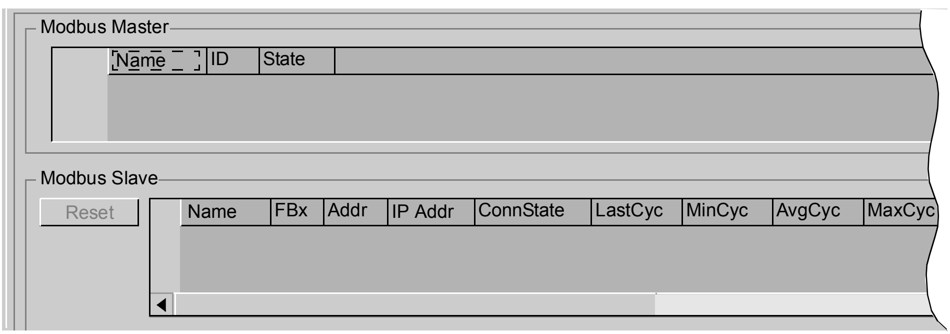
- XPSMFMMRTU
- XPSMFMMTCP

It is possible to use the product as a master for a limited period for testing etc. without purchasing a license. However, after the trail period the PLC will return to the Stop state!

Communication Details

The communication details when using the **Modbus Master** functionality are available within the online help (**Hardware Management** window).

Representation



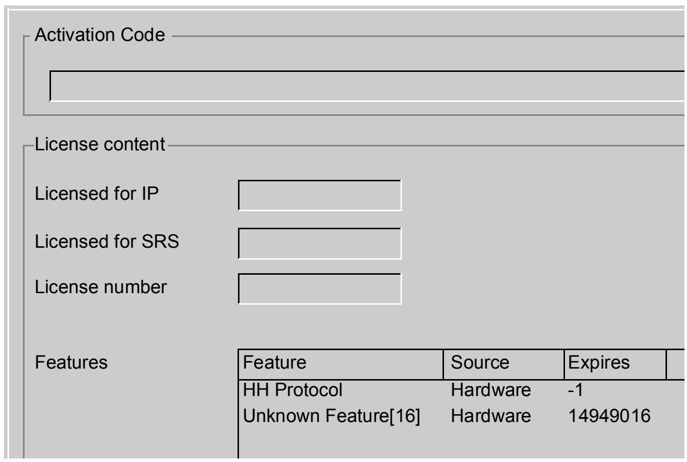
Sub Window Tabs, License

Overview

Within the **License** tab information is provided to which protocols are available within the PLC. The details of any additional licenses are available here after entering the activation code within the COM of the resource.

The license **Unknown Feature** is available for use during which it is possible to use Modbus TCP/IP client, Modbus RTU Serial Master (if internal hardware is available), and Ethernet/IP protocols for a limited duration.

Representation

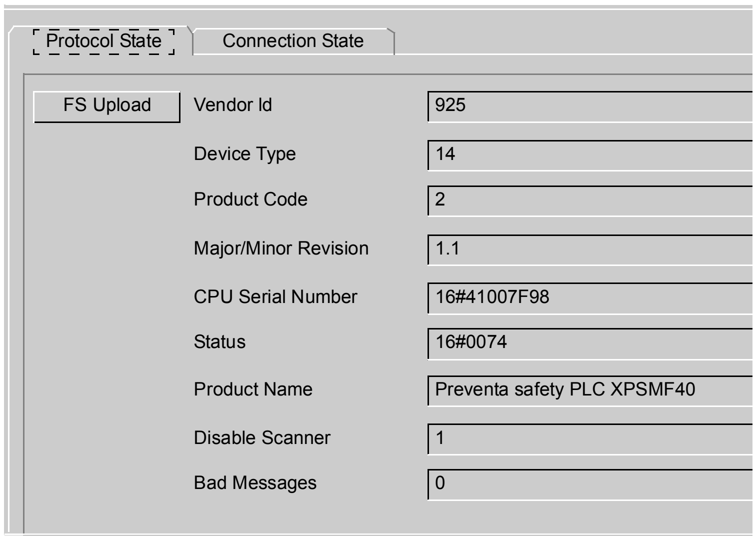


Sub Window Tabs, Ethernet IP

Overview

The license for **Ethernet IP** will be made available around End 2007. For more information regarding this protocol please see *Online help*.

Representation



6.13 Diagnostics Window

At a Glance

Overview

This section provides information on the **Diagnostics** window.

What's in this Section?

This section contains the following topics:

Diagnostics WIndow

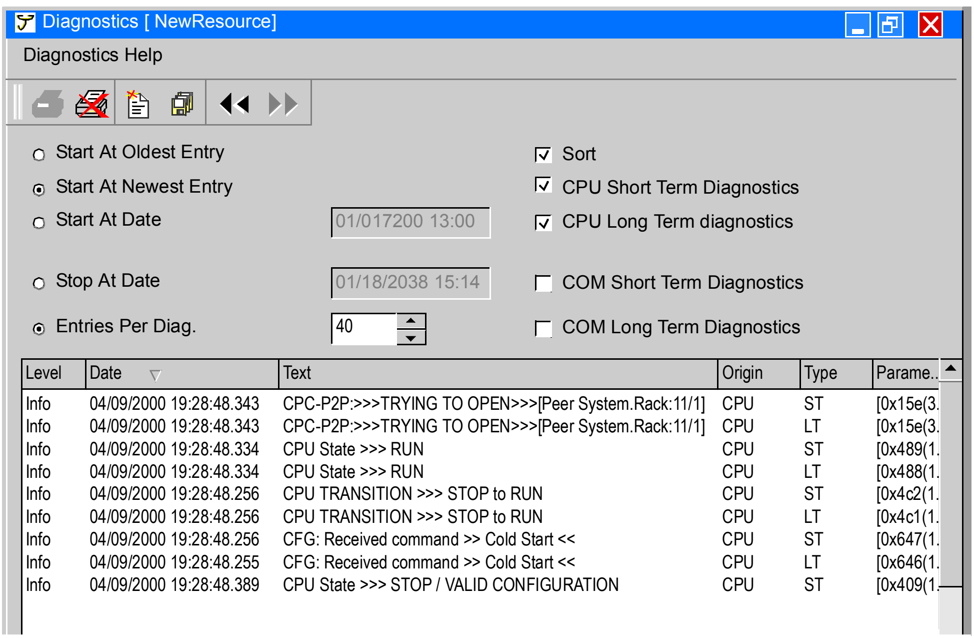
Overview

The **Diagnostics** window displays the entries saved within the long term and short term diagnostic memory of a PLC or Remote I/O.

> **Note:** The number of diagnostic entries that can be saved in a PLC or Remote I/O depends on the product type.

The **Diagnostics** window is controlled by the commands within the menu bar and the buttons in the toolbar. The diagnostic data can be filtered in a number of ways and is displayed in a table with different colors to improve the readability.

Representation



Diagnostic Menu

Overview

The **Diagnostic** menu contains the following functions:

- Diag. Online
- Diag. Offline
- Clear
- Export...
- Previous Entries
- Next Entries

Note: The functions of the **Diagnostics** menu can also be carried out by the buttons in the toolbar.

Diag. Online

The **Diag. Online** function automatically becomes active after the diagnostic is started. Within a very short space of time, all diagnostic data is transferred from the resource into the buffer of the programming terminal.

The diagnostic data is filtered according to the user-defined filter settings and displayed in the diagnostic table.

As long as **Diag. Online** is active, all new diagnostic data resulting from the current state of the resource is read into the buffer of the programming terminal. If the data match the filter, it is transferred to the diagnostic table.

This enables the short term diagnostic to be recorded with the programming terminal over a long period of time, even if the short term diagnostic memory in the resource is full and older entries are already being deleted.

Note: Diag. Online can also be carried out by the button in the toolbar.

Diag. Offline

Diag. Offline ends communication with the resource. New diagnostic data is no longer read into the buffer of the programming terminal and transferred into the diagnostic table

Note: Diag. Offline can also be carried out by the button in the toolbar.

Clear

The **Clear** function erases the buffer of the diagnostic data in the programming terminal. At the same time, all entries are removed from the diagnostic table.

If the diagnostic is restarted (i.e closed and reopened), the diagnostic buffer of the resource is read out again. The readout does not delete the diagnostic memory of the resource.

Note: Clear can also be carried out by the button in the toolbar.

Export...

The **Export...** function enables all diagnostic data located in the buffer of the programming terminal to be saved in a text file with the extension *.log*.

A standard Windows® dialog box opens in which the path and file name can be entered.

Note: Export can also be carried out by the button in the toolbar.

Previous Entries

The **Previous Entries** function allows the user to scroll forwards through the diagnostic data in the buffer.

The entries are filtered and displayed in the diagnostic table.

Note: Previous Entries can also be carried out by the button in the toolbar.

Next Entries

The **Next Entries** function allows the user to scroll backwards through the diagnostic data in the buffer.

The entries are filtered and displayed in the diagnostic table.

Note: Next Entries can also be carried out by the button in the toolbar.

Options

Start at Oldest Entry

Start At Oldest Entry transfers all data (starting with the oldest entry) from the buffer into the diagnostic table, the oldest entry being at the end of the table.

The number of rows in the diagnostic table depends on the **Entries per Diag.** setting.

Sort is switched off as standard so that the data appears in the chronological order that it was recorded in the resource.

Start at Newest Entry

Start At Newest Entry (default setting) transfers all data (starting with the newest entry) from the buffer into the diagnostic table, the newest entry being at the start of the table.

Start at Date

Start at Date defines the date and time from which data is displayed in the diagnostic table.

The entry, the date and time which matches the value entered or which is only slightly more recent, is located at the end of the table.

The date and time must be entered in the format displayed. If no time is entered, 00:00 hr is accepted.

The number of rows in the diagnostic table depends on the **Entries per Diag.** setting.

Stop at Date

Stop at Date defines the most recent date and time up to which data is displayed in the diagnostic table.

The entry, the date and time which matches the value entered or which is only slightly more recent, is located at the end of the table.

Entries Per Diag.

Entries Per Diag. allows the user to define the maximum number of entries per diagnostic type that the table can contain.

If all 4 diagnostic filters are switched on and **Entries Per Diag.** = 20, the table can contain a maximum of 80 rows.

The available diagnostic filters are
- CPU Short Term Diagnostic,
- CPU Long Term Diagnostic,
- COM Short Term Diagnostic, and
- COM Long Term Diagnostic.

The software buffer is designed for a maximum of 5000 entries per diagnostic type.

Note: Sorting and filtering large amount of data places demands on the performance of the computer. For the online check, normally 100 entries per diagnostic are sufficient.

Sort

As long as the **Sort** check box is switched off, all the data in the diagnostic table is displayed in the same order as it was recorded in the resource.

The data is grouped according to CPU, COM and long term and short term diagnostic.

After switching on **Sort**, all the data in the diagnostic table is sorted according to date. The sorting direction is indicated by a small triangle in the column header.

By clicking on the column header, the sorting direction can be reversed or another column selected for sorting.

Short and Long Term Diagnostics

By checking the filter criteria boxes for short and long term diagnostic, the user can determine which diagnostic data of the CPU and/or the COM should be displayed in the diagnostic table.

Note: The **CPU Long Term Diagnostic** is switched on as standard.

Properties

Level

The different types of level are:
- Info
- Warning
- Error

Date

Displays the date and time from the PLC or Remote I/O when the diagnostics entry has been recorded.

Text

Description of the event recorded

Origin

The diagnostic entry can come from either the CPU or the COM.

Type

Identifies the entry either as:
- Long term diagnostic (**LT**)
- Short term diagnostic (**ST**)

Parameters

This information comes directly from the CPU or the communication module and is of no significance for the user.

It is only used for error analysis by the manufacturer.

Abbrevations

Abbrevations in Diagnostics

Abbrev.	Description	Origin
BO	Boot Booting and download of OS	[CPU/COM Module]
BT	Basic dataStores IP, routing parameters, System Rack Slot ID etc.	[Remote I/O/ COM Module]
CC/TCC	Internal Dual Port RAM communication between CPU and COM.	[COM Module]
CFG	CPU configuration and instruction processingFor nearly all commands from the Control Panel to the CPU	[CPU Module]
CI	COM configuration and Instruction processing For nearly all commands from the Control Panel to the COM	[COM Module]
CNN/TNN	Network configuration of the COM	[COM Module]
CNO/TCN	Not safety-related communication of process data from COM to CPU	[COM Module]
CO	Operating System kernel	[CPU Module]
COM	General COM messages	[COM Module]
CPC	Communication of process data Loading configuration, Peer-to-Peer coordination (TCS) and non-safe protocols (TCN) to the COM. Also for handling the Communication Time Slice.	[CPU Module]
CPD/TDI	Readout of long term and short term diagostic	[CPU Module]
CS	Configuration server for Remote I/O For loading, starting and stopping a Remote I/O.	[COM Module]
DCP/TCC	Internal Dual Ported RAM communication between CPU and COM	[CPU Module]
DD	COM process data storage	[COM Module]
DX	Data Exchange For system variables and emergency shutdown.	[CPU Module]
FS/TPS	File system of the COM Stores the configuration of the PES. Read out by COM, CPU and Remote I/O. Also used for check of consistency.	[COM Module]
HHC, HHP, HHH, HHS, HHL	Hima-Hima Protocol on EthernetTransportation media for P2P and HIMA OPC Server.	[CPU/COM Module]
HW	Hardware Driver	[CPU/COM Module]
IN/TNN	InternetTCP/IP protocol stack for Remote I/O. Used for PADT, Master-COM and P2P.	[Remote I/O]
IOA	I/O processing Loading, cyclic I/O processing.	[CPU Module]

Abbrev.	Description	Origin
IOT	CPU self tests	[CPU Module]
KEM/TCD	Central process data storage of the CPUFor process data retention (signals), loading process data configuration and all Forcing activities.	[CPU Module]
LS	Logic Solver For loading, execution and testing (Online Test) of the application program.	[CPU Module]
MC	Central sequence monitoring	[CPU Module]
MCM	Central sequence monitoring of the COM	[COM Module]
MOD	Modbus Protocol	[COM Module]
NL	Network driver	[Remote I/O/ COM Module]
PBD	PROFIBUS Protocol	[COM Module]
PGS	Interface to Hardware anagement and to Remote I/O Also used for Login.	[Remote I/O/ COM Module]
SNTP	Time synchronization protocol	[Remote I/O/ COM Module]
TCO/TSS	Internal message passing	[CPU/COM Module]
TOC	Internal memory management	[CPU/COM Module]
TOM/TOS	Generic internal functions	[CPU/COM Module]
TPG	Basic messages between CPU - COM - PADT - Remote I/O	[CPU/COM Module]

6.14 Signal Monitoring with the Force Editor

At a Glance

Overview

This section describes signal monitoring with the force editor.

What's in this Section?

This section contains the following topics:

General

Overview

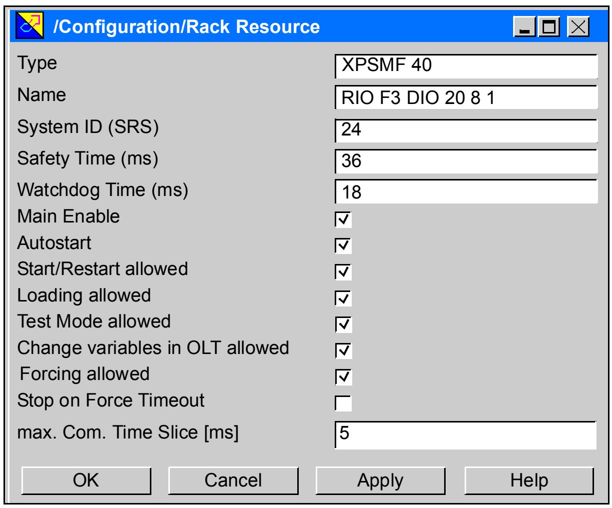
The **Force Editor** is opened for an individual resource.

All signals assigned to this resource are available in the force editor.

If the signals of more than one resource have to be monitored simultaneously, the force editor can be opened several times.

Forcing

Note: The following "Forcing" option must be selected to enable forcing:



⚠ WARNING

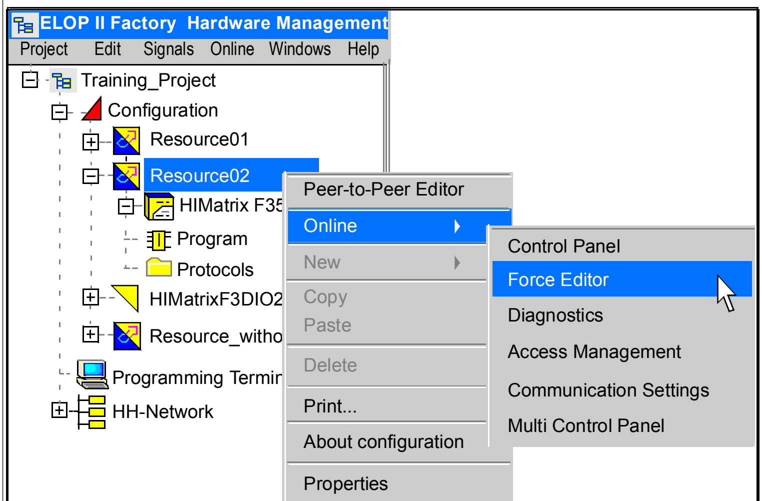
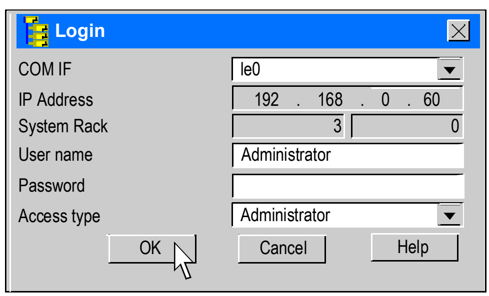
UNINTENDED EQUIPMENT OPERATION

Do not use Forcing during normal operation.
Limit use of Forcing to system commissioning and de-bugging.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

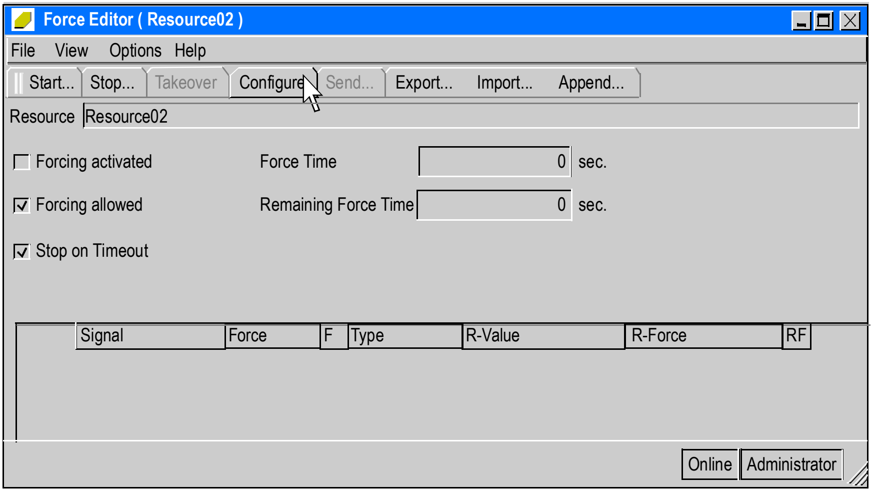
Signal Monitoring with the Force Editor

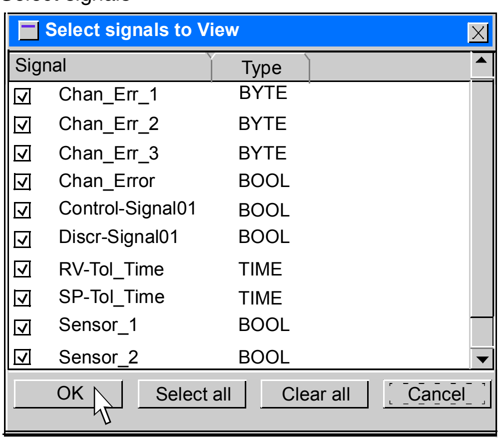
Step 1: Open the Force Editor

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Click Online → Force Editor in a resource context menu.
Opening the force editor

</td></tr>
<tr><td>2</td><td>Login is required if there is no communication between the resource and the programming terminal.
Login dialog

</td></tr>
</table>

Step 2: Selecting Signals

Select signals for viewing in the force editor.

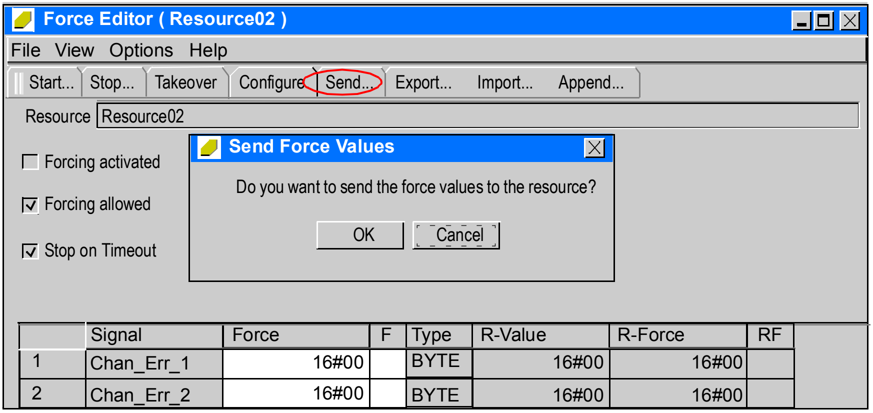
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Initially there are no signals in the Force Editor. Click Configure to open the Select signals to View window.
Force Editor


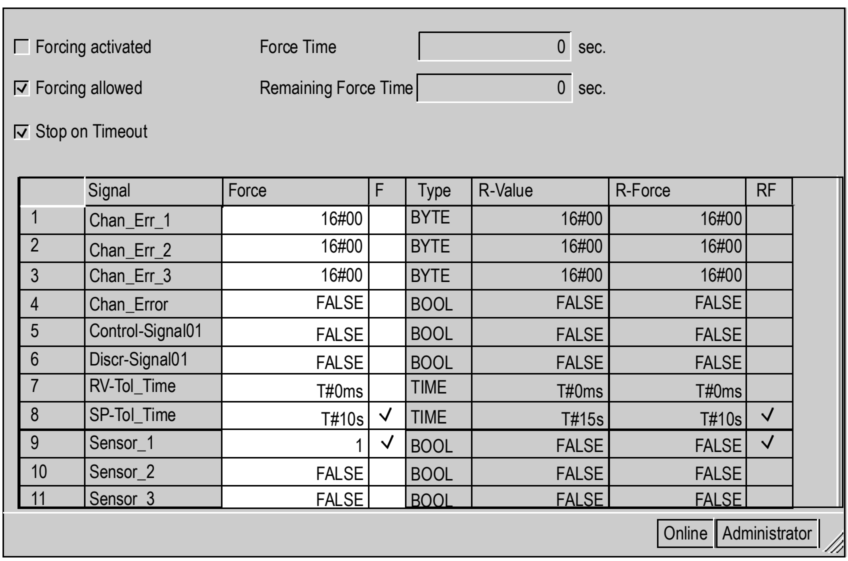
</td></tr>
<tr><td>2</td><td>In the Select signals to View window you can choose signals, which you want to force or display in the force editor.
Click the selection boxes or Select all. Close with OK
Select signals


</td></tr>
</table>

Step 3: Prepare for Forcing

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Enter the desired force value in the Force column. The input is activated by double-clicking it.</td></tr>
<tr><td>2</td><td>In column F specify whether the signal is to be forced (double-click).
Signal list in the force editor
<table>
<tr><th></th><th>Signal</th><th>Force</th><th>F</th><th>Type</th><th>R-Value</th><th>R-Force</th><th>RF</th></tr>
<tr><td>1</td><td>Chan_Err_1</td><td>16#00</td><td></td><td>BYTE</td><td>16#00</td><td>16#00</td><td></td></tr>
<tr><td>2</td><td>Chan_Err_2</td><td>16#00</td><td></td><td>BYTE</td><td>16#00</td><td>16#00</td><td></td></tr>
<tr><td>3</td><td>Chan_Err_3</td><td>16#00</td><td></td><td>BYTE</td><td>16#00</td><td>16#00</td><td></td></tr>
<tr><td>4</td><td>Chan_Error</td><td>FALSE</td><td></td><td>BOOL</td><td>FALSE</td><td>FALSE</td><td></td></tr>
<tr><td>5</td><td>Control-Signal01</td><td>FALSE</td><td></td><td>BOOL</td><td>FALSE</td><td>FALSE</td><td></td></tr>
<tr><td>6</td><td>Discr-Signal01</td><td>FALSE</td><td></td><td>BOOL</td><td>FALSE</td><td>FALSE</td><td></td></tr>
<tr><td>7</td><td>RV-Tol_Time</td><td>T#0ms</td><td></td><td>TIME</td><td>T#0ms</td><td>T#0ms</td><td></td></tr>
<tr><td>8</td><td>SP-Tol_Time</td><td>T#10s</td><td>√</td><td>TIME</td><td>T#15s</td><td>T#0ms</td><td></td></tr>
<tr><td>9</td><td>Sensor_1</td><td>1</td><td>√</td><td>BOOL</td><td>FALSE</td><td>FALSE</td><td></td></tr>
<tr><td>10</td><td>Sensor_2</td><td>FALSE</td><td></td><td>BOOL</td><td>FALSE</td><td>FALSE</td><td></td></tr>
<tr><td>11</td><td>Sensor_3</td><td>FALSE</td><td></td><td>BOOL</td><td>FALSE</td><td>FALSE</td><td></td></tr>
</table>
Note: Boolean values can be input with <code>TRUE</code> and <code>FALSE</code> or with <code>1</code> and <code>0</code>. Note the unit in the case of time signals.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>
The force values and the selection of the signals for forcing must be transferred to the PLC. To do this click the Send button.
Sending the force values


Overview of signals in the Force Editor after sending. Forcing is not activated yet!
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.
Note: If the force value is active when forcing is started depends on if there is a tick in the RF column.
</td></tr>
</table>

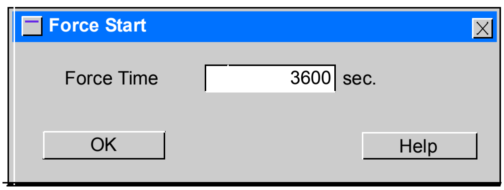
Step 4: Start the Forcing

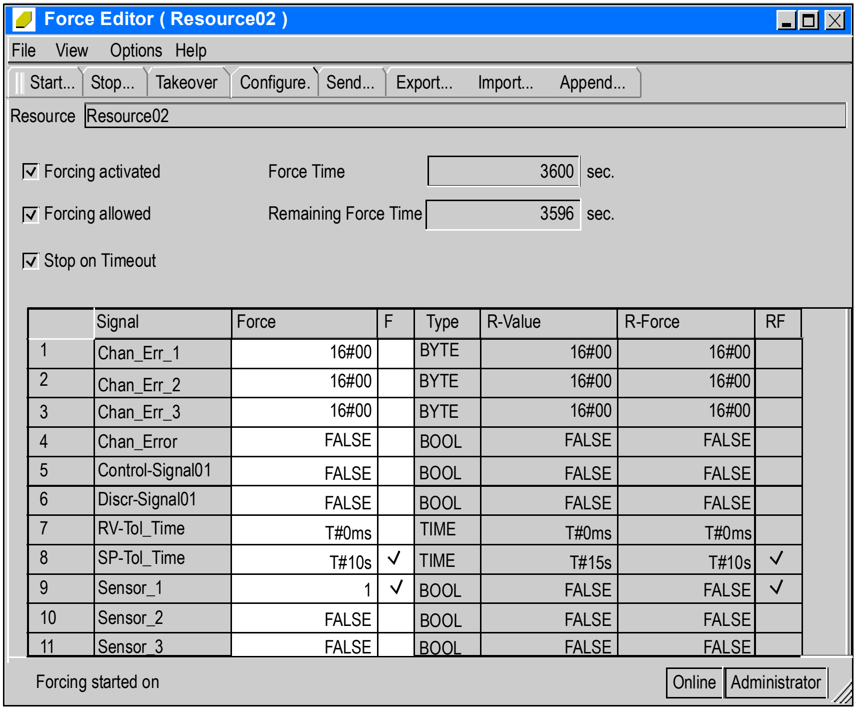
⚠ CAUTION

UNINTENDED EQUIPMENT OPERATION

Be aware that when starting the forcing 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 PLC returns to STOP after the forcing process.

Failure to follow these instructions can result in injury or equipment damage.

Step	Action
1	Pressing the **Start** button opens a prompt for the force time in seconds. Setting the force time 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Press OK to start the forcing.
Result: Forcing activated


</td></tr>
<tr><td>3</td><td>Forcing can be stopped manually with the Stop button. In this case the PLC remains in the RUN state, because the timeout was not reached.</td></tr>
</table>

Saving and Loading Signal Selection for Signal Monitoring

Overview

A signal selection can be saved and loaded again at a later time so you do not lose control of the number and variety of signals.

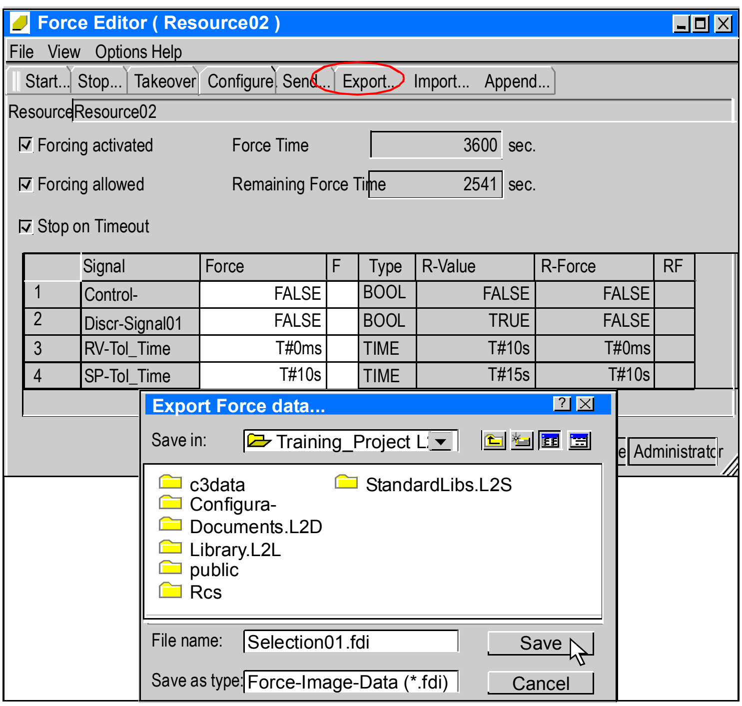
Frequently the same signals are needed over and over again in the force editor.

Step 1: Prepare the Selection

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

Step 2: Save the Selection

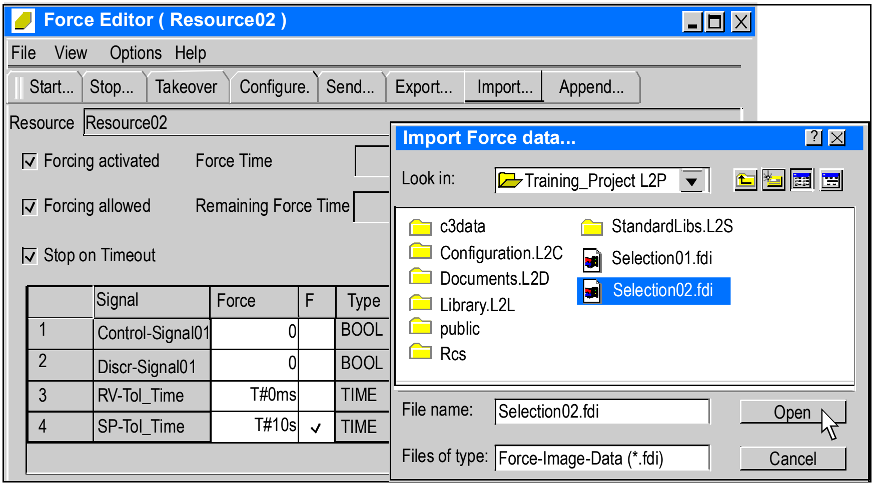
Step	Action
1	Press **Export** to save the selection under the desired name with the extension **.fdi*. Save signal selection (export)



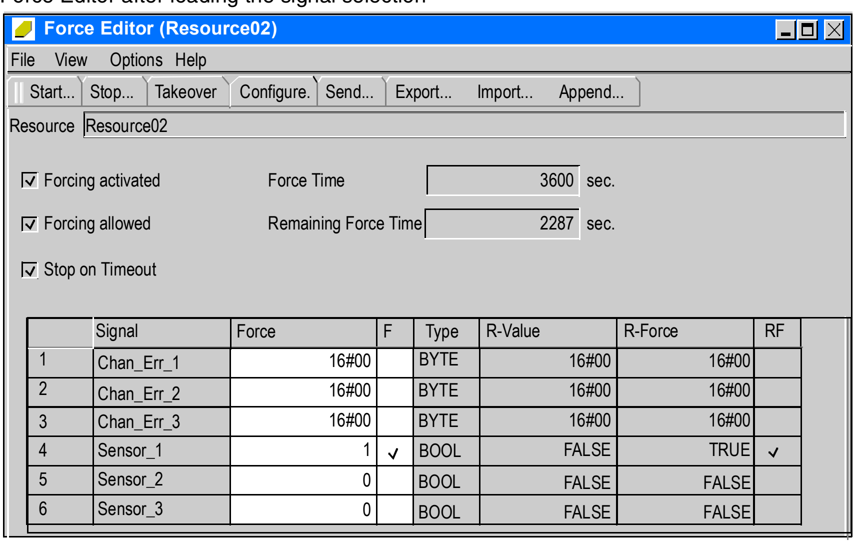
Step 3: Load Signal Selection

Load the saved signal selection with **Import**.

Step	Action
1	Press **Import** to open the **Import Force data...** dialog box.
2	Select the desired file and click **Open** or double-click the file directly. Load signal selection (import)

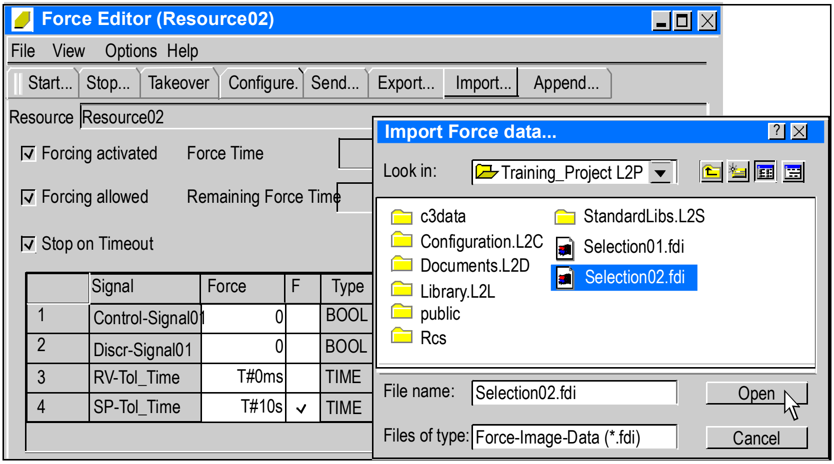
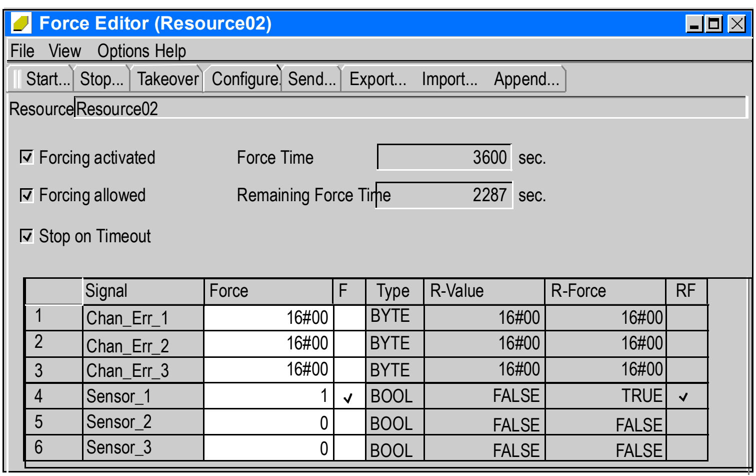


Note: The former signals are replaced by the loaded selection.
Force Editor after loading the signal selection



Step 4: Append Signal Selection

Load a saved signal selection with **Append**.

Step	Action
1	Press **Append** to open the **Append Force data...** dialog box.
2	Select the desired file and click **Open** or double-click the file directly. Load signal selection (import)  **Result:** The previous signals remain in the force editor and the newly loaded signals are appended to the signal list. Force editor after appending the signal selection  **Note**: 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.

6.15 Access Administration

Access Management

Overview

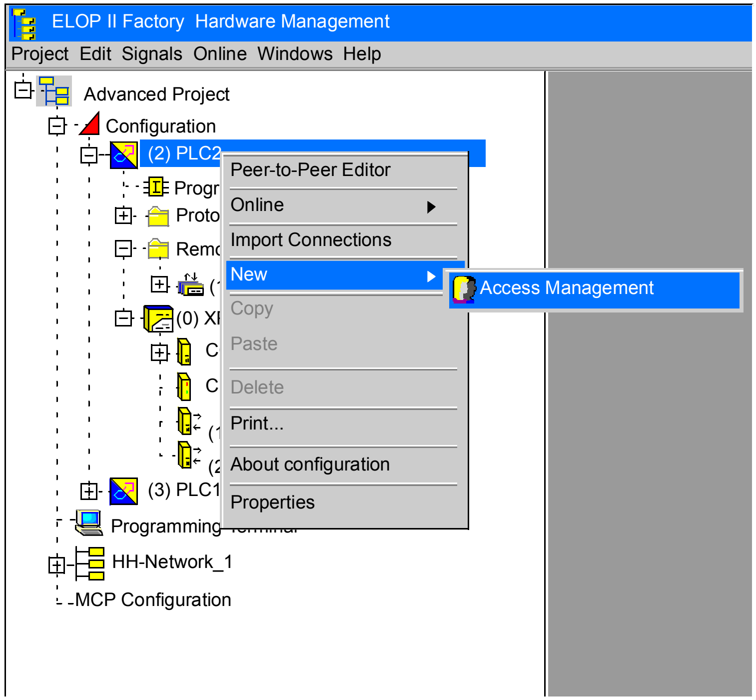
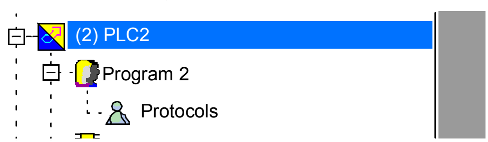
The access management enables a system engineer to set and manage access rights for a maximum of 10 users per controller. The access rights are stored in the buffered NVRAM of the controller and remain there even after disconnecting the operating voltage.

Note: Downloading new user accounts overwrites the user accounts previously stored on the controller or cleans the **Administrator** standard setting.

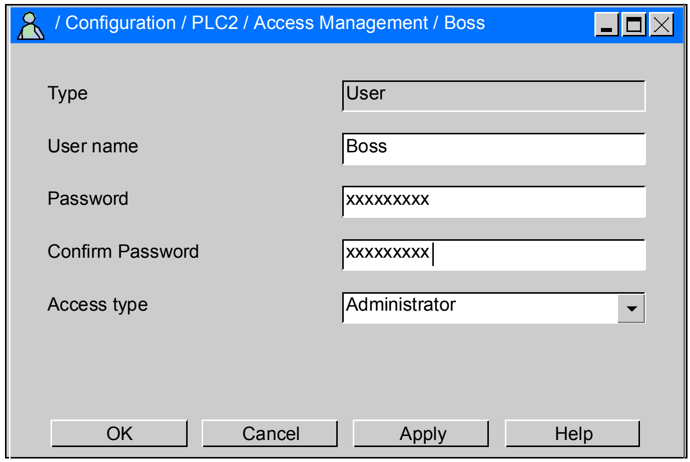
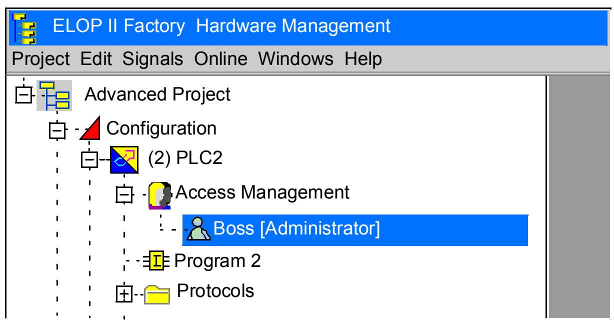
This also applies to user accounts generated online with **Online** → **Access Management**.

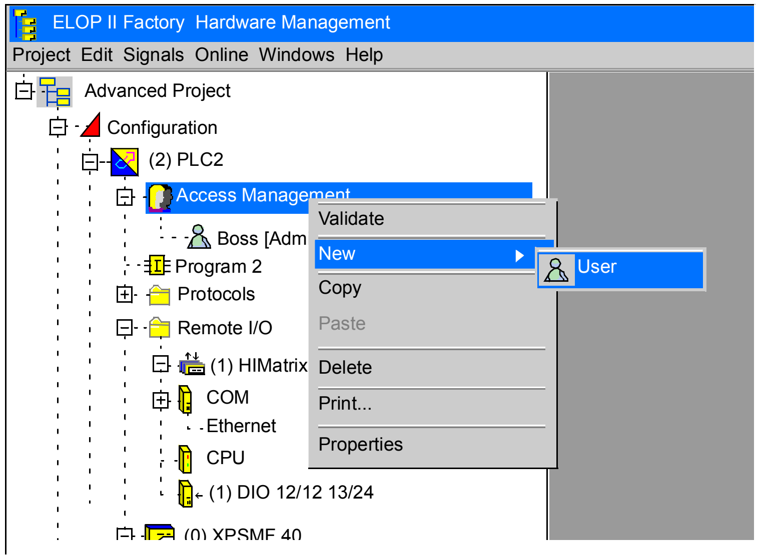
Setting Up a Resource

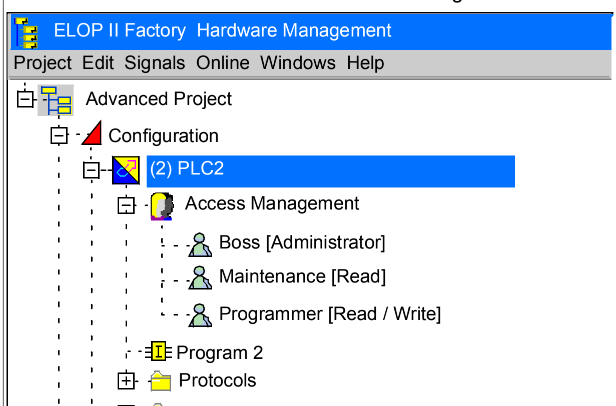
Each resource (PLC) must be set up individually.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>To assign access administration, right-click on PLC2 from the Configuration → New → Access Management.

</td></tr>
<tr><td>2</td><td>The Access Management folder appears under the Resource (PLC2) with a single user in read only mode.


You must begin by setting up a new administrator for the PLC. When there is no administrator you will not have complete access to the PLC!</td></tr>
<tr><td>3</td><td>Right-click on the default User (Read) → Properties.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>Enter a User name, Password and Access type (Administrator).

</td></tr>
<tr><td>5</td><td>Select Apply and OK. This will now appear within the place of the default user.

</td></tr>
</table>

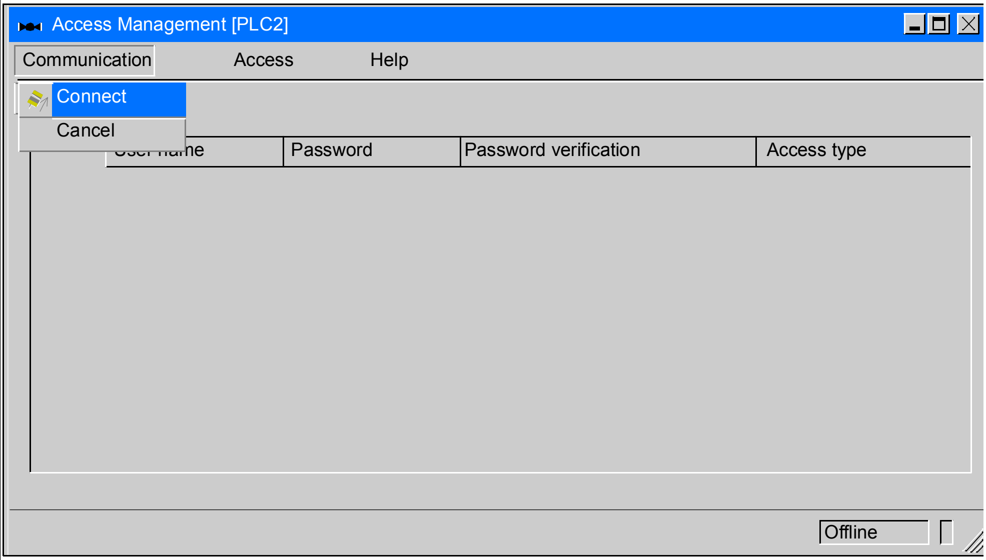
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>Enter now 2 users, one with Read access and one with Read/Write access by right-clicking on Access Management → New → User and filling the appropriate data.


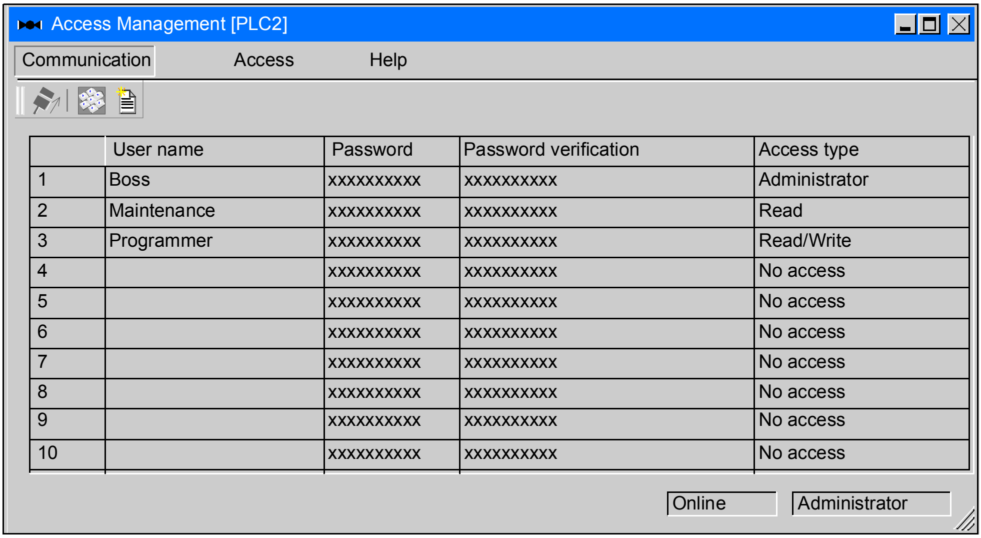
Your window should look similar to the following:


The accounts will be activated on download to the PLC.</td></tr>
<tr><td>7</td><td>Repeat procedure for PLC1.
Note: Remote I/O do not require access management set up, as this is managed via the parent resource (PLC).</td></tr>
</table>

Connecting Users

Once the administration settings have been downloaded via the program to the PLC, it is possible to connect and view all user data stored in each PLC.

Step	Action
1	Right-click on the resource (**PLC1**) from the **Hardware Management** window and select **Online** → **Access Management**.
2	Within the pop up window select **Communication** → **Connect**. 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>Enter the User name and Password for the administrator.

</td></tr>
</table>

Basic Project Configuration

7

At a Glance

Overview

This chapter provides information on the configuration of a basic project.

What's in this Chapter?

This chapter contains the following topics:

Introduction

Overview

This section is provided to guide you, as a new user, through the programming environment by means of creating a simple program.

11 Steps

The configuration is composed of 11 simple and easy to follow steps.

Creating a basic project includes the following steps:

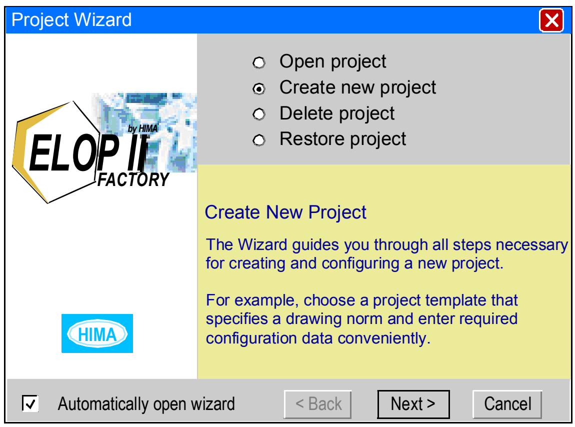
Step	Action	Time to Complete
1	Creating a New Project Using the **Project Wizard**	3 minutes
2	Opening a Configuration and Creating a Program	2 minutes
3	Editing a Program	8 minutes
4	Offline Simulation of Program	5 minutes
5	Defining Hardware and Settings	5 minutes
6	Setting PC Communication	2 minutes
7	Setting Hardware I/O	1 minute
8	Setting PLC Communication	1 minute
9	Code Generation and Program Size	2 minutes
10	Online, Download and Run	3 minutes
11	Online Test with Hardware	3 minutes

Step 1: Creating a New Project Using the Project Wizard

Connect the Hardlock

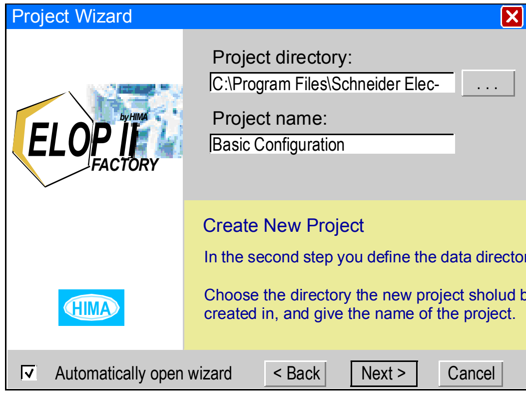
Connect the USB hardlock (dongle) to the programming terminal (PC).

Open XPSMFWIN

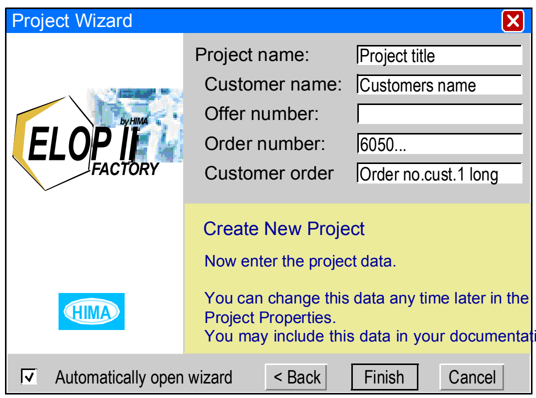
Step	Action
1	Select **Start** → **Programs** → **Schneider Electric** → **Safety Suite V1**. **Result:** The **Safety Suite** dialog box is displayed.
2	Click on the XPSMFWIN (ELOPII Factory) icon. **Result:** The **Project Wizard** dialog box is displayed. 

Create New Project

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Create new project and click Next.</td></tr>
<tr><td>2</td><td>Click Next again to confirm the Project template (ELOPII Factory Project).</td></tr>
<tr><td>3</td><td>As Project name enter Basic Configuration.



</td></tr>
<tr><td>4</td><td>Click Next.
Result:



</td></tr>
<tr><td>5</td><td>Enter customer related information and select Finish.
Note: The customer related details (project name, customer name etc.) will be included in documentation.</td></tr>
</table>

Open Project

The above is done every time you create a new project.

When opening an existing project, select **Open project** from the first window of the **Project Wizard**.

Step 2: Opening a Configuration and Creating a Program

Overview

After clicking **Finish** in the **Project Wizard**, 2 windows are opened:

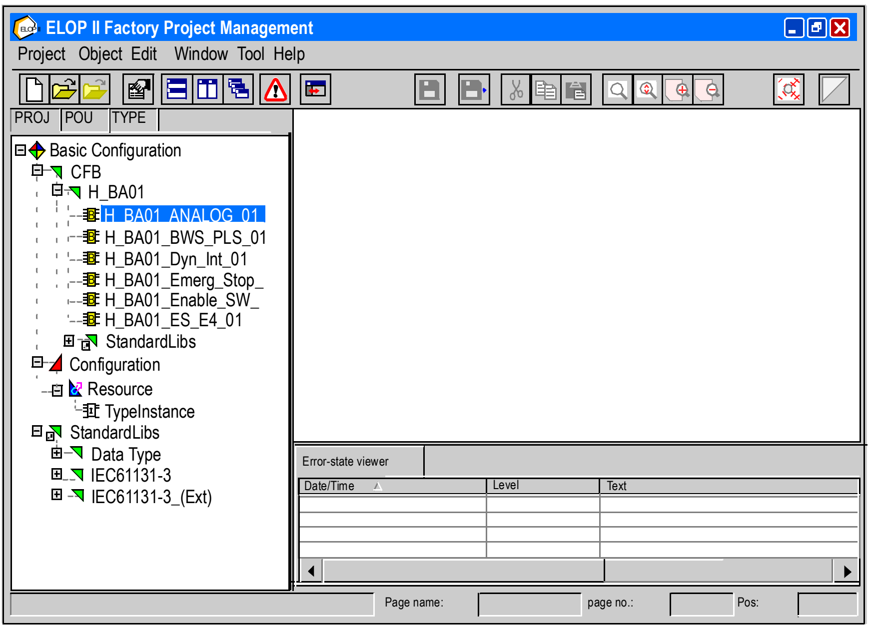
- Project Management
- Hardware Management

At first you will work using the **Project Management** window.

Project Management

Take a moment to familiarize yourself with the functionality of the **Project Management** window, by referring to the project management section of this manual (see *Project Management, p. 31*).

Within the structure window expand the **CFB**, **Configuration** and **StandardLibs** trees by selecting the **+**.



CFB

The **CFB** tree contains all certified function blocks (CFB) for various safety-related applications such as emergency stop.

Configuration

The **Configuration** contains a **Resource** with a **TypeInstance** connected.

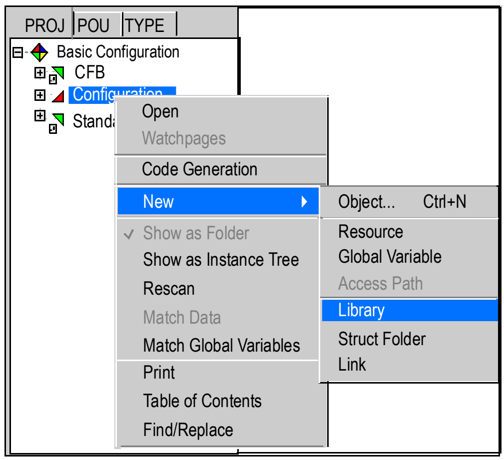
Within the configuration tree all resources can be created and connected to various user programs.

StandardLibs

The **StandardLibs** tree contains all IEC 61131-3 standard libraries such as logical gates, timers, mathematical functions, etc.

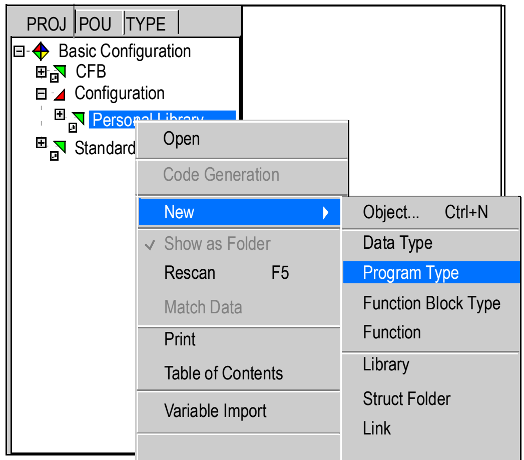
Creating Personal Library

The first step of creating a program is to create a personal library to store user programs and functions.

Step	Action
1	Right click on the **Configuration** to open the context menu.
2	Click on **New** → **Library**.  **Result:** A new library is created.
3	Change the name of the new library from **NewLib** to **Personal Library**. There are four possible ways to do this: ● 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. ● 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. ● Click on **NewLib**. Open the context menu by clicking the right mouse button. Select **Rename** and change the name to **Personal Library**. ● Click on **NewLib**. Press F2. Enter the new name and terminate the input with the ENTER key.

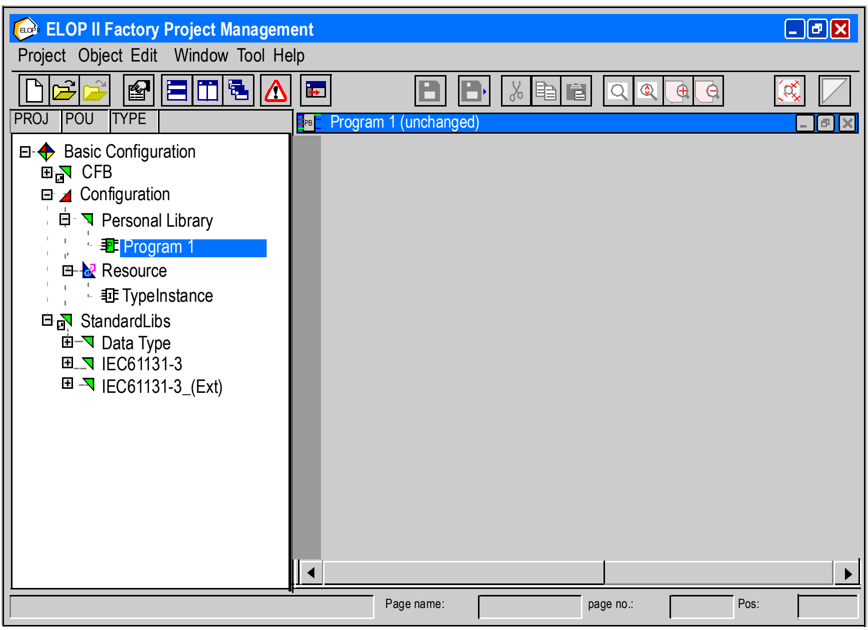
Creating a Program Type

Creating a program type allows all resources to be connected to a single program containing the same logic.

Step	Action
1	Right-click on **Personal Library** to open the context menu.
2	Select **New** → **Program Type**. 
3	Click on the **+** in front of **Personal Library**. **Result**: A new function block **NewProgramType**  is added to the structure.
4	Change the name of the new program type from **NewProgramType** to **Program 1**. **Note**: The program type contains the function logic that is to be executed in the PLC and can contain function blocks, as well as certified function blocks (CFB's).

Optimizing Working Area

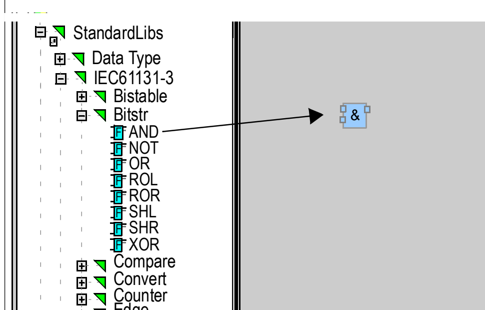
The new program type **Program 1** is now within your **Personal Library** folder.

Step	Action
1	Double-click on **Program 1** in your personal library folder. **Result:** The function block diagram (FBD) editor is displayed.
2	To optimize the working area hide the **Error-state Viewer** by clicking the  button. **Note:** When the **Error-state Viewer** has new information the  icon will flash in the Microsoft windows task bar (bottom right hand side of window).
3	To further optimize the working area hide the **Variable Declaration Editor** by clicking the  button. The **Project Management** should now look like the diagram below. 

Step 3: Editing a Program

Dragging a Function Block to Drawing Field

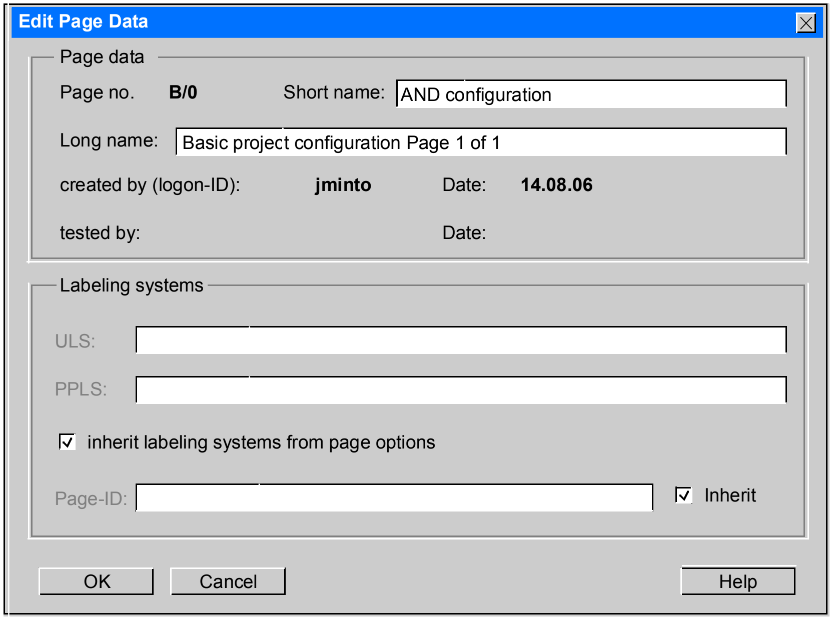
Drag-and-drop logic elements from the libraries to the drawing field

Step	Action
1	Open the library **StandardLibs** in the structure window by clicking on the **+** icon.
2	Open the library **IEC61131-3** from the **StandardLibs** and then **Bitstr**. **Note:** This folder contains all logical function blocks (`AND`, `OR`, `NOT` etc.)
3	Click on the `AND` function block, press and hold the left mouse button and drag the function block from the structure window into the drawing field. A preview of the logic element is displayed while dragging.
4	After releasing the mouse button the block is placed at the mouse pointer position.  **Note**: The procedure for inserting functions blocks from libraries by drag-and-drop is also valid for user libraries.

Editing Page Data

Because placing the `AND` function block is the first change to the contents of this page (see also *Function Diagrams with Centered Starting Point, p. 56*), the **Edit Page Data** dialog opens automatically.

Enter the **Short name** and **Long name** for the page.



These details will be included in documentation for the project.

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.

Signals for Physical I/Os

You now require 2 inputs and 1 output connection.

These connections are directly made to 2 inputs of the safety PLC and 1 output on the safety PLC.

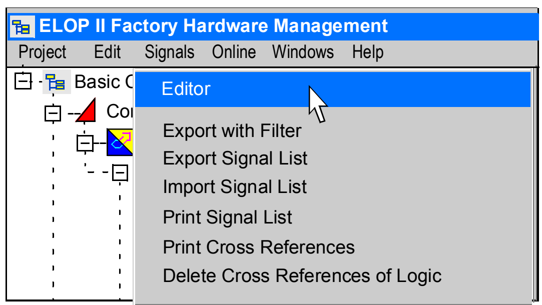
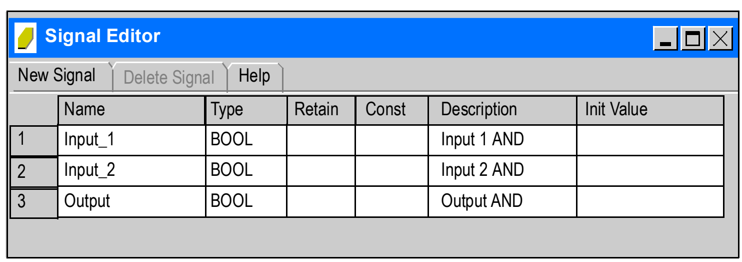
In this case as you require connection to physical I/O you need to create signals.

> **Note:** Variables are used when you do not require connection to the external environment (e.g. an internal initialisation variable). Signals are used when an external connection is required, and in this case you want to connect to inputs and outputs on the safety PLC, therefore you must create signals.

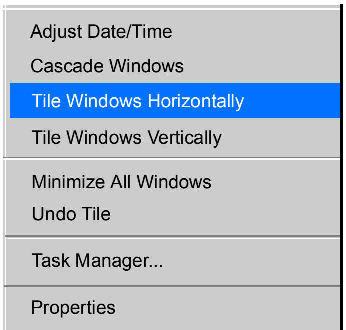
For details on variables and signals please refer to *Signals, p. 163*.

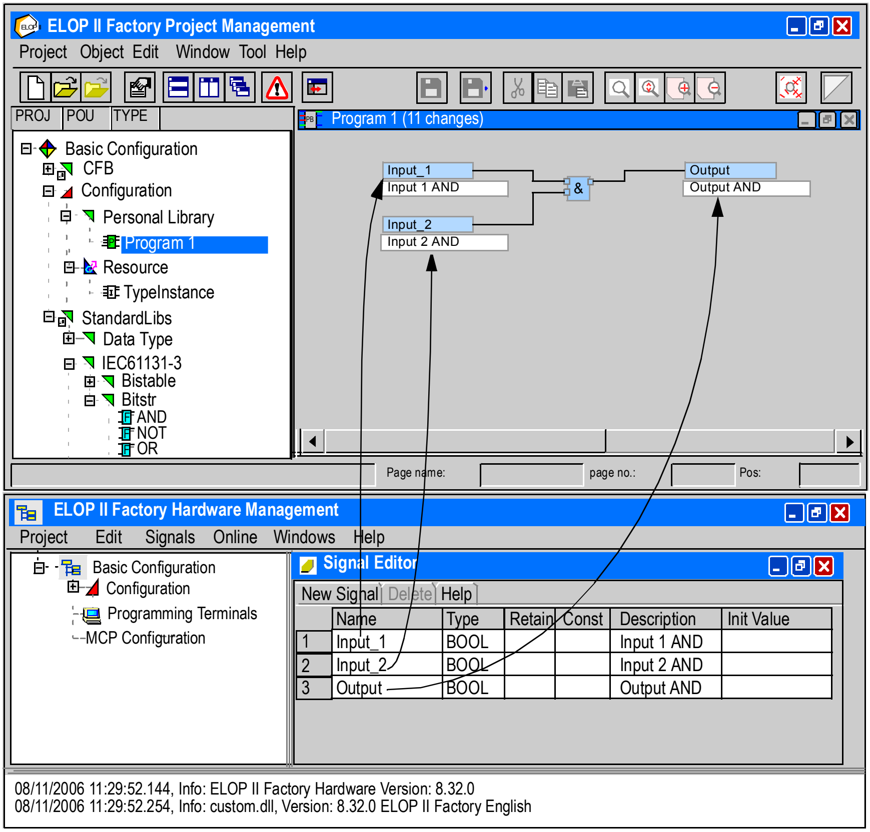
Creating Signals

The creation of signals is done in the **Hardware Management** window.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>In the menu bar of the Hardware Management click on Signals → Editor.

</td></tr>
<tr><td>2</td><td>Click on New Signal and create 3 signals.

</td></tr>
<tr><td>3</td><td>Rename the signals and create a description for each.</td></tr>
</table>

Connecting Signals to Logical Program

Step	Action
1	Minimize all programs which are running except the **Hardware Management** and **Project Management** window.
2	On the Windows task bar select **Tile Windows Horizontally**. 
3	Select the signal from the **Signal Editor** in the **Hardware Management** and drag the signal to a free space in the **Project Management** window.

Step	Action
4	Repeat until all 3 signals are in the **Project Management**. **Result:** 
5	Minimize the **Hardware Management** and maximize the **Project Management** window.
6	To connect the signals to the AND gate: Left-click on a node of the AND function block, press and hold the left mouse button and drag to a signal.
7	The program is now complete, use the save button to save the program, and close the program after saving.

TypeInstance

The **Resource** contains a **TypeInstance**, which can be used to create a user program.

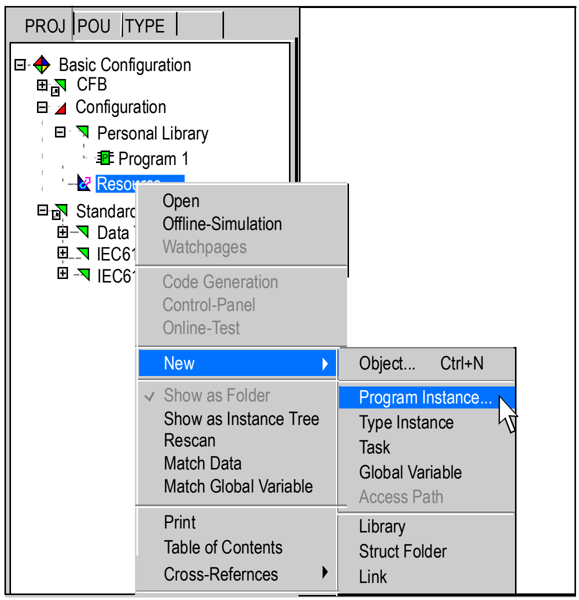
Type instances are programmed directly on the resource. They work in the same way a program works but can only be associated with a single resource.

In this case use your program which you created and not the type instance.

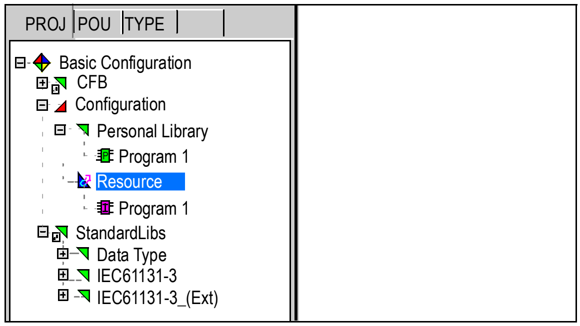
Creating Program Connection to a Resource

The following steps show the procedure to delete the type instance and assign the program.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Delete the default type instance from the resource by clicking on TypeInstance and press DEL (delete).</td></tr>
<tr><td>2</td><td>Right-click on Resource and select New → Program Instance... to attach your configuration from the personal library folder.



</td></tr>
<tr><td>3</td><td>From Personal Library select Program 1.</td></tr>
</table>

Step	Action
4	When the program instance is connected a purple **I** can be seen attached to the **Resource**.  Now the logical program is connected to the resource (however the hardware type is not defined).

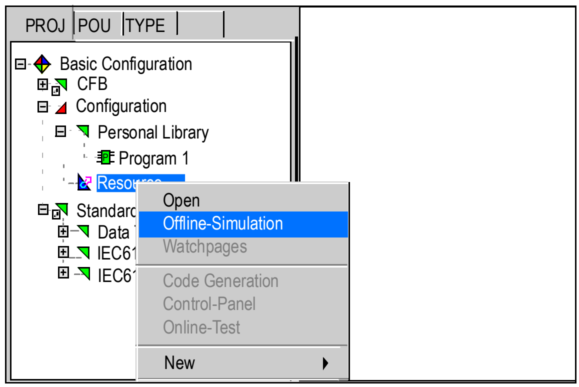
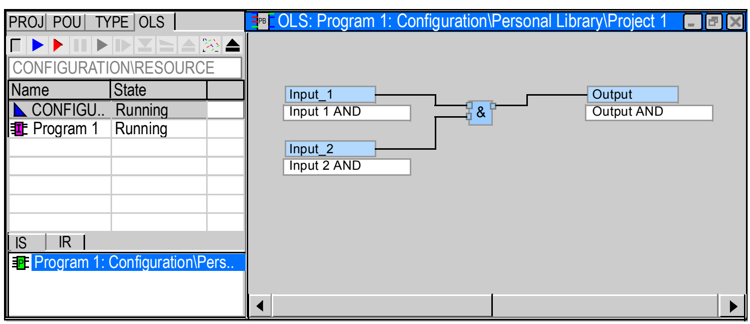
Step 4: Offline Simulation of Program

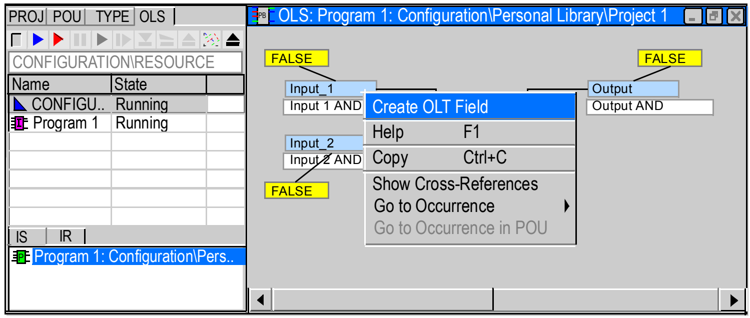
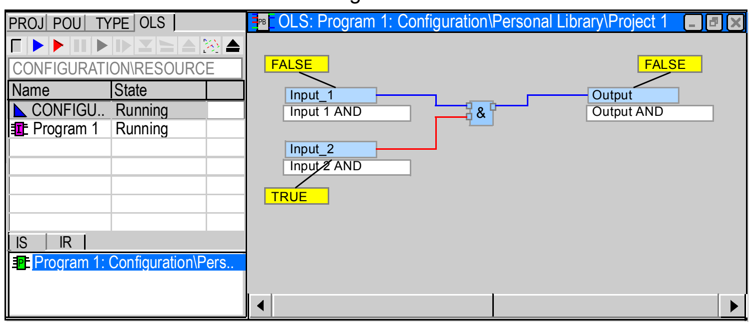
Overview

To check if the logic within your program is functioning in the correct way XPSMFWIN contains an **Offline-Simulation** mode.

This simulation allows you to check the logic without the need for connecting hardware.

Offline Simulation

Step	Action
1	In the **Project Management**, right-click on the **Resource** and select **Offline-Simulation**. 
2	Double-click on the **Program 1** in the **Offline-Simulation** window. **Result:** 

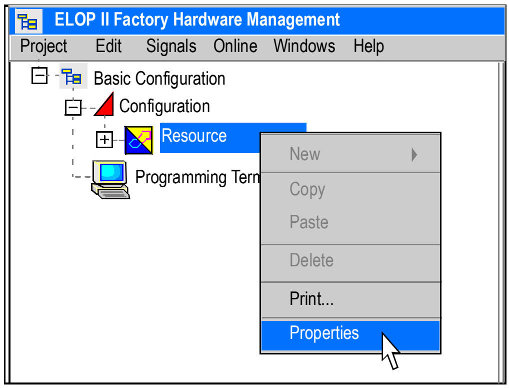
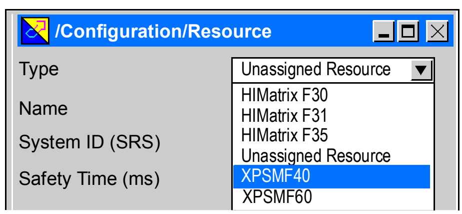
Step	Action
3	Within the drawing field right-click on a signal and select **Create OLT Field** from the context menu. **Note:** Online test fields are used to view and change the status of signals. 
4	Click the cold start button from the left tool bar.
5	Double-click on an OLT field to change the value from FALSE to TRUE.  **Result:** The connector changes from blue to red.
6	Click the close button to exit the offline simulation.

Step 5: Defining Hardware and Settings

Overview

The **Hardware Management** is used to define all hardware related settings. Such as hardware type, communication settings, I/O assignment, etc.

Defining Hardware Type

Step	Action
1	In the **Hardware Management** window open the **Configuration** folder from the structure tree
2	Right-click on **Resource** and select **Properties** from the context menu. 
3	Select the PLC type from the **Type** list box. 
4	Enter the **System ID [SRS]** value 2 to 65,535. **Note**: For further information on system ID (SRS = system-rack-slot) please refer to *SRS (System Rack Slot) Value, p. 156*.
5	Click **Apply**. **Note:** Clicking **Apply** allows all the menu´s sub-options to be selected.
6	For this program example, set the values as listed in the table below. **Note**: For further information please refer to *Setting the Properties of a Resource, p. 154*.
7	Click **OK**.

Property Values

Properties	Value
Safety time	30 ms
Watchdog time	15 ms
Main enable	selected
Auto start	selected
Start/restart allowed	selected
Loading allowed	selected
Test mode	selected
Change variables in OLT allowed	selected
Forcing allowed	selected
Stop on force timeout	selected
Max. com time slice	10 ms

Step 6: Setting PC Communication

Overview

To connect up to the safety PLC systems a common network must be established (either point to point or complete network).

To program and go online with each PLC your PC must be on the same network, or be able to access the network via a gateway or router.

The safety PLCs all have a set MAC address that is used for initial addressing therefore allowing you to set the IP address on any Ethernet network.

Before connection to the PLCs is established the programming terminals IP address must be defined such that it contains the same IP address range as the PLCs.

For further information please refer to *Communication Settings, p. 179*.

Setting the IP Address

Set the IP address of the programming terminal (PC)

Step	Action
1	Open the control panel of the PC with **Start** → **Settings** → **Control Panel**.
2	Double-click **Network and Dial-Up Connections** (Windows 2000).
3	Open the properties for the **TCP/IP protocol**.
4	Select **Use the following address** and enter **192.168.000.254** for **IP address**.
5	Enter **255.255.252.0** for **Subnet mask**.
6	Leave **Default Gateway** area blank.
7	Disable and enable the connection.

Step 7: Setting Hardware I/O

Overview

All the compact safety PLC systems and safety remote I/O modules listed below, have fixed I/O's, which are predefined when the PLC or remote I/O is assigned.

For modular safety PLC (XPSMF60), I/Os must be defined, depending on which module cards are used (see *Adding I/O Modules to the Modular Resource (XPSMF60 Only), p. 311*).

Compact Safety PLCs

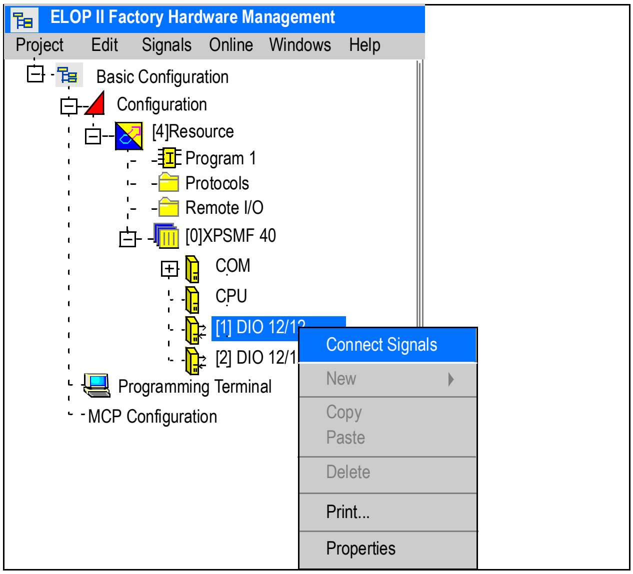
- XPSMF3022
- XPSMF31222
- XPSMF3502/F3522/F3542
- XPSMF4000/F4002/F4020/F4022/F4040/F4042

Safety Remote I/O Modules

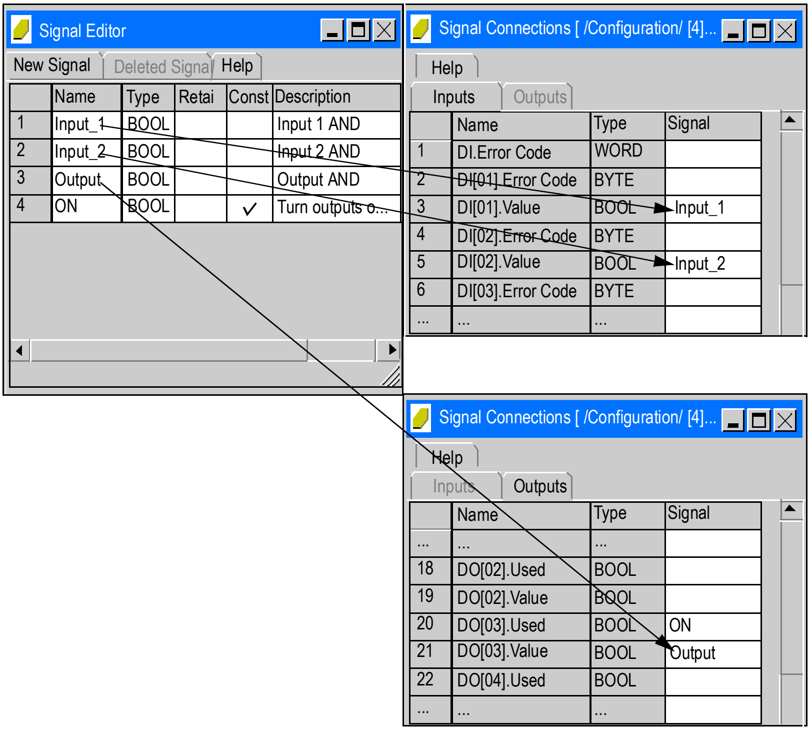
- XPSMF1DI1601
- XPSMF2DO801/DO802/DO1601/DO1602
- XPSMF3DIO8801/DIO16801/DIO20802
- XPSMF3AIO8401

Compact Safety PLCs

Configuring the I/O on the compact range of safety PLCs.

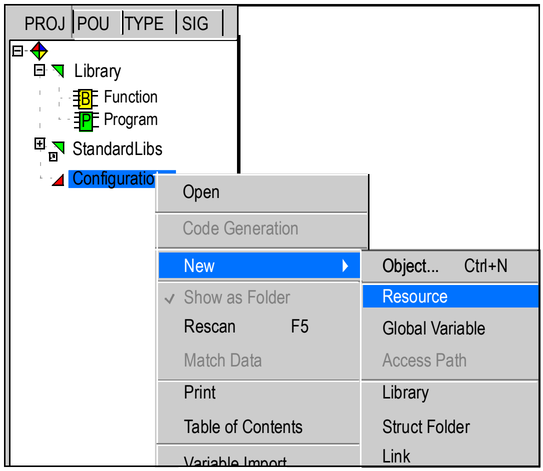
Step	Action
1	In the **Hardware Management** window, open the folder containing the **Resource** and then the product type (**XPSMF 40**).
2	Right-click on **[1] DIO 12/12** and select **Connect Signals** from the context menu.  **Note:** The **Signal Connections** window contains two tabs one for **Inputs** and one for **Outputs**.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>To connect the signals which you have created to the I/O of the PLC, open the Signal Editor from the Signals menu and arrange the Signal Connections window and the Signal Editor side by side.


</td></tr>
<tr><td>4</td><td>Drag the first 2 signals (Input_1 and Input_2) from the Signal Editor into the DI[01].Value and DI[02].Value signal column of the Signal Connections window.
Note: The Signals can be placed on any of DI(XX). They do not need to be assigned to consecutive digital inputs.</td></tr>
<tr><td>5</td><td>Select the Output tab from the Signal Connections window.</td></tr>
<tr><td>6</td><td>Drag the Output signal to DO[03].Value.
Note: To activate an output on the XPSMF40 safety PLC, you must set the output to ON as the PLC´s I/O can be configured as either inputs or outputs. As a default and for safety reasons the I/Os are always set to inputs.</td></tr>
<tr><td>7</td><td>To turn an output on, create a new signal within the Signal Editor with the Name ON, select the Const. attribute and give it an initial value of 1.</td></tr>
<tr><td>8</td><td>Drag this ON signal to DO[03].Used.
Note: Defining the output as ON is only used for the XPSMF40 safety PLC as the DIO are configurable as inputs or outputs.</td></tr>
</table>

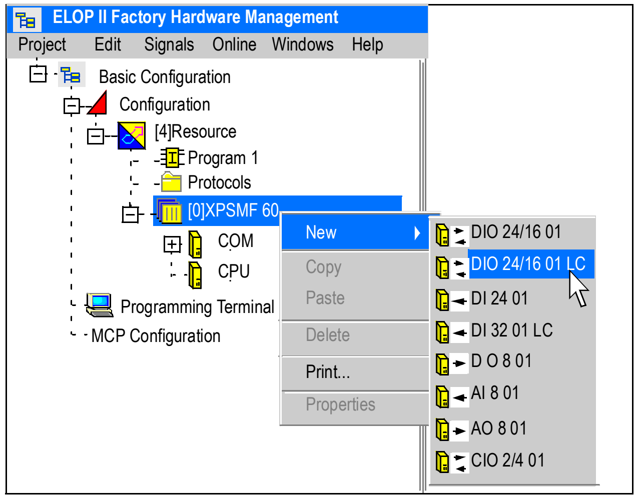
Define the Modular PLC as the Resource

If you want to carry out the following procedure *Adding I/O Modules to the Modular Resource (XPSMF60 Only), p. 311* but have not defined the Modular PLC as the resource then:

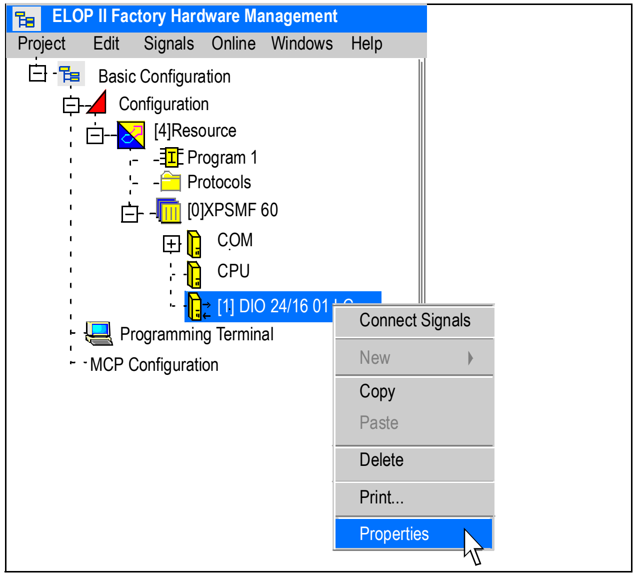
Step	Action
1	Open the **Project Management**.
2	With the right mouse button click on the **Configuration** in the **Project Management** and open the context menu.
3	Click on **New** → **Resource**. 
4	Rename the resource to `Modular PLC` (in the project management click twice slowly on the name of the resource in the structure window. **Result**: An entry field opens and the name can be changed.
5	Delete the **Typeinstance** from the new resource **Modular PLC** and assign the program **Program 1** from the personal library by right-clicking on **Modular PLC** → **New** → **Program instance**, and select **Program 1** from the list. See *Assign Program Type to Resource, p. 109* of the project management for more information regarding attaching programs to resources.
6	Within the **Hardware Management** window right-click on the **Modular PLC** resource from the configuration folder and select **Properties**.
7	Assign the XPSM F60 reference from the **Type** and enter a unique SRS value. Enter the same properties that were set for the first resource (see *Property Values, p. 305*).
8	Now follow the procedure outlined in the following table (Adding IO modules to the Modular Resource).

Adding I/O Modules to the Modular Resource (XPSMF60 Only)

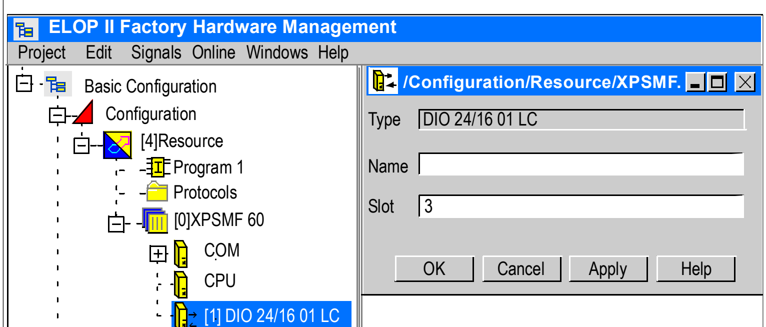
If you have defined a modular XPSMF60 PLC within your configuration follow this procedure:

Step	Action
1	Open the structure by clicking the **+** icon in front of the **Resource**.
2	Right-click on **XPSMF 60** and select **New** and the respective module from the context menu. 

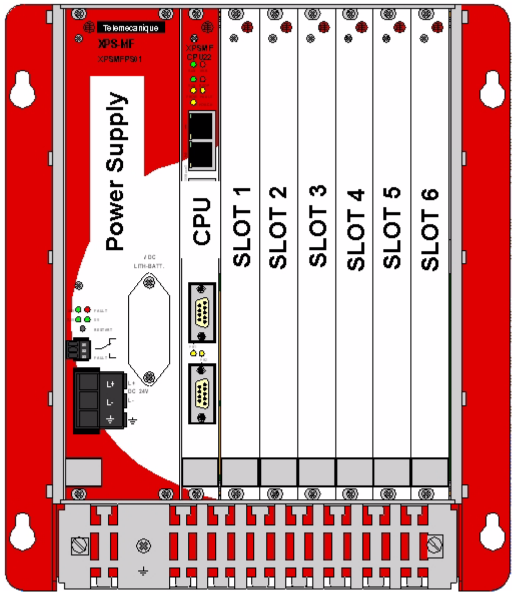
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>Right-click on the input/output module (just added) and select Properties from the context menu.



</td></tr>
<tr><td>4</td><td>Enter the correct slot number.



</td></tr>
<tr><td>5</td><td>Complete the input by pressing OK.
Note: A newly added module always has slot number 1 (within the software). 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 fixed.</td></tr>
</table>

Step	Action
6	Repeat this procedure until all I/O cards are assigned to the rack. 

Modular Safety PLC (XPSMF60)

The procedure for configuring the I/O on the modular range of safety PLCs (XPSMF60) is the same as described for the compact safety PLCs above (see *Compact Safety PLCs, p. 308*).

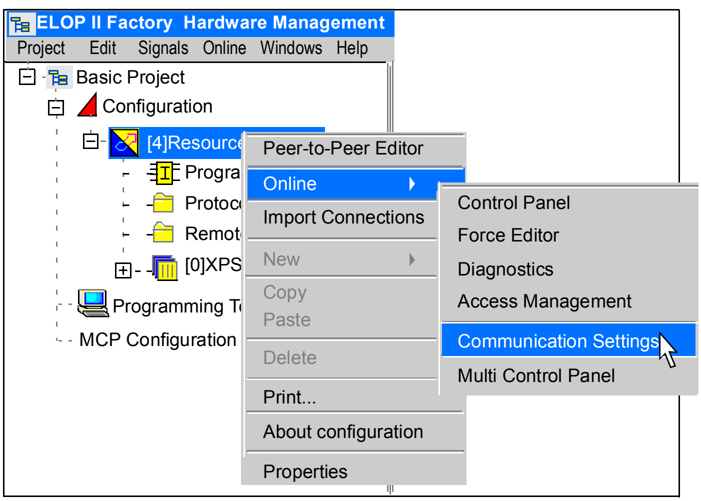
Step 8: Setting PLC Communication

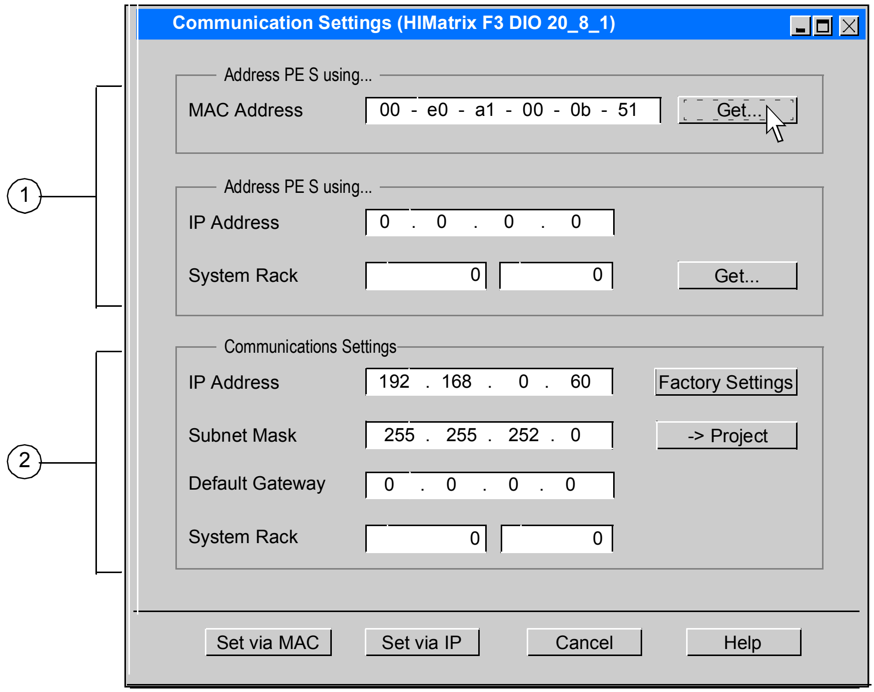
PLC Connection

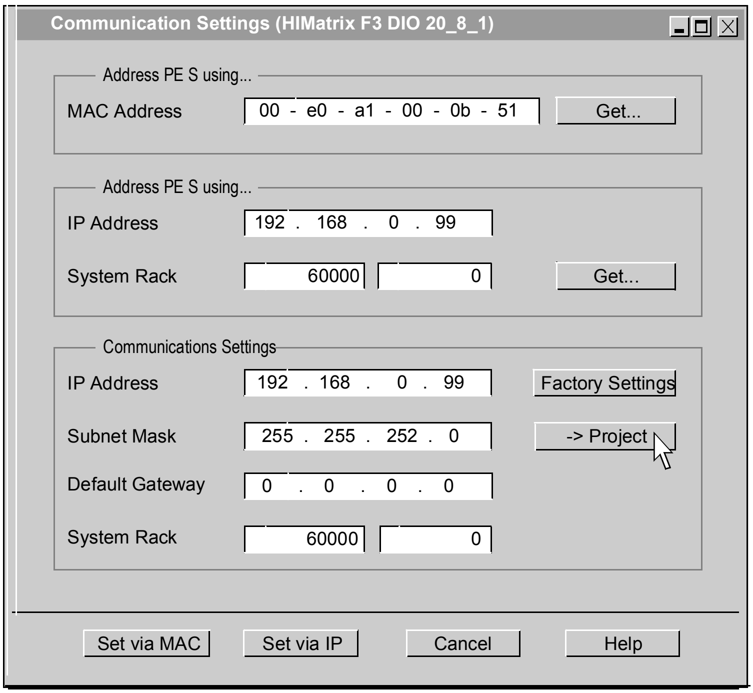
The following description is valid for resources (PLCs) and remote I/O.

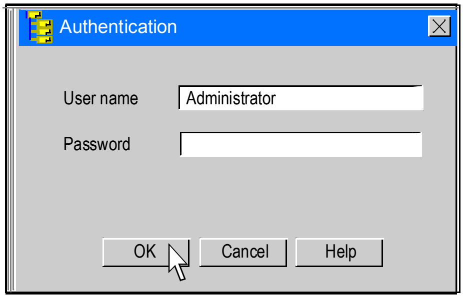
Step	Action
1	Connect the PLC to a sufficiently sized power supply. Do not make connections to the Ethernet or to input/output channels.
2	After 30s check if the PLC is in RUN mode (RUN-LED is continuously on).
3	Depending on whether the PLC is in RUN or STOP mode after 30s, there are two ways to proceed: ● See *Setting Parameters (PLC in STOP Mode), p. 314*. ● See *Setting Parameters (PLC in RUN Mode), p. 318*.

Setting Parameters (PLC in STOP Mode)

Step	Action
1	Connect the network card of the programming terminal to one of the Ethernet ports on the PLC.
2	Select **Online** → **Communication Settings** in the **Resource** context menu. 

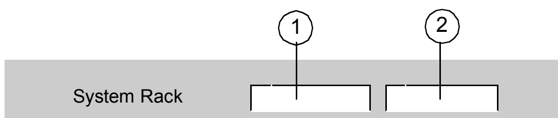
Step	Action
3	Enter the **MAC Address** of the PLC in the **Communication Settings** window. You will find the MAC address on the bottom side of the PLC near the Ethernet connectors or for the XPSMF60 (XPSMFCPU22) directly on the CPU module.  **1** Addressing area of PLC or Remote I/O **2** Area for Assigning IP Address of the PLC or Remote I/O

Step	Action
4	Click on the **Get** button. **Result**: The **Communication Settings** from PLC or remote I/O are displayed. ● IP address and subnet mask ● system rack values (SRS), defined in the properties of the PLC or remote I/O 
5	Enter the required **IP Address**/**Subnet Mask**/**Default Gateway** and **System Rack** within the **Communication Settings** area of the window (see *System Rack (SRS) Definition, p. 317*).
6	Click on the -> Set via MAC to save the communication settings to the PLC

Step	Action
7	Set the new IP address to the PLC by entering the default setting **Administrator** without a password.  **Note**: See the message in the Error-State-Viewer for successful data change. After setting the parameters for all resources and remote I/Os you can download your application into the resources. **Note:** You must code generate after setting the IP configuration.
8	Click on the **Project** button and set the communication properties to the Resource or Remote I/O Module.

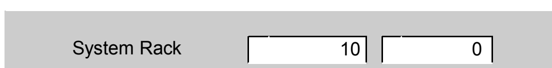
System Rack (SRS) Definition

For the **System Rack** (SRS) the definition is as follows:

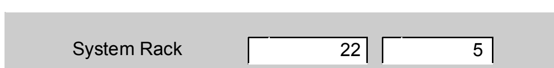


1 SRS of PLC/Resource
2 SRS of Remote I/O

Example 1: When assigning the **System Rack** of a resource with SRS = 10, set the following value:



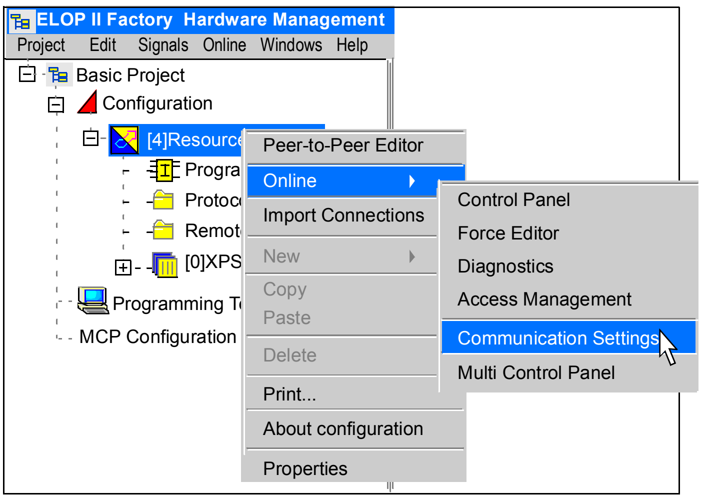
Example 2: When assigning a remote I/O with parent resource (PLC) SRS = 22 and Remote I/O SRS = 5, set the following value:

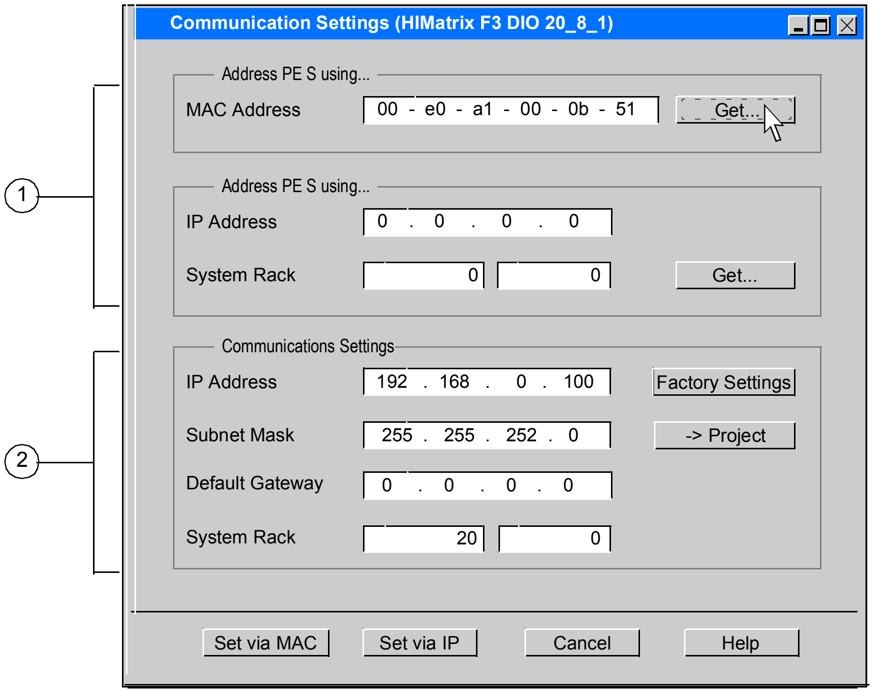


Note: See *SRS (System Rack Slot) Value, p. 156* for more information.

Setting Parameters (PLC in RUN Mode)

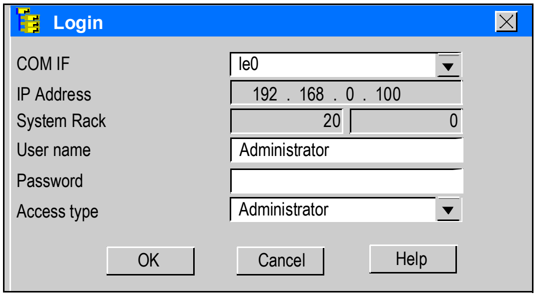
In Run mode for Safety Reasons it is impossible to change the IP address settings of the PLC. therefore in order to download or change the IP address settings of the safety PLC the resource (PLC) must be in STOP mode.

Step	Action
1	Create a new **Dummy** resource within the project management window: Right-click on the configuration folder and **Select** → **New** → **Resource** **Note**: there is no need to rename the **Dummy** resource as the resource is only temporary.
2	Within the **Hardware Management** window: Right-click on the **Dummy** resource which was just created and select the same resource type as the hardware from the properties window, and enter any SRS value between 2 and 65500. **Note**: the SRS is in this case irrelevant as it will change in the next steps.
3	Follow the same procedure outline in Steps 1 and 2 of the PLC connection (*Step 8: Setting PLC Communication, p. 314*).Connect the PLC (1 to 1) to the programming terminal(PC) with Ethernet cable.
4	Select **Online** → **Communication Settings** on the **New Resource** Dummy resource. 

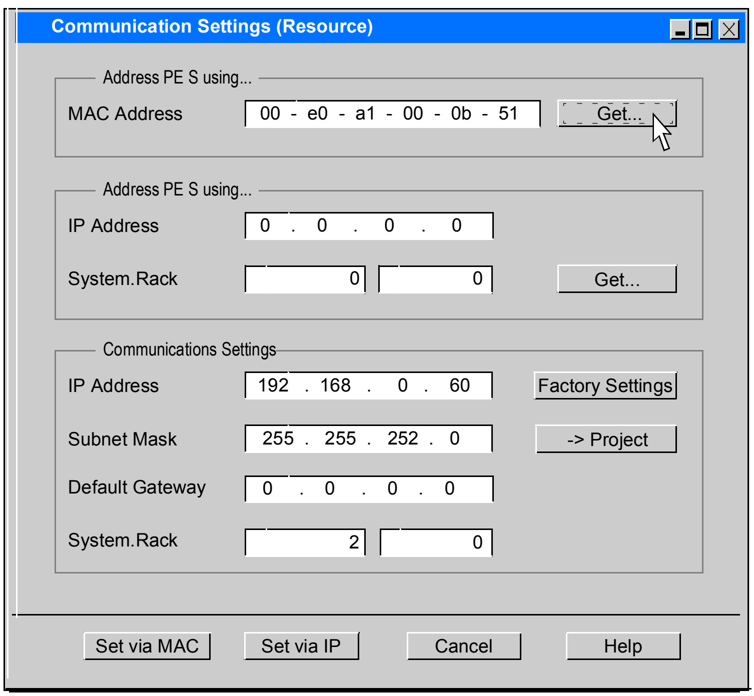
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>5</td><td>Enter the MAC address of the PLC within the communication settings window, and select Get.


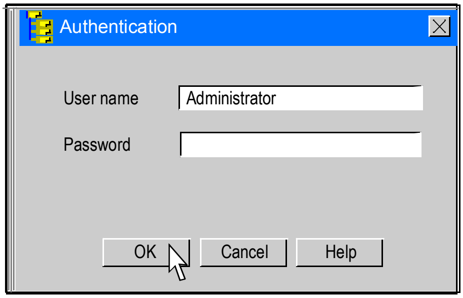
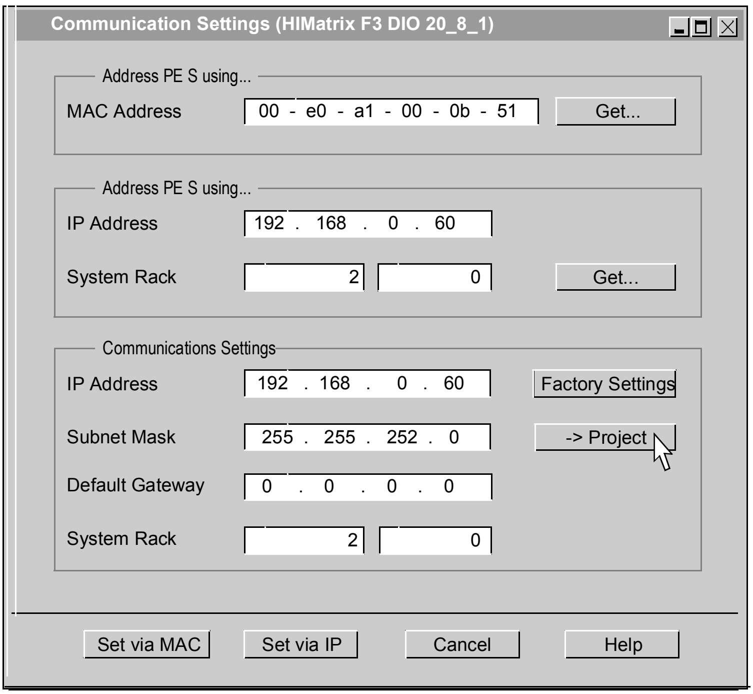
1 Addressing area of PLC or Remote I/O
2 Area for Assigning IP Address of the PLC or Remote I/O</td></tr>
<tr><td>6</td><td>Click on the Project button and select the Newresource to apply the communication settings to the project
Note: You are setting the current PLC IP address and SRS settings to the Dummy resource.</td></tr>
<tr><td>7</td><td>Ensure you PC IP address is within the same network range of the PLC. If not, adjust your PCs' communication settings (see Step 6: Setting PC Communication, p. 306).</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>8</td><td>Right-click on the Newresource (Hardware Management) and select Online → Control Panel. Select Ctrl A to enter the default user name (Administrator), Password: none, and access rights: Administrator
Log in as Administrator for User name and Access type (shortcut: CTRL+A).



Note: When the IP address is within the same range as the PLC, connection will occur.</td></tr>
<tr><td>9</td><td>Select Stop on the PLC and close the online control panel.</td></tr>
<tr><td>10</td><td>Reopen the communication settings window of Resource (original PLC in configuration) and reenter the MAC address and Select Get
Enter the required IP address and SRS of the Resource.


</td></tr>
</table>

Step	Action
11	Select **Set via MAC** and enter **Administrator** within the username field, with no password. 
12	Select **Project** and select the **Resource**. 
13	Return to the project management window and now delete the **New Resource** from the configuration

Step 9: Generating Code

Overview

After your project is completed, you can generate code for the entire project.

For safety-related applications the code generator must be carried out twice and must be downloaded two times.

The checksums (CRCs) of the two generated code versions must be compared.

The code is guaranteed to be error-free only if the checksums are identical.

Note: Downloading will be carried out in Step 10 of this basic configuration guide

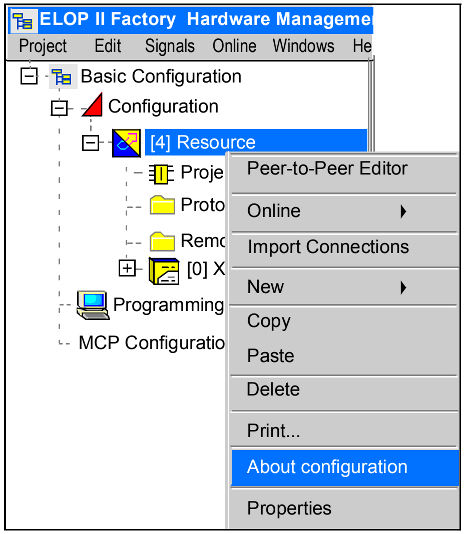
Generating Code (First Time)

Step	Action
1	In the **Project Management**, right-click on the **Configuration** and select **Code Generation** → **Start**.
2	Click the error state viewer  icon to view the code generation details.
3	Analyze and resolve any error.
4	Review any warning message to ensure that a dangerous event can not occur from the warning message.

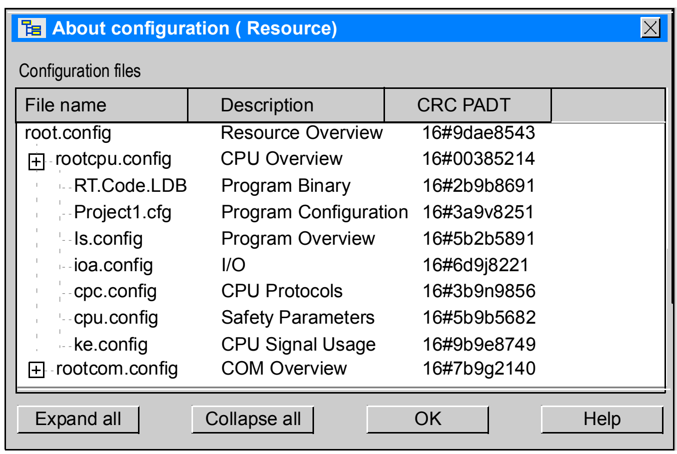
Checksum (First Time)

If code generation was successful, write down the generated checksums.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Within the hardware management window right click on the resource and select About Configuration



</td></tr>
<tr><td>2</td><td>Print the About Configuration using the print screen key of your keybord.



</td></tr>
</table>

Generating Code (Second Time)

Generate code a second time as described above.

Checksum (Second Time)

When CRC values are the same for both code generation versions, the program has been compiled correctly.

Printouts

Open the **About Configuration** as outlined above and print the **CRC** information.

Compare the two **About configuration** printouts, ensuring that the **CRC PADT** values are the same on both code generations.

These two sheets should be signed, and dated and kept with the final project documentation on completion of programming.

Program Size

Step	Action
1	In the **Hardware Management**, right-click on **Program 1** and select **Properties** from the context menu.
2	Within the properties dialog box check the **Data Size** and **Code Size** values (bytes).

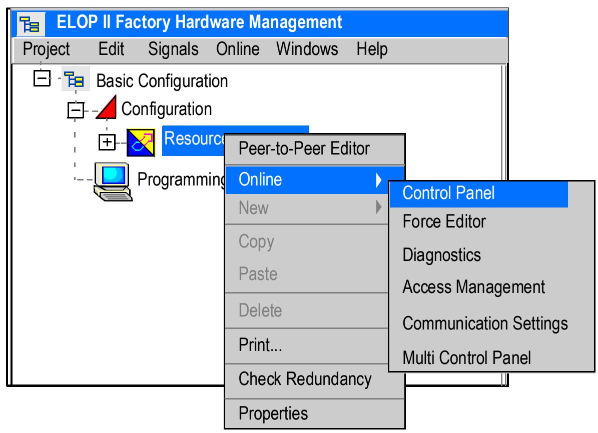
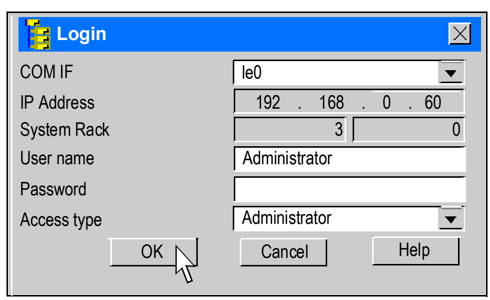
3	Ensure that the program does not exceed the memory size of the safety PLC.
	● Compact safety PLC range
	  ● Data size max: 250 kB
	  ● Code size max: 250 kB
	● Modular safety PLC range
	  ● Data size max: 500 kB
	  ● Code size max: 500 kB

Step 10: Online, Download, Run

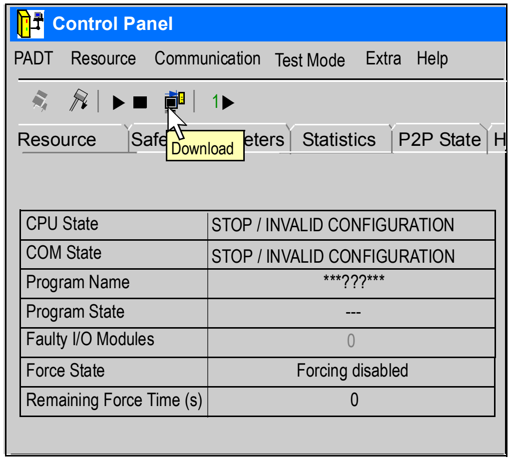
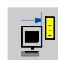
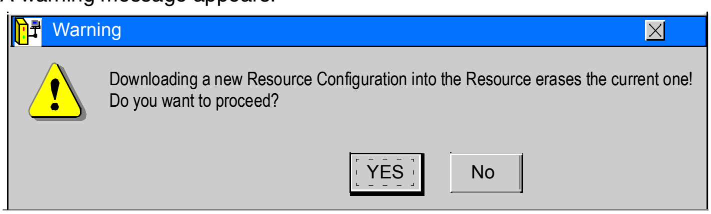
Overview

Before the configuration can be loaded into the resource, the code must have been generated and the programming terminal and the resource must have valid communication settings.

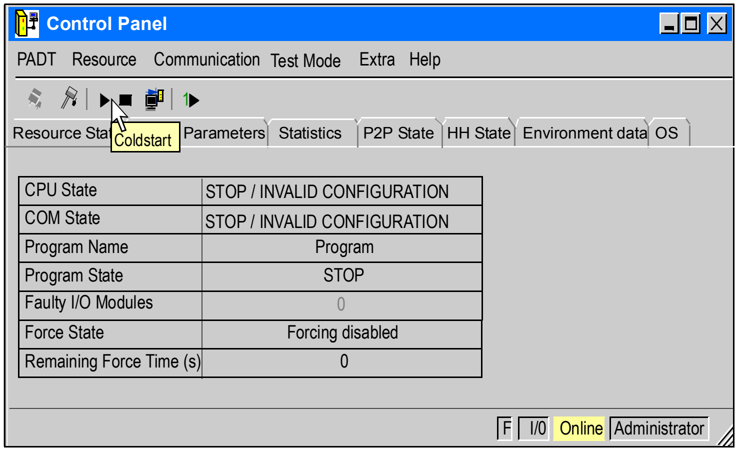
Opening the Control Panel

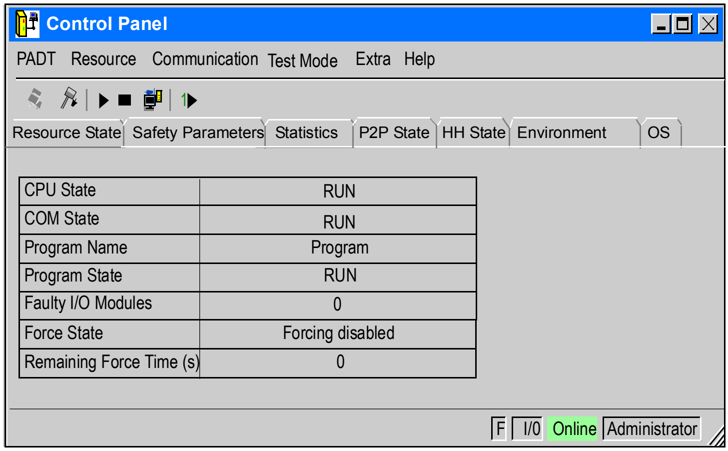
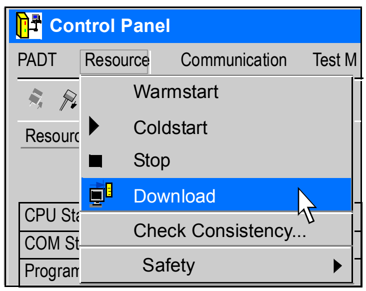
Step	Action
1	In the **Hardware Management** window open the **Configuration** folder from the structure tree.
2	Select **Resource** → **Online** → **Control Panel**. 
3	Log in as **Administrator** for **User name** and **Access type** (shortcut: CTRL+A). 

Loading Resource Configuration

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>To download a configuration the PLC must be in STOP mode. When in run mode Resource → Stop.

</td></tr>
<tr><td>2</td><td>Click the download  button.
A warning message appears.

</td></tr>
<tr><td>3</td><td>Confirm with Yes.
Result: The download starts.</td></tr>
<tr><td>4</td><td>Select from the online control panel Resource → Check Consistency and check and print the CRC data stored within the PLC and PADT (PC) are equal. In addition, compare these values with the values printed out from the About Configuration menu of the Resource.</td></tr>
<tr><td>5</td><td>Download once more the program and verify that the CRC information is correct.</td></tr>
</table>

Starting User Program

Step	Action
1	Click the coldstart  button. 

Step	Action
2	After the coldstart **CPU State**, **COM State** and **Program State** change to RUN.  **Note**: The **Start**, **Stop** and **Download** functions can also be run from the **Resource** menu. 

Step 11: Online Test with Hardware

Overview

The online test allows you to view in a near real time manner the states of all inputs and outputs in the program.

The online test is very similar to the offline simulation except that the hardware is connected and providing real values to the online test.

The online test is very useful for identifying possible errors in coding.

Opening the Online Test

Step	Action
1	In the **Project Management** right-click on the **Resource** and select **ON-LINE Test** from the context menu. **Note:** The PLC must be connected and online with the **Online Control Panel.**
2	Double-click on the program in the **OLT** window to open the logic. **Note:** Values are displayed in online test fields (OLT field). In the case of boolean values the connection line has a color 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.

Note: See *Procedure, p. 100* for more details of the Online Test.

Concluding the Basic Project Configuration

Overview

Congratulations, you have concluded the basic project configuration.

To ensure that all features from this simple configuration were understood, close the project, create a new project and carry out all of the steps a second time.

Advanced Project Configuration

8

At a Glance

Overview

This chapter provides information on the configuration of a advanced project.

What's in this Chapter?

This chapter contains the following sections:

8.1 Introduction

Introduction

Overview

This section is provided to guide you through the programming environment by means of creating a complex program.

10 Steps

The configuration is composed of 9 simple and easy to follow steps.

Creating a advanced project includes the following steps:

Step	Action	Time to Complete
1	Creating a Project and Programs Including Offline Simulation	30 minutes
2	Defining Hardware and Remote I/O Settings and Peer-to-Peer Communication	10 minutes
3	Creating Non-Safety Protocol Communication	5 minutes
4	Setting Hardware I/O Digital Including Line Control Signal Assignment, Communication Setup	15 minutes
5	Access Administration	10 minutes
6	Code Generation and Program Size	2 minutes
7	Online, Download, Run and Online Test	5 minutes
8	Force Editor	10 minutes
9	Hot Swapping Remote I/O Requirements	-

8.2 Step 1: Creating a Project and Programs Including Offline Simulation

At a Glance

Overview

This section is provided to assist you in creating complex programs, and to assist you in optimizing your knowledge with using the XPSMFWIN programming environment.

What's in this Section?

This section contains the following topics:

Section Overview and Initial Software Start Up

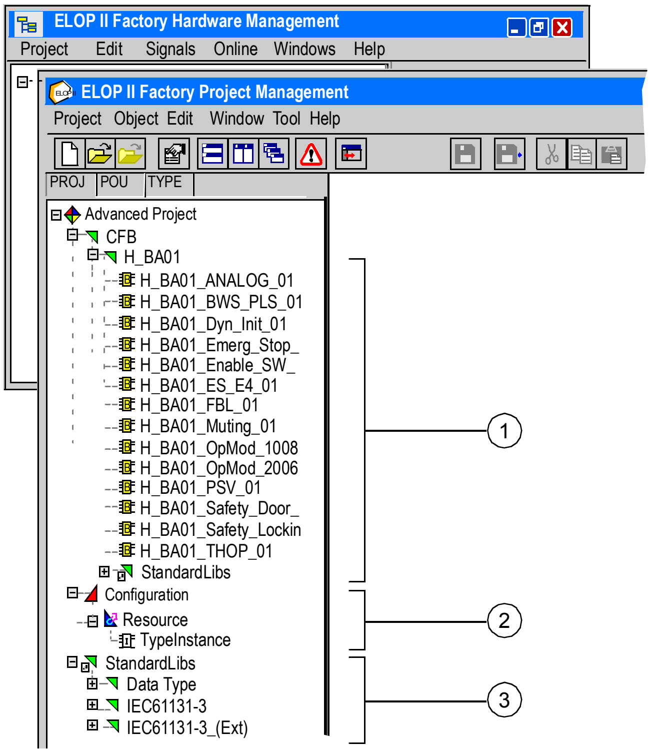
Creating a Project

Create a new project using the **Project Wizard** as described in the *Basic Project Configuration* section (see *Step 1: Creating a New Project Using the Project Wizard, p. 289*).

Enter for the project name: `Advanced project`

Hardware and Project Management Window

At this stage the **Hardware Management** and **Project Management** windows should be open with **Advanced Project** as the project name.



Item	Description
1	certified function block (CFB) library
2	area for creating resources (PLCs) and assigning programs to each default: resource assigned with type instance **Note:** This area is hardware independent.
3	logical libraries (e.g. logical gates (AND, OR), timers, numeric functions, etc.

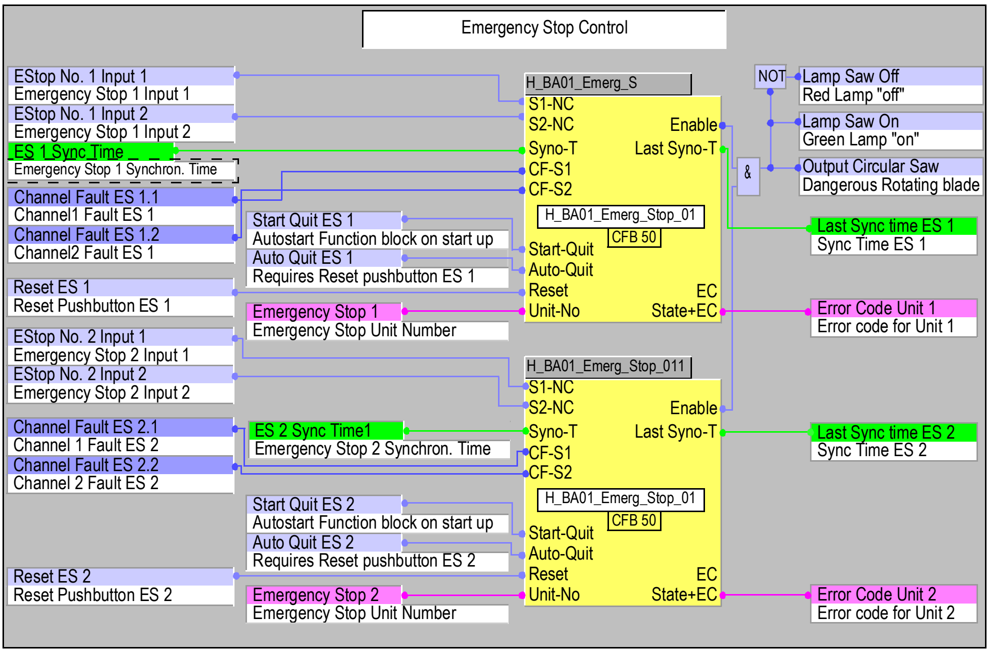
Creating Programs

Within this step you will learn the 2 available areas where you can create programs.

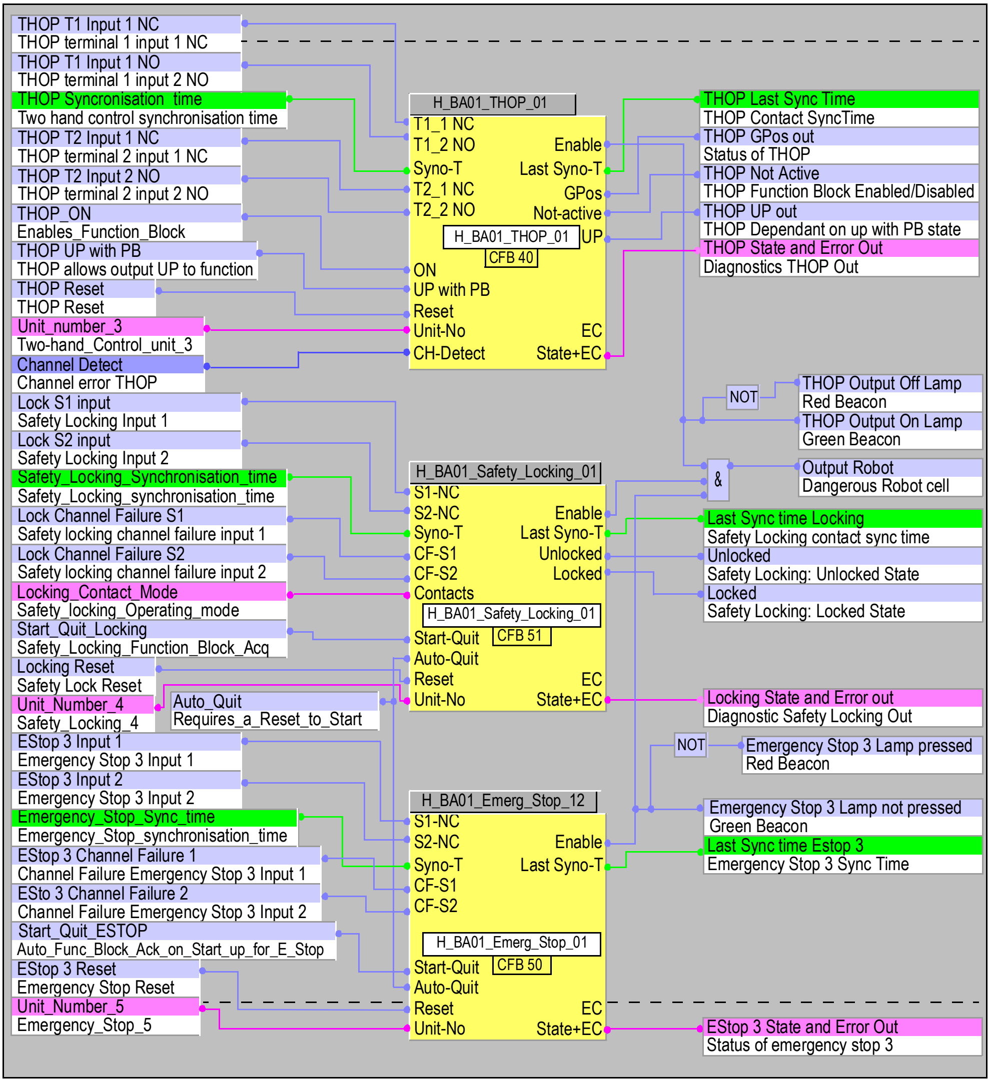
To understand this you will utilize certified function blocks (CFBs) and create personal function blocks to create a configuration for multiple resources.

First Program

The following diagram is the first program, which you will create. It is an emergency stop control for 2 emergency stop push buttons (Cat 4, SIL 3). The output is set to safe state when 1 emergency stop or both are pressed.

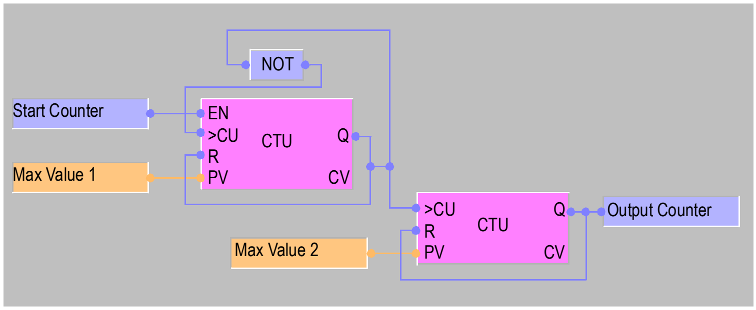


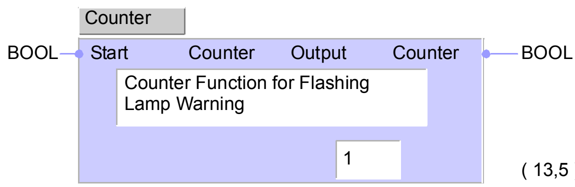
Second Program The following diagram is the second program, which you will create for monitoring emergency stop functionality, safety locking and 2-hand control operation.



Creating a Timer Function Block

The following diagram is the function block creation for creating a timer, which counts upwards, which will be used to create a flashing LED Lamp.



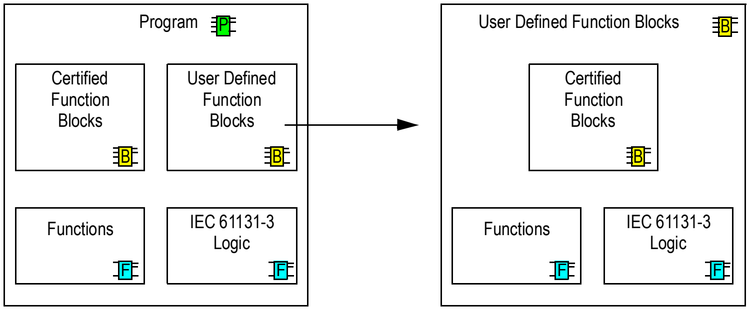


Function Blocks

The **CFB** folder contains all certified function blocks, which enable you to create a project very quickly, and reduces the time required for certification.

These CFBs can be used within a function block or directly within a user program.

The program structure and user defined function block structure are shown below.



The program is the highest order of the set, and can contain:
- certified function blocks
- user defined function blocks
- user defined functions
- IEC 61131-3 logical function blocks

The user defined function blocks can be used within the program and can contain:
- certified function blocks
- user defined functions
- IEC 61131-3 logical function blocks

Note: User defined function blocks can reduce the repetitive programming and helps provide the program with structure.

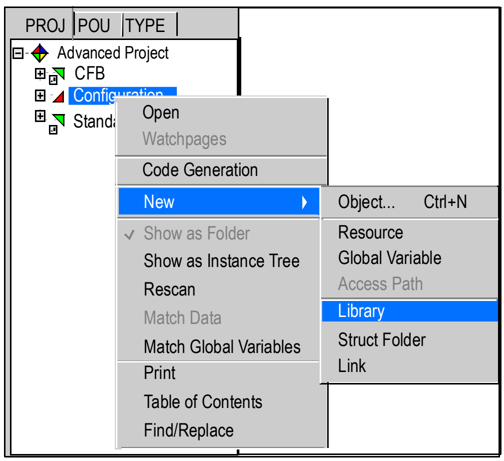
Creating Program 1

Overview

The first program is an emergency stop control for 2 emergency stop push buttons (Cat 4, SIL 3). The output is set to safe state when 1 emergency stop or both are pressed.

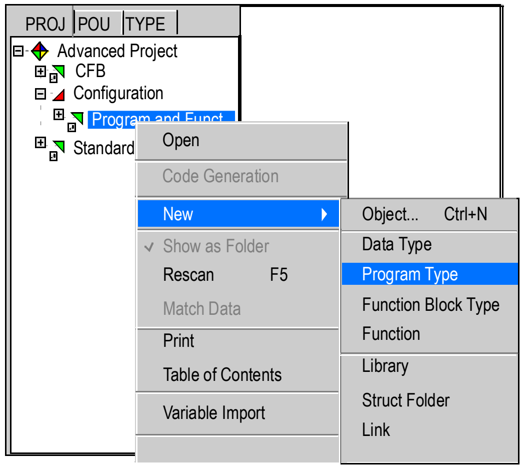
Creating Personal Library

The first step of creating a program is to create a personal library to store user programs and functions.

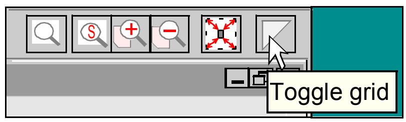
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Close all of the opened areas of the project tree in the Project Management window created in the section before by clicking the ⊟ symbols.
(See Section Overview and Initial Software Start Up, p. 336).</td></tr>
<tr><td>2</td><td>Right click on the Configuration to open the context menu.</td></tr>
<tr><td>3</td><td>Click on New → Library.


Result: A new library is created.</td></tr>
<tr><td>4</td><td>Change the name of the new library from NewLib to Program and Function Blocks.</td></tr>
</table>

Creating a Program Type

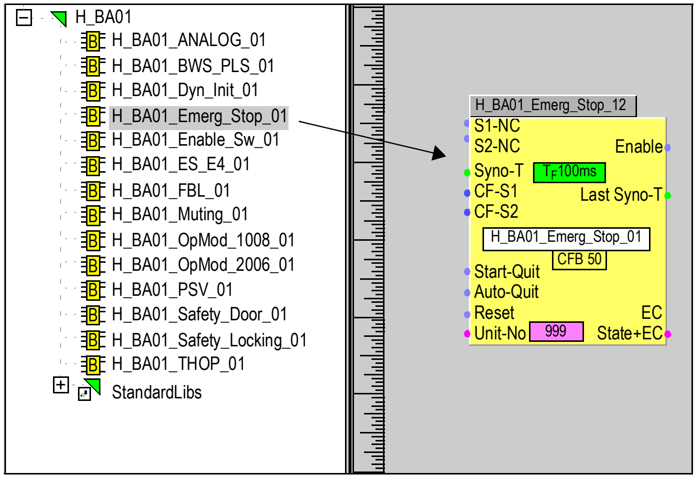
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Right-click on Program and Function Blocks Library to open the context menu.</td></tr>
<tr><td>2</td><td>Select New → Program Type.

</td></tr>
<tr><td>3</td><td>Click on the + in front of Program and Function Blocks.
Result: A new function block NewProgramType  is added to the structure.</td></tr>
<tr><td>4</td><td>Change the name of the new program type from NewProgramType to Program 1.
Note: The program type contains the function logic that is to be executed in the PLC and can contain function blocks, as well as certified function blocks (CFBs).</td></tr>
</table>

Optimizing Working Area

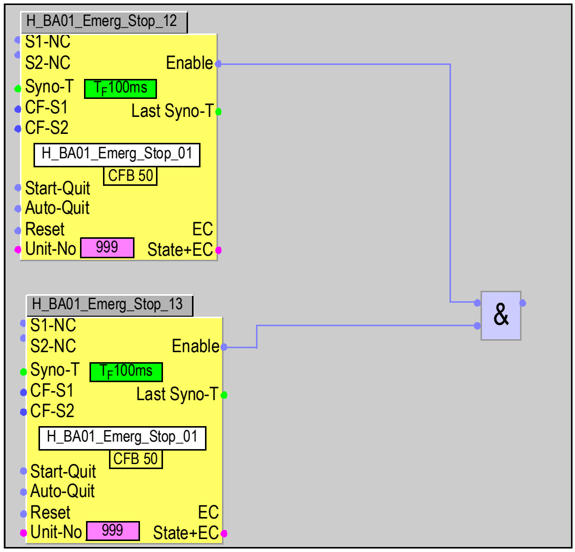
Step	Action
1	Double-click on **Program 1** in your **Program and Function Blocks** folder. **Result:** The function block diagram (FBD) editor is displayed.
2	To optimize the working area hide the **Error-state Viewer** by clicking the button.
3	To further optimize the working area hide the **Variable Declaration Editor** by clicking the button.
4	Toggle the grid.
5	Zoom in to the section that you want to edit.  **Note**: The buttons on the right side refer to the drawing field of the open function block.

Dragging Function Blocks to Drawing Field

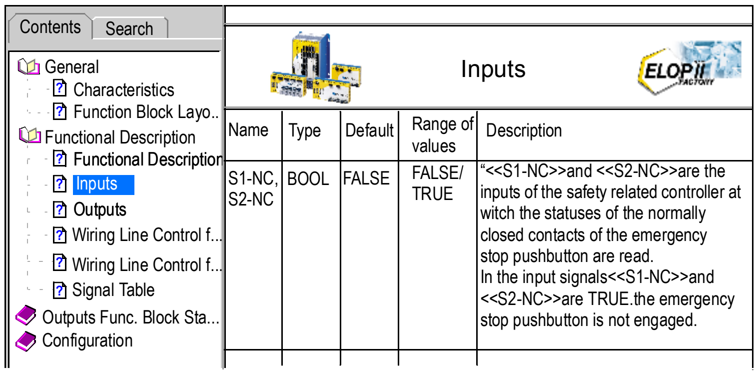
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>From the CFB folder in the structure tree find the certified function block <code>H_BA01_Emerg_Stop_01</code>.</td></tr>
<tr><td>2</td><td>Click on the function block, press and hold the left mouse button and drag the function block from the structure tree into the drawing field.



Note: The function block characteristics (input and output information and wiring information) can be viewed by selecting the F1 function key, either when selecting the function block from the structure tree or in the drawing field.
Result: When placing the certified function block on the page the Edit Page Data window will open (please refer to the Basic Project Configuration section Editing Page Data, p. 296).</td></tr>
<tr><td>3</td><td>Drag a second emergency stop function block into the drawing field.</td></tr>
<tr><td>4</td><td>Drag a <code>AND</code> gate from the IEC 61131-3 Libraries Bitstr folder into the drawing field.</td></tr>
</table>

Step	Action
5	Connect the `Enable` pin from each of the emergency stop function blocks to the each of the inputs of the `AND` gate. 

Information on Inputs and Outputs

Step	Action
1	To open the online help functionality of the function block select the function block and then use the **F1** function key and select **Functional Description** from the **Contents** window.
2	Select the **Inputs** and **Outputs** tab of the functional description folder to display information for the inputs (outputs) of the function block. 

Signals/ Variables

Using the information provided by the online help of a function block (see diagram above), you can define whether to use signals or variables.

- Signals are used when a variable requires connection to the external environment (hardware I/O, Modbus/Profibus transfer etc.).
- Variables should be used when no external interface is required.

Creating Variables

Create the variables listed in the table below in the `VAR` tab of the **Variable Declaration Editor**. (See *Creating a Variable, p. 68*).

Name	Declaration	Initial Value	Longname	Attribute
Auto Quit ES 1	BOOL	0	Requires Reset pushbutton ES 1	C
Auto Quit ES 2	BOOL	0	Requires Reset pushbutton ES 2	C
ES 1 Sync Time	TIME	t#50ms	Emergency Stop 1 Synchron. Time Channel 1 and Channel 2	-
ES 2 Sync Time1	TIME	t#50ms	Emergency Stop 2 Synchron. Time Channel 1 and Channel 2	-
Emergency Stop 1	UINT	1	Emergency Stop Unit Number	C
Emergency Stop 2	UINT	2	Emergency Stop Unit Number	C
Start Quit ES 1	BOOL	0	Autostart Function block on start up	C
Start Quit ES 2	BOOL	0	Autostart Function block on start up	C

Global Variables

Name	Declaration	Initial Value	Longname
Channel Detect	Array [1 to 4] of Byte	0	Channel error THOP

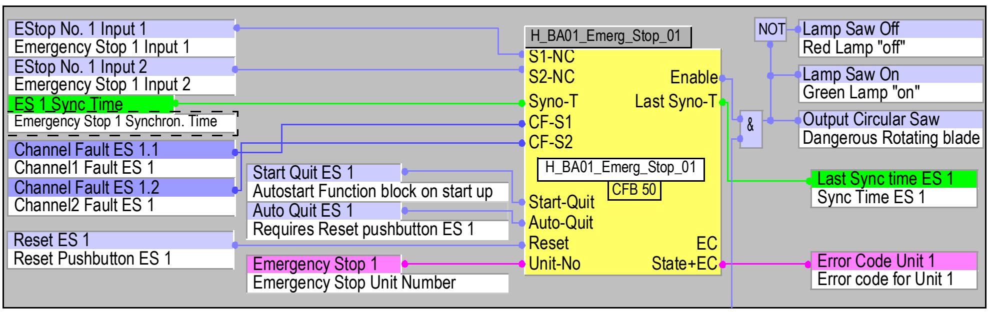
Creating Signals Create the signals listed in the table below in the **Signal Editor**. (See *Creating Signals, p. 297*. You do not need to define whether the signal is an input or output.

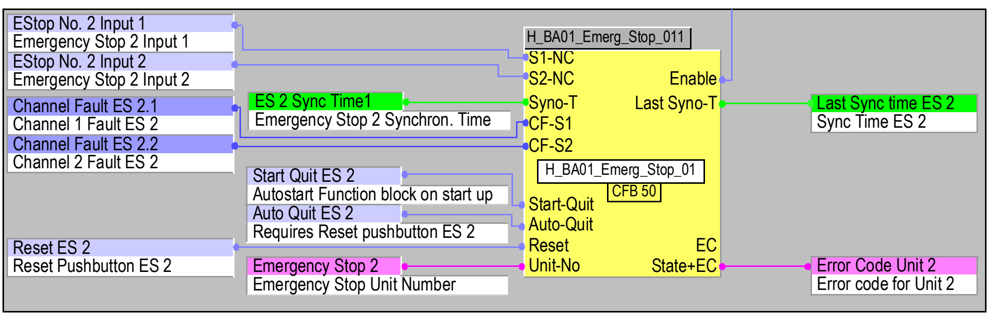
Name	Type	Retain	Constant	Description	Init Value
Output Circular Saw	BOOL	-	-	Dangerous Rotating blade	-
EStop No. 1 Input 1	BOOL	-	-	Emergency Stop 1 Input 1	-
EStop No. 1 Input 2	BOOL	-	-	Emergency Stop 1 Input 2	-
EStop No. 2 Input 1	BOOL	-	-	Emergency Stop 2 Input 1	-
EStop No. 2 Input 2	BOOL	-	-	Emergency Stop 2 Input 2	-
Channel Fault ES 1.1	BYTE	-	-	Channel 1 Fault ES 1	-
Channel Fault ES 1.2	BYTE	-	-	Channel 2 Fault ES 1	-
Channel Fault ES 2.1	BYTE	-	-	Channel 1 Fault ES 2	-
Channel Fault ES 2.2	BYTE	-	-	Channel 2 Fault ES 2	-
Reset ES 1	BOOL	-	-	Reset Pushbutton ES 1	-
Reset ES 2	BOOL	-	-	Reset Pushbutton ES 2	-
Error Code Unit 1	UDINT	-	-	Error code for Unit 1	-
Error Code Unit 2	UDINT	-	-	Error code for Unit 2	-
Lamp Saw On	BOOL	-	-	Green Lamp on	-
Lamp Saw Off	BOOL	-	-	Red Lamp off	-
Last Sync time ES 1	TIME	-	-	Sync Time ES 1	-
Last Sync time ES 2	TIME	-	-	Sync Time ES 2	-

Note: When an initial value is not set in either the signal Editor or variable declaration editor the initial value is taken as **0**.

Signals can be created in the **Signal Editor** in the hardware management or can be imported from external files or databases (e.g. CSV files, Excel files, databases) into a project.

Dragging the Variable and Signals to Drawing Field

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>From the Variable Declaration Editor drag the variables into the drawing field. See Drag-And-Drop of Variables, p. 70.</td></tr>
<tr><td>2</td><td>From the Signal Editor drag the signals into the drawing field. See Connecting Signals to Logical Program, p. 298.</td></tr>
<tr><td>3</td><td>After dragging the variables and signals into the drawing field of the Project Management window and in addition a Not Function block from the Bitstr library, connect the variables and signals to the inputs and outputs of the emergency stop function blocks.
Emergency Stop 1


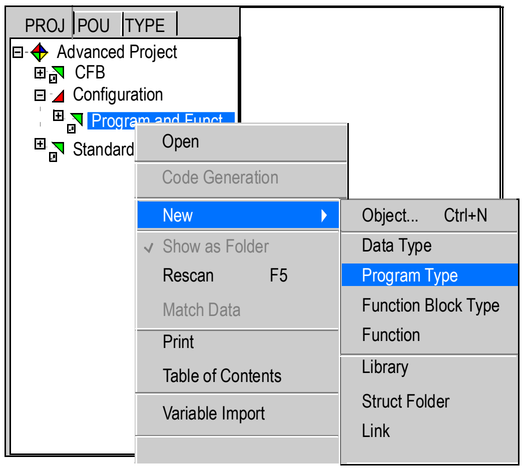
Emergency Stop 2

</td></tr>
<tr><td>4</td><td>When completed verify that all inputs and outputs are connected, save the program and close program 1.</td></tr>
</table>

Creating Program 2

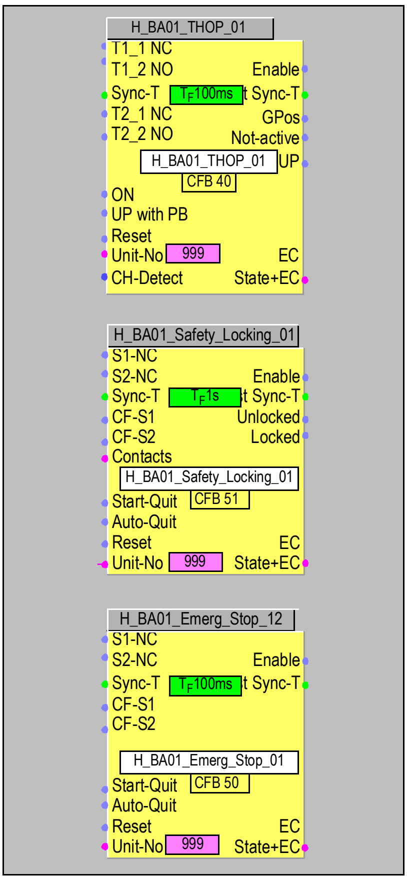
Overview

The second program is used for monitoring emergency stop functionality, safety locking and 2-hand control operation.

Creating a Program Type

Step	Action
1	Right-click on **Program and Function Blocks** to open the context menu.
2	Select **New** → **Program Type**. 
3	Click on the **+** in front of **Program and Function Blocks**. **Result**: A new function block **NewProgramType** is added to the structure.
4	Change the name of the new program type from **NewProgramType** to **Program 2**.

Dragging Function Blocks to Drawing Field

Step	Action
1	From the **CFB** folder in the structure tree find the certified function block `H_BA01_THOP_01`.
2	Click on the function block, press and hold the left mouse button and drag the function block from the structure tree into the drawing field.
3	Repeat step 1 and 2 for the `H_BA01_Safety_Locking_01` function block.
4	Repeat step 1 and 2 for the `H_BA01_Emerg_Stop_01` function block. Result: 

Signals/ Variables

The function block characteristics (input and output information and wiring information) can be viewed by selecting the **F1** function key, either when selecting the function block from the structure tree or in the drawing field.

Using the information provided by the online help of a function block, you can define whether to use signals or variables. See *Signals/Variables, p. 346*.

Creating Variables

Create the variables listed in the table below in the `VAR` tab of the **Variable Declaration Editor**. (See *Creating a Variable, p. 68*).

Name	Declaration	Init. Value	Longname	Attribute
Auto_Quit	BOOL	0	Requires_a_Reset_to_Start	C
Emergency_Stop_Sync_time	TIME	t#50ms	Emergency_Stop_synchronisation_time	C
Locking_Contact_Mode	UINT	1	Safety_locking_Operating_mode	C
Safety_Locking_Sync_time	TIME	t#100ms	Safety_Locking_synchronisation_time	C
Start_Quit_ESTOP	BOOL	1	Automatic_Function_Block_Ack_on_Start_up_for_E_Stop	C
Start_Quit_Locking	BOOL	0	Safety_Locking_Function_Block_Acq.	C
THOP_ON	BOOL	1	Enables_Function_Block	C
THOP_Sync_time	TIME	t#50ms	Two-hand_control_synchronisation_time	C
Unit_Number_4	UINT	4	Safety_Locking_4	C
Unit_Number_5	UINT	5	Emergency_Stop_5	C
Unit_number_3	UINT	3	Two-hand_Control_unit_3	C
Global Variables				
Channel Detect	ARRAY (1 to 4) of BYTE	0	Channel_Error_THOP	-

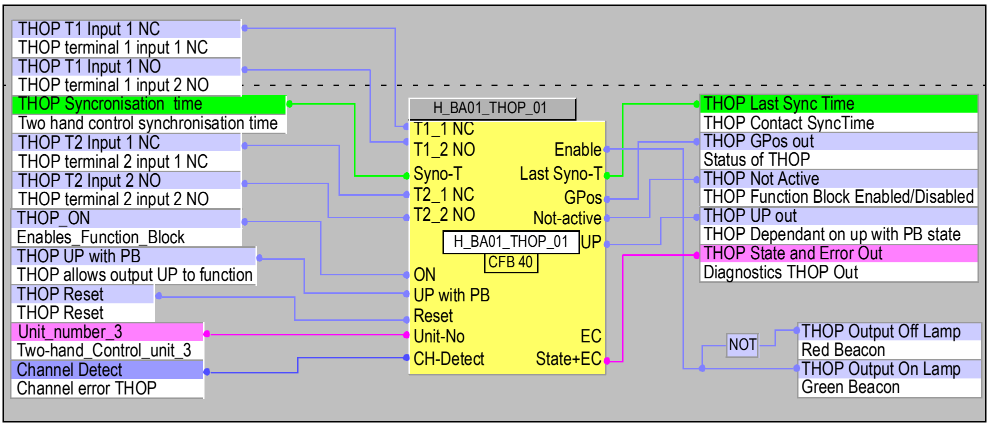
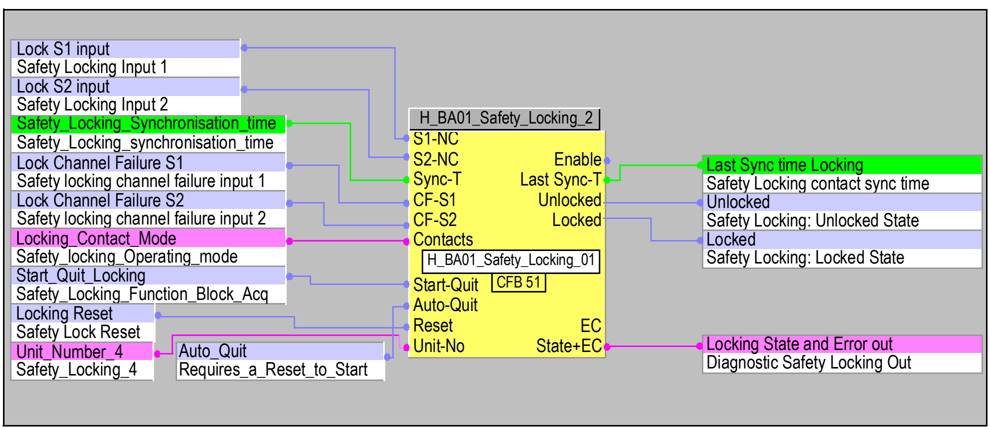
Creating Signals Create the signals listed in the table below in the **Signal Editor**. (See *Creating Signals, p. 297*. You do not need to define whether the signal is an input or output.

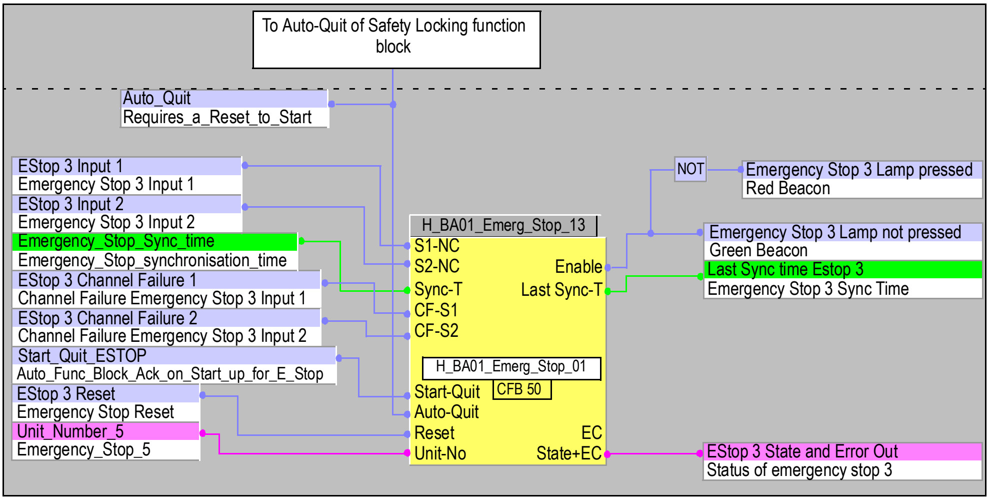
Name	Type	Retain	Constant	Description	Value
EStop 3 Channel Failure 1	BYTE	-	-	Channel Failure Emergency Stop 3 Input 1	-
EStop 3 Channel Failure 2	BYTE	-	-	Channel Failure Emergency Stop 3 Input 2	-
EStop 3 Input 1	BOOL	-	-	Estop 3 Input 1	-
EStop 3 Input 2	BOOL	-	-	Estop 3 Input 2	-
EStop 3 Reset	BOOL	-	-	Emergency Stop Reset	-
EStop 3 State and Error Out	UDINT	-	-	Status of emergency stop 3	-
Emergency Stop 3 Lamp not pressed	BOOL	-	-	Green Beacon	-
Emergency Stop 3 Lamp pressed	BOOL	-	-	Red Beacon	-
Last Sync time Estop 3	TIME	-	-	Emergency Stop 3 Sync Time	-
Last Sync time Locking	TIME	-	-	Safety Locking contact sync time	-
Lock Channel Failure S1	BYTE	-	-	Safety locking channel failure input 1	-
Lock Channel Failure S2	BYTE	-	-	Safety locking channel failure input 2	-
Lock S1 input	BOOL	-	-	Safety Locking Input 1	-
Lock S2 input	BOOL	-	-	Safety Locking Input 2	-
Locked	BOOL	-	-	Safety Locking: Locked State	-
Locking Reset	BOOL	-	-	Safety Lock Reset	-
Locking State and Error out	UDINT	-	-	Diagnostic Safety Locking Out	-
Output Robot	BOOL	-	-	Dangerous Robot cell	-
THOP Gpos out	BOOL	-	-	Status of THOP	-
THOP Last Sync Time	TIME	-	-	THOP Contact Sync Time	-
THOP Not Active	BOOL	-	-	THOP Function Block Enabled/Disabled	-
THOP Output Off Lamp	BOOL	-	-	Red Beacon	-
THOP Output On Lamp	BOOL	-	-	Green Beacon	-
THOP Reset	BOOL	-	-	THOP Reset	-
THOP State and Error Out	UDINT	-	-	Diagnostics THOP Out	-
THOP T1 Input 1 NC	BOOL	-	-	THOP terminal 1 input 1 NC	-
THOP T1 Input 2 NO	BOOL	-	-	THOP terminal 1 input 2 NO	-
THOP T2 Input 1 NC	BOOL	-	-	THOP terminal 2 input 1 NC	-
THOP T2 Input 2 NO	BOOL	-	-	THOP terminal 2 input 2 NO	-
THOP UP out	BOOL	-	-	THOP Dependant on up with PB state	-
THOP UP with PB	BOOL	-	-	THOP allows output UP to function	-
Unlocked	BOOL	-	-	Safety Locking: Unlocked State	-

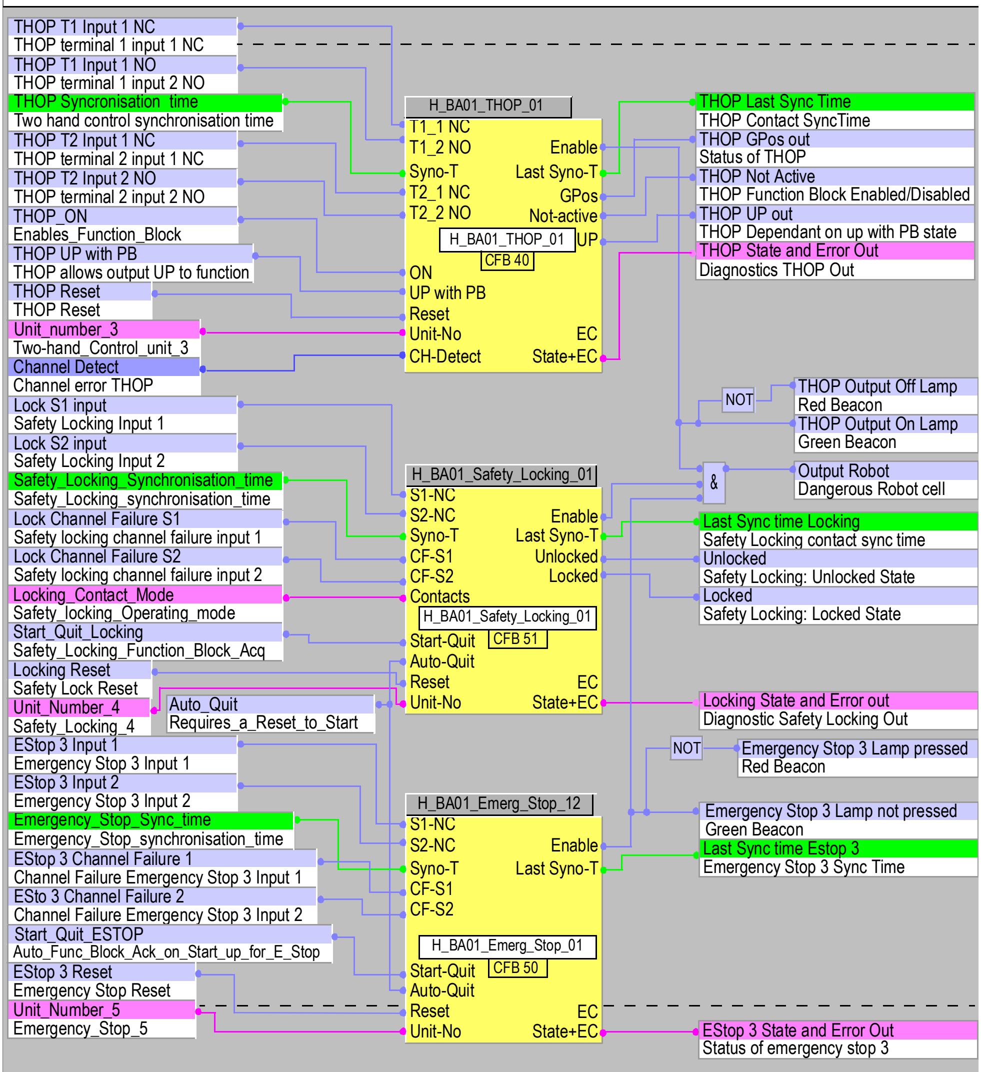
Note: When an initial value is not set in either the signal Editor or variable declaration editor the initial value is taken as 0.

Signals can be created in the **Signal Editor** in the hardware management or can be imported from external files or databases (e.g. CSV files, Excel files, databases) into a project.

Dragging the Variable and Signals to Drawing Field

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>From the Variable Declaration Editor drag the variables into the drawing field. See Drag-And-Drop of Variables, p. 70.</td></tr>
<tr><td>2</td><td>From the Signal Editor drag the signals into the drawing field. See Connecting Signals to Logical Program, p. 298.</td></tr>
<tr><td>3</td><td>After dragging the variables and signals into the drawing field of the Project Management window, connect the variables and signals to the inputs and outputs of the 2-hand operation function blocks.
2-hand control operation

</td></tr>
<tr><td>4</td><td>Safety locking (do not connect the <code>Enable</code> pin at this stage)

</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>5</td><td>Emergency stop (connecting the <code>Auto_Quit</code> variable to the emergency stop <code>Auto_Quit</code> input)

</td></tr>
<tr><td>6</td><td>Drag a <code>AND</code> gate from the IEC 61131-3 Libraries Bitstr folder into the drawing field as shown in the diagram below.
Note: When the <code>AND</code> gate is initially placed on the drawing field, the gate only contains 2 inputs. To expand the <code>AND</code> gate to have 3 inputs, place the mouse over the lower edge of the <code>AND</code> gate, left-click and move the mouse downwards expanding the number of inputs to 3.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>7</td><td>Connect all of the <code>Enable</code> outputs from the 3 certified function blocks to the inputs of the <code>AND</code> gate and connect the final output signal <code>Output Robot</code>
The final logic should be as follows:

</td></tr>
<tr><td>8</td><td>To enter a comment for the page, right-click in a free place within the drawing field, then select Create AC from the context menu, and enter text.</td></tr>
<tr><td>9</td><td>When completed verify that all inputs and outputs are connected, save the program and close program 2.</td></tr>
</table>

Creating a Flashing Lamp Warning Function Block

Overview

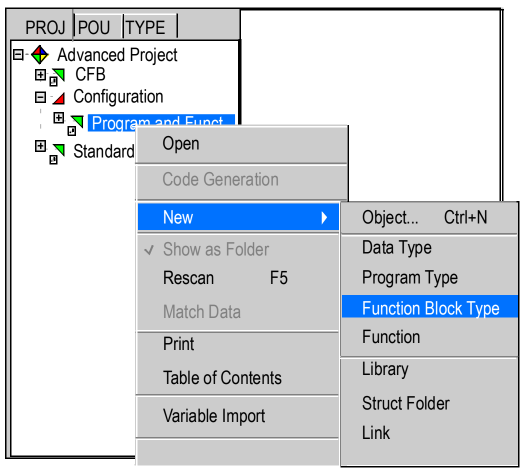
Final logical design is to create a flashing lamp warning function block.

This section will clearly explain how to create personal function blocks, which can be used within programs.

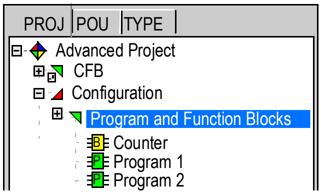
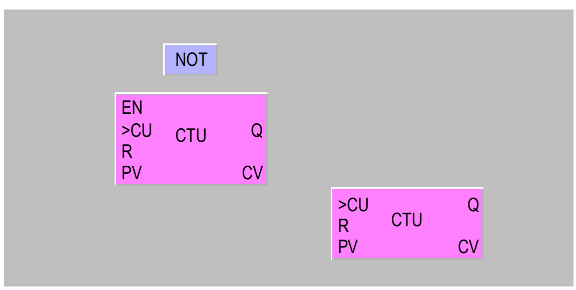
Function blocks reduce the overall engineering time and provide structure in a project. Therefore any repetitive logic should be created within a function block.

The function block you will create is a timer, counting upwards, which will be used to control a flashing LED lamp.

Creating the Counter Function Block Type

Step	Action
1	Right-click on **Program and Function Blocks** to open the context menu.
2	Select **New** → **Function Block Type**. 
3	Click on the **+** in front of **Program and Function Blocks**. **Result**: A new function block **New Function Block Type** is added to the structure.
4	Change the name of the new function block type from **NewFB** to **Counter**.

Dragging Function Blocks to Drawing Field

Step	Action
1	Double click on **Counter** from the **Program and Function Blocks** library to open it.  **Note:** To create a logical within the function block is done in the same methodology as within a program.
2	From the **IEC 61131-3** library open the **Counter** folder and find the function block `CTU`.
3	Click on the `CTU` function block, press and hold the left mouse button and drag the function block from the structure tree into the drawing field. **Note:** The function block characteristics can be viewed by selecting the **F1** function key, either when selecting the function block from the structure tree or in the drawing field. **Result:** When placing the certified function block on the page the **Edit Page Data** window will open (please refer to the *Basic Project Configuration* section *Editing Page Data, p. 296*).
4	Repeat step 3.
5	Drag a `NOT` gate from the IEC 61131-3 libraries **Bitstr** folder into the drawing field. Result: 

Note: To display the "EN" field: Right-click on the function block and select **Show EN/ENO** from the context menu.

Creating Variables

The function block in this case does not need any external I/O connections (they will be connected outside the function block), therefore, you will only require variables.

Create the variable listed in the table below in the VAR_INPUT tab of the **Variable Declaration Editor**. (See *Creating a Variable, p. 68*).

Name	Declaration	Initial Value	Longname	Attribute
Start Counter	BOOL	0	initiates counting sequence	-

Note: The VAR_INPUT definition will create a physical input connection on the function block.

Create the variables listed in the table below in the VAR tab.

Note: These variables are not shown on the function block.

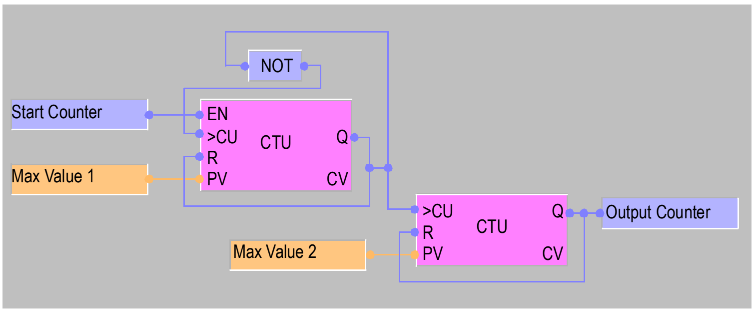
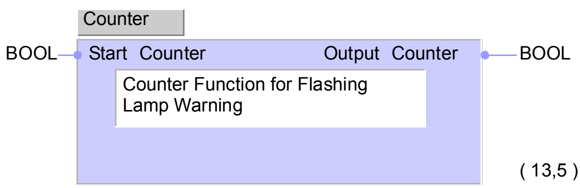
Name	Declaration	Initial Value	Longname	Attribute
Max Value 1	INT	1	maximum count	C
Max Value 2	INT	20	maximum count	C

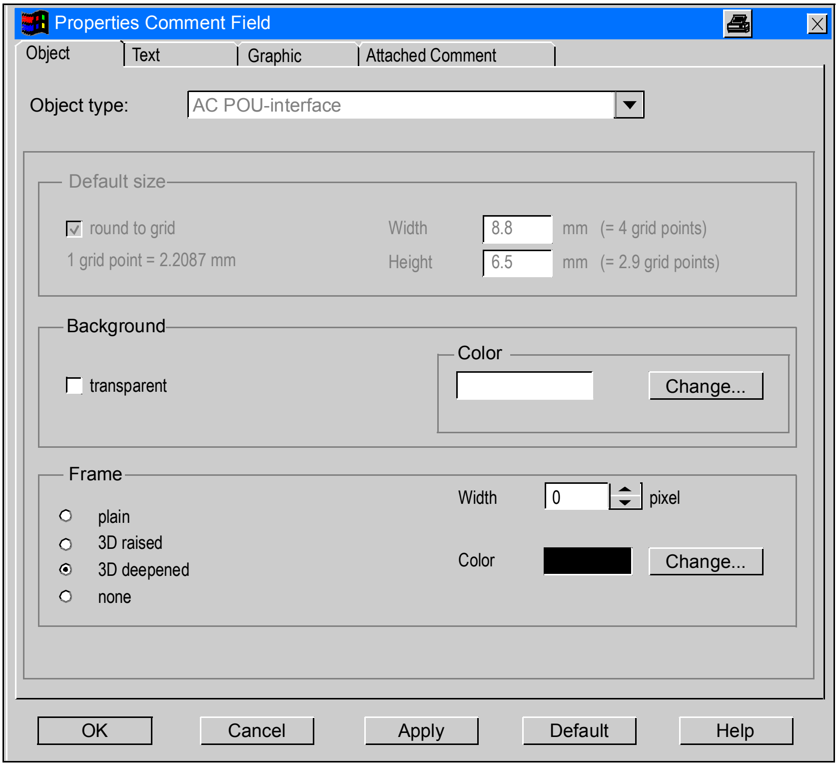
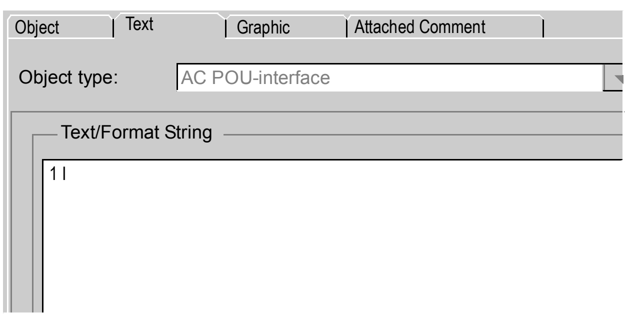
Create the variable listed in the table below in the VAR_OUTPUT tab.

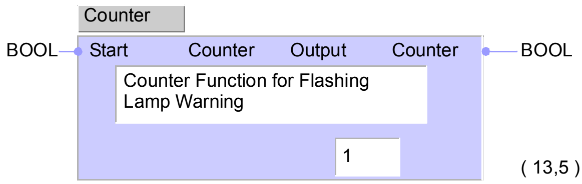
Name	Declaration	Initial Value	Longname	Attribute
Output Counter	BOOL	0	output value from counter	-

Note: The VAR_INPUT and VAR_OUTPUT definition will create a physical input/output connection on the function block.

Dragging the Variables to Drawing Field

Step	Action
1	From the **Variable Declaration Editor** drag the variables into the drawing field. See *Drag-And-Drop of Variables, p. 70*.
2	After dragging the variables into the drawing field of the **Project Management** window, connect the variables according to the diagram below. 
3	Right-click on the **Counter Function** block, select **Create Comment Field** from the context menu and enter `Counter Function for Flashing Lamp Warning` into the comment field. 
4	Then again right-click on the function block and select **Create an Attribute Comment Field**.
5	Right-click on the function block and select **Properties**. Within the **Instance Name** tab enter `Counter`.

Step	Action
6	Right-click on the attribute comment field, select **Properties Comment Field** and subsequently the **Text** tab. 
7	Enter `1` in the **Text/Format String** area. 

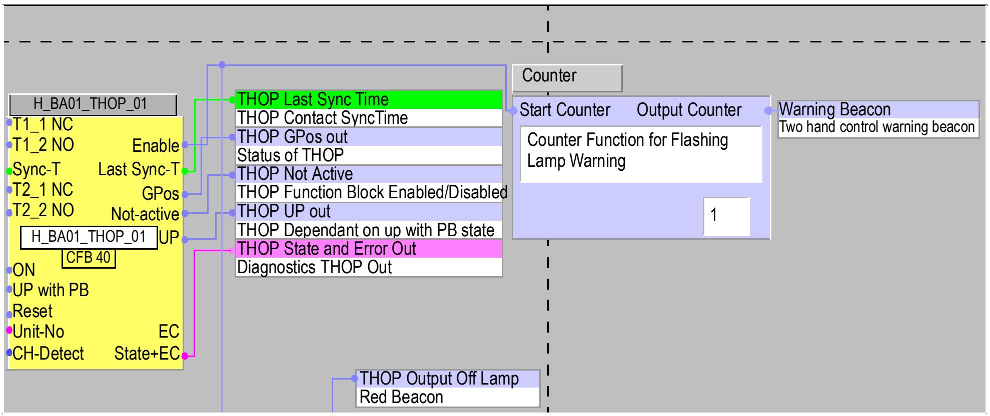
Step	Action
8	Click **Apply** and then **OK**. The counter function block should look as follows. 
9	When completed verify that all inputs and outputs are connected, save the counter function block and close. **Note:** The counter function block can now be used in the same way as the certified function blocks.

Creating a Signal

Create the following new signal in the **Signal Editor** of the hardware management. (See *Creating Signals, p. 297*.

Name	Type	Retain	Constant	Description	Init Value
Warning Beacon	BOOL	-	-	two hand control warning beacon	0

Integrating the Counter Function Block

Step	Action
1	Open **Program 2**.
2	Drag the **Counter Function** block on to a free space within the drawing field.
3	Connect the `Warning Beacon` signal to the output of the counter function, and connect the `Enable` pin of the `2-hand operation function block` to the input of the counter function. Result (detail): 
4	Save **Program 2** and close.

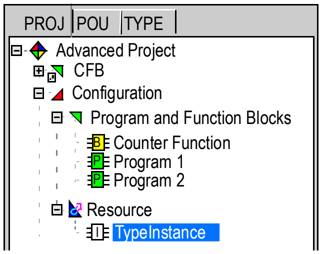
Connecting Programs to Resources

Overview

Program 1 and program 2 must be connected to a PLC resource in order for a PLC to run the program.

In XPSMFWIN it is possible to have multiple resources (PLCs) within an overall project.

As a default, the software contains 1 resource with an attached type instance.



The **TypeInstance** attached to the resource is similar to the program type, which you have created the logic, and can also be used to create logic.

Programming Benefits

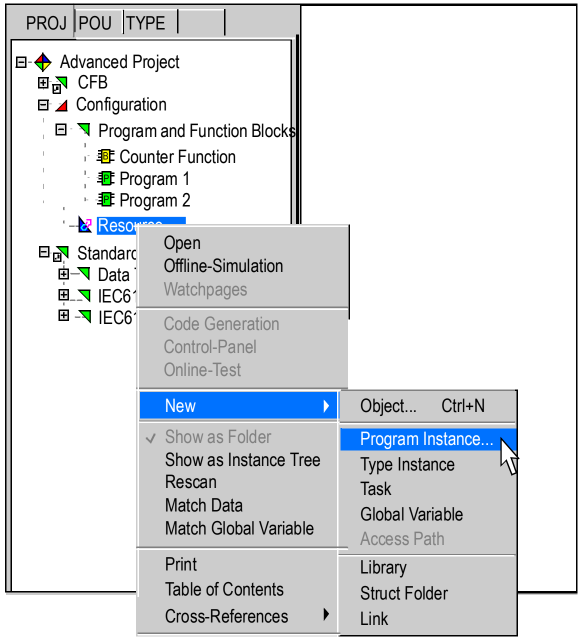
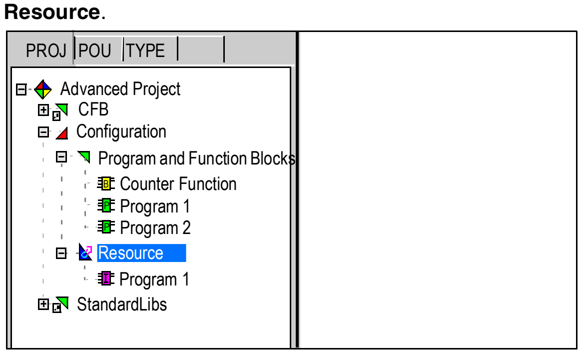
The following table explains the benefits of programming in either a program library (as you have carried out) or directly in a TypeInstance on the resource

Programming Where	Memory Size	Loss of Data	Use of Logic on Multiple PLCs
TypeInstance on the resource	lower, as it only contains data from single PLC	possible, if the resource is deleted the program is deleted	must copy and paste it to another PLC, time consuming
program folder	slightly higher	not possible, due to data stored in individual folder	the program can be assigned to as many PLCs as required in the configuration

It is clearly best to create all program logic within a personal library (as you have created).

Note: The resource (PLC) can only run 1 program.
A program can be used as many times as required.

Connecting Program 1 to a Resource

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Click on the TypeInstance and press DEL (delete).
Note: Before you can connect a program to a resource (PLC), the TypeInstance must be deleted from the resource.</td></tr>
<tr><td>2</td><td>Right-click on Resource and select New → Program Instance...

</td></tr>
<tr><td>3</td><td>From Program and Function Blocks select Program 1.</td></tr>
<tr><td>4</td><td>When the program instance is connected a purple I can be seen attached to the Resource.

</td></tr>
</table>

Connecting Program 2 to a Resource

Program 2 can also be connected to a resource in the same way.

As the default only provides a single resource within the configuration, right-click on the **Configuration** folder and select **New** → **Resource**.

To rename the new resource, select the resource, press function key **F2** and enter a new name.

Note: In the case where more than 1 PLC is using the same program, simply connect the program in the same way as described above to both PLCs (resources).

Offline Simulation of Program

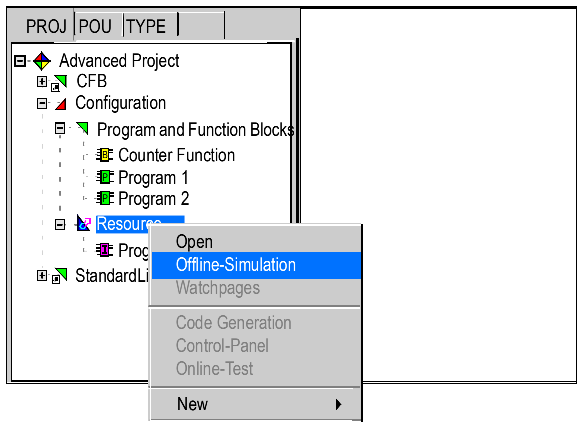
Overview

The offline simulation allows you to check the connection flow ensuring that the overall program logic functionality is correct without having to connect and download to a PLC.

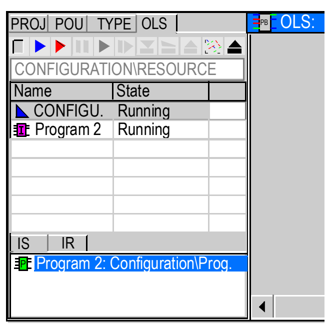
To run the offline simulation (OLS) the logical programs must be attached to the resource folder. See *Connecting Programs to Resources, p. 365*.

Offline Simulation

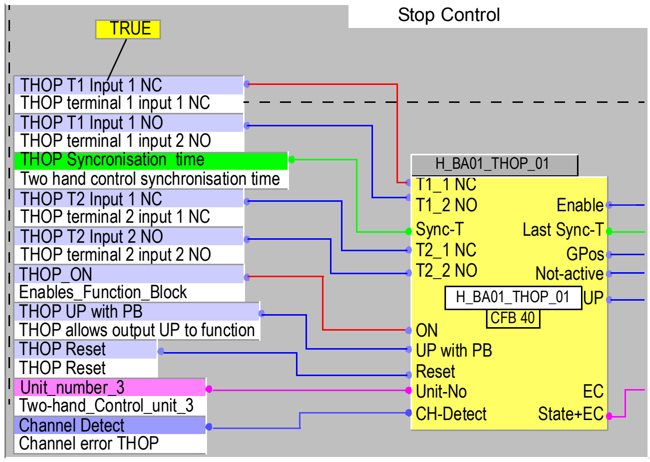
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>In the Project Management, right-click on the Resource and select Offline-Simulation.



Result:
The following window opens with the new tab OLS.



</td></tr>
</table>

Step	Action
2	Double-click on **Program 2** in the **Offline-Simulation** window. The logic opens with all states. 
3	Within the drawing field right-click on a signal or a variable and select **Create OLT Field** from the context menu (a online test field (TRUE) is shown in yellow in the diagram above). **Note:** Online test fields are used to view and change the status of signals and variables.
4	Double-click on an OLT field to change the value from FALSE to TRUE. **Result:** The connector changes from blue to red.
5	Left-click on an OLT field to change numerical values, entering the values manually. For more detailed information regarding the functionality of the offline simulation, see *Offline Simulation of Function Diagrams (OLS), p. 91*.

8.3 Step 2: Defining Hardware and Remote I/O Settings and Peer-to-Peer Communication

At a Glance

Overview

This section is provided to assist you in defining hardware settings and remote I/O settings, and the peer-to-peer communication.

What's in this Section?

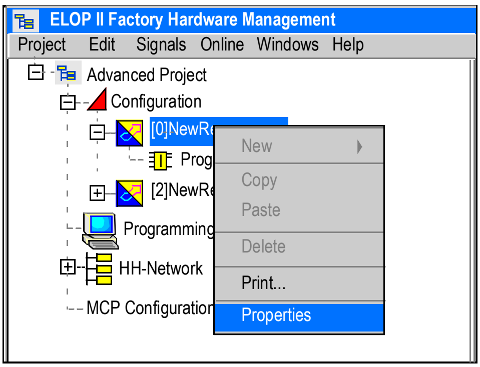
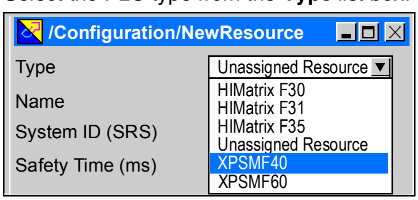
This section contains the following topics:

Defining PLC Hardware Type

Overview

The **Hardware Management** is used to define all hardware related settings, such as hardware type, communication settings, I/O assignment, etc.

Defining PLC Hardware Type

Step	Action
1	In the **Hardware Management** window open the **Configuration** folder from the structure tree.
2	Right-click on **Resource** and select **Properties** from the context menu. 
3	Select the PLC type from the **Type** list box.  The example shows XPSMF40 PLC being used.
4	Enter the **System ID [SRS]** value 2 to 65,535. **Note**: For further information on system ID (SRS = system-rack-slot) please refer to *SRS (System Rack Slot) Value, p. 156*.
5	Click **Apply**. **Note:** Clicking **Apply** allows all the menu´s sub-options to be selected.
6	For this program example, set the values as listed in the table below. **Note**: For further information please refer to *Setting the Properties of a Resource, p. 154*.
7	Click **OK**.
8	Repeat step 1 to 7 for the second resource.

Property Values

Properties	Value
Safety Time	30 ms
Watchdog Time	15 ms
Main Enable	selected
Autostart	selected
Start/Restart allowed	selected
Loading allowed	selected
Test Mode allowed	selected
Change variables in OLT allowed	selected
Forcing allowed	selected
Stop on Force Timeout	selected
max. Com.Time Slice	10 ms

Defining Remote I/O Settings

Overview

The remote I/O modules do not process the logical program.

On every cycle they send their data to their parent resource (PLC: XPSMF40, XPSMF60, XPSMF30, XPSMF31, or XPSMF35).

A remote I/O module can only have 1 parent resource.

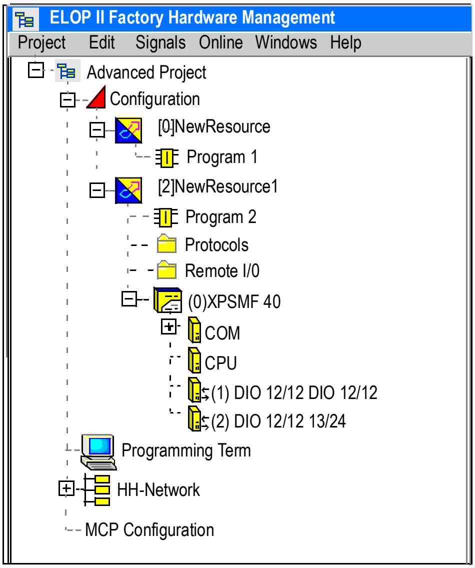
In a project it is possible to assign up to 64 remote I/O modules to each parent resource.

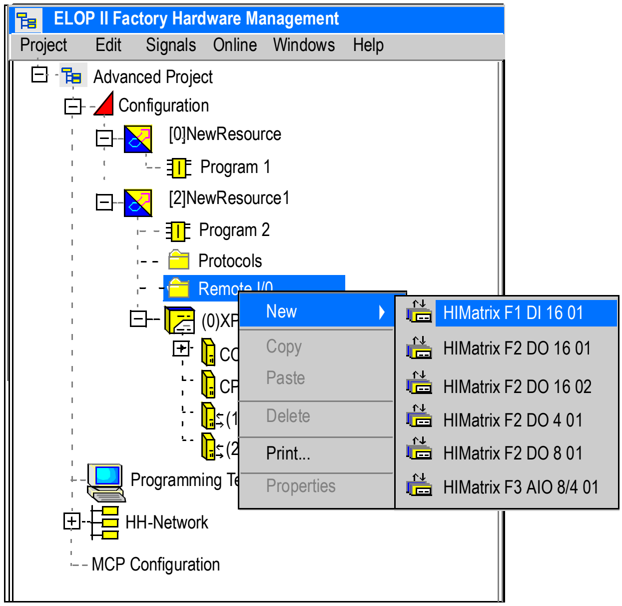
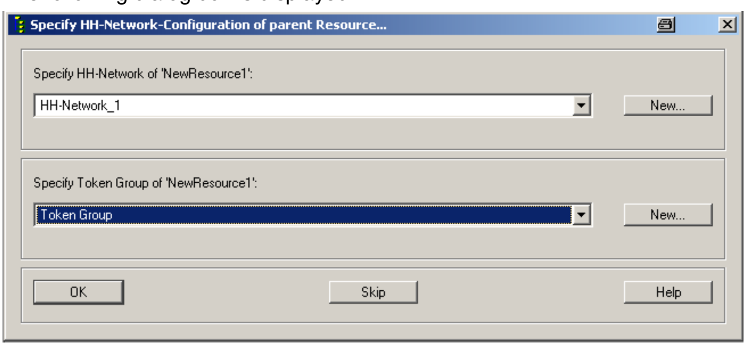
To assign remote I/O modules to a resource is simple. The main requirement is that the parent resource must be assigned as 1 of the PLCs (XPSMF40...) mentioned above.

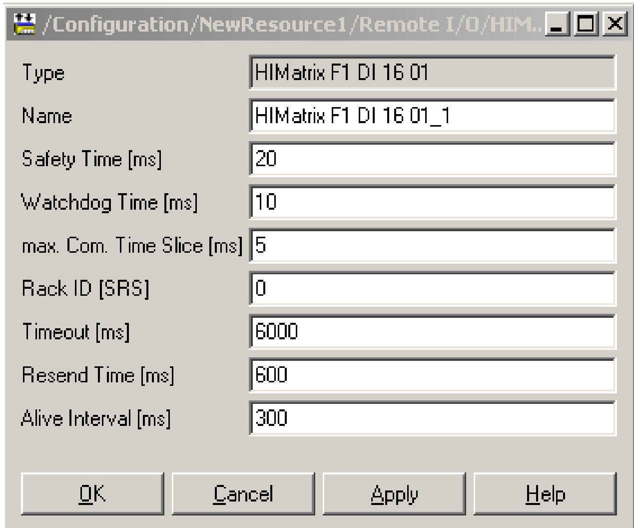
Remote I/O Modules

The following are the safety remote I/O modules:
- XPSMF1DI1601
- XPSMF2DO801/DO802/DO1601/DO1602
- XPSMF3DIO8801/DIO16801/DIO20802
- XPSMF3AIO8401

Assigning I/O Modules

Step	Action
1	Expand the structure of **NewResource1** in the **Hardware Management**.  The folder **Remote I/O** under **NewResource1** is where you assign the remote I/O.

Step	Action
2	Right-click on the **Remote I/O** folder and select **New** → **F1 DI 1601**.  The following dialog box is displayed:  For more information about token groups see *Peer-to-Peer Communication (P2P Communication), p. 187*.
3	Click **OK**. **Result:** The new remote I/O is now assigned within the remote I/O folder.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>Right-click on the remote I/O and select Properties from the context menu.
The following dialog box is displayed:

</td></tr>
<tr><td>5</td><td>Change the values of the following parameters:
Safety Time to 30 ms
Watchdog Time to 15 ms
Rack ID [SRS] to a value between 1 and 511
Note: When assigning the rack ID, 2 remote I/O modules from the same resource can not have the same rack ID. For additional information see Properties of a Resource, p. 153.</td></tr>
<tr><td>6</td><td>Click Apply and OK.</td></tr>
<tr><td>7</td><td>Repeat step 1 to 6 for each remote I/O you wish to define.</td></tr>
</table>

Token Groups

A token group is group of PLCs and remote I/Os which communicate their data over a network which all has the same profile (same network conditions).

For example for a normal Ethernet network, the typical network profile is **Fast** and **Noisy**, therefore all devices within this profile should be within the same token group.

If there is an area where you are using a different medium, or perhaps there is a higher influence of EMC on the network then you can assign a separate network profile to cater for the communication between the systems, therefore creating a new token group profile.

For more information see *Peer-to-Peer Communication (P2P Communication), p. 187*

Defining Peer-to-Peer Communication

Overview

Peer-to-peer communication is very useful, and very important. It enables the resources (PLCs) to communicate information between each other via SafeEthernet.

Peer-to-peer communication is automatically set-up between remote I/O modules and their parent resource. This is carried out when defining the remote I/O module.

Creating peer-to-peer communication is very simple, however the underlining principles behind the exact settings takes time to master.

Network Adjustment Required

In most cases the default settings for peer-to-peer communication are sufficient. However, in applications requiring very fast response times or transmission via serial/wireless network adjustments are required.

Benefit for Diagnostics

In some cases there is no need to create peer-to-peer communication; however you will find it very beneficial for diagnostic possibilities.

State of Your Advanced Project

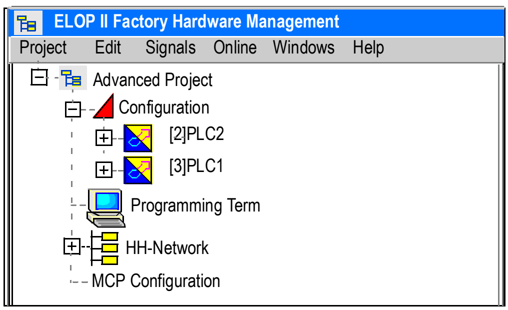
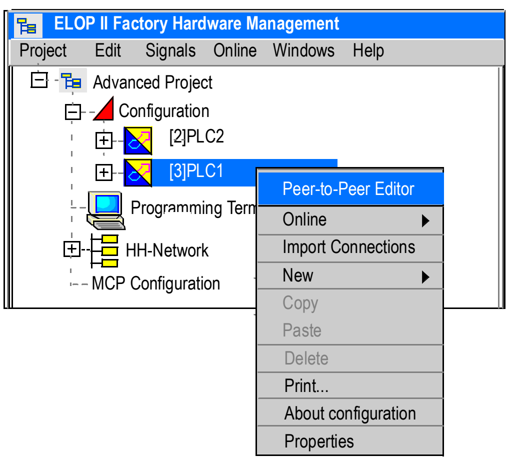
At this stage you have created your logic configuration, and have defined your 2 different PLCs within your project, and have possibly 1 or more remote I/O modules attached to a single resource or both resources.

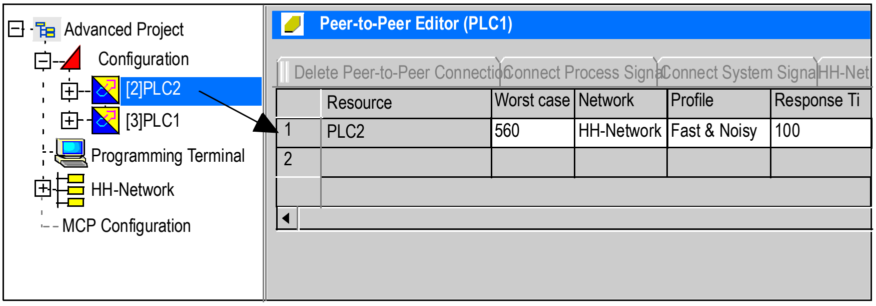
Signals Transfer

> **Note:** To transfer information between PLC1 and PLC2, only the input signals are transferred to the second PLC and never the output signals.

Defining Peer-to-Peer Communication

As the peer-to-peer communication is mostly logic independent (except you require signals from the signal editor to send some information between 2 PLCs), it is normally ideal to set this up as soon as you define your hardware.

Step	Action
1	To simplify this example change the names of the 2 resources as follows: ● PLC1 (resource with program 1) ● PLC2 (resource with program 2) **Note:** The name change can only be carried out within the **Project Management** window.
2	Expand the structure of **Configuration** in the **Hardware Management**. **Result:** 
3	Right click on **PLC1** and select **Peer-to-Peer Editor**. 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>Click on PLC2 and drag it into the Peer-to-Peer Editor and subsequently click OK in the Token Group dialog box.


Result:
The diagram above shows that now PLC1 and PLC2 are connected to one another.
Note: The Worst case is the time required when calculating the safety distances and can be reduced by adjusting Network, Profile, Response Time, and ReceiveTMO.
For more information regarding settings, see Peer-to-Peer Communication (P2P Communication), p. 187.</td></tr>
<tr><td>5</td><td>At this stage, leave the settings with the default values. Therefore, the peer-to-peer communication is now set up.</td></tr>
</table>

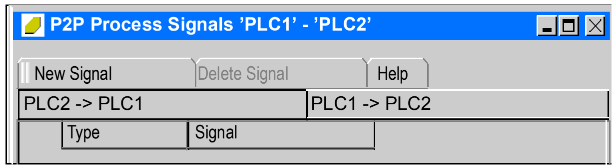
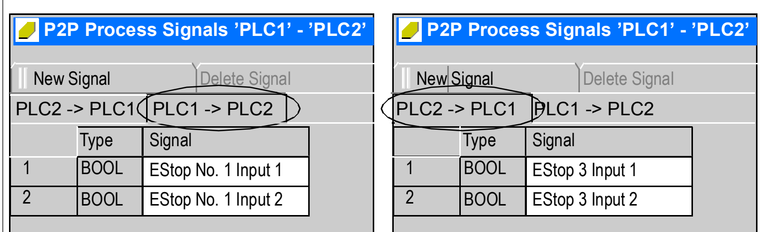
Tabs of the Peer-to-Peer Editor

After clicking on the number **1** to the left of the **PLC2** resource, the following options are available:

- Delete Peer-to-Peer Connection (enables you to remove the peer-to-peer connection)
- Connect Process Signals (enables you to transfer input signals from 1 PLC to another)
- Connect System Signals (enables you to transfer operation signals between 2 PLCs, such as connection state, etc.
- HH-Network Configuration (provides information within the **Error State Viewer** of the state of the peer-to-peer connection

Connecting Process Signals

Now you define which data is to be transmitted between the 2 PLCs.

Step	Action
1	Select the **Connect Process Signals** tab.
2	Select the tab for the direction of communication (**PLC2->PLC1**). 
3	Open the signal editor (**Signals** → **Editor**) and position it beside the **P2P Process Signals** window.
4	Click on a signal you want to transfer between the 2 PLCs and drag it to **P2P Process Signals** window. In this example use 2 signals in each direction. For PLC1->PLC2 use: ● EStop No. 1 Input 1 ● EStop No. 1 Input 2 For PLC2->PLC1 use: ● EStop 3 Input 1 ● EStop 3 Input 2  **Note:** The signals you choose are dependant on what information you need in each PLC. The order to which the signals are listed is not important.

Process Signals

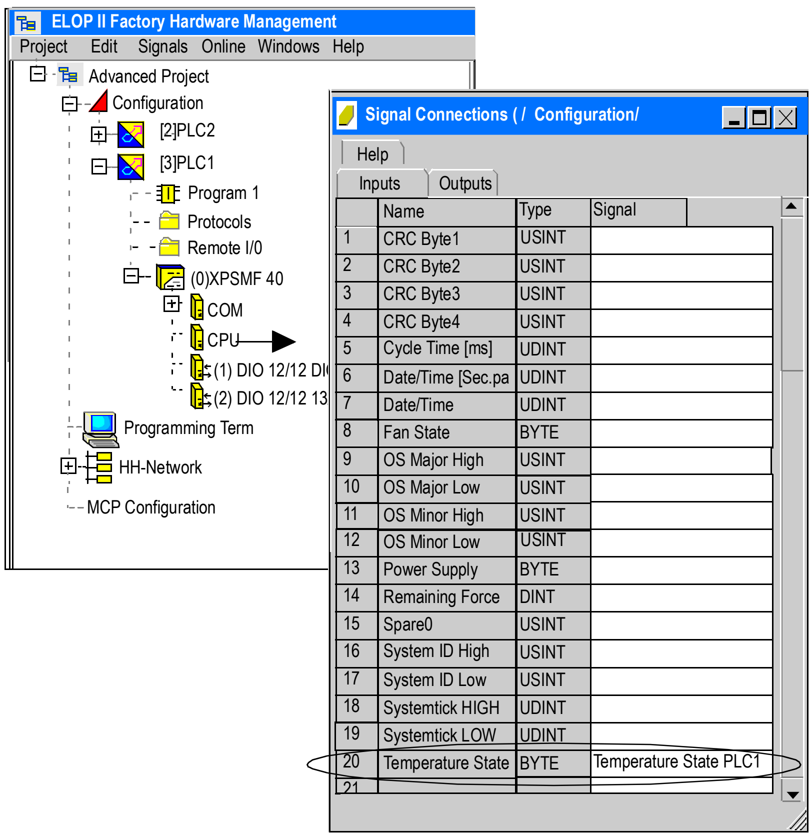
Additional process signals which are not within the projects logical program can also be transferred, the method is identical.

These process signals can come from input data from non-safety related communications, or from the system diagnostic information provided within the CPU tab of each PLC.

To use any of the predefined signal inputs, signals must be created within the **Signal Editor**.

For example create a `Temperature State PLC1` signal (BYTE) in the **Signal Editor** and connect it to the `Temperature State` input from the PLC1 CPU connect signals window.

The following diagram shows the PLC1 CPU signal connections with the `Temperature State` signal connected.

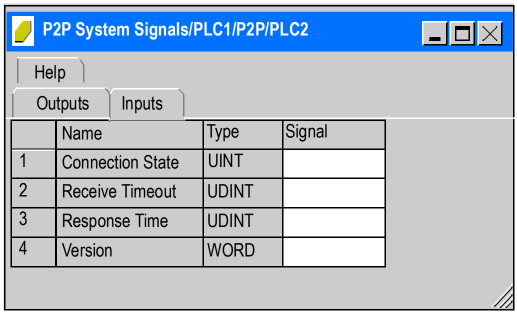


To now transfer this data to **PLC2**, select **Connect Process Signals** from the **Peer-to-Peer Editor**, and enter the signal within the **PLC1->PLC2** tab.

System Signals

The **Connect System Signals** tab of the **Peer-to-Peer Editor** enables you to use the predefined inputs/outputs for diagnostic information between the 2 PLCs (useful to send this information via Modbus etc. to a standard control PLC).

To use any of the system signals, create a new signal within the **Signal Editor** and attach it to the desired input or output.



Using the Signals

To now make use of the signals which are to be transferred between the 2 PLCs, drag the signals

- EStop No. 1 Input 1
- EStop No. 1 Input 2

into the program 2 located within the **Project Management** window.

Drag the signals

- EStop 3 Input 1
- EStop 3 Input 2

into the program 1 located within the **Project Management** window.

Note: The EStop signals come from a safe source and are transferred between the 2 PLCs using SafeEthernet communication protocol. Therefore, the signals can be used within another emergency stop control function block to react on an output a distance away from the PLC.

At this stage leave the signals not connected within the programs, save the programs and close.

8.4 Step 3: Creating Non-Safety Protocol Communication

At a Glance

Overview

This section is provided to assist you in creating non-safety protocol communication.

What's in this Section?

This section contains the following topics:

Modbus Slave (TCP/IP and RTU)

Overview

The XPSMF safety PLCs can communicate to non-safety related automation devices via a number of protocols.

The most typical communication options are available with Schneider Electric PLC systems:

- Modbus (RTU and TCP/IP)
- Profibus DP

Protocols for Safety PLCs

The Modbus Slave will be placed within the **Protocols** folder of the **PLC2** resource when assigned to the PLC.

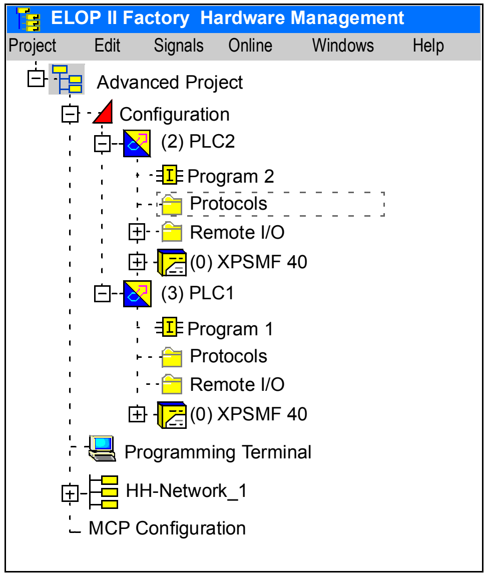
The Modbus slave can be either/both Modbus TCP/IP server or Modbus RTU slave.

Note: All of the XPSMF safety PLCs (except the XPSMF4000/ F4020/F4040) are always provided with the Modbus TCP/IP server protocol. Only specific versions have Modbus RTU slave protocol, and Profibus DP slave protocol.

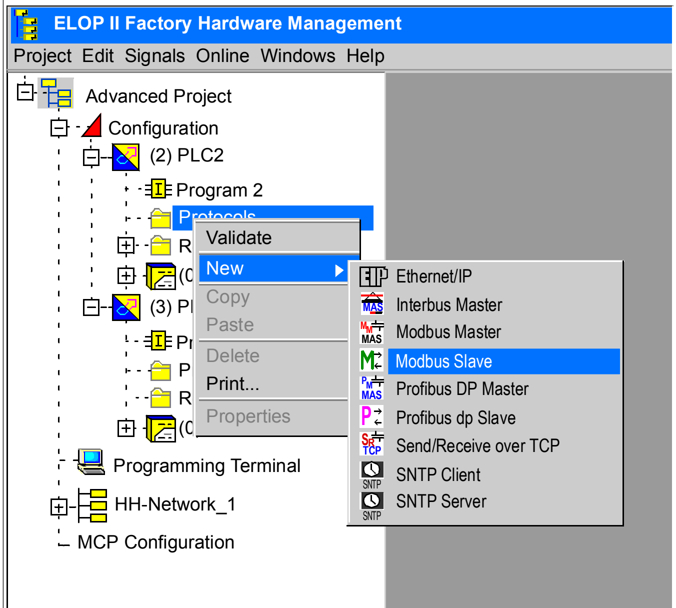
To identify if your version of safety PLC contains the Modbus RTU slave, check the hardware reference:

Safety PLC	Protocol
XPSMF4000	none
XPSMF4002	Modbus TCP/IP server
XPSMF4020	Modbus RTU slave
XPSMF4022	Modbus TCP/IP server and Modbus RTU slave
XPSMF4040	Profibus DP slave (V0)
XPSMF4042	Profibus DP slave and Modbus TCP/IP server
XPSMF3022	Modbus TCP/IP server and Modbus RTU slave
XPSMF31222	Modbus TCP/IP server
XPSMF3502	Modbus TCP/IP server
XPSMF3522	Modbus TCP/IP server and Modbus RTU slave
XPSMF3542	Modbus TCP/IP server and Profibus DP slave (V0)
Modular PLC (XPSMFCPU22)	Modbus TCP/IP server and Modbus RTU slave
Remote I/O	no non-safe communications as this is managed by the parent safety PLC.

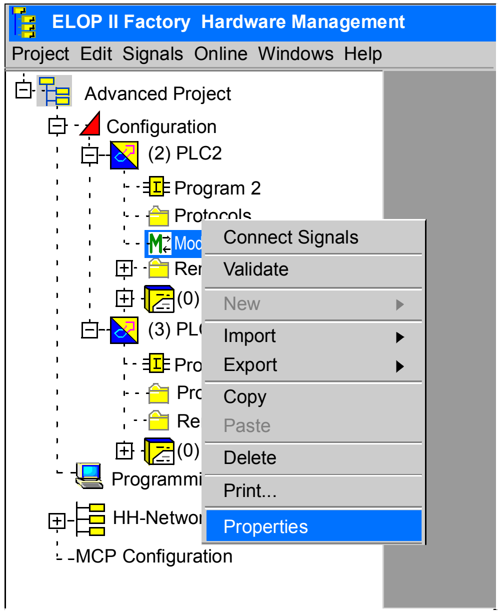
Adding a New Modbus Slave

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Within the Hardware Management window open the project tree as shown below:


</td></tr>
</table>

Step	Action
2	Right-click on the **Protocols** of **PLC2** and select **New** → **Modbus Slave** from the context menu. 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>To assign the properties of the Modbus slave (either or both Modbus TCP/IP server and Modbus RTU slave), right-click on the Modbus Slave and select Properties.



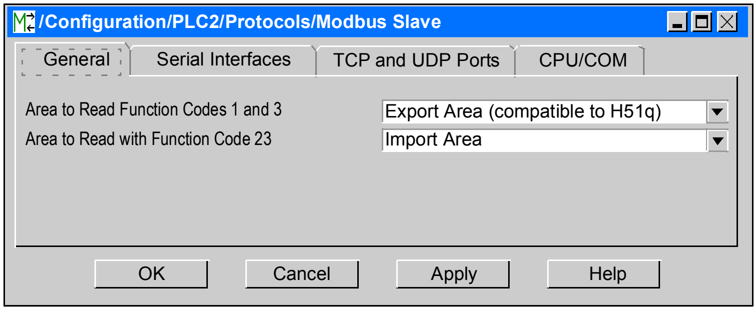
</td></tr>
<tr><td>4</td><td>For this example only the Modbus TCP/IP server will be selected.
The following are the properties for each of the 4 tabs:

General: default values, no change required
Serial Interfaces: default values, no change required
TCP and UDP Ports: select both Enable TCP and Enable UDP with the Maximum number of connections as 5
CPU/COM: default values, no change required
</td></tr>
<tr><td>5</td><td>Confirm with OK.</td></tr>
</table>

Modbus Slave Properties Dialog

The properties dialog has 4 tabs:
- General
- Serial Interface
- TCP and UDP Ports
- CPU/COM

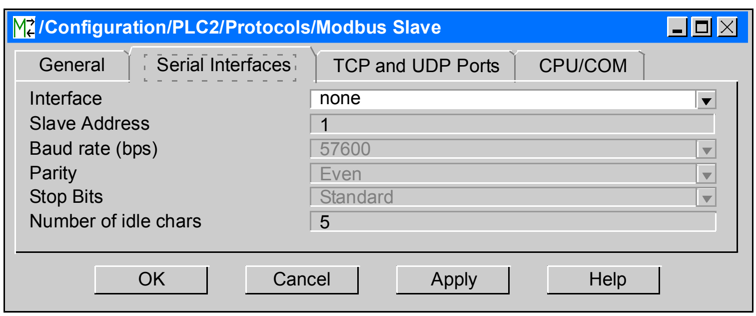
General tab



The **General** tab defines the areas to read function codes from
- import area: input from Modbus Master
- export area: output to be read by Modbus master

Typically, the default values are okay, however when communicating with specific Modbus masters you may need to adjust the read areas for the above function codes.

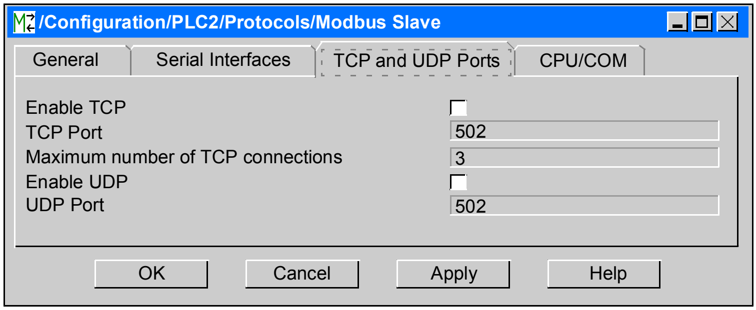
Serial interfaces tab



The **Serial Interfaces** tab allows you to define the Modbus RTU slave properties. Selecting the interface as a FB1, FB2 or FB 3 (depending on PLC) and selecting **Apply** enables you to adjust the slave address, baud rate, parity, stop bits, and number of idle characters.

When you do not require/or do not have the serial interface leave the interface with **none**.

TCP and UDP Ports

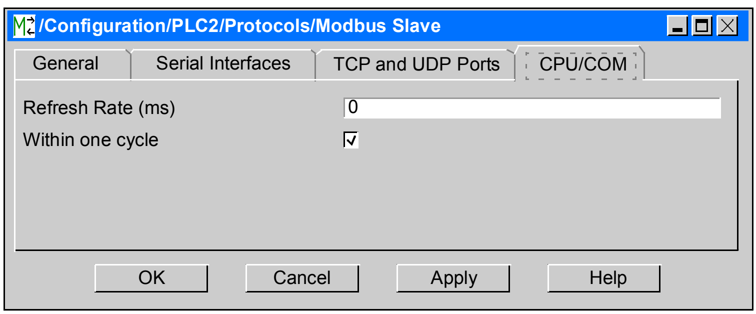


The **TCP and UDP Ports** tab allows you to define your Modbus TCP/IP server properties.

Selecting the **Enable TCP** and/or **Enable UDP** (depending on PLC) and selecting **Apply** enables you to adjust the port address and the maximum number of Modbus TCP/IP connections (number of Modbus TCP/IP clients which will be communicating with the PLC).

When you do not require/or do not have the Modbus TCP/IP server protocol do not select the **Enable TCP** or **Enable UDP** check boxes.

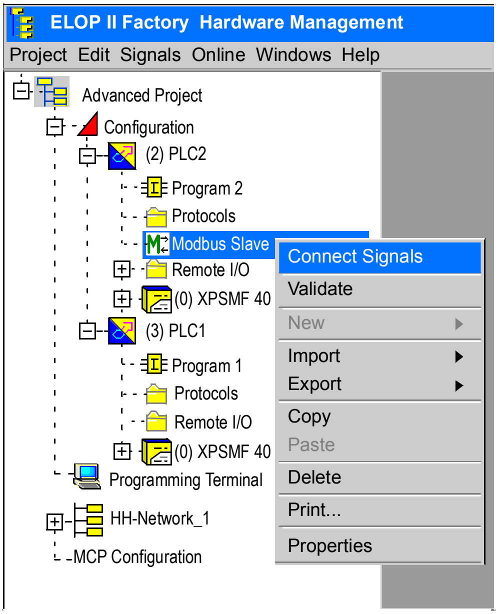
CPU/COM tab

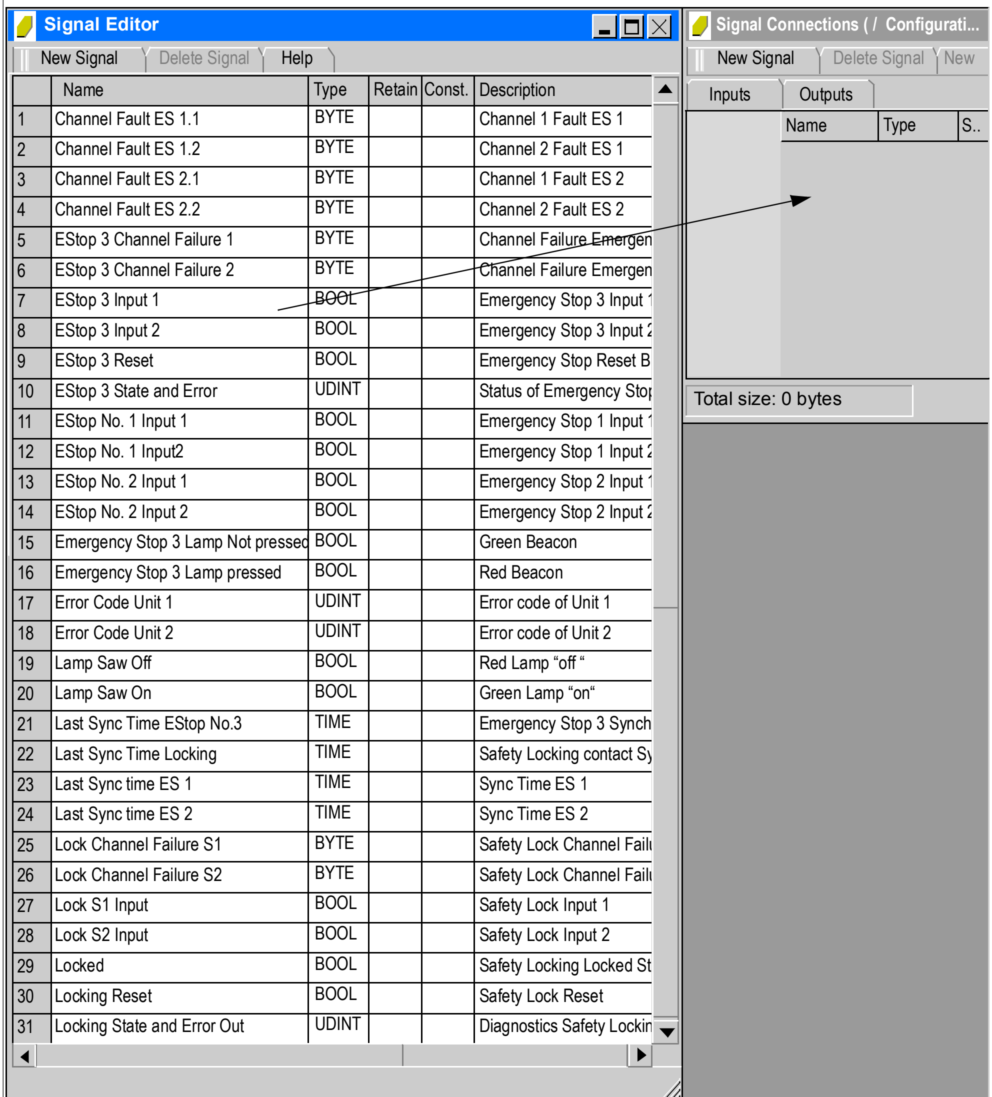


The **CPU/COM** tab allows you to define you're input and output data refresh rate Modbus properties.

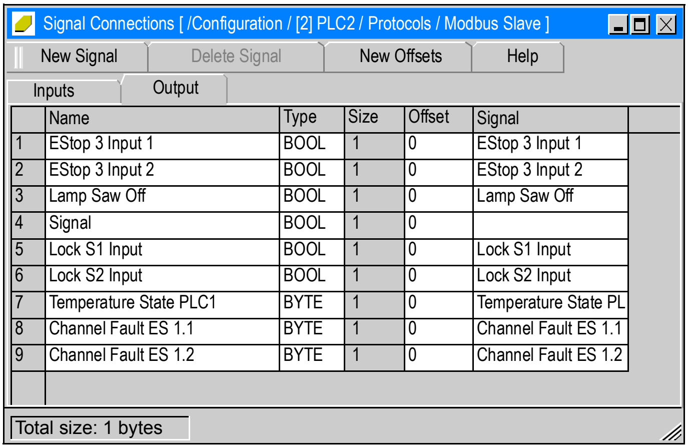
The **Refresh Rate** can be made longer when the data is not needed on every cycle. Typically, the default settings are sufficient.

Connecting Signals

Step	Action
1	To set-up the signals which you require read or written to, right-click on **Modbus Slave** and select **Connect Signals** from the context menu.  **Note:** The **Signals Connections** dialog box has 2 tabs, **Inputs** and **Outputs**. The **Inputs** tab refers to all data being written to the Modbus slave. The **Outputs** tab refers to all data which will be read from the Modbus slave

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Additionally open the Signal Editor from the Hardware Management window and tile the 2 windows next to each other (side by side).

</td></tr>
</table>

Step	Action
3	Select from the **Signal Editor** several signals which you wish to transfer to the Modbus master and drag and drop the signals into **Output** of the **Signals Connections** window. Example



	Name	Type	Size	Offset	Signal
1	EStop 3 Input 1	BOOL	1	0	EStop 3 Input 1
2	EStop 3 Input 2	BOOL	1	0	EStop 3 Input 2
3	Lamp Saw Off	BOOL	1	0	Lamp Saw Off
4	Signal	BOOL	1	0	
5	Lock S1 Input	BOOL	1	0	Lock S1 Input
6	Lock S2 Input	BOOL	1	0	Lock S2 Input
7	Temperature State PLC1	BYTE	1	0	Temperature State PL
8	Channel Fault ES 1.1	BYTE	1	0	Channel Fault ES 1.1
9	Channel Fault ES 1.2	BYTE	1	0	Channel Fault ES 1.2

Data Transfer Size

When the signals are present within the Modbus **Signal Connections** window, you can see the size of each of the signals which need to be transferred.

The size is represented in bytes.

The following data types will be transferred in:

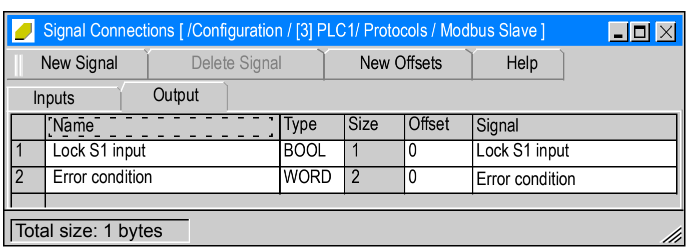
Data Type	Size
BOOLEAN	1 Byte
BYTE	1 Byte
USINT	1 Byte
SINT	1 Byte
INT	2 Bytes
UINT	2 Bytes
WORD	2 Bytes
DWORD	4 Bytes
DINT	4 Bytes
REAL	4 Bytes
UDINT	4 Bytes
LREAL	8 Bytes
TIME	8 Bytes

The minimum data transfer size is 2 bytes (WORD), therefore if you are sending only 1 BOOLEAN, BYTE, USINT, or SINT you must enter a blank signal (of the same type) after the first signal.

This can be done by simply clicking on the **New Signal** button from the Modbus slaves' **Signal Connections** window.

In addition, you must ensure that between 2 different types of signals (e.g. BOOL, and WORD) that there is an even number of boolean signals to ensure that the data read is not being split between 2 words.

Case 1

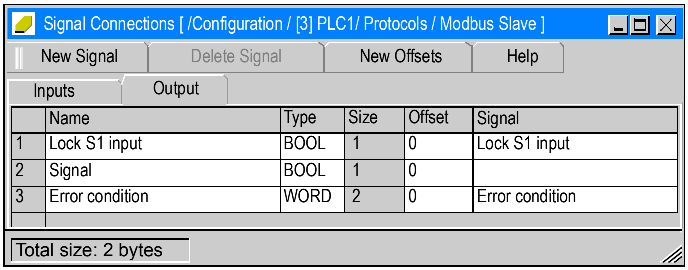


Case 1 shows 2 data types, with BOOL (1 BYTE) and WORD (2 BYTES).

As the signals are packed within a 2 byte packet, when reading this information, the boolean value would be correct, however, there will be an issue with the start address of the WORD as the first half of the signal is packed with the boolean signal, and the second half is packed within a second address.

To overcome this at the Master side is possible with programming, however it is much simpler to organize the above condition in the following way.

Case 1 fixed



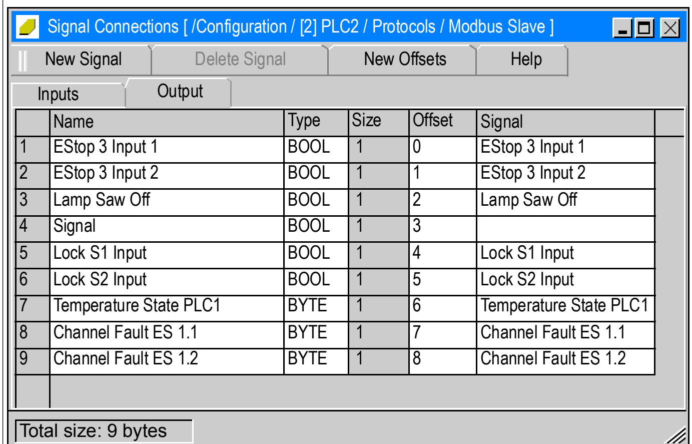
Now, the WORD will be received within 1 address without any problems.

The blank signal can remain empty for future use.

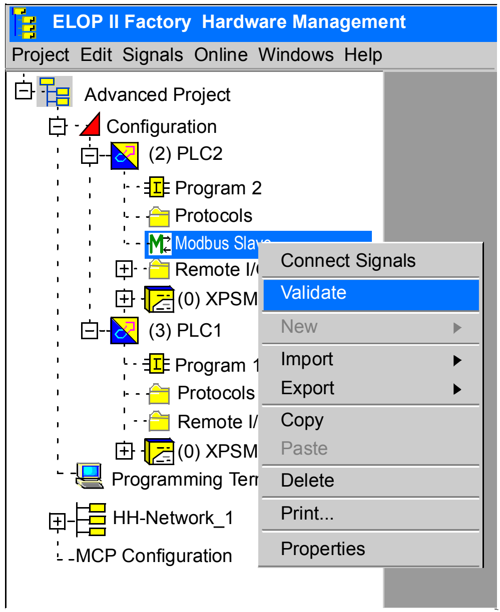
To optimize data transfer, it is advisable that boolean signals are packed together within a word and transmitted, as the boolean signals are only using 1/8th of the available space within the telegram.

This can be done within the **Project Management** area of the software.

New Offset and Validate

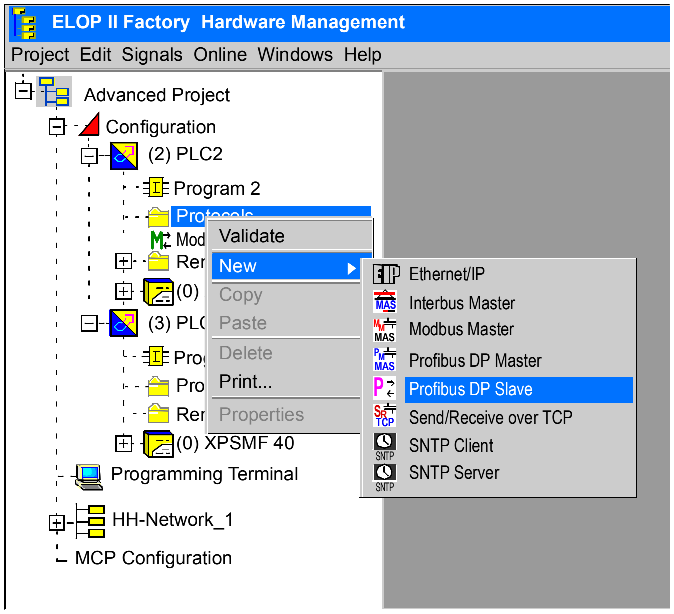
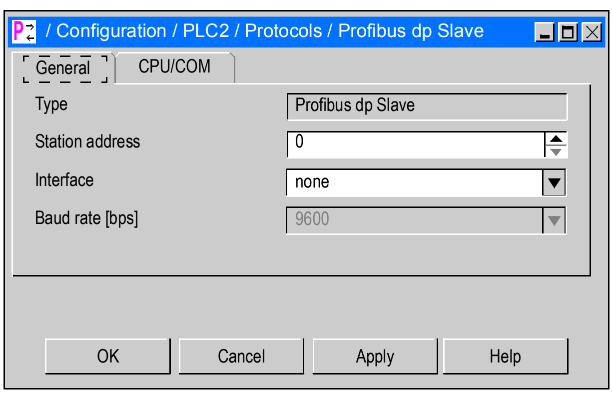
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>After all the required signals have been inserted select the New Offsets button. This automatically sets all of the addresses

</td></tr>
<tr><td>2</td><td>Repeat the above, for the inputs from the Modbus master.
After all of the required signals have been set within the Modbus slave close the window.</td></tr>
</table>

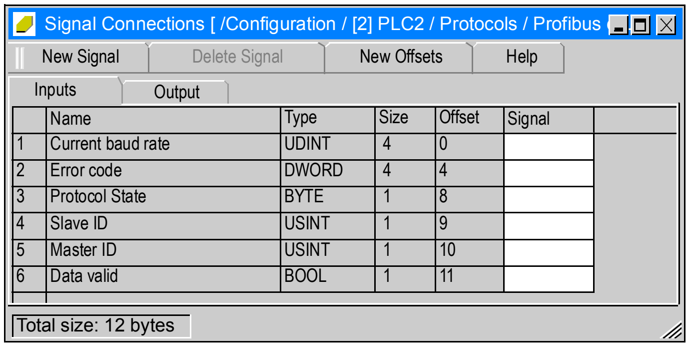
Step	Action

3	To ensure that the Modbus slave has been correctly configured, right-click on the Modbus slave and select **Validate**.
	
	
	Result: Information will be provided within the **Error-State-Viewer**:
	`02/10/2007 18:42:03, 149, Info: [Modbus Slave] Validate started`
	`02/10/2007 18:42:03, 149, Info: [Modbus Slave] Validate finished. Warnings: 0, Errors: 0.`

Profibus DP Slave

Configuration

Step	Action
1	From the **Hardware Management** right-click on **Protocols** of **PLC2** and select **New** → **Profibus DP Slave** from the context menu. 
2	Right-click on **Profibus DP Slave** and select **Properties** from the context menu. **Result:** 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>Enter a Station address and the relevant Interface on the safety PLC FB 1, FB 2 or FB 3.</td></tr>
<tr><td>4</td><td>Click Apply and adjust the Baud rate according to the Profibus DP master requirements.</td></tr>
<tr><td>5</td><td>Within the CPU/COM tab, adjust the refresh rate according to the application requirements and confirm with OK.</td></tr>
<tr><td>6</td><td>Connect signals to the Input and Output areas in the same way as with the Modbus Slave.
Open the Signal Editor and drag and drop the required signals to the Input and Output areas of the Profibus DP slave.


<table>
<tr><th></th><th>Name</th><th>Type</th><th>Size</th><th>Offset</th><th>Signal</th></tr>
<tr><td>1</td><td>Current baud rate</td><td>UDINT</td><td>4</td><td>0</td><td></td></tr>
<tr><td>2</td><td>Error code</td><td>DWORD</td><td>4</td><td>4</td><td></td></tr>
<tr><td>3</td><td>Protocol State</td><td>BYTE</td><td>1</td><td>8</td><td></td></tr>
<tr><td>4</td><td>Slave ID</td><td>USINT</td><td>1</td><td>9</td><td></td></tr>
<tr><td>5</td><td>Master ID</td><td>USINT</td><td>1</td><td>10</td><td></td></tr>
<tr><td>6</td><td>Data valid</td><td>BOOL</td><td>1</td><td>11</td><td></td></tr>
</table>
Note: The Profibus DP slave contains preset diagnostics information within the Signals Connections input area. To use any of this information within your program, simply create new signals with the same data type and connect them to the Signal area of each function.</td></tr>
<tr><td>7</td><td>When complete, validate the configuration to ensure the Profibus DP slave has been configured correctly.</td></tr>
</table>

8.5 Step 4: Setting Hardware I/O

At a Glance

Overview

All compact safety PLC systems and safety remote I/O modules, listed below, have fixed I/Os, which are predefined when the PLC or remote I/O is assigned.

Compact safety PLC systems
- XPSMF3022
- XPSMF31222
- XPSMF3502/F3522/F3542
- XPSMF4000/F4002/F4020/F4022/F4040/F4042

Safety remote I/O modules
- XPSMF1DI1601
- XPSMF2DO801/DO802/DO1601/DO1602
- XPSMF3DIO8801/DIO16801/DIO20802
- XPSMF3AIO8401

The modular safety PLC (XPSMF60) I/Os must be defined depending on which module cards are used.

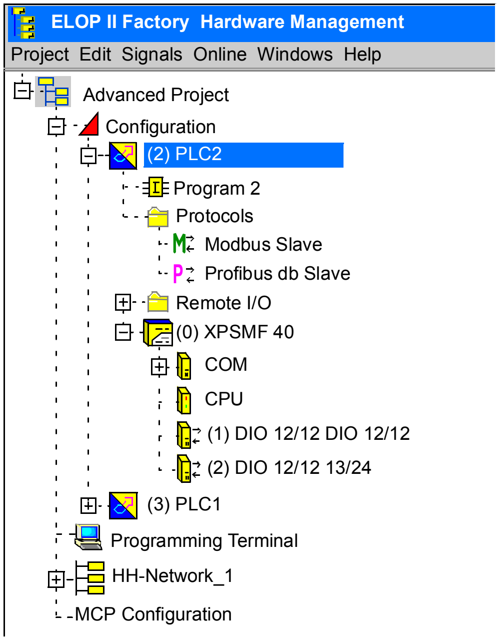
In this advanced project only the compact range is explained, for I/O configuration on the modular safety PLC refer to the basic project configuration.

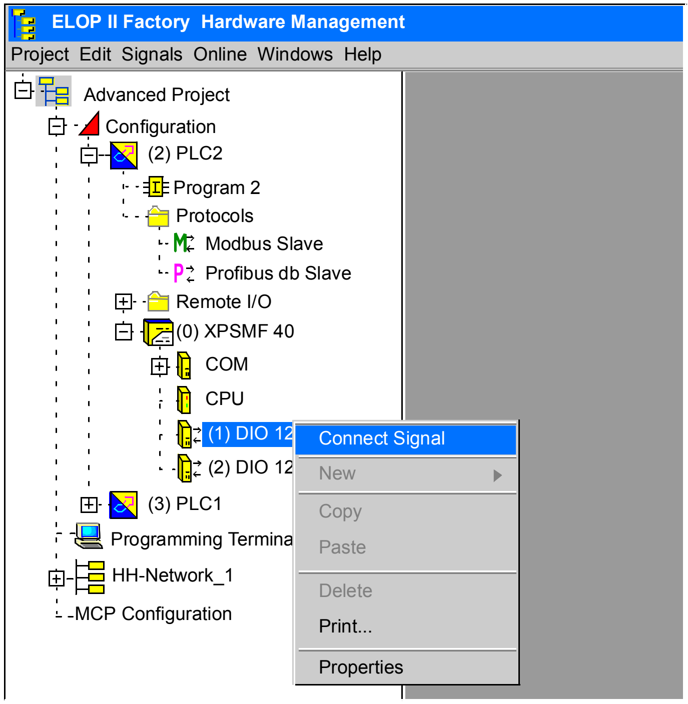
What's in this Section?

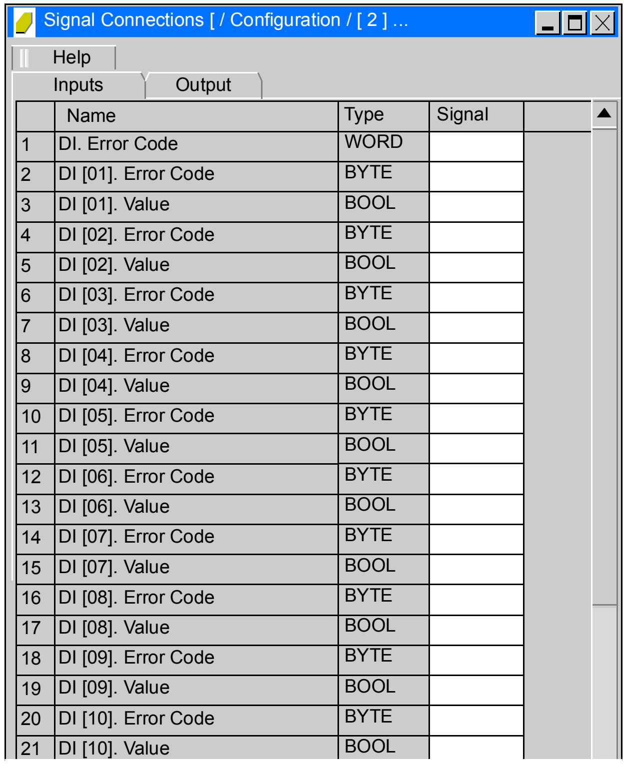
This section contains the following topics:

Configuring I/Os on Compact Safety PLCs and Remote I/O

Configuration

Step	Action
1	In the **Hardware Management** window, expand the folder containing **PLC1**, and then the **XPSMF40** (or relevant PLC reference). 

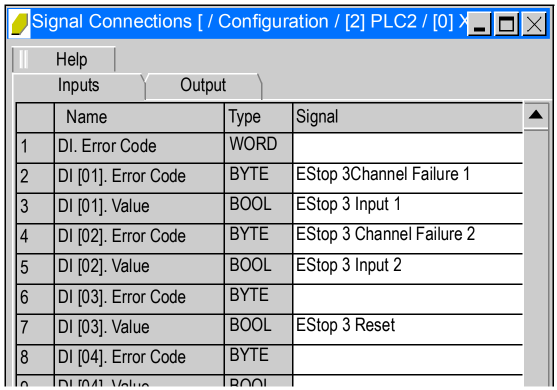
Step	Action
2	Right-click on the input/outputs **(1) DIO 12/12** of the resource and select **Connect Signal**. 

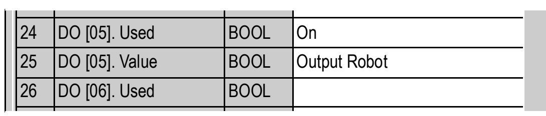
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>The XPSMF40**´s Signal Connections window contains 2 tabs, 1 for Inputs and 1 for Outputs.


<table>
<tr><th></th><th>Name</th><th>Type</th><th>Signal</th></tr>
<tr><td>1</td><td>DI. Error Code</td><td>WORD</td><td></td></tr>
<tr><td>2</td><td>DI [01]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>3</td><td>DI [01]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>4</td><td>DI [02]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>5</td><td>DI [02]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>6</td><td>DI [03]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>7</td><td>DI [03]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>8</td><td>DI [04]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>9</td><td>DI [04]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>10</td><td>DI [05]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>11</td><td>DI [05]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>12</td><td>DI [06]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>13</td><td>DI [06]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>14</td><td>DI [07]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>15</td><td>DI [07]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>16</td><td>DI [08]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>17</td><td>DI [08]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>18</td><td>DI [09]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>19</td><td>DI [09]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>20</td><td>DI [10]. Error Code</td><td>BYTE</td><td></td></tr>
<tr><td>21</td><td>DI [10]. Value</td><td>BOOL</td><td></td></tr>
</table>
Note: The XPSMF40** safety PLC contains configurable I/O terminals. As a default to prevent a dangerous event occurring, all terminals are inputs, until they are defined as outputs.</td></tr>
<tr><td>4</td><td>To connect the signals which you have created to the I/O of the PLC, open the Signals Editor from the menu and arrange the Hardware I/O and Signals Editor side by side.</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>5</td><td>Drag across the following signals into the respective value fields:

EStop 3 Input 1 → DI(01)
EStop 3 Input 2 → DI(02)
EStop 3 Channel Failure 1 → DI(01).Error Code
EStop 3 Channel Failure 2 → DI(02).Error Code
EStop 3 Reset → DI(03).Value



</td></tr>
<tr><td>6</td><td>Select the Output tab from the Signal Connections window and drag the Output Robot signal to DO(05).Value.</td></tr>
<tr><td>7</td><td>To activate an output on the XPSMF40 safety PLC, set the output to ON. To turn an output on, create a new signal within the Signal Editor with the Name ON, select the constant attribute and give it an initial value of 1.</td></tr>
<tr><td>8</td><td>Drag this signal into the DO(05).Used field.


Note: Defining the output as ON in the used column is only used for the XPSMF40 safety PLC as the terminals are configurable as inputs or outputs.</td></tr>
<tr><td>9</td><td>Connect the remaining signals to remote I/O and PLC I/O terminals after executing the steps described in Line Control Signal Assignment, p. 406.</td></tr>
</table>

Line Control Signal Assignment

Overview

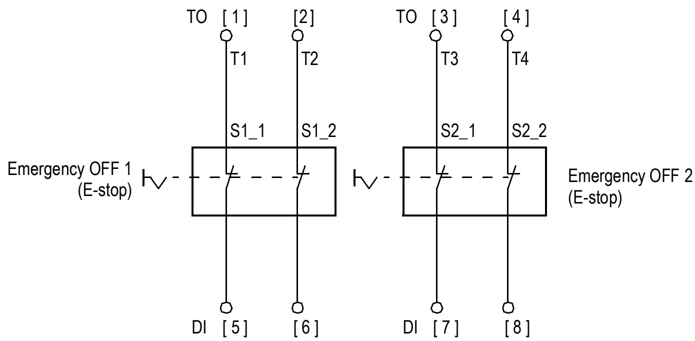
Line control signal assignment is used for short circuit and line break monitoring.

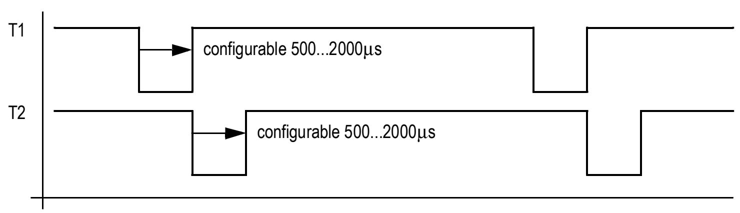
To reach SIL 3 according to EN/IEC 61508, Category 4 according to EN 954-1 you must use this feature.

Connection

Line control outputs should be connected to the digital inputs of the same system.

The following diagram shows connection of digital outputs and digital inputs.

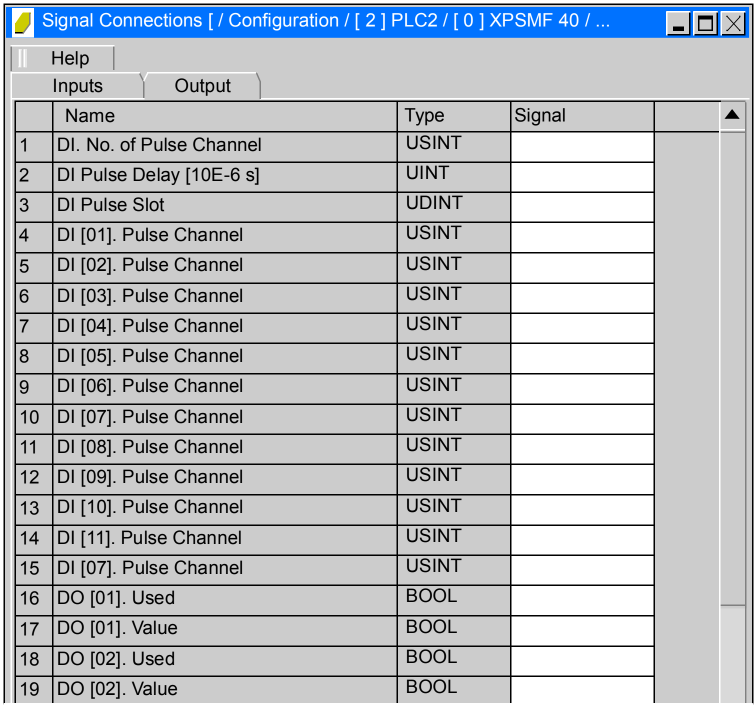




TO Line Control Output
DI Digital Input

Requirements

Click on the **Output** tab from the **(1) DIO 12/12** of the XPSMF40 safety PLC.



The first 3 signals are constants:

Signal	**Description**
DI No. of Pulse Channel	number of digital inputs which use line control
DI Pulse Delay (10E-6 s)	between 500 and 2000 microseconds dependant on the length of cable (normal 500 microseconds)
DI Pulse Slot	where the line control output is originating from slot 1, 2 or 3.

Identify what you require:

Signal	**Requirement**
DI No. of Pulse Channel	Count the number of inputs which require line control signals. In your example you will use only 2 for **EStop 3 Input 1** and **EStop 3 Input 2**.
DI Pulse Delay (10E-6 s)	This is typically found by experience, however within a typical application the pulse delay should be anywhere between 500 and 1500 microseconds. In the example above you will use 1000 microseconds.
DI Pulse Slot	Use the following table to help you identify which line control outputs slot you are using depending on hardware. For your example use slot 1.

Hardware/Slot

Line control outputs slot depending on hardware

Hardware	Slot
XPSMF40**	for terminals 1-12: slot 1 for terminals 13-24: slot 2
XPSMF3022	outputs configurable as line control: slot 2
XPSMF31222	outputs configurable as line control: slot 2
modular PLC	varies depending on location of I/O card XPSMFDIO241601
XPSMF1DI1601 (remote I/O)	slot 1
XPSMF3DIO8801 (remote I/O)	slot 3
XPSMF3DIO16801 (remote I/O)	first line control output slot 1 second line control output slot 2
XPSMF3DIO20802 (remote I/O)	slot 2

Activating Line Control

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Create 3 new signals within the hardware management Signal Editor.
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Retain</th><th>Const.</th><th>Description</th><th>Init Value</th></tr>
<tr><td>56</td><td>Number of Line control inputs</td><td>USINT</td><td></td><td>✓</td><td>Number of Line control inputs F40</td><td>2</td></tr>
<tr><td>57</td><td>Pulse delay</td><td>UINT</td><td></td><td>✓</td><td>Pulse delay time</td><td>1000</td></tr>
<tr><td>58</td><td>Slot</td><td>UDINT</td><td></td><td>✓</td><td>area where the pulse is originating</td><td>1</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>On the Output tab from the (1) DIO 12/12 of the XPSMF40 safety PLC drag and drop from the Signal Editor the signals into the appropriate I/O slots:
Signal Connections [/ Configuration / [2] PLC2 / [0] XPSMF 40 / ...
Help
Inputs | Output
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Signal</th></tr>
<tr><td>1</td><td>DI. No. of Pulse Channel</td><td>USINT</td><td>Number of Line</td></tr>
<tr><td>2</td><td>DI Pulse Delay [10E-6 s]</td><td>UINT</td><td>Pulse Delay</td></tr>
<tr><td>3</td><td>DI Pulse Slot</td><td>UDINT</td><td>Slot</td></tr>
<tr><td>4</td><td>DI [01]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>5</td><td>DI [02]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>6</td><td>DI [03]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>7</td><td>DI [04]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>8</td><td>DI [05]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>9</td><td>DI [06]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>10</td><td>DI [07]. Pulse Channel</td><td>UDINT</td><td></td></tr>
<tr><td>11</td><td>DI [08]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>12</td><td>DI [09]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>13</td><td>DI [10]. Pulse Channel</td><td>USINT</td><td></td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>
To define different pulses for each of the line control inputs, you need to define new signals within the Signal Editor.
The convention should be as follows:

Pulse1: USINT value 1
Pulse2: USINT value 2
Pulse3: USINT value 3
Pulse4: USINT value 4
Pulse5: USINT value 5
Pulse6: USINT value 6
Pulse7: USINT value 7
Pulse8: USINT value 8

The name is user defined, however the values must be incremental according to the number of line control outputs available from the safety PLC or remote I/O.
Within the XPSMF40 safety there are 2 sets of 4 line control outputs.
Therefore numbering can only be from 1 to 4 on each of the line control terminals, e.g.

(1) DIO 1/12: line control signal values 1 to 4
(2) DIO 13/24: line control signal values 1 to 4

</td></tr>
<tr><td>4</td><td>
In the example above, you only require 2 different line control signals, therefore create the following 2 new signals:
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Retain</th><th>Const.</th><th>Description</th><th>Init Value</th></tr>
<tr><td>59</td><td>P1</td><td>USINT</td><td></td><td>✓</td><td>Pulse signal 1</td><td>1</td></tr>
<tr><td>60</td><td>P2</td><td>USINT</td><td></td><td>✓</td><td>Pulse signal 2</td><td>2</td></tr>
</table>
</td></tr>
<tr><td>5</td><td>
On the output tab from the (1) DIO 12/12 of the XPSMF40 safety PLC drag and drop from the Signal Editor the 2 signals DI (01). Pulse Channel and DI(02).Pulse Channel.
Signal Connections [/ Configuration / [2] PLC2 / [0] XPSMF 40 / ...
Help
Inputs | Output
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Signal</th></tr>
<tr><td>1</td><td>DI. No. of Pulse Channel</td><td>USINT</td><td>Number of Line</td></tr>
<tr><td>2</td><td>DI Pulse Delay [10E-6 s]</td><td>UINT</td><td>Pulse Delay</td></tr>
<tr><td>3</td><td>DI Pulse Slot</td><td>UDINT</td><td>Slot</td></tr>
<tr><td>4</td><td>DI [01]. Pulse Channel</td><td>USINT</td><td>P1</td></tr>
<tr><td>5</td><td>DI [02]. Pulse Channel</td><td>USINT</td><td>P2</td></tr>
<tr><td>6</td><td>DI [03]. Pulse Channel</td><td>USINT</td><td></td></tr>
<tr><td>7</td><td>DI [04]. Pulse Channel</td><td>USINT</td><td></td></tr>
</table>
This now links pulse value 1 (P1) to EStop 3 Input 1 and pulse value 2 (P2) to EStop 3 Input 2.
Note: In this case the 2 pulses must be of different values as you are supplying the input of 2 emergency stop contacts of the same emergency stop control. In order to detect any short circuit condition, line break or wrongly wired connection you must use 2 different signals.
</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>To activate the line control outputs, you need to turn on 2 of the 4 line control outputs.
These outputs are listed within the Output tab from the (1) DIO 12/12 of the XPSMF40 safety PLC with the following names:

TO(01).Value
TO(02).Value
TO(03).Value
TO(04).Value
</td></tr>
<tr><td>7</td><td>You will use only TO(01) and TO(02).

TO(01) provides the pulse for P1
TO(02) provides the pulse for P2

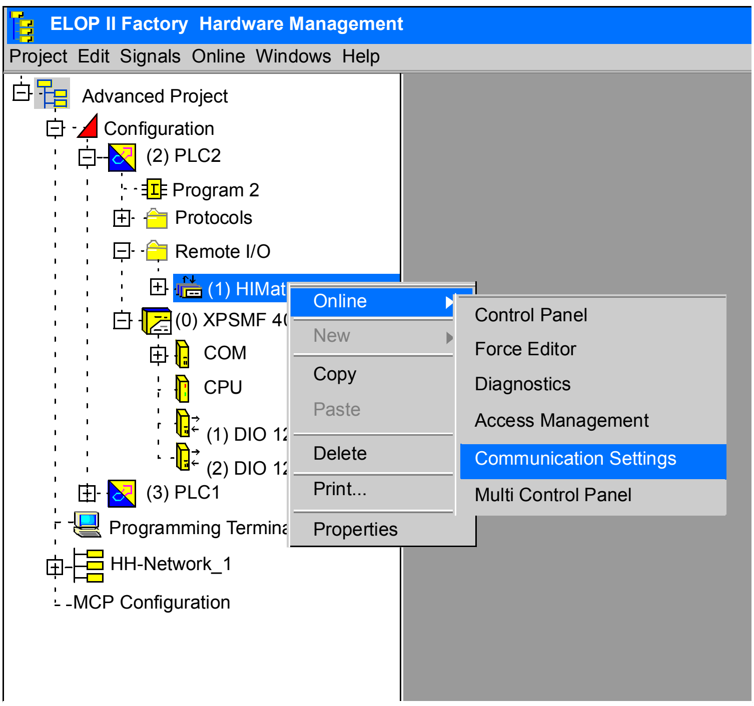
To activate the line control use the signal ON from the Signal Editor and drag this into T0(01).Value and TO(02).Value slots.
<table>
<tr><th></th><th>Name</th><th>Type</th><th>Signal</th></tr>
<tr><td>40</td><td>TO [01]. Value</td><td>BOOL</td><td>ON</td></tr>
<tr><td>41</td><td>TO [02]. Value</td><td>BOOL</td><td>ON</td></tr>
<tr><td>42</td><td>TO [03]. Value</td><td>BOOL</td><td></td></tr>
<tr><td>43</td><td>TO [04]. Value</td><td>BOOL</td><td></td></tr>
</table>
Once done, the line control monitoring has been set up.The line control monitoring is hardware dependant, and therefore does not require any settings within the Project Management area of the software.</td></tr>
<tr><td>8</td><td>Finish assigning all the remaining signals onto the hardware I/O and close any open panes within the software.</td></tr>
</table>

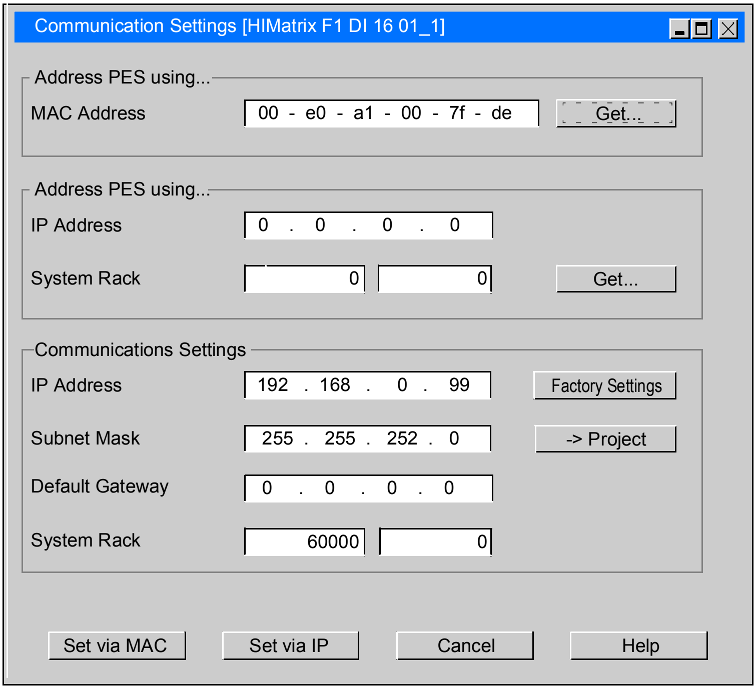
Setting PLC Communication

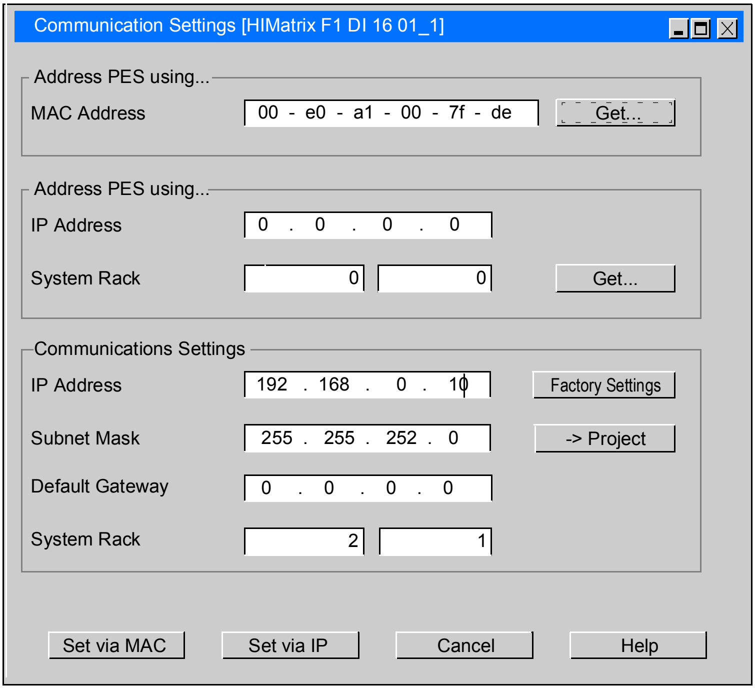
Communication Settings

Set the PLC communication as described in chapter *Basic Project Configuration*, *Step 8: Setting PLC Communication, p. 314*.

Setting Remote I/O

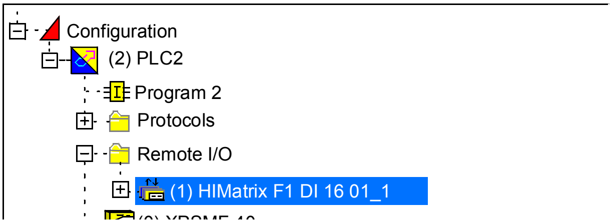
Step	Action
1	Connect the network card of the programming terminal to one of the Ethernet ports of the safety PLC, remote I/O, or network switch. **Note:** You do not need to make a 1 to 1 connection with the safety PLC or remote I/O you are working with. The connection can be made via a Ethernet network switch, wireless, etc..
2	Right-click on the **Remote I/O** → **Online** → **Communication Settings**. 

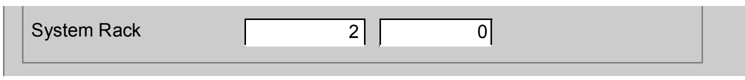
Step	Action
3	Enter the **MAC Address** of the remote I/O in the **Communication Settings** window. (MAC address located either on the front panel or next to the switched Ethernet port on the remote I/O) and click **Get**.  **Result:**The default **IP Address**, **Subnet Mask** and **System Rack** are received.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>Enter a new IP Address and Subnet Mask within the Communications Settings area


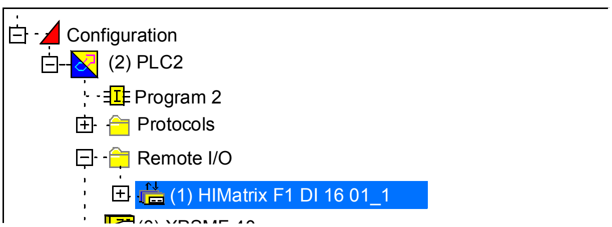
</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>5</td><td>
Enter now the System Rack.
For the safety PLCs the system rack is the same as the SRS (originally defined within the hardware properties) and can be seen to the left of the PLC name.

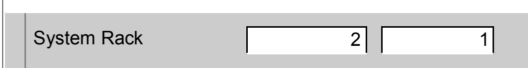




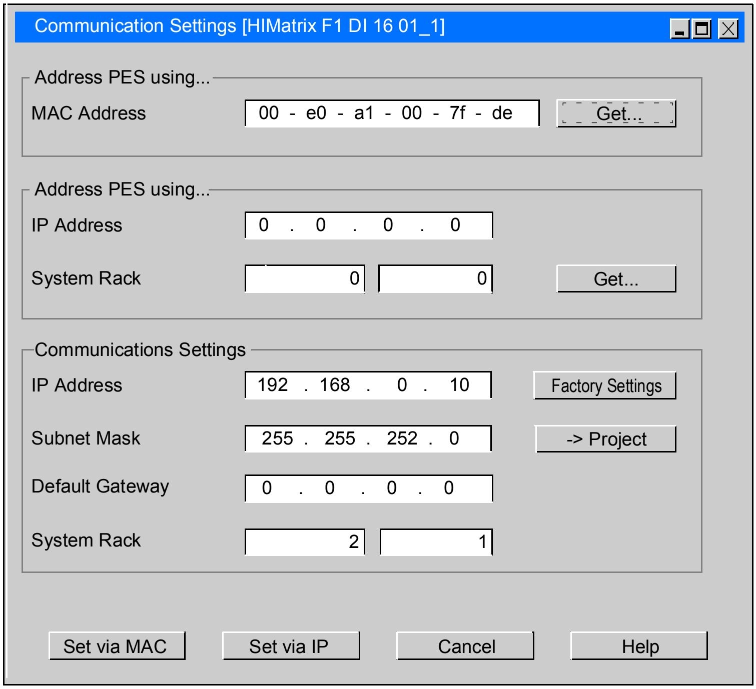
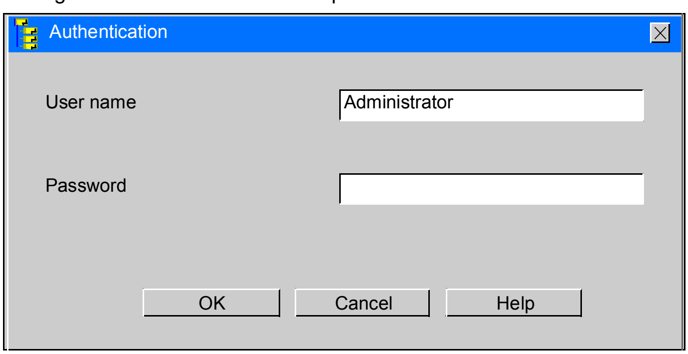
The remote I/O are set in a similar way however, with a remote I/O you must also define the parent PLC. Therefore, with our example we have defined the system rack of the remote I/O as 1. Its' parent PLC is SRS value 2



To assign the remote I/O you must have the following configuration:



</td></tr>
</table>

Step	Action
6	Click on **Set via MAC** 
7	By asking for the authentication enter a user name with administrator rights and a password. The factory setting is **Administrator** without a password.  **Note:** See the message in the Error-State-Viewer for successful data change.
8	Click on **Project** and select the respective resource to apply the communication settings to the project. **Note:** To ensure the PLC communication settings are the same as set in the hardware and software, set the IP address to the project by clicking the **Project** button.

8.6 Step 5: Access Administration

Access Administration

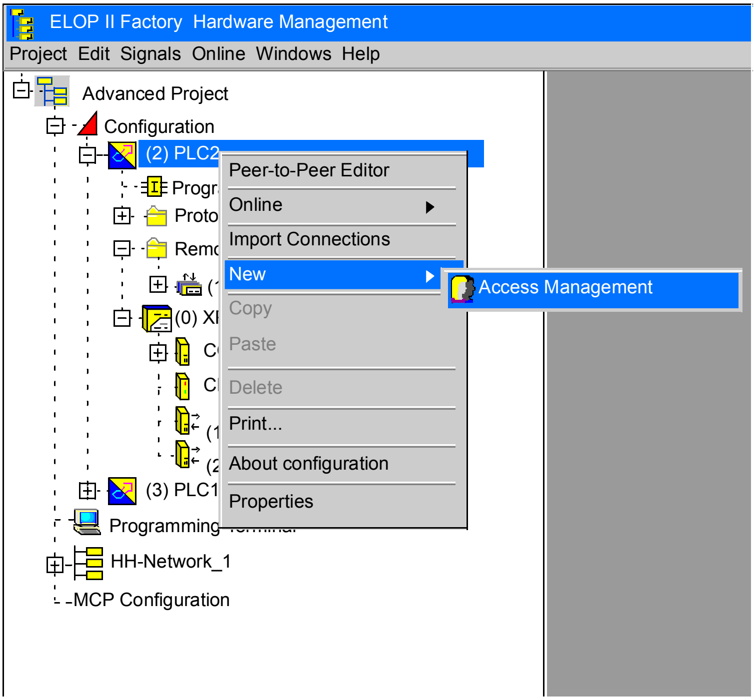
Overview

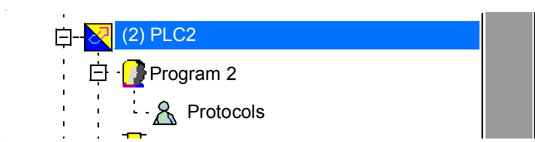
The access management enables a system engineer to set and manage access rights for a maximum of 10 users per controller. The access rights are stored in the buffered NVRAM of the controller and remain there even after disconnecting the operating voltage.

Note: Downloading new user accounts overwrites the user accounts previously stored on the controller or cleans the **Administrator** standard setting.

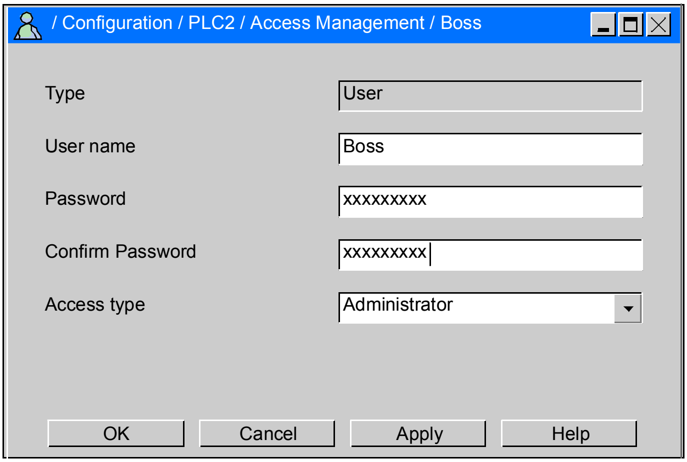
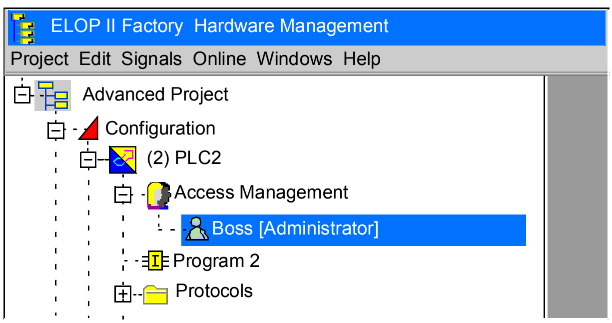
This also applies to user accounts generated online with **Online** → **Access Management**.

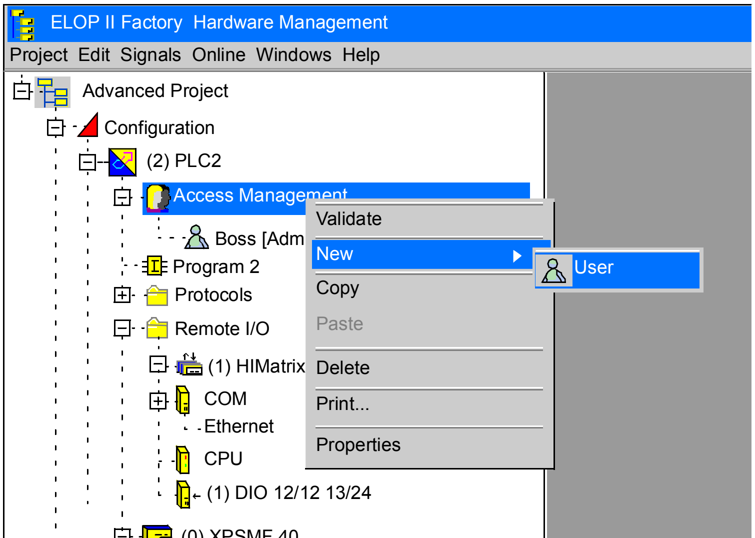
Procedure Each resource (PLC) must be set up individually.

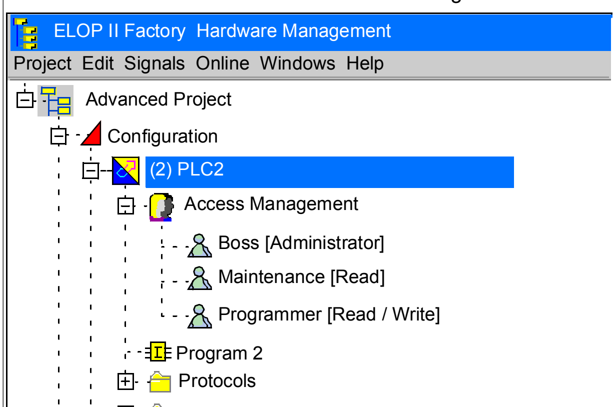
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>To assign access administration, right-click on PLC2 from the Configuration → New → Access Management.


</td></tr>
<tr><td>2</td><td>The Access Management folder appears under the Resource (PLC2) with a single user in read only mode.


You must begin by setting up a new administrator for the PLC. When there is no administrator you will not have complete access to the PLC!</td></tr>
<tr><td>3</td><td>Right-click on the default User (Read) → Properties.</td></tr>
</table>

Step	Action
4	Enter a **User name**, **Password** and **Access type** (Administrator). 
5	Select **Apply** and **OK**. This will now appear within the place of the default user. 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>Enter now 2 users, one with Read access and one with Read/Write access by right-clicking on Access Management → New → User and filling the appropriate data.


Your window should look similar to the following:


The accounts will be activated on download to the PLC.</td></tr>
<tr><td>7</td><td>Repeat procedure for PLC1.
Note: Remote I/O do not require access management set up, as this is managed via the parent resource (PLC).</td></tr>
</table>

8.7 Step 6: Code Generation and Program Size

At a Glance

Overview

This section describes how to code generate the entire project.

What's in this Section?

This section contains the following topics:

Code Generation

Overview

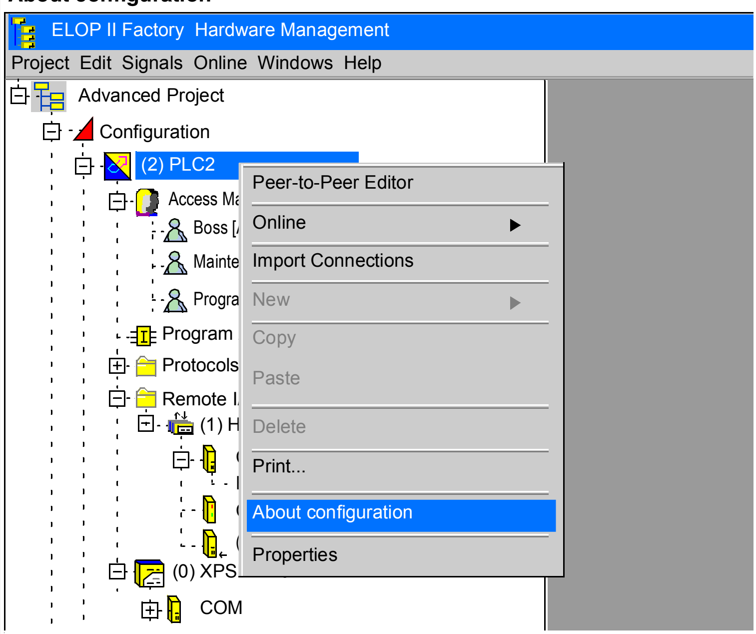
Now that the project is complete, you can generate the code of the entire project.

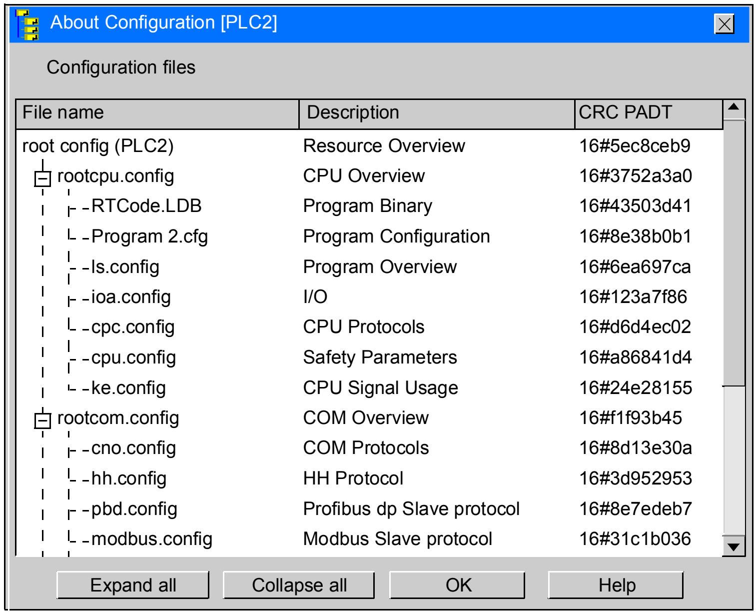
Start Code Generation

Step	Action
1	Using the **Project Management** window right-click on **Configuration**, select **Code Generation** and **Start**.
2	Click on the **Error State Viewer**  icon to view the code generation details. Any errors must be analyzed and resolved, and any warning messages must be reviewed to ensure that a dangerous event cannot occur from the warning message.

Code Generate

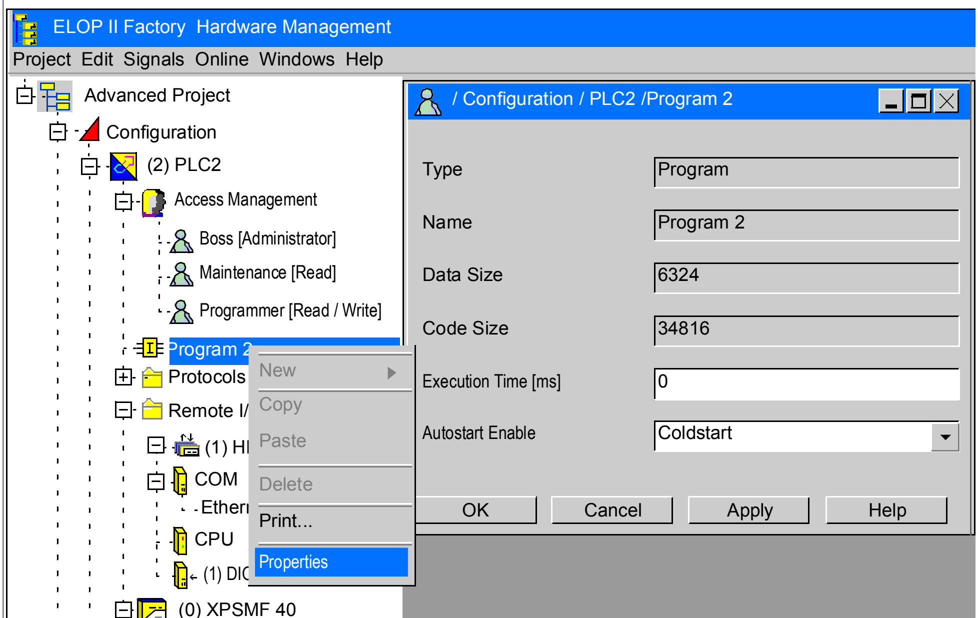
Code generation must be carried out twice in the following way:

Step	Action
1	Check for errors, review warning messages.Right-click on each PLC and select **About configuration** 

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Print the About configuration information for each PLC.

</td></tr>
<tr><td>3</td><td>Code generate 2nd time.</td></tr>
<tr><td>4</td><td>Check for errors, review warning messages.
View About configuration and check that there are no changes between the first generation and the second, for each PLC.</td></tr>
</table>

Program Size

Procedure In the **Hardware Management** window proceed the following steps:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Right-click on the program instances Program 2, and Program 1 for PLC1 and view data and code size for each program.

</td></tr>
<tr><td>2</td><td>Ensure that the programs do not exceed the memory size of the safety PLC, you can view the Data Size and Code Size in Octets (Bytes) within the properties.
For the compact safety PLC range:
data size max: 250 KB
code size max: 250 KB
For the modular safety PLC range:
data size max: 500 KB
code size max: 500 KB</td></tr>
</table>

8.8 Step 7: Online, Download, Run and Online Test

At a Glance

Overview

This section describes how to procede for starting online, run, download and online tests on the PLCs.

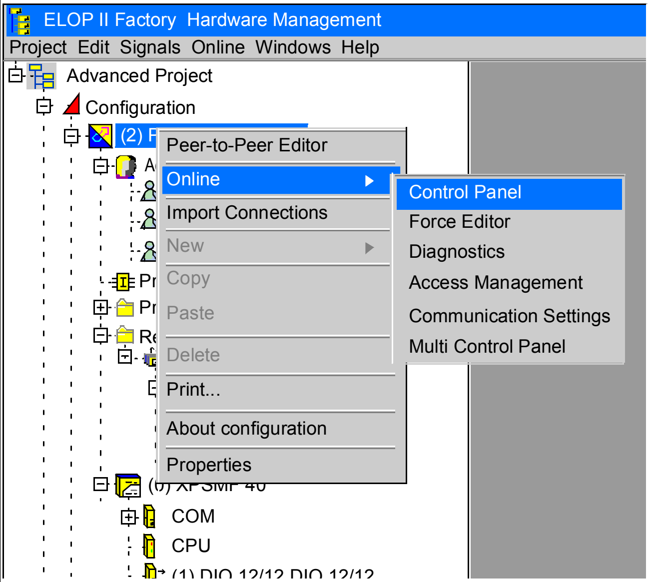
What's in this Section?

This section contains the following topics:

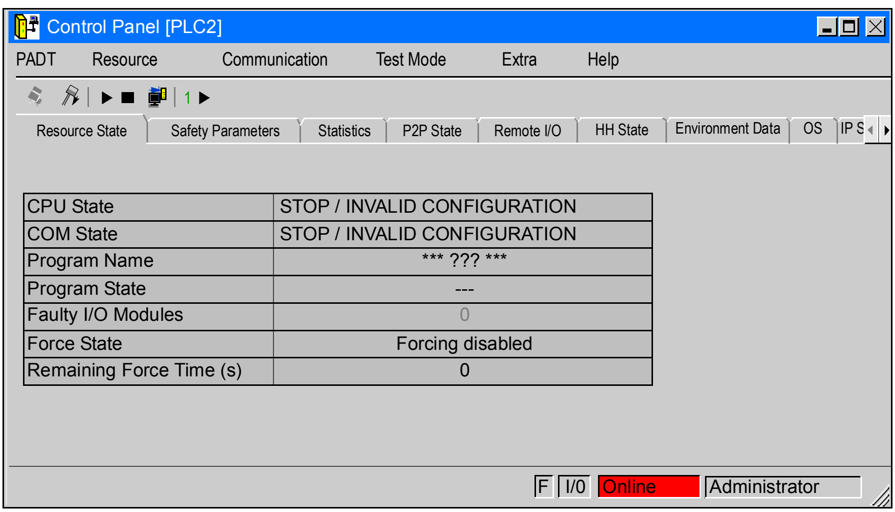
Online, Download and Run

Online

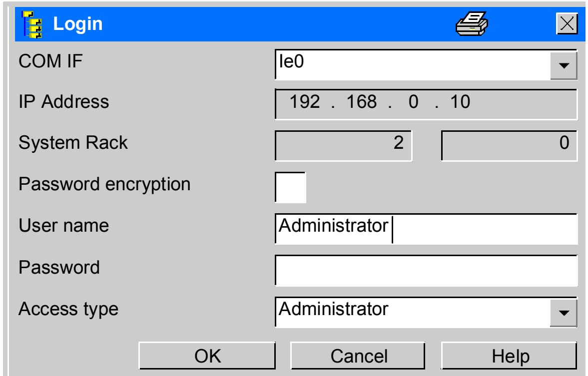
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Right-click on the resource (PLC2 or PLC) and select Online → Control Panel.



Result:



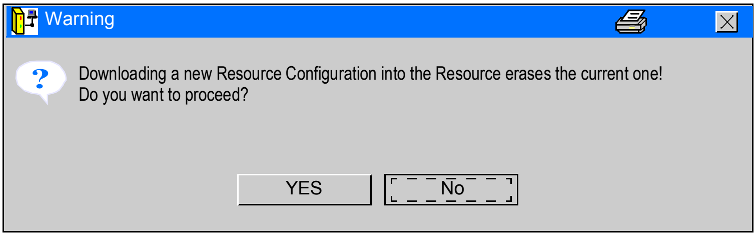
</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Enter the default user access connection by pressing CTRL A.


Note: The new access setting will be activated only on download to the PLC.</td></tr>
</table>

Download

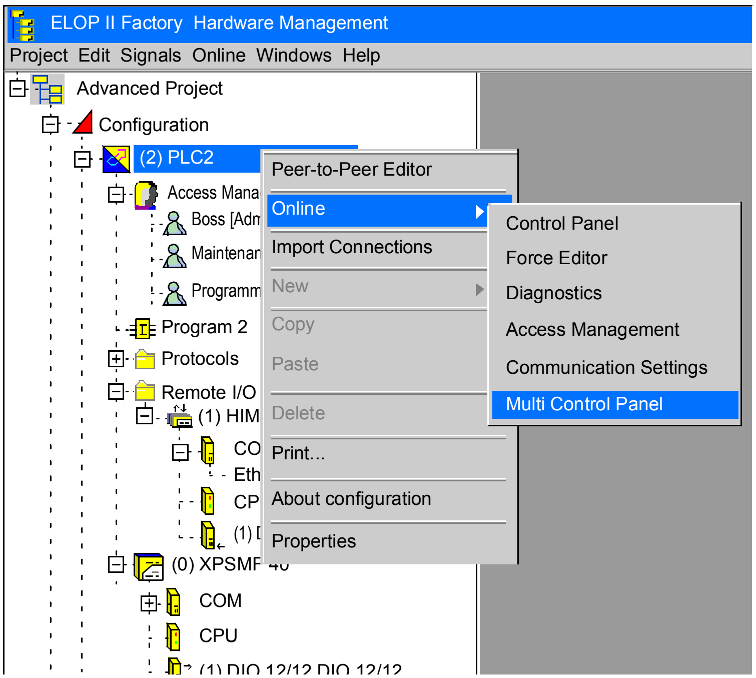
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select the download Icon from the menu bar and select YES in the following window. (Ensure the PLC is in stop mode!)



</td></tr>
<tr><td>2</td><td>When download is okay, the Program Name, CPU State and COM State will provide information regarding the status.

</td></tr>
</table>

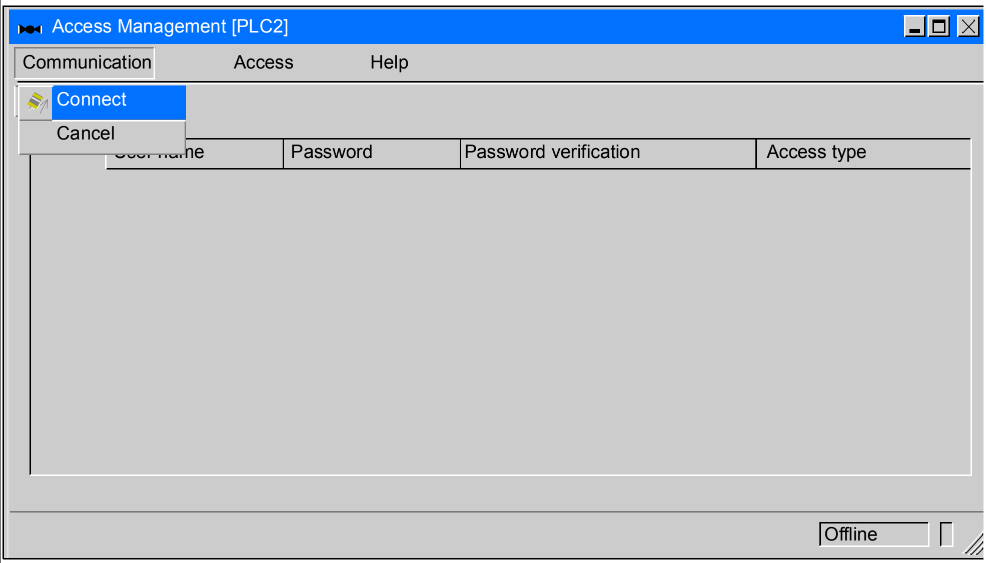
Run

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Set the PLC to run by selecting the icon (ensure that it is not connected to any machine which could cause harm to a user as this is only a configuration example).
Note: Downloading to remote I/O will not function, as their program is within the parent PLC</td></tr>
<tr><td>2</td><td>Repeat this for the second PLC. When there are multiple PLCs and remote I/O, it is simpler to use the Multi Control Panel.
To use the multi control panel, right-click on the resouce (PLC1 or PLC2) and select Online → Multi Control Panel.

</td></tr>
<tr><td>3</td><td>Enter the User name and Password.
Note: If you have already downloaded to each PLC then use the new administration user name and passwords.
The first PLC is automatically placed within the multi control panel.
Drag and drop the second PLC within the same window.
Note: Using this window it is possible to download much quicker, and view overall statistics in a simpler way.
It is also possible using the multi control panel to access the online control panel of each PLC or remote I/O.</td></tr>
</table>

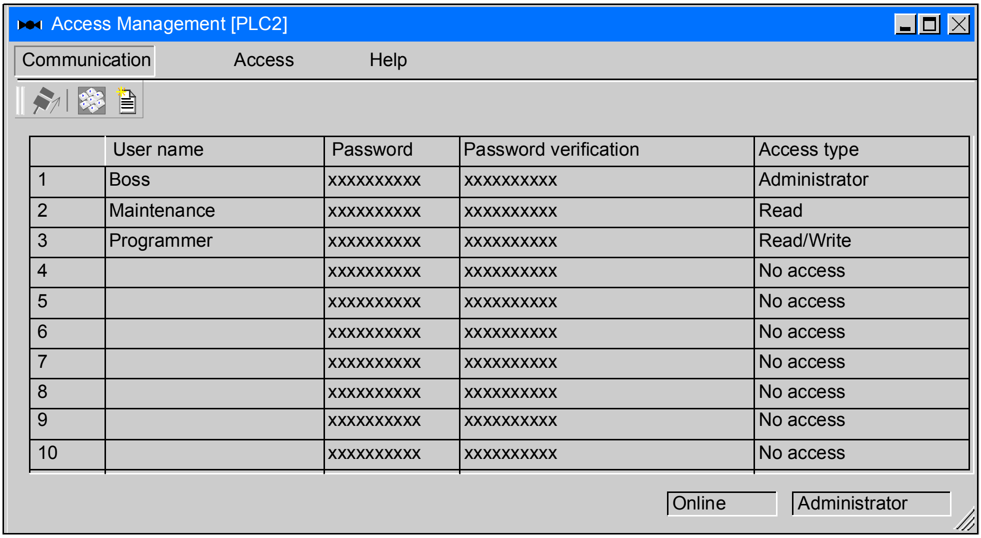
Administration Access

Procedure

Once the administration settings have been downloaded via the program to the PLC, it is possible to connect and view all user data stored in each PLC.

Step	Action
1	Right-click on the resource (**PLC1**) from the **Hardware Management** window and select **Online** → **Access Management**.
2	Within the pop up window select **Communication** → **Connect**. 

Step	Action
3	Enter the **User name** and **Password** for the administrator.



	User name	Password	Password verification	Access type
1	Boss	xxxxxxxxxx	xxxxxxxxxx	Administrator
2	Maintenance	xxxxxxxxxx	xxxxxxxxxx	Read
3	Programmer	xxxxxxxxxx	xxxxxxxxxx	Read/Write
4		xxxxxxxxxx	xxxxxxxxxx	No access
5		xxxxxxxxxx	xxxxxxxxxx	No access
6		xxxxxxxxxx	xxxxxxxxxx	No access
7		xxxxxxxxxx	xxxxxxxxxx	No access
8		xxxxxxxxxx	xxxxxxxxxx	No access
9		xxxxxxxxxx	xxxxxxxxxx	No access
10		xxxxxxxxxx	xxxxxxxxxx	No access

Online Test with Hardware

Overview

The online test allows you to view in a near real time manor the states of all inputs and outputs in the program.

The online test is very similar to the offline simulation except that the hardware is connected and providing real values to the online test.

The online test is very useful for identifying possible errors in coding.

Activate Online Test

Step	Action
1	Ensure that you are connected to the PLC and have the **Online Control Panel** open within the **Hardware Management**.
2	Right-click on the **Project** → **Program 1** or **Program 2** in the **Project Management** window.
3	Select **Online Test**. **Note:** The PLC must be connected and online and the offline simulation must not be running.)
4	Double click on the **Project** in the **OLT** window to open the **Logic**.
5	To change any values within the online test, you must use the force editor (see *Step 8: Force Editor, p. 435*).
6	For more information regarding the online test see *Online Test (Power Flow), p. 99* of this software manual.

8.9 Step 8: Force Editor

Force Editor

⚠ WARNING

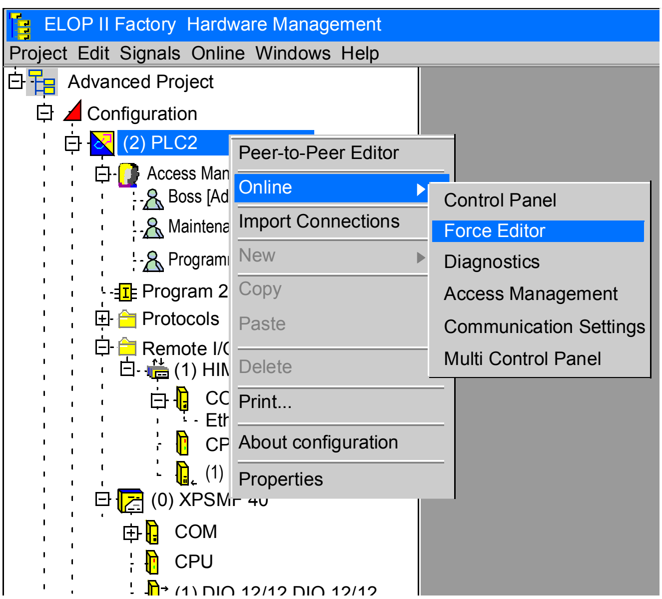
UNINTENDED EQUIPMENT OPERATION

Do not use Forcing during normal operation.
Limit use of Forcing to system commissioning and de-bugging.

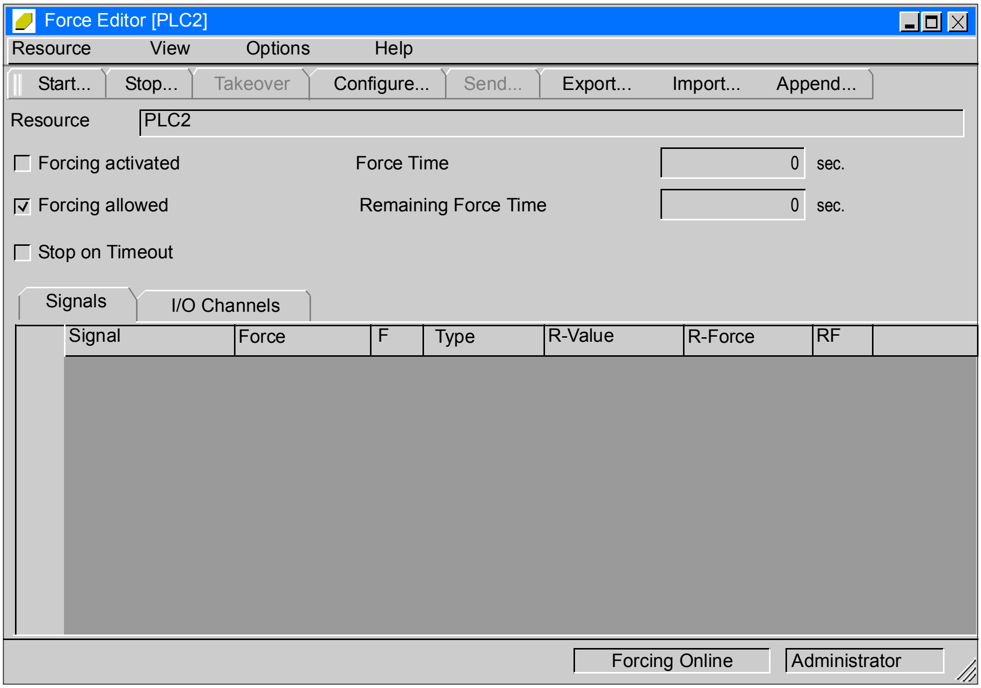
Failure to follow these instructions can result in death, serious injury, or equipment damage.

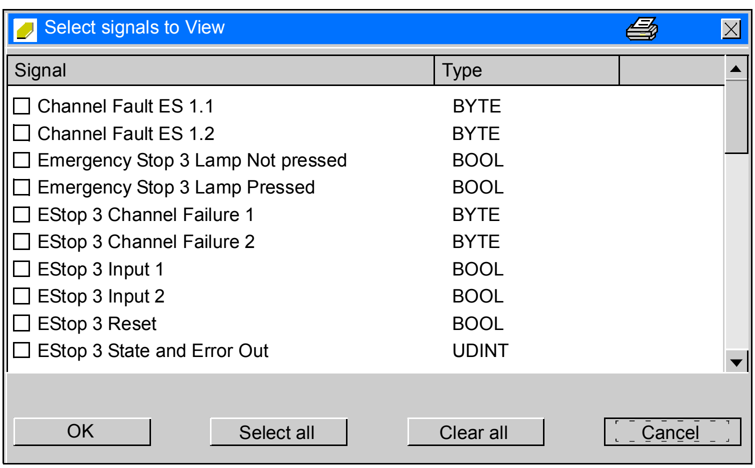
Procedure

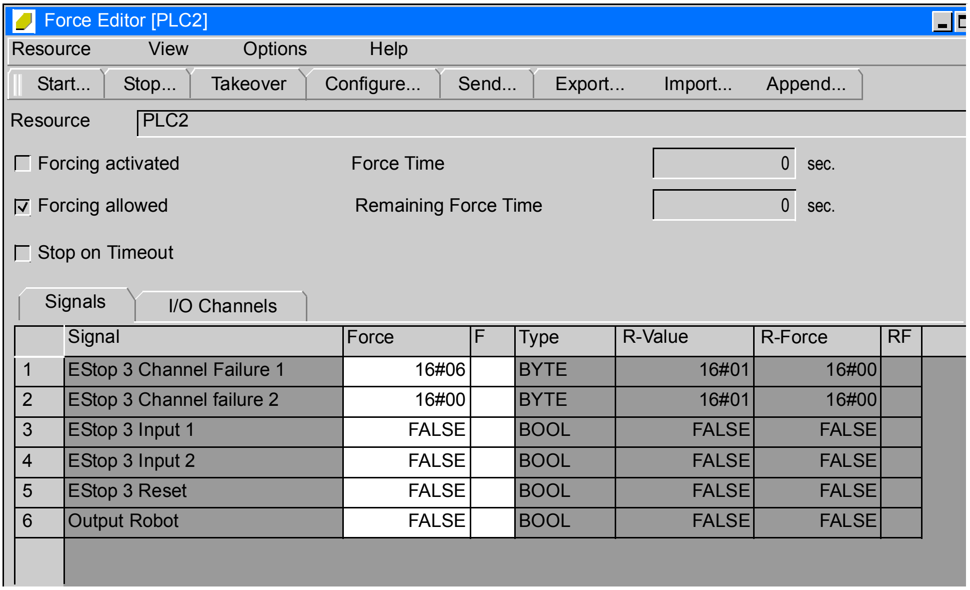
To change values of signals or physical I/O, you must use the force editor. This can be used along side the online test, or alone.

Step	Action
1	To access the force editor, within the **Hardware Management** window from either resource (**PLC1** or **PLC2**) right-click and select **Online** → **Force Editor**.  **Note:**The online control panel must be connected with the PLC(s).

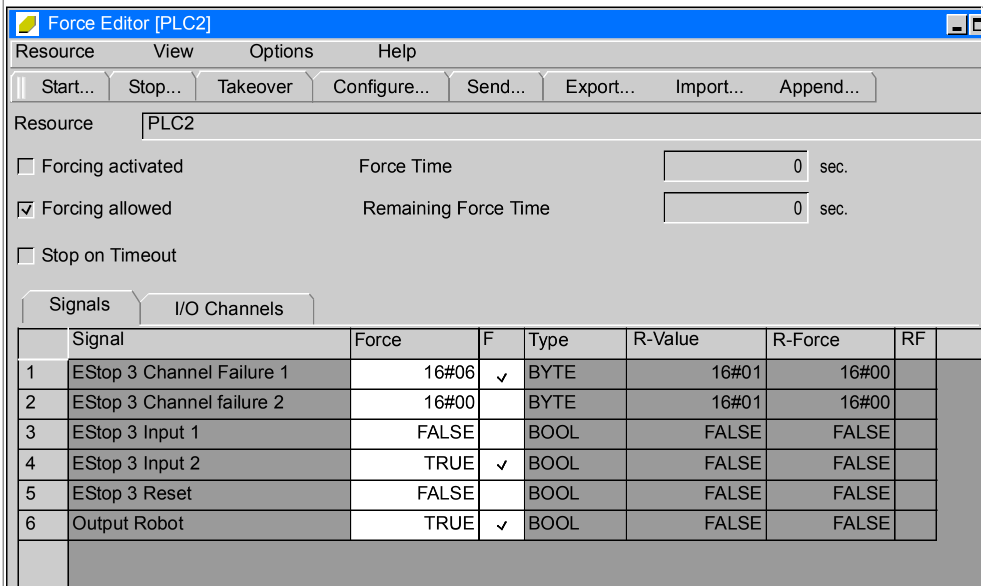
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>2</td><td>Click on **Configure** from the **Force Editor** window.


</td></tr>
<tr><td>3</td><td>Select from the pop up window the signals for forcing.

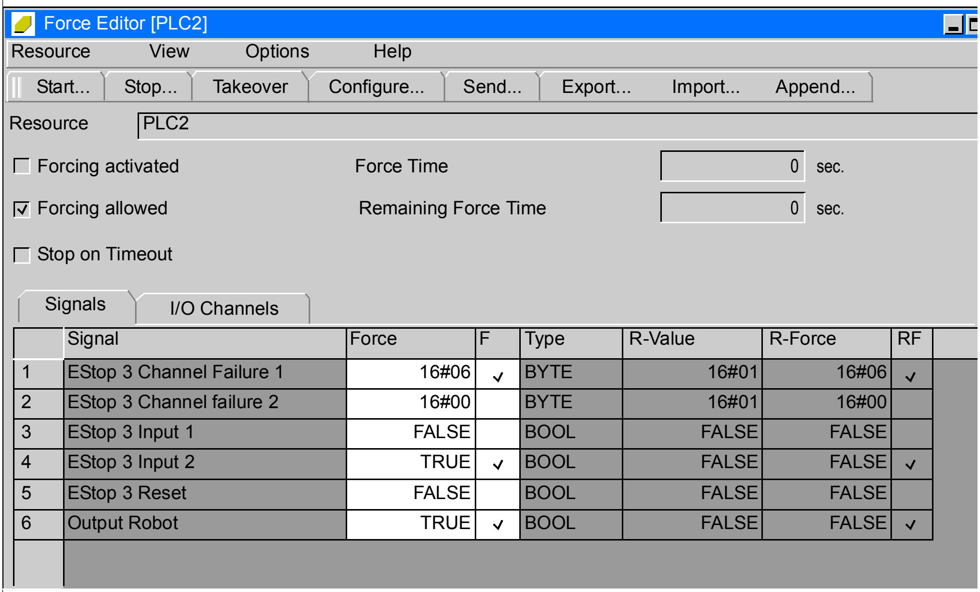

</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>To save the forcing configuration, click on Export and save. To import a configuration which was previously saved click on Import.

</td></tr>
</table>

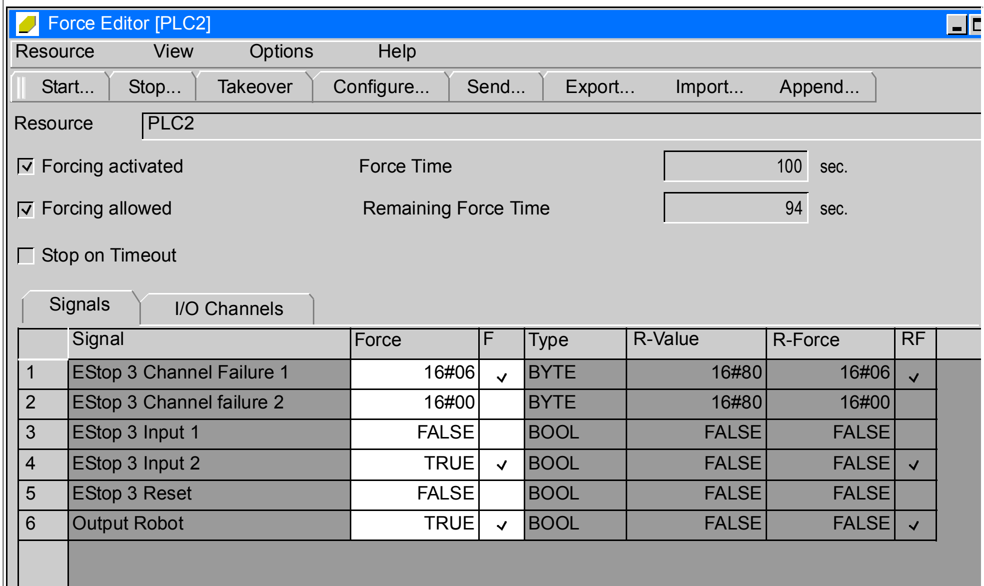
Step	Action
5	To force values enter new values into the **Force** area and select the **F** box.



Step	Action
6	To activate the new force values click on the **Send** button. The values are now ready within the safety PLC.



<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>7</td><td>Start forcing by clicking on Start and enter the duration.



Note: The PLC must be in Run Mode.</td></tr>
</table>

8.10 Step 9: Hot Swapping Remote I/O

Hot Swapping Remote I/O

Overview

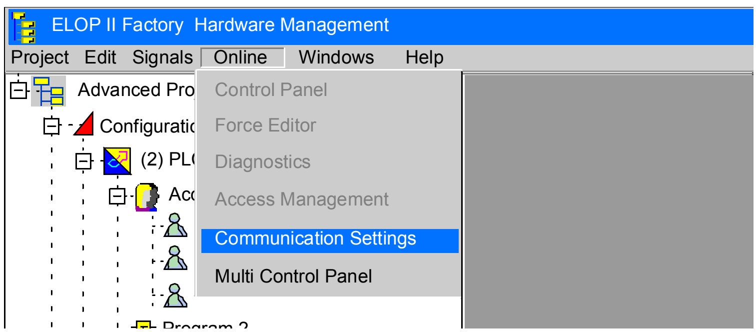
All of the XPSMF remote I/O modules are hot swappable.

If in the unlikely event of a remote I/O failure, the same product can be replaced with a product of the same product reference with out the need to download a program, stop the safety PLC or stop the network.

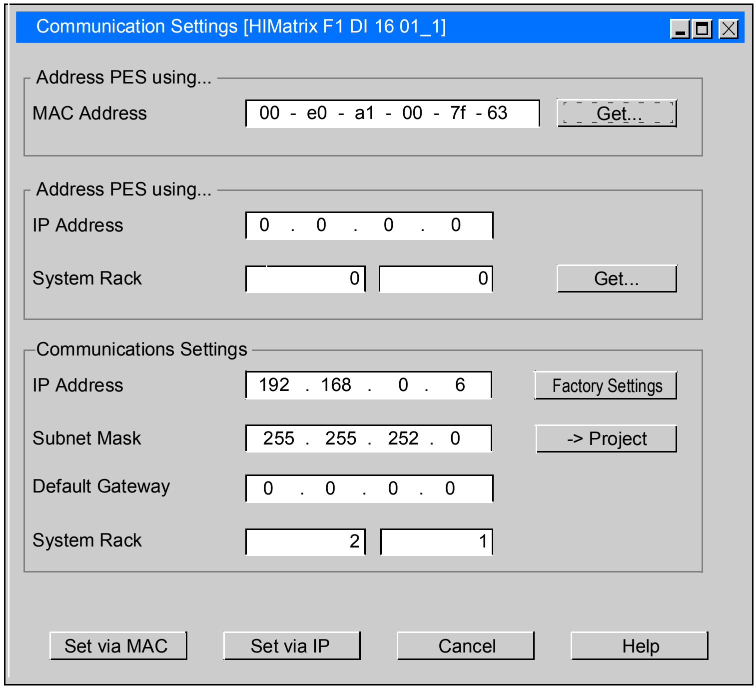
Swap Remote I/O

To swap remote I/O is simple and can be managed by the following steps:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Check the remote I/Os' IP address and SRS settings.
These can be found on the PLC (if written) or within the program application.</td></tr>
<tr><td>2</td><td>Connect the new safety remote I/O to the programming terminal, and within the hardware management of the same program, or a new blank project.</td></tr>
<tr><td>3</td><td>Click Online → Communication Setting from the menu bar.



Or directly on the remote I/O within the configuration right-click and select Online → Communication Settings.</td></tr>
</table>

Step	Action
4	Enter the new MAC address and click **Get**, and set the same IP address and system rack from the faulty hardware. 
5	Click **Set via MAC**.
6	Enter the user name and password (as per your access administration).
7	Disconnect the failed module, with a check to all I/O wiring and simply plug the new product in and after approximately 30 seconds (after boot-up) the parent will find the new remote I/O.

Conclusion

This concludes the advanced project configuration guide.

To ensure that all features from this configuration was understood close the project and create a new project and carry out all of the steps a second time without using the configuration guide.

Additional Features for Programming Areas

9

At a Glance

Overview

This chapter provides information on additional features for programming areas.

What's in this Chapter?

This chapter contains the following sections:

9.1 Importing and Exporting Signals

At a Glance

Overview

Within XPSMFWIN it is possible to import and export all signals and non-safety related protocol communication signals.

Importing and exporting signals is very useful and helpful to reduce the overall programming time for an application.

What's in this Section?

This section contains the following topics:

Creating Signals in Excel

Overview

An Excel based software or similar can be used to create the entire signal list for importing and likewise, for exporting to other automation software.

Correct Layout for Excel Sheet

For a successful import or export the layout of the Excel sheet must be in the following way:

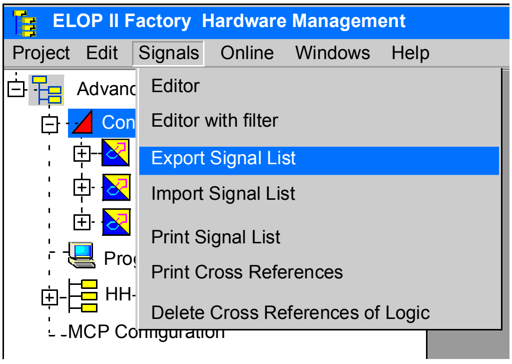
Name	Type	Constant	Initial Value	Description	Retain

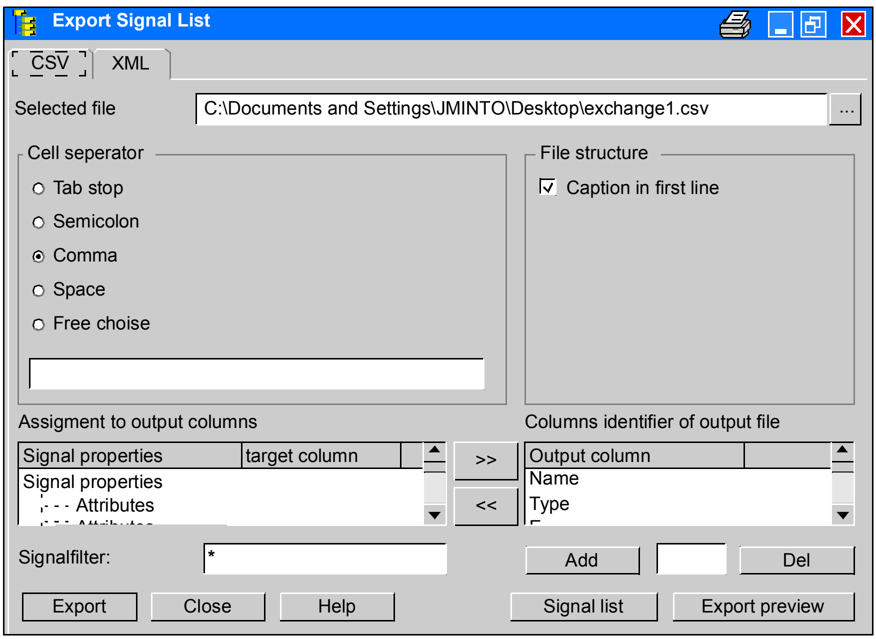
The parameters are defined as follows:

Parameter	**Description**
Name	Signal name (must be unique)
Type	unit type of signal e.g. **BOOL**, **BYTE**, etc.
Constant	if the signal is constant: **TRUE** or **FALSE**
Initial Value	a value according to the data type
Description	signal name description
Retain	in warm start mode keep last value: **TRUE** or **FALSE**

Creating Signal List

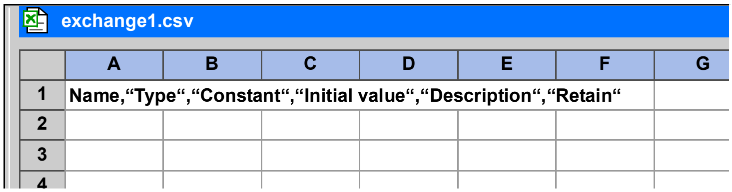
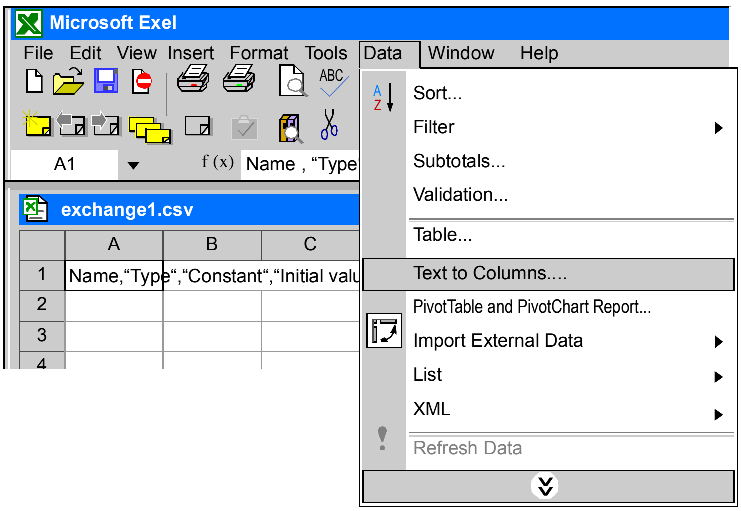
To ensure that you create the correct layout create within the **Hardware Management** the following actions:

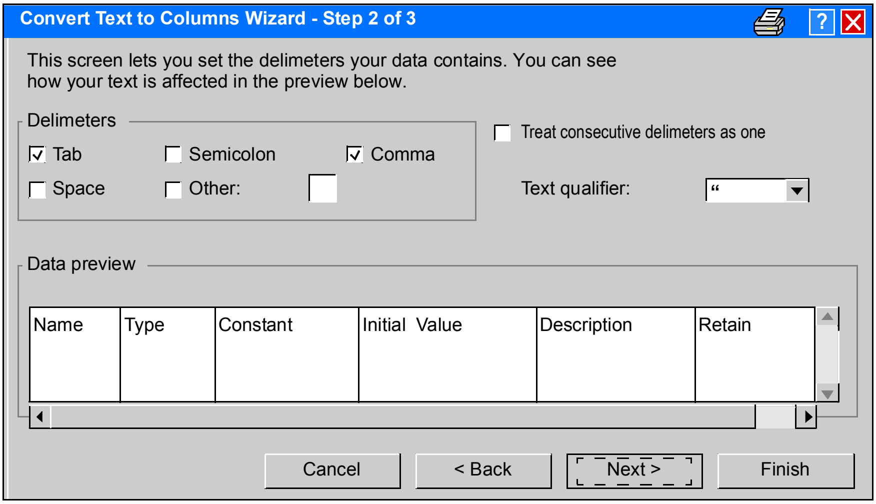
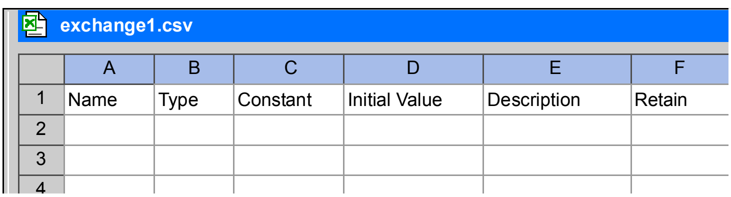
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Signals → Export Signal List from the menu bar of the Hardware Management window.


Result: The Export Signal List window opens.


</td></tr>
<tr><td>2</td><td>Select in the Export Signal List window the location you wish to store the file, and ensure that the Cell separator is a Comma.</td></tr>
<tr><td>3</td><td>Select the Export button.</td></tr>
</table>

Editing Signal List in Excel

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Start Microsoft Excel and open the file which you have just created within the Hardware Management. The file layout should look as follows (all data on one cell):

</td></tr>
<tr><td>2</td><td>Separate the data into individual cells by selecting the first cell A1 and then select Data → Text to Columns... from the menu bar.

</td></tr>
<tr><td>3</td><td>Select the option box Delimited in the first window and click Next>.</td></tr>
</table>

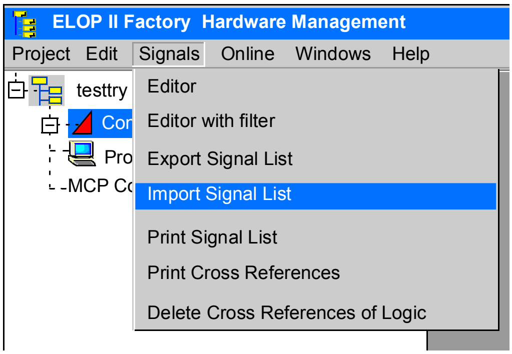
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>Ensure that the options under Delimiters are selected as follows:

</td></tr>
<tr><td>5</td><td>Click Next> and then Finish.
Your worksheet should now look like the following:

</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>To create your signal list, enter the names and values according to your needs within the Excel sheet, for example:
exchange1.csv
<table>
<tr><th></th><th>A</th><th>B</th><th>C</th><th>D</th><th>E</th><th>F</th></tr>
<tr><td>1</td><td>Name</td><td>Type</td><td>Constant</td><td>Initial Value</td><td>Description</td><td>Retain</td></tr>
<tr><td>2</td><td>Output Circular Saw</td><td>BOOL</td><td>FALSE</td><td></td><td>Dangerous Rotating blade</td><td>FALSE</td></tr>
<tr><td>3</td><td>EStop No.1 Input 1</td><td>BOOL</td><td>FALSE</td><td></td><td>Emergency Stop 1 Input 1</td><td>FALSE</td></tr>
<tr><td>4</td><td>EStop No.1 Input 2</td><td>BOOL</td><td>FALSE</td><td></td><td>Emergency Stop 1 Input 2</td><td>FALSE</td></tr>
<tr><td>5</td><td>EStop No.2 Input 1</td><td>BOOL</td><td>FALSE</td><td></td><td>Emergency Stop 2 Input 1</td><td>FALSE</td></tr>
<tr><td>6</td><td>EStop No.2 Input 2</td><td>BOOL</td><td>FALSE</td><td></td><td>Emergency Stop 2 Input 2</td><td>FALSE</td></tr>
<tr><td>7</td><td>Channel Fault ES 1.1</td><td>BYTE</td><td>FALSE</td><td></td><td>Channel 1 Fault ES 1</td><td>FALSE</td></tr>
<tr><td>8</td><td>Channel Fault ES 1.2</td><td>BYTE</td><td>FALSE</td><td></td><td>Channel 2 Fault ES 1</td><td>FALSE</td></tr>
<tr><td>9</td><td>Channel Fault ES 2.1</td><td>BYTE</td><td>FALSE</td><td></td><td>Channel 1 Fault ES 2</td><td>FALSE</td></tr>
<tr><td>10</td><td>Channel Fault ES 2.2</td><td>BYTE</td><td>FALSE</td><td></td><td>Channel 2 Fault ES 2</td><td>FALSE</td></tr>
<tr><td>11</td><td>Reset ES 1</td><td>BOOL</td><td>FALSE</td><td></td><td>Reset Pushbutton ES 1</td><td>FALSE</td></tr>
<tr><td>12</td><td>Reset ES 2</td><td>BOOL</td><td>FALSE</td><td></td><td>Reset pushbutton ES 2</td><td>FALSE</td></tr>
<tr><td>13</td><td>Error Code Unit 1</td><td>UDINT</td><td>FALSE</td><td></td><td>Error code for Unit 1</td><td>FALSE</td></tr>
</table>
</td></tr>
<tr><td>7</td><td>Save your file as either the same name or another according to your needs.</td></tr>
</table>

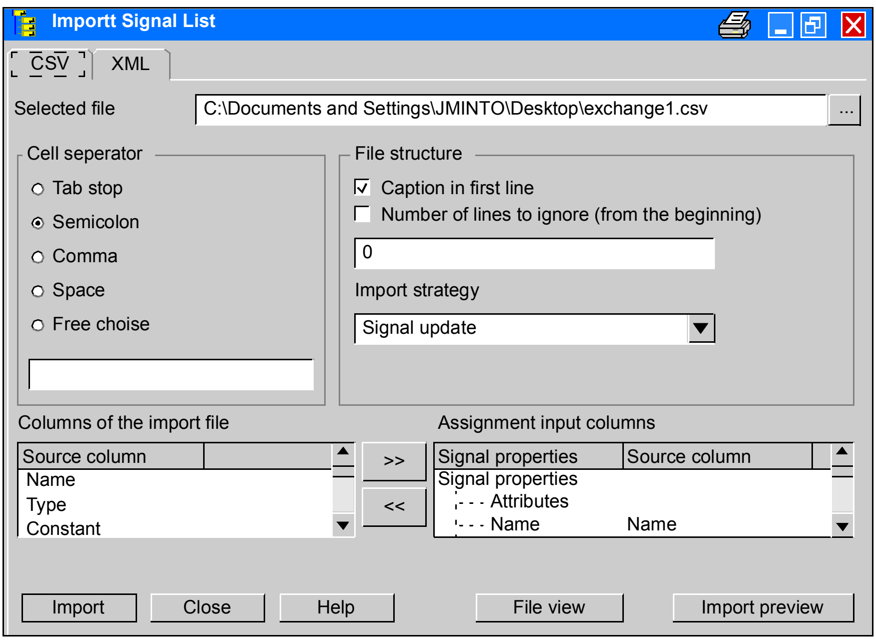
Importing Signal List in XPSMFWIN

To import your signal list into your program follow the following procedure:

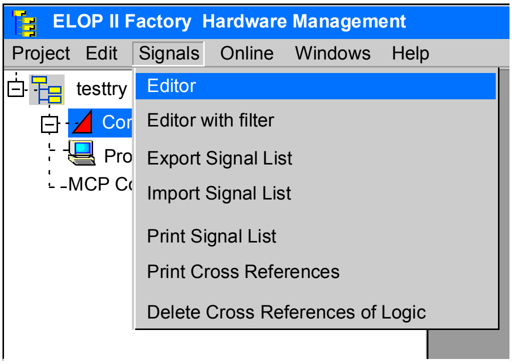
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Signals → Import Signal List from the menu bar of the Hardware Management window.



Result: The Import Signal List window opens.



</td></tr>
<tr><td>2</td><td>Select in the Import Signal List window the import file location and select the option Semicolon for Cell separator.</td></tr>
</table>

Step	Action
3	Select in the **Import Signal List** window the option **Signal update** for **Import strategy** and then click the **Import** button.
4	To check the **Signals Editor** to see if the signals have been correctly imported, select **Signals** → **Editor** from the **Hardware Management** menu bar.  These signals can now be exported in the same way as outlined above.

Importing and Exporting Communication Signals

Overview

In some cases it is faster to assign all communication signals in the local programming environments. However, for large programs, it can be very time consuming to ensure all Signals are correctly assigned to the correct addresses.

Importing

Within XPSMFWIN it is simpler to create all communication signals within the software rather than importing all communication signals.

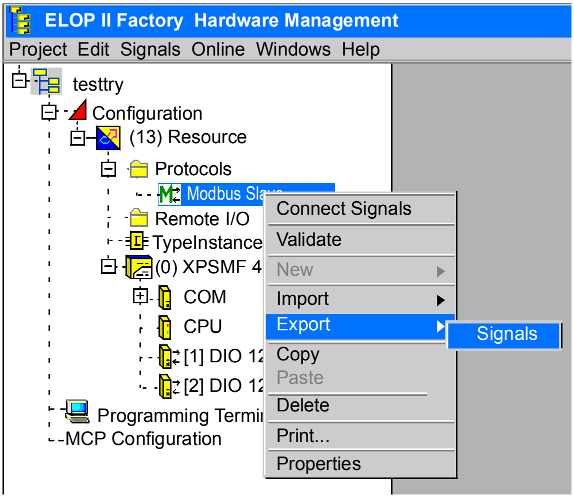
The creation of communication signals is carried out within the **Protocols** tab of each Resource (PLC). The method is simple to do as it only entails dragging the appropriate signals from the **Signals Editor** into the respective inputs or output areas.

See *Step 3: Creating Non-Safety Protocol Communication, p. 385* for details.

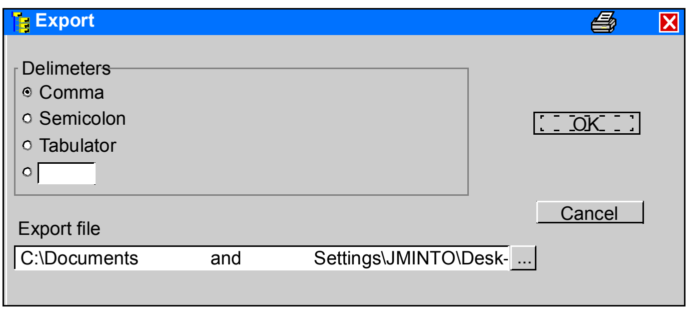
Exporting

Once the non-safety communication signals have been created for the relevant non-safety related protocols, carry out the following procedures to export the communication signals list:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>From the Protocols tab of each Resource (PLC) you wish to export, right-click on the protocol (e.g. Modbus slave, Profibus DP slave) and select Export → Signals.



Result: The Export window opens.

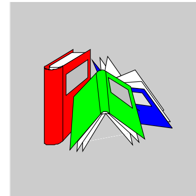

</td></tr>
<tr><td>2</td><td>Select in the Export window the Delimiter as Comma and select the export location and file name.
The export file can be used now as and where required.
Note: In most cases the format must be changed according to where you wish to import the data to.</td></tr>
</table>

XPSMF PLC Image Generator

We have created the conversion tool XPSMF PLC image generator which enables you to automatically change the format when importing the data into the Unity programming environment (for Quantum, Premium and Modicon M340 PLCs).

This tool is available from our *www.telemecanique.com* website.

Glossary



A

AC — Attached comment within the project management window of XPSMFWIN. Used to create a comment for a page, signal or variable.

C

centred starting point — Method used by XPSMFWIN 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.

CFB library — The CFB library contains 14 certified function blocks for various safety functions. These function blocks can be used alone or with other CFBs and logic such as `AND`, `OR` etc.

CFG — configuration

CG — Code generator used for compiling the project, after completion of logical programming.

configuration — Used to store a specific set of information, such as resources, programs, variable information.

CONST — constant

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.

CRC Cyclic redundancy check, a specific number which is directly dependant to a specific configuration. If even a small change is carried out within the program the CRC value will also vary.

CRF cross reference (information on inputs and outputs signal connections)

CSV data format for import and export functions
ASCII format with comma separation character (comma separated value)

D

D&D drag-and-drop, method of programming using function block diagram language.

data type Defines the properties of the value range of a variable for example `BOOL`, `Byte`, `Word`.

document editor Editor for collecting, structuring and printing program organization 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 USB hardlock, used to access programming environment. Used to prevent unauthorized access to programs.

drawing field Area of the function block diagram editor in which the logic is programmed.

DXF Drawing eXchange Format
data exchange format defined by the Autodesk company
industry standard for exchanging drawings between different CAD systems

E

Error-State-Viewer (ESV)	Area in the project management or hardware management in which the error and status messages from XPSMFWIN are output.

F

FBL	function block language
focus	navigation option in XPSMFWIN 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	A folder can contain other folders and also file objects, it is the same as a directory.
font	type and thickness of characters.
format string	element of an XPSMFWIN 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 organization 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) i.e. they have no memory.
function block (FB)	A function block is a program organization unit (POU) of the FUNCTION_BLOCK type, which is used and linked in the function block diagram editor. Function blocks For example: AND, OR, Emergency stop function block. 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 graphical programming language. Function block Diagram language is used within XPSMFWIN to reduce the programming complexity, and thus reducing the possibility of a safety failure occurring.
function block diagram editor (FBD Editor)	Editor used to create the logic in function blocks.

G

GV

Global variable is a variable which is used in various programs, function blocks, and functions which has the same name in each case.

H

hardlock

hardware protection for the XPSMFWIN program package
A device that plugs into the USB port of a computer, required for operation of XPSMFWIN. It contains the access authorization for product components of XPSMFWIN 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.

HW

hardware

I

IL

Instruction list is a programming language not used within XPSMFWIN.

instance

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

interface declaration editor

a block editor
Area of the function block language editor in which the graphic view of a block is set.

IO

input/output

IP address

individual addressing of a PES or the programming terminal for communication

L

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.

M

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.

maximize Enlarges a window to its maximum size.

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

minimize Shrinks a window to icon size.

O

offline simulation Program that enables the graphic test of the created program instance or program organization unit: the logic is **animated**. This enables errors to be detected and corrected early.

OLS offline simulation

OLT field online test field

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

overview window Area of the function block diagram editor that shows the logic pages of the function block diagram editor in a minimized 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.

P

P2P	Peer-to-peer communication is automatically set up between the safety remote IO and safety PLC on definition.
PADT	programming and debugging tool (PC)
PES	programmable electronic system (controller)
POU	program organization unit (block)
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 organization 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.
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 management	**1.** The main XPSMFWIN 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.
project tree	Display of the structure in the form of a tree inside the project.

Q

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

R

resource (RES) 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.

retain Latching properties enables the last values received to be stored and used.

revision term from the XPSMFWIN 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.

RIO remote I/O, controller with no user program

S

sequential function chart (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 XPSMFWIN window that shows the status information.

structure folder folder without special XPSMFWIN 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.

T

tab	Window element that shows the user associated information and selection options and simplifies navigation through different pages.
template project	XPSMFWIN 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 preventa PLCs/ remote IOs that exchange signals must be listed in token groups.
toolbar	bar with icons for fast access to commands

V

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	XPSMFWIN function with which the variable lists from external files or databases (e.g. CSV files, Excel files, databases) can be imported into a project.

W

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

Z

zoom

navigation option in XPSMFWIN
The visible and displayed area in the function block diagram editor can be enlarged or reduced.

Index

D

E

F

G

H

I

L

M

N

O

P

Q

R

S