Version Comparison

For Controllers Programmed with SILworX

SAFETY NONSTOP

HIMA

All of the instructions and technical specifications in this manual have been written with great care and effective quality assurance measures have been implemented to ensure their validity. If you have questions, please contact HIMA directly. HIMA appreciates any suggestions, for example concerning information which should be included in the manual.

Equipment subject to change without notice. HIMA also reserves the right to modify the written material without prior notice.

For further information, please refer to the documentation on the HIMA DVD and our websites at http://www.hima.de and http://www.hima.com.

Contact

HIMA address:

HIMA Paul Hildebrandt GmbH

P.O. Box 1261

68777 Brühl, Germany

Phone: +49 6202 709-0

Fax: +49 6202 709-107

E-mail: info@hima.com

Document designation	Description
HI 801 285 D, Rev. 2.02 (1607)	German original document
HI 801 286 E, Rev. 2.02.00 (1739)	English translation of the German original document

Table of Contents

1 Introduction

This manual describes the use of the SILworX version comparison for the programmable, safety-related controllers of the HIMax, HIMatrix and HIMatrix M45 system families.

Due to the differing hardware structures, there are differences between them, especially in the file structure of the resource configuration and in the display of hardware changes. Unless specifically stated otherwise, all of the statements apply to all of the system families mentioned above.

1.1 Target Audience and Required Competence

This document is aimed at the planners, design engineers and programmers of automation systems as well as the persons authorized to start up, operate and maintain the devices and systems concerned. Specialized knowledge of safety-related automation systems is required.

HIMA strongly recommends participating in a SILworX system training session to better understand the contents of this document.

All specialist staff (planning, installation, start-up) must be instructed concerning the risks and the associated possible consequences which can arise as a result of modifications to a safety-related automation system.

The operator is responsible for qualifying the operating and maintenance personnel and providing them with appropriate safety instructions.

1.2 Using the Version Comparison

The HIMax safety manual expressly requires the use of the version comparison:

> A safe version comparison must be used to check program changes before they are loaded to the controller!

The technical standards of functional safety (e.g. IEC 61508, IEC 61511) require that appropriate, standards-compliant modification procedures be employed for planned changes to a safety-relevant system.

A significant part of these modification procedures is the evaluation of the effects of planned changes on functional safety as well as subsequent revalidation (testing).

The version comparison provides the user with the information required for this purpose with a sufficient level of technical detail and informative value.

> The version comparison not only provides information about safety-relevant changes, but also concerning changes affecting availability! This means it also helps avoid unnecessary system shutdowns through the detection of unintentional changes.

In short:

Use of the version comparison is mandatory before every loading operation to a safety-relevant controller! Moreover, the version comparison is also a useful tool during the design and implementation phase of user applications, e.g., for comparing different project states.

2 Writing Conventions

To ensure improved readability and comprehensibility, the following writing conventions are used in this document:

Format	Description
Bold	To highlight important parts. These are names of buttons, menu functions and tabs that can be clicked and used in SILworX.
Italics	Parameters, system variables and references to other text passages.
`Courier`	Literal user input or displays which are exactly identical to the printed value.
RUN	Operating states are designated by capitals.
Chapter 1.2.3	Cross-references to other chapters. They are implemented as hyperlinks. Click the hyperlink to jump to the referenced position in the document.

Safety notices and operating tips are marked specially, as described below.

2.1 Safety Notices

The safety notices must be strictly observed to ensure that the risk to which users are exposed is as low as possible.

The safety notices are represented as follows:

- Signal word: warning, caution.
- Type and source of risk.
- Consequences arising from non-observance.
- Risk prevention.

SIGNAL WORD
Type and source of risk!
Consequences arising from non-observance
Risk prevention

Meaning of the signal words:

- **Warning:** Indicates hazardous situations which, if not avoided, could result in death or serious injury.
- **Caution:** Indicates hazardous situations which, if not avoided, could result in minor or modest injury.

2.2 Operating Tips

Additional information is structured as follows:

The text giving additional information is located here.

3 Principle

During the code generation, SILworX creates various files. This entity is referred to as the *resource configuration*. The complete resource configuration is loaded to the resource whenever a download or reload is performed.

During a version comparison, different resource configurations are compared to one another and the differences between the individual files are detected. The result has SIL 3 quality and is based on the files that describe the executable code.

Essentially, there are three types of resource configurations:

1. The created resource configuration is the result of the last code generation (→ Code generator).
2. The loaded resource configuration is the resource configuration transferred to the controller by performing a download (→ Download) or reload.
3. An unknown resource configuration represents any resource state that was exported and saved (→ IM).

4 Preparations

4.1 Programming Recommendations

When programming safety-relevant logic, the consequences of future changes should be taken into account in the early programming stages. The following measures are recommended to ensure that the version comparison results can be interpreted as easily as possible:

- Structured programming and process-specific partitioning of the logic into individual programs and function blocks.
- Individual and process-specific instance names (used function blocks).
- The names assigned to logic pages should be significant and unique.
- Connectors should not be used within a large number of logic pages. This causes widely branched networks and thus reduces clarity, in particular during the version comparison.

4.2 Preparing the Version Comparison

To achieve the version comparison quality described in this document, the code for the project to be compared must have been generated using SILworX V4 or higher.

Prior to making the planned changes, a copy of the project should be created so that at the end, both a project [OLD] with no changes and a project [NEW] with changes are available.

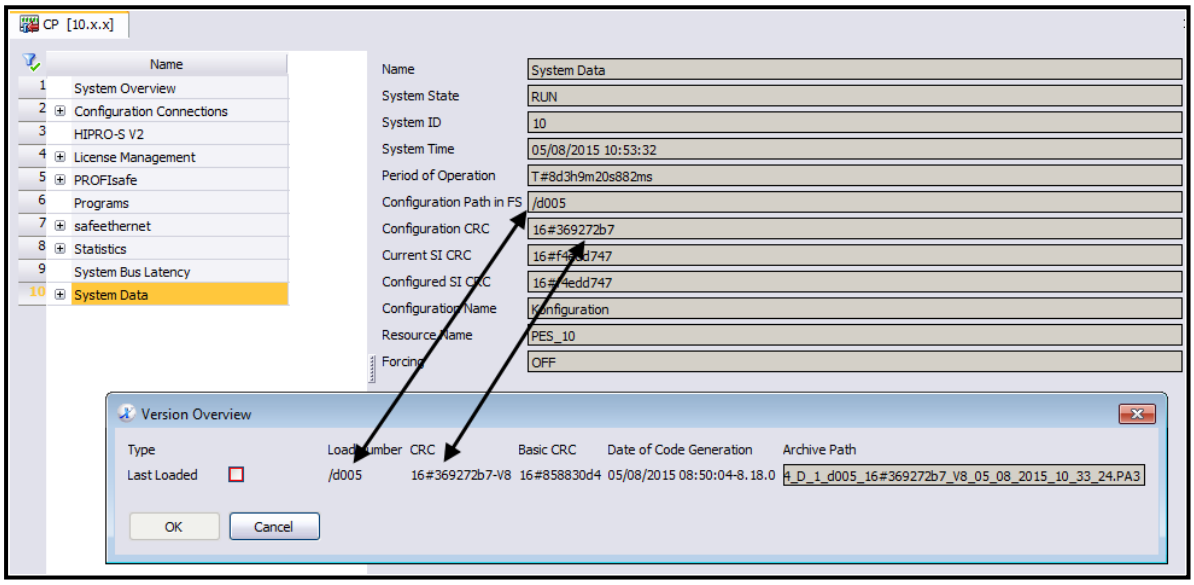
Project [OLD] represents the inspected version that was approved for safety-related operation and is identified by a unique CRC. Project [OLD] must be documented entirely.

The objective of the version comparison is to identify changes between an imported resource configuration currently loaded in a controller for project [OLD] and the modified (code generated) resource configuration for project [NEW].

4.3 Comparing Resource Configurations Online

A check must be made to ensure that the resource configuration saved and *loaded* in the SILworX project is the same as the resource configuration actually loaded in the controller.

- Open the Control Panel (CP) of the resource for which you want to carry out the comparison.
- Click **System Data**.
- In the menu, click **Online, Version Comparison**. The *Version Overview* dialog box appears.

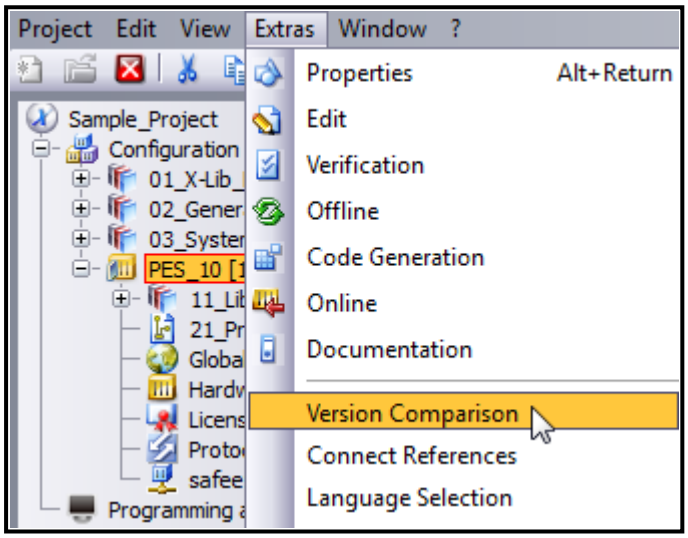


- Compare the *load number* and the *CRC*. These must match the values displayed in the Control Panel.

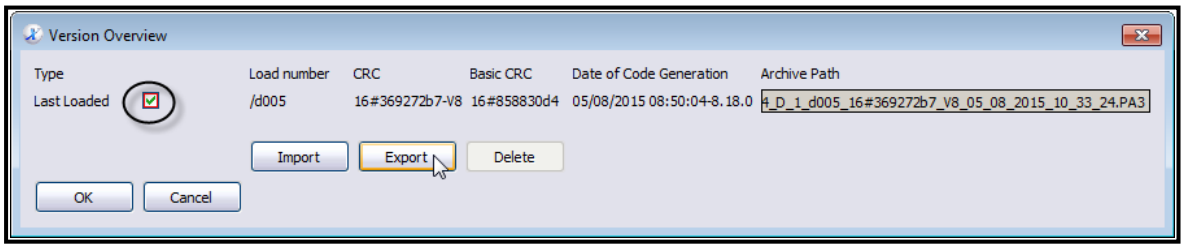
4.4 Exporting the Resource Configuration

The last loaded resource configuration is exported from project [OLD] that was selected as the basis for the comparison. To this end, proceed as follows:

- In the structure tree, select the resource for which the version comparison should be performed.
- Click the **Extras, Version Comparison** menu functions. The *Version Overview* dialog box appears.



- Check the **Last Load** option in the *Version Overview* dialog box. This ensures the export of the last version loaded into the resource.



- Click **Export**. The *Archive* dialog box appears. SILworX automatically creates an *Archive Name* with all relevant information.

Example:

PES_10_WGL_4_D_1_DL_0xf9403ba0_V3_28_03_2011_15_53_58

PES_10_	WGL_4_D_1_	DL_	0xf9403ba0_V3_	28_03_2011_15_53_58
Resource name	Project name	DL = download = loaded file Identical for reload and download	Resource configuration CRC	Date and time of the code generation

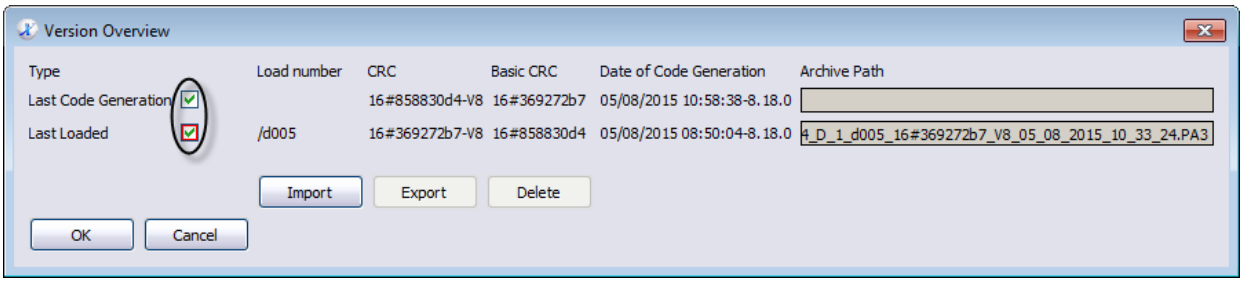
If required, adjust the *Archive Directory*. If desired, a comment can be added. Finally, click **OK** to save the archive.

4.5 Selecting Configurations for Comparison

A code comparison can be performed at any point in time.

According to the rules for functional safety and the HIMA safety manual, the newly generated configuration (CG) must be compared to the most recently loaded configuration (DL) immediately after the code generation and before a change is loaded. In this regard, please observe the corresponding sections in the HIMax and HIMatrix safety manuals.

The following screen shows an example of a situation of this nature.



The following configurations can be compared to one another:

- Imported configuration(s) → IM
- Last loaded configuration → DL
- Last generated configuration → CG

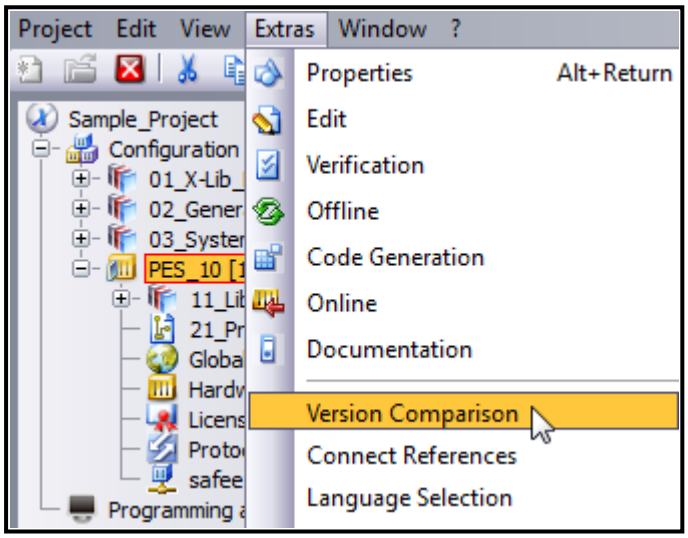
The final proof of the changes made occurs after the tests have been successfully completed. The last loaded configuration (DL) is then compared to the imported, original configuration (IM).

4.5.1 To import the resource configuration

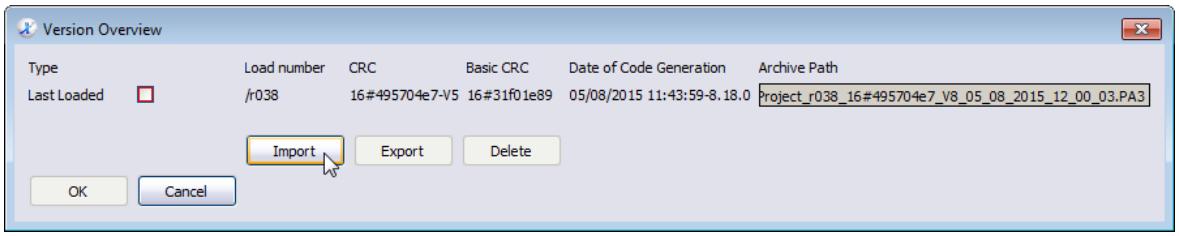
Make sure that a suitable resource configuration has been exported (see *Exporting the Resource Configuration*).

In the structure tree, select the resource for which the version comparison should be performed.

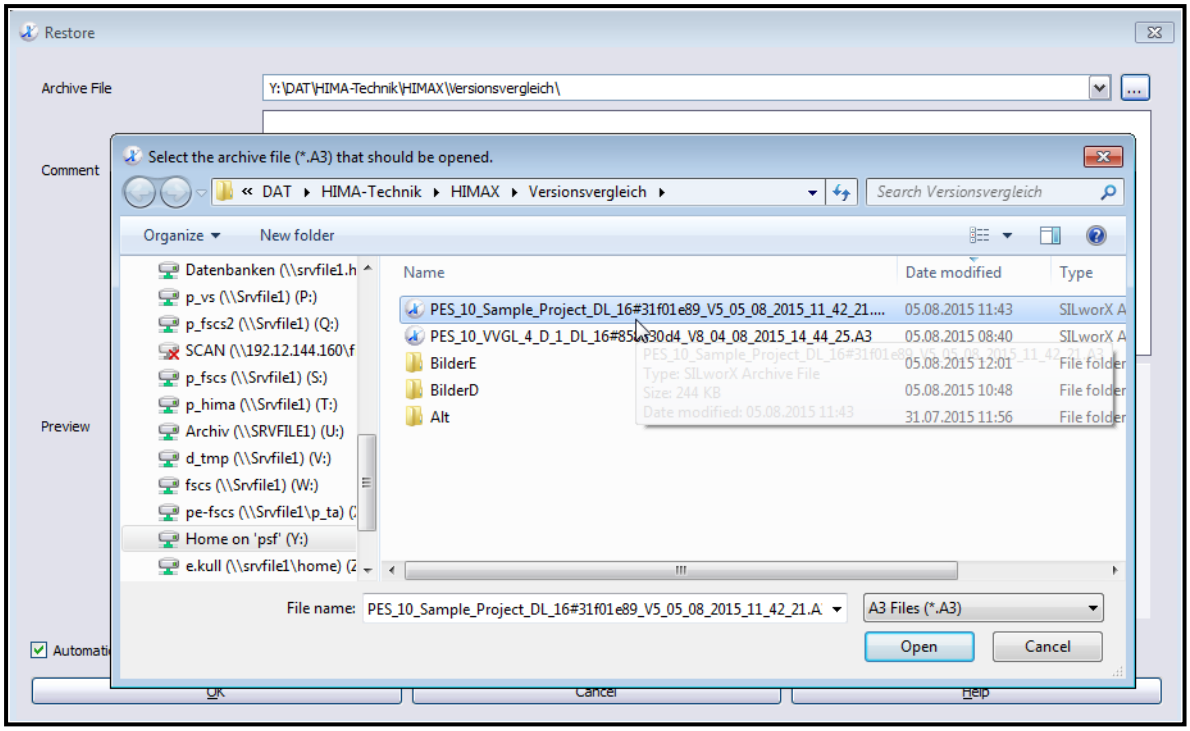
Click the **Extras, Version Comparison** menu functions. The *Version Overview* dialog box appears.



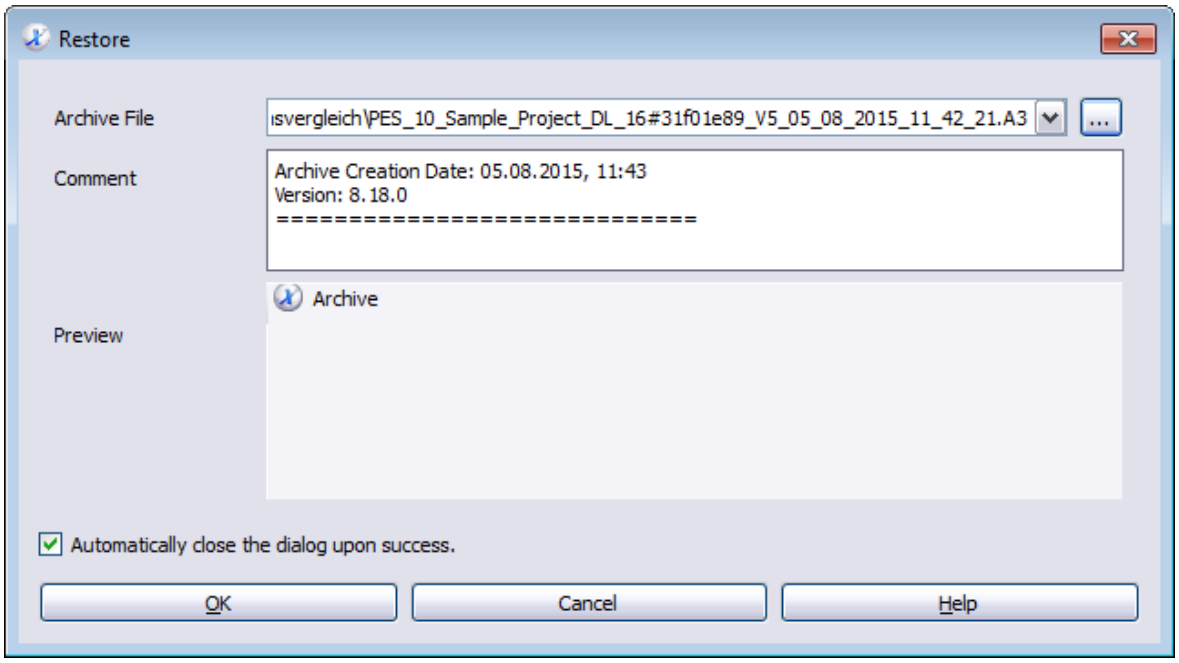
Click the **Import** button. The *Restore* dialog box appears.



Open the drop-down *Archive File* list and select an archive.
If the drop-down list does not contain the desired archive, click the button to the right of the text field and select the file via the Windows dialog box as shown in the following screen.

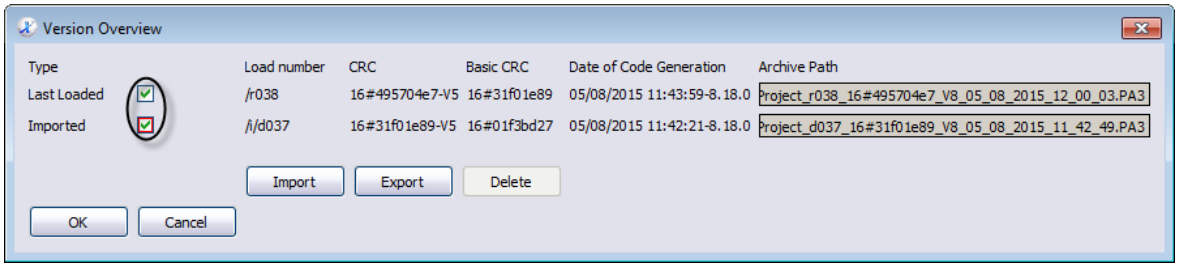


Observe the details specified in the *Comment* and *Preview* fields. They can help identify the archive.



Click **OK** to restore the archive. The successfully restored archive is displayed and the *Version Overview* dialog box appears.

Select the versions you wish to compare. To do so, check the boxes to the right of *Last Loaded* and *Imported*.



Click **OK** to start the version comparison. The result is subsequently shown in tabular form.

5 Displaying the Version Comparison

5.1 CRC Comparison

The version comparison is based on the checksums (CRCs) created by the code generator for the various function groups of the project. The function groups have a hierarchical structure and at least one configuration file.

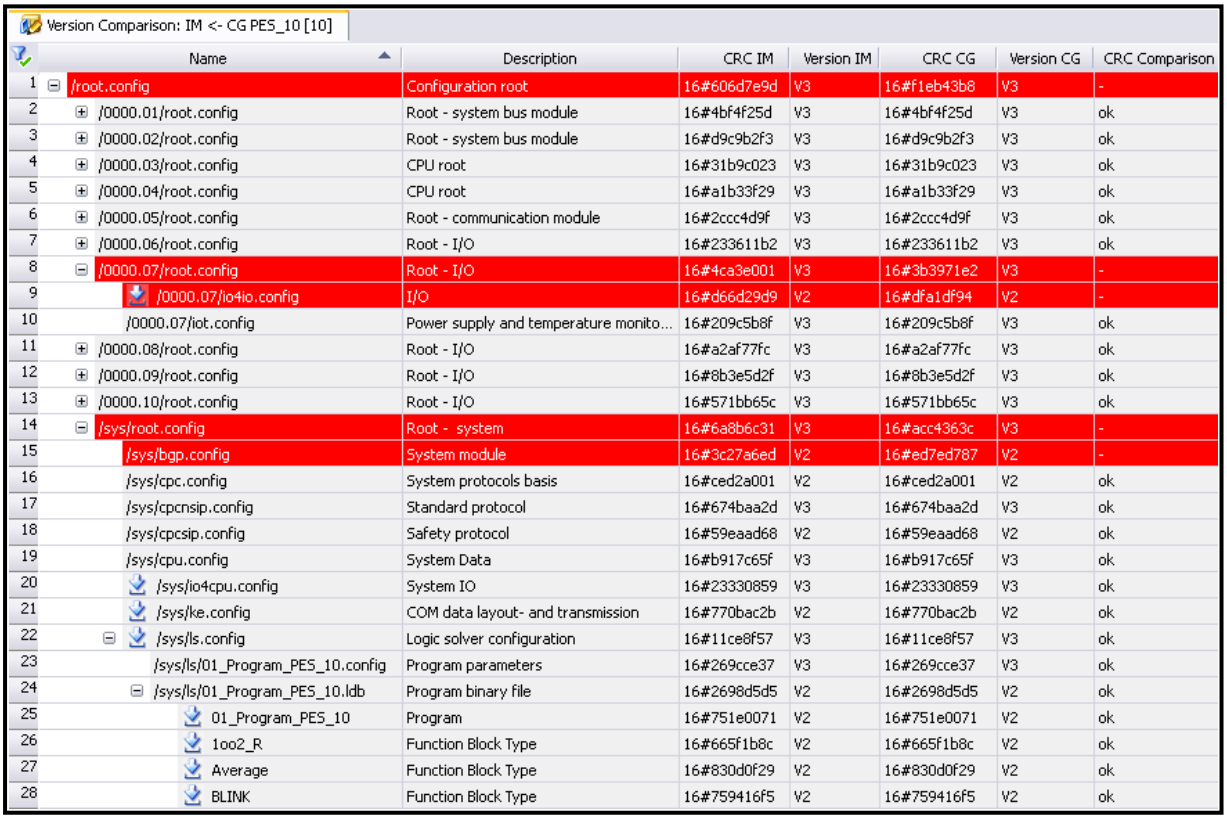
The configuration files are contained in a list and highlighted in color if changes are made.

Red The function group described by this configuration file was changed.

Yellow The function group described by this configuration file is new or was deleted.

The first */root.config* line corresponds to the higher-level code version as it is displayed in the logbook or in the Control Panel's system data. The higher-level code version combines the code versions of all function blocks. Click the (+) sign to the left of the line to display the subordinated objects.

A detailed check of the code comparison results is only necessary if the higher-level code version has changed.



Version Comparison: IM <- CG PES_10 [10]

	Name	Description	CRC IM	Version IM	CRC CG	Version CG	CRC Comparison
1	/root.config	Configuration root	16#606d7e9d	V3	16#f1eb43b8	V3	-
2	/0000.01/root.config	Root - system bus module	16#4bf4f25d	V3	16#4bf4f25d	V3	ok
3	/0000.02/root.config	Root - system bus module	16#d9c9b2f3	V3	16#d9c9b2f3	V3	ok
4	/0000.03/root.config	CPU root	16#31b9c023	V3	16#31b9c023	V3	ok
5	/0000.04/root.config	CPU root	16#a1b33f29	V3	16#a1b33f29	V3	ok
6	/0000.05/root.config	Root - communication module	16#2ccc4d9f	V3	16#2ccc4d9f	V3	ok
7	/0000.06/root.config	Root - I/O	16#233611b2	V3	16#233611b2	V3	ok
8	/0000.07/root.config	Root - I/O	16#4ca3e001	V3	16#3b3971e2	V3	-
9	/0000.07/io4io.config	I/O	16#d66d29d9	V2	16#dfa1df94	V2	-
10	/0000.07/iot.config	Power supply and temperature monito...	16#209c5b8f	V3	16#209c5b8f	V3	ok
11	/0000.08/root.config	Root - I/O	16#a2af77fc	V3	16#a2af77fc	V3	ok
12	/0000.09/root.config	Root - I/O	16#8b3e5d2f	V3	16#8b3e5d2f	V3	ok
13	/0000.10/root.config	Root - I/O	16#571bb65c	V3	16#571bb65c	V3	ok
14	/sys/root.config	Root - system	16#6a8b6c31	V3	16#acc4363c	V3	-
15	/sys/bgp.config	System module	16#3c27a6ed	V2	16#ed7ed787	V2	-
16	/sys/cpc.config	System protocols basis	16#ced2a001	V2	16#ced2a001	V2	ok
17	/sys/cpcnsip.config	Standard protocol	16#674baa2d	V3	16#674baa2d	V3	ok
18	/sys/cpcsip.config	Safety protocol	16#59eaad68	V2	16#59eaad68	V2	ok
19	/sys/cpu.config	System Data	16#b917c65f	V3	16#b917c65f	V3	ok
20	/sys/io4cpu.config	System IO	16#23330859	V3	16#23330859	V3	ok
21	/sys/ke.config	COM data layout- and transmission	16#770bac2b	V2	16#770bac2b	V2	ok
22	/sys/ls.config	Logic solver configuration	16#11ce8f57	V3	16#11ce8f57	V3	ok
23	/sys/ls/01_Program_PES_10.config	Program parameters	16#269cce37	V3	16#269cce37	V3	ok
24	/sys/ls/01_Program_PES_10.ldb	Program binary file	16#2698d5d5	V2	16#2698d5d5	V2	ok
25	01_Program_PES_10	Program	16#751e0071	V2	16#751e0071	V2	ok
26	1oo2_R	Function Block Type	16#665f1b8c	V2	16#665f1b8c	V2	ok
27	Average	Function Block Type	16#830d0f29	V2	16#830d0f29	V2	ok
28	BLINK	Function Block Type	16#759416f5	V2	16#759416f5	V2	ok

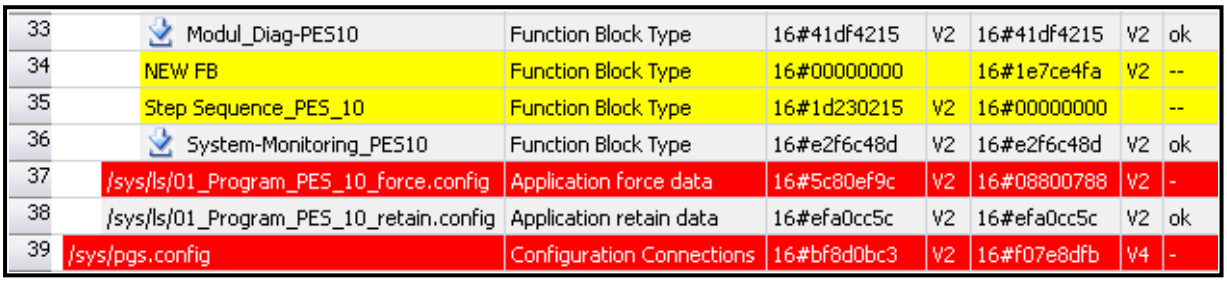
In exceptional cases, changes to a function group may have no functional effect on the code that is generated, e.g., when an input variable is renamed (see *Renaming Variables*).

If the code version did not change, no further functional check is necessary.

Caution! Hazard due to unintentional system shutdowns!
Renaming variables, program names, FB names, FB instances, etc. during a reload leads to a re-initialization of the objects mentioned. Saved values and states are lost!

5.1.1 Indication of Added, Deleted and Changed Function Groups

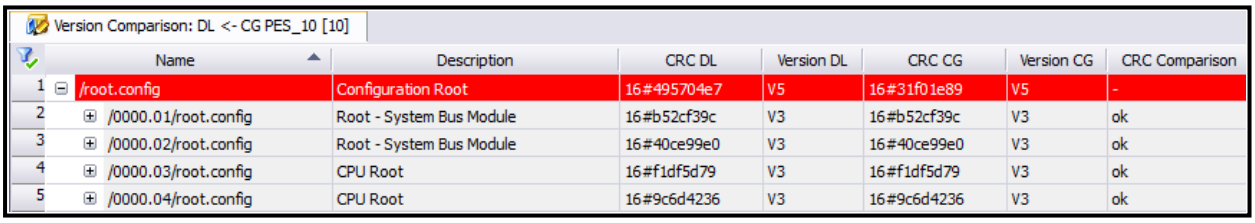
The following picture shows added and deleted function blocks (highlighted in yellow) as well as modified function blocks (highlighted in red).



33	Modul_Diag-PES10	Function Block Type	16#41df4215	V2	16#41df4215	V2	ok
34	NEW FB	Function Block Type	16#00000000		16#1e7ce4fa	V2	--
35	Step Sequence_PES_10	Function Block Type	16#1d230215	V2	16#00000000		--
36	System-Monitoring_PES10	Function Block Type	16#e2f6c48d	V2	16#e2f6c48d	V2	ok
37	/sys/ls/01_Program_PES_10_force.config	Application force data	16#5c80ef9c	V2	16#08800788	V2	-
38	/sys/ls/01_Program_PES_10_retain.config	Application retain data	16#efa0cc5c	V2	16#efa0cc5c	V2	ok
39	/sys/pgs.config	Configuration Connections	16#bf8d0bc3	V2	16#f07e8dfb	V4	-

5.2 Content and Relevance of the Most Important Files

This section merely presents the file structure which serves as the basis for the version comparison. The exact evaluation of any possible changes that may be shown is explained in the *Detailed Evaluation*.



Version Comparison: DL <- CG PES_10 [10]

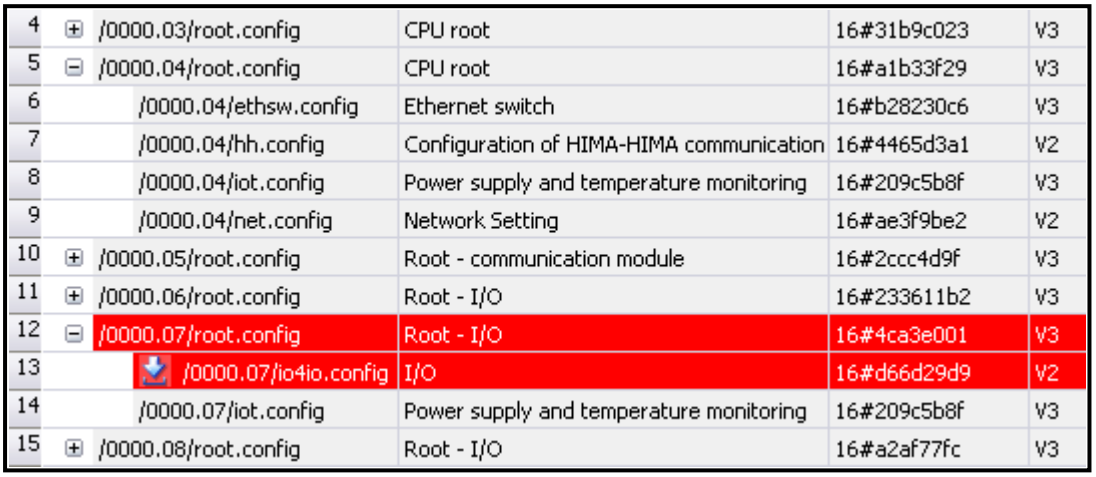
	Name	Description	CRC DL	Version DL	CRC CG	Version CG	CRC Comparison
1	/root.config	Configuration Root	16#495704e7	V5	16#31f01e89	V5	-
2	/0000.01/root.config	Root - System Bus Module	16#b52cf39c	V3	16#b52cf39c	V3	ok
3	/0000.02/root.config	Root - System Bus Module	16#40ce99e0	V3	16#40ce99e0	V3	ok
4	/0000.03/root.config	CPU Root	16#f1df5d79	V3	16#f1df5d79	V3	ok
5	/0000.04/root.config	CPU Root	16#9c6d4236	V3	16#9c6d4236	V3	ok

The information is displayed as follows:

Column	Description
Name	Module position in the Rack.Slot format, followed by the configuration file name.
Description	Short description of the configuration file.
CRC IM	Checksum of the imported configuration file.
Version IM	For the minimum operating system version required for the module (imported), see *Operating System Version Required for an Object*.
CRC CG	Checksum of the configuration file created by the code generator.
Version CG	Like Version IM, but for the configuration file created by the code generator.
Version DL	Minimum operating system version required to run the configuration file on the corresponding module.
CRC Comparison	OK: No change -: Change

5.2.1 Hardware, Modules

For each module, a file exists that groups all configuration files associated with it. Changing a module configuration (e.g., IP settings, scaling values, line monitoring, channel activation, etc.), also modifies the file content.



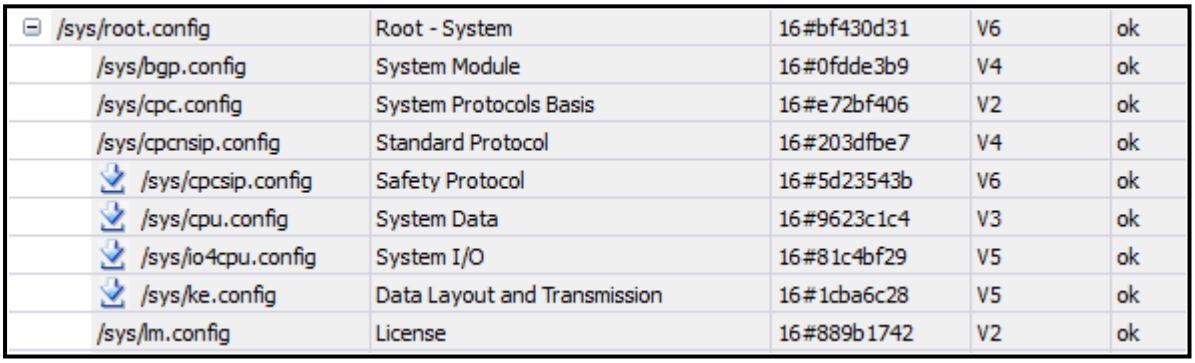
4	⊞ /0000.03/root.config	CPU root	16#31b9c023	V3
5	⊟ /0000.04/root.config	CPU root	16#a1b33f29	V3
6	/0000.04/ethsw.config	Ethernet switch	16#b28230c6	V3
7	/0000.04/hh.config	Configuration of HIMA-HIMA communication	16#4465d3a1	V2
8	/0000.04/iot.config	Power supply and temperature monitoring	16#209c5b8f	V3
9	/0000.04/net.config	Network Setting	16#ae3f9be2	V2
10	⊞ /0000.05/root.config	Root - communication module	16#2ccc4d9f	V3
11	⊞ /0000.06/root.config	Root - I/O	16#233611b2	V3
12	⊟ /0000.07/root.config	Root - I/O	16#4ca3e001	V3
13	/0000.07/io4io.config	I/O	16#d66d29d9	V2
14	/0000.07/iot.config	Power supply and temperature monitoring	16#209c5b8f	V3
15	⊞ /0000.08/root.config	Root - I/O	16#a2af77fc	V3

Configuration file	Description
/0000.04/root.config	Main file of the CPU module in rack 0, slot 4. This configuration file is referenced to subordinated configuration files and always changes if one of the subordinated files is modified.
/0000.04/ethsw.config	Properties of the Ethernet switches of the CPU module. No detail view is available. When changes are shown, check the CPU settings in the *Ethernet Switch*, *VLAN* and *Port Mirroring* tabs of the Hardware Editor.
/0000.04/hh.config:	safe**ethernet** communication properties of the CPU module. No detail view is available. When changes are shown, check the settings in the safe**ethernet** Editor. Typical changes are new or deleted connections.
/0000.04/iot.config	Power supply (single or redundant) and temperature monitoring of the CPU module. No detail view is available. When changes are shown, check the properties of the rack in the Hardware Editor.
/0000.04/net.config	Network settings of the CPU module. No detail view is available. When changes are shown, check the settings of the CPU in the *Module* and *Routings* tabs of the Hardware Editor. Typical changes are modifications of the IP address.
/0000.05/root.config	Main file of the communication module in rack 0, slot 5.
/0000.06/root.config	Main file of the I/O module in rack 0, slot 6.
/0000.07/root.config	Main file of the I/O module in rack 0, slot 7.

Configuration file	Description
	Red: The version comparison revelaed a difference between the imported and the generated configuration files.
/0000.07/io4io.config	Configuration file for the I/O module Red: The version comparison revealed a difference between the imported and the generated configuration files. A detail view is available if there are changes.
/0000.07/iot.config	Power supply (single or redundant) and temperature monitoring of the I/O module. No detail view is available. When changes are shown, check the properties of the rack in the Hardware Editor.

5.2.2 CPU Configuration and System Data

Central and higher-level data of the CPU module are grouped in the *sys/root.config* configuration file.



/sys/root.config	Root - System	16#bf430d31	V6	ok
/sys/bgp.config	System Module	16#0fdde3b9	V4	ok
/sys/cpc.config	System Protocols Basis	16#e72bf406	V2	ok
/sys/cpcnsip.config	Standard Protocol	16#203dfbe7	V4	ok
/sys/cpcsip.config	Safety Protocol	16#5d23543b	V6	ok
/sys/cpu.config	System Data	16#9623c1c4	V3	ok
/sys/io4cpu.config	System I/O	16#81c4bf29	V5	ok
/sys/ke.config	Data Layout and Transmission	16#1cba6c28	V5	ok
/sys/lm.config	License	16#889b1742	V2	ok

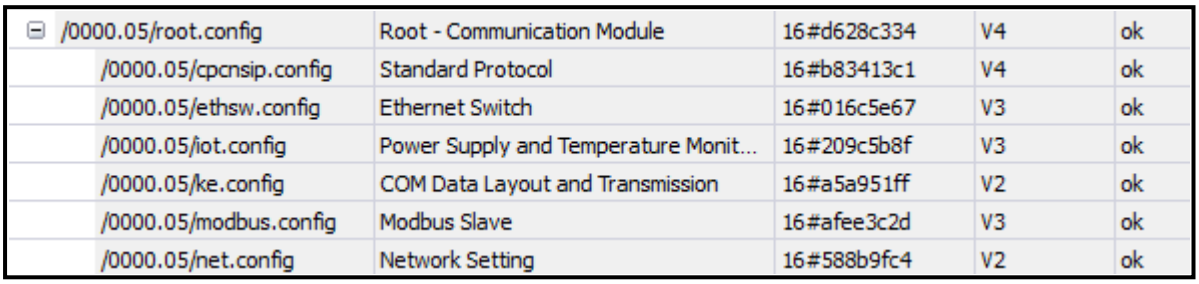
Configuration file	Description
/sys/root.config	Main file for the CPU module. This configuration file is referenced to sub-ordinated configuration files and always changes if one of the subordinated files is modified.
/sys/bgp.config	The module configuration describes all module data such as the assignment of modules to slots. This file almost always changes if the module data is modified (see *Hardware, Modules*). No detail view is available. When changes are shown, check the *Hardware*, *Modules* area.
/sys/cpc.config	Number of protocols, communication time slice ASYNC, SYNC. No detail view is available. This file changes when the number of protocols which exist or the parameter *Max. Comm. Time-Slice Async* has changed. Changes are shown in detail in other files which have also changed, e.g. *cpu.-config*.
/sys/cpcnsip.config	General rules for the protocols used to transfer data from the COM module to the CPU module. No detail view is available.

Configuration file	Description
	This file changes when fundamental properties of protocols which exist on the COM module and which are not safe (e.g. Modbus) have been changed. Changes are also shown in other files which have also changed, e.g. *ke.config*.
/sys/cpcsip.config	safe**ethernet** Parameters, properties of safe**ethernet** connections. If there are changes, detailed information is available; see *Add new Variable to Existing Connection* and *Changes in safeethernet Communication*
/sys/cpu.config	Resource settings such as allowed actions, safety time or watchdog time. If there are changes, detailed information is available.
/sys/io4cpu.config	Redundancy details of the I/O modules, scaling of analog values and counter inputs. If there are changes, detailed information is available; see *System Level (CPU): io4cpu.config*.
ke.config	Configuration file for assigning (using) global variables to hardware, protocols, POUs, etc. (*ke* = communication endpoint = global variable). If there are changes, detailed information is available; see *Assigning Global Variables to Another Hardware Input* and *New Initial Value for a Global Variable*.
lm.config	Configuration file for license management. No detail view is available. This file changes when the quantity or the designation of licenses has been changed. These changes are fundamentally of no safety relevance. It is possible for the file to change even though no licenses were changed. The reason for this is found in modified internal sorting criteria as of SILworX V6 when several licenses exist. Changes of this nature can be ignored.

5.2.3 COM Configuration and Protocols

The COM module data (e.g., protocols, interfaces, etc.) are saved in individual configuration files subordinated to *root.config*, the main configuration file for the COM module.

Example



/0000.05/root.config	Root - Communication Module	16#d628c334	V4	ok
/0000.05/cpcnsip.config	Standard Protocol	16#b83413c1	V4	ok
/0000.05/ethsw.config	Ethernet Switch	16#016c5e67	V3	ok
/0000.05/iot.config	Power Supply and Temperature Monit...	16#209c5b8f	V3	ok
/0000.05/ke.config	COM Data Layout and Transmission	16#a5a951ff	V2	ok
/0000.05/modbus.config	Modbus Slave	16#afee3c2d	V3	ok
/0000.05/net.config	Network Setting	16#588b9fc4	V2	ok

Configuration file	Description
/0000.05/root.config	Main file for the COM module. This configuration file is referenced to subordinated configuration files and always changes if one of the subordinated files is modified.
/0000.05/cpcnsip.config	General rules for the protocols used to transfer data from the COM module to the CPU module, e.g., behavior if the connection is lost. No detail view is available. This file changes when fundamental properties of non-safe protocols existing on the COM module (e.g., Modbus) have been changed. Changes are also shown in other files which have also changed, e.g., *ke.config*.
/0000.05/ethsw.config	Properties of the Ethernet switches of the COM module. No detail view is available. When changes are shown, check the settings of the COM in the *Ethernet Switch*, *VLAN* and *Port Mirroring* tabs of the Hardware Editor.
/0000.05/iot.config	Power supply (single or redundant) and temperature monitoring of the COM module. No detail view is available. When changes are shown, check the properties of the rack in the Hardware Editor.
/0000.05/ke.config	Configuration file for reading and writing global variables in protocols (*ke* = communication endpoint = global variable). No detail view is available. When changes are shown, further information is available at the system level in the *ke.config* file; see *Assigning Global Variables to Another Hardware Input* and *New Initial Value for a Global Variable*.

Configuration file	Description
/0000.05/modbus.config	Configuration file for the Modbus protocol properties. No detail view is available. This file changes when fundamental properties of the Modbus protocol have been changed. In this case, also check for additional changed files, e.g. *ke.-config*.
/0000.05/net.config	Configuration file for the network settings of this COM module. No detail view is available. When changes are shown, check the settings of the COM module in the *Module* and *Routings* tabs of the Hardware Editor. Typical changes are modifications of the IP address.

5.2.4 Logic Data

The checksums of the displayed function blocks (POUs) are so-called source code CRCs. If the executable code of a function block changes, the binary file (.ldb) of the program in which the function block is used will also change.

Not all function block changes affect the executable code, e.g., when renaming a local variable. The code comparison identifies a change in the source code, i.e., the line becomes red, but the binary file does not change. In this case, no code-relevant effects are to be expected from the change and no additional check is necessary.

Caution! Hazard due to unintentional system shutdowns!
The renaming of variables, program names, FB instances etc. leads to a reinitialization of these variables when a reload is performed. Saved values and states are lost!

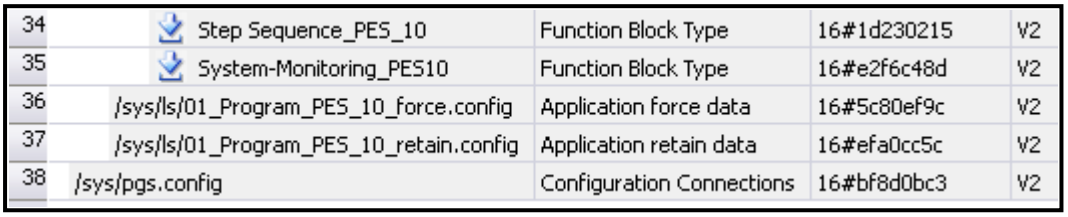
Example

21	/sys/ls.config	Logic solver configuration	16#11ce8f57	V3
22	/sys/ls/01_Program_PES_10.config	Program parameters	16#269cce37	V3
23	/sys/ls/01_Program_PES_10.ldb	Program binary file	16#2698d5d5	V2
24	01_Program_PES_10	Program	16#751e0071	V2
25	1oo2_R	Function Block Type	16#665f1b8c	V2
26	Average	Function Block Type	16#830d0f29	V2
27	BLINK	Function Block Type	16#759416f5	V2
28	BUFFER	Array	16#f0d16020	V2
29	Diag_AI32_01	Function Block Type	16#26961d85	V2
30	Global Variables	Global Variables	16#c67682a5	V2
31	LIMH_R	Function Block Type	16#939cbd0b	V2
32	LIML_R	Function Block Type	16#882b701d	V2
33	Modul_Diag-PES10	Function Block Type	16#41df4215	V2
34	Step Sequence_PES_10	Function Block Type	16#1d230215	V2
35	System-Monitoring_PES10	Function Block Type	16#e2f6c48d	V2
36	/sys/ls/01_Program_PES_10_forc...	Application force data	16#5c80ef9c	V2
37	/sys/ls/01_Program_PES_10_reta...	Application retain data	16#efa0cc5c	V2
38	/sys/pgs.config	Configuration Connections	16#bf8d0bc3	V2

Configuration file, object data	Description
/sys/ls.config	Main logic file (logic solver). This configuration file is referenced to subordinated configuration files and always changes if one of the subordinated file is modified, e.g., multitasking properties.
/sys/ls/Programm.config	Program properties, multitasking settings, allowed actions, etc. No detail view is available. When changes are shown, check the properties of the program.
/sys/ls/Programm.ldb	The binary file (loadable) is the executable code of the entire logic and changes whenever the logic is modified. Detailed information is available as of SILworX V7, see *Logic Changes (Logic Solver)*.
01_Programm01 (= Name of a program)	CRC of the program (as a POU). If there are changes, detailed information is available, see *Logic Changes (Logic Solver)*.
1oo2_R (= Name of a function block)	CRC of the function block (as a POU). If there are changes, detailed information is available, see *Logic Changes (Logic Solver)*.
Buffer (= name of a data type)	CRC of a user-defined data type. No detail view is available. When changes are shown, check the properties of the mentioned data type.
Global Variables	Properties concerning how global variables are used in function blocks, e.g., data type, sequence (sorting order), etc. If a change made to one of this property affects the executable code, the binary file (.ldb) also changes. In this case, further information is shown in the detail view of the concerned function block. If applicable, changes to *ke.config* are also shown. Further details can also be found in the ke.config.
/sys/ls/force.config	Additional support information for forcing in the logic. It may also change if the use of a global variable is modified in the logic, see *Assigning Global Variables to Another Hardware Input* and *New Initial Value for a Global Variable*. If applicable, changes to *ke.config* are also shown. Further details can also be found in the ke.config.
/sys/ls/retain.config	Retain information about the global variables used in the logic. It may also change if the use of a global variable is modified in the logic, see *Assigning Global Variables to Another Hardware Input* and *New Initial Value for a Global Variable*. If applicable, changes to *ke.config* are also shown. Further details can also be found in the ke.config.

5.2.5 PGS Data (Configuration Connections, User Management)

The module protocol data (BGP), the remote I/O connections and the user management are saved in the *pgs.config* configuration file.



34	Step Sequence_PES_10	Function Block Type	16#1d230215	V2
35	System-Monitoring_PES10	Function Block Type	16#e2f6c48d	V2
36	/sys/ls/01_Program_PES_10_force.config	Application force data	16#5c80ef9c	V2
37	/sys/ls/01_Program_PES_10_retain.config	Application retain data	16#efa0cc5c	V2
38	/sys/pgs.config	Configuration Connections	16#bf8d0bc3	V2

Configuration file	Description
/sys/pgs.config	Configuration connection data, e.g., *Max. Duration of Configuration Connections*, *PES User Management*. No detail view is available. A change that is shown can be due to the following reasons: • The setting *Max. Duration of Configuration Connections* (resource property) was changed. • The PES user management settings (if present) were changed. • The settings of the remote I/O connections (if present) were changed.

5.2.6 Operating System Version Required for an Object

The checksum and the operating system version are displayed for each configuration file. The operating system version required for a module depends on the functions used. For instance, the *Max. Duration of Configuration Connections* can only be modified with an operating system V4 or higher; see the example below.

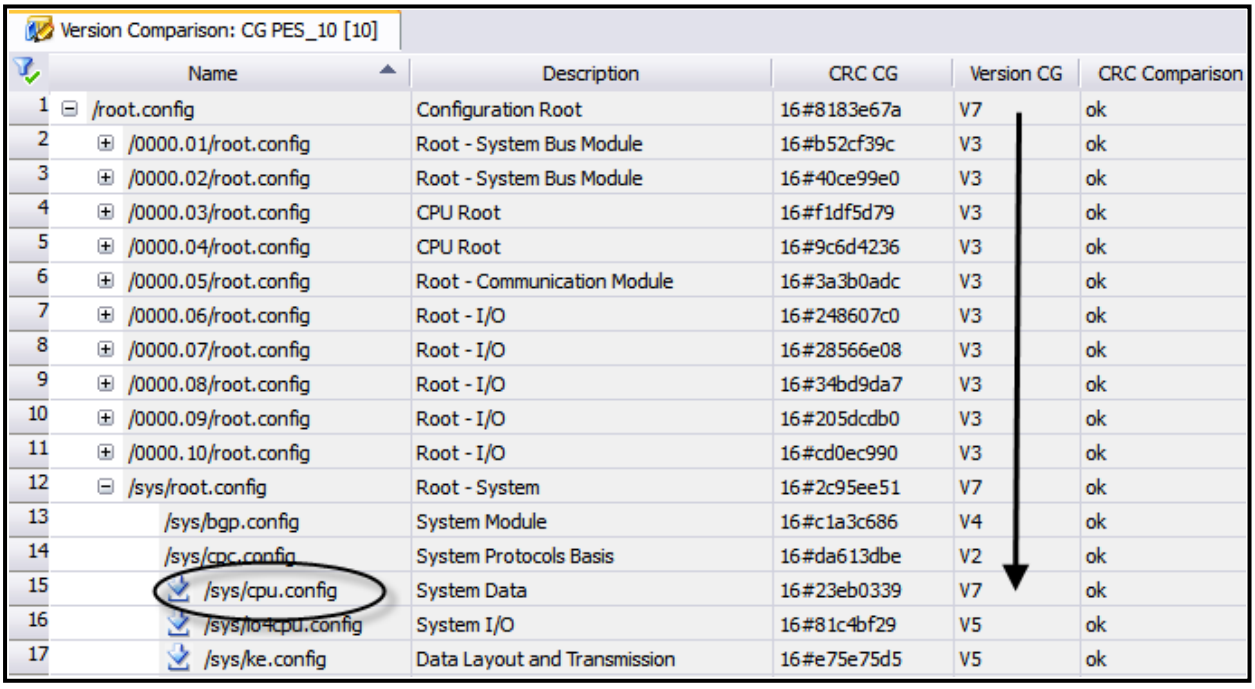
The CRC and the highest operating system version required anywhere are already displayed in the logbook upon completion of the code generation.

In the */root.config line, i.e., the main configuration file*, the version comparison also displays the highest operating system version required anywhere.

> A configuration can only be loaded into a controller if all modules in use are equipped with at least the operating system version determined by the code generator. Modules with unsuitable operating system versions reject the configuration as invalid.

For all files that cannot be allocated to an individual module according to the SRS format (System Rack Slot, e.g., /0000.03/ = Rack0, Slot3), this function is carried out by the CPU. In such a case, the CPU must be equipped with the required operating system version.

To display the subordinated objects, click the (+) sign to the right of the line number in the hierarchical list of the configuration files.



	Name	Description	CRC CG	Version CG	CRC Comparison
1	/root.config	Configuration Root	16#8183e67a	V7	ok
2	/0000.01/root.config	Root - System Bus Module	16#b52cf39c	V3	ok
3	/0000.02/root.config	Root - System Bus Module	16#40ce99e0	V3	ok
4	/0000.03/root.config	CPU Root	16#f1df5d79	V3	ok
5	/0000.04/root.config	CPU Root	16#9c6d4236	V3	ok
6	/0000.05/root.config	Root - Communication Module	16#3a3b0adc	V3	ok
7	/0000.06/root.config	Root - I/O	16#248607c0	V3	ok
8	/0000.07/root.config	Root - I/O	16#28566e08	V3	ok
9	/0000.08/root.config	Root - I/O	16#34bd9da7	V3	ok
10	/0000.09/root.config	Root - I/O	16#205dcdb0	V3	ok
11	/0000.10/root.config	Root - I/O	16#cd0ec990	V3	ok
12	/sys/root.config	Root - System	16#2c95ee51	V7	ok
13	/sys/bgp.config	System Module	16#c1a3c686	V4	ok
14	/sys/cpc.config	System Protocols Basis	16#da613dbe	V2	ok
15	/sys/cpu.config	System Data	16#23eb0339	V7	ok
16	/sys/io+cpu.config	System I/O	16#81c4bf29	V5	ok
17	/sys/ke.config	Data Layout and Transmission	16#e75e75d5	V5	ok

6 Detailed Evaluation

6.1 Hardware Changes (in the Hardware Editor)

6.1.1 I/O Modules: io4io.config

The *io4io.config* configuration file for an I/O module changes if changes are made to the configuration data of the I/O module . This includes:

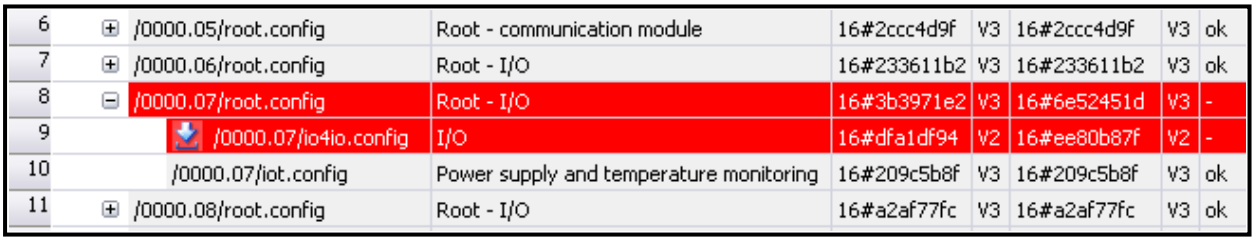
- Changes to the *Module* tab, e.g., noise blanking.
- Changes to the *I/O Submodule* tab, e.g., a supply's activation.
- Changes to fixed values, such as *OC Limit* or *scaling values*.

> Note that assigning the properties of a module (e.g., channel value) to a global variable does not necessarily cause a change in the configuration data.

Because the *Channel Used* parameter is implicitly configured, changing, adding and deleting a global variable causes changes in the configuration files of the following modules:

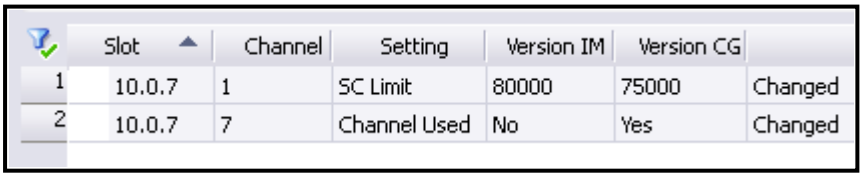
• X-DI 32 02 • X-DI 32 05	• X-AI 32 01 • X-AI 32 02	• X-CI 24 01

Example



6	/0000.05/root.config	Root - communication module	16#2ccc4d9f	V3	16#2ccc4d9f	V3	ok
7	/0000.06/root.config	Root - I/O	16#233611b2	V3	16#233611b2	V3	ok
8	/0000.07/root.config	Root - I/O	16#3b3971e2	V3	16#6e52451d	V3	-
9	/0000.07/io4io.config	I/O	16#dfa1df94	V2	16#ee80b87f	V2	-
10	/0000.07/iot.config	Power supply and temperature monitoring	16#209c5b8f	V3	16#209c5b8f	V3	ok
11	/0000.08/root.config	Root - I/O	16#a2af77fc	V3	16#a2af77fc	V3	ok

Double-click the *io4io.config* line in the configuration file (line 9 in the screen above) to open the detail view.



	Slot	Channel	Setting	Version IM	Version CG	
1	10.0.7	1	SC Limit	80000	75000	Changed
2	10.0.7	7	Channel Used	No	Yes	Changed

The detailed list specifies the following information:

Column	Description
Slot	I/O module slot in the System.Rack.Slot format.
Channel	I/O module channel concerned.
Setting	Relevance of the parameter or function.
Version IM	Value in the imported configuration file version.
Version CG	Value in the configuration file version created by the code generator.

6.1.2 System Level (CPU): io4cpu.config

Some changes affect the io4io.config configuration file as well as the *io4cpu.config* configuration file. This is the case, for instance, when the scaling values for analog input modules are changed.

Since the CPU module supports the I/O modules in calculating the scaling, changes to the scaling values affect both the *io4io.config* and *io4cpu.config* configuration files (see the following screen).

Example

13	/0000.10/root.config	Root - I/O	16#571bb65c	V3	16#613d43d1	V3	-
14	/0000.10/io4io.config	I/O	16#b602116e	V2	16#0f7b4b73	V2	-
15	/0000.10/iot.config	Power supply and temperature monito...	16#209c5b8f	V3	16#209c5b8f	V3	ok
16	/sys/root.config	Root - system	16#b6791639	V4	16#958ff17b	V4	-
17	/sys/bgp.config	System module	16#37524232	V2	16#20941c5c	V2	-
18	/sys/cpc.config	System protocols basis	16#ced2a001	V2	16#ced2a001	V2	ok
19	/sys/cpcnsip.config	Standard protocol	16#674baa2d	V3	16#674baa2d	V3	ok
20	/sys/cpcsip.config	Safety protocol	16#59eaad68	V2	16#59eaad68	V2	ok
21	/sys/cpu.config	System Data	16#b917c65f	V3	16#b917c65f	V3	ok
22	/sys/io4cpu.config	System IO	16#23330859	V3	16#6bda7994	V3	-
23	/sys/ke.config	Data layout and transmission	16#29a3146b	V2	16#29a3146b	V2	ok

Double-click the *io4cpu.config* line in the configuration file (line 22 in the screen above) to open the detail view.

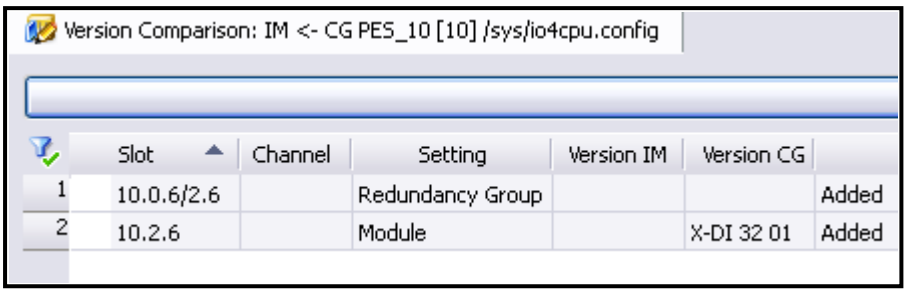
In the following picture, the modules in slot 9 and slot 10 are redundantly connected such that the change affects both modules.

Version Comparison: IM <- CG PES_10 [10] /sys/io4cpu.config

	Slot	Channel	Setting	Version IM	Version CG	
1	10.0.9	1	4 mA	100.0	110.00000000000364	Changed
2	10.0.9	1	20 mA	100000.0	100000.00000000001	Changed
3	10.0.10	1	4 mA	100.0	110.00000000000364	Changed
4	10.0.10	1	20 mA	100000.0	100000.00000000001	Changed

As a result of the internal structure with mantissa and exponent, REAL numbers might be represented with decimal places (see *Version CG* column). The decimal places can be ignored!
Even if only one parameter has changed, usually both the *4 mA* and the *20 mA* base points are affected by the underlying mathematical function.

The following picture shows an X-DI 32 01 module in a redundancy group. Since the redundancy evaluation is performed in the CPU module, this action also affects the *io4cpu.config* file.



	Slot	Channel	Setting	Version IM	Version CG	
1	10.0.6/2.6		Redundancy Group			Added
2	10.2.6		Module		X-DI 32 01	Added

6.1.3 Adding new Modules

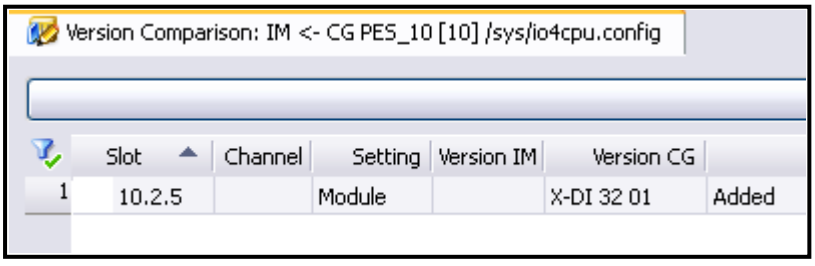
If a module is added to a system, the configuration file of the new module is highlighted in yellow in the version comparison.

Line 12 of the following picture shows that a new module was added to rack 2, slot 5. This also affects the */sys/bgp.config* module management. The module configuration is saved in the */sys/io4cpu.config* and */sys/ke*.config files.

The module must be equipped with an operating system no older than V3 as the rack was configured for temperature monitoring (line 14).

11	/0000.10/root.config	Root - I/O	16#571bb65c	V3	16#571bb65c	V3	ok
12	/0002.05/root.config	Root - I/O	16#00000000		16#7af285d6	V3	--
13	/0002.05/io4io.config	I/O	16#00000000		16#f03c1e0f	V2	--
14	/0002.05/iot.config	Power supply and temperature monito...	16#00000000		16#209c5b8f	V3	--
15	/sys/root.config	Root - system	16#67382b6f	V3	16#59d9ddc8	V3	-
16	/sys/bgp.config	System module	16#c9c01aa6	V2	16#4f6e1c97	V2	-
17	/sys/cpc.config	System protocols basis	16#ced2a001	V2	16#ced2a001	V2	ok
18	/sys/cpcnsip.config	Standard protocol	16#674baa2d	V3	16#674baa2d	V3	ok
19	/sys/cpcsip.config	Safety protocol	16#59eaad68	V2	16#59eaad68	V2	ok
20	/sys/cpu.config	System Data	16#b917c65f	V3	16#b917c65f	V3	ok
21	/sys/io4cpu.config	System IO	16#6a115cda	V3	16#b525b14c	V3	-
22	/sys/ke.config	Data layout and transmission	16#2f53e299	V2	16#3f58a831	V2	-
23	/sys/ls.config	Logic solver configuration	16#422dd7ed	V3	16#422dd7ed	V3	ok

Double-click the */sys/io4cpu.config* line (line 21 in the screen shown above) to display further details on the added module.



	Slot	Channel	Setting	Version IM	Version CG	
1	10.2.5		Module		X-DI 32 01	Added

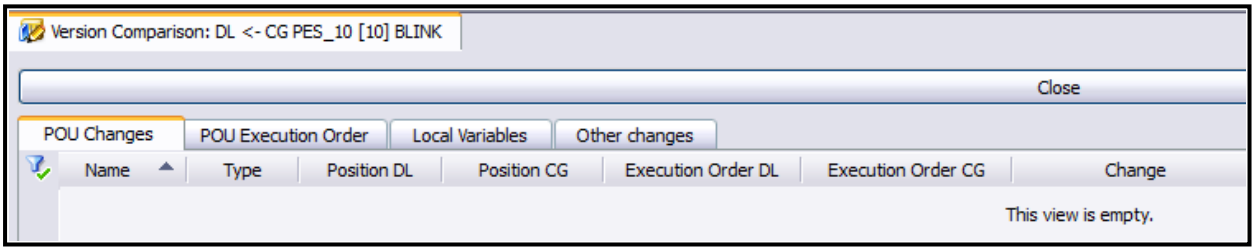
6.2 Logic Changes (Logic Solver)

If changes are performed to the logic, at least one configuration file subordinated to the */sys/ls.config* logic solver configuration file will change. Additionally, changes also affect the *Program.ldb* program binary file.

The source code is displayed for all POUs. The source code is converted to executable code during the code generation. The program binary file (loadable) will only change if the executable code has changed (functional change).

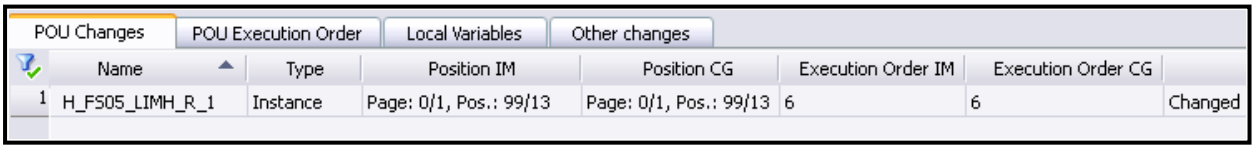
/sys/ls.config	Logic Solver Configuration	16#52a3f499	V3	16#3cc4b9f2	V3	-
/sys/ls/21_Program_PES_10.config	Program Parameters	16#ed8b2329	V3	16#7813f844	V3	-
/sys/ls/21_Program_PES_10.ldb	Program Binary File	16#90ba99c0	V2	16#bf165f3b	V2	-
2oo3	Function Block Type FBD	16#0c925672	V2	16#0c925672	V2	ok
2oo3B	Function Block Type FBD	16#2a457e09	V2	16#2a457e09	V2	ok
21_Program_PES_10	Program	16#10a20dca	V2	16#10a20dca	V2	ok
Average	Function Block Type FBD	16#020731ca	V2	16#020731ca	V2	ok

Double-click the line associated with the changed POU to open the detail view. Changes are shown in at least one tab. Check all tabs for changes! The following examples explain some evaluation details.



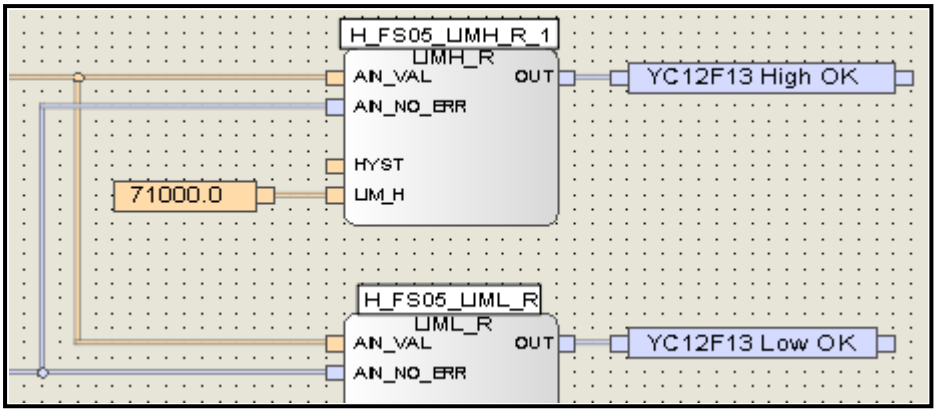
6.2.1 Changing the Value Field at the Input of a Function Block

In the picture below, *Changed* noted in the *POU Changes* tab indicates that changes were made to the input information of the *LIMH_R_1* function block instance. The *position* is the upper left corner of the function block instance on the worksheet.



Double-click the *LIMH_R_1* line to open the FBD Editor and center the logic to the modified POU.

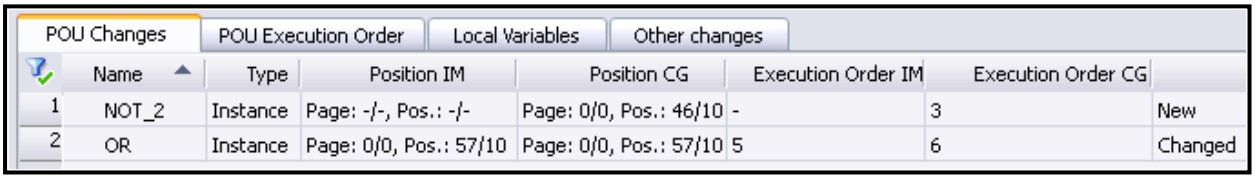
Details can be recognized by making a direct comparison of logic page (IM) with logic page (CG), e.g., by evaluating a corresponding POU documentation. In the example below, the value field changed to `70010.0`.



6.2.2 Adding a New Object in the Logic

The empty *Position IM* and *Execution Order IM* columns in the *POU Changes* tab indicate that the object *NOT_2* did not exist in the imported configuration. The most recently generated configuration contains details on the *Position CG* and *Execution Order CG*. Additionally, the POU is marked as *New*.

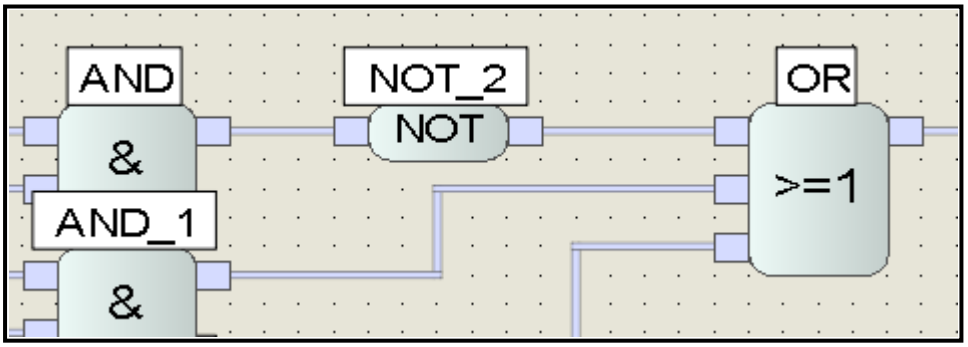
The input information for the *OR* object changed.



POU Changes | POU Execution Order | Local Variables | Other changes

	Name	Type	Position IM	Position CG	Execution Order IM	Execution Order CG	
1	NOT_2	Instance	Page: -/-, Pos.: -/-	Page: 0/0, Pos.: 46/10	-	3	New
2	OR	Instance	Page: 0/0, Pos.: 57/10	Page: 0/0, Pos.: 57/10	5	6	Changed

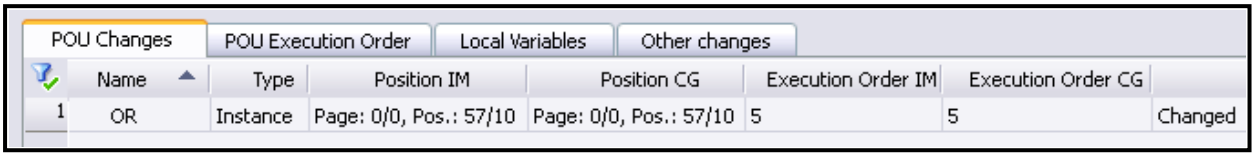
Double-click the *NOT_2* line to open the FBD Editor and center the logic to the modified POU.



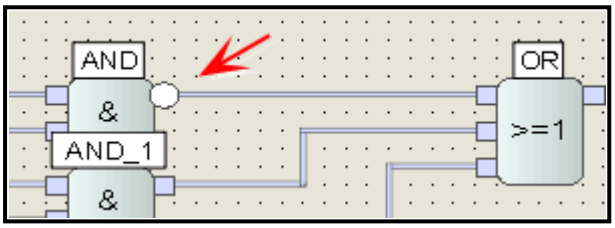
Additionally, the POU instance *NOT_2* was added to the logic.

6.2.3 Inverting a Function Output

Changed noted in the *POU Changes* tab indicates that logic changes were made to one of the inputs of the *OR* object.



Double-click the *OR* line to open the FBD Editor and center the logic to the modified POU.

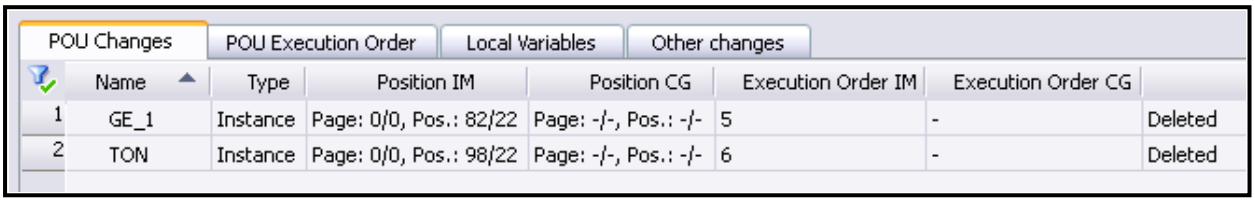


The signal associated with the first input of the OR instance is inverted.

Changes shown always refer only to the input processing of instances of functions and function blocks.
The AND function, however, was not changed because the inversion is carried out after the AND has been executed.

6.2.4 Deleting the POU from the Logic

The empty *Position CG* and *Execution Order CG* columns in the *POU Changes* tab indicate that the POU instances *G_1* and *TON* do not exist in the most recently generated configuration of the project. The imported configuration contains details on the *Position IM* and *Execution Order IM*. Additionally, the POU instances are marked as *Deleted*.

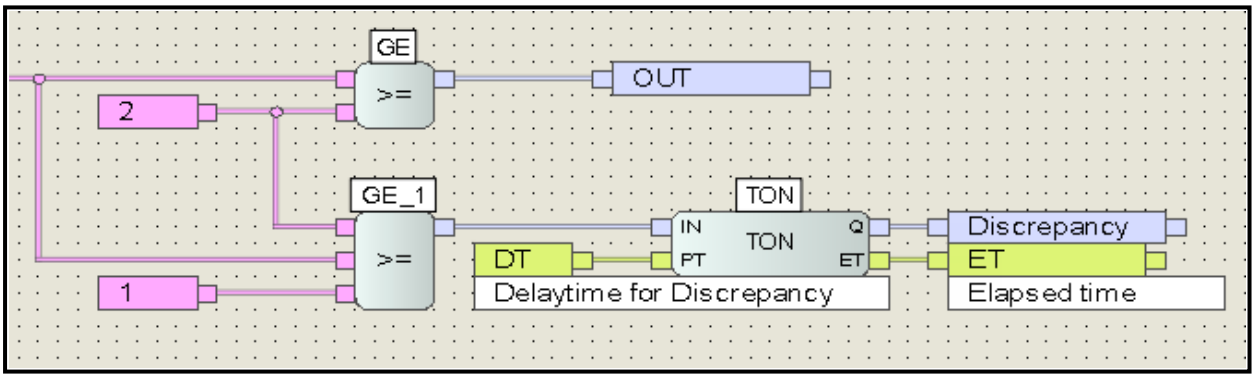


	Name	Type	Position IM	Position CG	Execution Order IM	Execution Order CG	
1	GE_1	Instance	Page: 0/0, Pos.: 82/22	Page: -/-, Pos.: -/-	5	-	Deleted
2	TON	Instance	Page: 0/0, Pos.: 98/22	Page: -/-, Pos.: -/-	6	-	Deleted

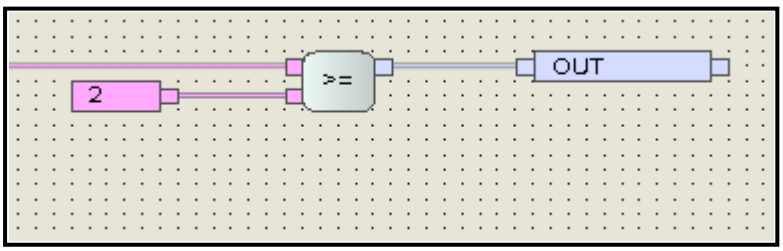
Double-click one of the lines to open the FBD Editor.

> The logic cannot be centered to deleted POU instances. Use the positions shown to find the locations at which logic elements were deleted.

Part of the logic of the imported configuration:

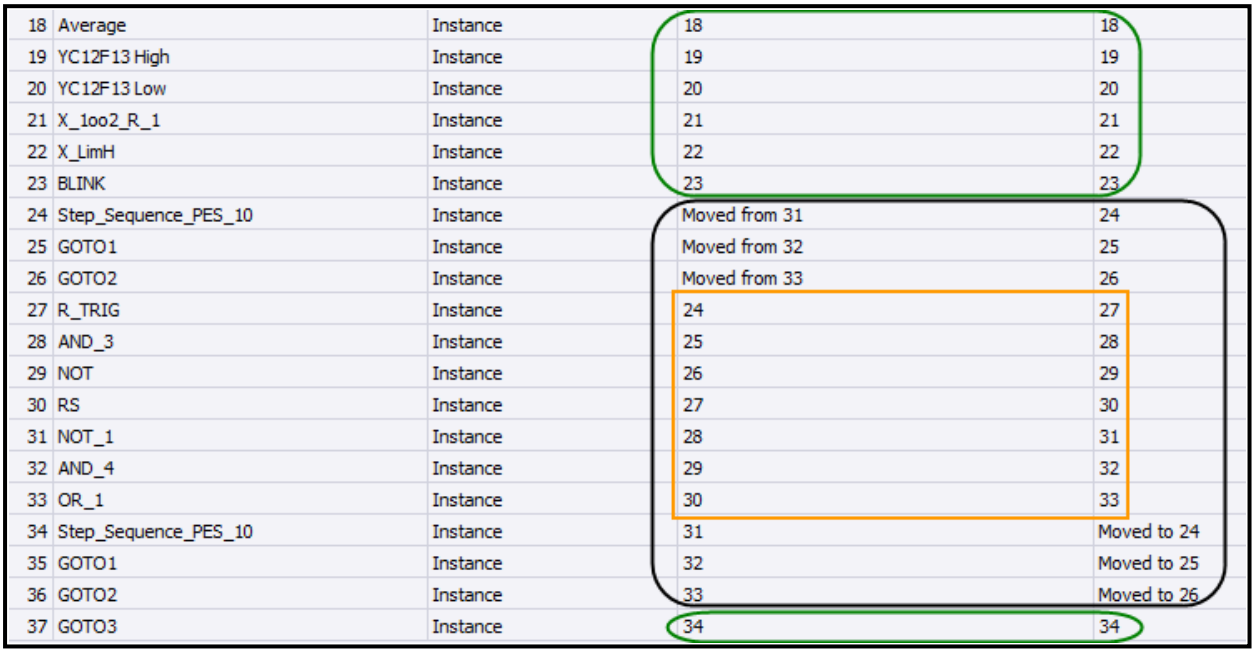


Part of the modified logic of the most recently generated configuration:



6.2.5 Moving a Network (Changing the Execution Order)

In the *POU Execution Order* tab, the details contained in the *Execution Order IM* and *Execution Order CG* columns indicate that the execution sequence of the POUs has changed. The position of all the remaining instances did not change.

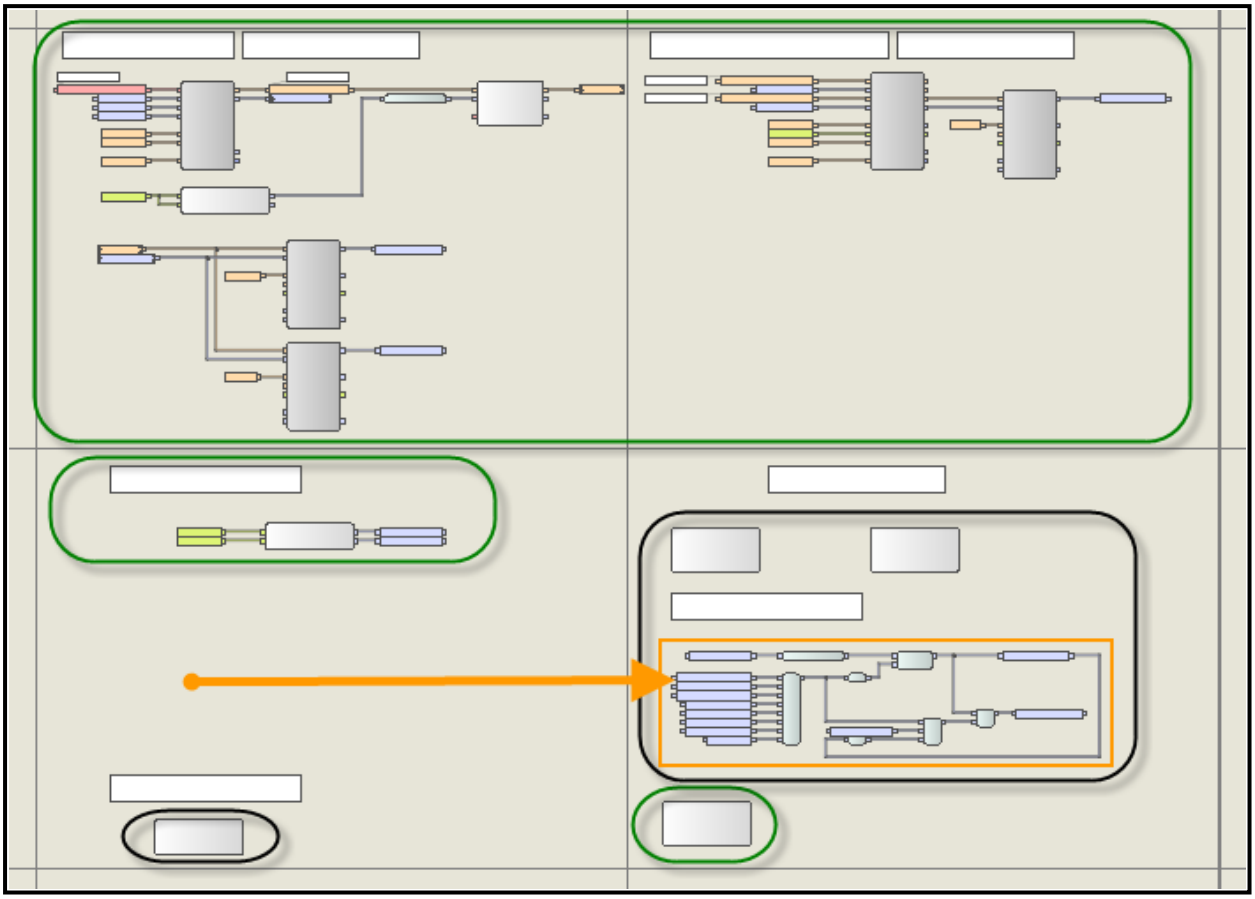


18	Average	Instance	18	18
19	YC12F13 High	Instance	19	19
20	YC12F13 Low	Instance	20	20
21	X_1oo2_R_1	Instance	21	21
22	X_LimH	Instance	22	22
23	BLINK	Instance	23	23
24	Step_Sequence_PES_10	Instance	Moved from 31	24
25	GOTO1	Instance	Moved from 32	25
26	GOTO2	Instance	Moved from 33	26
27	R_TRIG	Instance	24	27
28	AND_3	Instance	25	28
29	NOT	Instance	26	29
30	RS	Instance	27	30
31	NOT_1	Instance	28	31
32	AND_4	Instance	29	32
33	OR_1	Instance	30	33
34	Step_Sequence_PES_10	Instance	31	Moved to 24
35	GOTO1	Instance	32	Moved to 25
36	GOTO2	Instance	33	Moved to 26
37	GOTO3	Instance	34	34

SILworX attempts to keep the number of instances marked as *Moved* as small as possible. The result is that the instances shown as having been moved are not necessarily the ones that actually have been moved.

- The execution order for the objects marked in green in the screen above has not changed.
- The execution order for the objects marked in black in the screen above has changed.
- The objects that have actually moved are marked in orange.

The actually shifted logic for the example above



The consequences of the modified execution orders must be checked individually. Particular attention must be paid to modified write/read sequences of variables.
A modified execution order can lead to the new value of a changed variable only being recognized in the following cycle when reading the variable. This may affect the response times of the SIF (safety instrumented function).

For the instances marked as *Moved*, the check must be carried out for all of the variables that are read or written to by other parts of the logic. To do so, use the *cross-references* in SILworX.

The check can be limited to the modified area (marked in black in the example below). Pay special attention to logical connections. The functions must be checked in their entirety.

Variables of this logic with exclusive cross-references to hardware (I/O, system) or communication (Modbus, safe**ethernet** etc.) do not change their behavior as a result of a different execution order.

The instances shown as *Moved* surround the area to be examined like a frame, as the following examples demonstrate.

12	2oo3B CH789	Instance	12	12
13	2oo3	Instance	13	13
14	OR_2	Instance	14	14
15	BLINK	Instance	Moved from 23	15
16	X_Hx_AI	Instance	15	16
17	BLINK_1	Instance	16	17
18	R_TRIG_1	Instance	17	18
19	Average	Instance	18	19
20	YC12F13 High	Instance	19	20
21	YC12F13 Low	Instance	20	21
22	X_1oo2_R_1	Instance	21	22
23	X_LimH	Instance	22	23
24	BLINK	Instance	23	Moved to 15
25	R_TRIG	Instance	24	24
26	AND_3	Instance	25	25
27	NOT	Instance	26	26

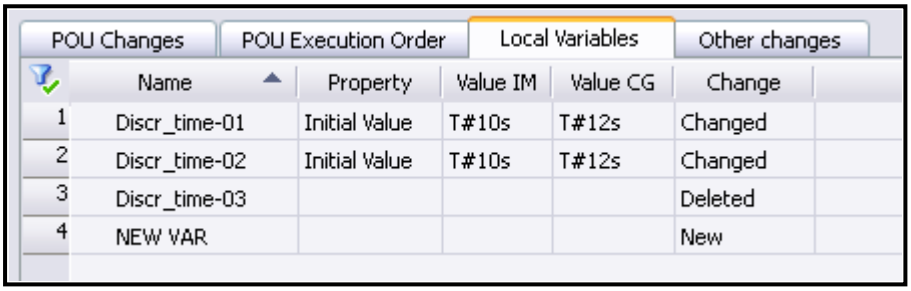
10	TON	Instance	10	10
11	2oo3B_CH456	Instance	11	11
12	2oo3B CH789	Instance	12	12
13	2oo3	Instance	13	13
14	OR_2	Instance	14	14
15	X_1oo2_R_1	Instance	Moved from 21	15
16	X_LimH	Instance	Moved from 22	16
17	X_Hx_AI	Instance	15	17
18	BLINK_1	Instance	16	18
19	R_TRIG_1	Instance	17	19
20	Average	Instance	18	20
21	YC12F13 High	Instance	19	21
22	YC12F13 Low	Instance	20	22
23	X_1oo2_R_1	Instance	21	Moved to 15
24	X_LimH	Instance	22	Moved to 16
25	BLINK	Instance	23	23
26	R_TRIG	Instance	24	24
27	AND_3	Instance	25	25

6.2.6 Changing Local Variables (new, delete, initial value)

In the *Local Variables* tab shown below, the information in the *Change* column indicates the following changes:

- The *NEW VAR* variable was added.
- The initial values of the *Discr_time-01* and *Discr_time-02* variables changed (see the *Value IM* and *Value CG* columns).
- The *Var_1* variable was deleted.

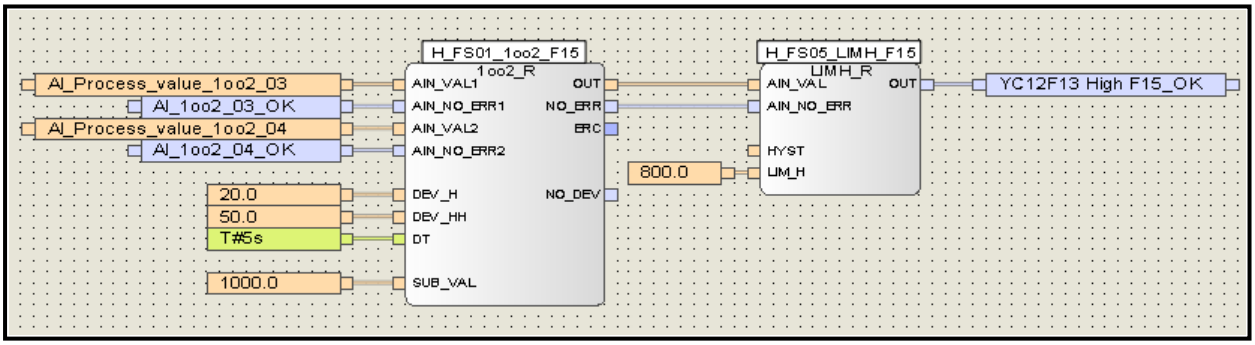
Double-click a line to open the FBD Editor and center the logic to the changed variable (this does not apply to deleted variables).



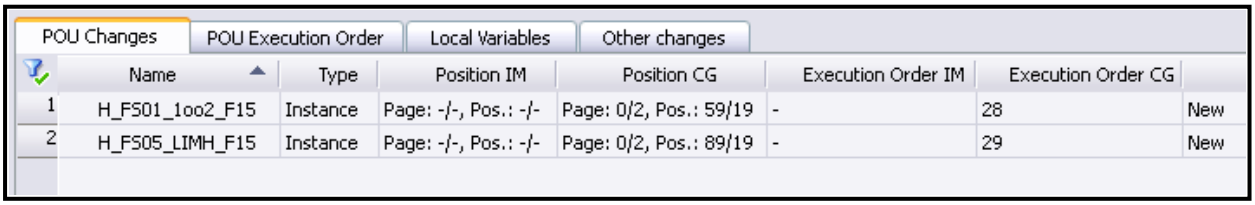
	Name	Property	Value IM	Value CG	Change
1	Discr_time-01	Initial Value	T#10s	T#12s	Changed
2	Discr_time-02	Initial Value	T#10s	T#12s	Changed
3	Discr_time-03				Deleted
4	NEW VAR				New

6.2.7 Creating New Networks

An existing logic is supplemented with the following programming and is then recompiled (CG):

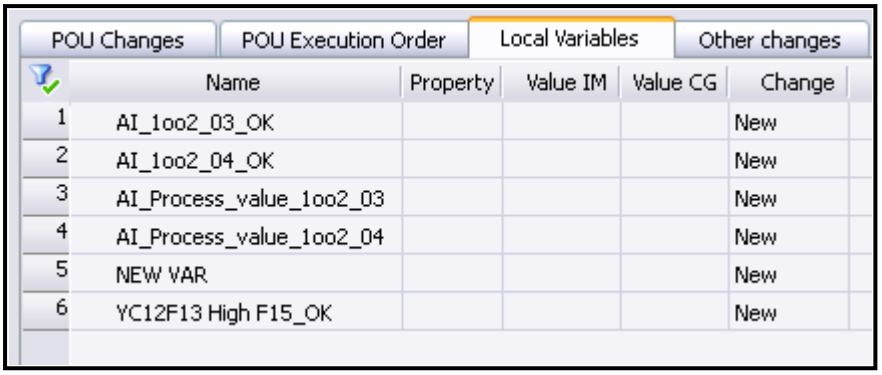


The empty *Position IM* and *Execution Order IM* columns in the *POU Changes* tab indicate that the POU instances *1002_F15* and *LIMH_15* do not exist in the imported configuration. The new, most recently generated configuration contains details on the *Position CG* and *Execution Order CG*. Additionally, the POU instances are marked as *New*.



	Name	Type	Position IM	Position CG	Execution Order IM	Execution Order CG	
1	H_FS01_1oo2_F15	Instance	Page: -/-, Pos.: -/-	Page: 0/2, Pos.: 59/19	-	28	New
2	H_FS05_LIMH_F15	Instance	Page: -/-, Pos.: -/-	Page: 0/2, Pos.: 89/19	-	29	New

In the *Local Variables* tab, the information specified in the *Change* column indicates that the three local variables are new.



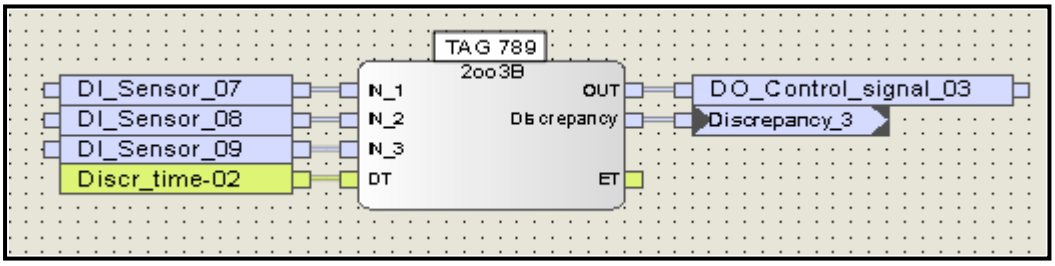
Double-click one of the lines to open the FBD Editor and center the logic to the selected POU.

6.2.8 Renaming a Function Block Instance

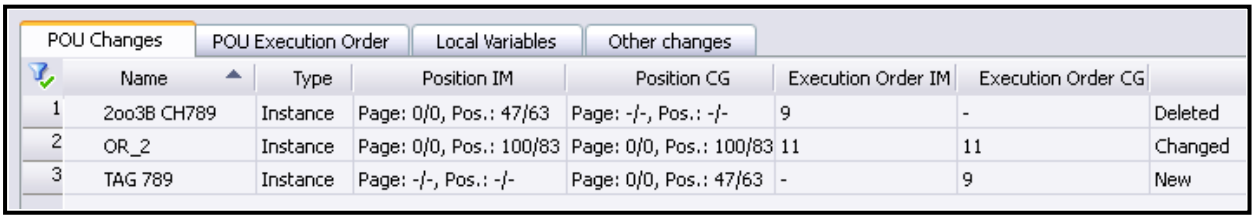
Renaming an instance is handled as if an instance is deleted and a new instance is added. The old instance name is deleted and the new instance name is added.

Caution! Hazard due to unintentional system shutdowns!
The renaming of function block instances leads to a reinitialization of all internal data of the function block when a reload is performed. Saved values and states are lost!

The *2oo3B_1* POU instance was renamed to *TAG 789*.



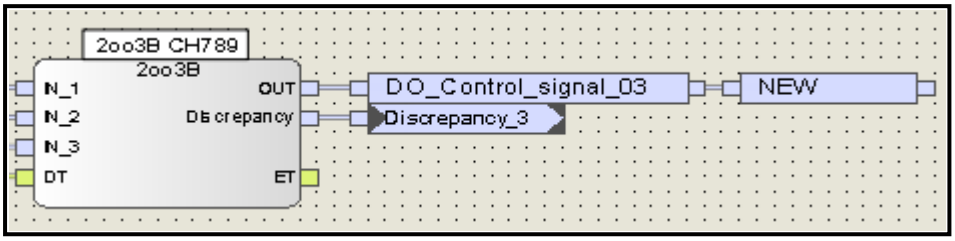
The empty *Position IM* and *Position CG* columns in the *POU Changes* tab and the additional *Deleted/New* information indicate the changes that have been made.



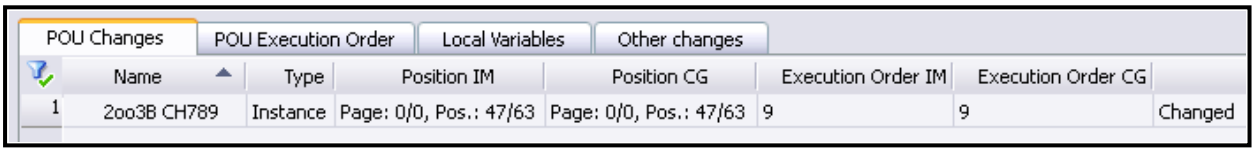
- The *2oo3B_1* POU instance was deleted.
- A new *TAG 789* POU instance was created.
- The new instance is located at the same position as the deleted instance and the execution order is identical.
- The *OR_2* instance is marked as *Changed*, since the *Discrepancy_3* connector is connected to the new *TAG 789* POU instance.

6.2.9 Assigning new Global Variables

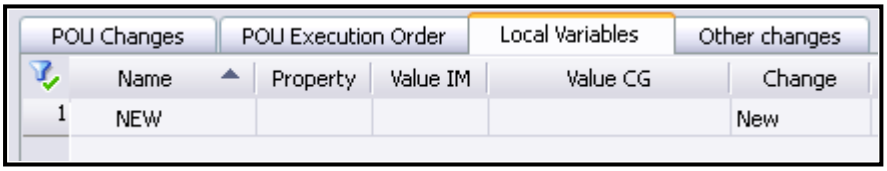
The global variable *NEW* was added to the most recently generated configuration and connected to the logic as shown.



In the *POU Changes* tab, the POU instance *2oo3B_1* is marked as *Changed*. Since variables have no instance name, the instance describing the variable is displayed.

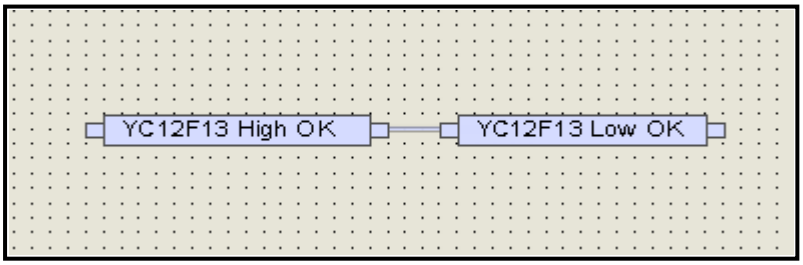


In the *Local Variables* tab, the variable *NEW* is marked as *New*.

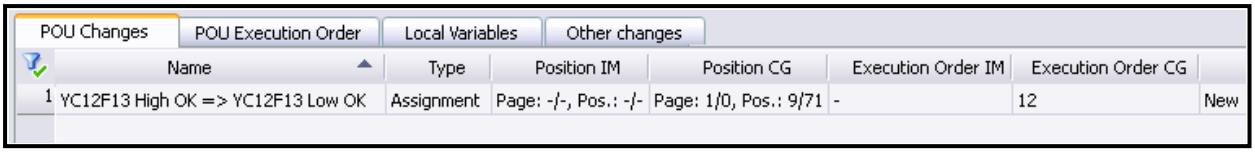


6.2.10 Adding new Variable Assignments

The change shown below was carried out in the most recently generated configuration. The variable *YC12F13 Low OK* was added and was assigned the value of the variable *YC12F13 High OK*.



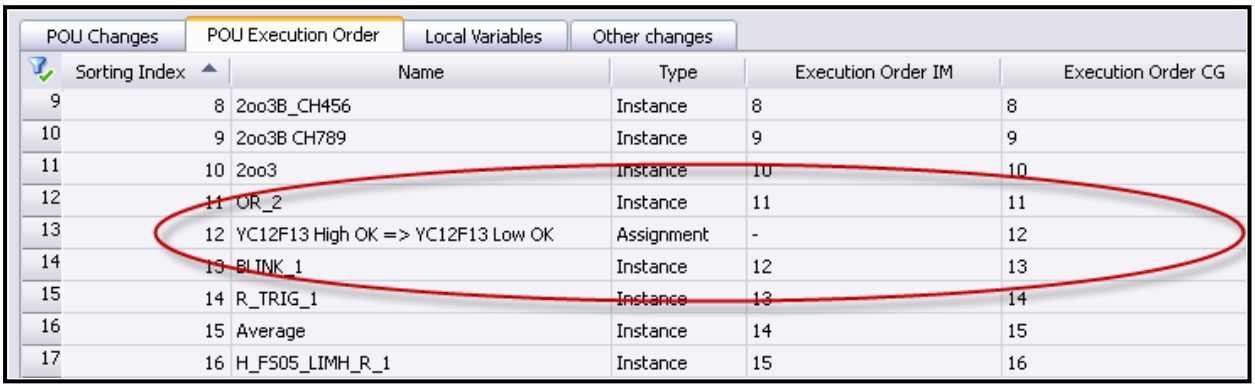
- In the *POU Changes* tab, the type of change is classified as *Assignment*.
- The direction of the assignment is indicated in the *Name* column.
- The blank *Position IM* and *Execution Order IM* columns indicate that the assignment did not exist in the imported configuration (IM).
- The new, most recently generated configuration (CG) contains details on the *Position CG* and *Execution Order CG*. Additionally, the assignment is marked as *New*.



POU Changes | POU Execution Order | Local Variables | Other changes

	Name	Type	Position IM	Position CG	Execution Order IM	Execution Order CG	
1	YC12F13 High OK => YC12F13 Low OK	Assignment	Page: -/-, Pos.: -/-	Page: 1/0, Pos.: 9/71	-	12	New

Changes in the execution order caused by an assignment are indicated in the *POU Execution Order* tab.



POU Changes | POU Execution Order | Local Variables | Other changes

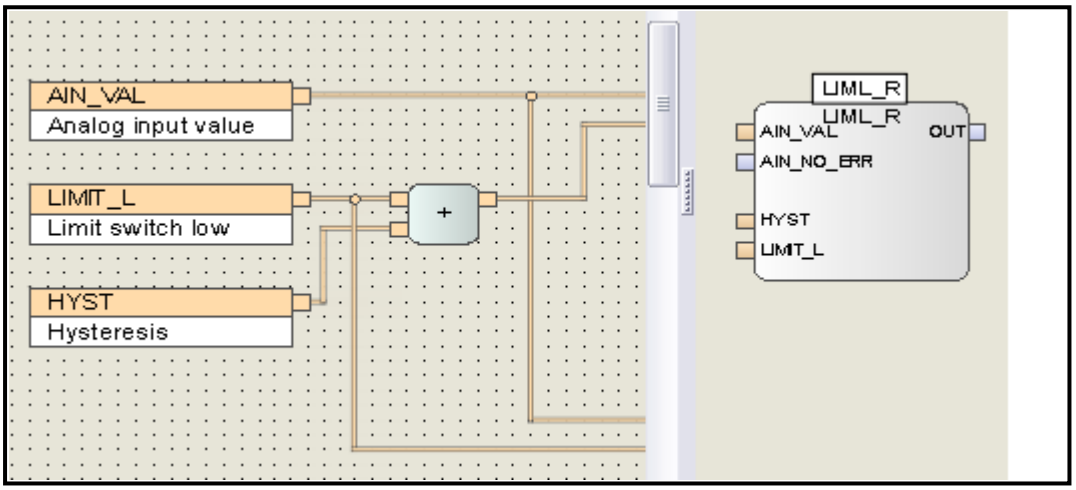
	Sorting Index	Name	Type	Execution Order IM	Execution Order CG
9	8	2oo3B_CH456	Instance	8	8
10	9	2oo3B CH789	Instance	9	9
11	10	2oo3	Instance	10	10
12	11	OR_2	Instance	11	11
13	12	YC12F13 High OK => YC12F13 Low OK	Assignment	-	12
14	13	BLINK_1	Instance	12	13
15	14	R_TRIG_1	Instance	13	14
16	15	Average	Instance	14	15
17	16	H_FS05_LIMH_R_1	Instance	15	16

6.2.11 Renaming Variables

Renaming a variable of VAR_INPUT type is handled as if the variable is deleted and a new variable is added. The previous variable is deleted and the new variable is added.

Caution! Hazard due to unintentional system shutdowns!
The renaming of existing variables leads to a reinitialization of these variables when a reload is performed. Saved values are lost!

In the following example, the input variable *LIM_L* was changed to *LIMIT_L*.



Renaming input variables does not result in a functional change of the logic. The newly generated */sys/ls/01_Program01.ldb* binary file is identical to the imported version. The checksums of the two versions are identical. Consequently, no functional rechecking is required.

Renaming the input variable, however, results in a changed checksum for the LIML_R function block type. This change is displayed in the version comparison.

Not all changes to the source code result in functional changes. Renaming a VAR input or VAR output results in a changed checksum (CRC) for the function block and the corresponding line in the version comparison is highlighted in red, but the program binary file does not change. This means that no functional change was made and no further check is necessary.

/sys/ls.config	Logic solver configuration	16#422dd7ed	V3	16#422dd7ed	V3	ok
/sys/ls/01_Program_PES_10.config	Program parameters	16#f385533d	V3	16#f385533d	V3	ok
/sys/ls/01_Program_PES_10.ldb	Program binary file	16#c0b12505	V2	16#c0b12505	V2	ok
01_Program_PES_10	Program	16#15595aaa	V2	16#17117c27	V2	-
1oo2_R	Function Block Type	16#665f1b8c	V2	16#665f1b8c	V2	ok
2oo3	Function Block Type	16#0c925672	V2	16#0c925672	V2	ok
2oo3B	Function Block Type	16#2a457e09	V2	16#2a457e09	V2	ok
Average	Function Block Type	16#020731ca	V2	16#020731ca	V2	ok
BLINK	Function Block Type	16#759416f5	V2	16#759416f5	V2	ok
BUFFER	Array	16#f0d16020	V2	16#f0d16020	V2	ok
Diag_DO24_01	Function Block Type	16#de5d274d	V2	16#de5d274d	V2	ok
Global Variables	Global Variables	16#dd70b312	V2	16#dd70b312	V2	ok
LIMH_R	Function Block Type	16#939cbd0b	V2	16#939cbd0b	V2	ok
LIML_R	Function Block Type	16#882b701d	V2	16#74f12d83	V2	-
Modul_Diag-PES10	Function Block Type	16#a9cb8161	V2	16#a9cb8161	V2	ok
Step Sequence_PES_10	Function Block Type	16#cb52c522	V2	16#cb52c522	V2	ok

Double-click the *LIML_R* line to display further details on the POU.

In the *Local Variables* tab, you can see that the variable *LIM_L* was renamed to *LIMIT_L*.

POU Changes | POU Execution Order | Local Variables | Oth

	Name	Property	Value IM	Value CG	Change
1	LIM_L				Deleted
2	LIMIT_L				New

The *ADD_1* and *LE_1* instances used in the *LIML_R* function block type are also marked as *Changed* in the *POU Changes* tab, since both instances are connected to the *LIMIT_L* input variable.

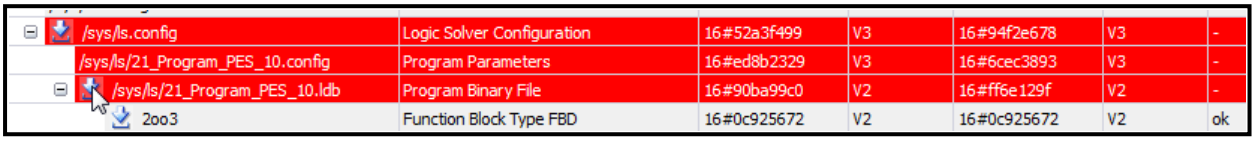
POU Changes | POU Execution Order | Local Variables | Other changes

	Name	Type	Position IM	Position CG	Execution Order IM	Execution Order CG	
1	ADD_1	Instance	Page: 0/0, Pos.: 41/23	Page: 0/0, Pos.: 41/23	0	0	Changed
2	LE_1	Instance	Page: 0/0, Pos.: 64/38	Page: 0/0, Pos.: 64/38	2	2	Changed

6.2.12 Special Changes in the Program.ldb File

As of SILworX V7, a detail view is available for the *Program.ldb* file. Here, changes are shown which do not result directly from POU changes, e.g., changes in the stack size or modified retain timer handling.

These changes must be verified by testing the affected objects.

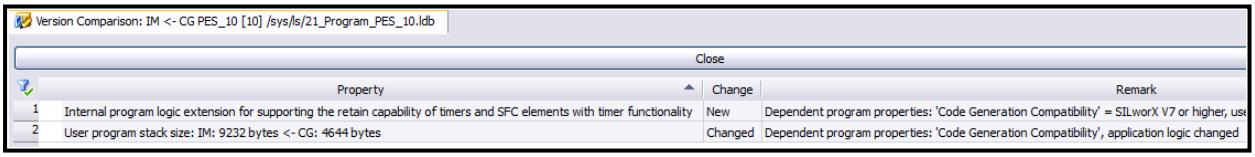


- If the parameter *Code Generation Compatibility* is changed from ≤ V3 to ≥ V4 in the program properties, the stack computation for user-defined data types changes (= error correction).
- If the parameter *Code Generation Compatibility* is changed from < V7 to ≥ V7 in the program properties, the retain-timer handling changes (= error correction).

> Basically, HIMA recommends retaining the *Code Generation Compatibility* for existing projects unless its change is necessary. This means that the CRC of the *Program.ldb* file remains unchanged and time-consuming post-testing can be avoided. For questions in this regard, please contact HIMA customer support.

Example

The following message is shown in the detail view of the *Program.ldb* as a result of the *Code Generation Compatibility* changing from V3 to ≥ V7.



- After the change, the program is ready for correct processing of retain timers.
- If there are retain timers and they are affected by the change, the */sys/ls/"Program"_retain.config* file changes. In this case, check the modified behavior of the retain timer.
- The computation of the stack requirements was changed. This requires a complete check of all of the logic parts of the affected program.

6.2.13 Changes in the Structured Text Logic

As of SILworX V6, you can also program logic in Structured Text (ST).

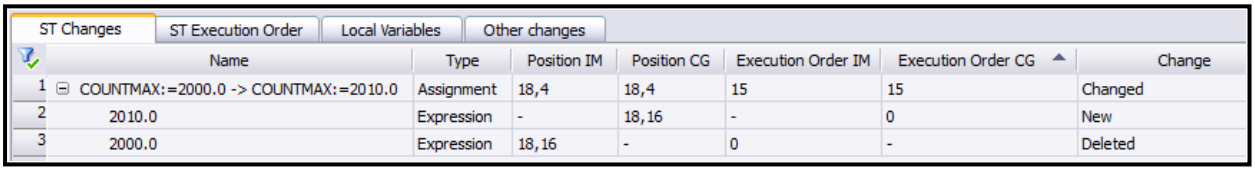
As a result of restrictions of the programming standard IEC 61131-3 and in accordance with IEC 61508/IEC 61511, SILworX Structured Text may be used as limited variability language (LVL) for programming safety-related logic.

Changes in a Structured Text POU are shown in detail in the version comparison.

/sys/ls/05_Program_ST.ldb	Program Binary File	16#25142913	V2	16#b587335e	V2	-
05_Program_ST	Program	16#181b6fc9	V2	16#181b6fc9	V2	ok
CYCLE TIME SIMULATOR	Function Block Type ST	16#96f8dbbb	V2	16#07b85f90	V2	-
Fahrenheit>Celsius	Function ST	16#ef614728	V2	16#ef614728	V2	ok

Double-click a line to display further details on the POU if there were changes (here: *CYCLE TIME SIMULATOR*).

Example 1: Changed Instructions

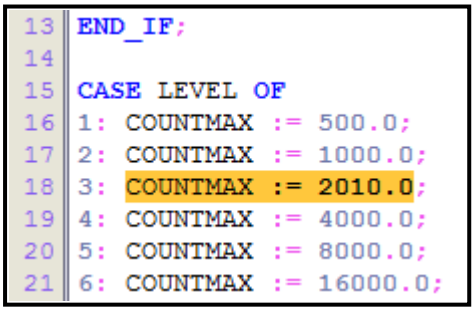


ST Changes | ST Execution Order | Local Variables | Other changes

	Name	Type	Position IM	Position CG	Execution Order IM	Execution Order CG	Change
1	COUNTMAX:=2000.0 -> COUNTMAX:=2010.0	Assignment	18,4	18,4	15	15	Changed
2	2010.0	Expression	-	18,16	-	0	New
3	2000.0	Expression	18,16	-	0	-	Deleted

The *Position IM* and *Position CG* columns list the objects by line and column. The specification "18,16" means line 18, column 16.

Double-click the modified assignment (line 1 in the screen above) to open the associated ST block in the ST Editor and the modified assignment is marked.



```
13 END_IF;
14
15 CASE LEVEL OF
16 1: COUNTMAX := 500.0;
17 2: COUNTMAX := 1000.0;
18 3: COUNTMAX := 2010.0;
19 4: COUNTMAX := 4000.0;
20 5: COUNTMAX := 8000.0;
21 6: COUNTMAX := 16000.0;
```

Example 2: Changed Execution Order

ST Changes | ST Execution Order | Local Variables | Other changes

	Sorting Index	Name	Type	Execution Order IM	Execution Order CG
1	0	TP(IN:=NOT TICK,PT:=INTERVALL,ET=>ET)	Statement	0	0
2	1	TICK:=TP.Q	Assignment	1	1
3	2	R_TRIG(CLK:=TP.Q)	Statement	2	2
4	3	IF R_TRIG.Q THEN	Statement	3	Moved to 6
5	4	IF R_TRIG.Q THEN	Statement	Moved from 3	6
6	5	LEVEL:=LEVEL+1	Assignment	4	Moved to 7
7	6	END_IF	Statement	5	5
8	7	IF LEVEL=17 THEN	Statement	6	3
9	8	LEVEL:=1	Assignment	7	4
10	9	LEVEL:=LEVEL+1	Assignment	Moved from 4	7
11	10	END_IF	Statement	8	8
12	11	CASE LEVEL OF	Statement	9	9

The execution order of the imported and the newly generated configurations can be seen in the *Execution Order IM* and *Execution Order CG* columns.

The execution order corresponds to the sequence of the actual instructions without blank lines or comments, and cannot be displayed in the logic.

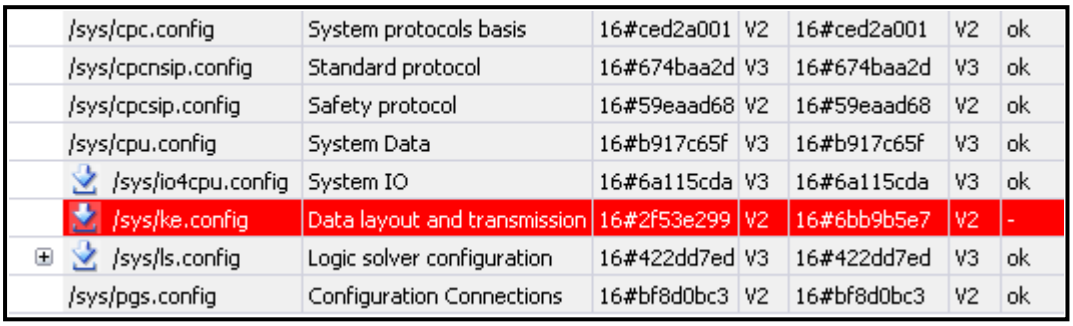
```
1
2  TP(IN := NOT TICK , PT := INTERVA
3  (* Alternative: TP(IN := NOT TICK
4  TICK := TP.Q;
5  R_TRIG(CLK :=TP.Q); (* also R_TRI
6
7  IF R_TRIG.Q THEN (* also IF R.TRI
8  LEVEL := LEVEL + 1;
9  END_IF;
10
11 IF LEVEL = 17 THEN
12 LEVEL := 1;
13 END_IF;
14
15 CASE LEVEL OF
16 1: COUNTMAX := 500.0;
17 2: COUNTMAX := 1000.0;
```

```
1
2  TP(IN := NOT TICK , PT := INTERVA
3  (* Alternative: TP(IN := NOT TICK
4  TICK := TP.Q;
5  R_TRIG(CLK :=TP.Q); (* also R_TRI
6
7  IF LEVEL = 17 THEN
8  LEVEL := 1;
9  END_IF;
10
11 IF R_TRIG.Q THEN (* also IF R.TRI
12 LEVEL := LEVEL + 1;
13 END_IF;
14
15 CASE LEVEL OF
16 1: COUNTMAX := 500.0;
17 2: COUNTMAX := 1000.0;
```

6.3 Modifying the Assignment of Global Variables

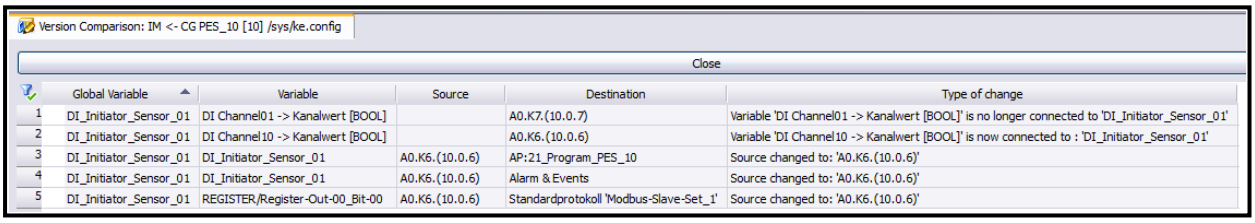
6.3.1 Assigning Global Variables to Another Hardware Input

Modifying the assignment of global variables (new source, new destination, modified initial values) always affects the */sys/ke.config* configuration file for reading and writing global variables.



	/sys/cpc.config	System protocols basis	16#ced2a001	V2	16#ced2a001	V2	ok
	/sys/cpcnsip.config	Standard protocol	16#674baa2d	V3	16#674baa2d	V3	ok
	/sys/cpcsip.config	Safety protocol	16#59eaad68	V2	16#59eaad68	V2	ok
	/sys/cpu.config	System Data	16#b917c65f	V3	16#b917c65f	V3	ok
	/sys/io4cpu.config	System IO	16#6a115cda	V3	16#6a115cda	V3	ok
	/sys/ke.config	Data layout and transmission	16#2f53e299	V2	16#6bb9b5e7	V2	-
⊞	/sys/ls.config	Logic solver configuration	16#422dd7ed	V3	16#422dd7ed	V3	ok
	/sys/pgs.config	Configuration Connections	16#bf8d0bc3	V2	16#bf8d0bc3	V2	ok

Double-click the *sys/ke.config* line to display further details on the configuration file.



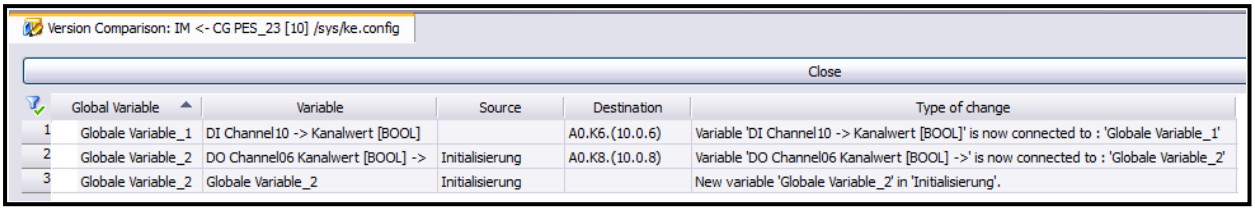
Version Comparison: IM <- CG PES_10 [10] /sys/ke.config

Close

	Global Variable	Variable	Source	Destination	Type of change
1	DI_Initiator_Sensor_01	DI Channel01 -> Kanalwert [BOOL]		A0.K7.(10.0.7)	Variable 'DI Channel01 -> Kanalwert [BOOL]' is no longer connected to 'DI_Initiator_Sensor_01'
2	DI_Initiator_Sensor_01	DI Channel10 -> Kanalwert [BOOL]		A0.K6.(10.0.6)	Variable 'DI Channel10 -> Kanalwert [BOOL]' is now connected to : 'DI_Initiator_Sensor_01'
3	DI_Initiator_Sensor_01	DI_Initiator_Sensor_01	A0.K6.(10.0.6)	AP:21_Program_PES_10	Source changed to: 'A0.K6.(10.0.6)'
4	DI_Initiator_Sensor_01	DI_Initiator_Sensor_01	A0.K6.(10.0.6)	Alarm & Events	Source changed to: 'A0.K6.(10.0.6)'
5	DI_Initiator_Sensor_01	REGISTER/Register-Out-00_Bit-00	A0.K6.(10.0.6)	Standardprotokoll 'Modbus-Slave-Set_1'	Source changed to: 'A0.K6.(10.0.6)'

Line	Description
1	For channel 10 of the module with the SRS `10.0.6`, the *Channel Value* parameter was reconnected to the global variable *DI_Initiator_Sensor_01*.
2	For channel 01 of the module with the SRS `10.0.7`, the parameter *Channel Value* is no longer connected to the global variable *DI_Initiator_Sensor_01*.
3	The source of the global variable *DI_Initiator_Sensor_01* defined as *Alarm & Events* has changed and is now the module with the SRS `10.0.6`.
4	The source of the global variable *DI_Initiator_Sensor_01* used in the user program *21_Program_Pes_10* has changed and is now the module with the SRS `10.0.6`.
5	The source of the global variable *DI_Initiator_Sensor_01* that was transmitted by the standard protocol *Modbus-Slave-Set_1* in the *Register-Out_Bit-00* has changed and is now the module with the SRS `10.0.6`.

Example: HIMatrix



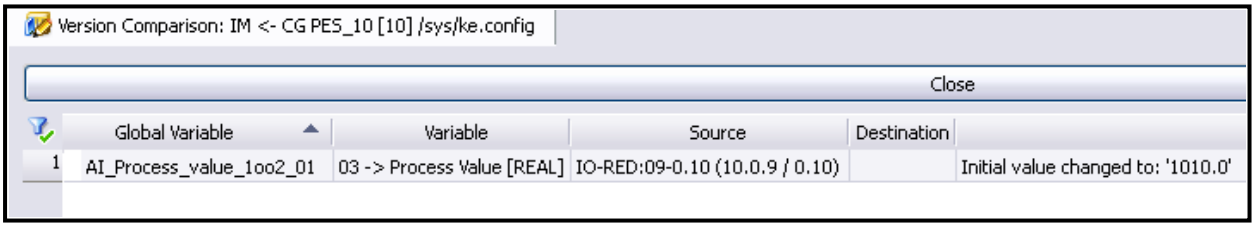
The version comparison takes all the effects of the change into account. The users must decide which changes they want to check.

> Even if the logic was not changed, the program might behave differently if, for instance, a variable edited there now has a different source.

6.3.2 New Initial Value for a Global Variable

The initial value of a global variable can be set or changed in the Global Variable Editor. This information is saved in the */sys/ke.config* configuration file. Changing one or more initial values also has an effect on the */sys/ke.config* file.

Double-click the */sys/ke.config* line to display further details on the global variable. The following picture shows that the initial value of the global variable *AE_Process_Value_Channel_03* was changed to `1010.0`.



6.4 Safety-Relevant and Non-Safety-Relevant

SILworX provides the option of dividing system tasks into several different programs and thereby of creating a separation between safety-relevant logic elements (e.g., ESD functions) and non-safety-relevant logic elements (e.g., data preparation for a control system).

> A separation of safety-relevant and non-safety-relevant logic into different programs simplifies the display of changes and reduces the effort entailed in required retesting. HIMA recommends also making this distinction for the global variables, e.g., by means of appropriate naming conventions. Further information on this can be found in IEC 61511 Part 1, Chapter 12.

The version comparison allows changes made to the logic to be identified. If changes were made to the safety-relevant logic, this will generally require safety-relevant retesting.

In the following screen, the version comparator detected a change in the binary file of */sys/ls/01_Program_01.ldb*. A change was made to the *2oo3B* function block type. This POU must be tested.

However, in the binary file of */sys/ls/02_Program02*, the version comparison did not detect any changes.

/sys/ls.config	Logic solver configuration	16#79638bbc	V3	16#ad79a4dd	V3	-
/sys/ls/Programm01.config	Program parameters	16#5936c3f5	V3	16#5936c3f5	V3	ok
/sys/ls/Programm01.ldb	Program binary file	16#c0b12505	V2	16#48949b97	V2	-
1oo2_R	Function Block Type	16#665f1b8c	V2	16#665f1b8c	V2	ok
2oo3	Function Block Type	16#0c925672	V2	16#0c925672	V2	ok
2oo3B	Function Block Type	16#2a457e09	V2	16#6985693e	V2	-
Average	Function Block Type	16#020731ca	V2	16#020731ca	V2	ok
BLINK	Function Block Type	16#759416f5	V2	16#759416f5	V2	ok
BUFFER	Array	16#f0d16020	V2	16#f0d16020	V2	ok
Diag_DO24_01	Function Block Type	16#de5d274d	V2	16#de5d274d	V2	ok
Global Variables	Global Variables	16#dd70b312	V2	16#dd70b312	V2	ok
LIMH_R	Function Block Type	16#939cbd0b	V2	16#939cbd0b	V2	ok
LIML_R	Function Block Type	16#74f12d83	V2	16#74f12d83	V2	ok
Modul_Diag-PES10	Function Block Type	16#a9cb8161	V2	16#a9cb8161	V2	ok
Programm01	Program	16#17117c27	V2	16#17117c27	V2	ok
Step Sequence_PES_10	Function Block Type	16#cb52c522	V2	16#cb52c522	V2	ok
System-Monitoring_PES10	Function Block Type	16#e2f6c48d	V2	16#e2f6c48d	V2	ok
/sys/ls/Programm01_force.config	Application force data	16#3b01122a	V2	16#3b01122a	V2	ok
/sys/ls/Programm01_retain.config	Application retain data	16#af3502b3	V2	16#af3502b3	V2	ok
/sys/ls/Programm02.config	Program parameters	16#85f48dfd	V3	16#85f48dfd	V3	ok
/sys/ls/Programm02.ldb	Program binary file	16#ce11e72d	V2	16#ce11e72d	V2	ok
/sys/ls/Programm02_force.config	Application force data	16#fd339f20	V2	16#fd339f20	V2	ok
/sys/ls/Programm02_retain.config	Application retain data	16#afa9e6ec	V2	16#afa9e6ec	V2	ok

Changes to the assignment of global variables (new source, new destination) require particular attention. They do not necessarily lead to a change in syntax and therefore to a change in the program binary file. However, these changes may have safety relevance! Changes to the global variables are described in the detail view of the *ke.config* configuration file; see *Assigning Global Variables to Another Hardware Input*.

If global variables of multiple programs are read and changes are made to the *ke.config* configuration file, the programs must be checked individually.

The cross-reference list displayed in the Global Variable Editor can be used, for instance, to determine if a variable that was changed in the hardware assignment is written to by a safety-relevant or a non-safety-relevant program. If the variable is used read-only by a safety-relevant program, the change is always safety-relevant!

The safety relevance of changes to other central configuration files (e.g., module configuration files, system settings, etc.) must also be checked on an individual basis. Because the changes to these central configuration files have no direct relevance to the programs, the distinction made between *safe and non-safe* programs is irrelevant in this context.

6.5 Memory Overview for Code and Data

Double-click the */sys/ls.config* line to open the memory overview.

- In a HIMax system, either 5 MB (for X-CPU 31) or 10 MB (for X-CPU 01) memory is available for program code and data, depending on the used CPU module. If several programs are processed in one controller, the memory is fragmented.
- In a HIMatrix system, 5 MB memory is available.

/sys/ls.config	Logic solver configuration	16#79638bbc	V3	16#ad79a4dd	V3	-
/sys/ls/Programm01.config	Program parameters	16#5936c3f5	V3	16#5936c3f5	V3	ok
/sys/ls/Programm01.ldb	Program binary file	16#c0b12505	V2	16#48949b97	V2	-
1oo2_R	Function Block Type	16#665f1b8c	V2	16#665f1b8c	V2	ok

6.5.1 Example of Memory Overview Based on HIMax

The memory overview shows the load altogether and for each individual program. The percentages refer to the total memory (in the screen below: HIMax with

X-CPU 01).

Version Comparison: IM <- DL PES_10 [10] /sys/ls.config

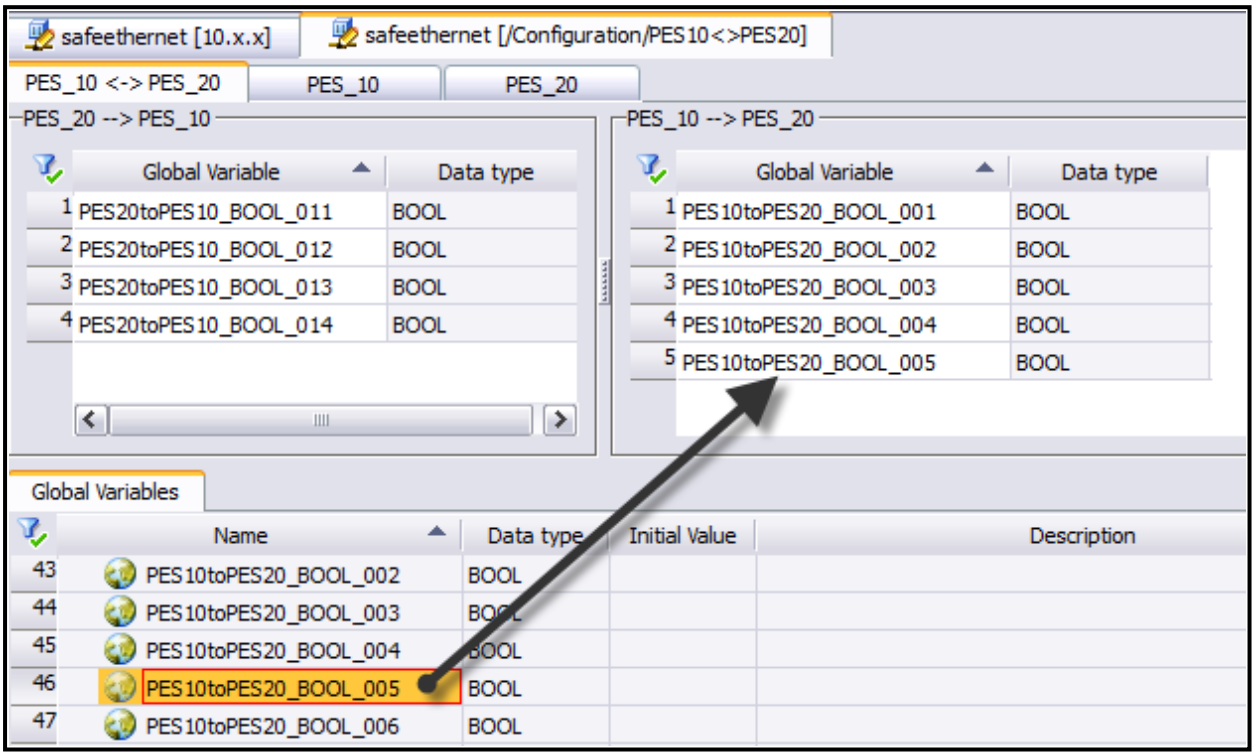
Close

	Info	Info on last download	Info on import configuration	max.	
1	|Sum				
2	Code Size	115464(1%)	115464(1%)	10481664	-
3	Data Size	16784(0%)	16784(0%)	10481664	-
4	Retain Data Size	8(0%)	8(0%)	32768	-
5	Programm01	-	-		-
6	Code Size	114592(1%)	114592(1%)	10481664	ok
7	Data Size	12552(0%)	12552(0%)	10481664	ok
8	ID	1	1		-
9	Retain Data Size	8(0%)	8(0%)	32768	ok
10	Programm02	-	-		-
11	Code Size	872(0%)	872(0%)	10481664	ok
12	Data Size	4232(0%)	4232(0%)	10481664	ok
13	ID	2	2		-
14	Retain Data Size	0(0%)	0(0%)	32768	ok

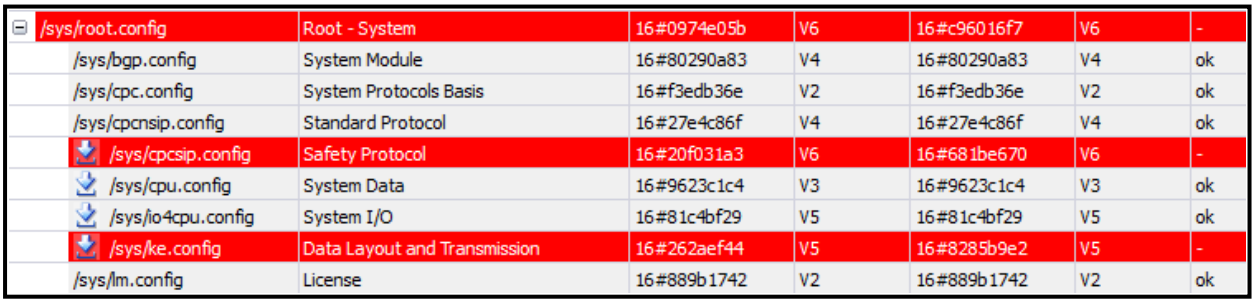
6.6 Changes in safeethernet Communication

6.6.1 Add new Variable to Existing Connection

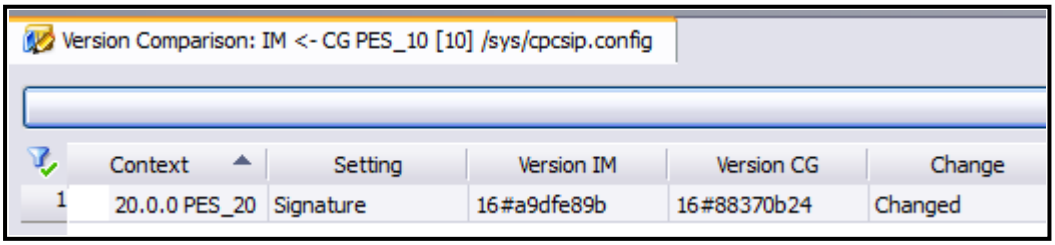
In the following example, the variable *PES10toPES20_BOOL_005* is added to a safe**ethernet** connection.



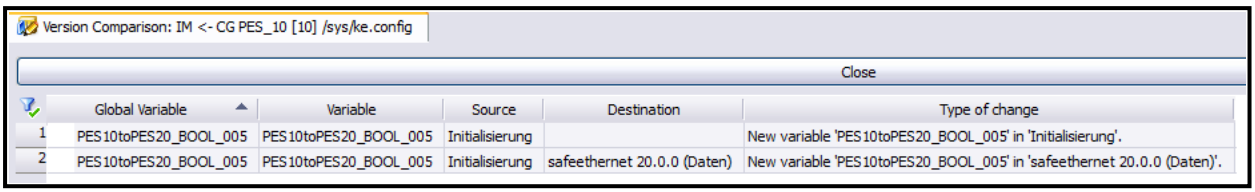
This change is shown in the version comparison as follows:



Double-click the */sys/cpcsip.config* line to open the detail view.



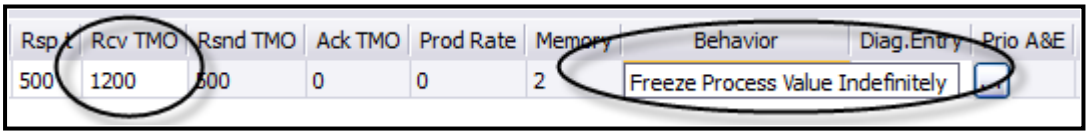
The signatures (CRC) of the safety-related data have changed! This is due to the modified data which is shown in detail in the *ke.config* file.



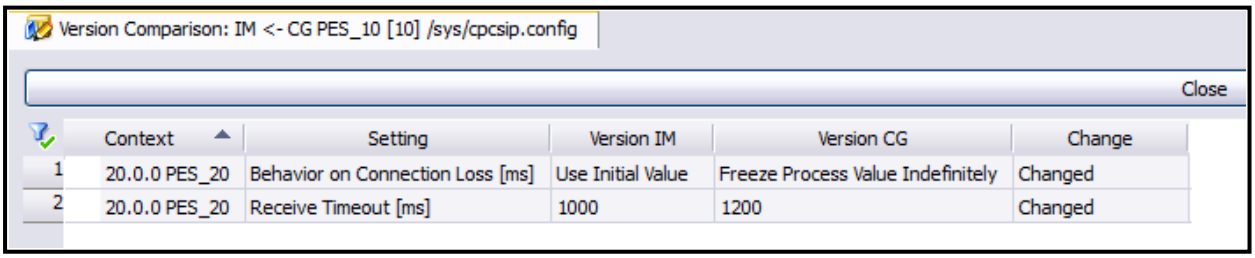
Line	Description
1	The variable *PES10toPES20_BOOL_005* is being used for the first time and must therefore be initialized.
2	The variable *PES10toPES20_BOOL_005* is written to the data area of the safe**ethernet** connection 20.0.0. Because the safe**ethernet** connection is the target, the variable is sent to the partner. Connection `20.0.0` stands for the following parameters: • System ID of the partner: 20 • Rack ID of the partner: 0 • Connection ID: 0

6.6.2 safeethernetParameter Changes

In the following example, the Receive Timeout and the setting for *Behavior* were changed. The version comparison recognizes the changes in the */sys/cpcsip.config* file.



Double-click the */sys/cpcsip.config* line to open the detail view in which the modified parameters can be seen.



7 Printout of the Comparison Information

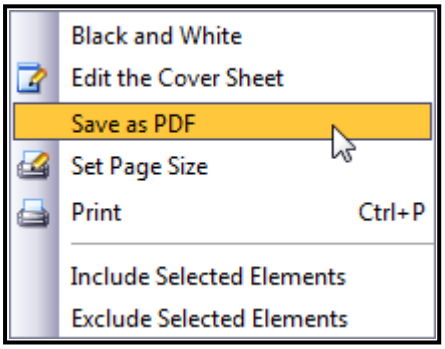
The printout of the comparison information is performed with the help of the documentation editor and serves as a record of the changes. A complete, written record of the changes usually consists of three parts:

1. Printout of the comparison information as a result of the version comparison.
2. Printout of the logic (objects) after the change.
3. Printout of the logic (objects) before the change.

The values before and after the change are shown automatically for changes of parameters in tables (e.g., hardware) or properties dialogs. An additional printout that would once again document these states is therefore not usually required.

Instead, only the location of the change is shown in the logic. In this case, it is necessary for the states before and after the change to be printed out!

In SILworX, the documentation can either be output as a paper printout or be saved as a PDF file.



The result of the version comparison carried out most recently is always used for the documentation. Consequently, it is important that the version comparison be carried out first, before documentation editor is opened!

The printout contains the same information that is displayed on the monitor (What You See Is What You Get).

7.1 Preparing the Printout

HIMA recommends carrying out the steps shown below in the stated sequence in order to document changes to a project. No further changes (offline) are permitted between the individual steps. Otherwise, the printout would also document the undesired changes.

1. Prepare the changes offline.
2. Perform the code generation.
3. Have the changes approved in accordance with existing modification procedures.
   Important: The changes are to be loaded and tested only after approval has been obtained. Proceed with **step 5** to create the record for the approval.
4. Load approved changes to the PES.

> ! According to operative modification procedures as per the technical standards for functional safety, an **effect analysis and the approval of the planned changes** are required before the changes are loaded to the PES. To do so, step 4 is omitted and a documented comparison between the newly generated configuration and the loaded version must be carried out.

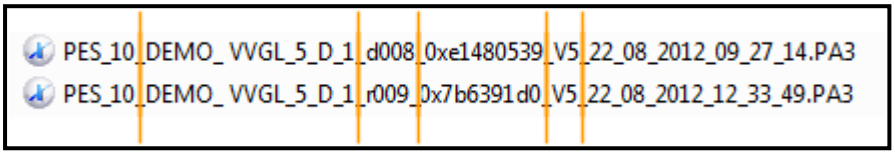
5. Perform the version comparison. Afterwards, carry out the evaluation on the monitor as described in the previous chapters.
6. Launch the documentation editor and edit the cover sheet. For example, enter the CRC.
7. Select the pages of the comparison documentation which are to be printed.
8. Select the changed pages (logic, hardware, etc.). The following chapters provide more information on this.
9. Print out the document.

> i The final record of the changes made that is to be presented to the approval authorities can be created by carrying out the steps mentioned above once again but omitting step 3. This creates a final documentation of the changes as carried out, loaded and tested.

7.2 Referencing to the Project Archive

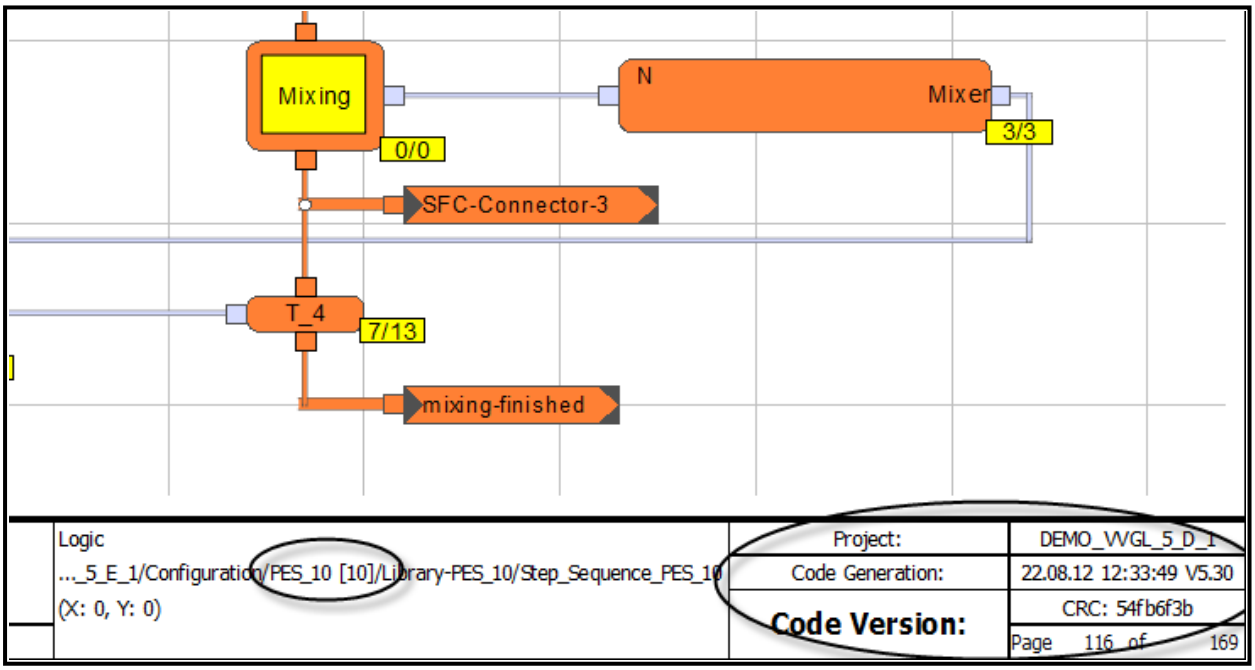
As of SILworX V5, it is possible to automatically save a project archive for every loading operation to a controller. This project archive contains the most recently loaded configuration files.

The name of the project archive contains the following information:



1. Resource name
2. Project name
3. Load number (d = download, r = reload)
4. Resource CRC
5. SILworX version
6. Date and time of the code generation

The revision status of the configuration that is saved in the project archive cannot be modified and is therefore the ideal reference for the documentation of the printout. How to use the revision status in the printout is explained in *Filling in the Cover Sheet*.



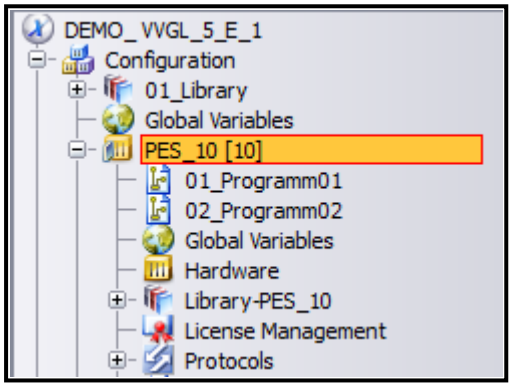
7.3 Generating Documentation (Printout)

The following sections describe in chronological order the steps which must be carried out to create project documentation.

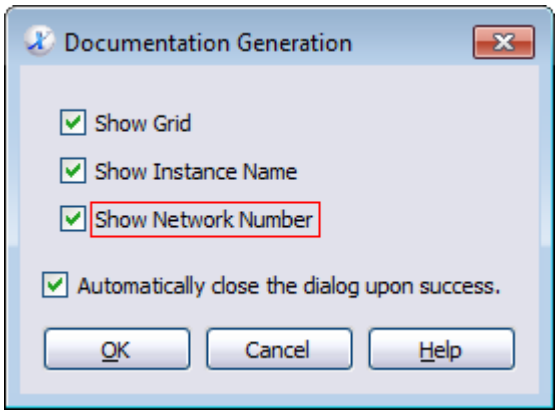
7.3.1 Starting the Documentation

Before starting the documentation editor, first select the desired resource in the structure tree. This selects all pages of this resource for printing as the initial selection.

➤ Select the desired resource in the structure tree and click the **Documentation** button on the Action Bar.

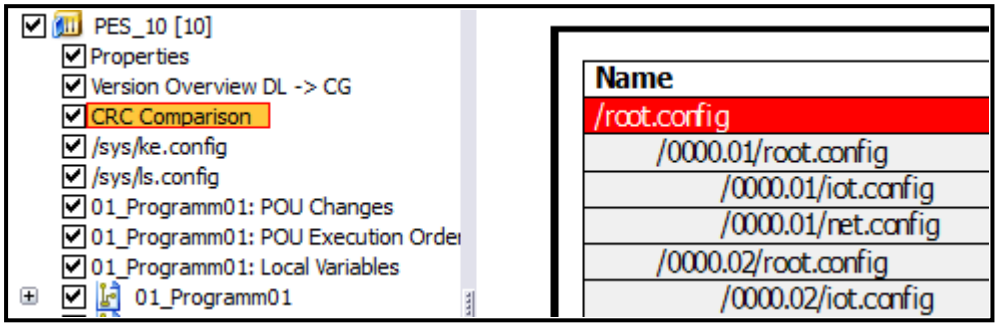


➤ In the *Documentation Generation Parameters* dialog box, activate all of the options if the printout is to serve the purpose of documenting changes.



7.3.2 Filling in the Cover Sheet

In the documentation editor, open the desired resource and select the *CRC Comparison* element.



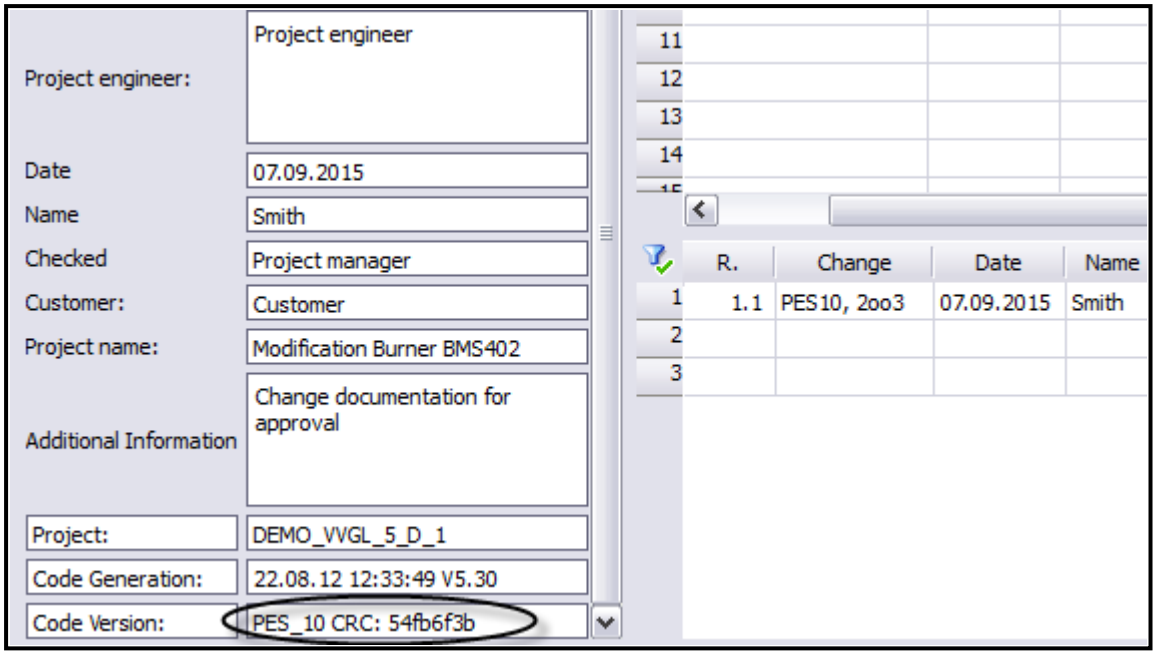
Read out the current CRC.

In the example below, the change (CRC CG) is to be presented for approval after the code has been generated.

	CRCDL	Version DL	CRCCG	Version CG
	16#7b6391d0	V4	16#54fb6f3b	V4
	16#239441f4	V3	16#239441f4	V3
	16#209c5b8f	V3	16#209c5b8f	V3

In the **Documentation** menu, select the **Edit the Cover Sheet** function. This opens the cover sheet template.

Enter all of the project particulars on the cover sheet, such as processed by, comments and CRC.



Project engineer:	Project engineer
Date	07.09.2015
Name	Smith
Checked	Project manager
Customer:	Customer
Project name:	Modification Burner BMS402
Additional Information	Change documentation for approval
Project:	DEMO_VVGL_5_D_1
Code Generation:	22.08.12 12:33:49 V5.30
Code Version:	PES_10 CRC: 54fb6f3b

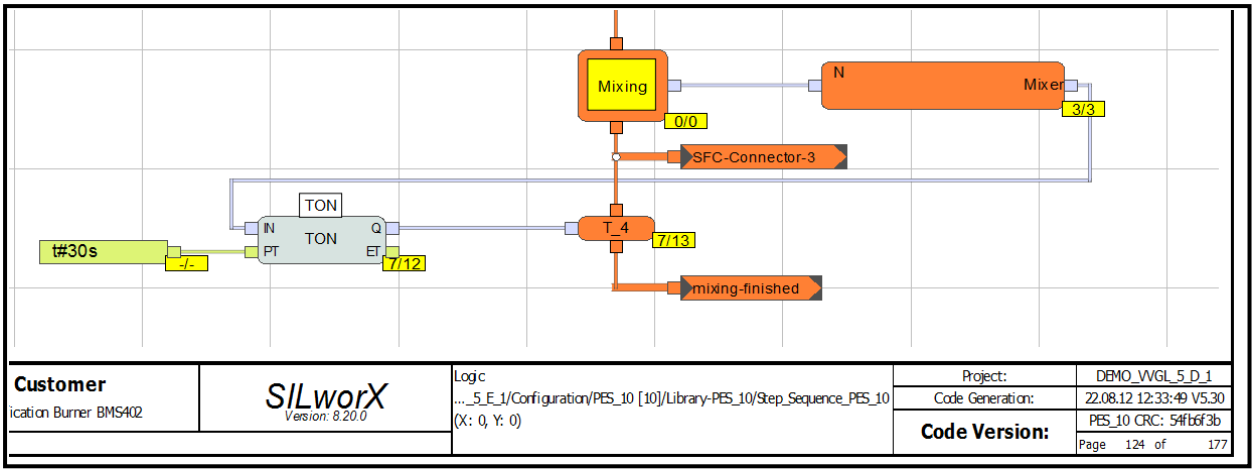
	R.	Change	Date	Name
1	1.1	PES10, 2oo3	07.09.2015	Smith
2				
3				

Save the modified cover sheet.

This area will appear later on every printed page.

	16#c4244af2	V3	16#c4244af2	V3	ok
	16#fdf80e3f	V4	16#fdf80e3f	V4	ok
	16#4039051b	V2	16#86615d08	V2	-
	16#889b1742	V2	16#889b1742	V2	ok
	16#ad5a6f01	V3	16#b45e0929	V3	-
	16#5d743564	V3	16#4674ceeb	V3	-
	16#1c373749	V2	16#16be0ebf	V2	-
	16#665f1b8c	V2	16#665f1b8c	V2	ok
	16#e3059f4f	V2	16#ca71ec2b	V2	-
	16#793e4866	V2	16#793e4866	V2	ok
	16#1e4e7f96	V2	16#1e4e7f96	V2	ok
	16#830d0f29	V2	16#830d0f29	V2	ok
	16#7c7eda91	V2	16#7c7eda91	V2	ok
	16#f0d16020	V2	16#f0d16020	V2	ok
	16#3c68356e	V2	16#3c68356e	V2	ok

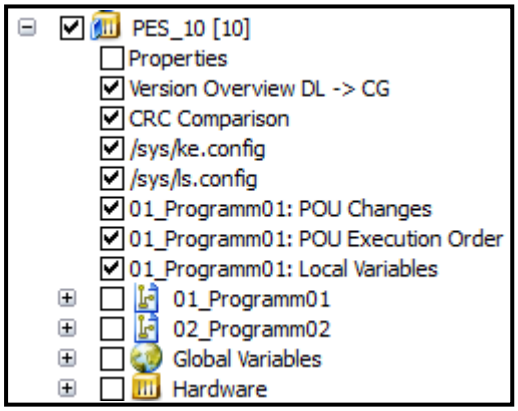
	CRC Comparison /DEMO_ VVGL_5_E_1/Configuration/PES_10 [10]	Project:	DEMO_VVGL_5_D_1
		Code Generation:	22.08.12 12:33:49 V5.30
		Code Version:	PES_10 CRC: 54fb6f3b
			Page 58 of 177



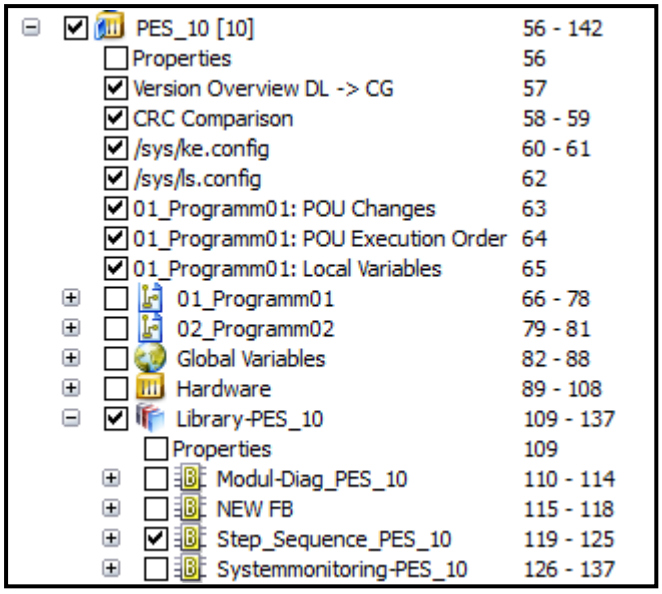
7.3.3 Selecting Objects (Pages) to be Printed

The quantity and choice of the objects (pages) to be printed depends on the results of the version comparison.

Always select the complete version comparison for the printout.
Normally, this includes all objects between *Properties* and the next group; in the example below, *01_Program01*. The list is sorted according to the alphabetical sequence of the objects.

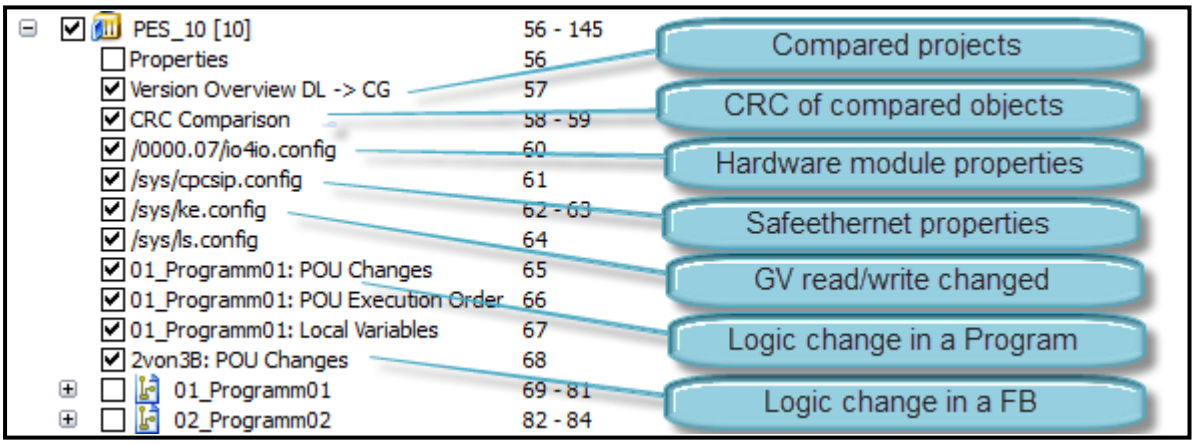


Also select the changed objects, in the example, the POU *Step_Sequence_PES_10*.



7.3.3.1 Overview of the Most Important Documents in the Version Comparison

The following screen provides an overview of the most important documents in the version comparison.



7.4 Documentation of Changed Objects

When making changes in the logic, bear in mind that function blocks and functions may be stored in higher-order libraries. Elements of these libraries can be used in different resources.

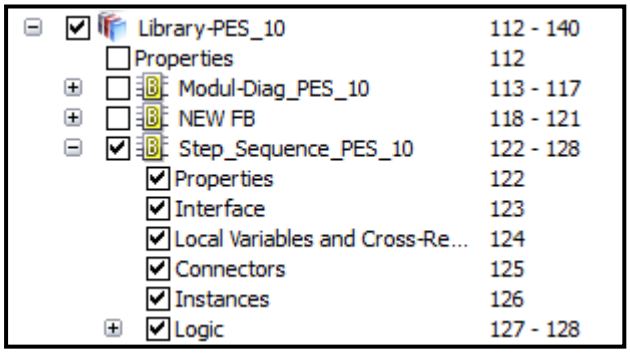
Consequently, two different cases must be differentiated:

1. The library is a part of the resource and the modified block is only used in this resource. The modified block can be printed out directly.
2. The library is a part of the configuration or of the project and the modified block is used in several resources. In this case, the effects of the changes must be considered for all of the affected resources.
   For the version comparison, only the version of the block of the selected resource is taken into account.

> **!** If an earlier version of the printed-out block is loaded in additional resources for which no code generation was carried out, the block can then no longer be shown in these resources in the online test.
> Consequently, HIMA recommends performing the entire modification procedure including loading for all affected resources.

7.4.1 Selection of Pages to be Printed

Function blocks with a limited scope of functions can be selected most easily directly in the library along with all of their properties.

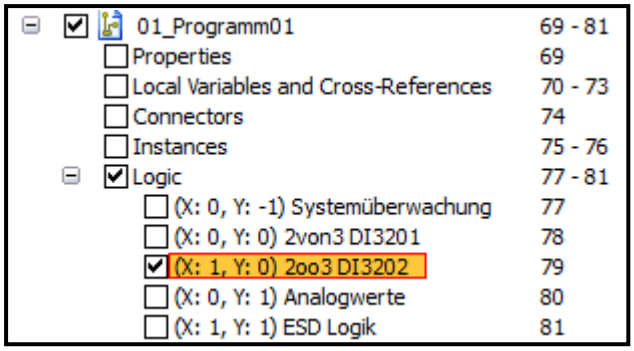


The position of the changes can be seen in the protocol of the version comparator which is also printed out.

Name	Type	Position DL	Position CG
OR	Instance	Page: 0/0, Pos.: 59/59	Page: 0/0, Pos.: 59/59
XOR	Instance	Page: 0/0, Pos.: 76/73	Page: 0/0, Pos.: 76/73

For function blocks with a larger scope of functions, you can also select individual areas, e.g., certain logic pages only. To do so, the information from the version comparator must be analyzed more precisely.

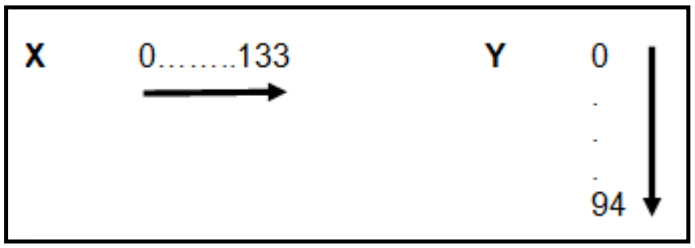
In the result, the extent of the printout is reduced because only the truly relevant pages are printed.



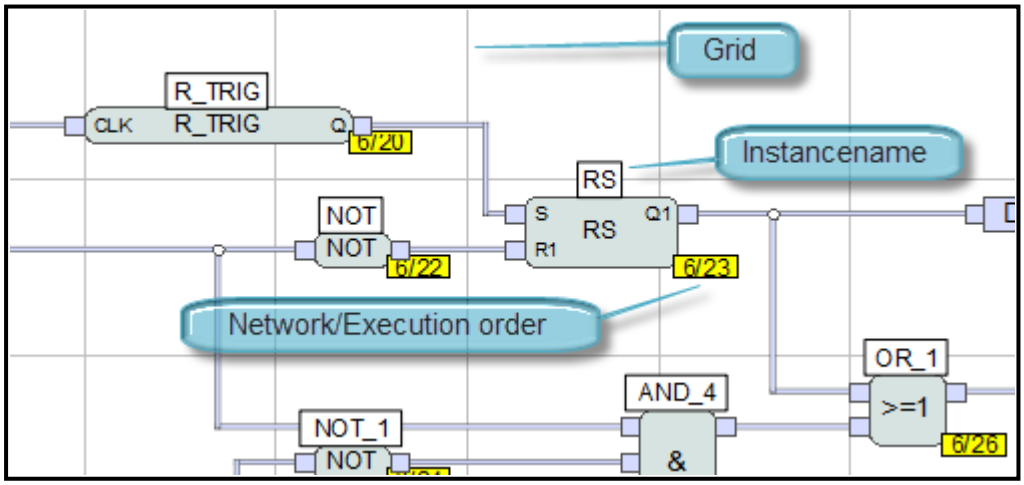
To make it easier to find the displayed position in the printout, activate all of the options in the *Documentation Generation Parameters* dialog box as described in *Starting the Documentation*.

The position displayed is the upper left corner of an object.

- The X-coordinates are counted sheet-wise from left to right.
- The Y-coordinates are counted sheet-wise from top to bottom.



The printed grid has a spacing of 10 units.



7.4.2 Documentation of Hardware Changes

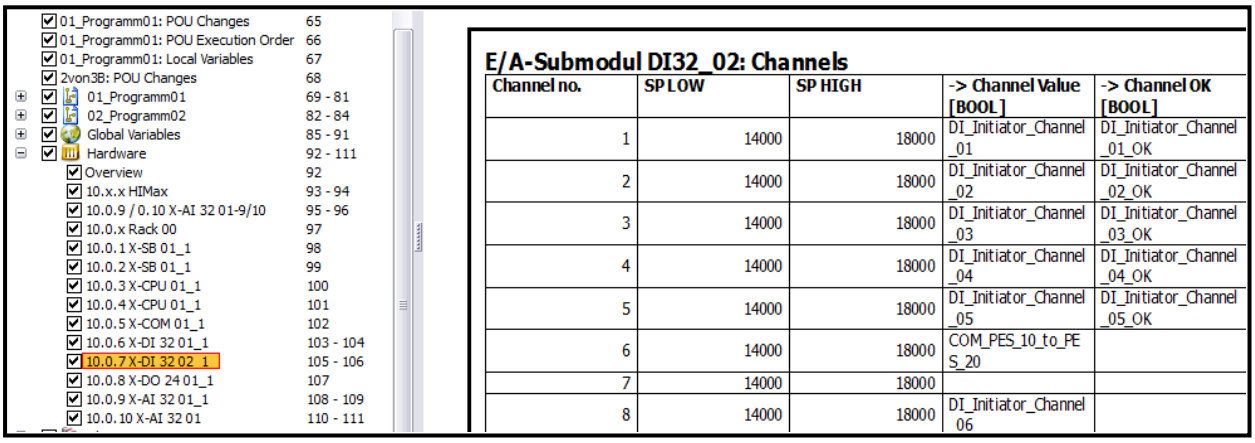
Changes to hardware settings, e.g., module parameters, are shown in the results of the version comparison (value: old/new – in the example, DL/CG).

Consequently, no additional documentation is usually necessary.

Context	Setting	Version DL	Version CG
10.0.7:1	SC Limit	65500	62000
10.0.7:7	Channel Used	Yes	No
10.0.7:8	Channel Used	No	Yes

If this information is not sufficient, all settings can also be printed out just as they appear in the Hardware Editor.

In the example, the settings for the module *X-DI 32 02* with the *SRS 10.0.7* have been selected for printing:



E/A-Submodul DI32_02: Channels

Channel no.	SP LOW	SP HIGH	-> Channel Value [BOOL]	-> Channel OK [BOOL]
1	14000	18000	DI_Initiator_Channel_01	DI_Initiator_Channel_01_OK
2	14000	18000	DI_Initiator_Channel_02	DI_Initiator_Channel_02_OK
3	14000	18000	DI_Initiator_Channel_03	DI_Initiator_Channel_03_OK
4	14000	18000	DI_Initiator_Channel_04	DI_Initiator_Channel_04_OK
5	14000	18000	DI_Initiator_Channel_05	DI_Initiator_Channel_05_OK
6	14000	18000	COM_PES_10_to_PES_20	
7	14000	18000		
8	14000	18000	DI_Initiator_Channel_06	

7.4.3 Documentation of Changed Variable Assignments

Changes to variable assignments are shown in the results of the version comparison. Consequently, no additional documentation is usually necessary.

Global Variable	Variable	Source	Destination	Type of change
DI_Channel_02	DI-Kanal02 -> Channel Value [BOOL]		X-DI 32 01_1.(10.0.6)	Variable 'DI-Kanal02 -> Channel Value [BOOL]' is no l
DI_Channel_02	DI-Kanal07 -> Channel Value [BOOL]		X-DI 32 02_1.(10.0.7)	Variable 'DI-Kanal07 -> Channel Value [BOOL]' is nov
DI_Channel_02	DI_Channel_02	X-DI 32 02_1.(10.0.7)	AP:01_Programm01	Source changed to: 'X-DI 32 02_1.(10.0.7)'

HIMA does not recommend printing out the entire list of global variables because the list can be very extensive.

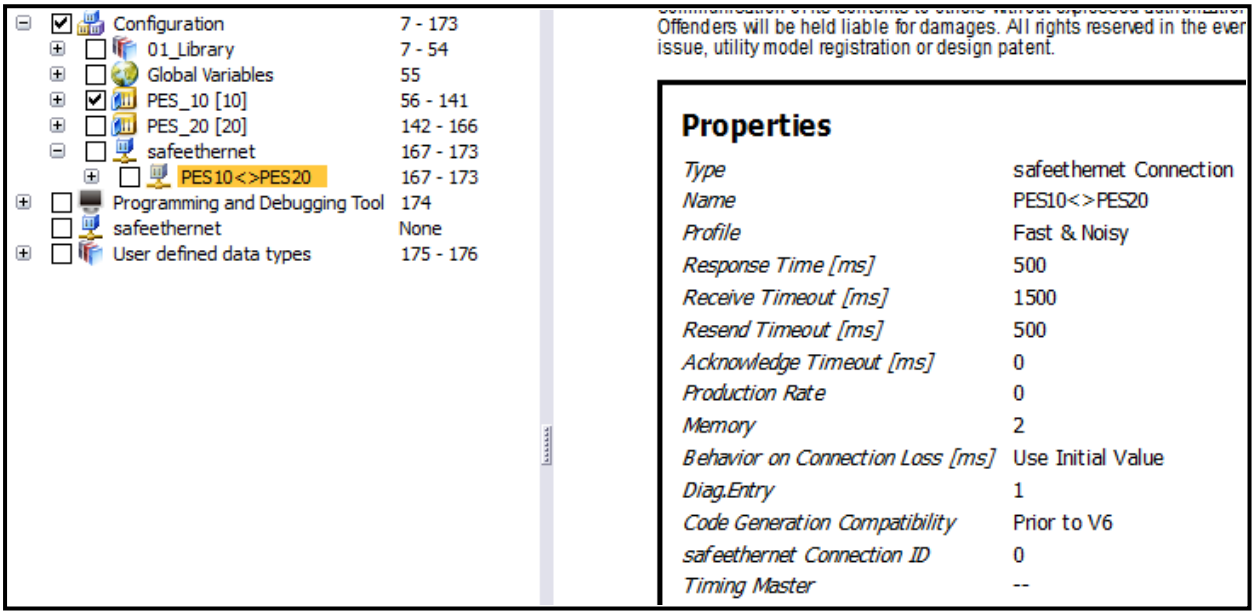
Name	Data type	Initial Value	Description	Additional Comment
Structure Element			Data type	Initial Value
Use	Structure Info		Info	
DI_Initiator_Channel_02	BOOL			
1x Reading	External POU		01_Programm01	
Writing	HW [10.0.7 - 2]		-> Channel Value [BOOL]	
1x Reading	External POU		Step_Sequence_PES_10	
DI_Initiator_Channel_02_OK	BOOL			
1x Not Connected	External POU		01_Programm01	
Writing	HW [10.0.7 - 2]		-> Channel OK [BOOL]	
DI_Initiator_Channel_03	BOOL			
1x Reading	External POU		01_Programm01	
Writing	HW [10.0.7 - 3]		-> Channel Value [BOOL]	

7.4.4 Documentation of Changed safeethernet Settings

Changes to safe**ethernet** settings are shown in the results of the version comparison as of SILworX V5 (value: old/new – in the example, DL/CG). Consequently, no additional documentation is usually necessary.

Context	Setting	Version IM	Version CG
20.0.0 PES_20	Receive Timeout [ms]	1300	1500

If this information is not sufficient, all settings can also be printed out just as they appear in the safe**ethernet** Editor.



HI 801 286 E

HIMA Paul Hildebrandt GmbH
Albert-Bassermann-Str. 28 | 68782 Brühl, Germany
Phone +49 06202 709-0 | Fax +49 06202 709-107
info@hima.com | www.hima.com

SAFETY
NONSTOP