

SIMATIC NET

Industrial Remote Communication - Telecontrol ST7cc Control Center

Operating Instructions

Preface

Introduction and installation

1

Integrating ST7cc in a
SINAUT network

2

Creating an ST7cc project

3

Configuring data with ST7cc
Config

4

ST7cc server

5

Diagnostics and trace
options

6

PM-AQUA link

7

ACRON link

8

Technological typicals

9

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Preface

Purpose of the manual

This manual supports you when using SINAUT ST7cc to connect WinCC to SINAUT ST7 or SINAUT ST1.

Aims

This manual provides you with the following information:

- Mapping a SINAUT ST7 configuration in the WinCC tag management
- Configuration of communications subscribers and communications objects
- Definition of WinCC preprocessing for messages and archiving
- Structuring a system using typicals
- Reusable units in the operating philosophy of WinCC

We assume that you are thoroughly familiar with your programs.

Structure of the documentation

The SINAUT ST7cc documentation includes the following:

- SINAUT ST7cc manual
- Readme files on the data medium of the software product

Validity of this manual / software version

This manual is valid for the following software version:

SINAUT ST7cc V3.1 + SP2

New in this release

- Support of a new operating system
- Support of newer versions of the SIMATIC NET Software

You will find the newest innovations in the section New functions of the current version of ST7cc (Page 16).

Replaced edition

Edition 03/2015

Cross references

In this manual there are often cross references to other sections.

To be able to return to the initial page after jumping to a cross reference, some PDF readers support the command <Alt>+<Left arrow>.

Further information on the Internet

You will find further information on ST7cc on the Internet at the following address:

Link: (<https://support.industry.siemens.com/cs/ww/en/ps/15927/man>)

License conditions

Note

Open source software

The product contains open source software. Read the license conditions for open source software carefully before using the product.

You will find license conditions in the following document on the supplied data medium:

- OSS_SINAUT-ST7cc_86.pdf

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens' guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit

Link: (<http://www.siemens.com/industrialsecurity>)

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under

Link: (<http://www.siemens.com/industrialsecurity>).

SIMATIC NET glossary

Explanations of many of the specialist terms used in this documentation can be found in the SIMATIC NET glossary.

You will find the SIMATIC NET glossary here:

- SIMATIC NET Manual Collection or product DVD

The DVD ships with certain SIMATIC NET products.

- On the Internet under the following address:

Link: (<https://support.industry.siemens.com/cs/ww/en/view/50305045>)

Training, Service & Support

You will find information on training, service and support in the multilanguage document "DC_support_99.pdf" on the Internet pages of Siemens Industry Online Support:

Link: (<https://support.industry.siemens.com/cs/ww/en/view/38652101>)

Table of contents

	Preface	3
1	Introduction and installation.....	13
1.1	Overview	13
1.2	Functions and properties	15
1.2.1	Telecontrol master with user-friendly diagnostics options	15
1.2.2	Preprocessing of process data	15
1.2.3	Simple, totally integrated project engineering.....	16
1.2.4	New functions of the current version of ST7cc	16
1.3	ST7cc installation options	16
1.3.1	Overview	16
1.3.2	Hardware requirements	17
1.3.3	Operating systems for the ST7cc PC	17
1.3.4	Required software.....	18
1.4	Licenses and license keys	19
1.5	Ordering data	21
1.6	Installing the ST7cc software package	22
1.7	PC settings for redundant ST7cc.....	27
1.7.1	Firewall settings with Windows Server 2008 (32-bit).....	28
1.7.2	Firewall settings with Windows 7 and Windows Server 2012	30
1.7.3	Firewall settings with Windows Server 2008 R2 (64-bit)	37
2	Integrating ST7cc in a SINAUT network	41
2.1	Task	41
2.2	Installing the hardware.....	42
2.3	Installation of the SIMATIC NET PC software products	42
2.4	Linking the SINAUT PC in the STEP 7 project	43
2.4.1	Integrating ST7cc in NetPro.....	44
2.4.2	Configuring an S7 connection between local TIMs and ST7cc	55
2.4.3	Integrating a redundant SINAUT PC	60
2.4.4	Time service on the MPI and Ethernet bus.....	66
2.4.5	Saving the configuration data in NetPro	72
2.5	Configuring the SINAUT PC	73
2.5.1	Initial configuration	74
2.5.2	Setting access points for the SINAUT PC	81
2.6	SINAUT ST7 configuration tool.....	84
2.6.1	Adapting SINAUT subscriber numbers.....	84
2.6.2	Configuring redundant ST7cc	87
2.6.3	Configuring SINAUT connections	91
2.6.4	Generating and compiling SINAUT data.....	98
2.6.5	Downloading SINAUT data to TIMs and CPUs	104

3	Creating an ST7cc project.....	107
3.1	Creating and opening an ST7cc project	107
3.1.1	Starting ST7cc Config	107
3.1.2	Creating a new ST7cc project.....	109
3.1.3	Opening an existing ST7cc project (ST7cc version 2).....	115
3.2	ST7cc Administration	116
3.2.1	Copy faceplates to a WinCC project.....	116
3.3	Project settings.....	117
3.3.1	Project settings: Server	118
3.3.2	Project settings: File paths	129
3.3.3	Project settings: Communication	132
3.3.4	Project settings: WinCC	136
3.3.5	Project settings: Archive	140
3.3.6	Project settings: Config	142
3.3.7	Project settings: Message protocol	146
3.4	Global settings	147
3.4.1	Global settings: Computer	149
3.4.2	Global settings: Project	151
3.4.3	Global settings: Language	153
4	Configuring data with ST7cc Config	155
4.1	What is ST7cc Config?	155
4.2	What does configuring mean?	156
4.3	Background knowledge on configuring.....	157
4.3.1	SINAUT subscriber	158
4.3.2	SINAUT object	158
4.3.3	SINAUT object types.....	159
4.3.4	ST7cc variable	163
4.3.5	Variable name	164
4.3.6	Type and subtype of a variable.....	165
4.3.7	Processing options for ST7cc variables.....	168
4.3.8	Object templates and typicals	168
4.3.9	Principle of decoding using typicals	171
4.3.10	Group display	174
4.4	Configuring.....	178
4.4.1	Starting ST7cc Config	180
4.4.2	ST7cc object tree	181
4.4.3	Object templates	182
4.4.4	System typicals	184
4.4.5	Creating a user typical	199
4.4.6	Setting up a subscriber	206
4.4.7	Creating a decoding	210
4.4.8	Creating a decoding with typicals	217
4.4.9	Copying and deleting decodings.....	222
4.4.10	Copying and deleting subscribers.....	226
4.4.11	Update scenarios for system typicals	228
4.5	Configuring processing functions.....	231
4.5.1	Working with a processing function	233

4.5.2	Message processing	235
4.5.3	Static additional texts	243
4.5.4	Counted value processing	245
4.5.5	Measured value processing	250
4.5.6	Archive	256
4.6	Variable list	257
4.7	SINAUT TD7 block structure in ST7cc Config	259
4.8	Generating for WinCC	263
4.8.1	Generating	264
4.8.2	The tag management	264
4.8.3	Message management	266
4.8.4	Archive tags	268
4.8.5	Generating subscriber picture typicals	269
4.8.6	Generating technological picture objects	270
4.8.7	Inserting picture typicals and faceplates in process pictures	270
5	ST7cc server	271
5.1	ST7cc server	271
5.1.1	Components and functions	271
5.2	Process image of the ST7cc server	273
5.3	ST7cc redundancy package:	274
5.3.1	General requests (GR) when starting up the redundant system	276
5.3.2	Description of the system statuses (redundant system)	277
5.3.3	ST7cc functions to ensure data consistency	283
5.4	Quality code of the WinCC tags supplied with values by ST7cc	285
5.5	Adopting the configuration data	288
5.5.1	Adopting the configuration data on a single system	288
5.5.2	Adopting the configuration data on a redundant system	289
5.6	Startup behavior and start order	289
5.7	Exiting ST7cc server and WinCC	290
5.8	Restarting WinCC with the ST7cc server active	290
5.9	ST7cc server status	291
5.10	Standard general request and accelerated general request	293
6	Diagnostics and trace options	297
6.1	Diagnostics: Log server messages	297
6.1.1	Messages relating to the process image	297
6.1.2	Error messages on communication	299
6.1.3	Status messages on communication	300
6.1.4	Messages on time-of-day synchronization	301
6.1.5	Diagnostic messages on parameter assignment errors	302
6.1.6	Messages on WinCC Tag Logging / Alarm Logging	303
6.1.7	Messages on the PM-AQUA interface	303
6.2	Diagnostics: Message protocol of the ST7cc server	304
6.3	Diagnostics: Subscriber typicals and faceplates	306

6.3.1	Picture typicals and faceplates for a station	306
6.3.2	Picture typicals and faceplates for a local TIM	311
6.3.3	Picture typical and faceplate for a server.....	314
6.4	Diagnostics: Trace	318
6.4.1	Trace output dialog	319
6.5	Diagnostics: System typical	320
7	PM-AQUA link.....	321
7.1	PM-AQUA process links	321
7.2	PM-Aqua configuration with ST7cc Config	322
7.3	Optimization of the handshake procedure	324
8	ACRON link	327
8.1	Importing historical files (CSV, DBASE)	327
8.2	ACRON project settings with ST7cc Config.....	330
8.3	ACRON configuration with ST7cc Config	331
9	Technological typicals	333
9.1	Aims	333
9.2	Overview	335
9.2.1	ST7cc typical and data structure of an information unit.....	337
9.2.2	Definition of the information units.....	338
9.2.3	Assigning information units to SINAUT objects	339
9.2.4	Typicals in ST7cc.....	341
9.2.5	Picture typicals in WinCC.....	341
9.2.6	Faceplates in WinCC	344
9.3	Templates for the pump technological object	346
9.3.1	ST7cc typicals	347
9.3.2	Corresponding picture typical	350
9.3.3	Corresponding faceplate	352
9.4	Templates for the 1Motor technological object	353
9.4.1	ST7cc typicals	353
9.4.2	Corresponding picture typical	357
9.4.3	Corresponding faceplate	359
9.5	Templates for the generator technological object.....	360
9.5.1	ST7cc typicals	360
9.5.2	Corresponding picture typical	364
9.5.3	Corresponding faceplate	366
9.6	Templates for the valve technological object.....	367
9.6.1	ST7cc typicals	367
9.6.2	Corresponding picture typical	371
9.6.3	Corresponding faceplate	373
9.7	Templates for the compressor technological object	374
9.7.1	ST7cc typicals	374
9.7.2	Corresponding picture typical	378
9.7.3	Corresponding faceplate	380

9.8	Templates for the Motor2 technological object	381
9.8.1	ST7cc typicals	382
9.8.2	Corresponding picture typical	386
9.8.3	Corresponding faceplate	388
9.9	Templates for the slider valve technological object	389
9.9.1	ST7cc typicals	389
9.9.2	Corresponding picture typical	393
9.9.3	Corresponding faceplate	396
Glossary	397
Index	403

Introduction and installation

1.1 Overview

Introduction

SINAUT ST7cc is the ideal control center system based on SIMATIC WinCC for both SINAUT ST7 and SINAUT ST1. It is specially designed for event-driven and time-stamped data transmission in the SINAUT system.

Along with the WinCC redundancy package, a fault-tolerant ST7cc control center can be implemented.

SINAUT ST7cc also takes on the function of a telecontrol center. A separate SIMATIC S7 CPU for this functionality is therefore not required.

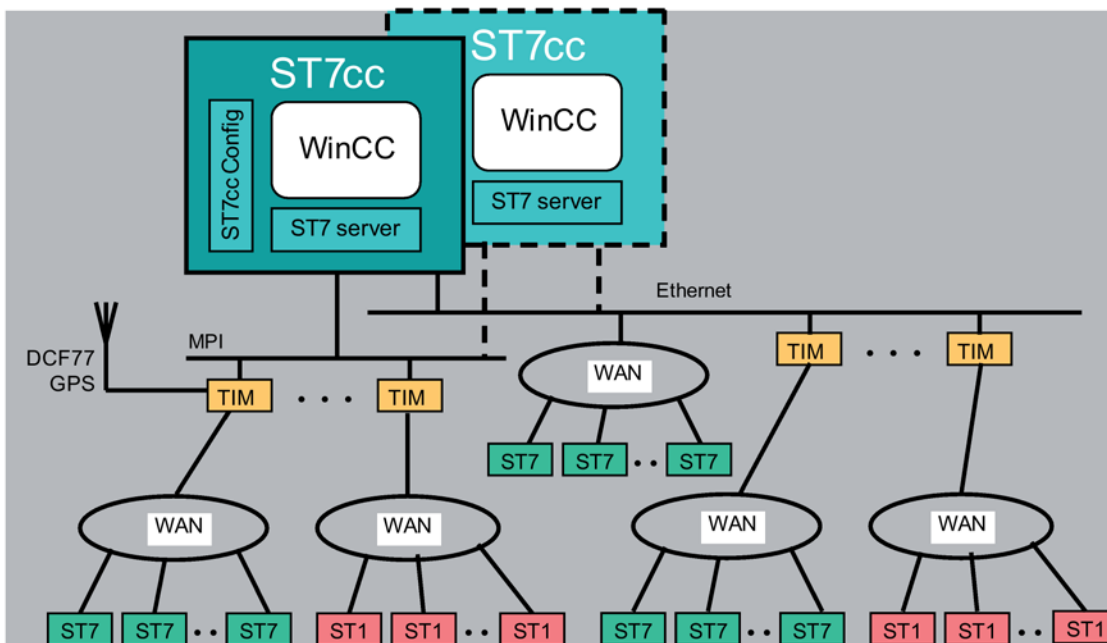


Figure 1-1 SINAUT ST7cc (single or redundant) with connected ST7 and ST1 stations

One or more SINAUT ST7 TIM communications processors are connected directly over the MPI S7 standard bus or Ethernet. Both ST7 and ST1 stations can be connected to these TIMs that are installed locally in the ST7cc control center (refer to the two WAN networks on the left in the figure and the two WAN networks on the right). With Ethernet-based WANs, no TIM is necessary in the ST7cc control center. The attachment of the stations (here, only ST7 stations are possible) is then directly to the Ethernet interface of the ST7cc computer (refer to the WAN in the middle in the figure).

1.1 Overview

When supplying the current time of day, the following situations must be distinguished:

- For TIMs (TIM 4x), connected to the ST7cc PC over MPI, time synchronization is possible only with a TIM equipped with a DCF77 receiver. This then serves as the central time synchronization source for the ST7cc PC and all stations.
- For TIMs (TIM 3V-IE, TIM 4R-IE) connected to the ST7cc PC over Ethernet, the time is synchronized over ST7cc.

A GPS receiver is recommended outside the reception range of the DCF77 time transmitter; this can determine the local time from the satellite-based GPS system (Global Positioning System).

The GPS kit 6NH9831-8AA that was available from Siemens has been discontinued since 11/2011.

Benefits

SINAUT ST7cc has the following benefits for the user:

- Interfacing SINAUT stations to SIMATIC WinCC over classic serial WAN or over Ethernet-based WAN
- Entry of messages, analog and counted values in the WinCC archive and the use of the event times supplied by the SINAUT stations.
- Protection of investment in existing SINAUT ST1 systems since ST1 stations can be connected
- Time and costs saved by simple configuration without detailed knowledge of SINAUT

Area of application

SINAUT ST7cc is specially designed for event-driven and time-stamped data transmission in the SINAUT system. It avoids the possible loss of data that can occur with cyclic polling in WinCC. It also ensures the use of the correct event time supplied by the SINAUT stations for all WinCC messages and archive entries. The process image integrated in ST7cc contains all process data as well as the status of all SINAUT subscribers in the network and makes this data available directly to WinCC for fast transfer to the process image.

The ST7cc Config configuration tool provides the user with fully integrated engineering based on the data messages that were configured in the SINAUT ST7 or ST1 stations. Configuration of WinCC including tag management is therefore generated automatically and updated consistently whenever changes occur.

For archives, logs and reports that meet the requirements of ATV H260 or Hirthammer, the additional use of the WinCC add-on ACRON is advisable. ST7cc provides a configurable data interface to these add-ons.

The WinCC add-on "Alarm Control Center" can be used to alert standby personnel by SMS, fax or e-mail.

Along with the WinCC redundancy package, a fault-tolerant ST7cc control center can be implemented.

1.2 Functions and properties

1.2.1 Telecontrol master with user-friendly diagnostics options

Functions

- Direct connection of SINAUT ST7 TIMs to ST7cc over MPI and Ethernet. A separate CPU as telecontrol master is not required.
- Availability of the most important status information of each SINAUT ST7 or ST1 subscriber with visualization in WinCC using provided picture typicals and faceplates.
- Option of controlling SINAUT subscribers with these faceplates.
- Identification of process values from subscriber stations with a disrupted connection to ST7cc.
- General request to affected stations following data transfer problems to allow the process image to be updated in ST7cc.
- Diagnostics with selective message protocol for individual or all SINAUT subscribers. Message visualization and evaluation as with TIM message monitor.
- Time synchronization by ST7cc for the TIMs connected to the ST7cc PC over Ethernet.

1.2.2 Preprocessing of process data

Preprocessing can be configured for binary, analog, and counted values. This takes into account the event time and adds the time stamp of the event time to related messages and archive entries.

Binary values

- Entry of current binary values in the assigned WinCC tags
- Entry of related messages into the WinCC message system taking into account the time stamp supplied by SINAUT ST7 or ST1.

Analog values (instantaneous and mean values)

- Floating-point numbers, integer values
- Linear raw value conversion (raw value --> physical value).
- Entry of analog values (with or without linear raw value conversion) in the assigned WinCC tags.
- Entry of analog values (with or without linear raw value conversion) in the WinCC archive taking into account the time stamp supplied by SINAUT ST7 or ST1.

Counted values

- Overflow handling with absolute counters.
- Counted value conversion using factors.
- Calculation of correctly timed interval quantities.
- Entry of currently accumulating interval quantities in the assigned WinCC tags.
- Entry of completed interval quantities in the WinCC archive taking into account the time stamps supplied by SINAUT ST7 or ST1.

Setpoints

- Floating-point numbers, integer values
- Linear raw value conversion (raw value > physical value) when necessary.

1.2.3 Simple, totally integrated project engineering

Configuration of the entire system is extremely user friendly with ST7cc Config. Extra WinCC configuration for tag management, archives, and the message system is restricted to a few preparations, such as creating the message classes and types and the archives in WinCC.

1.2.4 New functions of the current version of ST7cc

The version of ST7cc described here has the following new or modified functions:

- Windows operating systems, see section Operating systems for the ST7cc PC (Page 17) and Required software (Page 18).

1.3 ST7cc installation options

1.3.1 Overview

Overview of the software packages

SINAUT ST7cc is installed on a Windows PC.

The ST7cc server must be installed on the PC on which the WinCC server is also installed.

ST7cc Config can be installed on a WinCC server or WinCC client.

A license must be available for the WinCC full package. This can also be a runtime package if no configuration is necessary on the end computer. For redundant ST7cc, you also require the WinCC redundancy package including license.

For ST7cc, a license is required only for the ST7cc server.

The following table shows which software packages are required for the ST7cc single or redundant system.

ST7cc single system		ST7cc redundant system	
Qty.	Software package	Qty.	Software package
1	Windows 7 / Windows Server 2008 SP2 (32 Bit) / Windows Server 2008 R2 SP1 (64 Bit) / Windows Server 2012 R2 (64 Bit)	2	Windows 7 / Windows Server 2008 SP2 (32 Bit) / Windows Server 2008 R2 SP1 (64 Bit) / Windows Server 2012 R2 (64 Bit)
1	WinCC full package *	1	WinCC full package *)
		1	WinCC Runtime package
		1	WinCC redundancy package (with two licenses)
1	SIMATIC NET PC software with license for CP software	1	SIMATIC NET PC software with license for CP software
		1	ST7cc redundancy package (with two licenses)
1	<ul style="list-style-type: none"> • ST7cc S (license for six stations) or • ST7cc M (license for 12 stations) or • ST7cc L (license for >12 stations) 	2	<ul style="list-style-type: none"> • ST7cc S (license for six stations) or • ST7cc M (license for 12 stations) or • ST7cc L (license for >12 stations)

* Can also be a runtime package if no configuration is necessary on the runtime computer.

1.3.2 Hardware requirements

The requirements for installation are the same as for WinCC. For ST7cc, the following extra hard disk space is necessary:

- ST7cc program: approx. 25 MB
- ST7cc project: 200 to 400 MB

1.3.3 Operating systems for the ST7cc PC

Windows operating systems

The following operating systems are supported:

- Windows 7 SP1 (32 / 64 Bit)
- Windows 10 Pro (64 Bit)
- Windows Server 2008 SP2 (32 Bit)
- Windows Server 2008 R2 SP1 (64 Bit)
- Windows Server 2012 R2 (64 Bit)

1.3.4 Required software

Certain software packages must be installed on the PC. You need to distinguish between the software required for operation on the ST7cc PC and the software required for configuring connections with SINAUT stations. This configuration software is usually installed on a separate STEP 7 programming device, but, if required, it can also be installed on the ST7cc PC (see below).

To make clear what must or can be installed where, the following sections differentiate between the software required for the ST7cc PC and the STEP 7 programming device.

On the ST7cc PC:

The following software is required on the PC for the ST7cc runtime system and this must be installed in the order shown.

- **SIMATIC WinCC**

For:

- Windows Server 2008 SP2 (32 Bit):
Version V7.2 or V7.3
- Windows 7 SP1 (32 / 64 Bit), Windows 10, Windows Server 2008 R2 SP1 (64 Bit),
Windows Server 2012 R2 (64 Bit):
Version V7.2 or V7.3 or V7.4

- **ST7cc**

- **SIMATIC NET PC Software**

For:

- Windows 7 SP1 (32 / 64 Bit), Windows Server 2008 R2 SP1 (64 Bit),
Windows Server 2012 R2 (64 Bit):
SIMATIC NET PC Software V12.0 SP2
- Windows Server 2008 SP2 (32 Bit):
SIMATIC NET PC Software V7.1 SP6
- Windows 10 (64 Bit):
SIMATIC NET PC Software V13 SP2 or higher

Note

SIMATIC NCM PC is no longer supplied as of SIMATIC NET PC Software V12.

Optional:

- WinCC add-on "SIMATIC Process Historian":
Version 2014 SP2
- Visualization with VMware ESXi V5.5:
See release notes of SIMATIC WinCC and SIMATIC NET PC software.

On the STEP 7 programming device (configuration PC / engineering station):

The following software is required on the PC for configuring the ST7cc system:

- SIMATIC STEP 7 V5.4 + SP4 or higher
STEP 7 includes the function of SIMATIC NCM PC.
- SINAUT engineering software V5.0 or higher

Note

Windows Server 2012

If Windows Server 2012 is installed on the ST7cc PC (runtime system), you need to use a separate configuration PC with an operating system compatible with STEP 7, refer to the readme of STEP 7.

As of STEP 7 V5.5, the operating systems MS Windows 7 Professional, Ultimate and Enterprise (standard installation) are supported. As of SP1, you can use the released operating systems both in the 32-bit and 64-bit versions.

Windows Server 2012

If Windows Server 2012 is installed on the ST7cc PC (runtime system), you require a separate PC with Windows 7 for the configuration.

As of STEP 7 V5.5, the operating systems MS Windows 7 Professional, Ultimate and Enterprise (standard installation) are supported. As of SP1, you can use the released operating systems both in the 32-bit and 64-bit versions.

1.4 Licenses and license keys

Memory medium for license keys

As of ST7cc version V3, the product is licensed using license keys that ship with the product on a USB flash drive and that need to be transferred to your computer.

It is only necessary for the ST7cc license key to be installed on the ST7cc PC when you want to run ST7cc runtime. ST7cc Config can be used on the PC without an installed license key.

Automation License Manager

To license the product, you require the Automation License Manager (ALM), as of version V5.1 Service Pack 1 Update 3.

You can install the program required to display, install and uninstall licenses from the DVD with the SIMATIC NET PC Software. The program is started with the menu command Start > Programs > Siemens Automation > Automation License Manager.

For more detailed information on working with it, refer to the online help of the program itself.

NOTICE

almreadme.rtf

Note the information in the "almreadme.rtf" file in the installation directory of the Automation License Manager. If you do not adhere to the instructions, you may lose your license key irretrievably.

Upgrade

With ST7cc versions with license keys, you can run of the upgrade function with new "Upgrade" license keys.

If you have updated an existing ST7cc installation and the license key for this ST7cc is already installed on the PC, you may continue to use this license key.

Types of license

- **First license:**
when you install ST7cc V3.1 the first time, you will need to purchase the relevant license SINAUT ST7cc S, M, L (standard license) or SINAUT ST7cc R (redundancy license). You will also need to purchase licenses if you want to change from V2 to V3.1.
For a redundancy license, you require two additional single licenses.

Note

No upgrading of licenses with activation code

The upgrading of ST7cc licenses with activation code with which no diskette or USB stick was supplied to version V3 is not possible. Only licenses capable of ALM can be upgraded.

- **Upgrade:**
To upgrade an existing license SINAUT ST7cc S/M/L/R from version V2.4...V2.7 to version V3.1 and from V3.0 to V3.1 you require the license SINAUT ST7cc Upgrade (standard license) or SINAUT ST7cc Upgrade Redundancy (redundancy license). The existing license is then used to generate a new license key with the aid of the Automation License Manager that allows the use of powerpacks.
- **Powerpack:**
To expand a license (increase the number of stations) of an existing SINAUT ST7cc S/M/L/R version V3.1 license, you require the powerpacks ST7cc S > M, S > L or M > L.

1.5 Ordering data

Ordering data	Article number
SINAUT ST7cc Software for connecting SINAUT stations to WinCC. Single license for one installation of the runtime software. Runtime software, configuration software, and electronic manual German/English on CD-ROM; license key on USB flash drive.	
Standard licenses	
SINAUT ST7cc V3.1 Small license for max. 6 SINAUT stations	6NH7997-7CA31-0AA1
SINAUT ST7cc M V3.1 Medium license for max. 12 SINAUT stations	6NH7997-7CA31-0AA2
SINAUT ST7cc L V3.1 Large license for more than 12 SINAUT stations	6NH7997-7CA31-0AA3
SINAUT ST7cc RED V3.1 Redundancy license for ST7cc (contains 2 licenses) Note: In addition, two single ST7cc S, M, or L licenses are required	6NH7997-8CA31-0AA0
Powerpacks	
SINAUT ST7cc SM Powerpack V3.1 License expansion from ST7cc S to ST7cc M (from 6 to 12 stations)	6NH7997-7AA31-0AD2
SINAUT ST7cc SL Powerpack V3.1 License expansion from ST7cc S to ST7cc L (from 6 to more than 12 stations)	6NH7997-7AA31-0AD3
SINAUT ST7cc ML Powerpack V3.1 License expansion from ST7cc M to ST7cc L (from 12 to more than 12 stations)	6NH7997-7AA31-0AE3
Upgrades	
SINAUT ST7cc V3.1 UPGR V3.0 Upgrade of a standard license S, M, L from V3.0 to V3.1 or upgrade of a powerpack SM, SL or ML from V3.0 to V3.1	6NH7997-7CA31-0GA1
SINAUT ST7cc V3.1 RED UPGR V3.0 Upgrade of the redundancy license R from V3.0 to V3.1	6NH7997-8CA31-0GA0
SINAUT ST7cc V3.1 S UPGR V2.4...V2.7 Upgrade of a standard license S from V2.4...V2.7 to V3.1	6NH7997-7CA31-2GA1
SINAUT ST7cc V3.1 M UPGR V2.4...V2.7 Upgrade of a standard license M or a powerpack SM from V2.4...V2.7 to V3.1	6NH7997-7CA31-2GA2
SINAUT ST7cc V3.1 L UPGR V2.4...V2.7 Upgrade of a standard license L or a powerpack SL or ML from V2.4...V2.7 to V3.1	6NH7997-7CA31-2GA3
SINAUT ST7cc V3.1 RED UPGR V2.4...V2.7 Upgrade of the redundancy license R from V2.4...V2.7 to V3.1	6NH7997-8CA31-2GA0

1.6 Installing the ST7cc software package

Note

Before beginning the installation of ST7cc, WinCC any previous versions of ST7cc Runtime and ST7cc Config should be closed, otherwise all the files cannot be copied.

In the Windows Explorer, select the CD-ROM drive in which you inserted the ST7cc CD and start the installation by double-clicking on Setup.exe.

Selecting the setup language

In the dialog that now opens, you can select the setup language. The following options are available:

- German (Germany)
- English (USA)

Once you have selected the setup language, the installation wizard is prepared (displayed on the screen).

Among other things, setup also checks whether WinCC and the SIMATIC NET PC software are already installed on the PC. If they are not, the installation aborts with an error message to this effect.

Otherwise, the installation continues with the *Welcome* dialog, and after clicking on the *Next* button, the *Software License Agreement* dialog opens. If you agree to the license conditions, click the *Yes* button.

Setup now checks whether or not there is an existing ST7cc installation on your PC. If there is already an installation, setup automatically starts the Update installation; otherwise it starts a new installation.

New installation

In a new installation, the *User Information* dialog opens. Enter your name and name of your company in this dialog.

After clicking *Next*, the installation continues with the *Choose Folder* dialog.

Update installation

If there is already an ST7cc installation on the PC, the Update dialog opens.

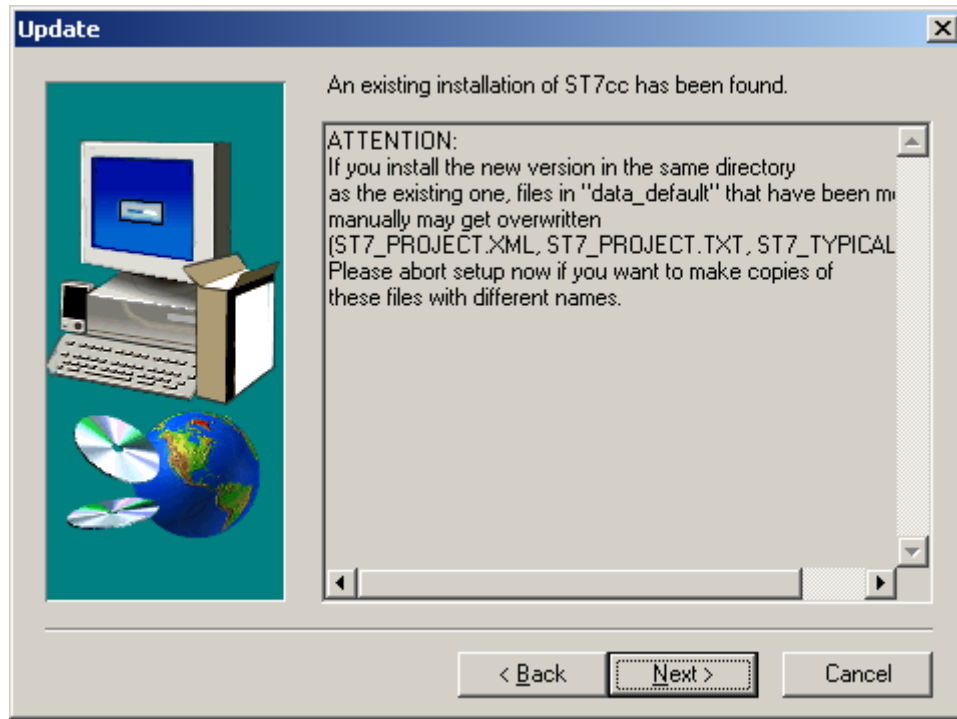


Figure 1-2 Dialog notifying an update installation

If you intend to install an update in the same target directory, in which you installed the existing ST7cc (recommended strategy), note the warning shown in the Update dialog. This warning applies only if the old version found is version 2.0 or higher.

The warning relates to the content of the `data_default` folder (see Figure). An update overwrites the contents of this folder. You may have changed files in this folder. For example, you may have entered your own defaults that you wish to use for all of your projects. In this case, back up these files first by copying them to another directory. Once the update is completed, you can customize the contents of the `data_default` folder according to your requirements using the backup files.

Selecting the target folder

The *Choose Folder* dialog now appears. Here, you select the drive and directory in which you want to install ST7cc on the hard disk.

If you agree to the proposed default folder, you can start the installation by clicking the *Next* button. Installation then continues at the following section Performing and completing the installation.

If you want to install in a different directory than the one proposed, click on the *Browse...* button. In the *Choose Folder* dialog, you can now select the target folder you wish to use.

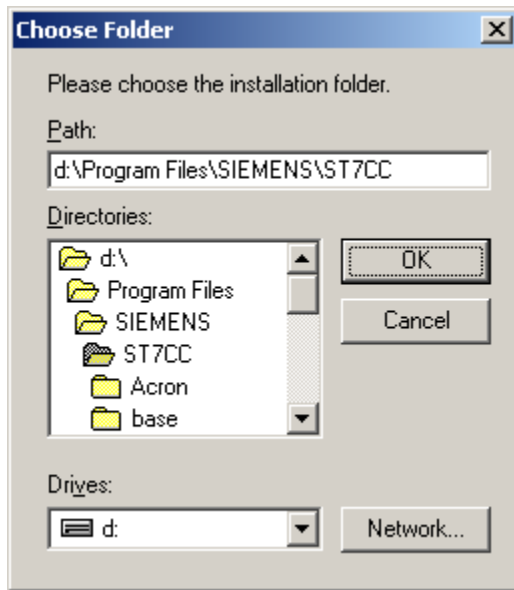


Figure 1-3 Changing the default folder

After clicking on the *OK* button, a message may appear if the folder you have specified does not yet exist. After you click on *Yes*, the missing folder is created automatically.

The *Choose Folder* dialog reappears with the new destination folder.

When you click the *Next* button, installation of the ST7cc software package starts.

Note

The folders "base" for computer settings and "log" for log entries are created, for example in Windows Server 2008 in "<drive> > Program Data > Siemens > ST7cc > ...". Remember that the folder must be shared using the folder options so that it is visible in the Explorer.

Performing and completing the installation

Once the installation directory has been selected, the installation begins. You can follow the progress of the installation on the screen.

Once the installation is completed, you are given the option of reading the readme file. Please read the readme file carefully. It contains important information that may not yet be included in the most recent ST7cc documentation.

You must restart your computer to activate the new ST7cc installation. The *Setup Complete* dialog therefore appears at the end of the installation. This gives you the option of restarting immediately. Accept the default setting *Yes, I want to restart my computer now* and click on the *Finish* button. After the restart, the new version of ST7cc is activated on the PC.

Notes on the ST7cc menu

You can now access the ST7cc menu with *Start / Programs / Siemens Automation / SIMATIC ... / ST7cc*.

The menu contains the following entries:

- ST7cc Config
The tool for configuring your ST7cc project.

- ST7cc Runtime
Must be started if your ST7cc PC needs to go online
- Service-Tools
ST7cc Trace: a diagnostics tool to log the message traffic between ST7cc and the local TIM(s).
CM Diagnostics: a diagnostics tool for monitoring the individual ST7cc applications.
CM Diagnostics: a diagnostics tool for monitoring the process connections.

Content of the ST7cc directory

The figure shows which folders have been created under the main directory ST7cc.

Some of the folders in the ST7cc directory are described below.

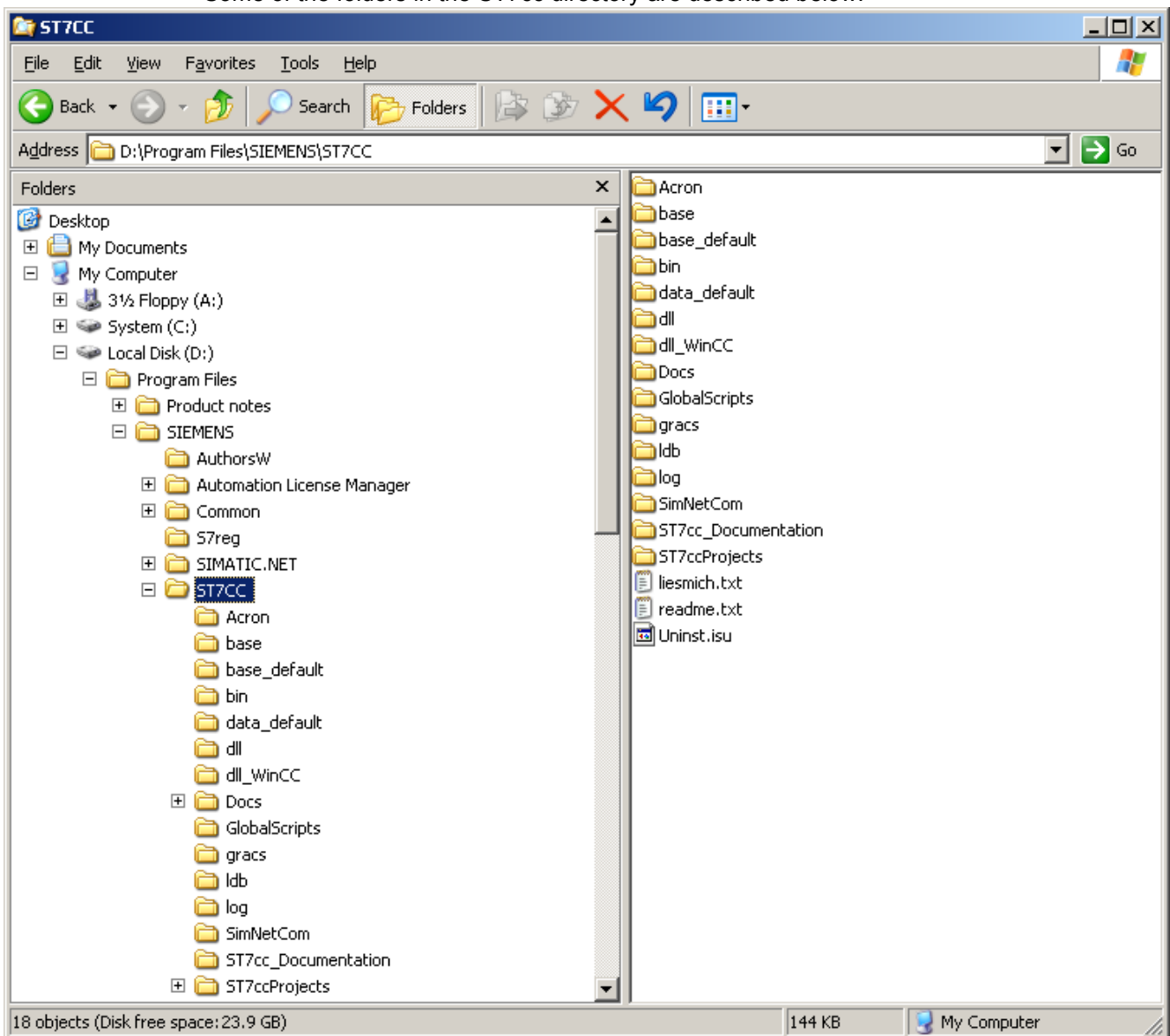


Figure 1-4 Overview of the folders installed under ST7cc

Notes on the *data_default* folder

The figure shows the content of the *data_default* folder. The default TXT and XML files are taken from this folder when you open a new ST7cc project on the PC. You can customize the content in these default files in the *data_default* folder, for example, by entering your own defaults that you wish to use for all of your projects.

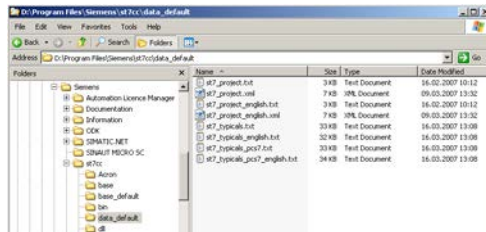


Figure 1-5 Content of the *data_default* folder

The warning message displayed prior to update installation relates to the content of this *data_default* folder. (see figure)

The *data_default* folder also includes new typical files for users wanting to use the group display (PCS 7 functionality). In this case, the *st7_typical_pcs7.txt* or *st7_typical_pcs7_english.txt* file must be used. In existing projects or in projects in which the group display is not required, the modified typical file *st7_typical.txt* or *st7_typical_english.txt* should be used. As of ST7cc version V2.7, these typical files also include the TIM typical modified for the TIM 4R-IE.

Installed ST7cc documentation

Installation of ST7cc includes installation of the latest documentation in German and English on the PC (*..ST7cc\Docs\deutsch* or *english*).

Standard faceplates for ST7cc

The following screenshot () shows the content of the *gracs* folder. This folder contains all standard picture typicals and faceplates for ST7cc and picture typicals and faceplates for technological objects. You can include these in your WinCC project. For more detailed information on this topic, refer to the section Copy faceplates to a WinCC project (Page 116).

The programming of the picture typicals and faceplates supplied as of version V2.7 is tailored to the use of the group display. They can, however, always be used regardless of whether a typical file is installed to support the group display functionality.

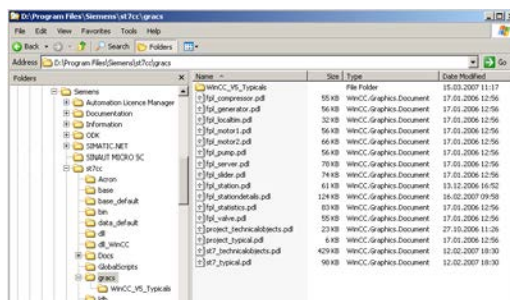


Figure 1-6 Picture typicals and faceplates in the *gracs* folder

Global scripts

The GlobalScripts folder (*..ST7cc\GlobalScripts*) contains various WinCC global scripts that you can use when necessary. The global scripts are simply examples and are not described here.

1.7 PC settings for redundant ST7cc

With a redundant ST7cc system, you need to open port 10100 in the firewall of both PCs for the ST7cc application. Follow the steps outlined below depending on the operating system.

Note

If you use a non-Windows firewall, refer to the information on enabling a port manually in the product-specific documentation.

If you want to synchronize the time-of-day of the redundant ST7cc by the TIM module of a station in Windows 7, you need to enable the required user rights for the ST7cc application (see section Time service on the MPI and Ethernet bus (Page 66)).

1.7.1 Firewall settings with Windows Server 2008 (32-bit)

1. Open the menu "Start > Settings > Control Panel > Windows Firewall".
The "Windows Firewall" window opens.
2. Open the "Exceptions" tab.

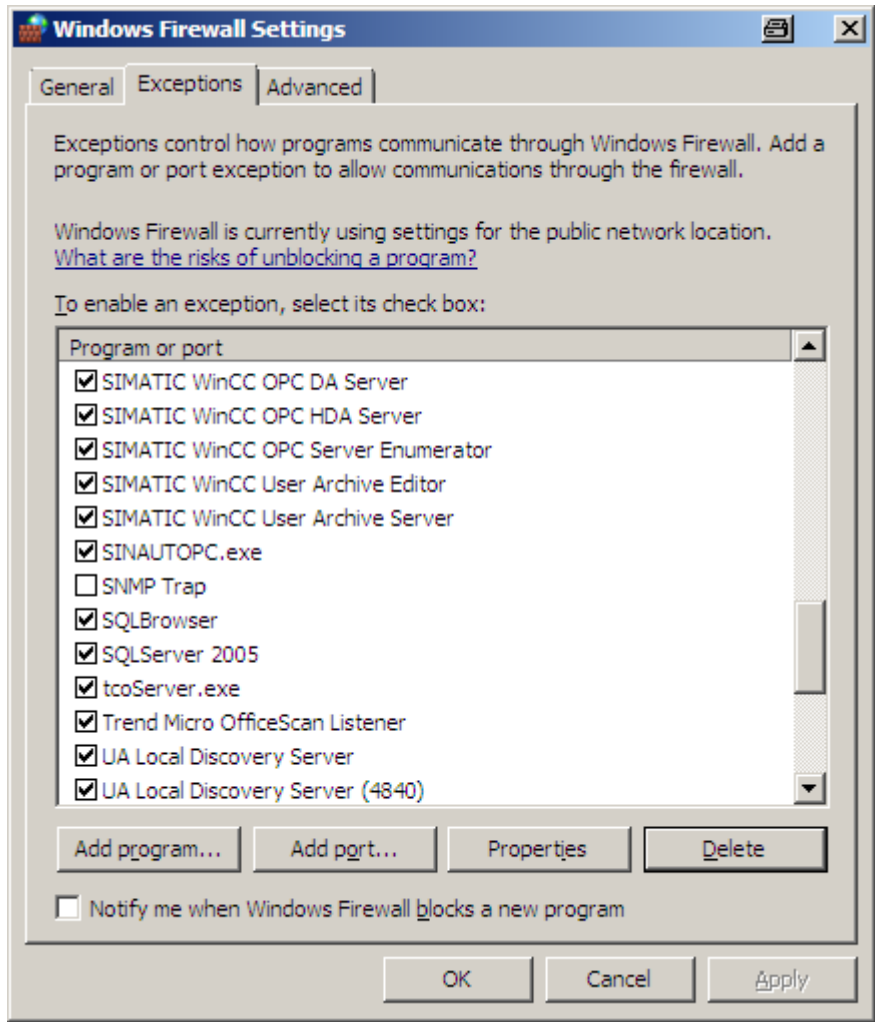


Figure 1-7 Windows Firewall window, Exceptions tab

3. Click the "Port" button.

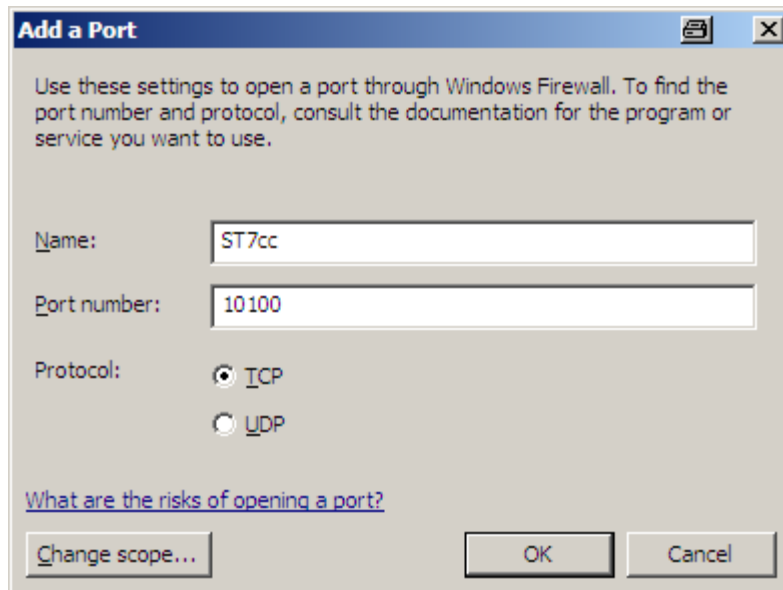


Figure 1-8 "Add Port" dialog box

4. Enter the name for the application (in the example "ST7cc") and the port number 10100.
5. Confirm the dialog box and the firewall window with OK.

1.7.2 Firewall settings with Windows 7 and Windows Server 2012

Firewall settings

1. Open the menu "Start > Control Panel > All Control Panel Items > Windows Firewall".

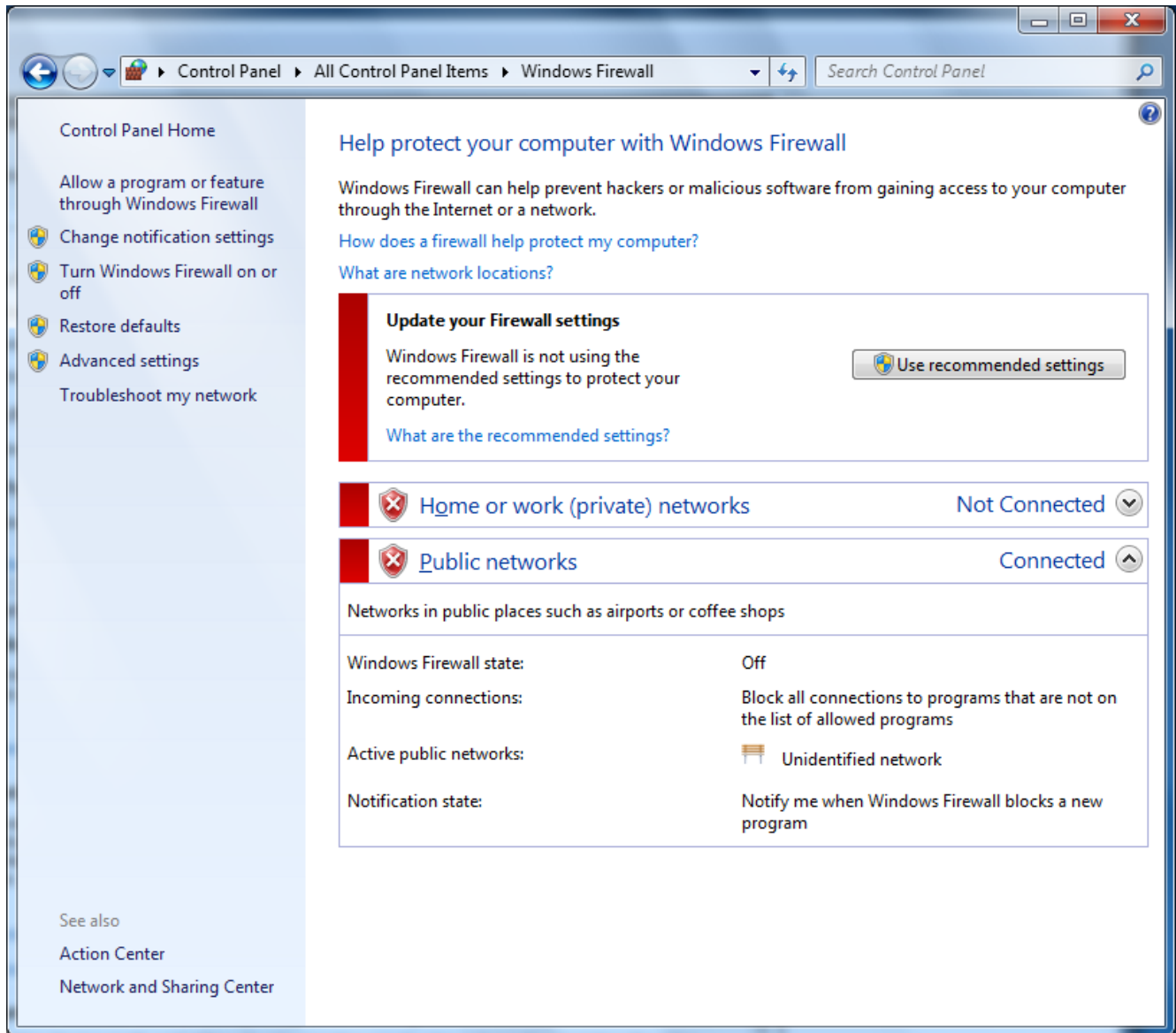


Figure 1-9 Window of the Windows Firewall

2. Click on "Advanced settings" in the navigation panel on the left.

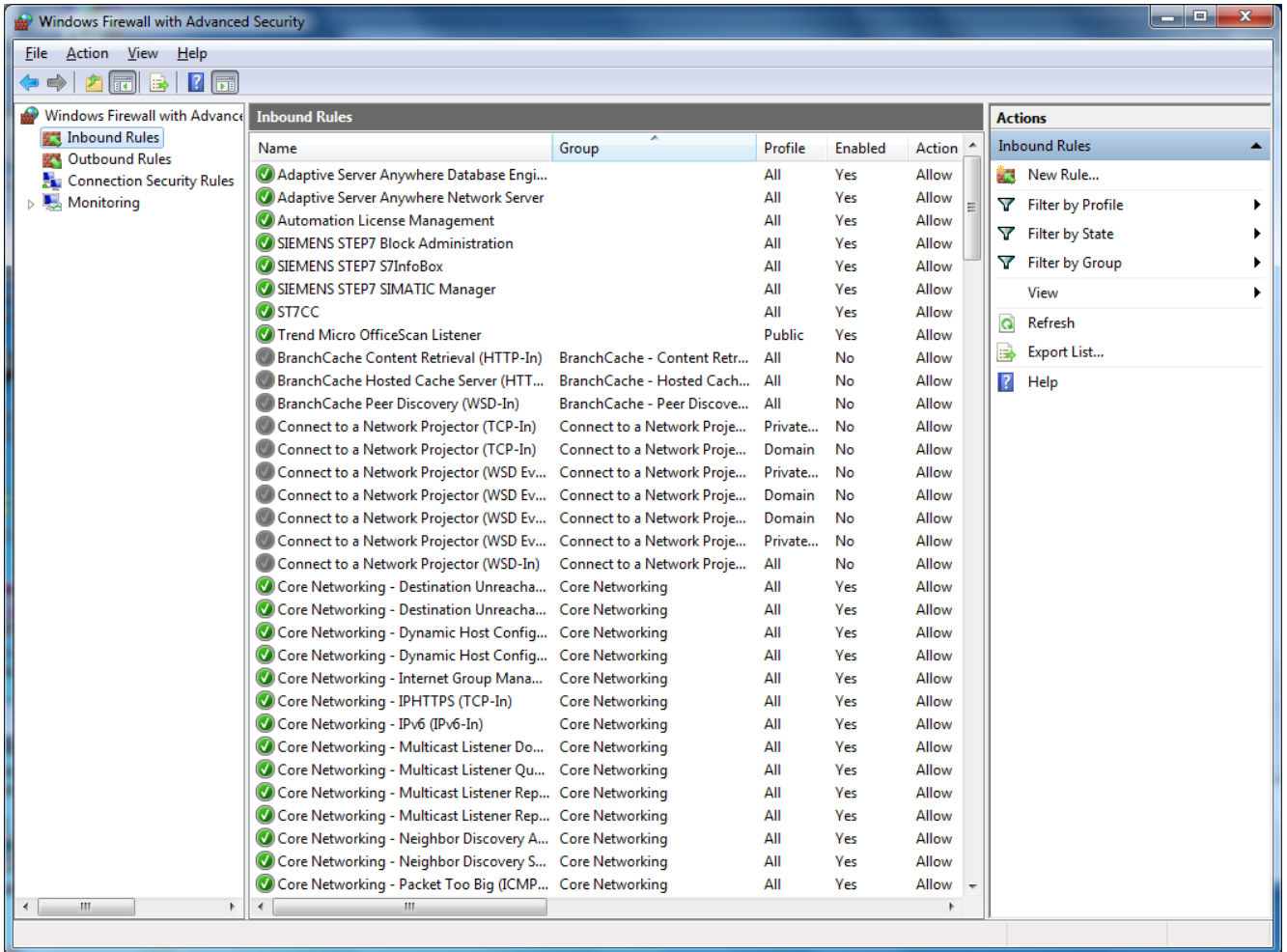


Figure 1-10 Windows Firewall window, Advanced settings

3. Select "New rule" in the shortcut menu (right mouse button) of "Inbound rules".

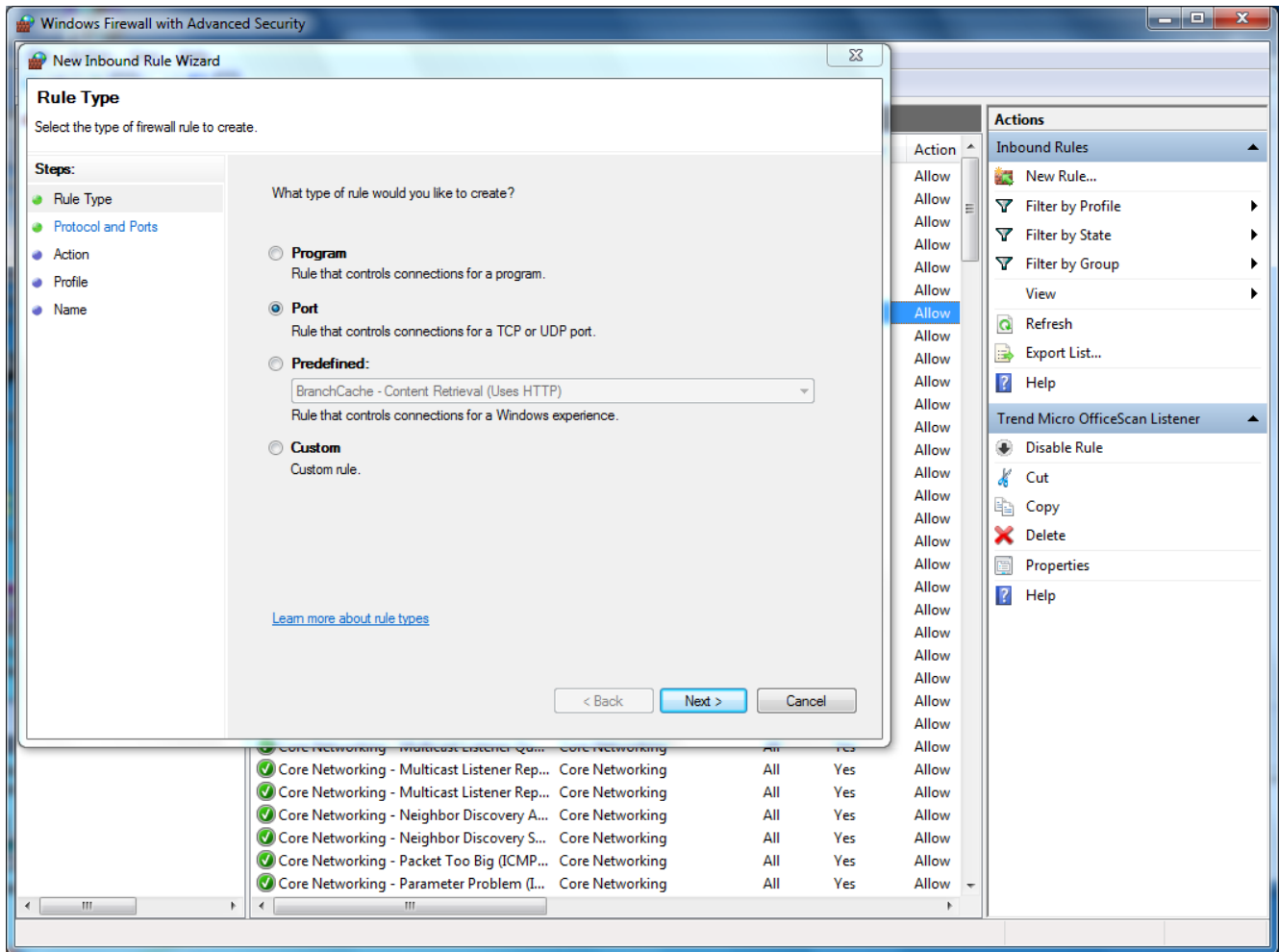


Figure 1-11 Windows Firewall window, Inbound Rules, Rule Type

4. Select the "Port" option and click "Next".

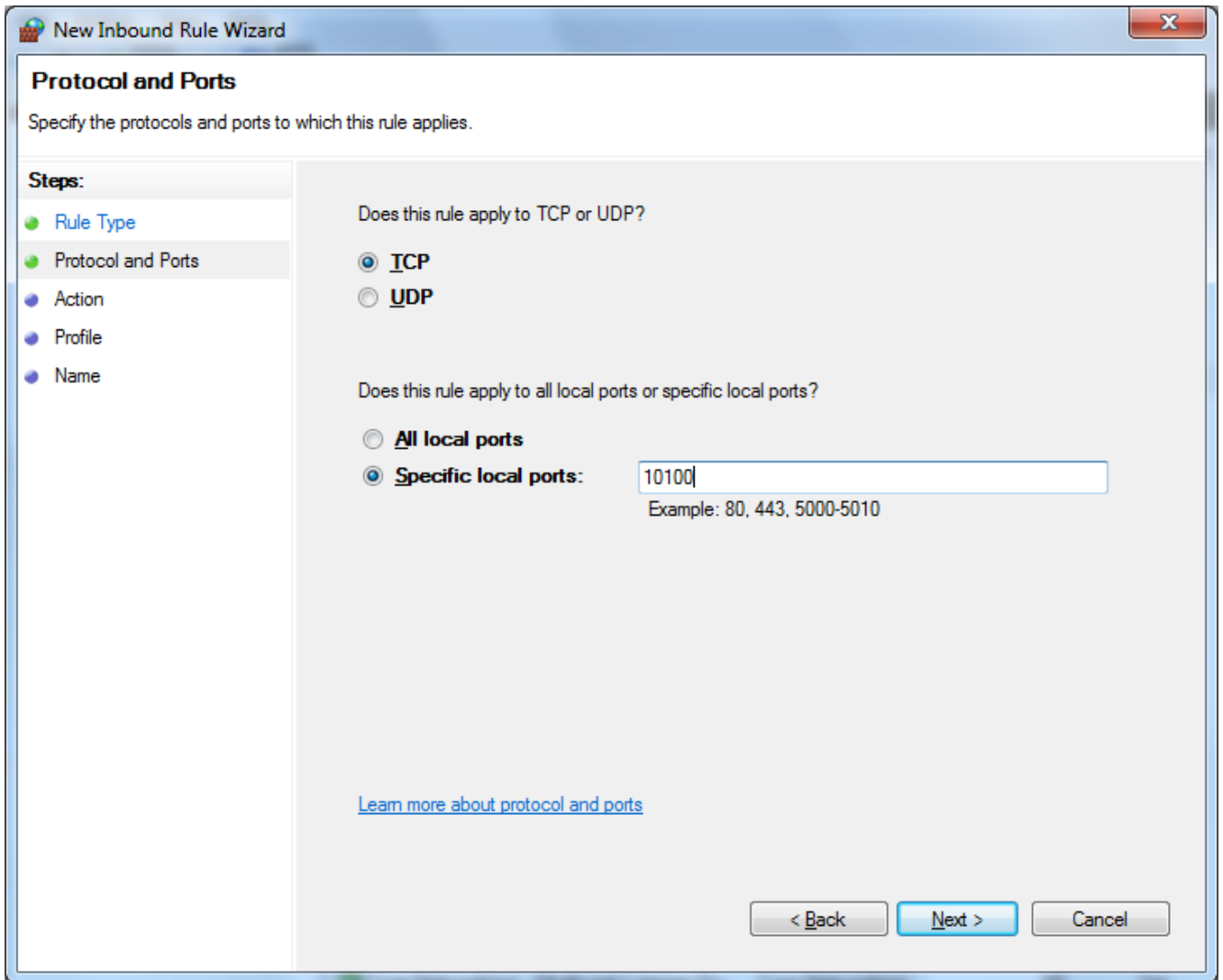


Figure 1-12 Windows Firewall window, Inbound Rules ... Protocol and Ports

5. Select the options "TCP" and "Specific local ports", enter the port number 10100 and click "Next".

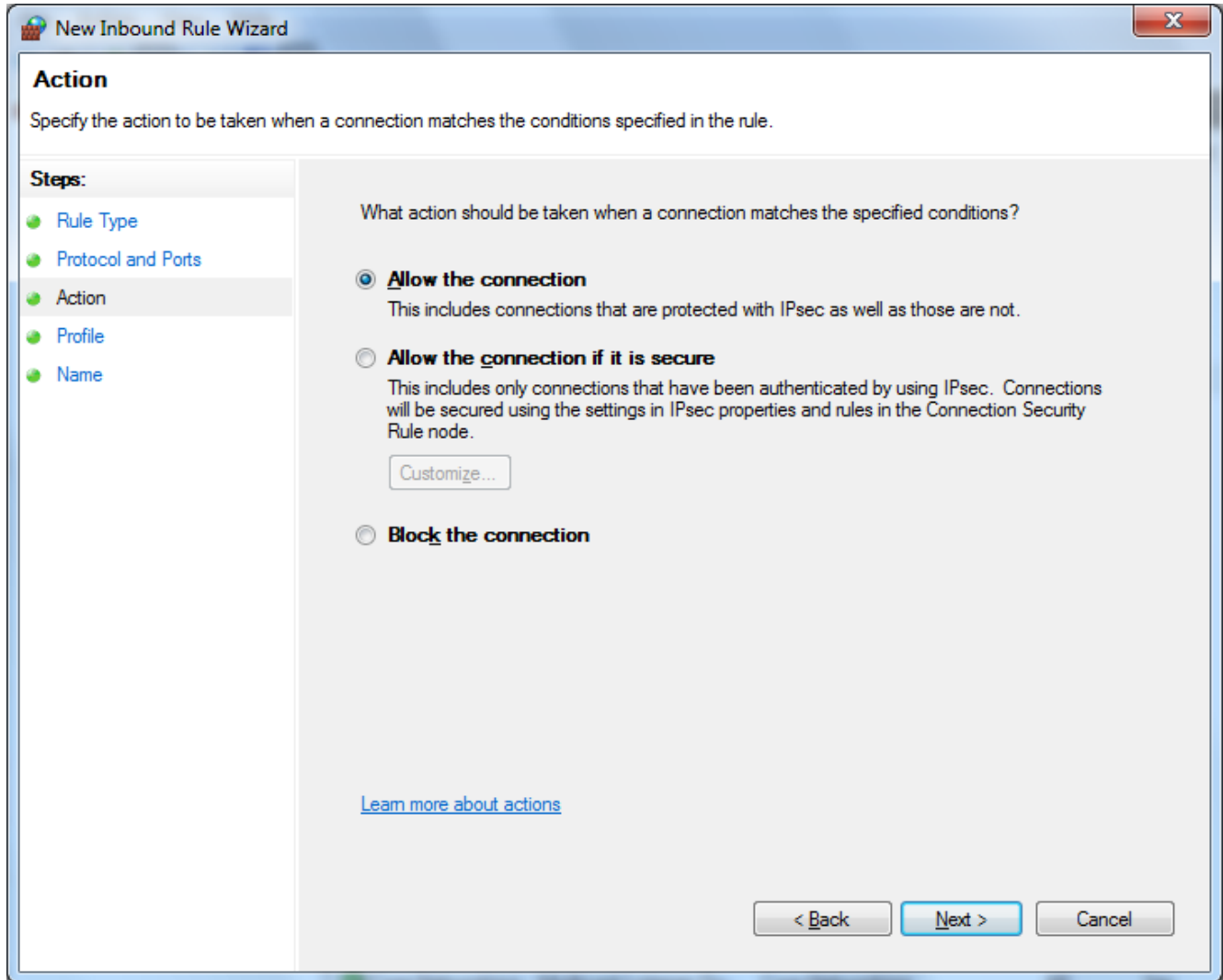


Figure 1-13 Windows Firewall window, Inbound Rules ... Action

6. Select the "Allow connection" option and click "Next".

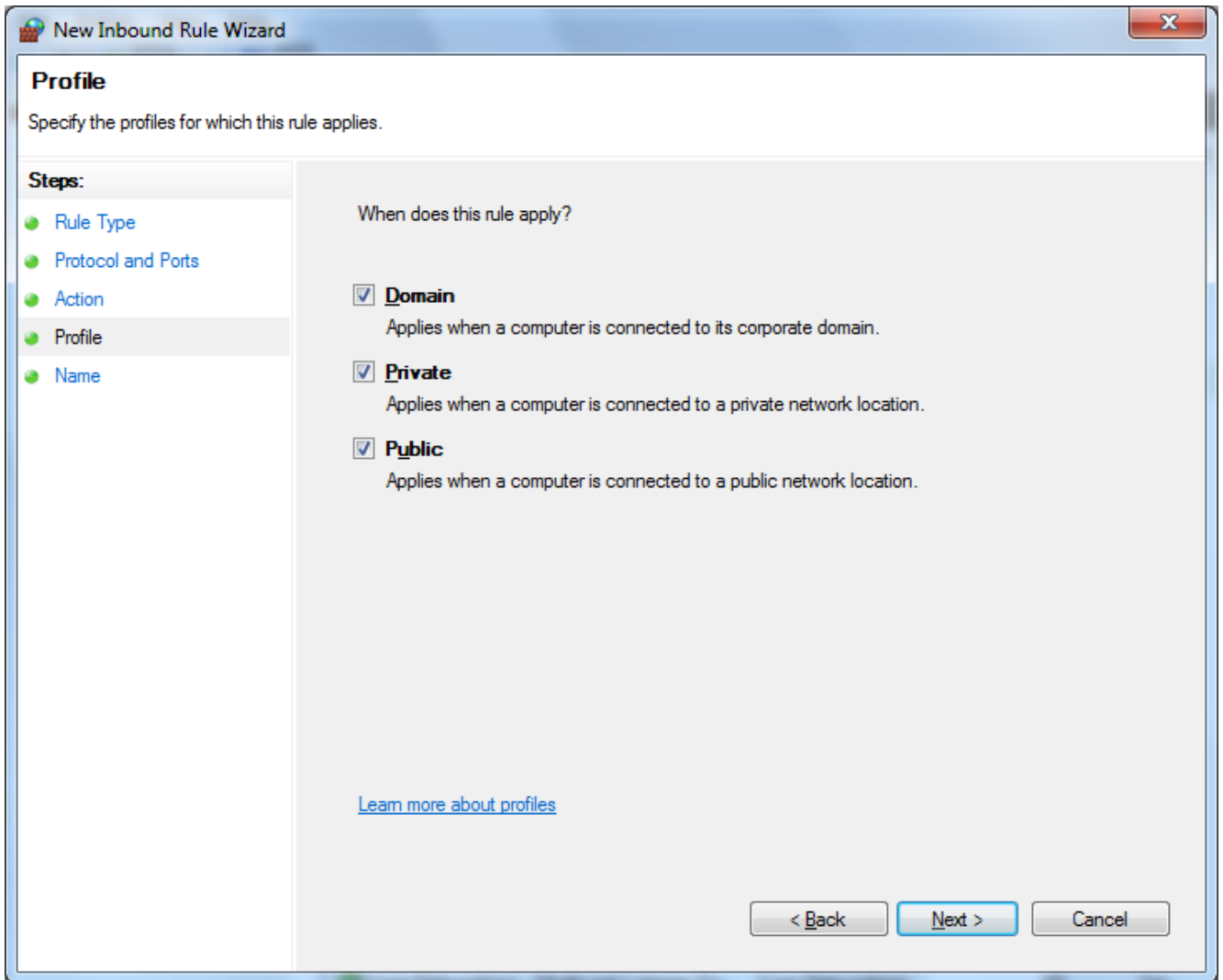


Figure 1-14 Windows Firewall window, Inbound Rules ... Profile

7. Enable the three options and click "Next".

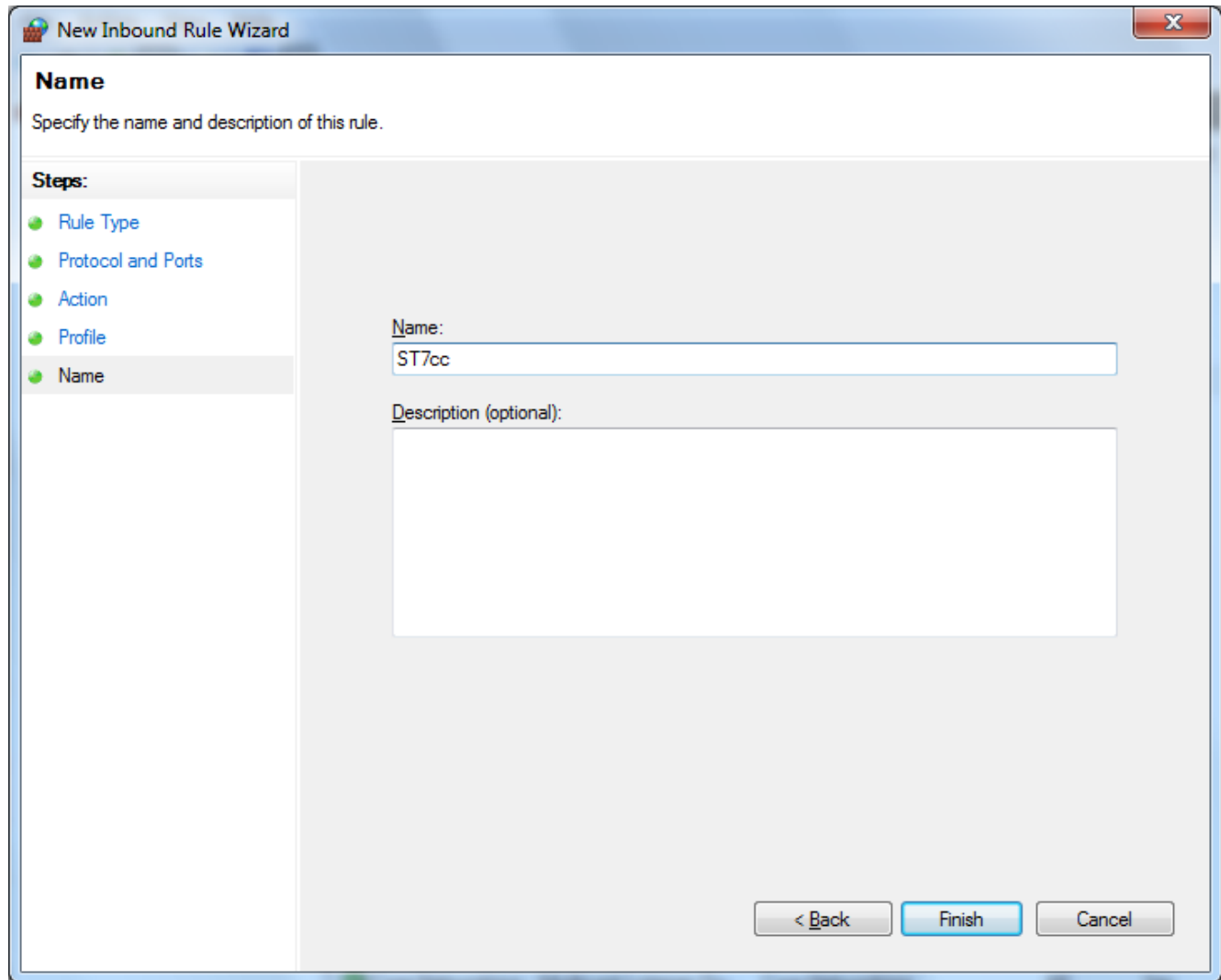


Figure 1-15 Windows Firewall window, Inbound Rules ... Name

8. Enter the name for the rule (in the example "ST7cc") and click "Finish".
Port 10100 is now opened.

1.7.3 Firewall settings with Windows Server 2008 R2 (64-bit)

Firewall settings with Windows Server 2008 R2 (64-bit)

1. Open the menu "Start > Control Panel > Windows Firewall".

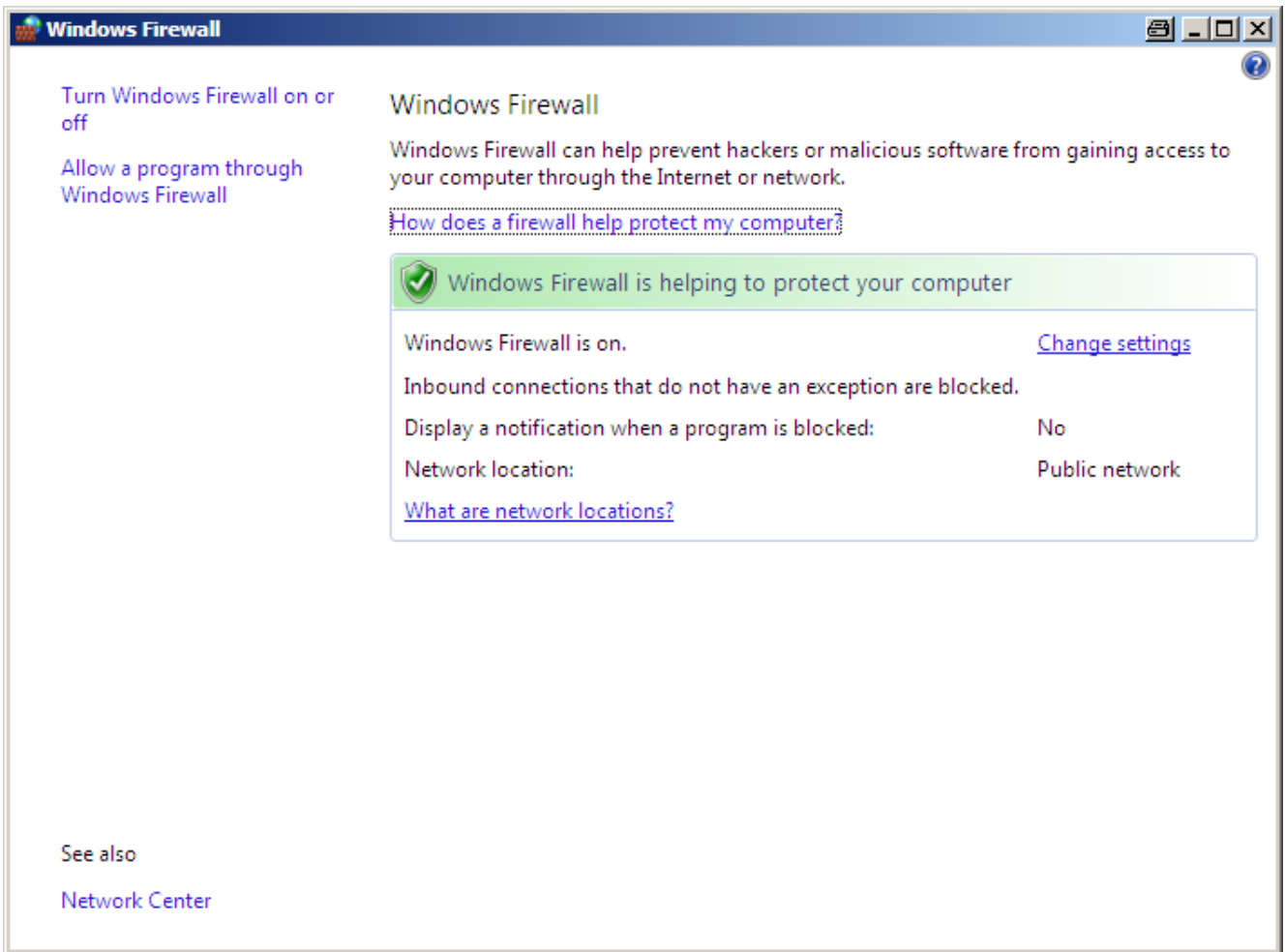


Figure 1-16 Window of the Windows Firewall

2. Click on "Change settings".

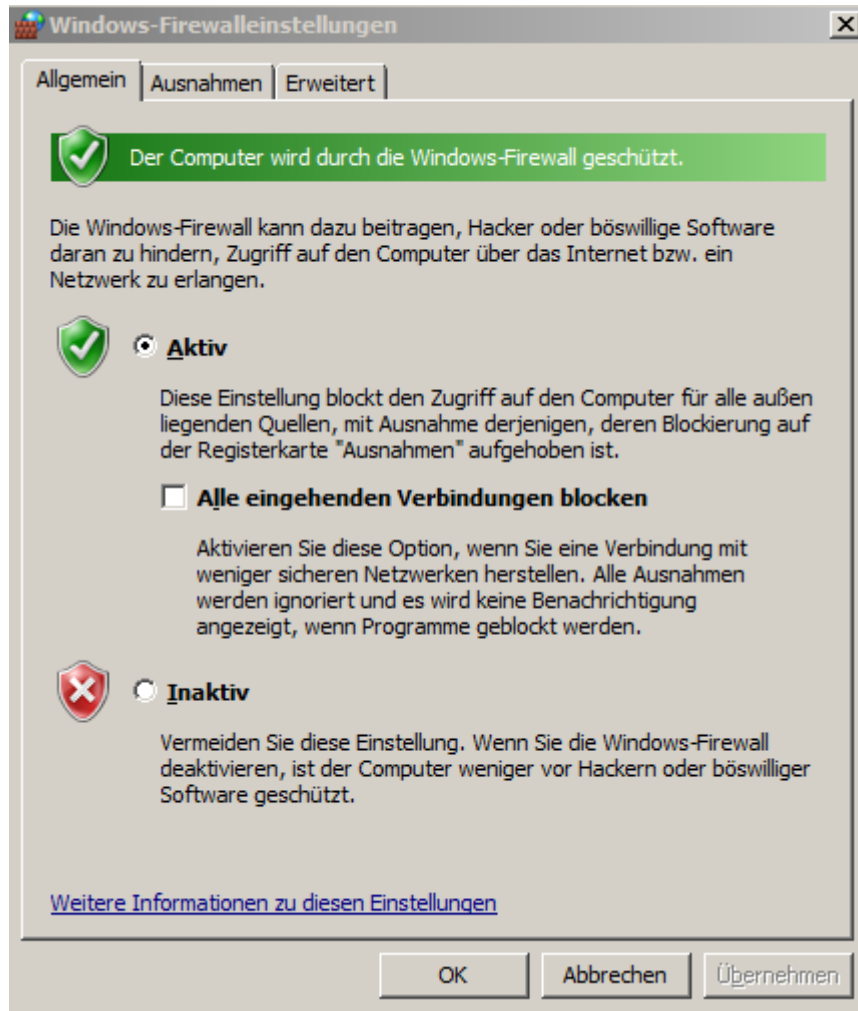


Figure 1-17 Window of the Windows Firewall

3. Open the "Exceptions" tab.

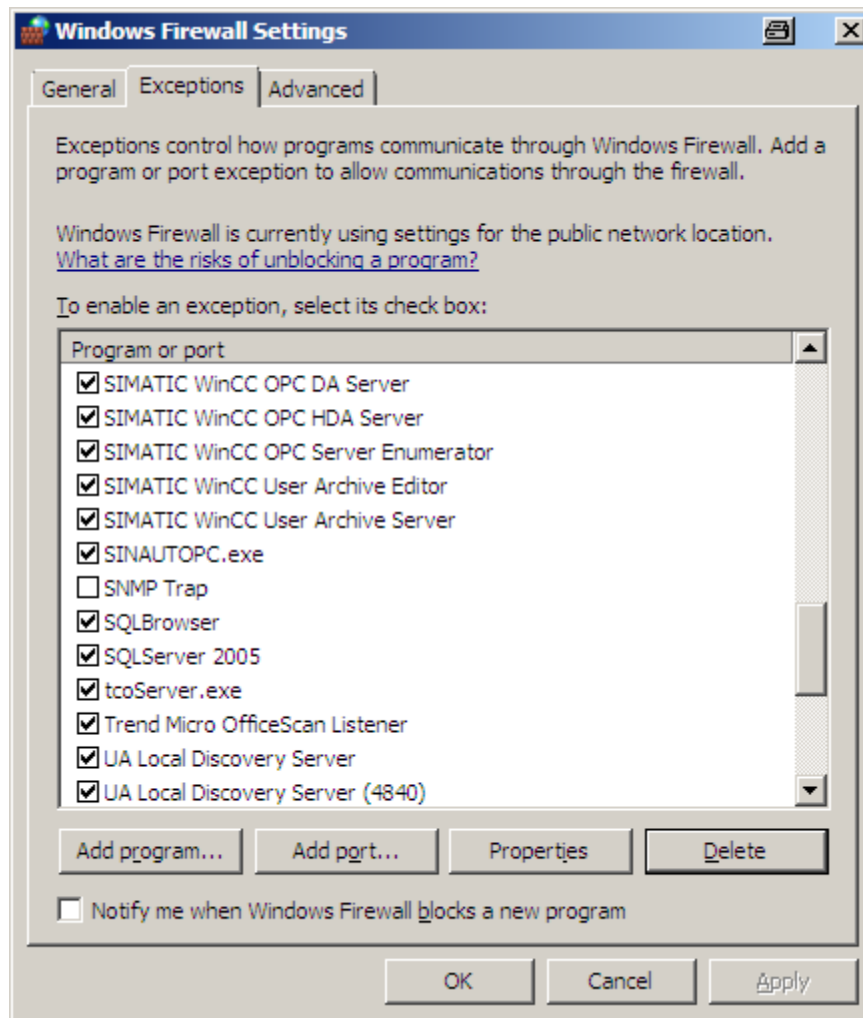


Figure 1-18 Windows Firewall window, "Exceptions" tab

4. Click the "Add port..." button.

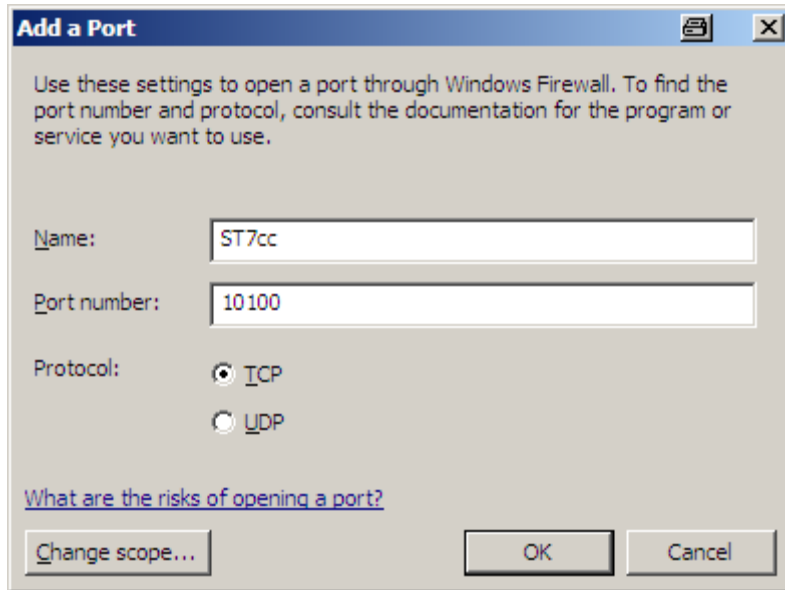


Figure 1-19 "Add Port" dialog box

5. Enter the name for the application (in the example "ST7cc") and the port number 10100.
6. Confirm the dialog box and the firewall window with OK.
Port 10100 is now opened.

Integrating ST7cc in a SINAUT network

2.1 Task

Task

To allow networked devices to communicate in an installation, configuration data for the components and the communication connections of these devices must be created and downloaded to the devices.

This configuration includes not only SIMATIC S7 stations and TIMs but also the PC stations to allow the communication relations between all devices of the installation to be specified. The PC on which ST7cc (runtime system) is installed is simply called the SINAUT PC in the remainder of the chapter.

The following descriptions assume that you have installed the hardware in your SINAUT PC, entered the IP addresses of the Ethernet CPs and configured a SINAUT network with SIMATIC STEP 7.

Integrating ST7cc in a SINAUT network

'Integrating ST7cc in a SINAUT network' means configuring the connections between ST7cc and the SINAUT network. The activities involved can be roughly divided into four stages:

Integrating the SINAUT PC in the STEP 7 project with the SINAUT stations.

- Inserting the PC station
 - Specifying the object properties of the PC station
 - Specifying the hardware configuration of the PC station; in other words, slots for the application (ST7) and communications modules.
- Configuring S7 connections between ST7cc and the local TIMs
- Including a redundant SINAUT PC (optional)
- Specifying the time service

Configuring the SINAUT PC under SIMATIC NET

- Downloading the configuration data to the PC
- Setting the Station Configuration Editor
- Configuring access points

Configuring the connections in the SINAUT configuration tool

- Adapting SINAUT subscriber numbers
- Configuring redundant ST7cc (optional)
- Configuring SINAUT connections
- Generating and compiling SINAUT data

2.2 Installing the hardware

Configuring the ST7cc project settings for communication. This configuration is described in Section Project settings: Communication (Page 132).

We recommend that you perform the configuration activities in this order.

To connect ST7cc with the SINAUT network, you require the software packages described in section Required software (Page 18).

2.2 Installing the hardware

How you install the communication modules for communication between the SINAUT PC and the SINAUT network is described in the SIMATIC NET documentation (see SIMATIC NET DVD). There, you will find all the Installation Instructions for the SIMATIC NET software, for the hardware (modules) and for configuration of and removal of drivers.

2.3 Installation of the SIMATIC NET PC software products

SIMATIC NET PC Software

To make the SINAUT PC capable of communication, the SIMATIC NET PC software and SIMATIC NCM PC / STEP 7 must be installed there.

Installation of the SIMATIC NET PC software is described on the DVD of the product. The Installation Instructions provide you with information on the following:

- Permitted operating systems
- You will find details on multilanguage versions and the required service packs for the supported operating systems in the readme file on the SIMATIC NET PC Software DVD.
- Permissions required for installation
- Problems after installing over a previous version
- Steps in installation. The installation steps are described in detail in the Installation Instructions and are only outlined here briefly.

Procedure

1. Register with the operating system with a login with administrator privileges
2. Close any active programs
3. Insert the DVD. With an automatic start, the greetings window appears. If this does not happen, start the "start.exe" program in the main directory of the DVD.
4. Read the readme file.
5. If not already installed, install the Acrobat Reader version 4.0 or higher.
6. Read the SIMATIC NET documentation, particularly information on converting if you have a SIMATIC NET PC software version installed on your computer lower than V12 or SIMATIC NET NCM PC lower than version 5.4 + SP4.

7. Install the SIMATIC NET PC software. Note the instructions on the Chinese language version.
8. Select the products you want to install. Activate the options "SIMATIC NET Vx" and "SIMATIC NCM PC Vx".
9. Install the licenses.
10. Complete installation.
11. Shut down your computer and install the communication modules in the PC.
12. Start your computer. After restarting your computer, the new hardware detection wizard opens. You will then be asked whether you want to install the software automatically. Select this option and confirm with Next and close the wizard when it has completed its tasks with Finish.

The computer now has the SIMATIC NET communication software that still needs to be configured.

2.4 Linking the SINAUT PC in the STEP 7 project

Using the NetPro SIMATIC tool, insert the SINAUT PC into the STEP 7 project with the SINAUT stations (see sections Integrating ST7cc in NetPro (Page 44) to Time service on the MPI and Ethernet bus (Page 66)).

Requirement

1. You already have a SINAUT project (STEP 7 project with SINAUT stations) in which only the interfacing of the SINAUT PC is missing.
2. This SINAUT project is located on a separate PG.

The data configured for the SINAUT PC on the PG will be downloaded later from the PG to the SINAUT PC over the MPI interface of the SINAUT PC.

Note

The SINAUT project can also be expanded on the SINAUT PC. To do this, STEP 7 and the SINAUT configuration software must be installed on the SINAUT PC. Refer to the documentation on the SIMATIC NET Software DVD.

2.4.1 Integrating ST7cc in NetPro

The following example shows the integration of a SINAUT PC in the STEP 7 project.

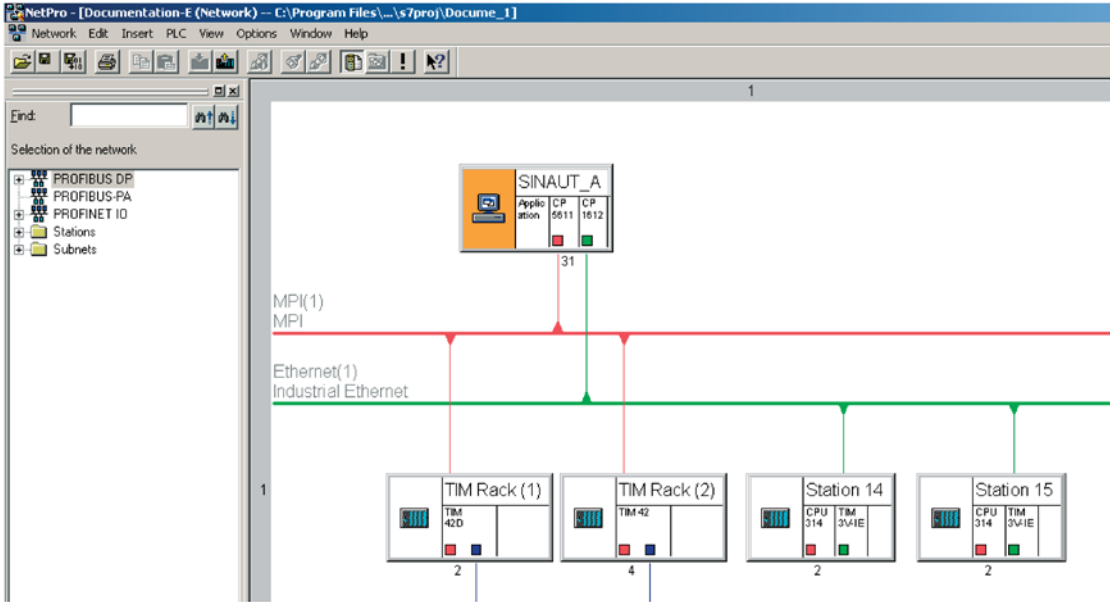


Figure 2-1 PC station *SINAUT_A* fully networked NetPro

Follow the steps outlined below:

1. Start the SIMATIC Manager and open your STEP 7 project with the SINAUT stations that you want to connect to ST7cc.
2. Start the NetPro SIMATIC tool by clicking on the Configure Network button in the toolbar of the SIMATIC Manager.

The NetPro dialog opens and displays the SINAUT project with its current networking status (see figure). If the Selection of the network objects window is not displayed, open it by clicking on the button shown in the figure.

Inserting a SIMATIC PC station and specifying the properties

From the Selection of the network objects tree (NetPro catalog), insert a SIMATIC PC station in your network configuration.

To display the name of the PC station, follow the steps below:

1. Right-click on the SIMATIC PC station.
2. In the open menu, select the Object Properties... option (see).

The Properties – SIMATIC PC Station dialog is displayed.

3. Enter the name of the PC station (for example SINAUT_A) in the Name input box of this dialog. The default name is SIMATIC PC station.
4. Close the dialog with *OK*.

Note

Note down the station name you enter here. When configuring your ST7cc device with the Station Configuration Editor, you must enter the same name for the PC station configured there (see section Initial configuration (Page 74)).

Specifying the hardware configuration of the PC station

To specify the hardware configuration of your PC station, start the HW Config tool by double-clicking on the icon of the SIMATIC PC station (figure). Before you change to HW Config, you may see a warning that you need to save changes made up to now in NetPro. Close this message by clicking the OK button.

If the catalog is not displayed, open it by clicking the catalog button shown in the figure.

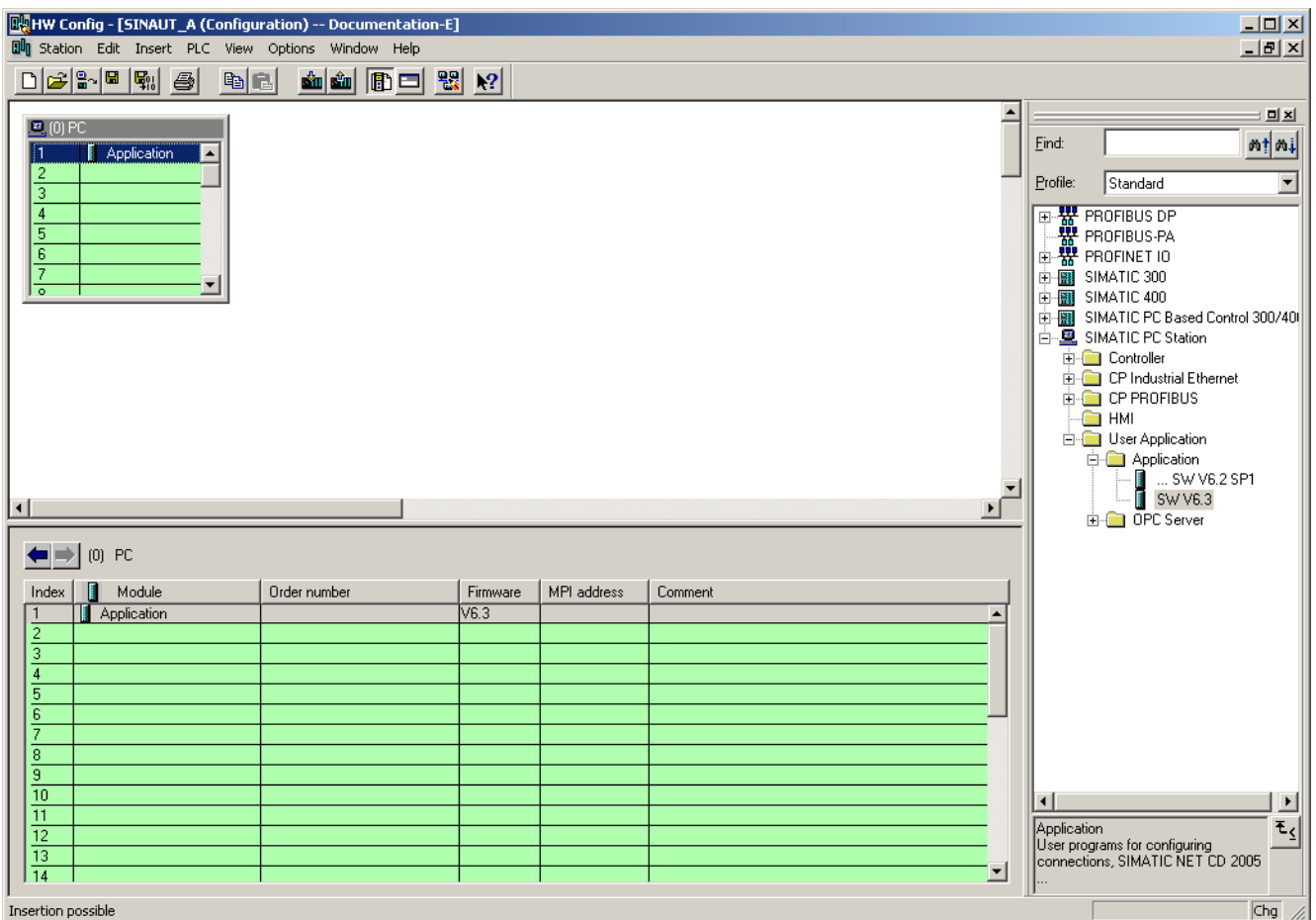


Figure 2-2 Empty HW Config dialog with the catalog open

The individual steps are described based on an example. In this example, the ST7 application is assigned to slot 1, the CP 5611 for MPI communication and the CP 1612 for Ethernet communication are assigned to slots 4 and 5.

Note

If you use the further applications in addition to ST7cc, for example OPC, you must use ST7 as the application name for ST7cc.

Note

Note down the configuration entered here in HW Config. When configuring your SINAUT PC in the SIMATIC NET PC Software, you must make the same settings.

Slot 1 (Application):

1. Select *SIMATIC PC Station > User Application > Application* in the catalog.
2. Drag the application to slot 1.

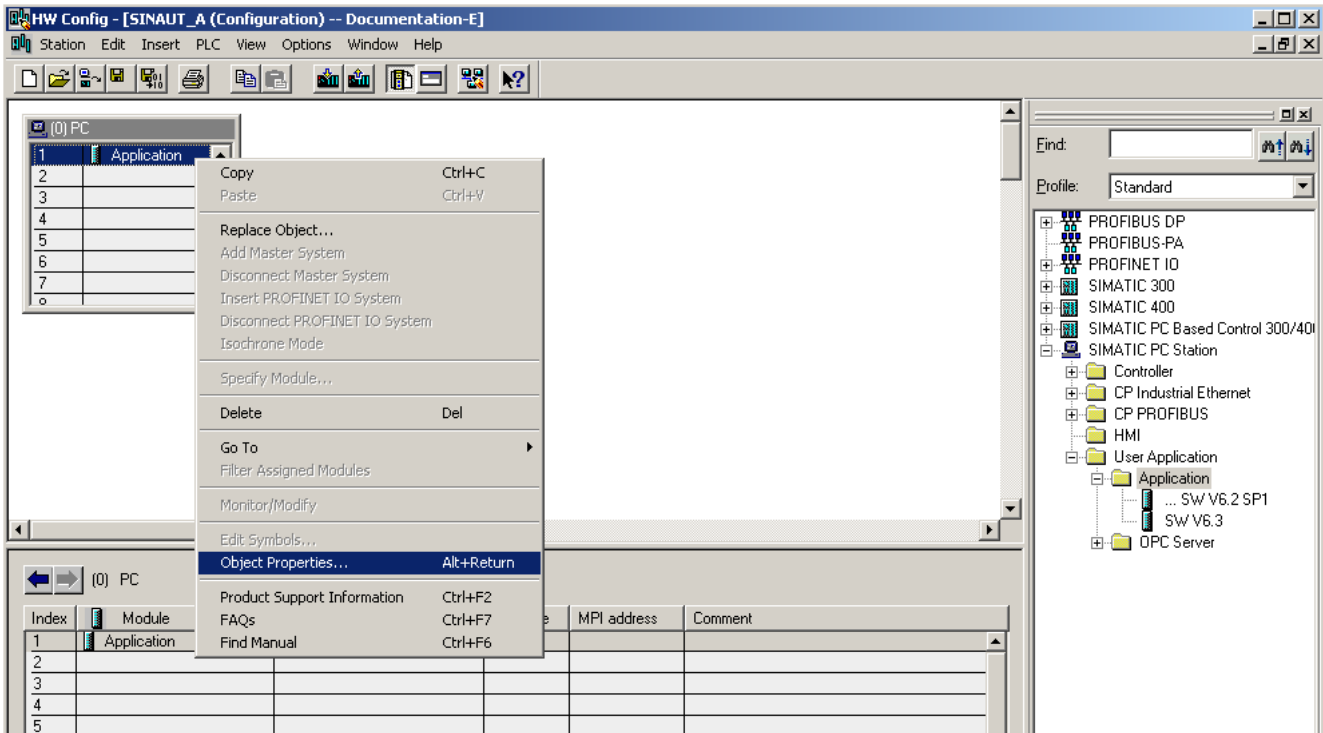


Figure 2-3 HW Config – Inserting the application and opening the object properties

Object properties of the application

1. Right-click on the application in slot 1.
2. Select *Object Properties...* in the menu that opens.

3. In the *General* tab of the *Properties – Application* dialog, change the name of the application to ST7.
4. Close the *Properties – Application* dialog with *OK*.

Slot 4 (CP for MPI communication)

1. Select *SIMATIC PC Station > CP PROFIBUS > CP 5611* in the catalog.
2. Drag the *CP 5611* to slot 4.

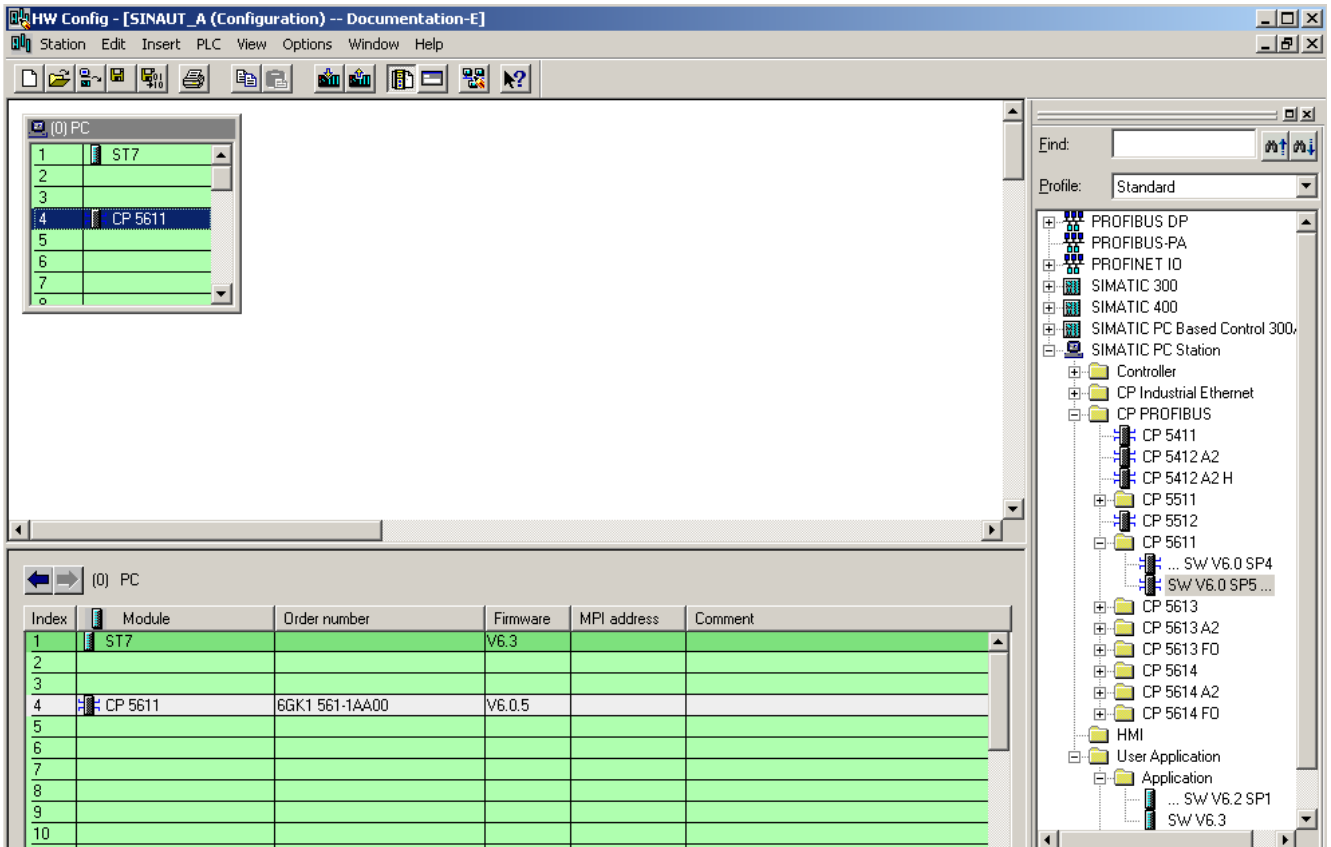


Figure 2-4 HW Config – inserting the CP 5611

3. When you insert the CP, the *Properties - PROFIBUS Interface CP 5611* dialog of the selected CP automatically opens.

The open *Properties - PROFIBUS Interface CP 5611* dialog does not require any entries. Close the dialog with *OK*.

In the next step, specify the object properties of the CP.

Specifying object properties of the CP

Since you are networking the module on an MPI network, you must set the type MPI for this module. Follow the steps outlined below:

1. Right-click on the *CP 5611* in slot 4.
2. In the open menu, select the *Object Properties...* option (see).

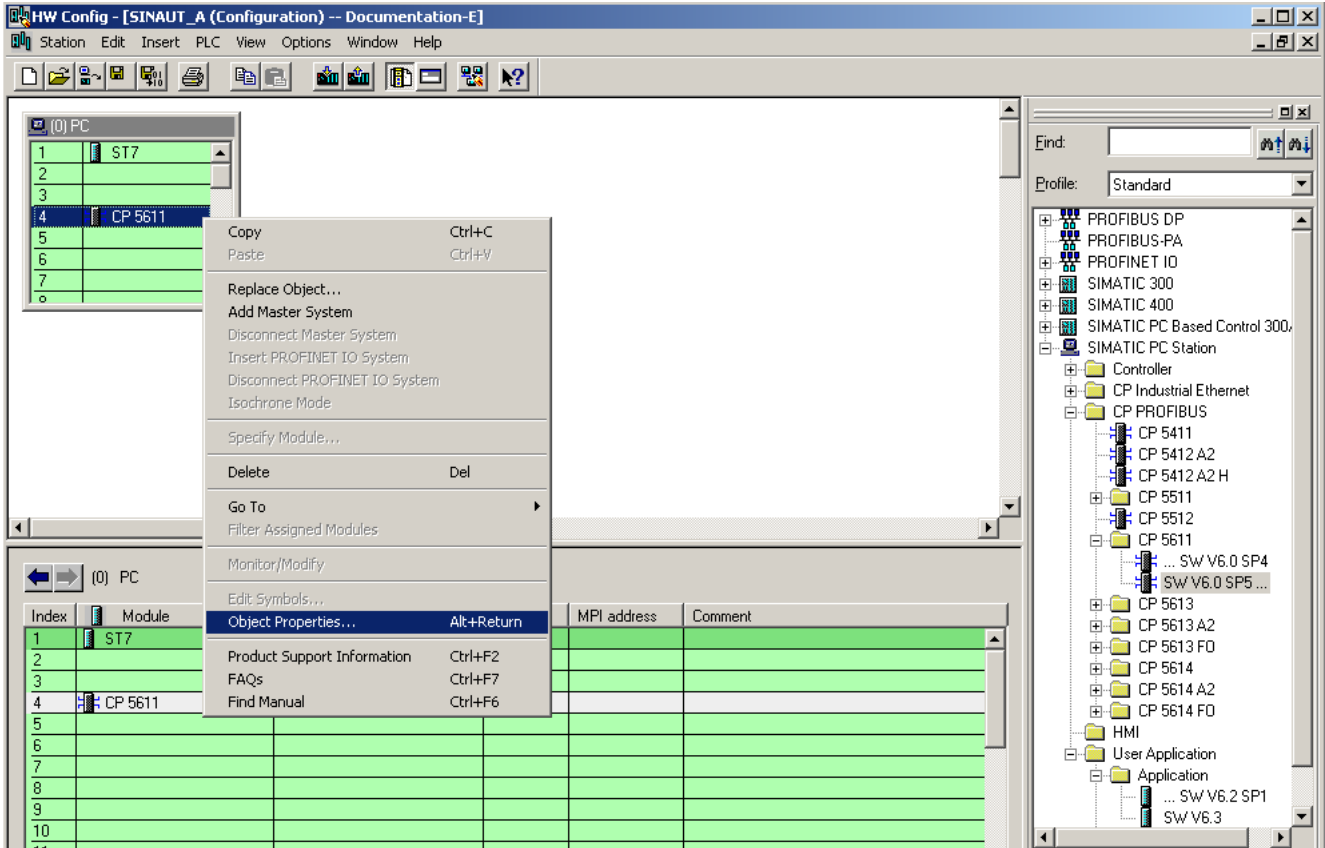


Figure 2-5 HW Config – Inserting the CP 5611, opening Object Properties

3. Select the option MPI in the Interface area in the Type list box (see figure).

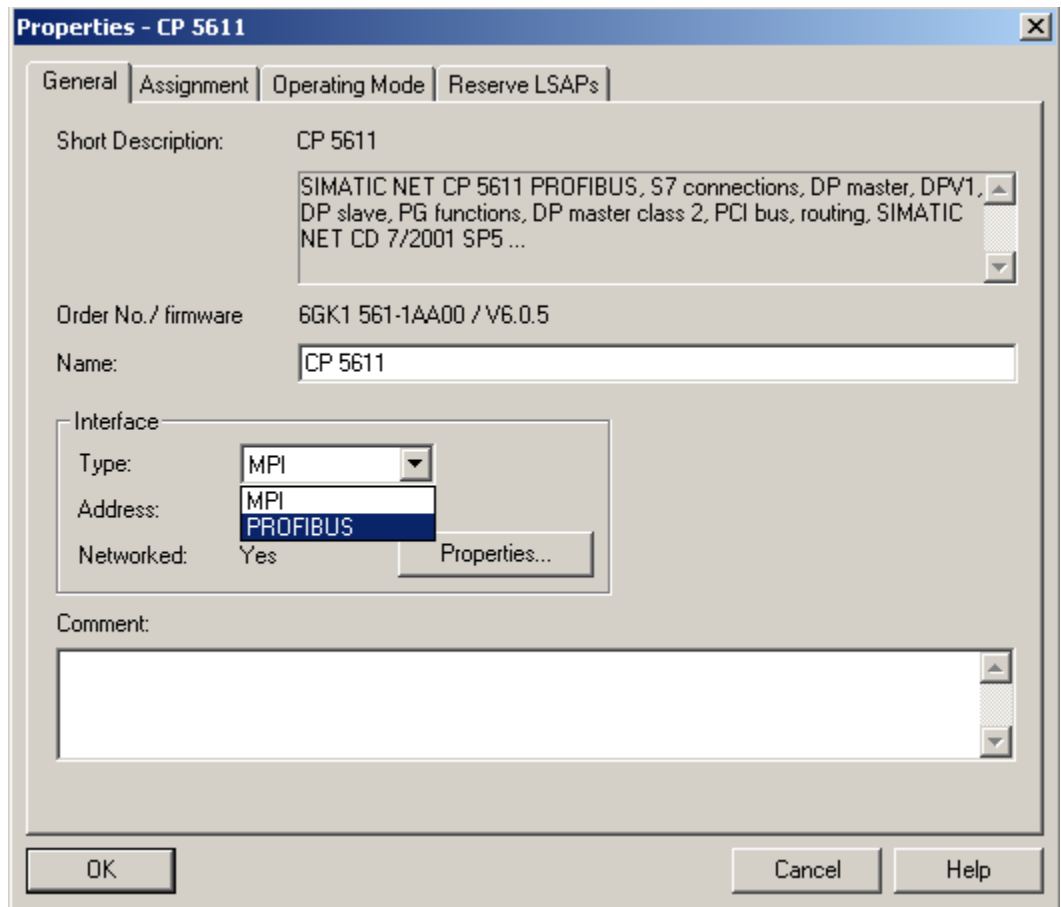


Figure 2-6 Object Properties CP 5611 – changing the interface type to MPI

4. Click on the Properties... button to make a further settings for the MPI interface.

Properties dialog of the MPI interface

The Properties - MPI interface CP 5611 dialog opens. Here, you create the connection to the MPI network and specify the MPI address (see figure).

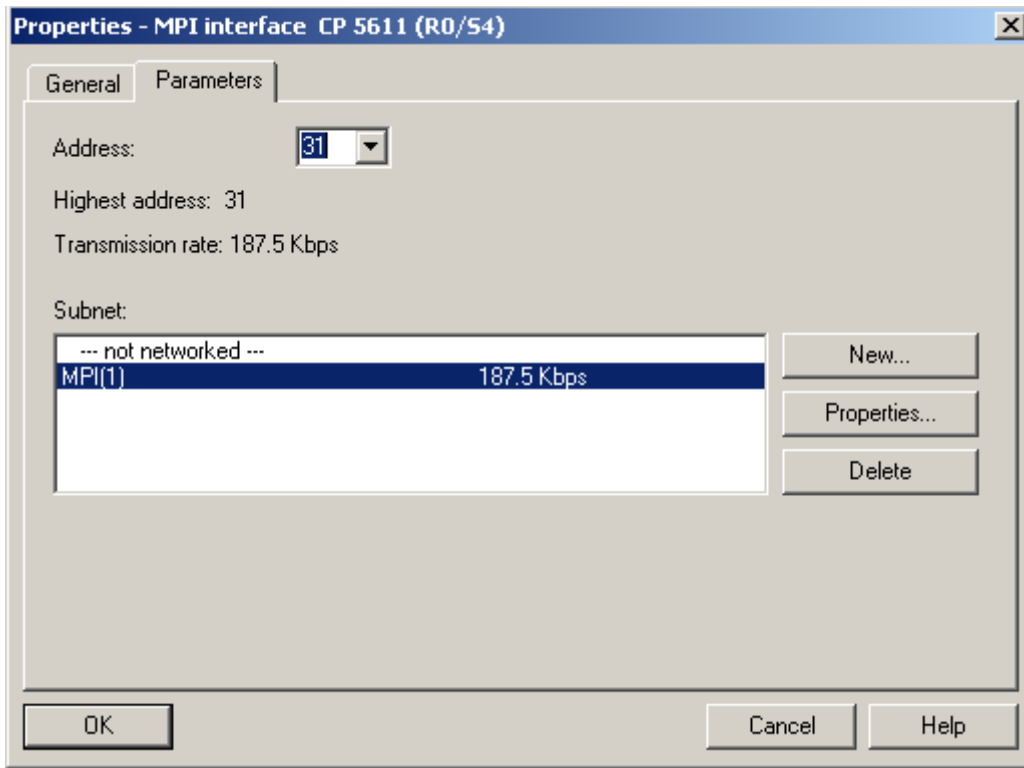


Figure 2-7 Properties dialog MPI interface CP 5611

1. Select the MPI network with which the PC station will be connected in the Subnet box.
Since there is only one MPI bus in this sample project, selecting the correct MPI bus in the Subnet list box is quite simple. If you use several MPI buses in your project, it is more efficient to assign names to the MPI buses. You assign these names in NetPro, in the Properties dialog, General tab.
2. Select a free MPI address, for example 31, from the Address list box.
3. Close the dialog with *OK*.

The Properties dialog of the CP 5611 opens again. The changes you have made are displayed.

4. No further entries are necessary in this dialog. Close the dialog with *OK*.

The HW Config dialog with the changed MPI address of the CP 5611 appears.

Note

You can also make the connection to the correct MPI bus directly in NetPro. There, the MPI node of the PC station can be connected to the correct MPI bus using the mouse. A free MPI address, for example 31 can also be set there.

Note

A station manager is inserted in slot 125. This is inserted automatically. You do not need to make any changes here.

If you do not want to insert anything else in your PC station, the settings in HW Config are complete. Click on the Save and Compile button to save the settings.

Note

The Save and Compile function automatically creates the XDB file containing the configuration data of the PC station. During initial configuration of the PC station with the Station Configuration Editor, you can adopt the configuration data in the SIMATIC NET PC Software without needing to enter it again.

If you also want to include an Ethernet CP in the hardware configuration of your PC station, for example to be able to connect stations over Ethernet as well, follow the steps below.

Slot 5 (CP for Ethernet communication)

Note

If you want to know more about specifying IP addresses, refer to the following documentation:

'IT in der Industrieautomatisierung; Planung und Einsatz von Ethernet-LAN-Techniken im Umfeld von SIMATIC-Produkten' by Mark Metter, Rainer Bucher, Publisher: Siemens Aktiengesellschaft, Berlin and Munich (ISBN 3-89578-166-5) Available only in German.

To assign an Ethernet CP to slot 5, follow the same steps as for slot 4:

1. Select *SIMATIC PC Station > CP Industrial Ethernet > CP 1612* in the catalog.
2. Drag the CP 1612 to slot 5

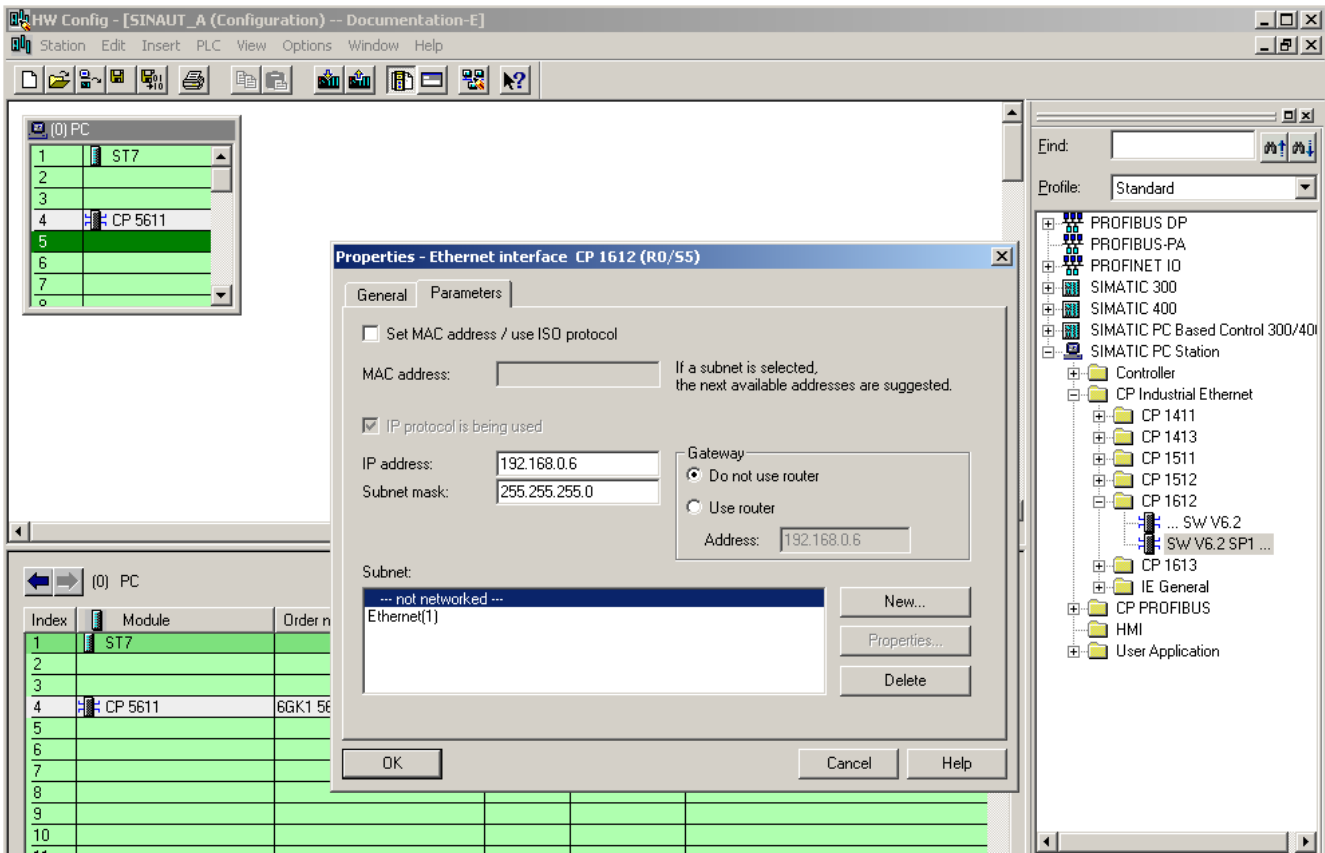


Figure 2-8 HW Config – completed configuration

When you insert the CP, the *Properties - Ethernet Interface* dialog of the selected CP opens.

Properties - Ethernet interface

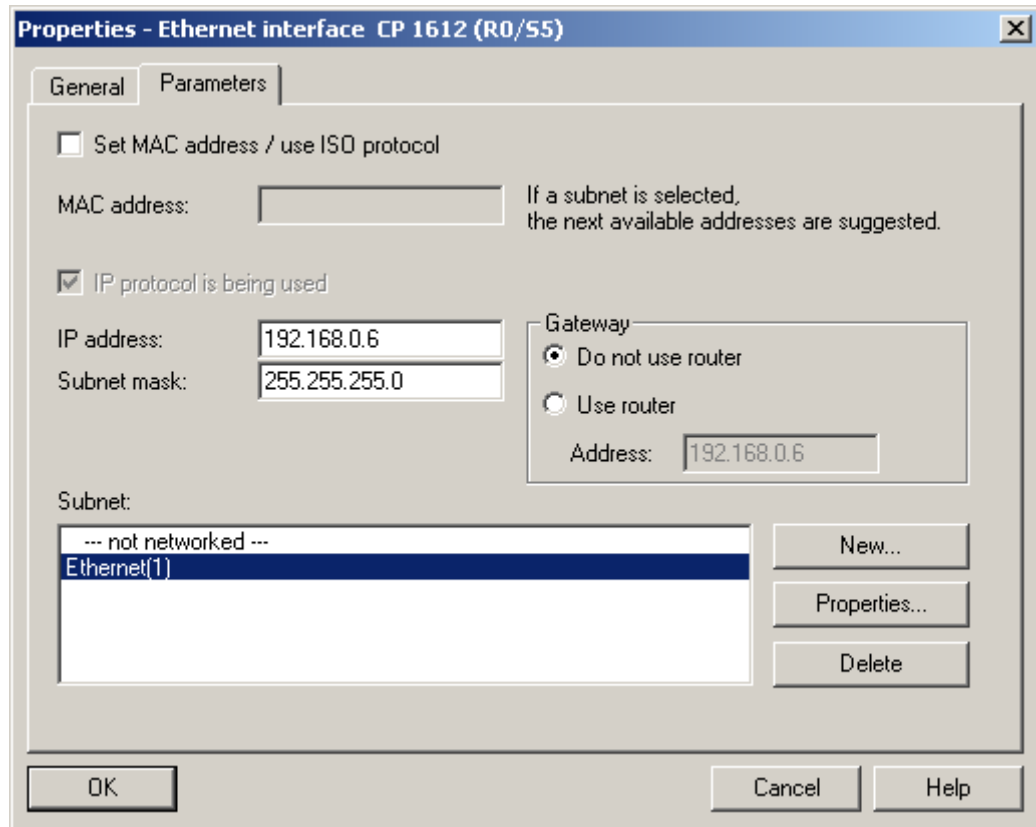


Figure 2-9 Properties - Ethernet interface CP 1612

1. In the IP address box, enter the IP address of your Ethernet interface (network node). The input box has a default entry that can be modified. Please make sure that you enter the correct IP address of your CP.
2. Enter the subnet mask in the Subnet mask input box to further specify the network part of the IP address. The input box has a default entry that can be modified. Please make sure that you enter the correct subnet mask.
3. In the Subnet list, select the subnet with which you want to network your CP.
4. In the Gateway area, select Do not use router if your communication nodes are all networked with the same subnet as this Ethernet interface.
5. Select Use router in the Gateway field if your communication nodes are also outside the subnet of this Ethernet interface; in other words, in other subnets.

In this case, enter the IP address of the router in the Address box. The input box has a default entry that can be modified.

Completing work in HW Config

When you have completed your activities in HW Config, save and compile your entries.

1. Click on the *Save and Compile* button to save the settings or select the *Station / Save and Compile* menu.
2. Once *Save and compile* is complete, close HW Config.

The NetPro dialog appears again. The PC station now has the name SINAUT_A and is connected to the MPI network MPI(1) over MPI address 31 and with the Industrial Ethernet network Ethernet(1)

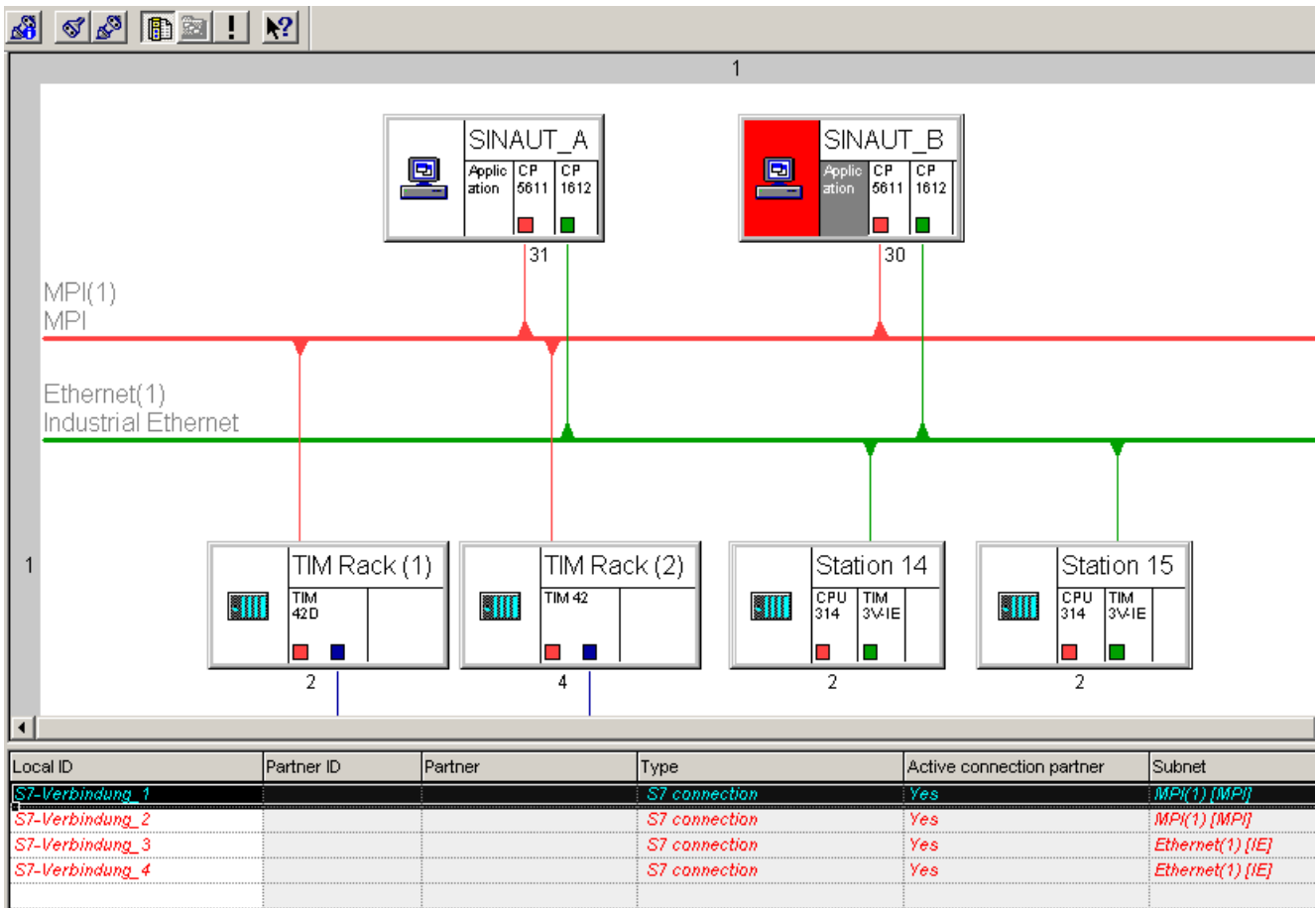


Figure 2-10 PC station SINAUT_A fully networked NetPro

2.4.2 Configuring an S7 connection between local TIMs and ST7cc

Configuring an S7 connection

Note

With the SINAUT ST7 configuration software, you normally generate all the data (SDBs) required by the TIMs to communicate with all partners in the network.

With the SINAUT ST7 configuration software up to version V3.4, it is, however, not possible to generate the connection data of ST7cc to a local TIM (in the example, the connection data from the SINAUT PC to the two TIMs in TIM rack(1) and (2)) and to TIMs on the Ethernet bus. These connections had to be configured in NetPro as usual in SIMATIC.

With the SINAUT ST7 configuration software version V3.5, the connection data for the local TIMs is also generated. During generation, the ST7 configuration software recognizes whether a complete S7 connection already exists and adopts this data without any changes.

If you are not working with the SINAUT ST7 configuration software version V3.5 or higher, the following example describes the configuration of the connections required between ST7cc and the local TIMs with NetPro:

- Connection between SINAUT_A and the TIM in TIM rack(1)
- Connection between SINAUT_A and the TIM in TIM rack(2)
- Connection between SINAUT_A and the TIM in station 14
- Connection between SINAUT_A and the TIM in station 15

Inserting connections with NetPro

To configure S7 connections, select the *application* in the PC station (in the example *SINAUT_A*).

An (empty) connection table opens in the lower half of the NetPro dialog. If the table is not displayed, move the cursor to the lower edge of the NetPro dialog until the cursor changes to two horizontal lines. Holding down the left mouse button, drag this boundary upwards. The connection table is now visible.

Inserting a new connection

Note

To insert a connection, make sure that you select the application (ST7) in the SINAUT PC as the starting point of the connection and not the TIM!

There are two ways of inserting a new connection.

Method 1, see figure:

1. Right-click on the *Application*.
2. Select the *Insert New Connection* option in the open menu.

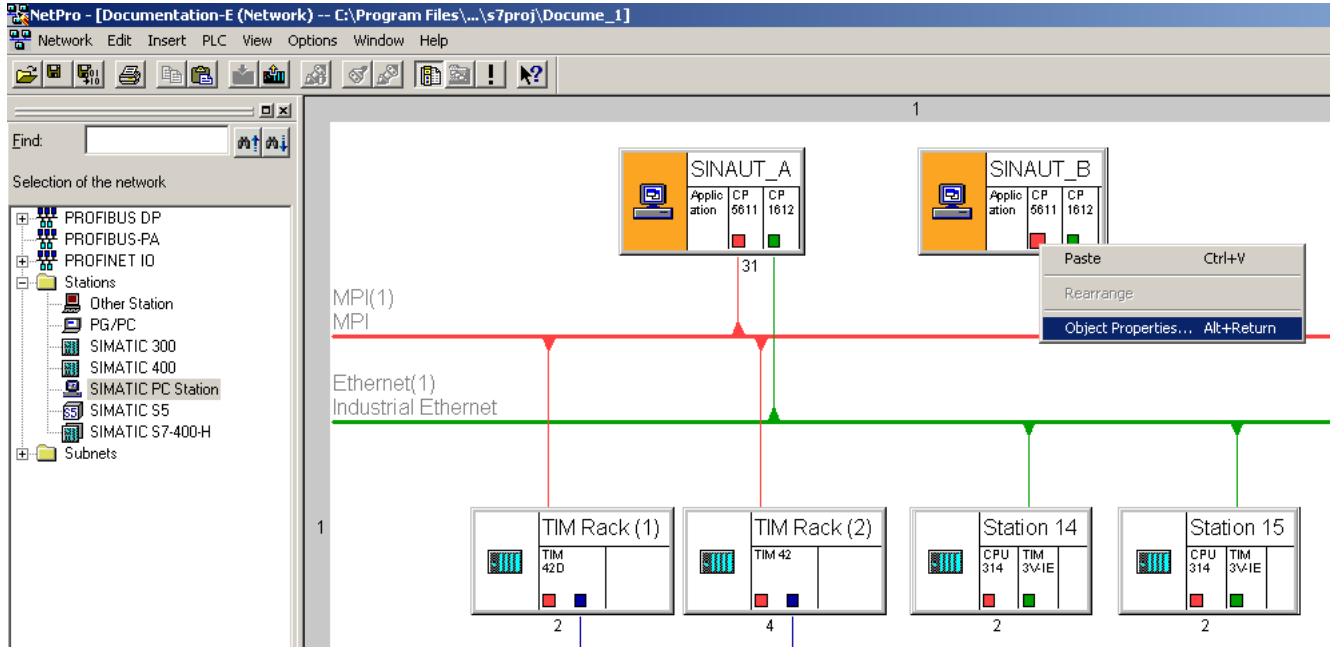


Figure 2-11 NetPro – Inserting a new connection, 1st method

Method 2:

The second method uses the connection table in the lower part of the NetPro dialog.

1. Right-click in the connection table.
2. Select the *Insert New Connection* option in the open menu.

After selecting the Insert New Connection option, the Insert New Connection dialog opens (see figure) and displays the possible connection partners in your project.

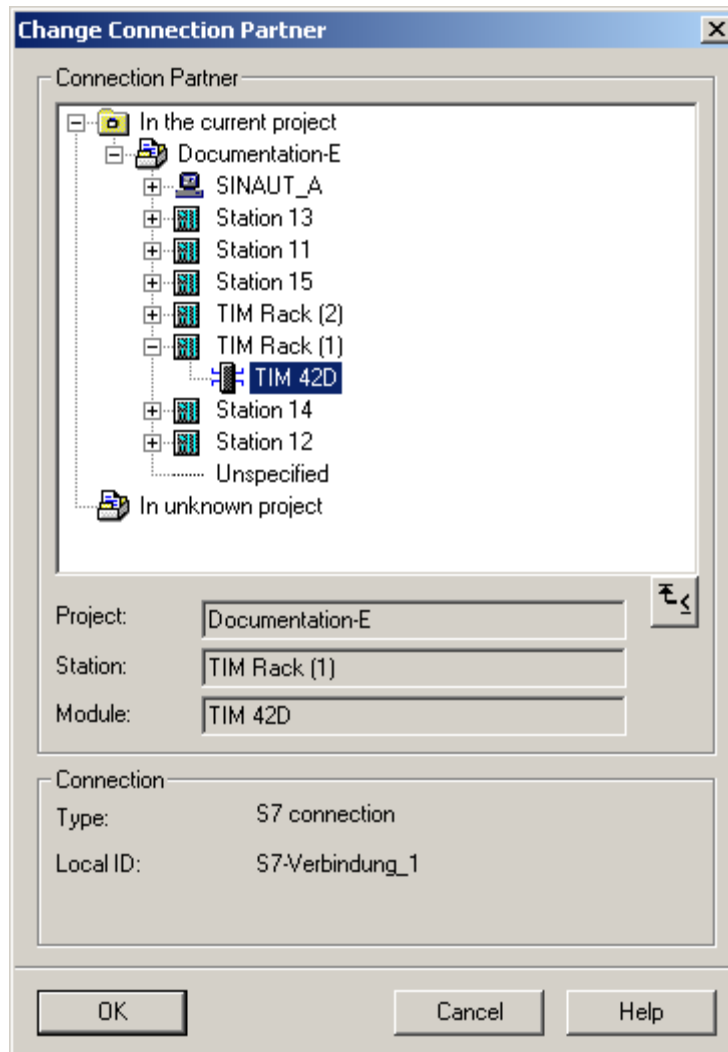


Figure 2-12 NetPro – Insert New Connection dialog

1. Select the connection partner for each connection from the Connection Partner object tree. The connection partner is always the local TIM, see figure.
2. Please make sure that the option *S7 connection* is entered in the *Type* list box.
3. Select the *Display properties before inserting* option so that a further dialog opens in which you can enter additional parameters.

- 4. Close the dialog with *OK*.

The Properties – S7 connection dialog opens (see figure). This displays the currently configured connection path over MPI or Ethernet from ST7cc to the local TIM.

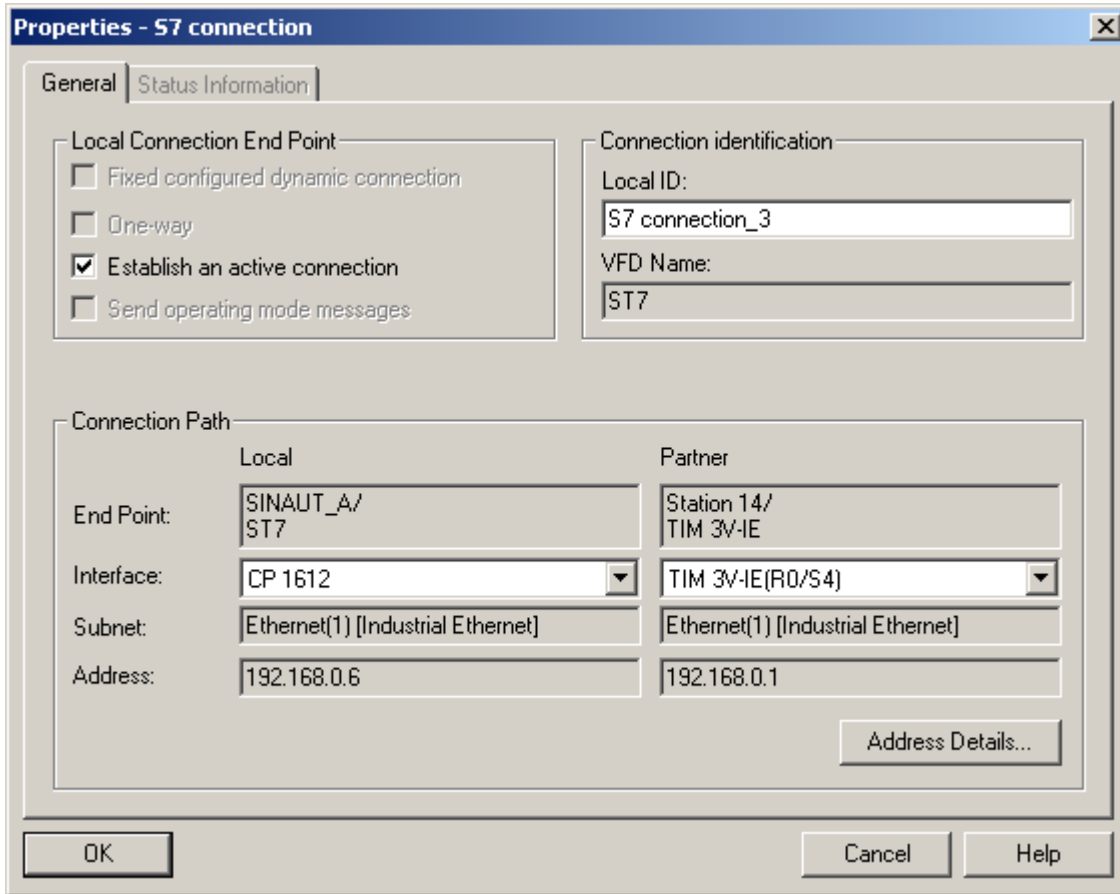


Figure 2-13 NetPro – Properties - S7 connection dialog

The Local ID input box displays the default ID of the connection (in the example S7 connection_1 or _3, see figure). You can change the connection ID. No change is necessary for SINAUT but this can nevertheless be changed by the user if necessary.

Note

If you are working with a SINAUT ST7cc version lower than V2.5, there are special rules for the connection IDs that you will find in the previous description.

- 5. Close the dialog with *OK*.

The NetPro dialog appears again and displays the connection table of the configured connection with all its parameters.

Table 2- 1 NetPro – PC station connection (SINAUT_A) fully configured

Local ID	Partner ID	Partner
S7_connection_1	1	TIM rack (1) / TIM 42D
S7_connection_2	1	TIM rack (2) / TIM 42
S7_connection_3	1	Station 14 / TIM 3V-IE
S7_connection_4	1	Station 15 / TIM 3V-IE

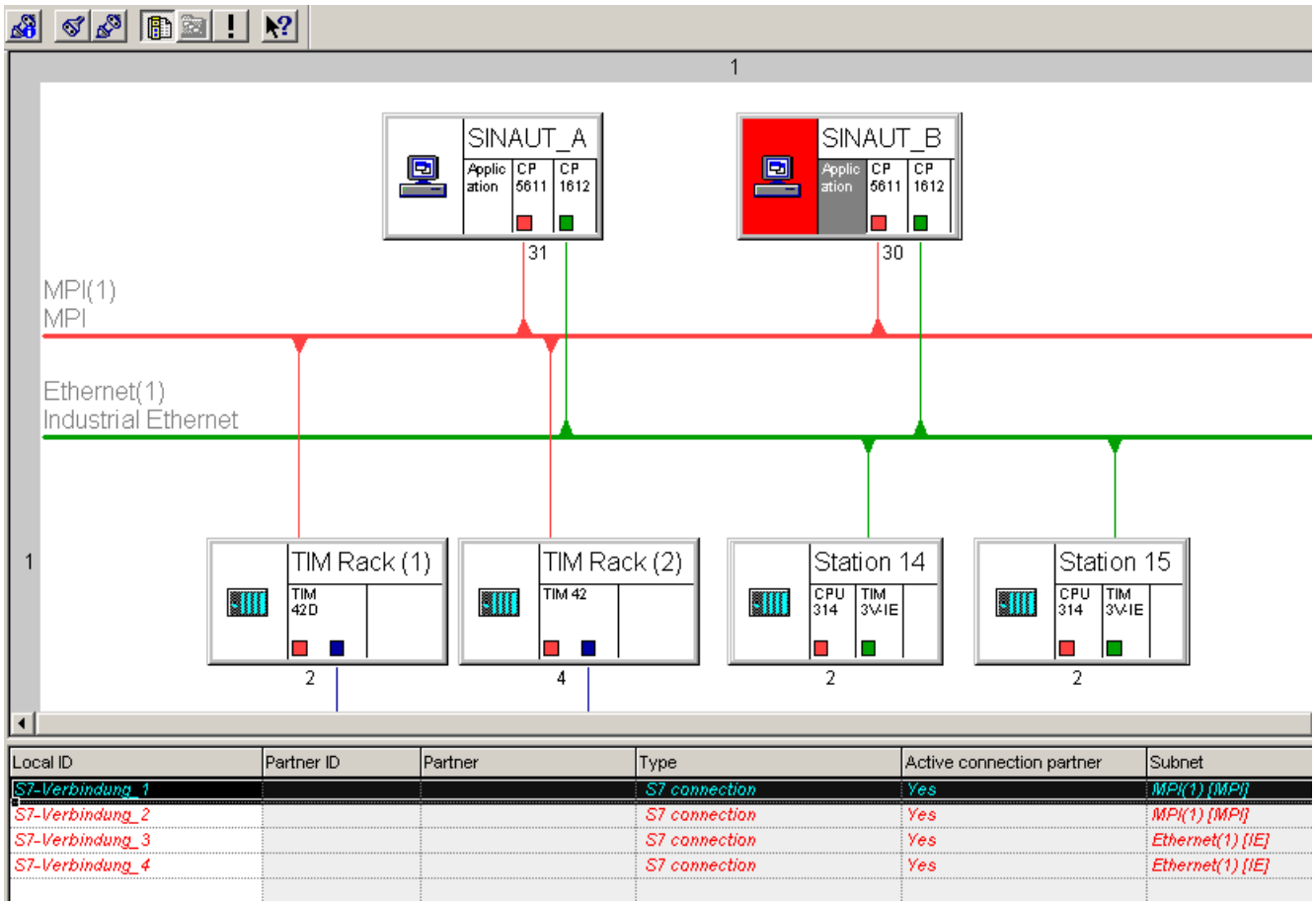


Figure 2-14 NetPro – PC station connection (SINAUT_A) fully configured

Note

Note down the local IDs you have specified here. When configuring ST7cc Project Settings: Communication (see section Project settings: Communication (Page 132)), you must specify the SINAUT subscriber number, the local ID and the application access point for each local TIM.

When you have configured all connections from the PC station to the local TIMs, integration of the SINAUT PC is completed.

Changing or checking the connection configuration later

If you want to review or modify the connection is configured in NetPro at a later point in time, you can do this using the connection table in the NetPro dialog (see figure).

1. First click on the required connection in the table (the row is shown on a black background).
2. Right-click on this row.
3. Select *Object Properties...* in the menu that opens.

Further activities

If you want to configure a redundant ST7cc, please read section Integrating a redundant SINAUT PC (Page 60). If you do not have a redundant ST7cc, continue at section Time service on the MPI and Ethernet bus (Page 66).

2.4.3 Integrating a redundant SINAUT PC

With a redundant ST7cc, a second SIMATIC PC station must be integrated in NetPro and the required connections configured. This procedure is described below.

Note

With a redundant ST7cc, the two PC stations are grouped together later in the SINAUT subscriber administration to form one SINAUT subscriber. This is possible with the SINAUT ST7 configuration software as of V3.1. For the user, this means that only one subscriber needs to be entered when configuring the SINAUT objects; the local TIM, nevertheless, supplies both redundant destination subscribers. The local TIM is not interested in the details of the redundancy functionality of the ST7cc target system. For the local TIM, the transmission of a message to the redundant target system is also successfully completed even when one redundant partner can be reached due to a disruption.

A redundant ST7cc functions only in connection with TIMs with firmware V3.58 or higher. This, however, applies only to the local TIMs connected directly to the redundant ST7cc over MPI or the Ethernet bus. In the sample project, these are the TIMs in TIM rack(1), TIM rack(2), station 14 and 15.

Inserting a second SIMATIC PC station

Two methods are possible:

1. You configure the second SINAUT PC step-by-step in exactly the same way as the first SINAUT PC (see section Integrating ST7cc in NetPro (Page 44)). As with the first SINAUT PC, you can then have the S7 connections between the local TIMs and ST7cc displayed automatically by the SINAUT ST7 configuration software or, as normal in SIMATIC, configure these in NetPro.
2. If you have already configured a SINAUT PC and the second PC is identical or almost identical to the first, the most efficient method is to make a copy of the SINAUT PC you have already configured. In this case, you must, however, delete the incomplete S7

connections in NetPro before you can have the S7 connections created automatically with the SINAUT ST7 configuration software. The incomplete S7 connections are displayed in red in the lower part of the NetPro dialog.

Inserting a SINAUT-PC by copying:

In the remainder of this example, it is assumed that the second SINAUT PC is identical to the first.

Copying:

To copy the SINAUT PC, follow the steps outlined below:

1. Right-click on the icon of the PC station, see figure.
2. Select the *Copy* function in the context menu.
3. Right-click on a free area of the NetPro dialog.
4. Select the *Paste* function in the context menu.

Normally, the copied SINAUT PC is not inserted and the location at which you right-clicked in the NetPro dialog. The window, however, changes automatically to the location where the SINAUT PC was actually inserted.

5. Move the SINAUT PC to the required position with the mouse, for example beside the previously installed SINAUT PC.
6. In the object properties, change the default name of the copied PC, for example to *SINAUT_B*.

Connecting CPs with the MPI or Ethernet bus

You must now connect the CP 5611 and CP 1612 of *SINAUT_B* to the same MPI or Ethernet bus as *SINAUT_A*.

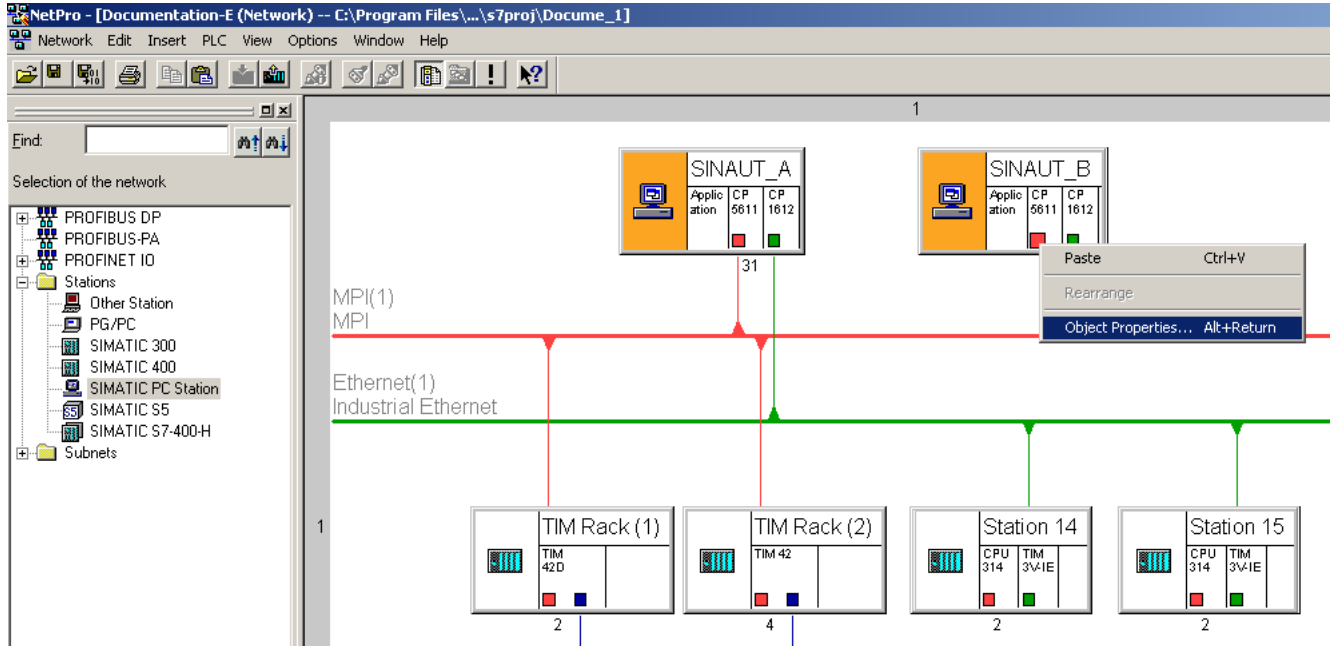


Figure 2-15 NetPro – selecting the object properties of the MPI node

1. Right-click on the red MPI node of *SINAUT_B*.
2. Select Object Properties... in the menu that opens. (see figure) The Properties – MPI interface... dialog opens.
3. Select the *Parameters* tab.
4. In the *Subnet* box, select the MPI network with which you want to connect *SINAUT_B*.
5. The *Address* list box then only displays the MPI addresses that are still free for this network.
6. Select one of the free MPI addresses (dialog, see figure).
7. Close the dialog with *OK*.
8. In the NetPro window, *SINAUT_B* is now connected to the MPI network and has the selected MPI address.
9. Now connect the *SINAUT_B* PC station with the Ethernet bus.
10. Right-click on the green Ethernet node of *SINAUT_B*.
11. Select the Object Properties option in the open menu as shown in the figure. The Properties - Ethernet interface... dialog opens.
12. Select the *Parameters* tab.
13. In the Subnet box, select the Ethernet network with which you want to connect *SINAUT_B*.
14. Select a free IP address (dialog, see figure)

15. Close the dialog with *OK*.
16. In the NetPro dialog, *SINAUT_B* is now also connected to the Ethernet network.

Configuring S7 connections for the second PC

Once the MPI and Ethernet connections have been created, you need to configure the connections from SINAUT_B to the local TIMs. When you copy, the four connections of SINAUT_A are also, it; these are, however, incomplete. You can only see that the connections are incomplete when you have saved and compiled in NetPro.

Note

If you want to generate the S7 connections using the SINAUT ST7 configuration software, delete the incomplete connections in NetPro first.

1. Select the *Application* with the name *SINAUT_B* in the SINAUT PC.

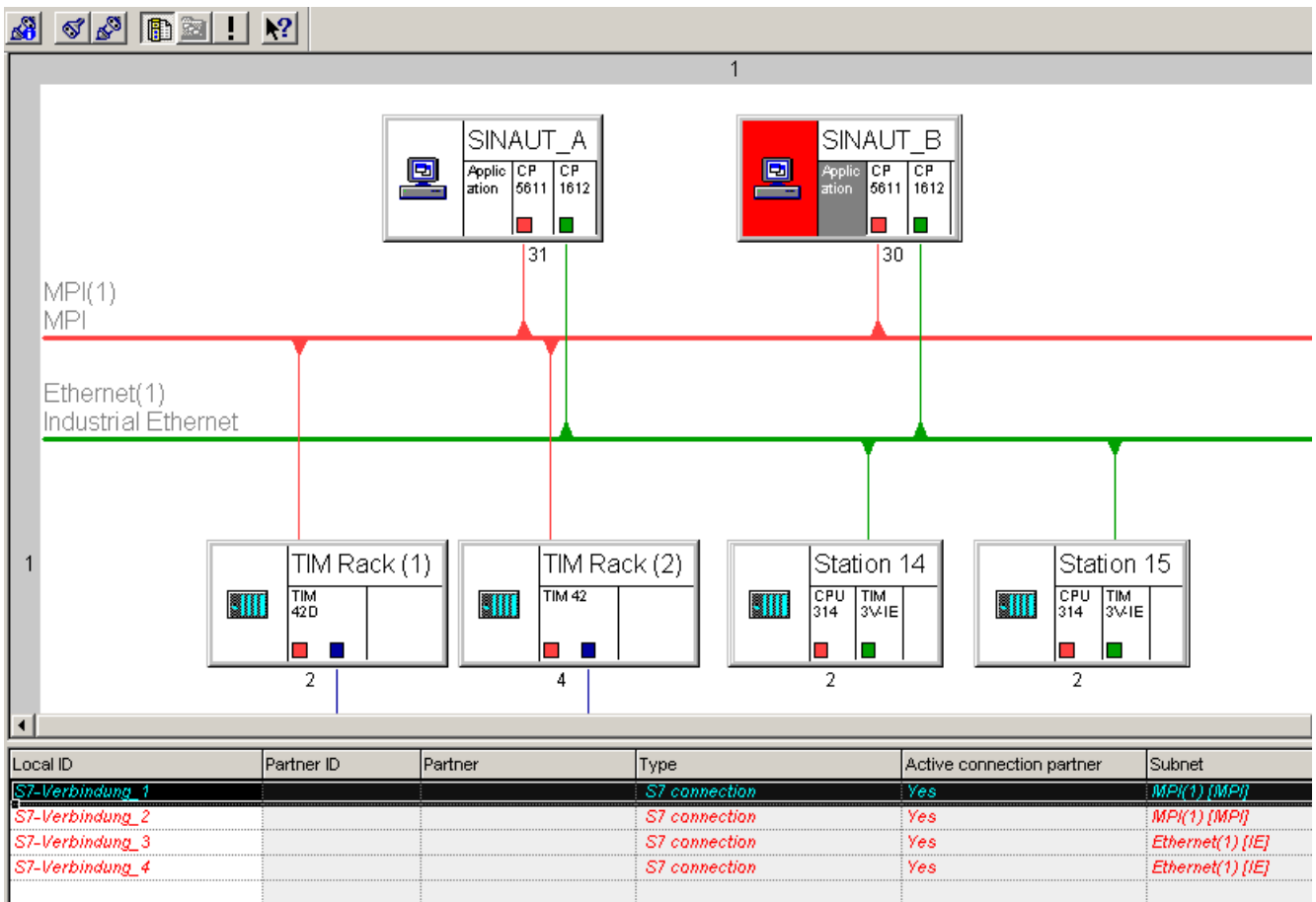


Figure 2-16 NetPro – copied, incomplete connections (red text)

In the lower part of the NetPro dialog, you now see the connection table (see figure) with the incomplete connections (red text).

Completing the connections:

1. First click on the required connection in the table (the row is shown on a black background).
2. Right-click on this row.

Note

The selection window that can be opened with the right mouse button always relates to the connection row shown on a black background regardless of where you right-click within the connection table. You should therefore always left-click on the required connection first so that the row is highlighted with a black background. You cannot select a connection row with the right mouse button.

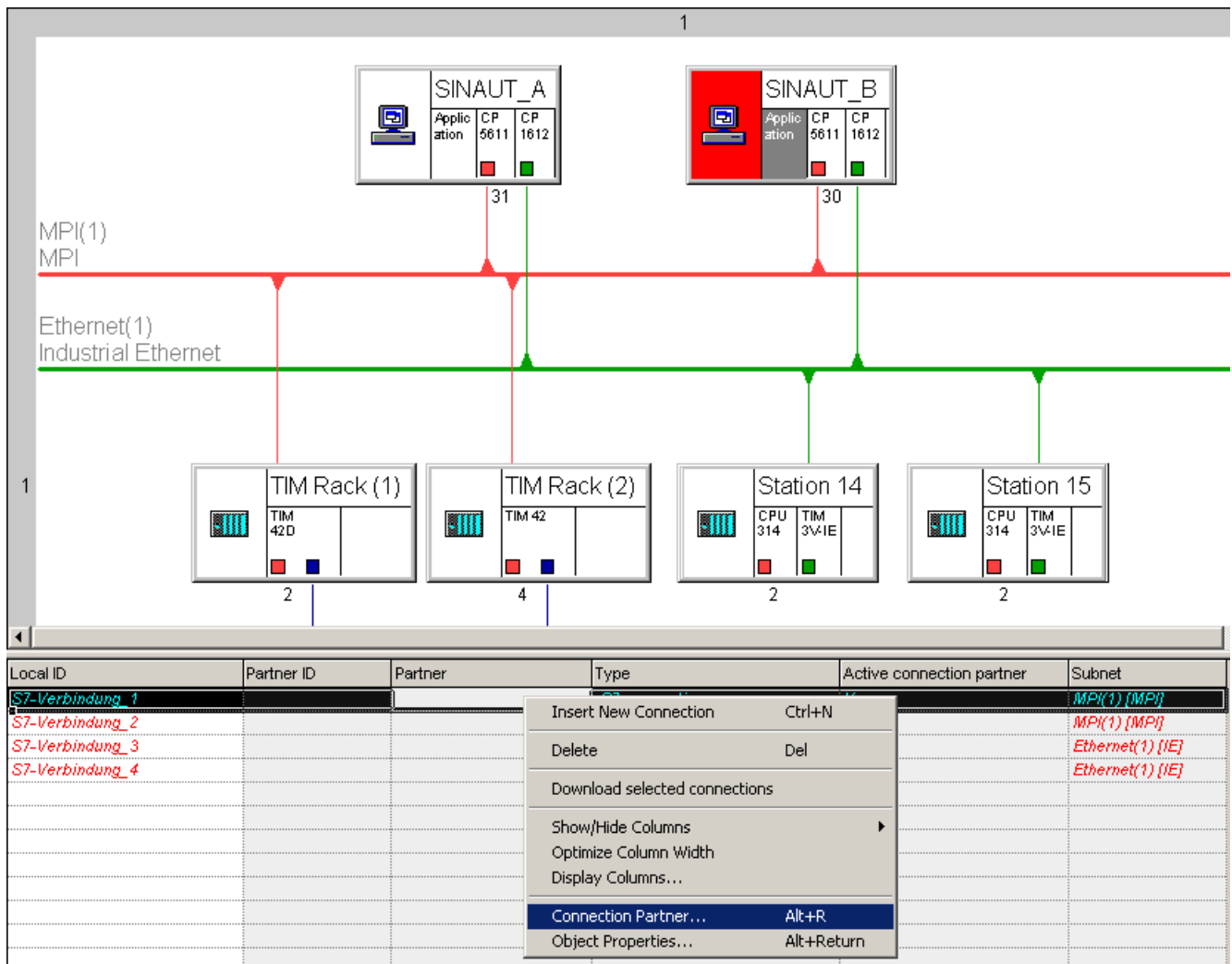


Figure 2-17 NetPro – selecting the Connection Partner... option

3. Select *Connection Partner...* in the menu that opens.
 The *Change Connection Partner* dialog opens.

4. Select the relevant partner in the *Connection Partner* list.

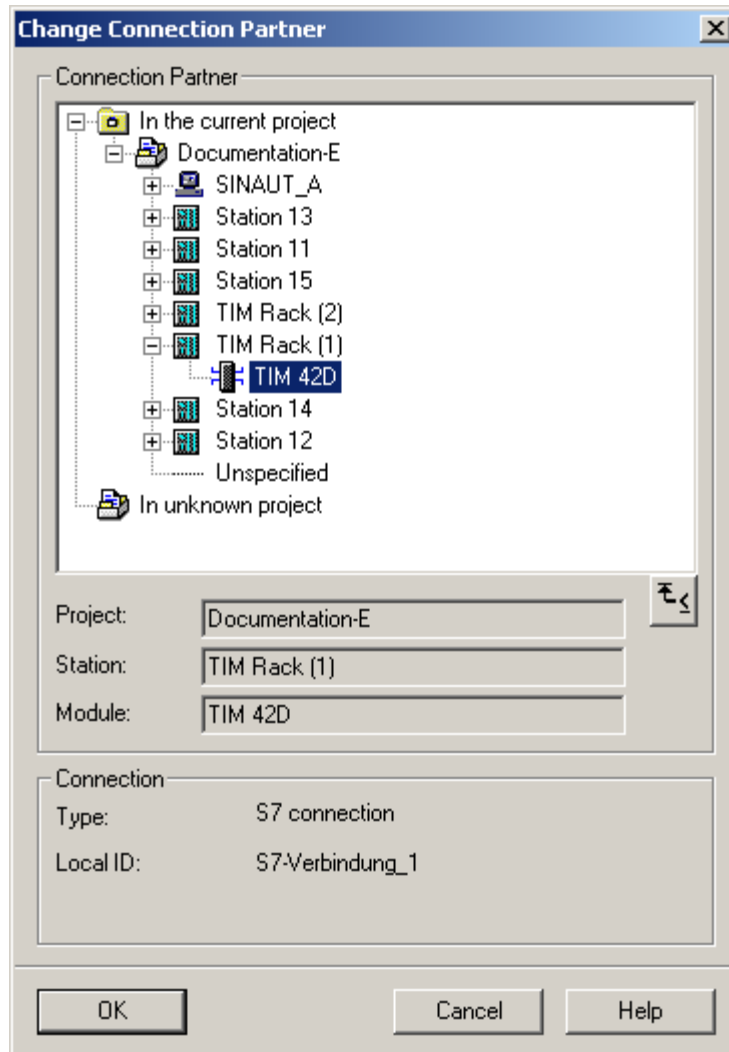


Figure 2-18 NetPro – Change Connection Partner dialog

You should select exactly the same S7 connections for the second SINAUT PC as for the first PC.

5. Select the TIM in TIM rack(1) as the partner for the first connection.
6. Close the dialog with *OK*.

In the connection table in the NetPro dialog, the first connection has now been completed with all the missing data.

7. Complete the other connections using the same steps as described above.

The integration of the second, redundant SINAUT PC and creation of the connections from this PC to the local TIMs is now completed.

2.4.4 Time service on the MPI and Ethernet bus

Time-of-day synchronization of the ST7cc PC

The time of day of the ST7cc PC can be synchronized as follows:

- **Internally on the PC (using the PC's own clock)**

The ST7cc PC uses the time of its own PC clock but should then be synchronized by a DCF77/GPS timer. Time-of-day synchronization is via the Ethernet bus.

Redundant ST7cc:

In a redundant ST7cc system where the PC's own time is used, there must be synchronization using DCF77 or GPS to ensure that both PCs are synchronized..

- **Time-of-day synchronization by a local TIM**

The ST7cc PC uses the time of a local TIM with DCF77 via the MPI bus.

Which of these options is used is specified by the configuration engineer with the ST7cc configuration tool. You will find the necessary configuration in the section Project settings: Server (Page 118) in the Options box.

Time-of-day synchronization of the ST7cc PC by local TIM with DCF77/GPS

Local TIMs connected via MPI are not synchronized by the ST7cc PC but by a local TIM with a DCF77 receiver connected to the MPI bus.

Requirement:

- One of the TIMs is equipped with a DCF77 receiver. This TIM then adopts the function of time master on the MPI bus and synchronizes the other TIMs connected to the MPI bus. We recommend that you also synchronize the ST7cc PC.

Redundant system

In redundant systems, both ST7cc systems are synchronized. If additional SINAUT stations are connected over Ethernet, the synchronized ST7cc PC can synchronize the time for these Ethernet stations.

To configure time synchronization on the MPI bus, follow the steps outlined below:

1. Open the relevant project in STEP 7.
2. Select the relevant local TIM with the DCF77 receiver.
3. Select Object Properties... in the menu that opens.

4. Select the Time Service tab in the Properties dialog.

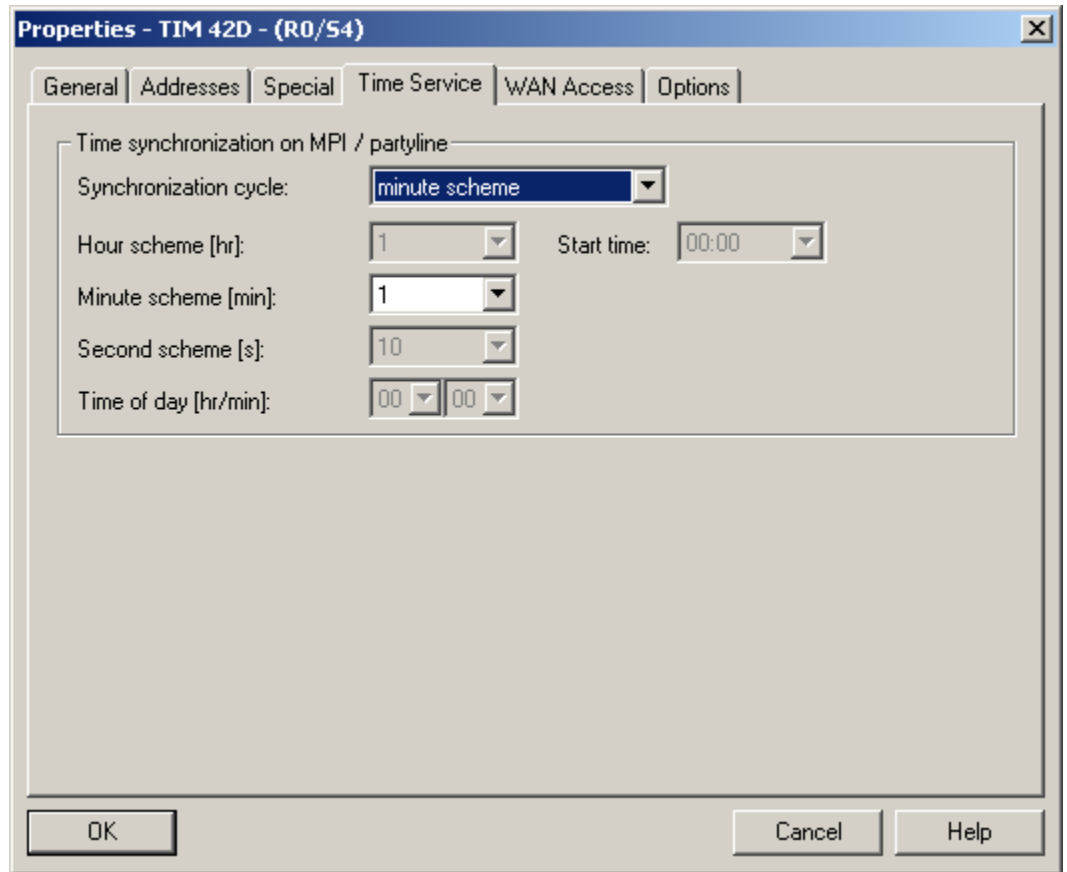


Figure 2-19 Setting time synchronization on MPI / partyline

5. Set the Synchronization cycle parameter to "minute scheme".
6. In the Minute scheme (min) list box, set the option 1.
7. Close the dialog with OK.
8. If further local TIMs are connected to the MPI bus, make the same settings as in the figure for these TIMs.
If the time master TIM fails, one of these TIMs automatically takes over the time master function. If this happens to be a TIM without a DCF77 receiver, it must have already been synchronized once by the TIM with the DCF77 receiver so that it has a correct time of day.

Note

The setting for time synchronization on the MPI has no influence on the time synchronization of the SINAUT stations in the WAN. Synchronization in the WAN is configured separately in the Properties dialog of the WAN, Time Service tab.

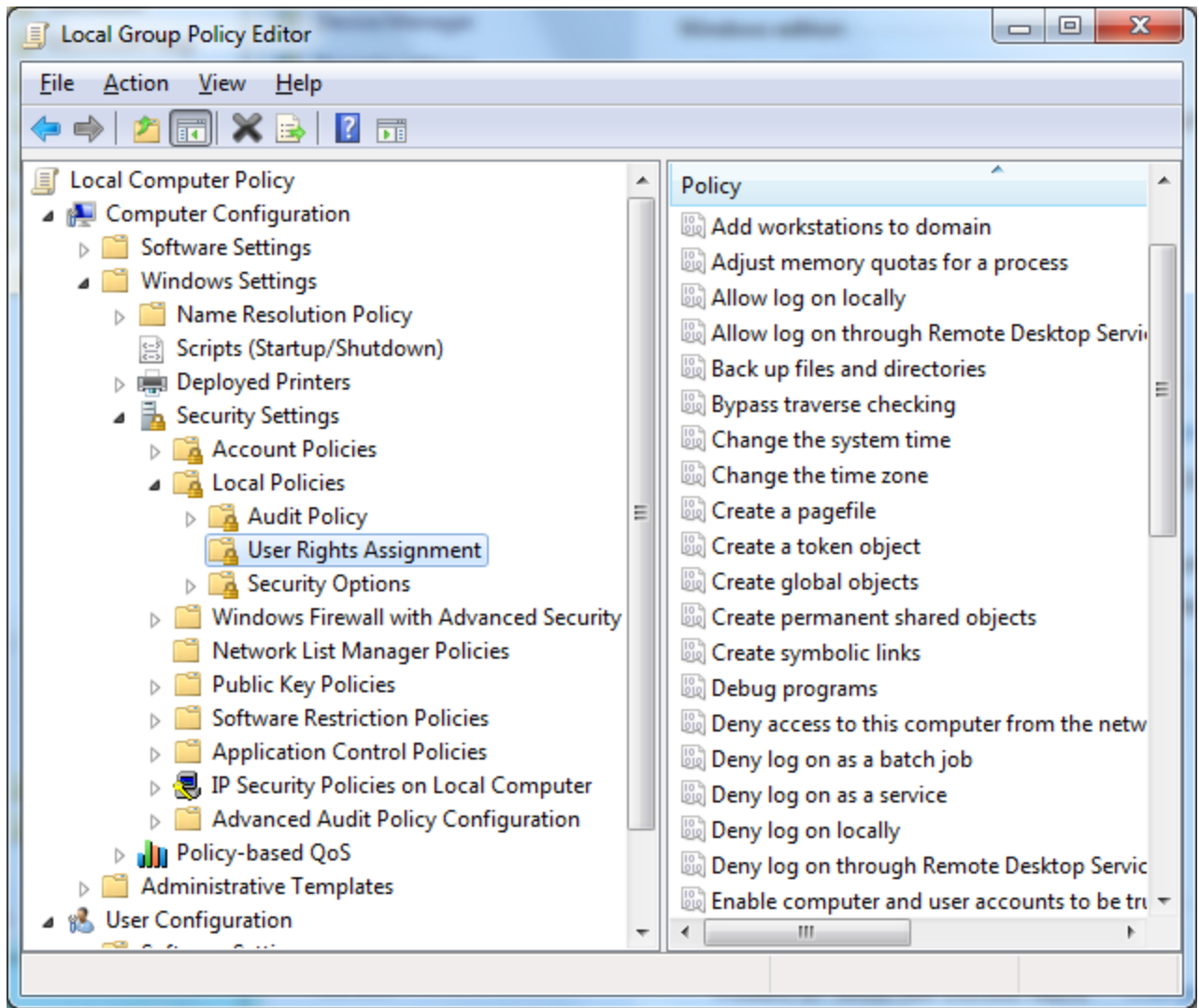
User settings for time-of-day synchronization of the ST7cc PC

The following information applies only to the following operating systems:

- Windows 7
- Windows Server 2008

If ST7cc is started by a user and not by the administrator, time-of-day synchronization of the ST7cc PC by a connected TIM works only with additional settings. The ST7cc application needs the right to change the time-of-day in the Local Group Policy editor. Follow the steps outlined below:

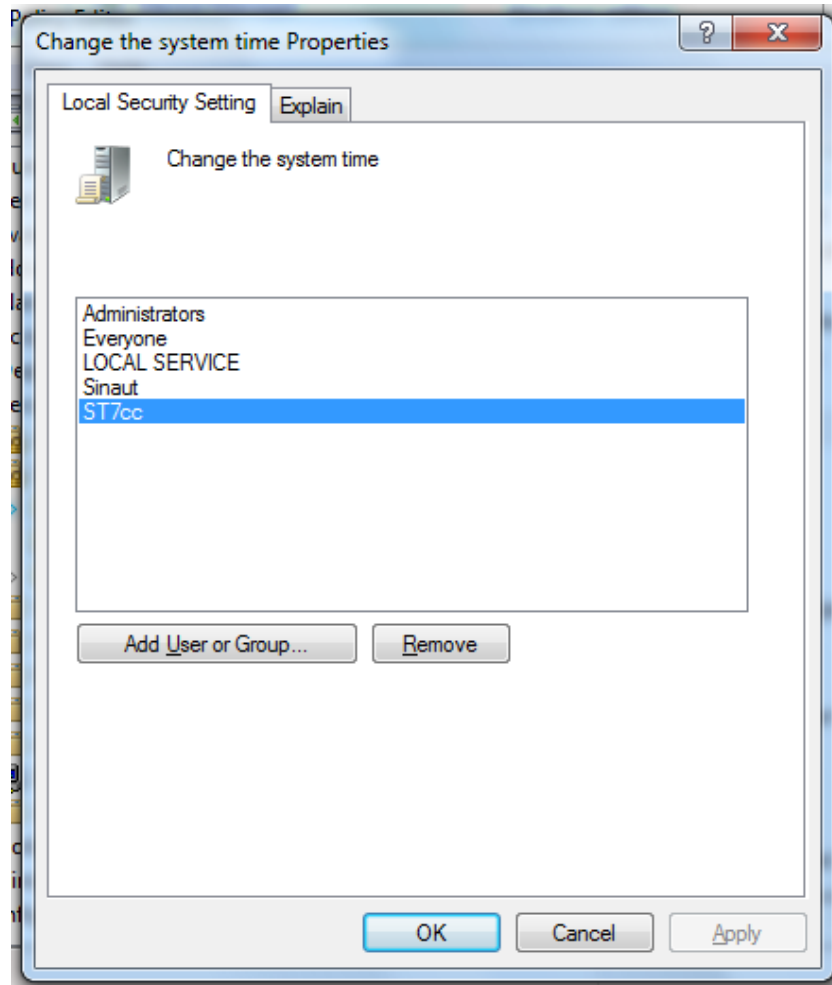
1. Open the local group policy editor with "Start > search line entry".
The input box opens.
2. Enter the text "gpedit.msc" in the input box.
The local group policy editor opens.



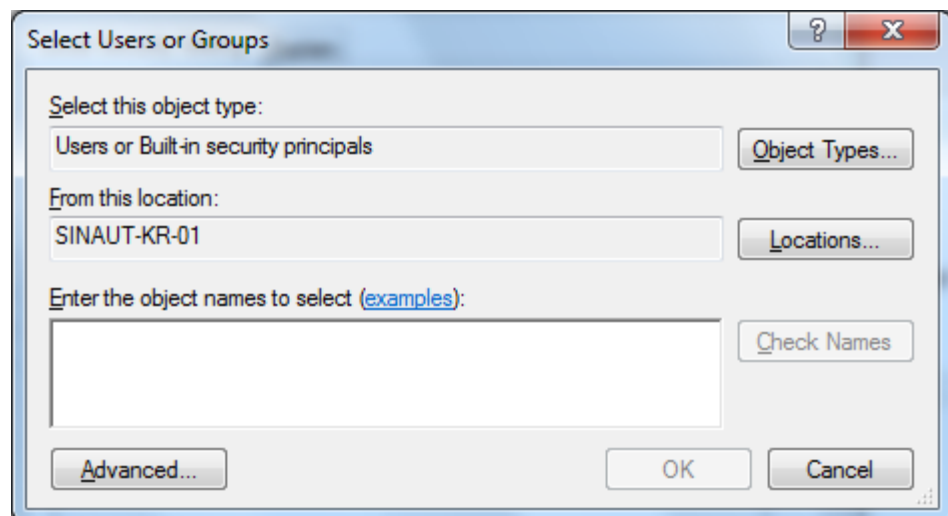
3. Open the navigation as shown:
Computer Configuration > Windows Settings > Security Settings > Local Policies

4. Click on "User Rights Assignment".

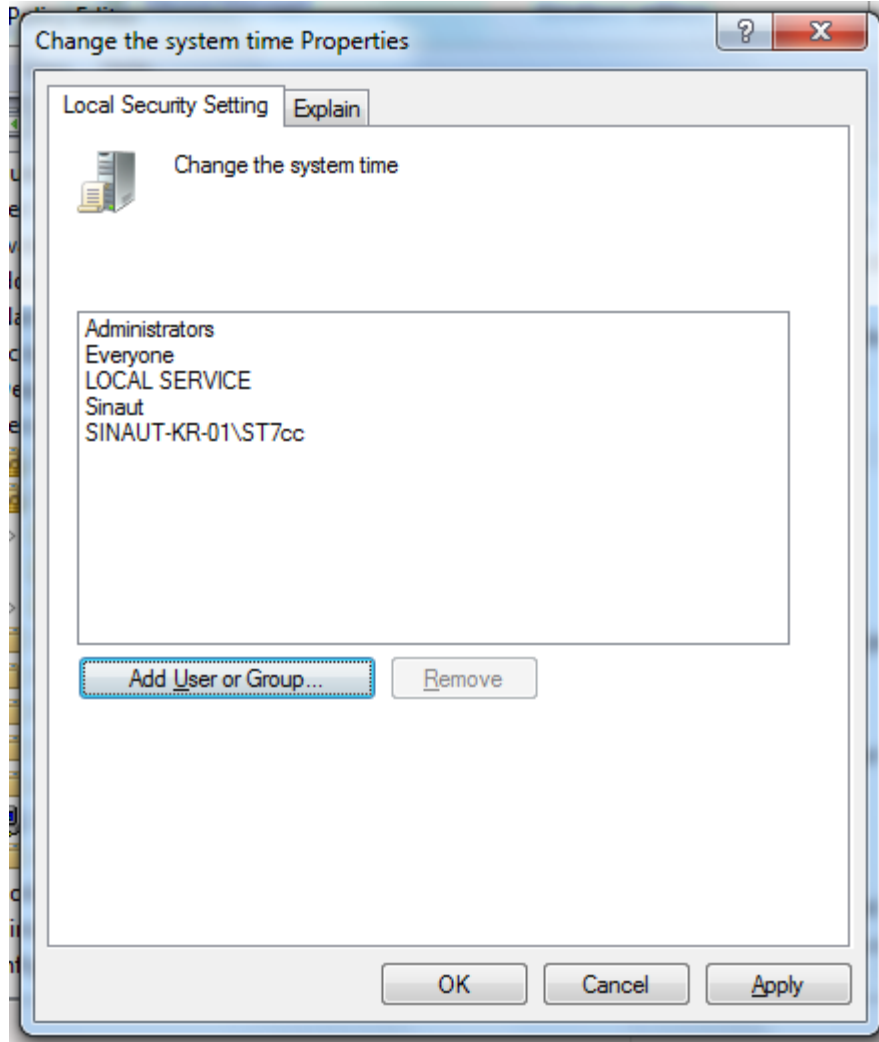
The "Change the system time Properties" dialog opens.



5. Click on "Add User or Group...".



6. Enter your Windows user name in the input box (in the example "ST7cc") and click OK. ST7cc is adopted in the "Change the system time Properties" dialog.



7. Click "Apply".
ST7cc now has the right to set the PC time.

Time-of-day synchronization of the TIM modules on the Ethernet bus

The Ethernet TIMs without a DCF77 receiver receive the current time from the ST7cc PC. The time of the local TIMs on the Ethernet bus is synchronized by the TIMs querying the current time from the ST7cc PC in the configured synchronization cycle (see figure).

Time-of-day synchronization on the Ethernet bus

To configure time synchronization of the local TIMs on Ethernet, follow the steps outlined below:

1. Open the relevant project in STEP 7.
2. Right-click on the local Ethernet TIM for which you want to set the time parameters.

3. Select Object Properties... in the menu that opens.
4. Select the Time Service tab in the Properties dialog.
5. Select the relevant Ethernet interface.

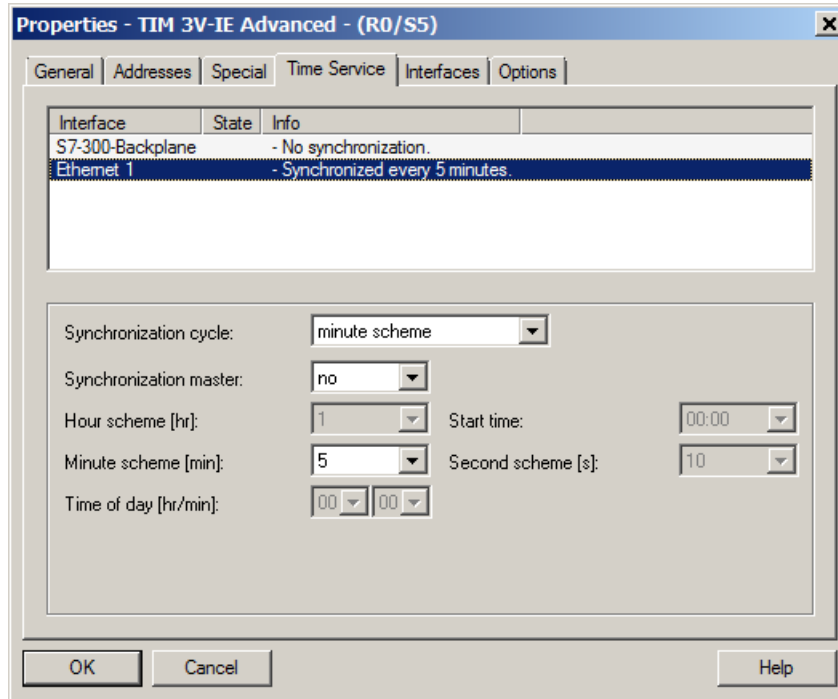


Figure 2-20 Setting time synchronization of a local TIM on the Ethernet bus

6. Set the Synchronization cycle parameter to "minute scheme". In the Minute scheme (min) list box, set the option 1 (see figure). This specifies the cycle in which the TIM queries the current time from the ST7cc PC over Ethernet.
7. Make sure that the Synchronization master parameter in the Time synchronization on Ethernet area is set to no.
8. If further local TIMs are connected to the Ethernet bus, make the same settings as in the figure for these TIMs.
9. Close the dialog with OK.

2.4.5 Saving the configuration data in NetPro

This completes the configuration work in NetPro.

1. Save the configured data by clicking on the Save and Compile button in the NetPro toolbar (see figure).

The Save and Compile dialog opens (see figure).

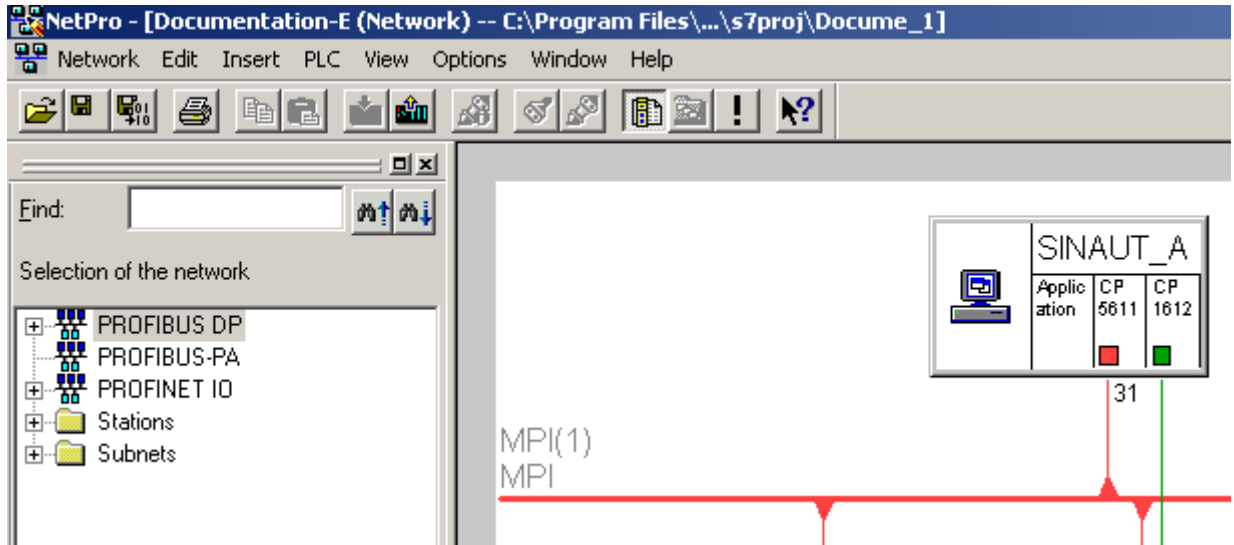


Figure 2-21 Saving configuration data in NetPro

2. Select the Compile changes only option.
3. Close the dialog with OK.

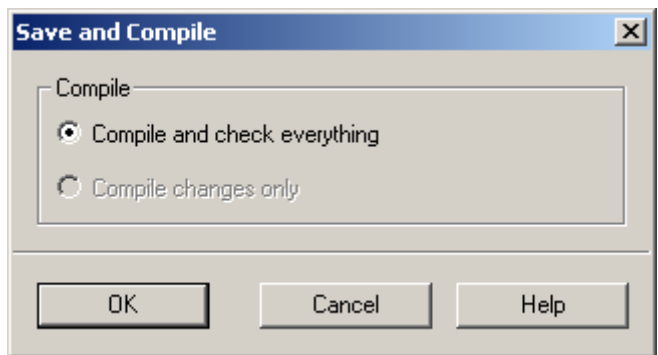


Figure 2-22 Saving and compiling

Once this is completed, the Outputs for consistency check dialog is displayed. If this does not indicate any errors, you can close the dialog. Otherwise, check your configuration and restart Save and Compile.

The fully configured data can now be downloaded to the SINAUT PC (if you have a redundant ST7cc, to both). Section Configuring the SINAUT PC (Page 73) explains how to do this.

2.5 Configuring the SINAUT PC

The following steps in section Task (Page 41) have been performed up to now:

- Integrating the SINAUT PC in the STEP 7 project with the SINAUT stations.
 - Inserting the PC station
 - Specifying the object properties of the PC station
 - Specifying the hardware configuration of the PC station; in other words, slots for the application (ST7) and communications modules
 - Configuring S7 connections between ST7cc and the local TIMs
 - Including a redundant SINAUT PC (optional)
 - Specifying time of day, including a redundant SINAUT PC (optional)

Result: The configuration data for the PC station is available in the XDB file.

Configuration Console

This section describes how to commission your SINAUT PC as part of an industrial communication network.

With Advanced PC Configuration, the SIMATIC NET PC Software supports the option of configuring both programmable controllers and PC stations from a central engineering station (ES). The engineering station is a networked PC with the SIMATIC NCM PC program or STEP 7. For a detailed description of the various options, refer to 'SIMATIC NET > Commissioning PC Stations – Manual and Quick Start' and in the SIMATIC NET Configuration Console.

Start the SIMATIC NET Configuration Console from the start menu (Start > Simatic > SIMATIC NET > Configuration Console).

The Configuration Console dialog opens:

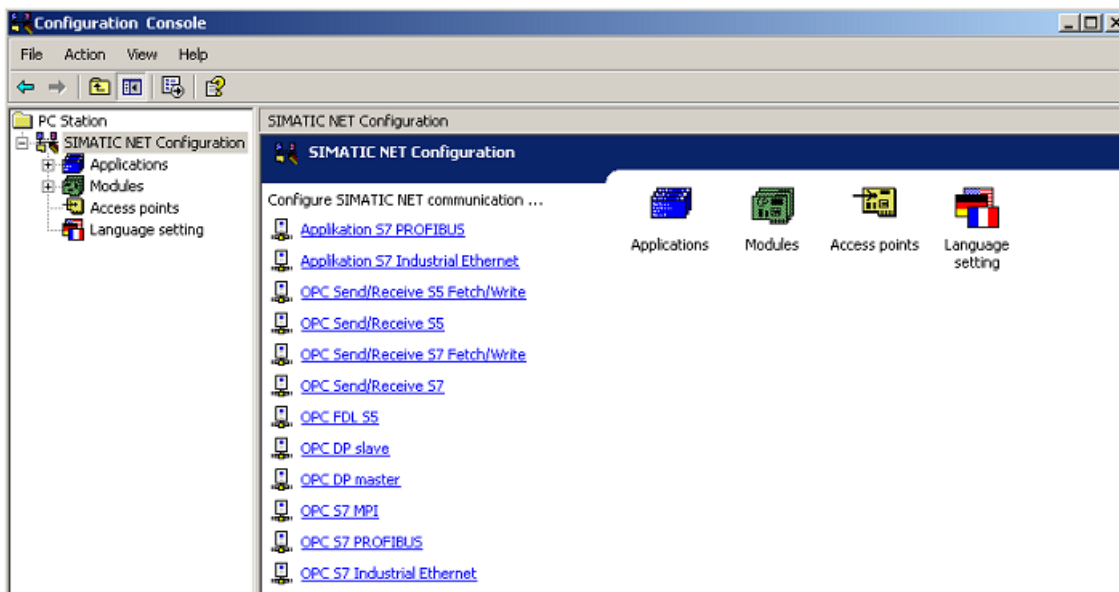


Figure 2-23 SIMATIC NET Configuration Console

If you select the Application S7 PROFIBUS or Application S7 Industrial Ethernet option, you will see instructions on how to set up your PC station. This description therefore only outlines the basic steps. SIMATIC NET Configuration Console

2.5.1 Initial configuration

Tools for initial configuration

For the initial configuration, use one of the following tools depending on the procedure:

- Station Configuration Editor
- STEP 7/ NCM PC

Why do we need an initial configuration?

When a module (CP) is put into operation for the first time, an initial configuration is necessary. The initial configuration is required for all newly installed modules.

Following the initial configuration, the PC station is prepared to receive configuration data.

Result of the initial configuration

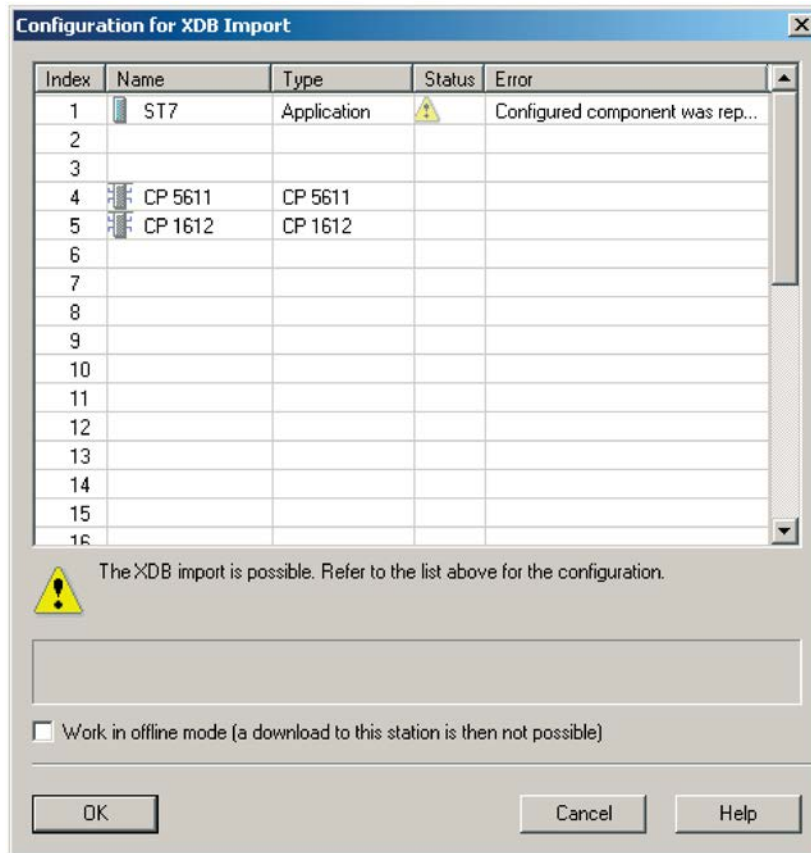


Figure 2-24 Station Configuration Editor

Note

For productive communication between the applications of the PC station (for example ST7cc) and the local TIMs, select the "Configured mode" option.

After starting the PC station, the PC module is in the "PG operation" mode.

By adding the communications module in the *Station Configuration Editor*, the module is automatically switched to "configured mode" and the index (the virtual slot number) of the module is set.

Options of initial configuration

For SINAUT ST7 installations, there are two possible procedures for initial configuration. These are:

- Initial configuration with an XDB file
- Initial configuration by remote configuration with STEP 7 / NCM PC

With these two procedures, it is assumed that the PC station has been configured with its components and applications in STEP 7 as described in the section Linking the SINAUT PC in the STEP 7 project (Page 43). This produces a database that is available to the user for initial configuration of the PC station.

The advantage of this is that the consistency between the configuration data and the PC configuration is guaranteed and the overall effort required is minimal. This procedure also resembles the general working procedures when configuring SINAUT installations.

Initial configuration with an XDB file

When you save and compile your STEP 7 project, an XDB file is created for each PC station. Initial configuration with an XDB file means that you fill out the configuration list of the Station Configuration Editor by importing the XDB file. Using this function, it is also possible to load the engineering and configuration even without a network connection to the target PC station. Follow the steps outlined below:

1. Start the Station Configuration Editor from the start menu (*Start > Station Configuration Data*)

The *Station Configuration Editor* dialog opens.

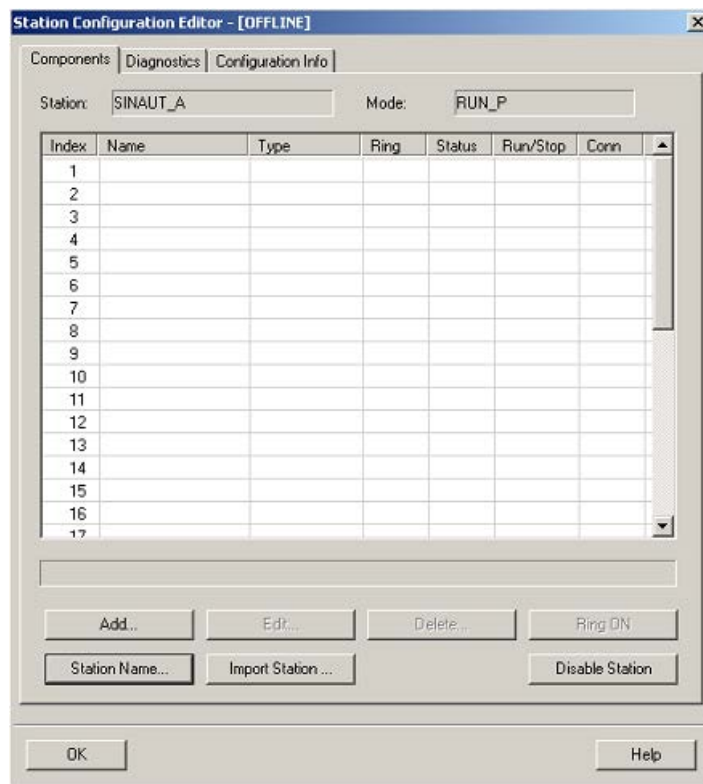


Figure 2-25 Station Configuration Editor

2. In the *Station* output box, check that the name you assigned to your SINAUT PC when creating the PC station in HW Config is displayed (in the example, the station name *SINAUT_A* or *SINAUT_B*).

3. If you need to change the station name, click on the *Station Name* button. The *Station Name* dialog opens in which you can enter the station name of your SINAUT PC. Confirm your entry with OK.
4. Click on the Import *Station* button to start importing the XDB data.
You will see a warning that the station must be restarted following import. Confirm this message with *Yes*. The dialog for selecting the XDB file then opens.
5. Enter the path on which the XDB file is located. This file is created in the STEP 7 project in the *XDBs* folder. If this project is not on your SINAUT PC, transfer a copy of the XDB file with a diskette or USB stick. Click the Open button to start the data import.

As a result, you will once again see which modules and applications are configured in the XDB, see figure.

Note

In the example, two PC stations (SINAUT_A and SINAUT_B) were configured. This is why the figure shows two XDB files in the XDBs folder: pcst_1.xdb for the PC station SINAUT_A and pcst_2.xdb for SINAUT_B.

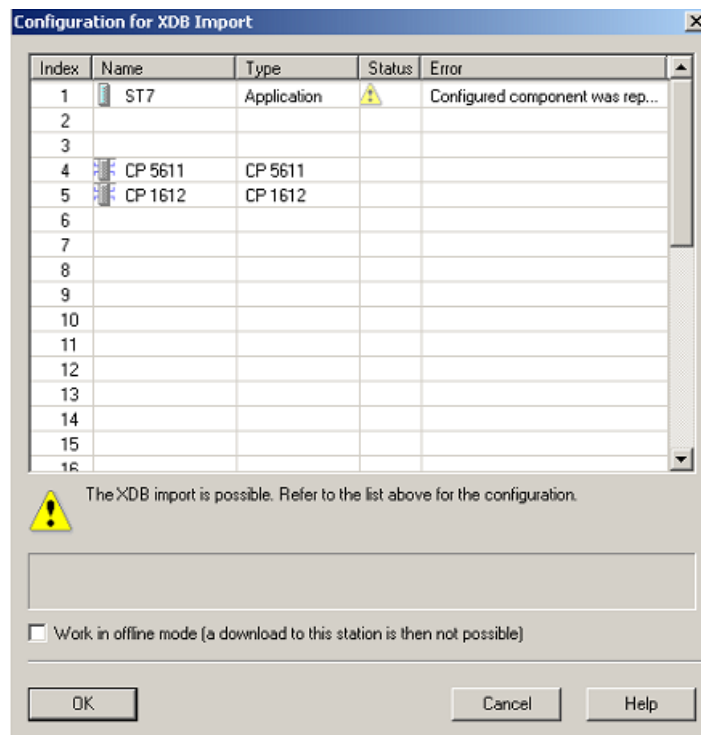


Figure 2-26 Station Configuration Editor

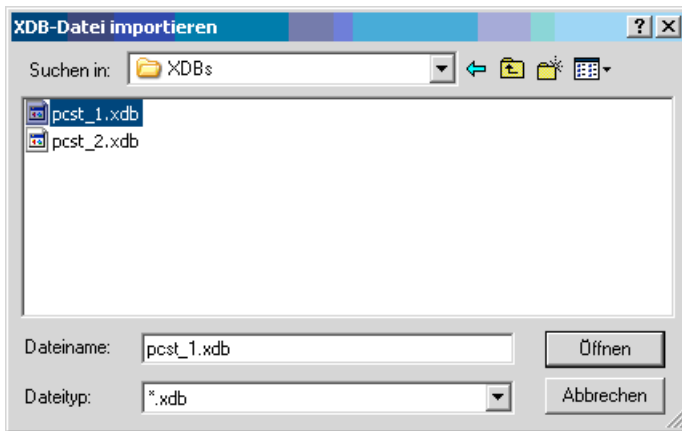


Figure 2-27 Browser window for XDB file import

1. If you want to prevent configuration data being transferred online at a later point in time, select the *Work in offline mode* option. The default setting allows configuration data to be transferred online.
2. Close the import of configuration data with OK.

Note

Importing is possible only when the imported configuration matches the existing local configuration.

To complete configuration of your SINAUT PC in SIMATIC NET, you still need to configure the access points, see section Setting access points for the SINAUT PC (Page 81).

Initial configuration by remote configuration with STEP 7 / NCM PC

Note

The description of the steps below assumes that you have installed the required version of the SIMATIC NET PC Software, see section Required software (Page 18).

In this procedure, it is assumed that you have configured your SINAUT project on an engineering PC and want to download the configuration data you created with SIMATIC NetPro to the SINAUT PCs SINAUT_A or SINAUT_B. the engineering PC is integrated as a PC station in your SINAUT project. The configuration data for the SINAUT PCs is stored on the engineering PC as XDB files for the SINAUT PCs in the STEP 7 project.

1. Connect the engineering PC to the Ethernet bus to which the SINAUT PCs you want to download to are connected.
2. Install the SIMATIC NET PC software products on the SINAUT PCs as described in the section Installation of the SIMATIC NET PC software products (Page 42).
3. In SIMATIC NetPro, click on the SINAUT PC to which you want to download the configuration data so that the name of the PC is highlighted on a blue background (see figure).

- Click on the Download selected station(s) button in the toolbar (see figure).

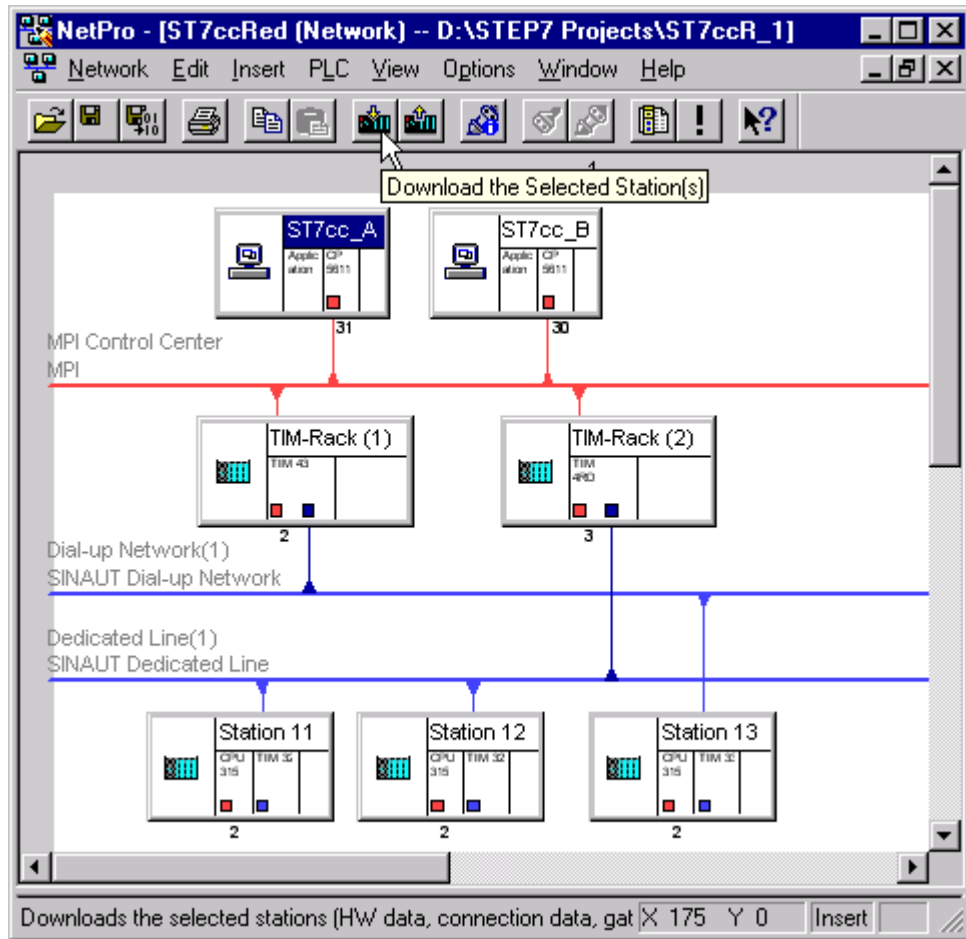


Figure 2-28 Downloading configuration data to the PC station SINAUT_A

Before the download is actually started, you will be prompted for confirmation (see figure).

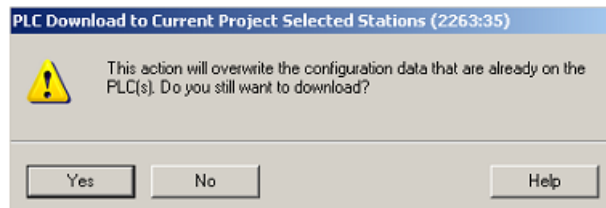


Figure 2-29 Prompt for confirmation before starting the download

- 5. Click on *Yes* to close the prompt.

The download then starts.

After some time, the Stop Target Modules dialog opens (see figure). This informs you that the CP to which you are downloading must be stopped.

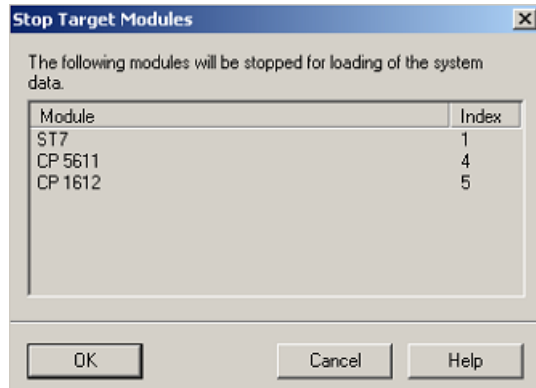


Figure 2-30 Dialog indicating the CP to be stopped for the download

- 6. Close the dialog with *OK*.

The download continues. It is completed when the Download progress bar closes automatically.

ST7cc is now capable of exchanging data with the local TIMs over the MPI or Ethernet bus. The data received by these TIMs from SINAUT stations can be forwarded to the PC. In the other direction, the PC can transfer data intended for the SINAUT stations to the local TIM responsible that then transmits the data to the destination station.

Repeating the download of configuration data

It is only necessary to download the configuration data (XDB) again if the following changes are made on the central MPI or Ethernet bus:

New local TIMs are added

Previously configured local TIMs are removed

The MPI or IP address of a TIM of the SINAUT PC is modified

Note

The data in the XDB file for the SINAUT PC that is specific to SINAUT relates only to the connection to the local TIMs; in other words, to the TIMs connected locally over MPI with the SINAUT PC and to the TIMs in the SINAUT stations connected directly to ST7cc over Ethernet. The XDB file contains no data relating to the connection to the SINAUT stations in the WAN.

For this reason: When the new stations are added or removed in the WAN, the XDB is unaffected.

Note on redundant ST7cc

With a redundant ST7cc, repeat the download for the second PC.

2.5.2 Setting access points for the SINAUT PC

Once the initial configuration is completed, the last step is to supply the access points with the correct interface parameter assignment. Follow the steps outlined below:

1. Start the SIMATIC NET Configuration Console from the start menu (*Start > Simatic > SIMATIC NET > Configuration Console*).

The *Configuration Console* dialog opens.

2. In the left-hand column under *PC station*, click on *Access points*.

The available access points are displayed in the right-hand column.

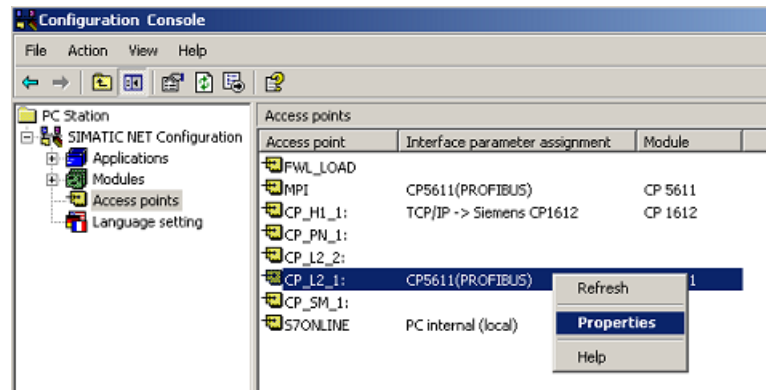


Figure 2-31 Selecting properties of the access point CP_L2_1

Access point for communication over the MPI bus

1. Right-click on the access point *CP_L2_1* if you want to communicate with a local TIM over a CP 5611 or CP 5613 via the MPI bus.
2. Select the Properties option in the menu that opens (see figure).
3. For the Associated interface parameter assignment option, set CP 5611(PROFIBUS) (see figure).
For a CP 5613, select CP5613(PROFIBUS).

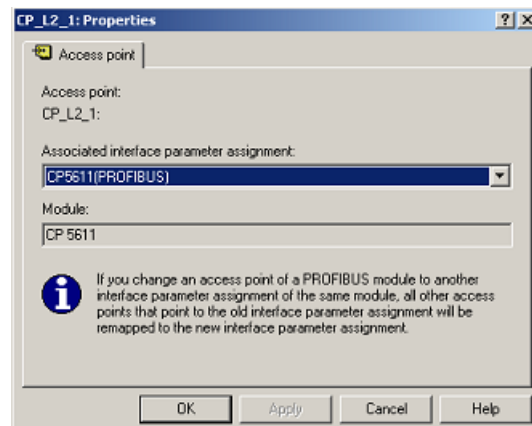


Figure 2-32 Setting the interface parameter assignment to CP561x(PROFIBUS)

Note

As default, ST7cc uses the access point *CP_L2_1* for communication with the TIMs on the MPI bus. As of ST7cc V2.5, a different access point can be selected if necessary.

4. Close the dialog with *OK*.

Note

Although the SINAUT PC is connected to the MPI bus over the CP 5611 (or 5613), you must nevertheless set the interface parameter assignment to PROFIBUS and not to MPI that is also available in the list.

The *Configuration Console* dialog now displays the interface parameter assignment you selected for access point *CP_L2_1*.

Access point for communication over the Ethernet bus

1. Right-click on the access point *CP_H1_1* if you want to communicate with a local TIM over a CP 1612 or CP 1613 via the Ethernet bus.
2. Select the Properties option in the menu that opens (see figure).
3. For the *Associated interface parameter assignment* option, set *TCP/IP -> Siemens CP1612* or *TCP/IP -> Siemens CP1613*.

S7ONLINE access point

The S7ONLINE access point is normally already set to the interface parameter assignment PC internal (local). (see figure)

- If the setting is different, change this as described above.

PG operation could be handled via this access point parallel to SINAUT communication over the *CP_L2_1* access point. PG operation is, of course, only possible if STEP 7 is installed on the SINAUT PC and if the SINAUT project is loaded. This allows not only the TIMs connected locally to MPI or other S7 nodes to be programmed or accessed for diagnostics. Since SINAUT also allows the PG routing function over the telecontrol network, the SINAUT PC can also access remote CPUs and TIMs to make program changes, read out the diagnostic buffer, etc.

Note

If the CP561x was configured as described, it is now in *configured mode*. It is not necessary to change the CP interface from *configured mode* to *PG operation*. *SINAUT communication* and *PG access* can be handled at the same time in configured mode.

Displaying the active bus nodes

The Configuration Console provides still further options. You can, for example, check whether the currently initialized MPI interface of the SINAUT PC was actually activated. It could then be accessed via the MPI bus, for example to download the configuration data to the SINAUT PC from a PG.

To run this check, follow the steps outlined below:

1. From the Modules directory, select the CP 5611 folder (see figure).
2. Select the *Bus Nodes* subfolder.

In the right-hand window, you will see the nodes active on the MPI bus.

If you activated your own MPI interface as described here, at least this MPI address is displayed as being active in the overview.

In the figure the MPI address 31 that was configured for the SINAUT PC is displayed as the active bus node. If the MPI interface to which the bus is connected is displayed and there are further MPI Notes active on it (for example your PG or perhaps the local TIMs), these MPI addresses are also indicated as being active.

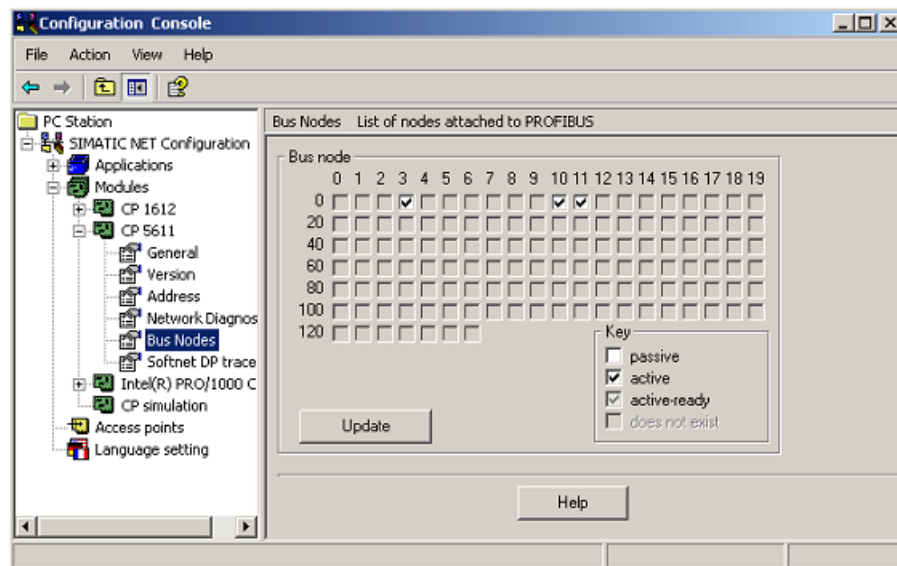


Figure 2-33 Display of the active bus nodes, for example MPI address 31

Note on redundant ST7cc

If you have a redundant ST7cc, you will need to perform the analogous steps on the second PC. If its configuration is identical to that of the first PC, you simply need to specify a different station name, for example *SINAUT_B* and a different MPI address, for example *30*. Everything else is configured exactly as described above.

2.6 SINAUT ST7 configuration tool

The previous sections describe the input necessary in SIMATIC NetPro and how the SINAUT PC itself is configured under the SIMATIC NET PC Software.

The SINAUT-specific data is then configured using the SINAUT ST7 configuration tool. From this data, the tool generates the system data blocks (SDBs) for the TIMs and CPUs. If TD7onCPU is used in some or all CPUs, the tool also prepares the records and communication data blocks for these CPUs and stores these along with several other blocks (FCs, FBs) required by the individual CPUs for SINAUT communication in the block containers of the CPUs.

2.6.1 Adapting SINAUT subscriber numbers

Opening the Subscriber Administration

1. Start the SINAUT ST7 configuration tool in the menu
Start > Simatic > SINAUT ST7 > Configuration
The still empty SINAUT configuration dialog is displayed.
2. Open your SINAUT project in the SINAUT configuration window.
The *SINAUT ST7: Configuration* window is displayed.
3. Select the Subscriber Administration option (see figure).

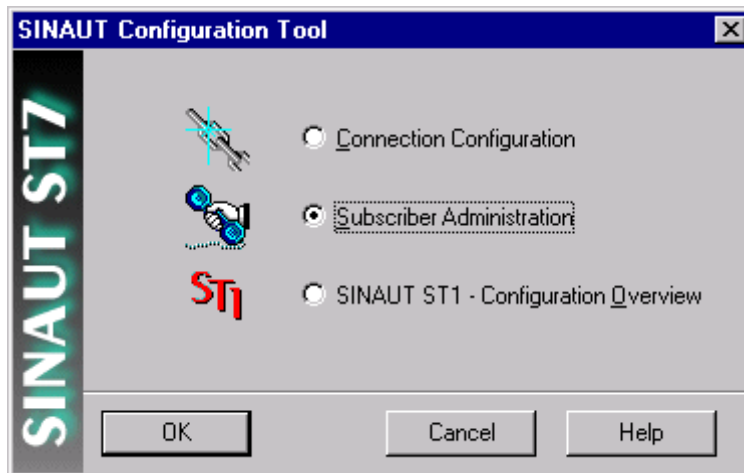


Figure 2-34 SINAUT configuration window – selecting subscriber administration

The right-hand part of the Subscriber administration window (see figure) shows you all the devices relevant for SINAUT. These are the CPUs, TIMs and SINAUT PCs.

The *Subscriber no.* column displays a SINAUT subscriber number for each SINAUT component. As default, the CPUs and the SINAUT PCs are assigned numbers starting at 1. For local TIMs, numbers starting at 1001 are assigned as default.

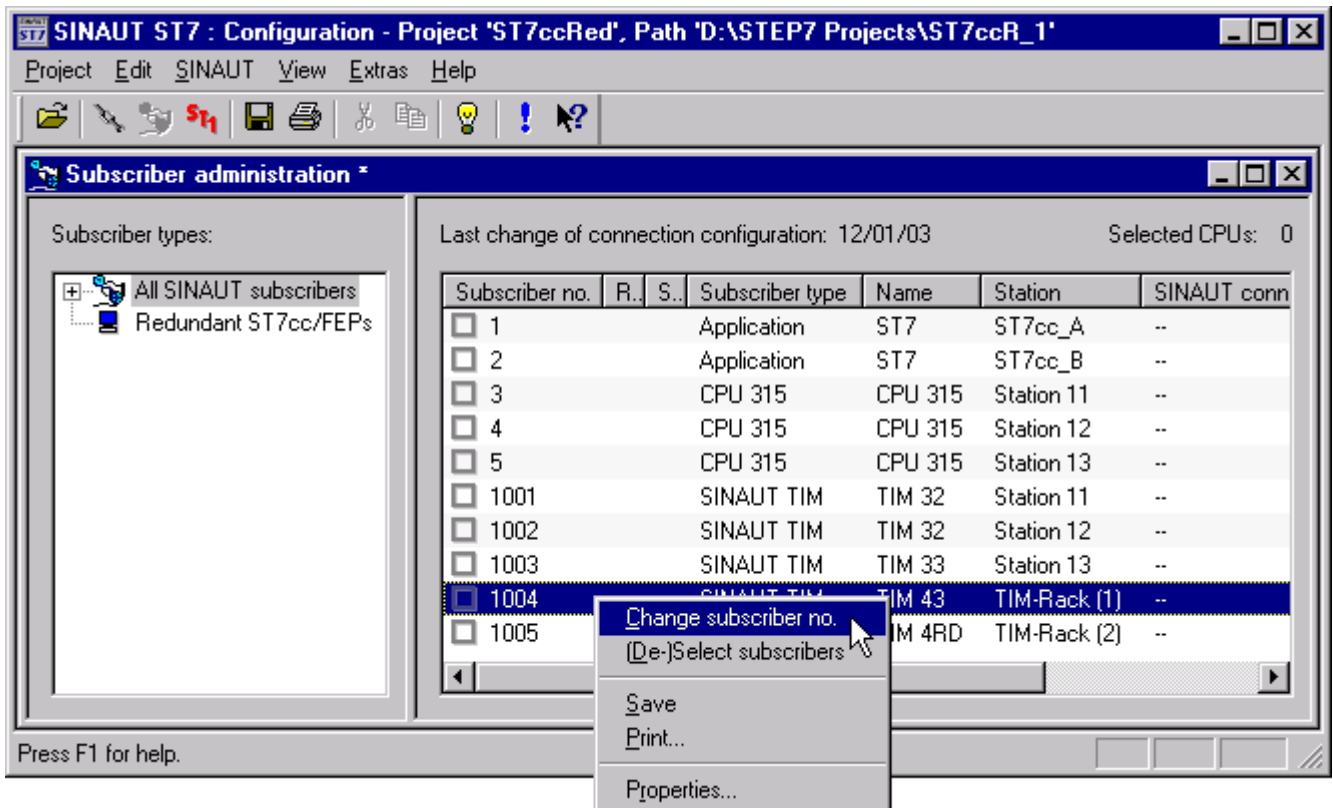


Figure 2-35 SINAUT ST7 subscriber administration, changing SINAUT subscriber numbers

4. Check these SINAUT subscriber numbers.

If you want to assign a different number to individual components, you can do this in the Subscriber administration. In the example, it would be practical to change the subscriber numbers 3, 4 and 5 for stations *Station 11*, *Station 12* and *Station 13* to the numbers 11, 12 and 13.

Changing a subscriber number:

1. Right-click on the relevant row.
2. Select the option Change subscriber no. in the open menu (see figure) and then enter the new subscriber number.

Recommendations for assigning SINAUT subscriber numbers

The background of this recommendation is the automatic message number assignment in ST7cc Config (see Section Message processing (Page 235)).

Note: 1 is reserved for WinCC.

- All **CPUs** should have subscriber numbers between **2 and 499**.
- The TIMs connected locally over MPI or Ethernet to ST7cc (**local TIMs**) should also have subscriber numbers between **2 and 499** (in the example, these are the two TIMs in TIM rack (1) and (2)).
- If you want to assign higher subscriber numbers for **CPUs and local TIMs**, numbers in the range **2 to 4095** are also permitted instead of 2 to 499. In this case, you will later need to change a default setting of ST7cc (see Section Project settings: Config (Page 142))
- For a **SINAUT PC**, you can specify any number in the range from **1 to 32000**. It is advisable to use a number outside the number range for CPUs and local TIMs, for example the number 1 if there is only one single SINAUT PC in the network.
- You can also specify any number in the range from **1 to 32000** for the TIMs in the stations (**station TIMs**).

the following figure shows the SINAUT components again. Some of these have now been given different subscriber numbers. The numbers for the local TIMs were, for example, changed to 101 and 102 so that they are within the range 2 to 499.

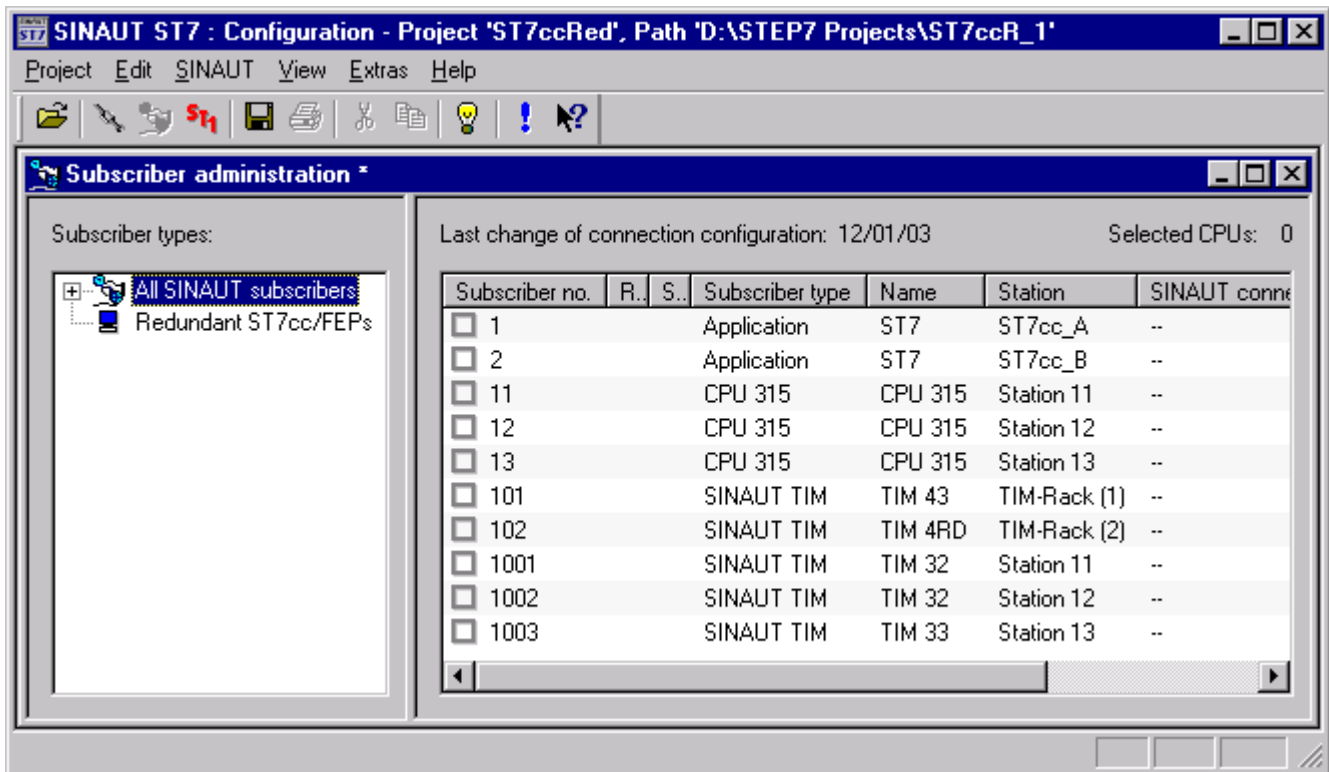


Figure 2-36 List of subscribers, some with changed subscriber numbers

If there is no redundant ST7cc in the project, continue as explained in the section Configuring SINAUT connections (Page 91). If, however, you do, continue as explained in Section Configuring redundant ST7cc (Page 87).

2.6.2 Configuring redundant ST7cc

Configuring redundant ST7cc

You can configure a redundant ST7cc in *Subscriber administration* by grouping two separate ST7cc subscribers together. You can then assign one of the two existing subscriber numbers to the grouped subscribers.

Note

The description of grouping two ST7cc systems to form **one** SINAUT subscriber is supported as of SINAUT configuration software SINAUT V3.1 or higher.

1. To specify which two SINAUT PCs represent a redundant pair, click on the row Redundant ST7cc/ST7sc server in the left-hand window (see figure).
If no redundant ST7cc pair has yet been specified, the right-hand window is initially empty.
2. Right-click on the *Redundant ST7cc/ST7sc Server* row.
3. In the context menu, select the *Add redundant ST7cc/ST7sc...* option.

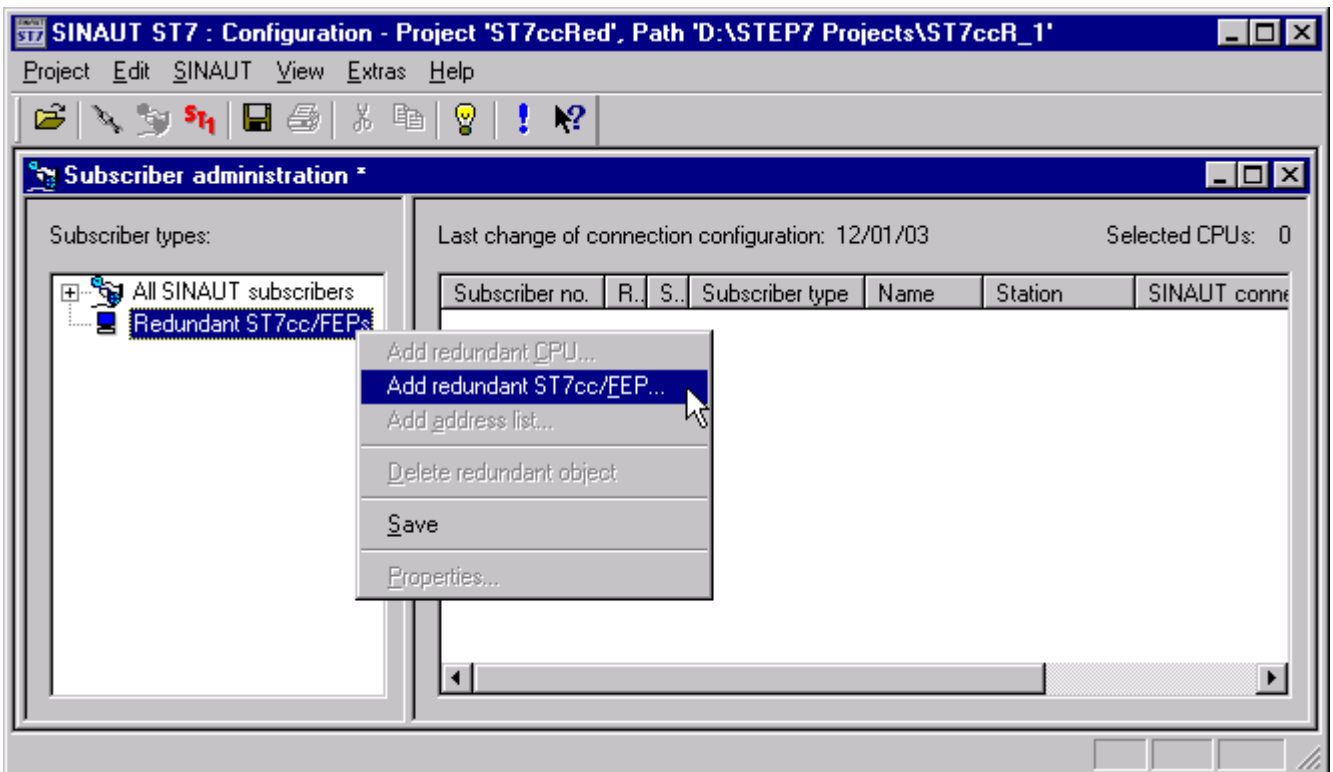


Figure 2-37 Selecting the Add redundant ST7cc/ST7sc... option

The Properties - ST7cc/ST7sc redundancy group dialog opens (see figure).

Here, you specify which SINAUT PCs belong to the redundant pair. You also specify which of the existing subscriber numbers will be used for the defined redundancy group.

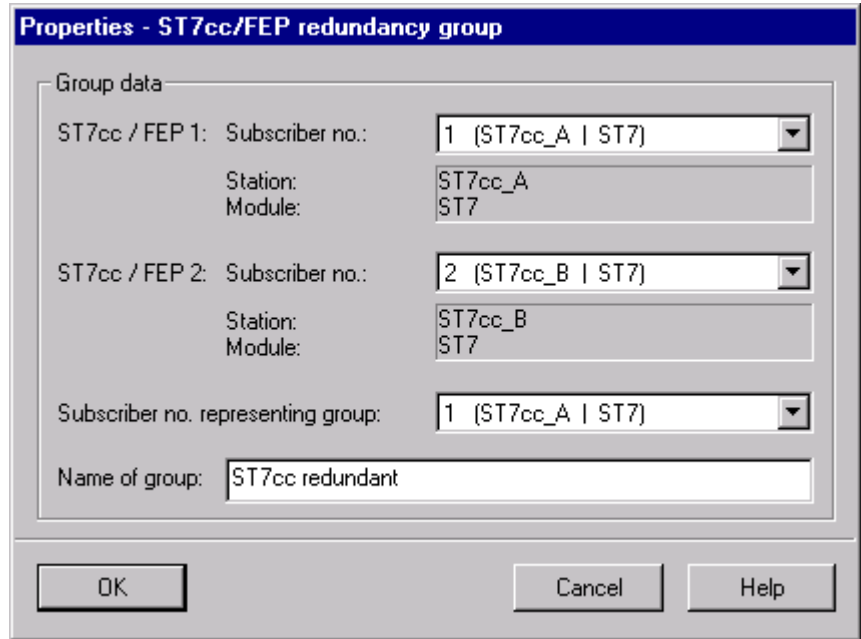


Figure 2-38 Properties dialog ST7cc/ST7sc redundancy group

4. Select the 1st subscriber (here *SINAUT_A*).
5. Select the 2nd subscriber (here *SINAUT_B*).
6. Change the default name for the redundant ST7cc to a name to suit your purposes (here *SINAUT-PC redundant*).

This name will then be entered in the left-hand window of *Subscriber administration*. You will also find the redundant ST7cc later under this name in *connection configuration*.

7. Close the dialog with *OK*.

In the right-hand window of subscriber administration, the two SINAUT PCs that make up the redundancy group are displayed.

In the column in which only *R...* can be seen in the figure, you will find the subscriber number specified for the redundant ST7cc (in the example 1).

In the column next to this, in which only *T...* can be seen in the figure, you will find the subscriber number of the redundant partner: In the example subscriber 2 belongs to subscriber 1 and vice versa (1 to 2).

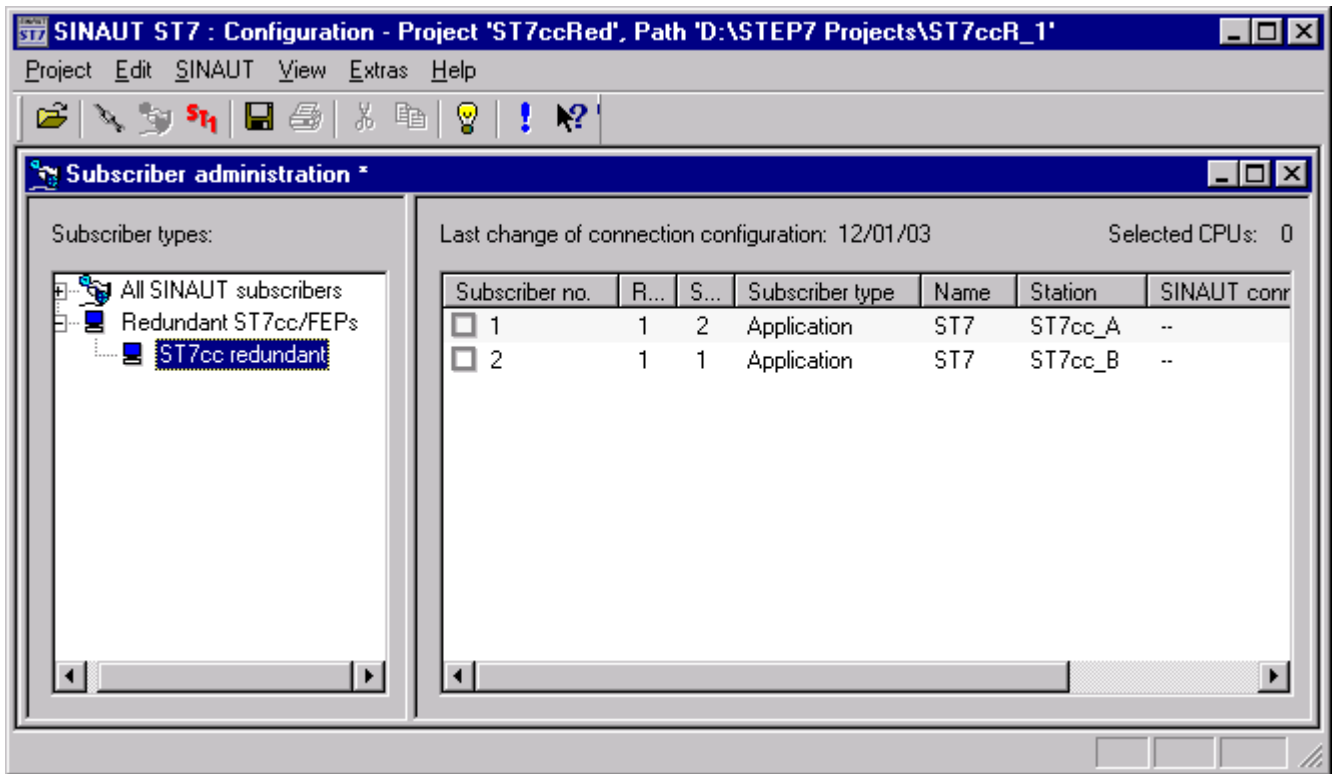


Figure 2-39 Display of the subscribers forming a redundancy group

8. Click on the *All SINAUT subscribers* row in the left-hand window.

The list of all subscribers is displayed again on the right-hand side (see figure).

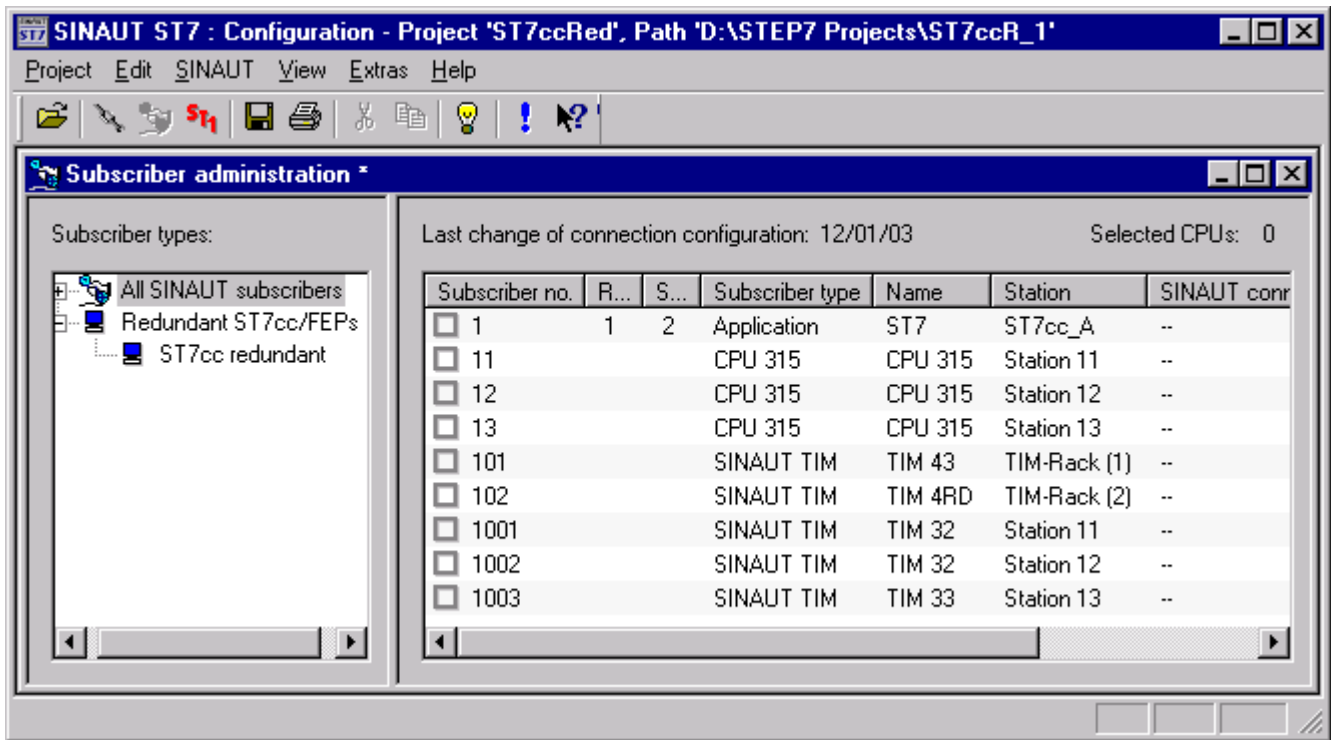


Figure 2-40 Overview of all SINAUT subscribers with a redundant ST7cc

The list displays only one of the original two SINAUT PCs as a SINAUT subscriber under the subscriber number you specified as the common subscriber number (in the example 1). In the two columns beside this, you can also see the common subscriber number (in the screenshot, the 1 in the *R... column*) and the subscriber number of the redundant partner (in the screenshot, the 2 in the *T... column*).

Both ST7cc systems can now be addressed under one single subscriber number, in the example, under number 1. Each message that a station sends to subscriber no. 1 is also transferred to the corresponding redundant subscriber (in the example to the SINAUT PC with subscriber number 2).

2.6.3 Configuring SINAUT connections

Changing from subscriber administration to connection configuration

To change from the SINAUT subscriber administration to the SINAUT connection configuration, click on the Connection configuration button in the toolbar (see figure).

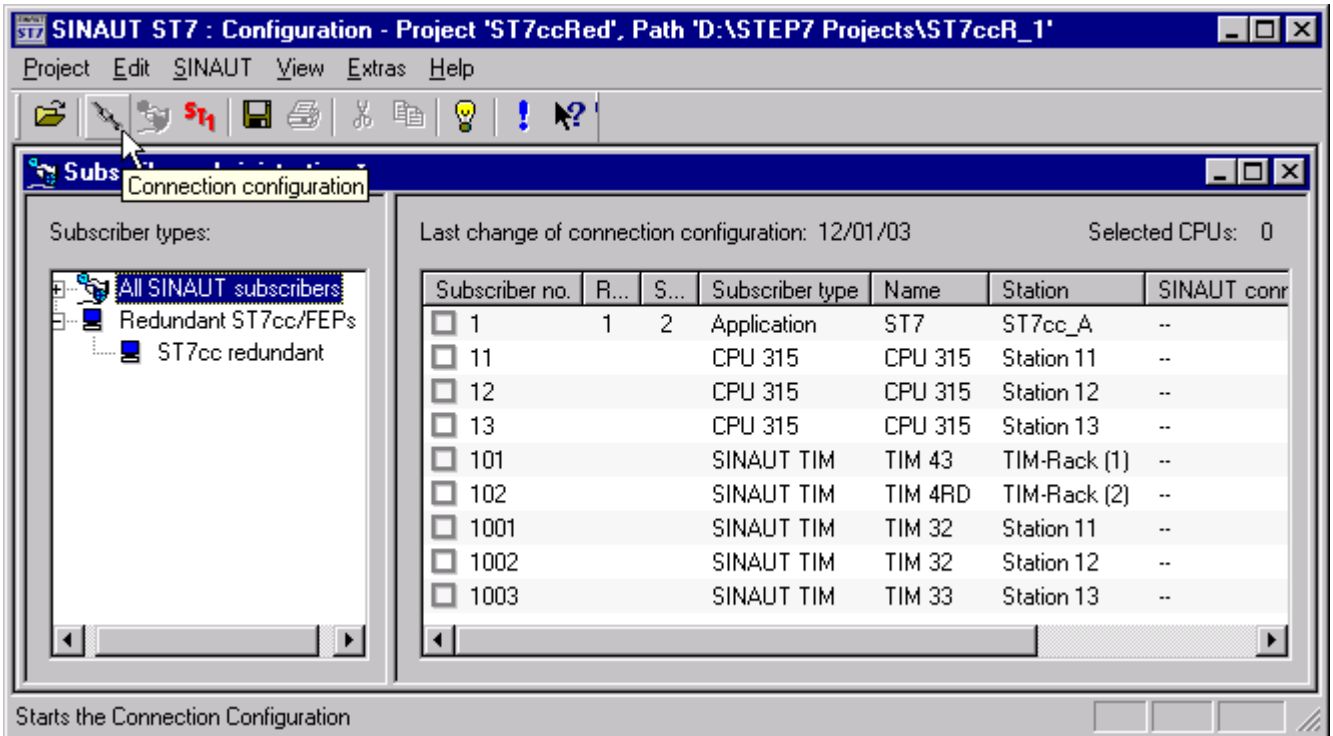


Figure 2-41 Selecting connection configuration

Changes saved?

If you have not yet saved changes made in subscriber administration, a prompt is displayed. This asks you whether you want to save your changes. Confirm this by clicking on Yes. The Options dialog appears (see figure).

Here, you should only save the subscriber numbers you have just changed in the SINAUT data management and the subscribers that form the redundant ST7cc group. Generation and compilation is unnecessary here.

1. Deactivate all options.
2. Close the dialog with OK.

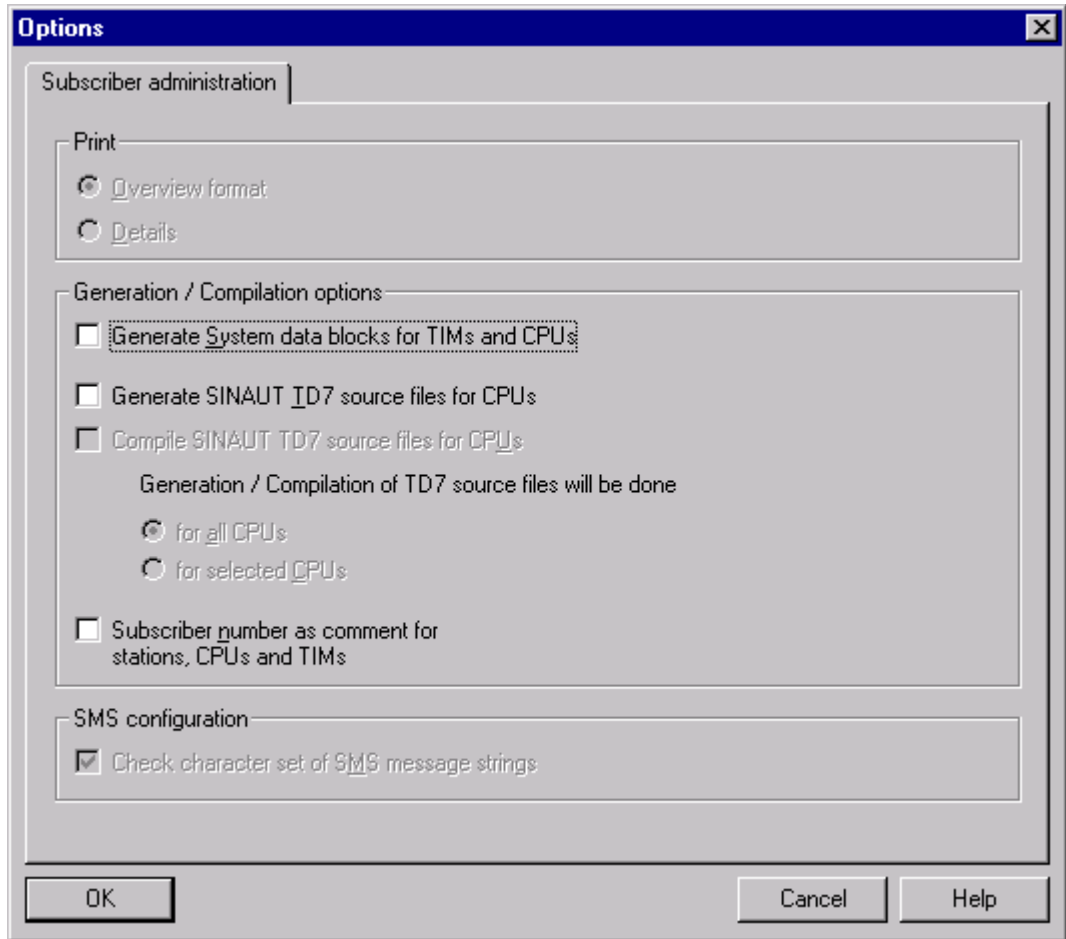


Figure 2-42 Saving the subscriber numbers without generating or compiling

A message is displayed to indicate when saving is complete.

3. Close the message dialog with *OK*.

The SINAUT connection configuration dialog is now opened (see figure).

Display of a redundant ST7cc in connection configuration

If you have a redundant ST7cc in your project, this ST7cc should only be displayed in the right-hand window in one row. If two rows are displayed for the redundant ST7cc (as shown in the figure in the first and last row), you will have to make a minor change to the display options.

Follow the steps outlined below:

1. Open the *Extras* menu.
2. Select Options... in the open menu (see figure).

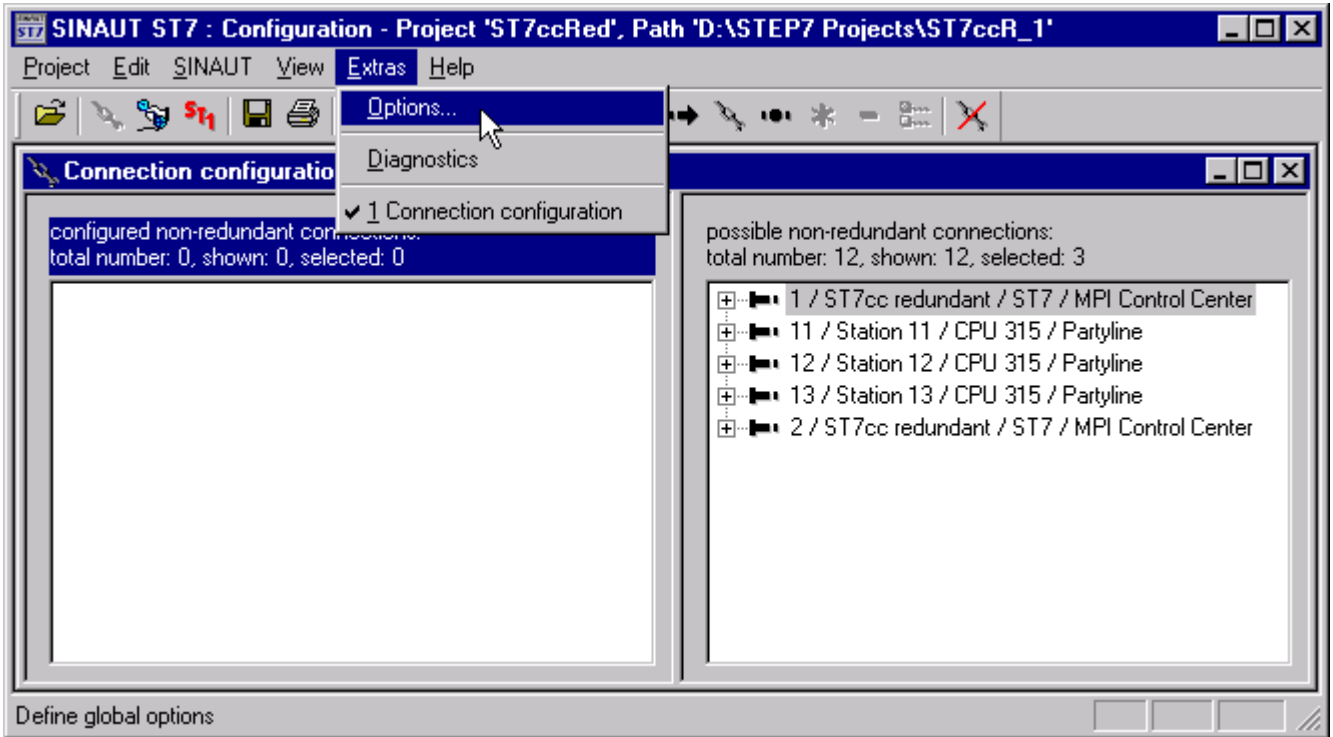


Figure 2-43 SINAUT ST7 connection configuration, selecting options

3. Adapt the format of the connection display by replacing the default entries &x with &y.

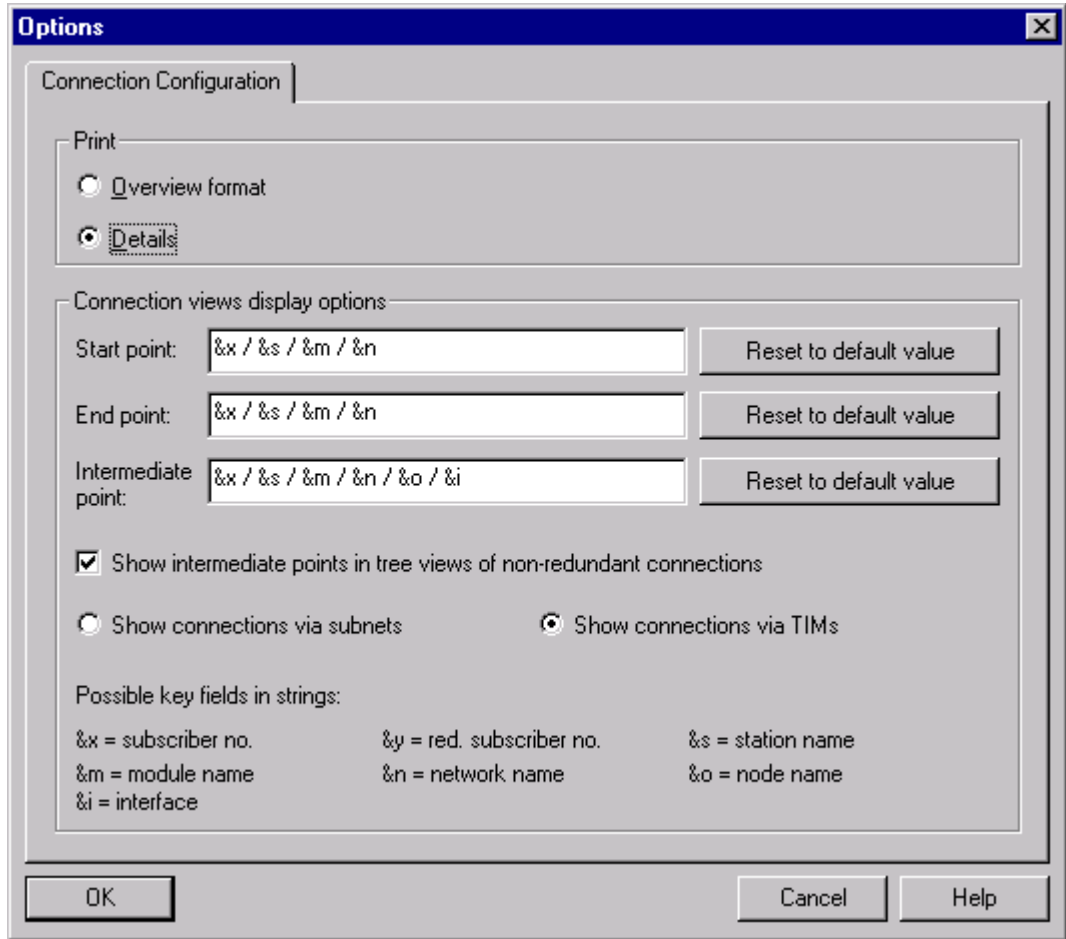


Figure 2-44 Changing the format of the connection display

4. Close the dialog with *OK*.

The connection configuration window is now displays the redundant ST7cc in one row (see figure).

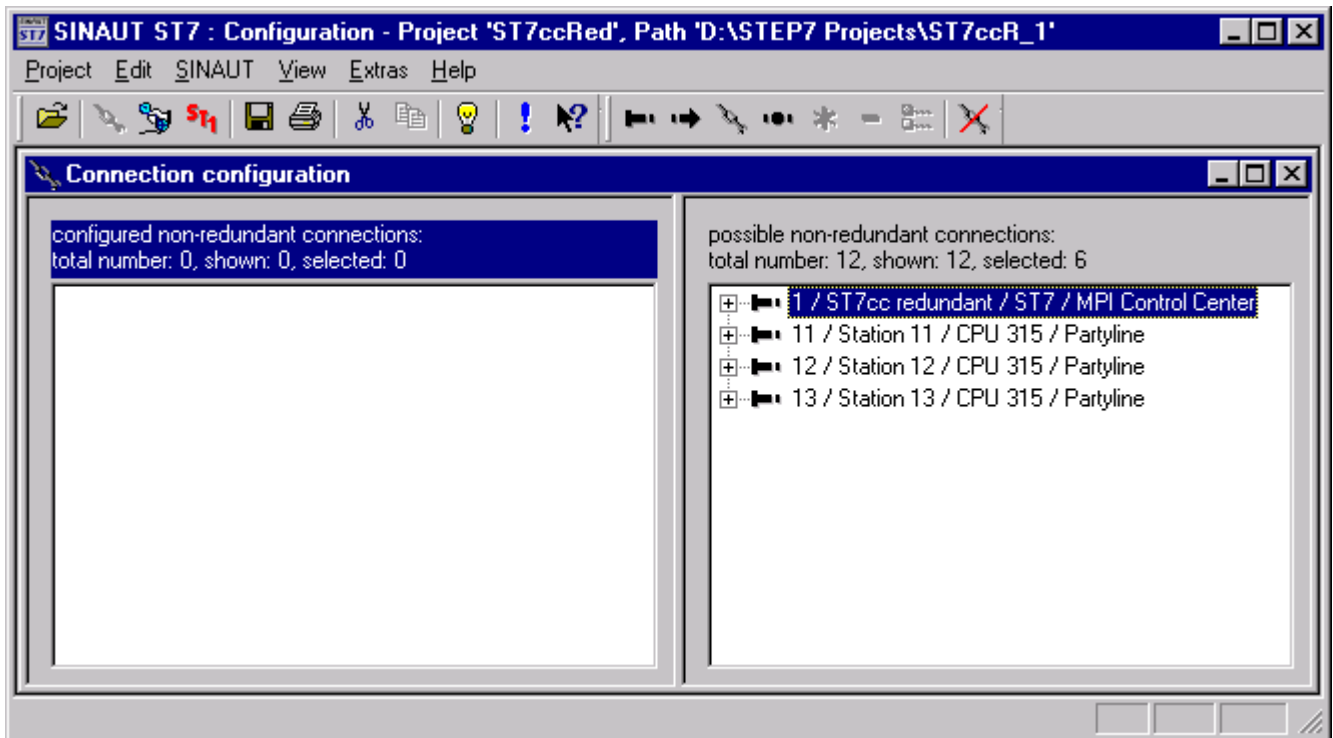


Figure 2-45 Redundant ST7cc displayed in only one row

Selecting SINAUT connections

When you open the SINAUT connection configuration, the program checks which devices were interconnected in SIMATIC NetPro and which SINAUT paths are theoretically possible between individual subscribers. The result is displayed in the right-hand window of SINAUT connection configuration. This window also shows all theoretically possible SINAUT connections. Your task is to select the SINAUT connections that you actually require in your project. After you have selected them, they are displayed in the left-hand window.

Note

SINAUT connection configuration reads in the currently available data only when it is opened and works with this data. If you make changes in one of the SIMATIC applications that are relevant to connection configuration while the connection configuration window is open, these are not included.

Solution:

Close the connection configuration and open it again.

Note:

Only close the connection configuration window, do not close the SINAUT ST7 Configuration parent window.

To see which subscribers can be interconnected, click on the + character to the left of the required subscriber.

In the figure, this was done for the ST7cc subscriber. All the subscribers with which a SINAUT connection would be possible are now listed below the ST7cc subscriber (in the example connections to stations 11, 12 and 13). Generally, ST7cc requires a connection to all stations.

In this case, follow the steps below:

In this case, follow the steps below:

1. Right-click on the ST7cc row.
2. Select Apply in the open menu (see figure).

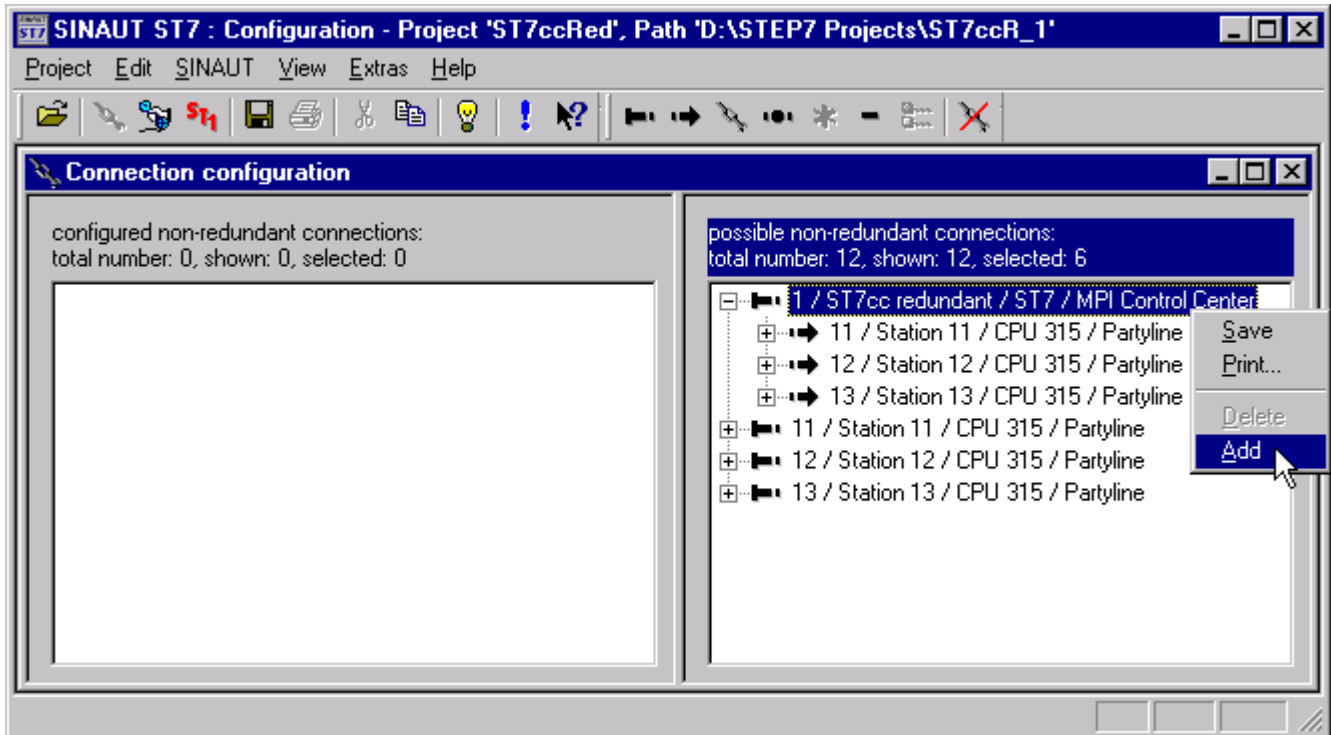


Figure 2-46 Applying SINAUT connections from the right-hand window

This enters all the SINAUT connections in the left-hand window that were displayed below the selected row (in the figure, the connections to stations 11, 12 and 13).

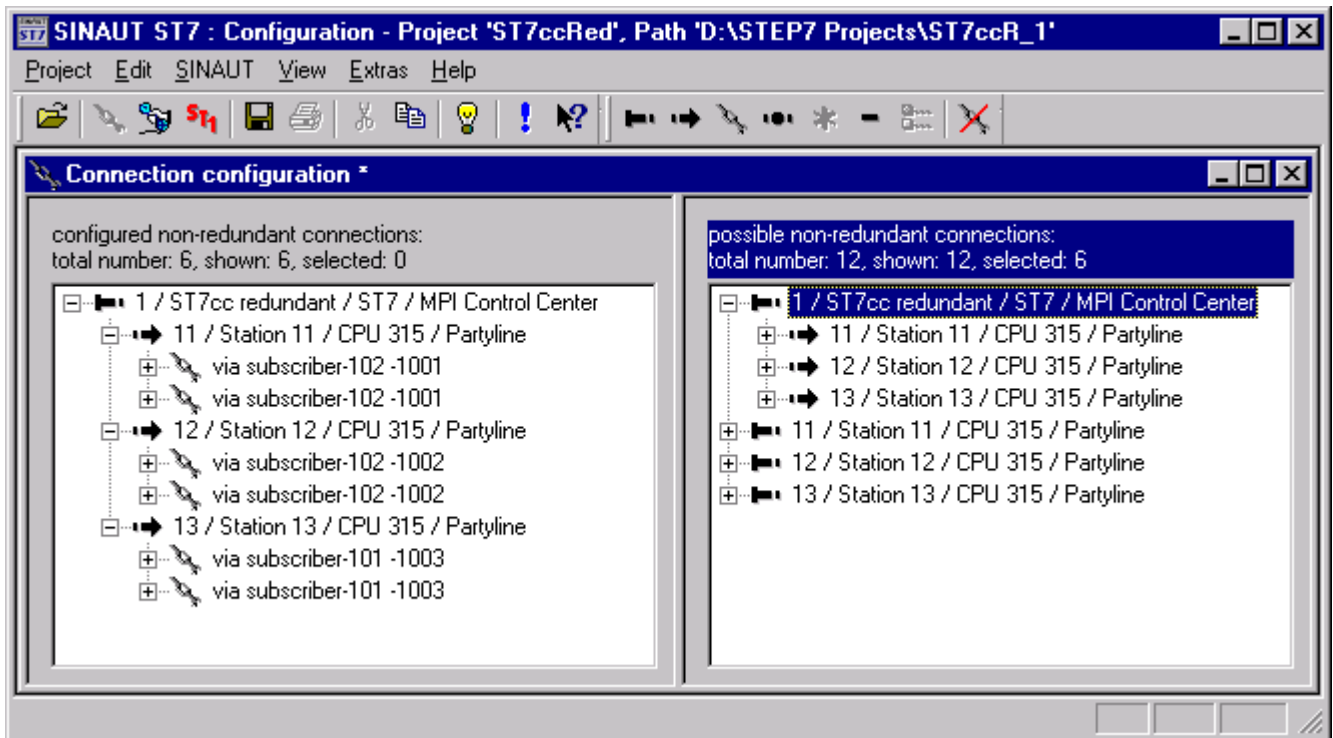


Figure 2-47 Display of the applied connections in the left-hand window

If no inter-station connections between individual stations are required, connection configuration is now completed.

If, however, you require inter-station connections, follow the steps outlined below:

1. In the same way as described above, select the connections required for one station to another.
2. Apply the connections as described above.

Note

For more detailed information on configuring SINAUT connections, refer to the SINAUT ST7 system manual.

As soon as connection configuration is completed, you can change back to *SINAUT subscriber administration* to start generation and compilation of the SINAUT data.

Changing from connection configuration to subscriber administration

To change from *SINAUT connection configuration* to the *SINAUT subscriber administration* click on the *Subscriber administration* button.

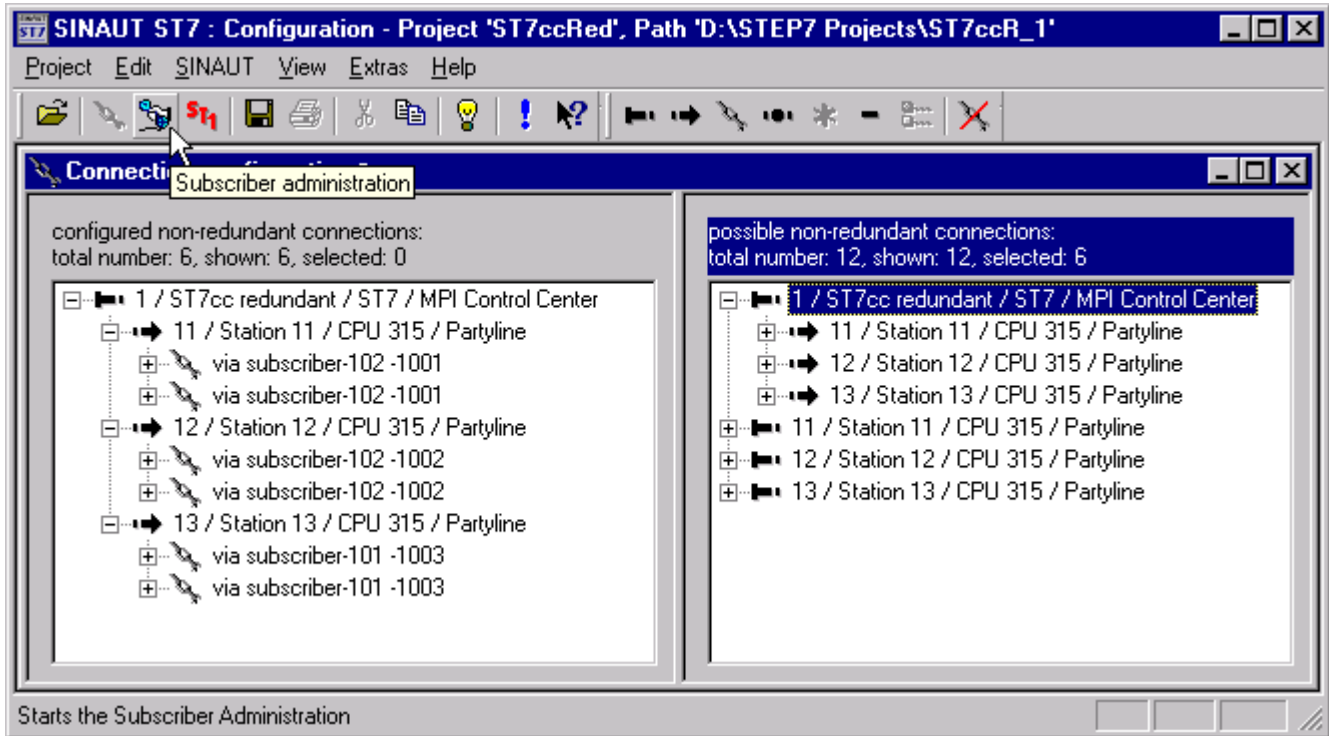


Figure 2-48 Selecting subscriber administration

Changes saved?

If you have not yet saved changes made in *Connection configuration*, a prompt is displayed. This asks you whether you want to save your changes. Click on *Yes*.

The *SINAUT subscriber administration* window opens (this can take a few moments).

2.6.4 Generating and compiling SINAUT data

Generating and compiling SINAUT data

On completion of connection configuration, *Yes* is entered in the *SINAUT-connected* column for all subscribers for which a SINAUT connection was configured.

Not only the CPUs and ST7cc are displayed as *SINAUT-connected* but also the TIM modules via which the configured SINAUT connections run.

Note

For more detailed information on configuration options available in the Subscriber administration dialog, refer to section PM-AQUA link (Page 321) in the SINAUT ST7 system manual.

To start generation and compilation, click on the Save button in the toolbar (see figure).

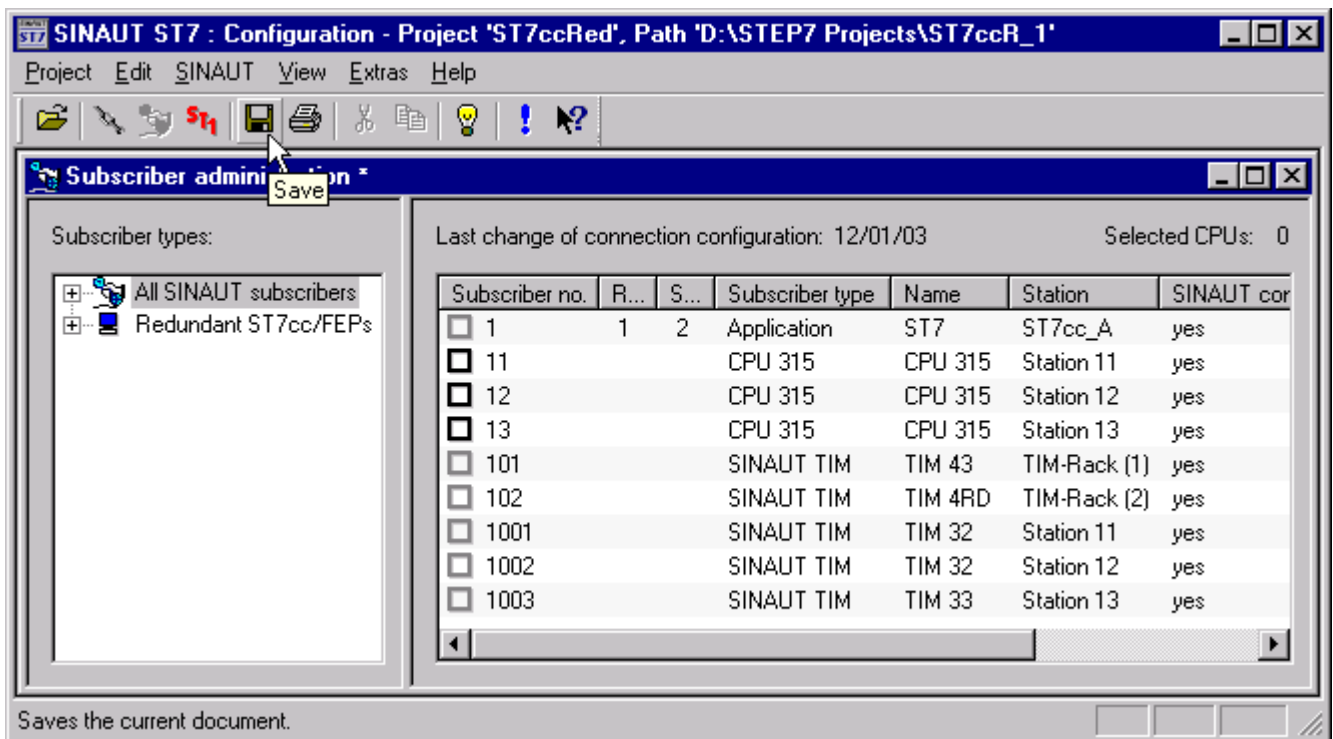


Figure 2-49 All subscribers SINAUT-connected

The warning shown in the figure appears.

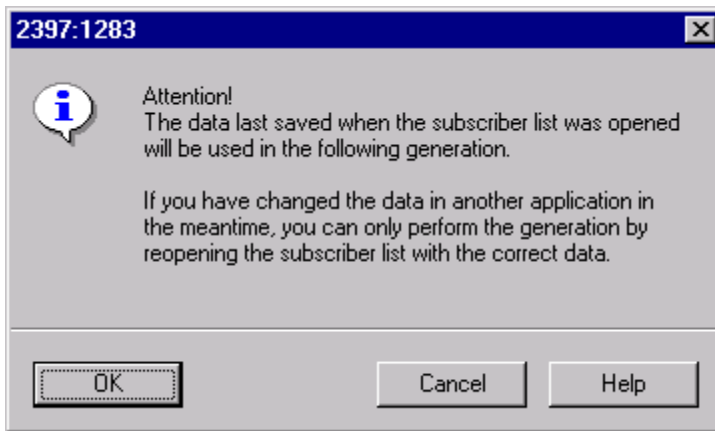


Figure 2-50 Warning at the start of the save function

If you are sure that you have made no further SINAUT-relevant changes in the SIMATIC Manager, SIMATIC HW Config or in SIMATIC NetPro after opening the subscriber administration, close the dialog with OK.

If you have made SINAUT-relevant changes in any of the SIMATIC applications listed above while the Subscriber administration window was open, follow the steps below:

1. Click the *Cancel* button.
2. Close *Subscriber administration* (close only the *Subscriber administration* window, not the *SINAUT ST7 Configuration* parent window).

A further user prompt appears asking you whether you want to save the changes in *Subscriber administration*.

1. Click on the *No* button to close the window.
2. Open the *subscriber administration* window again.

All the SINAUT data is read in again, including the data you may have changed in the SIMATIC applications listed above.

Note

You should make it a habit to close the Subscriber administration after saving is complete. It is then no longer possible that you make SINAUT-relevant changes in SIMATIC applications that are not included in subscriber administration because the window is still open.

Since SINAUT connections were configured for all subscribers in this case, you need to start a complete generation and compilation of the SINAUT data.

Follow the steps outlined below:

1. Select the options:
 - Generate System data blocks for TIMs and CPUs
 - Generate SINAUT TD7 source files for all CPUs
2. Close the dialog with *OK*.

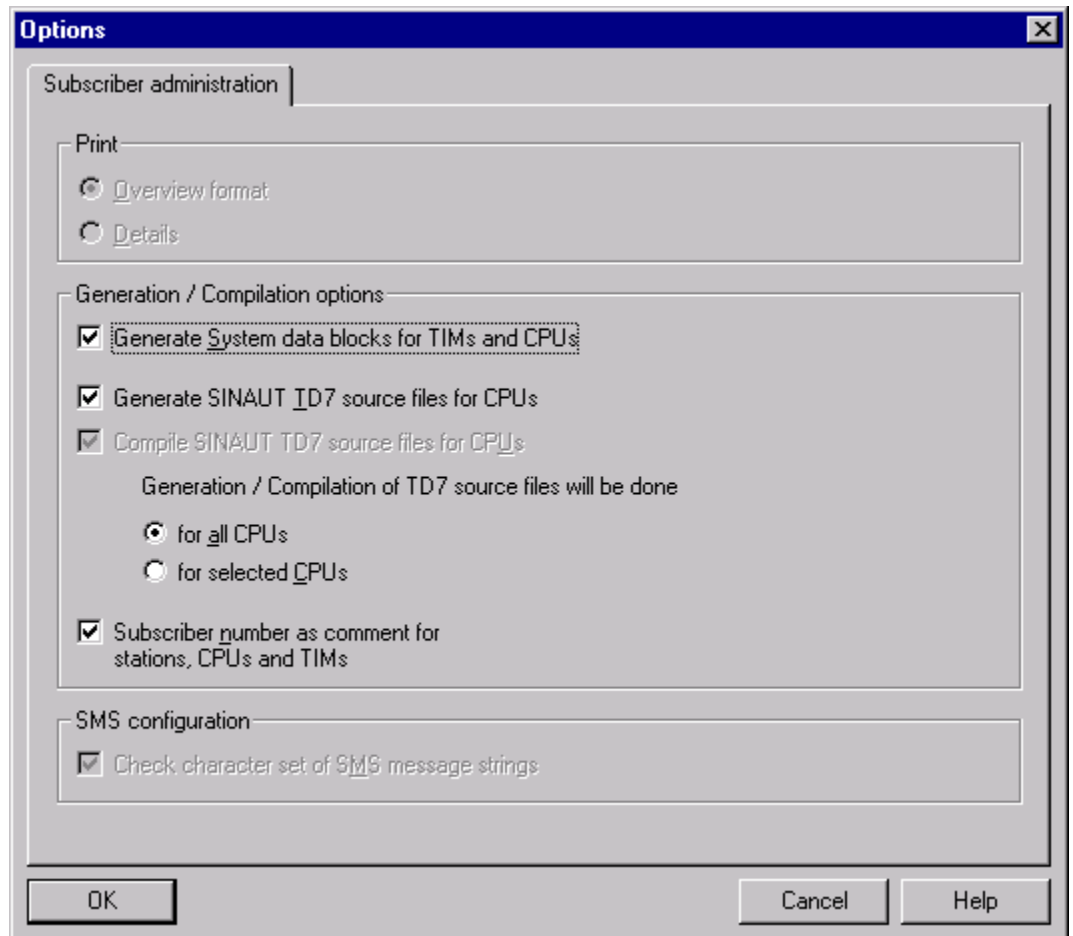


Figure 2-51 Selected options for generating and compiling all SINAUT data

The option Subscriber number as comment for stations, CPUs and TIMs also selected in the figure is not absolutely necessary but is recommended. This requires very little time during generation.

The window of the STL compiler is opened after a few moments. There, you can see how the SINAUT program sections are generated for the individual CPUs (figure).

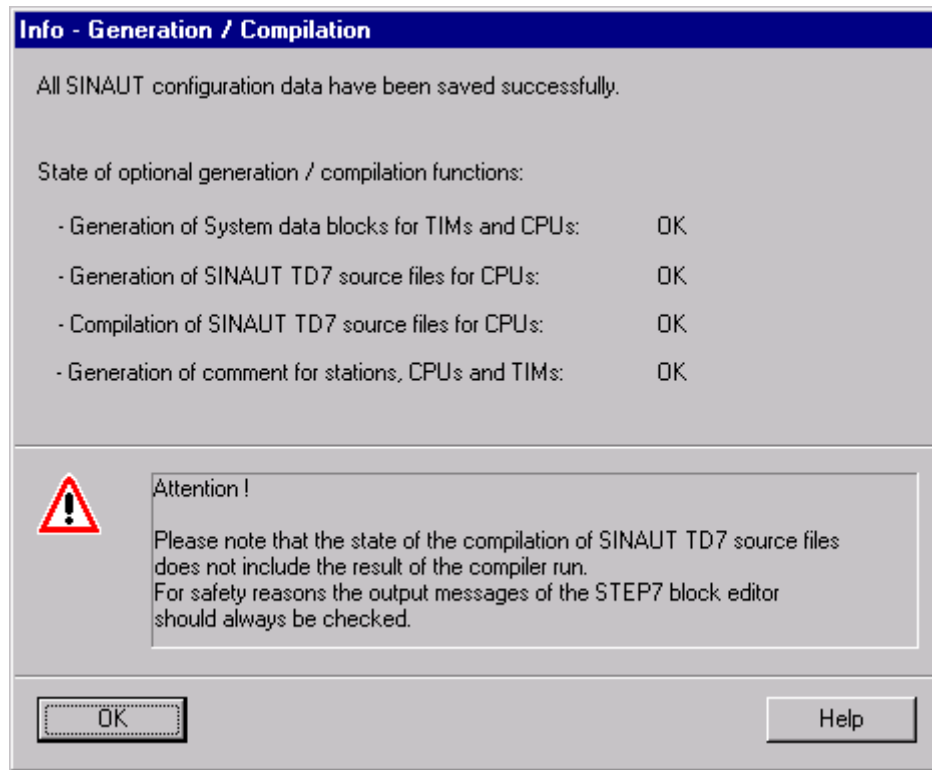


Figure 2-53 Message when saving is complete

The OK status should be displayed in this window for all started generation and compilation activities (see figure).

The Attention! note in this dialog has the following background. When SINAUT program sections are compiled in the STL compiler, errors can occur. These error messages are not transferred to the SINAUT save tool. The tool is only informed that compilation was completed without being informed of any errors. As shown in the figure, only the status OK can therefore be displayed for the compilation of the SINAUT TD7 source files. Whether or not compilation errors have occurred must be checked separately by the user.

Compilation errors are displayed in the bottom line of the compiler window. In the figure, you can see the relevant line, for example, with the message:

Compiler results: 0 errors, 262 warnings.

Here, no errors occurred. The warning this can be ignored. As long as 0 errors are displayed, the compilation for the CPU was OK.

1. Please check all the open windows in the STL compiler (a window is opened for every CPU) to make sure that 0 errors occurred.

Note

You can also check this during saving because the compiler result of the CPU that has just been processed is visible for a short period before the window for the next CPU is opened.

We recommend that you close all STL compiler Windows that were opened during saving on completion of the save function.

2.6.5 Downloading SINAUT data to TIMs and CPUs

Download

The SDBs generated for the TIMs now contain all the data required by the TIMs for the SINAUT connections. You can now download these SDBs to the TIMs.

Follow the steps outlined below:

- With a stand-alone TIM:

Connect your PG with the MPI interface of the TIM.

- With a TIM in an S7-300 station:

Insert your MPI cable in the MPI interface of the CPU.

1. Open the block container of the TIM to which you want to copy the SDBs in the SIMATIC Manager.
2. In the right-hand window, select the system data and start the download by clicking on the Download button (see figure).

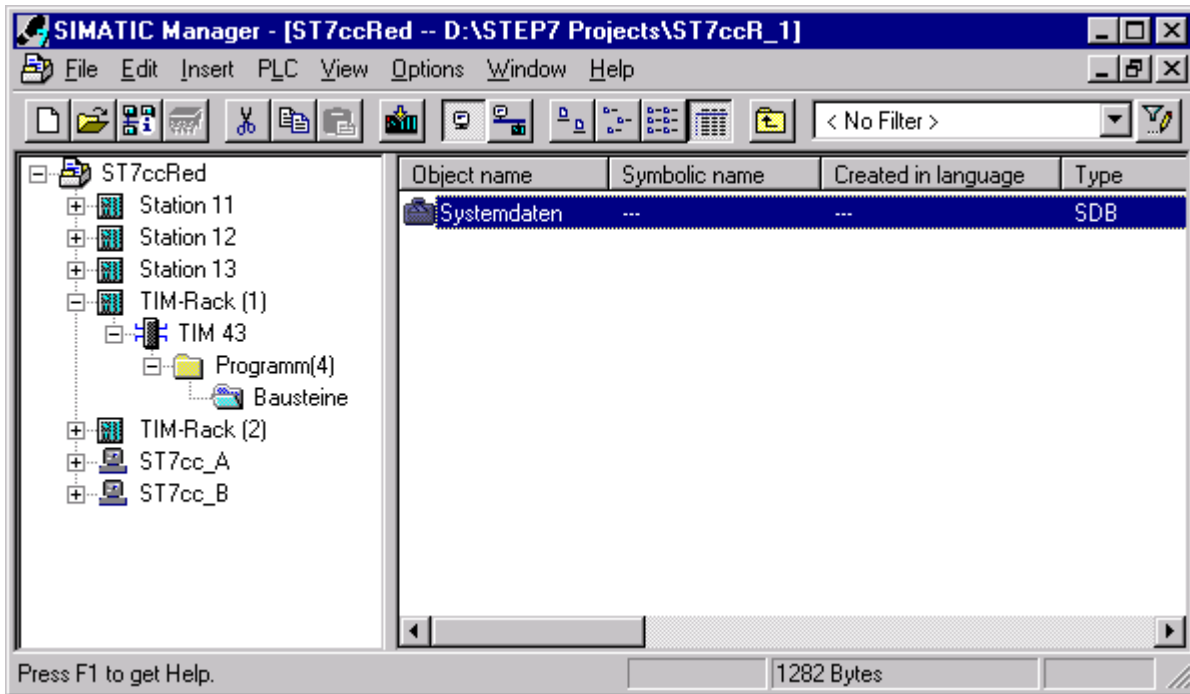


Figure 2-54 Downloading the system data to a TIM

Note

You should always download SDBs to a TIM in the SIMATIC Manager. Only then can you be sure that all SDBs have been copied to the TIM. If you download SDBs in other SIMATIC applications (HW Config, NetPro), only some of the SDBs will be downloaded in some situations.

During the download, you will be asked whether the TIM should be changed to stop. Click the *Yes* button.

At the end of the download, a dialog appears asking whether you want to *restart*. Once again, click the *Yes* button.

Note

The newly downloaded SDBs are activated on the TIM only after a restart.

The program created for the CPUs is prepared for SINAUT communication, however it must still be extended before it can execute. The figure shows the content of a CPU block container following SINAUT generation.

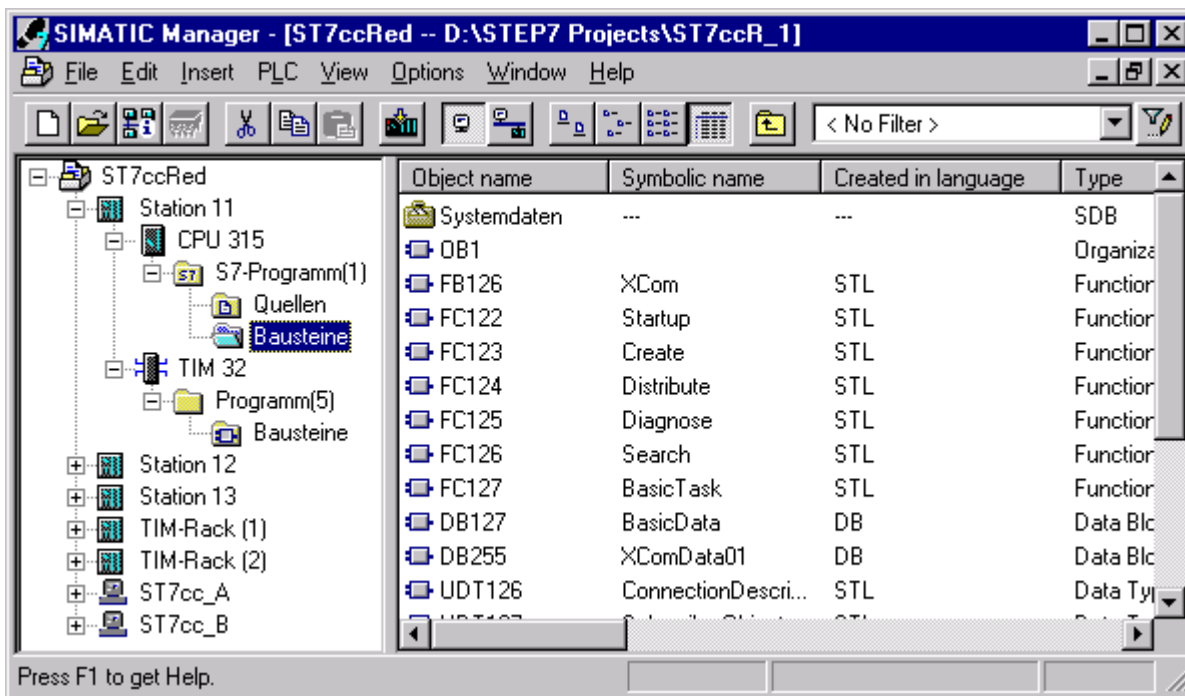


Figure 2-55 Standard SINAUT blocks in the block container of a CPU

The block container contains the FBs and FCs that are always required for SINAUT, the central SINAUT records data block BasicData and the communications data block XComData01. To obtain an executable program, you need to include the program required for SINAUT in the cyclic OB1 and in the startup OB100. You will find detailed information on this in the SINAUT ST7 system manual.

Once you have completed the CPU program, you can download the entire program to the CPU. After starting the CPU, it is ready to transfer data to the intended partners assuming that they are available and capable of receiving data.

ST7cc is not yet capable of this. The necessary configuration data is still missing. The following sections explain how to continue from here.

Creating an ST7cc project

3.1 Creating and opening an ST7cc project

3.1.1 Starting ST7cc Config

To start ST7cc Config, follow the steps outlined below:

1. Start ST7cc Config using the Windows menu sequence
Start > Simatic > ST7cc > ST7cc Config

ST7cc Config opens with an empty window, in other words, no project is initially loaded (see figure).

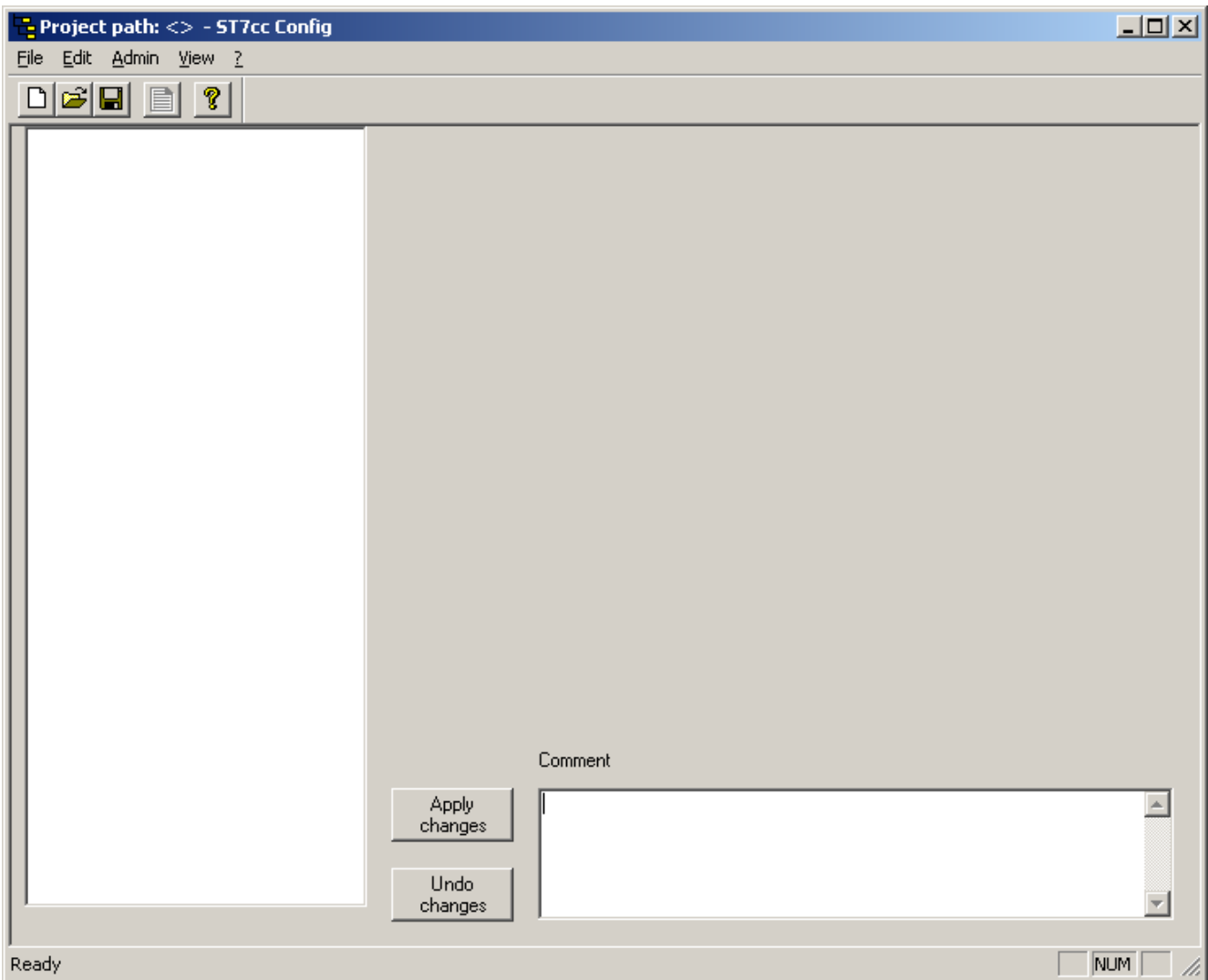


Figure 3-1 ST7cc Config opens with an empty window

The following menus are available in the menu bar for working with the configuration tool:

- File
- Edit
- Administration
- View
- ?

The following sections explain how to use these menus to create, manage and open an ST7cc project.

3.1.2 Creating a new ST7cc project

Creating an ST7cc project

There are two ways to create a new ST7cc project:

- With the New button
- Using the menu sequence File > New

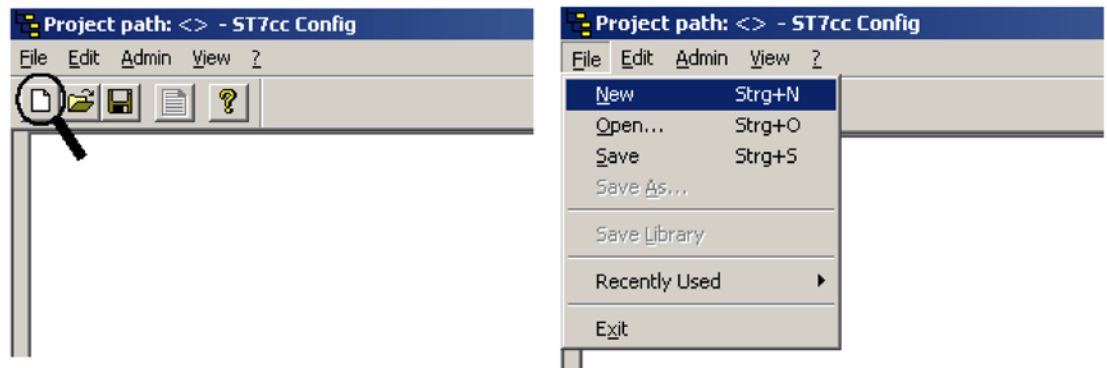


Figure 3-2 Creating a new project using the New button or File > New

You can create a new ST7cc project in the following locations:

- Directly in the WinCC project directory (recommended), for example:
C:\Siemens\WinCC\WinCCProjects\<Projectname>\ST7cc
- In any directory of your choice

Creating an ST7cc project in the WinCC project directory

To create a new ST7cc project, follow the steps outlined below:

1. Click on the New button (see figure).
The Directory selection opens (see figure).
2. Select the drive on which your WinCC project is stored.

3. In the WinCC project directory, in the figure for example, select the WinCC project ST7cc_Documentation.

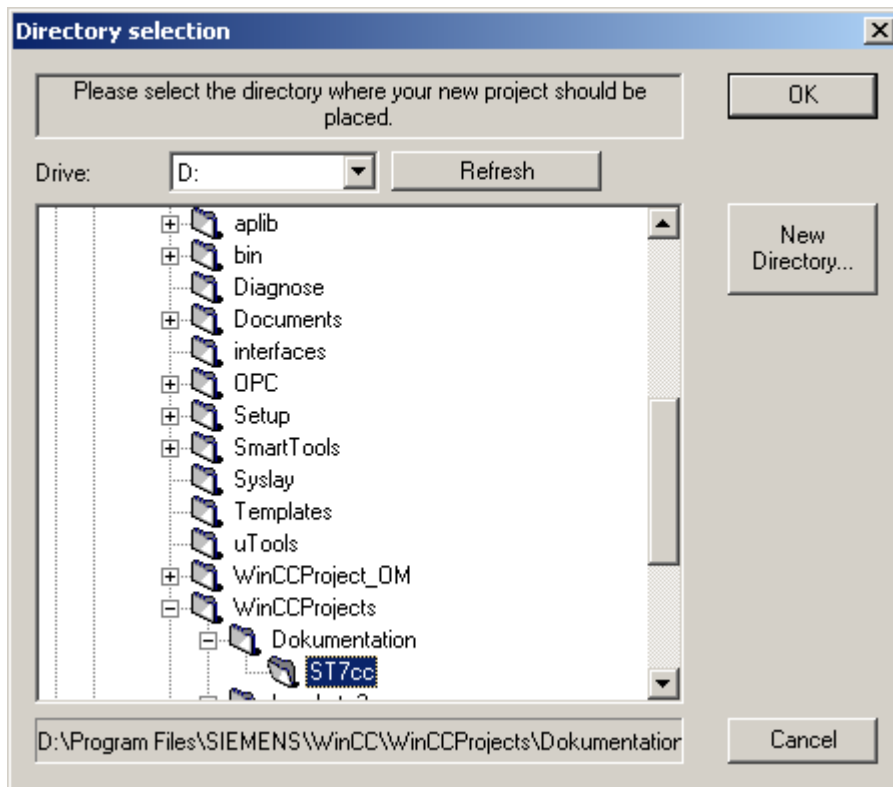


Figure 3-3 Dialog for creating an ST7cc project within the WinCC project.

4. In the WinCC project directory, select the *ST7cc* folder (if this does not exist, see the note below).
5. Close the dialog with OK.

The new project is created in the *ST7cc* folder.

Note

If the ST7cc folder is not displayed in the WinCC project directory, the WinCC project was probably created before ST7cc was installed. To remedy the situation, open the WinCC project and then click on the Refresh button in the Directory selection dialog. The ST7cc folder should then be displayed in the Directory selection dialog.

The ST7cc Config dialog now contains the library and the internal system subscriber 0 System (see figure). The subscriber 0 System is required for system-specific status variables and contains the ServerState object.

The title bar of the window (here shown blue) contains the project path with the assigned project name *Documentation*.

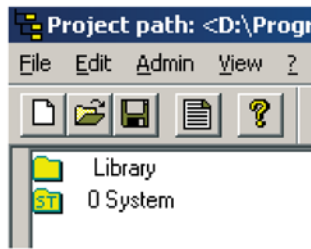


Figure 3-4 Newly created ST7cc project with library and subscriber 0 (subscriber: system)

Creating an ST7cc project in a directory of your choice

If you want to create a project in a project directory of your own choice, follow the steps below:

1. Click on the *New* button.
2. Select the drive where you want to create the new project directory.
3. Select the directory in which you want to create the new project directory, in the figure for example in ST7CC.

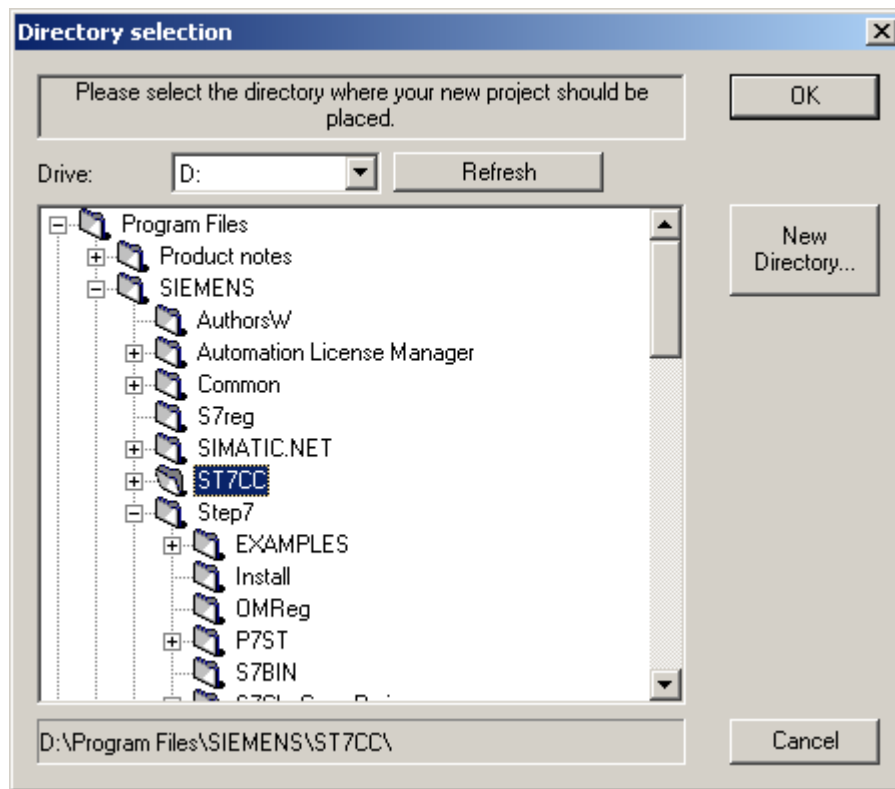


Figure 3-5 Directory selection dialog for creating a new project directory.

4. Click on the *New Directory...* button to enter a name for your new project directory.

3.1 Creating and opening an ST7cc project

5. In the dialog that now opens, enter the name of your new project (for example *ST7cc_Documentation*).

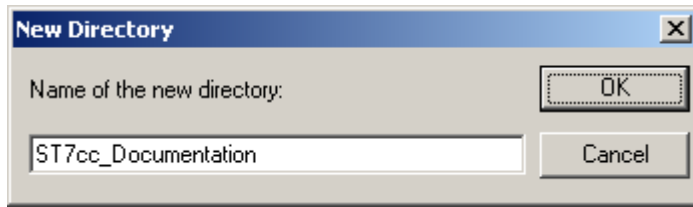


Figure 3-6 Dialog for entering the project name

6. Close the dialog with *OK*.

The new project directory has been created at the location you selected in the directory tree (see figure).

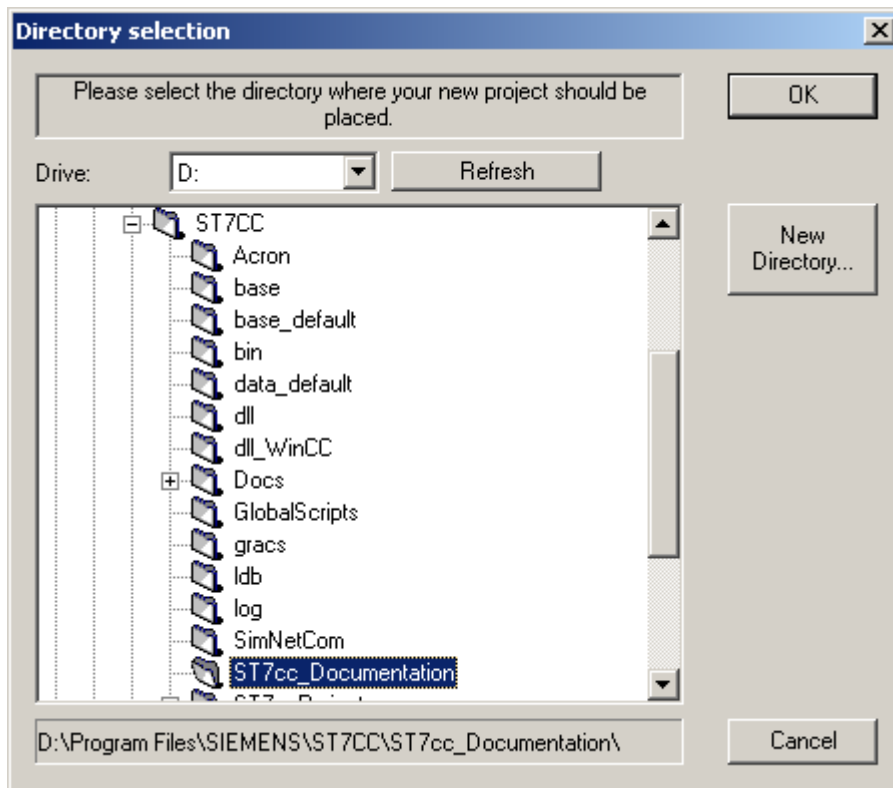


Figure 3-7 Newly created ST7cc project directory

7. Close the dialog with *OK*.

The new project in the *ST7cc* folder is completed.

Content of the newly created ST7cc project

A newly created ST7cc project always consists of five files:

- ST7_PROJECT.TXT
- ST7_PROJECT.XML

- ST7_PROJECT_ENGLISH.TXT
- ST7_TYPICALS.TXT
- ST7_TYPICALS_ENGLISH.TXT

More files are added later when your ST7cc PC goes online with the configured project. Files of the type .mmf are then created in the project directory. These contain the process image and are used as a buffer to temporarily store messages and data.

Once the ST7cc project is created, the ST7cc Config window contains the library and the internal system subscriber 0 System (see figure).

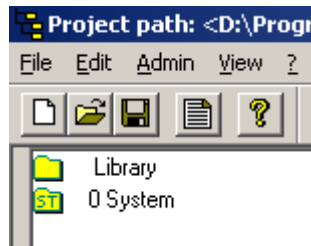


Figure 3-8 Newly created ST7cc project with library and subscriber 0 (subscriber: system)

1. Double-click on subscriber 0 (0 System).
 Within 0 System, there are two objects 1 ServerStatus and the 10 PM-AQUA object (see figure).
 The variables in object 1 *ServerStatus* allow the status of the ST7cc server to be displayed in WinCC using a supplied faceplate.
 Object 10 in subscriber 0 is reserved for the WinCC add-on PM-AQUA. If you do not want to use PM-AQUA, this object can be deleted to save variables.

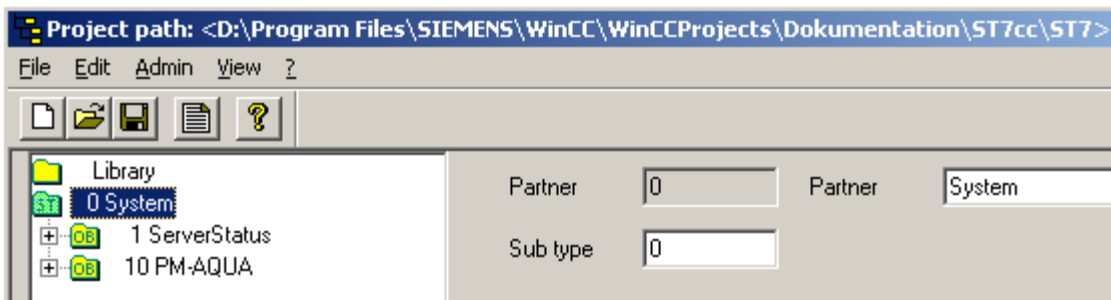


Figure 3-9 Standard object in subscriber 0 (system subscriber)

2. Click on the plus sign (+) in front of 1 *ServerStatus*.
 The typical instances for Server1 and Server2 are displayed.
 In the typical instance, you will find or the system variables that provide information on the status of the relevant server.

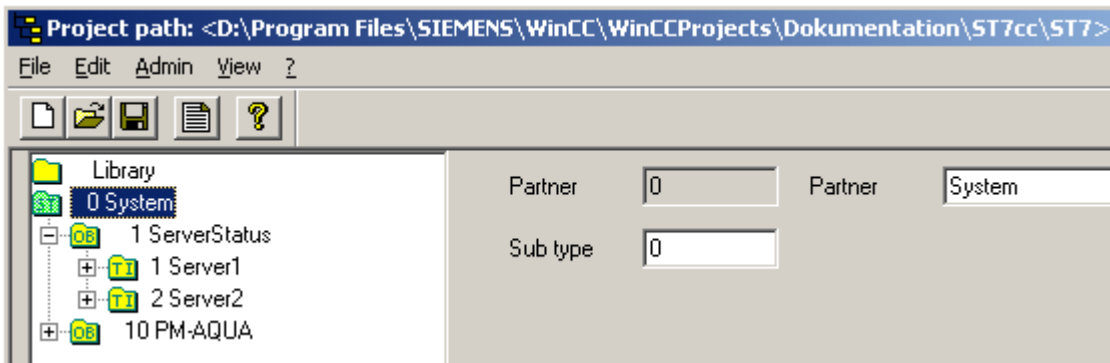


Figure 3-10 Typical instances for Server1 and Server2

If you do not require these status variables, you can delete the server typicals.

If you are using a non-redundant ST7cc system, you can delete the typical instance *Server2* in any case.

Note

Deleting unnecessary typicals saves WinCC tags.

If you delete both typical instances *Server1* and *Server2*, the *Server information* option must be disabled in the project settings.

Deleting server typicals means that information is lost! Make doubly sure that you do not require this information.

Number of WinCC tags required for ST7cc system data

You can save WinCC tags by deleting objects and typical instances that are created automatically for the ST7cc system but that are not required by the user.

More system data is added during the course of the project configuration, for each station and each TIM connected to the ST7cc PC locally over MPI or Ethernet. No WinCC tags can be saved for the system data of these subscribers.

The number of WinCC tags required for the system data is listed in the table below.

Table 3- 1 Number of WinCC tags required for ST7cc system data

System object	Number of WinCC tags	Note
System	5	Maximum required once
Server	21	Per server, max. of 2 servers for redundant ST7cc
PM-Aqua	4	Per raw data channel
Station	15	Per ST1 or ST7 station
Local TIM	8	Per local TIM

3.1.3 Opening an existing ST7cc project (ST7cc version 2)

There are several ways of opening an existing project:

- Using the *Open* toolbar button
- Using the menu sequence *File > Open*
- Using the menu sequence *File > Recently used*

After clicking on the Open button or after *File > Open*, the *Open* dialog appears.

1. In Look in: select the relevant project directory (see figure).

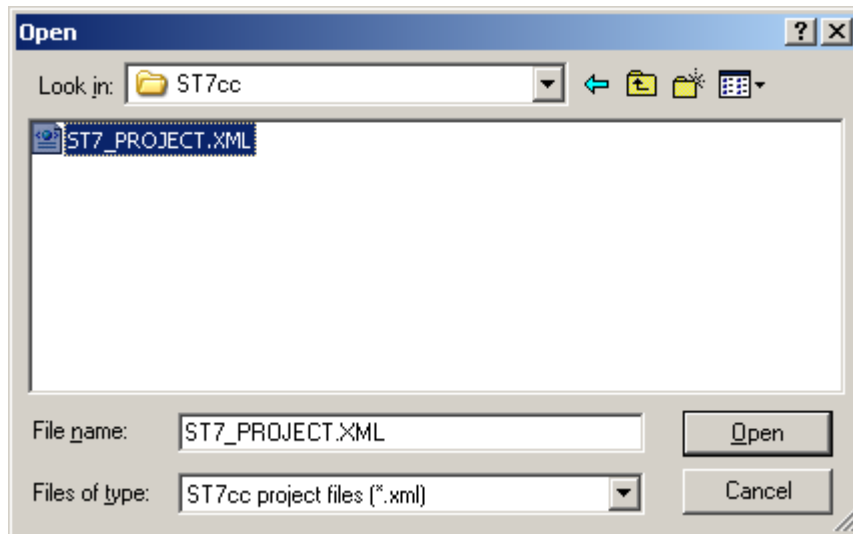


Figure 3-11 Selecting and opening the project ST7_PROJECT.XML

2. Double-click on *ST7_PROJECT.XML*.

Note

The ST7cc project file always has the name *ST7_PROJECT.XML*.

It is even easier to open the project using the Recently used menu command: Here, you see the names of the last ST7cc projects to be opened (see figure).

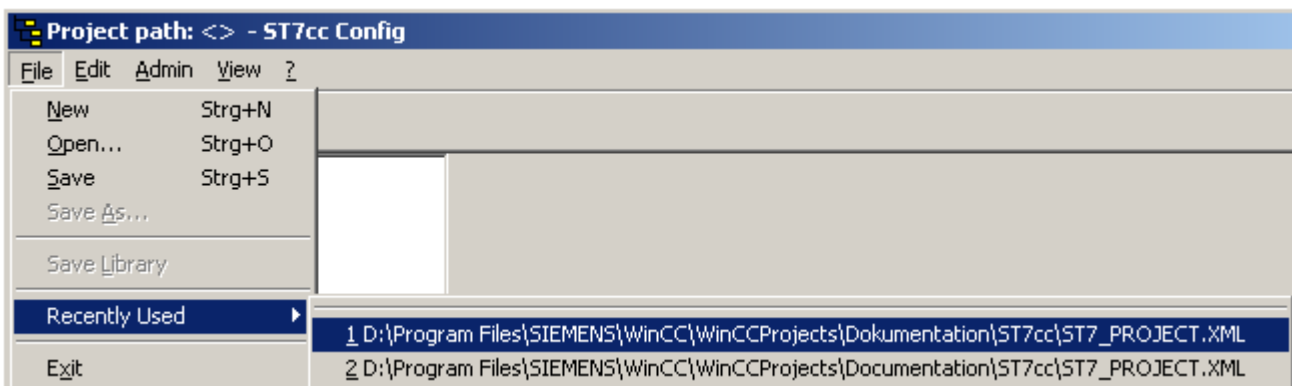


Figure 3-12 Opening an existing project using File > Recently Used

3.2 ST7cc Administration

The *Admin* menu command provides the following options:

- Copy faceplates to a WinCC project ...

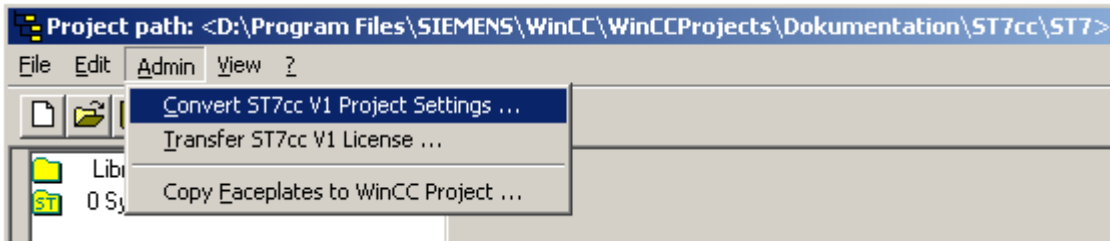


Figure 3-13 Options available with the Admin menu command

3.2.1 Copy faceplates to a WinCC project

ST7cc provides standard faceplates and picture typicals for stations, TIMs, and servers (see section Diagnostics: Subscriber typicals and faceplates (Page 306)). Picture typicals and faceplates are also supplied for commonly required technological objects (see section Technological typicals (Page 333)).

With the menu command described here, you can transfer these faceplates and picture typicals to your WinCC project.

1. Open the Admin menu and select the menu command Copy Faceplates to WinCC Project.
2. Open your WinCC project directory in this dialog. Select the mcp project file by clicking on it.

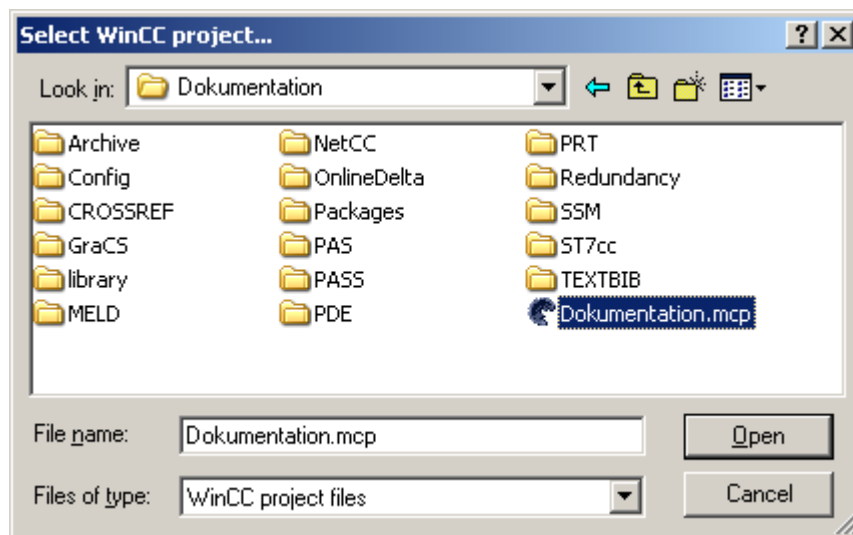


Figure 3-14 Dialog for selecting the WinCC project file

3. Click Open.

The following faceplates and picture typicals are now copied to your WinCC project:

Name	Explanation
st7_typical.pdl	This file contains the standard picture typicals for the ST7cc server, stations and local TIMs.
fpl_loclatim.pdl	Faceplate for a local TIM
fpl_mastertim.pdl	Faceplate for a master TIM
fpl_server.pdl	Faceplate for an ST7cc server
fpl_station.pdl	Faceplate for a station
fpl_stationdetails.pdl	Faceplate for the connection details of a station
fpl_statistics.pdl	Faceplate for message statistics for stations or local TIMs
st7_technicalobjects.pdl	Contains all picture typicals for technological objects
fpl_compressor.pdl	Faceplates for technological objects
fpl_generator.pdl	
fpl_motor1.pdl	
fpl_motor2.pdl	
fpl_pump.pdl	
fpl_slider.pdl	
fpl_valve.pdl	

3.3 Project settings

You can select the project settings with the menu sequence: Edit > Project Settings (see figure) or using the F2 key.

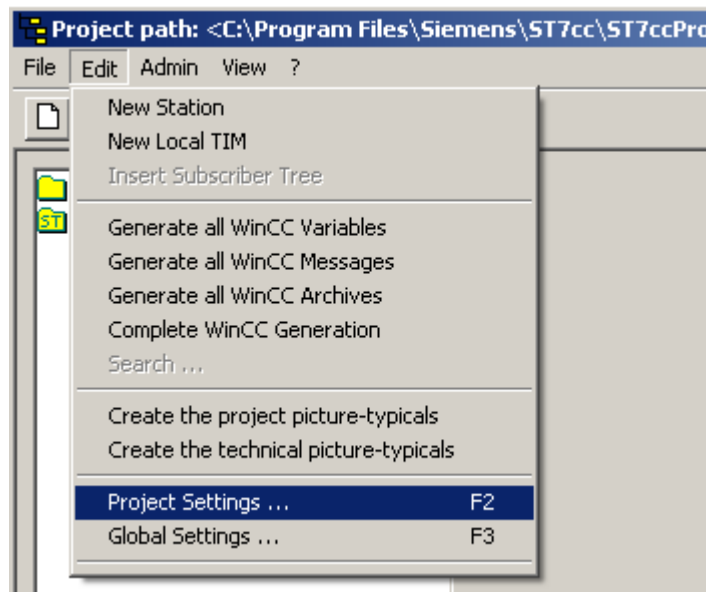


Figure 3-15 Opening the project settings from the Edit menu

The following sections describe the possible settings in the individual tabs.

Note

All project settings are saved in the file *ST7_PROJECT.XML* and are therefore located in the project directory.

3.3.1 Project settings: Server

The following figure shows the settings of the Server tab. Adapt the settings to suit your particular project. With a redundant ST7cc system, for example, activate the Configuration is redundant option, or if you connect stations over a dial-up network, increase the times for Timeout general request start and Timeout general request end.

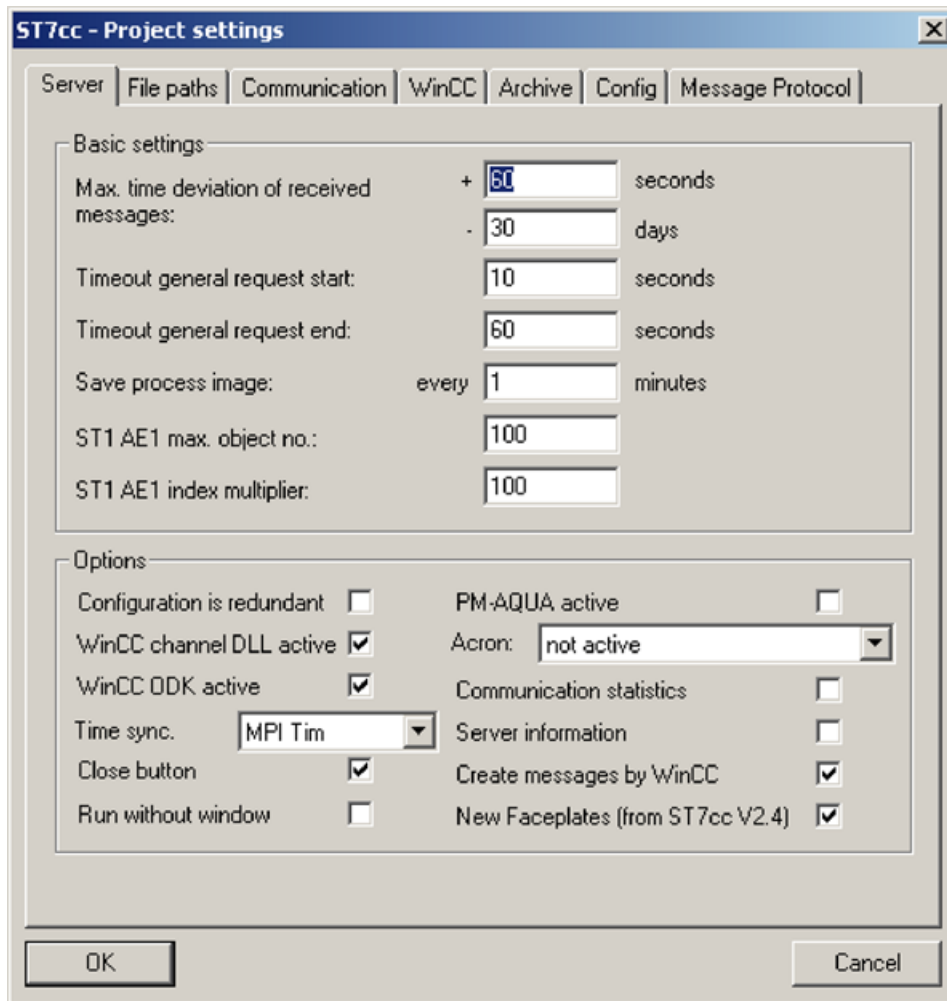


Figure 3-16 Project settings of the ST7cc server

Basic settings

Parameter:	Max. time deviation of received messages
Possible setting:	Positive time in seconds Negative time in days
Default:	+ 60 [seconds] – 30 [days]
Explanation:	ST7cc compares the time stamp in the received message with its own current time. Messages that are newer or older than specified here will be discarded. This is reported accordingly in the SINAUT Log window.

Parameter:	Timeout general request start
Possible setting:	Time in seconds
Default:	60 [seconds]
Explanation:	The maximum time that may elapse before a station reports back that a general request has started. If dial-up stations are connected to ST7cc, increase this time to a practical value, for example to 5 minutes. If this time is exceeded, a message to this effect is displayed in WinCC Alarm Logging and in the SINAUT log window.

Parameter:	Timeout general request end
Possible setting:	Time in seconds
Default:	60 [seconds]
Explanation:	The maximum time that may elapse after a general request start message before a station reports back that the general request has ended. If dial-up stations are connected to ST7cc, increase this time to a practical value, for example to 15 minutes. If this time is exceeded, a message to this effect is displayed in WinCC Alarm Logging and in the SINAUT log window.

Parameter:	Save process image every
Possible setting:	Time in minutes
Default:	1 [minute]
Explanation:	<p>The timebase used to write the ST7cc process image to the ST7_PROJECT.MMF file on the hard disk. If the ST7cc server stops, the ST7cc process image is saved immediately on the hard disk.</p> <p>After failure or stoppage of the ST7cc server, the ST7cc server loads the last saved process image. Starting with this process image, which is then constantly updated by the stations with the latest data, the ST7cc server supplies the WinCC tag management, the WinCC message system and the WinCC archive.</p> <p>The setting of 1 minute should only be changed when the system response becomes sluggish due to large amounts of data and when the throughput on the hard disk slows down.</p>

Note

The following parameters ST1 AE1 max. object no. and ST1 AE1 index multiplier are relevant only when ST7cc receives SINAUT ST1 messages with the address extension (AE1). An address extension is used with ST1 only when data is transferred to the SINAUT LSX control center system. In all other situations, ST1 stations send their data in messages without an address extension.

ST1 messages with an address extension normally require further explanation (including how to assign the two parameters named above). Please call the ST7cc hotline if you require further information.

Parameter:	ST1 AE1 max. object no.
Possible setting:	See note above
Default:	100
Explanation:	<p>Maximum ST1 object number.</p> <p>This parameter is relevant only when ST7cc receives SINAUT ST1 messages with an address extension (AE1).</p>

Parameter:	ST1 AE1 Index multiplier
Possible setting:	See note above
Default:	100
Explanation:	Multiplier for the ST1 index number. This parameter is relevant only when ST7cc receives SINAUT ST1 messages with an address extension (AE1).

Options

Option:	Configuration is redundant
Possible setting:	active not active
Default:	not active
Explanation:	<ul style="list-style-type: none"> Setting not active: The ST7cc PC is a single system. There is no redundant partner device. Setting active: The ST7cc PC is part of a redundant ST7cc system. There is a redundant partner device. <p>When it starts up, the ST7cc server then checks whether a redundancy license is installed, whether the computer name and IP address of the partner PC have been set, activates the mechanisms for synchronization of the redundancy partner, etc.</p>

Note

If redundancy is activated, and no redundancy license is found, the ST7cc server automatically switches to single computer mode during startup.

Note

Enabling or disabling of the Configuration is redundant option requires entries and actions in several other ST7cc Config dialogs before ST7cc will react correctly to the enabling or disabling of redundancy.

- In the Global Settings dialog, Computer tab:
The computer name and IP address must be entered for Server 2 (when enabling redundancy) or must be removed (when disabling redundancy). The modified data must then be activated by clicking on the Add server information to system button.
 - In the Global Settings dialog, Project tab:
The project must be activated again by clicking the Activate current project for ST7cc Runtime button.
-

3.3 Project settings

Option:	WinCC channel DLL active
Possible setting:	active not active
Default:	active
Explanation:	If WinCC is not intended as operator control and monitoring system, this option can be deselected. The two options <i>WinCC channel DLL active</i> and <i>WinCC ODK active</i> (see above) should always have the same status.

Option:	WinCC ODK active
Possible setting:	active not active
Default:	active
Explanation:	If WinCC is not intended as operator control and monitoring system, this option can be deselected. The two options <i>WinCC channel DLL active</i> and <i>WinCC ODK active</i> (see above) should always have the same status.

Option:	Time synchronization
Possible setting:	By own PC clock By local TIM with DCF77/GPS
Default:	By local TIM with DCF77/GPS
Explanation:	<ul style="list-style-type: none"> Internal on PC (uses own PC clock) The ST7cc PC uses the time of its own PC clock (own time transmitter). Any synchronization frame arriving from a time master TIM over the MPI bus is discarded. Redundant ST7cc: The setting <i>By own PC clock</i> is only practical, when an external time transmitter (for example <i>DCF77</i>) ensures that both ST7cc systems are exactly synchronized. MPI TIM (by local TIM with DCF77/GPS) The system clock of the PC is synchronized by a time master TIM over the MPI bus. The ST7cc PC can only be synchronized by the time master TIM if the synchronization time does not deviate by more than 60 seconds (positive or negative) from the time currently valid on the PC. Each time synchronization is rejected, a message to this effect is generated in the SINAUT log server. A pop-up window also appears in the foreground indicating the rejected synchronization. The clock in the ST7cc PC then has to be updated manually to the synchronization time. If it involves a redundant ST7cc system, both ST7cc PCs are synchronized by the time master TIM

Further notes of explanation:

Regardless of the way in which the time of the ST7cc PC is synchronized, any messages without time stamps will be stamped with the current PC time.

When using a redundant ST7cc configuration, the rule for local TIMs on the Ethernet bus is that one of the two ST7cc PCs is the preferred partner for time queries. If the preferred partner cannot be reached, the TIM queries the other ST7cc PC until the preferred partner is available again.

Option:	Close button
Possible setting:	active not active
Default:	active
Explanation:	<ul style="list-style-type: none"> • Setting active: The ST7cc server can be closed in the ST7cc server window using the relevant menu commands and buttons (4 options are available here for closing the server). • Setting not active: The ST7cc server window is open. The status information is displayed. The ST7cc server can no longer be closed in the ST7cc server window. All menu commands and buttons available for this function are disabled. See also section Startup behavior and start order (Page 289).

Option:	Run without window
Possible setting:	active not active
Default:	not active
Explanation:	<ul style="list-style-type: none"> • Setting active: The ST7cc server runs entirely in the background. It does not appear in the taskbar and cannot be displayed on the monitor. The ST7cc server cannot be exited. The SINAUT log window is not affected by this setting. The log window remains visible since it is started in a separate program. • Setting not active: The ST7cc server appears in the taskbar and can be displayed or hidden on the monitor.

Option:	PM-AQUA active
Possible setting:	active not active
Default:	not active
Explanation:	<ul style="list-style-type: none"> • Setting active: Data can be exported to PM-AQUA over a raw data channel. • Setting Not active: No data is exported to PM-AQUA. <p>For more detailed information on exporting data to PM-AQUA, refer to the section PM-AQUA link (Page 321).</p>

Option:	Acron
Possible setting:	Not active CSV archiving active CSV data logger active WinCC Tag Logging active
Default:	not active
Explanation:	<ul style="list-style-type: none"> • Setting not active: No data is exported to ACRON or any other archiving system. If you want to export data to ACRON, you have three possibilities: • Setting CSV archiving active: By linking an archive block with the name ACRON to an ST7cc variable, all the archive values of this variable will be written to a CSV file in the ACRON measured value format. There is no entry made in WinCC Tag Logging. The archive name ACRON does not, therefore, need to be configured in WinCC Tag Logging. • Setting CSV data logger active: With this setting, there is no need to link an archive block with the archive name ACRON to the variable being exported. All the data received from ST7cc as well as the system messages generated by the system typical, server typical, and subscriber typical are written to a CSV file in the ACRON measured value format. Since this is a global transfer of all data, the data not required by ACRON must be filtered out. • Setting WinCC Tag Logging active: If an archive block with the name ACRON is linked to an ST7cc variable, Acron takes the data from the WinCC Dbase archive. The archive name ACRON does not need to be configured in WinCC Tag Logging. <p>These export options are not restricted to ACRON. They can be used for any other archiving system that is compatible with these export interfaces and that accepts the ACRON measured value format.</p> <p>For more detailed information on exporting data to ACRON (or other archiving systems), refer to the section ACRON link (Page 327).</p>

Option:	Communication statistics
Possible setting:	active not active
Default:	not active
Explanation:	<ul style="list-style-type: none"> • Setting active: All variables of the system typical are supplied with values. Even the three variables for communication statistics available per subscriber typical are supplied with values. • Setting not active: None of the variables named above are supplied with values. <p>For more detailed information on these variables, refer to the section Picture typicals and faceplates for a station (Page 306).</p>

Option:	Server information
Possible setting:	active not active
Default:	active
Explanation:	<ul style="list-style-type: none"> • Setting active: All variables of the system typical are supplied with values. • Setting not active: The variables of the server typical are not supplied with values. This setting should be selected if the two default linked server typicals were both deleted. <p>For more detailed information on these variables, refer to the section Picture typical and faceplate for a server (Page 314).</p>

Option:	Create messages by WinCC
Possible setting:	active not active
Default:	active
Explanation:	<ul style="list-style-type: none"> • Setting not active: ST7cc enters every message, for which a message block was configured, directly into WinCC Alarm Logging taking into account the time stamp transferred with the message. • Setting active: ST7cc does not enter messages, for which a message block was configured, directly into WinCC Alarm Logging. The messages including their time stamps are transferred to the WinCC data manager. From this, WinCC generates the entries itself in WinCC Alarm Logging taking into account the time stamp transferred with the message. This setting is particularly suitable for a redundant ST7cc. A message acknowledgment is then automatically synchronized between the two WinCC systems. If the setting is not active, the acknowledgment must be made on both computers separately. <p>For a detailed description of the differences between creating messages in WinCC and creating messages in ST7cc, refer to section Message processing (Page 235).</p>

Option:	New faceplates
Possible setting:	active not active
Default:	not active
Explanation:	<ul style="list-style-type: none"> • Setting not active: The not active setting must be selected if some of the local TIMs do not have a firmware version V4.3 or higher. • Setting active: The active setting can be selected if all the local TIMs have a firmware version V4.3 or higher. The new faceplates contain more information compared with the old ones. <p>Make sure that you read section Configuring data with ST7cc Config (Page 155) Update scenarios.</p>

Option:	New faceplates
Option	Accelerated general request
Possible setting:	active not active
Default:	not active
Explanation:	<p>If ST7cc is restarted or a station can be reached again after a disruption, the ST7cc server automatically starts a general request (GR):</p> <ul style="list-style-type: none"> • When ST7cc starts up, a GR is sent to all connected stations. • When an individual station returns, the GR is sent only to this station. <p>With this parameter, you can specify whether ST7cc executes a GR initiated automatically as a standard or an accelerated general request.</p> <ul style="list-style-type: none"> • Setting not active: ST7cc starts an automatically initiated GR as a standard general request. • Setting active: ST7cc starts an automatically initiated GR as an accelerated general request. <p>Station TIMs can respond to an accelerated GR only under certain conditions. If these requirements are not met, then the TIM responds as with a standard GR.</p> <p>You will find a description of these conditions and the differences between the standard general request and an accelerated general request in section Standard general request and accelerated general request (Page 293).</p>

3.3.2 Project settings: File paths

In this dialog, you specify project-specific paths for some of the basic files and for buffers. When a project is created, the paths in this dialog are set to the default path configured for the project. Default names are also used for the files.

Here, changes are not normally necessary.

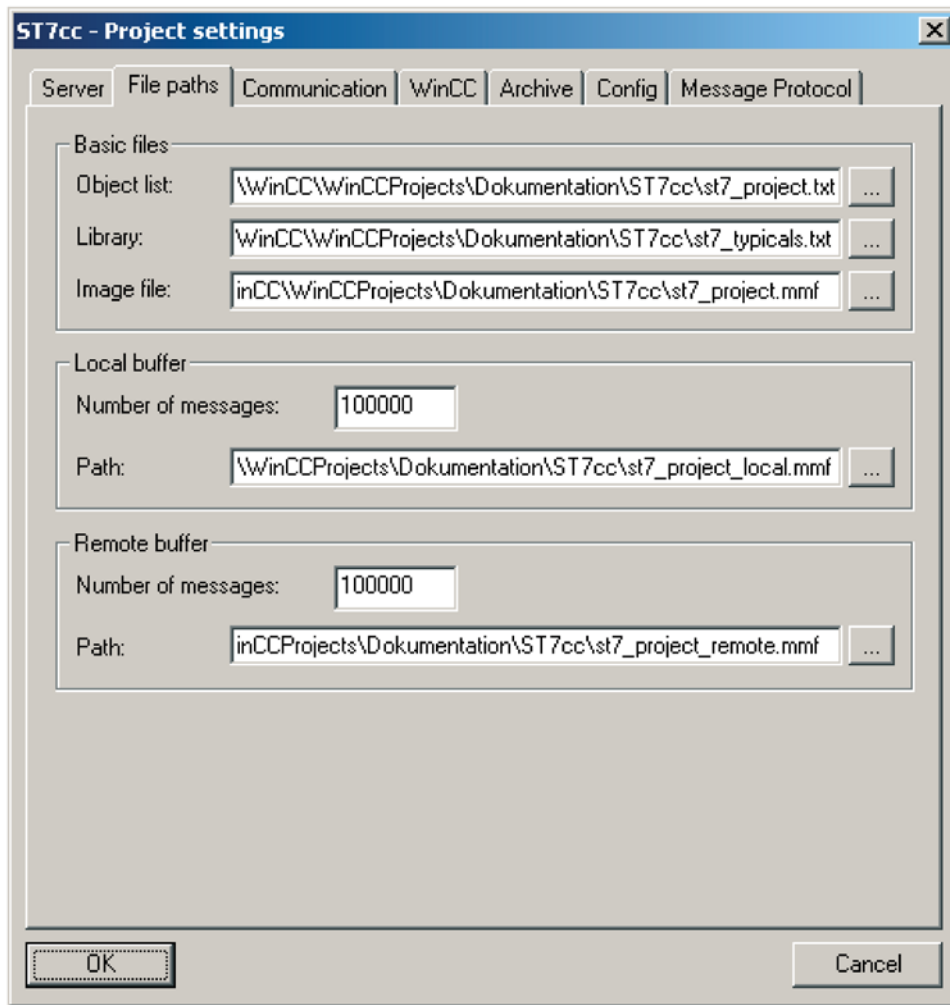


Figure 3-17 Tab for setting the file paths

Basic files

Parameter:	Object list
Default:	For Windows locale setting = German: <project path>\st7_project.txt For Windows locale setting ≠ German: <project path>\st7_project_english.txt
Explanation:	Name and path of the project file (object list). This file contains all the data configured with ST7cc Config for the project. Project and general settings are excluded from this. They are stored in other files.

Parameter:	Library
Default:	For Windows locale setting = German: <project path>\st7_typicals.txt For Windows locale setting ≠ "German": <project path>\st7_typicals_english.txt
Explanation:	Name and path of the library file.

Parameter:	Image file
Default:	<project path>\st7_project.mmf
Explanation:	Name and path of the process image file. The ST7cc process image is saved in this file regularly. The file is created in the specified directory only when the ST7cc server is started. The timebase within which the image is saved is set in the ST7cc – Project Settings dialog in the Server tab (see section Project settings: Server (Page 118)).

Local buffer

Parameter:	Number of messages
Default:	100 000
Permitted range of values:	1 000 – 3 000 000 process values
Explanation:	Size of the <i>local buffer</i> . This is where all the messages received from the SINAUT stations are temporarily stored if WinCC Runtime is deactivated. This buffer operates as a circulating buffer but it is filled only as long as WinCC Runtime is deactivated. Approximately 10 MB of hard disk is required for 100,000 messages.

Parameter:	Path
Default:	<project path>\st7_project_local.mmf
Explanation:	Name and path of the file for the <i>local</i> buffer. The file is created in the specified directory only when the ST7cc server is started.

Remote buffer

Parameter:	Number of messages
Default:	100 000
Permitted range of values:	1 000 – 3 000 000 process values
Explanation:	Size of the remote buffer. This is where all the messages received from the SINAUT stations are stored in case synchronization with the redundant partner becomes necessary. The received messages are always stored here regardless of whether there is actually a redundancy failure. Following a failure, the remote redundancy partner can be updated again from this buffer. This buffer operates as a circulating buffer and it is always 100% full following the initial fill phase. Approximately 10 MB of hard disk is required for 100,000 messages.

Parameter:	Path
Default:	<project path>\st7_project_remote.mmf
Explanation:	Name and path of the <i>remote</i> buffer. The file is created in the specified directory only when the ST7cc server is started.

Note

If ST7cc is not redundant, the default parameters for Remote buffer can be left unchanged. The buffer is created only when ST7cc redundancy is activated and when a suitable license is available (see section Project settings: Server (Page 118)).

3.3.3 Project settings: Communication

In this dialog, you enter the SINAUT subscriber number of the ST7cc PC itself and of the redundant partner (if it exists).

The subscriber number of the ST7cc PC can only be entered if the computer name has been entered in the Edit > Global settings in the "Computer" tab. See section Global settings: Computer (Page 149)

In this dialog, you also specify the SINAUT subscriber numbers, the local IDs and the application access points of the TIMs connected locally over the MPI bus or Ethernet to the ST7cc PC.

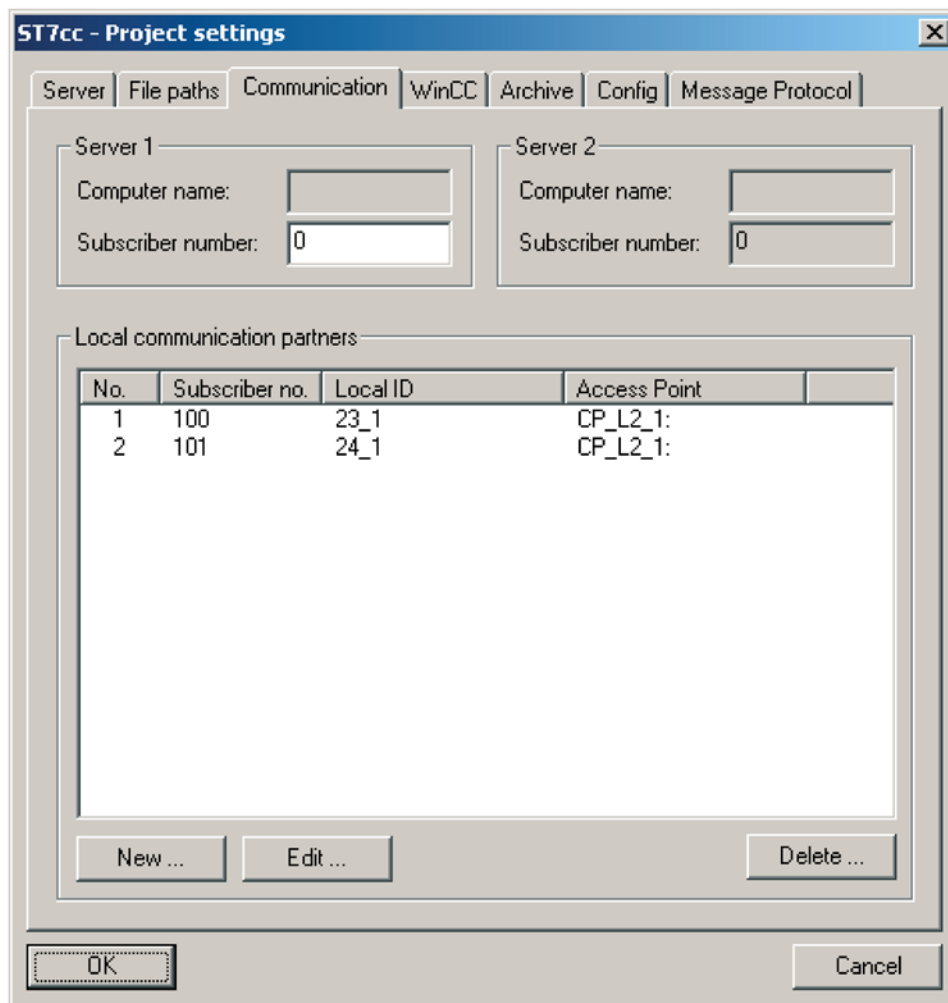


Figure 3-18 Tab for setting the communication parameters of the ST7cc server

Note

If you make changes in this dialog for a project that is already activated, you must repeat the activation to activate the settings changed here. Follow the steps outlined below:

1. Select *Edit > Global Settings* and then the *Project* tab.
2. Click the *Activate current project for ST7cc Runtime* button to repeat the project activation.

Server 1

Parameter:	Computer name
Explanation:	The computer name of <i>Server 1</i> is automatically entered from the <i>Computer</i> tab of the <i>Global Settings</i> dialog. The computer name cannot be changed here.

Parameter:	Subscriber number
Explanation:	The SINAUT subscriber number that is configured in the SINAUT project for the ST7cc PC (with the name displayed in <i>Computer name</i>).

Server 2

Information for Server 2 is relevant only if you are using a redundant ST7cc system.

Parameter:	Computer name
Explanation:	The computer name of <i>Server 2</i> is automatically entered from the <i>Computer</i> tab of the <i>Global Settings</i> dialog. The computer name cannot be changed here.

Parameter:	Subscriber number
Explanation:	The SINAUT subscriber number that is configured in the SINAUT project for the redundant ST7cc PC (with the name displayed in <i>Server 2 / Computer name</i>).

Local communication partners

Local communication partners relates only to TIMs connected to the ST7cc PC over the local MPI bus or Ethernet (therefore also known as 'local TIMs'). ST7cc communicates with the SINAUT stations over these TIMs. CPUs connected locally to ST7cc are not included as local communication partners because they cannot be configured as SINAUT stations.

In the Local communication partners area (see figure), make the following settings:

Enter the local TIM in the list

1. Click on the New... button, the Add Communication Partner dialog appears (see figure).

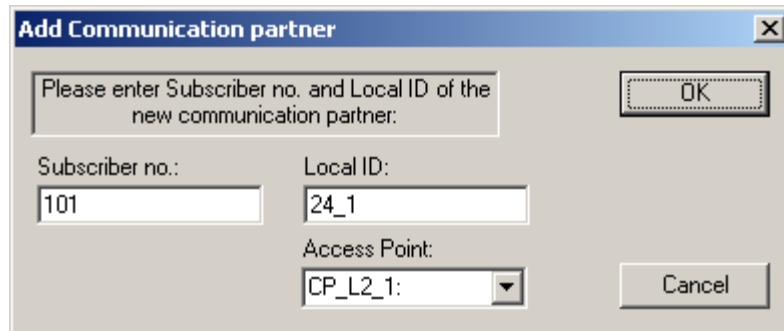


Figure 3-19 Dialog for configuring the data of a local TIM

2. Enter the following parameters in the Add Communication Partner dialog:
 - The *subscriber number*. This is the SINAUT subscriber number of the local TIM.
 - The *local ID*. The local ID is the ID of the S7 connection configured in NetPro between ST7cc and the local TIM (see section Configuring an S7 connection between local TIMs and ST7cc (Page 55)).
 - The *access point*. Applications access the communication module using the name of an access point (see section Setting access points for the SINAUT PC (Page 81)).

Specify the access point over which ST7cc can access the local TIM.

The access points set up and stored in the project are displayed when the dialog opens. There is no check as to whether the set access point actually exists on the PC/PG. This means that the user can also configure on a computer that is not the target machine.

If users require a different access point, or if no access points have yet been set up, they can enter the name of the access point here. The access point must then be set up later.

3. Close the dialog with *OK*.
The added TIM is now included in the list of local communication partners. The *No.* column only contains a consecutive number. It is automatically incremented for each new TIM added.
4. Now enter all local TIMs in the list of communications partners.

Editing data of a TIM already entered in the list

To edit the data of a TIM already in the list, follow the steps below:

1. Select the consecutive number of the TIM.
2. Click the *Continue* button.

The same dialog appears as for a *New...* entry, however, this time it contains the previously valid data that can now be edited.

Deleting a TIM from the Local Communication Partner list

To delete a TIM, follow the steps below:

1. Click on the consecutive number of the TIM you want to delete.
2. Click the *Delete* button.

3.3.4 Project settings: WinCC

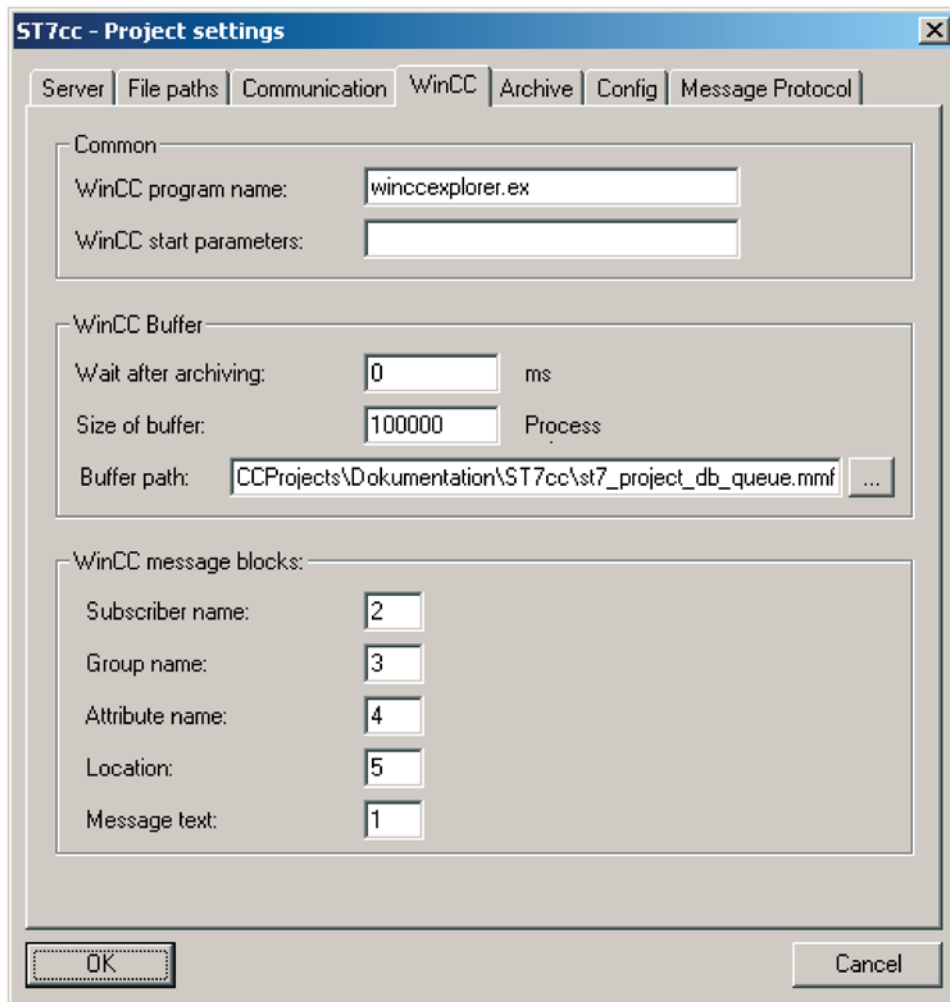


Figure 3-20 Tab for the WinCC project settings

General

In this section, you can specify whether or not WinCC Runtime should start automatically when ST7cc Runtime is started. With this setting, WinCC Runtime will then always start at the correct time; in other words after the ST7cc server has started. Otherwise the WinCC process pictures cannot be updated.

Parameter:	Program name WinCC
Possible setting:	No or incomplete entry WinCCExplorer.exe AutostartRT.exe
Default:	AutostartRT.exe
Explanation:	<ul style="list-style-type: none"> No entry or incomplete entry (for example the default setting "AutostartRT.ex"): WinCC Runtime is not started automatically. Setting winccexplorer.exe: WinCC is started automatically. WinCC starts in the mode that was active when the program was last closed: Configuration or Runtime mode Setting AutostartRT.exe: WinCC Runtime is started automatically but without WinCC Explorer. If you use this setting, the start parameters listed below must be specified.

Parameter:	WinCC start parameters
Possible setting:	No entry WinCC project with path and two additional start parameters: D:\WinCCProjects\ST7ccRed\ST7ccRed.mcp /Activ:yes /Lang:Deu (German language project) D:\WinCCProjects\ST7ccRed\ST7ccRed.mcp /Activ:yes /Lang:Eng (English language project)
Default:	C:\WinCCProjects\project.mcp /Activ:yes /Lang:Deu (German language project) C:\WinCCProjects\project.mcp /Activ:yes /Lang:Eng (English language project)
Explanation:	If AutostartRT.exe is used, the start parameters listed above must be specified.

WinCC buffer

Process data from the ST7cc server is transferred to WinCC over a buffer to compensate the different processing speeds of the ST7cc server and WinCC. You can enter the parameters for this buffer here or use the default entries.

Parameter:	Wait after archiving
Default:	0 [ms]
Explanation:	This value should not be changed.

Parameter:	Size of buffer
Default:	100 000
Permitted range of values:	1 000 – 3 000 000 process values
Explanation:	The length of the buffer is specified by the maximum number of process values that can be stored in the buffer.

Parameter:	Buffer path
Default:	<project path>\st7_project_db_queue.mmf
Explanation:	Name and path of the WinCC buffer.

WinCC message blocks

During configuration of the message system in WinCC, user text blocks can be deleted or added from a predefined list to be able to display additional static texts in a message. User texts (static additional texts) include, for example the plant designation, point of error etc.

The maximum length of a user text block is 254 characters. They are displayed, however, in one line and are limited to the screen width. Longer text is truncated in the display and cannot be shown.

If you create a message block for decoding a SINAUT object, the following texts must be entered:

- The message text
- The subscriber name
- The group name
- The attribute name
- The location

In the WinCC tab of the ST7cc Project Settings dialog, you specify which user text block the information will be assigned to.

Note

The defaults of the user text blocks differ in WinCC and PCS 7. With the WinCC User Text Blocks or PCS 7 User Text Blocks buttons, you can select the defaults you want to use.

Parameter:	Subscriber name
Possible setting:	1-10
Default WinCC/PCS7:	2 / 1
Explanation:	The subscriber name configured for a message in ST7cc is entered in the WinCC user text block with the number specified here (for example 2).

Parameter:	Group name
Possible setting:	1-10
Default WinCC/PCS7:	3 / 4
Explanation:	The group name configured for a message in ST7cc is entered in the WinCC user text block with the number specified here (for example 3).

Parameter:	Variable name
Possible setting:	1-10
Default WinCC/PCS7:	4 / 5
Explanation:	The variable name configured for a message in ST7cc is entered in the WinCC user text block with the number specified here (for example 4).

Parameter:	Location
Possible setting:	1-10
Default WinCC/PCS7:	5 / 2
Explanation:	The location configured for a message in ST7cc is entered in the WinCC user text block with the number specified here (for example 5).

Parameter:	Message text
Possible setting:	1-10
Default WinCC/PCS7:	1 / 3
Explanation:	The message text configured for a message in ST7cc is entered in the WinCC user text block with the number specified here (for example 1).

3.3.5 Project settings: Archive

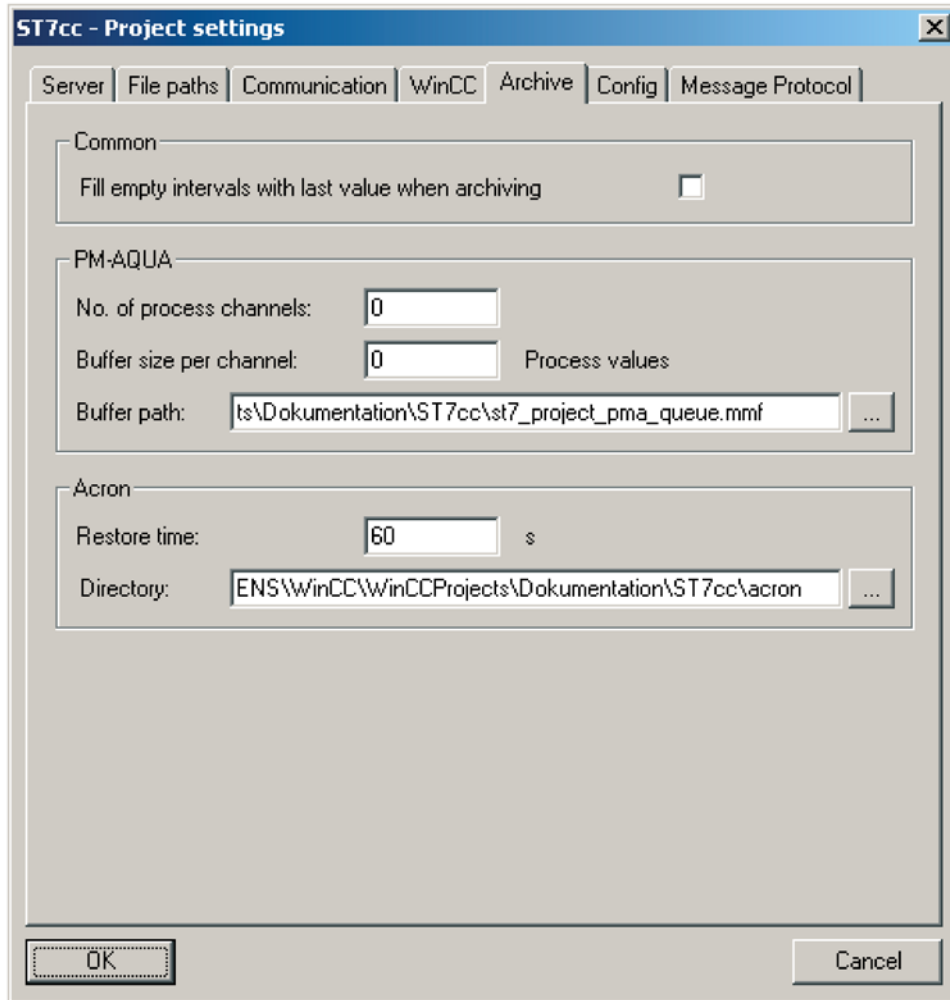


Figure 3-21 Archive tab in the Project settings dialog

In this tab, you make the settings required for archiving the process data.

The settings must be made to allow process data to be forwarded to the PM-AQUA or ACRON archiving systems.

General

Parameter:	Fill empty intervals with last value when archiving
Possible setting:	active not active
Default	not active
Explanation:	Setting active: If the value is missing for a compression interval, ST7cc fills this interval slot with the last valid value.

Parameter:	Use WinCC quality flags
Possible setting:	active not active
Default:	not active
Explanation:	Setting active: To indicate the quality of the process values, ST7cc does not use its own but rather the WinCC system (see also section Quality code of the WinCC tags supplied with values by ST7cc (Page 285))

PM-AQUA

Parameter:	Number of process data channels
Possible setting:	0-10 [data channels]
Default:	0
Explanation:	If you want to use the raw data channel for PM-AQUA, enter the number of available process data channels here. If you enter the value 0, no process data channel is used.

Parameter:	Buffer size per channel
Process values:	1 000 to 3 000 000
Default:	0
Explanation:	If you want to use the raw data channel for PM-AQUA, enter the maximum number of process values that can be stored in the interim buffer. If you enter the value 0, no process data channel is used.

Parameter:	Archive buffer
Default:	<project path>\st7_project_pma_queue.mmf
Explanation:	Name and path of the interim buffer for PM-AQUA.

ACRON

Parameter:	Restore time
Possible setting:	At least 60 seconds
Default:	60
Explanation:	If the ACRON CSV interface is used, you can set the restore cycle of the individual CSV files.

Parameter:	Directory
Default	<project path>\
Explanation:	Directory in which the CSV files will be stored.

3.3.6 Project settings: Config

In the Config tab (see figure), you make several settings required by the ST7cc Config tool. These include, for example, the format of the automatically generated WinCC message numbers as well as the texts used as defaults by ST7cc Config when a new object is created (subscriber, variable, etc).

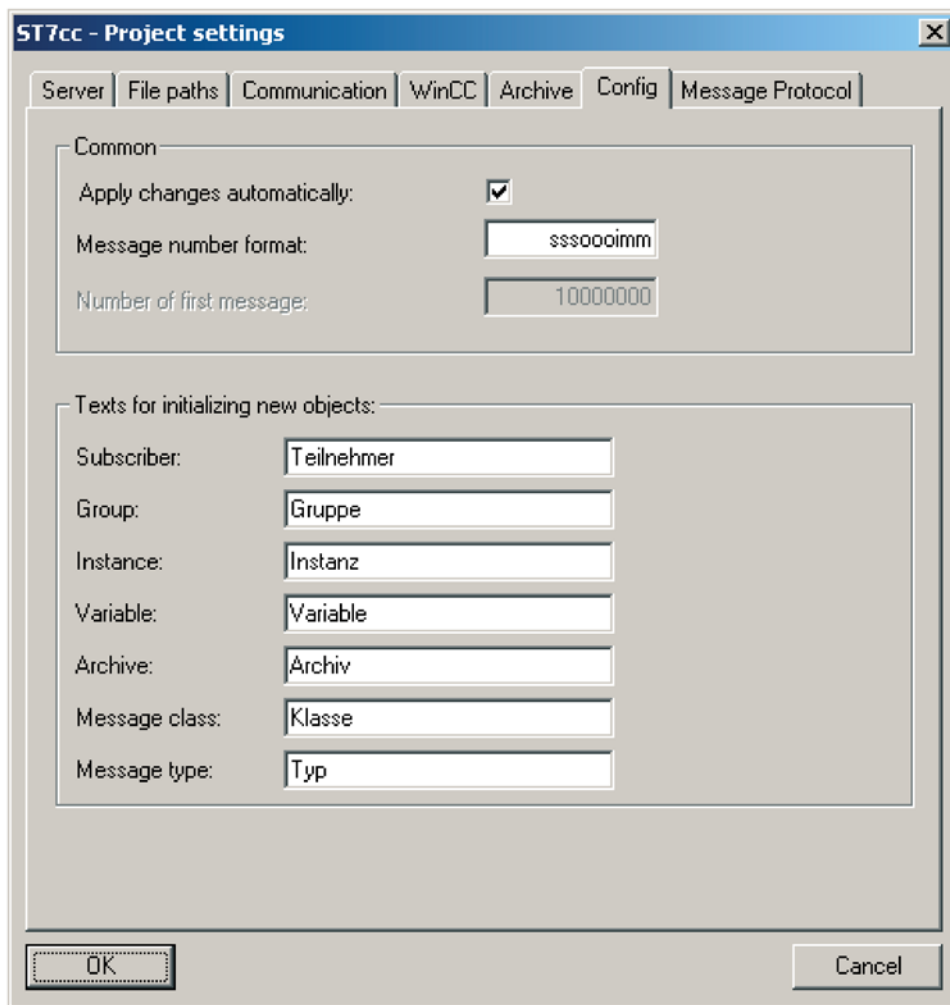


Figure 3-22 Dialog for project-specific settings of ST7cc Config

General

Parameter:	Apply changes automatically
Possible setting:	active not active
Default:	active
Explanation:	<ul style="list-style-type: none"> • Setting <i>active</i>: Changes to parameter settings are applied automatically. • Setting <i>not active</i>: When you change to another screen, changes you have made are lost again unless you click the "Apply changes" button.

Note

This *Apply changes automatically* option should always be activated during the configuration phase.

Message numbers:

With ST7cc version V2.7, in addition to the existing method of generating message numbers, an additional "new" method has also been created to accommodate the defaults of the WinCC and PCS 7 optional packages (see below). With the Change Method button, you can switch between the two methods. If you edit an existing ST7cc project with the new version, the default old is selected automatically, otherwise the default is new.

For more detailed information on the topic of WinCC message numbers, refer to section Message processing (Page 235).

Old method:

The previous, old method was to generate the message number based on the subscriber number (station number), object number, typical instance number and the consecutive number of the message block. The disadvantage of this method is that a very wide band of numbers is created. When using other WinCC or PCS 7 optional packages, this can lead to the band overlapping other bands of numbers.

Note

The total number of places (sss000imm = 9 places) must not be changed.

3.3 Project settings

Parameter:	Message number format
Possible setting:	ssssoommm ssssoommm ssssooimm ssssooimm
Default:	ssssooimm
Explanation:	To make sure that WinCC message numbers are unique throughout a project, certain structures have been approved (see below)

Table 3- 2 ssssoommm

Structure:	Description	Possible number range
sss	Subscriber number (station number)	11-4095 Note: 1 is reserved for WinCC.
ooo	Object number	1-999
mm	Consecutive number of the message block	1-999

Table 3- 3 ssssoommm

Structure:	Description	Possible number range
sss	Subscriber number (station number)	2-499 Note: 1 is reserved for WinCC.
ooo	Object number	1-999
mmm	Consecutive number of the message block	1-999

Note

Make sure that if the message number format "ssssoommm" is used, no typical instances are used.

Table 3- 4 ssssooimm

Structure	Description	Possible number range
sss	Subscriber number (station number)	2-4095 Note: 1 is reserved for WinCC.
oo	Object number	1-99
i	Typical instance number	1-9
mm	Consecutive number of the message block	1-99

Table 3- 5 sssoooimm

Structure	Description	Possible number range
sss	Subscriber number (station number)	2-499 Note: 1 is reserved for WinCC.
ooo	Object number	1-999
i	Typical instance number	1-9
mm	Consecutive number of the message block	1-99

New method

With the new method, you can set an offset (base number) and the message numbers are then generated consecutively starting at this number. There is then no longer a structured assignment of message numbers as with the "old" method.

If the old method is set as default, you can change to the new method as follows:

1. Click the Change Method button.
2. Confirm the change.

With this confirmation, the old message numbers of an existing ST7cc project are changed to the new message numbers.

With the Reorganize Message Numbers button, you can reorganize the message numbers following extensive reconfiguration or on completion of commissioning, for example to achieve consecutive numbering in your object list.

The offset is not included in the message numbers in the ST7cc object list. The set offset is only added to the message number in the ST7cc object list when generating the messages for WinCC Alarm Logging (in other words, the message number in WinCC Alarm Logging is formed from the "message number in the object list + offset -1").

As a result, it is possible to change the offset at any time in the ST7cc configuration without influencing the message number in the ST7cc object list.

Possible range of values for objects: 10 000-99 999

Note

Before the newly generated or reorganized message numbers take effect in your WinCC project, you will need to delete the old message numbers generated by ST7cc in WinCC and create the new message numbers in WinCC.

Texts for the initialization of new objects

When you create a new subscriber, variable, archive block, etc., ST7cc Config enters a default that can be modified by the user. With the parameters under *Texts for initializing new objects*, you can specify these defaults. The entries can be changed at any time during the project configuration. The changed default settings only affect objects created after the change is made. A default text is entered in each of the boxes.

3.3.7 Project settings: Message protocol

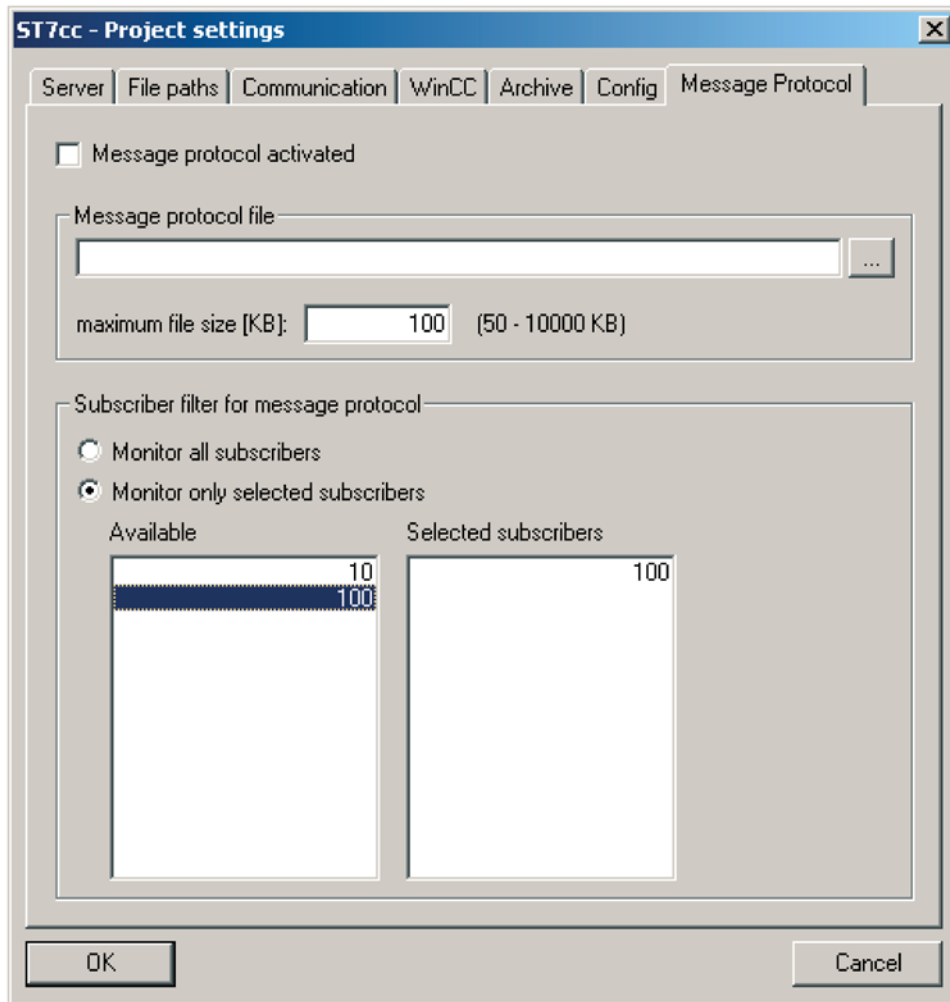


Figure 3-23 Project settings tab for ST7cc Config

In the Message Trace tab, you can activate message logging for all or selected subscribers with the Message trace activated option (see figure). The acquired messages are then stored in a file for further evaluation.

If the message trace is activated at this point, the trace starts when the server starts up/restarts. Messages continue to be logged until this is deactivated by the operator. If the server is then restarted, the message trace is reactivated and must, if required, be deactivated again by the operator. If the message trace is no longer required during startup; in other words, you want to deactivate it permanently, this is possible only in this dialog. It is nevertheless possible to activate or deactivate the message trace during operation. You should then use the message trace function of the ST7cc servers (see section Diagnostics: Message protocol of the ST7cc server (Page 304)).

To create a message protocol or log, follow the steps below:

1. Set the Message protocol activated option in the Message Protocol tab.
2. Click the button (...) beside "Message protocol file".

3. 5. In the dialog that now opens, select the file and path (default: ...Siemens\Step7\STTemp\) in which you want to save the message protocol (log) and click Open.
4. Under maximum file size, enter the maximum size of the protocol file. The optimum size is largely dependent on how many subscribers are being logged and the size the messages. As an average value, 35 bytes per message can be assumed. This results in a selectable number of messages to be logged of approximately 1,400 (50 KB) – 280,000 (10,000 KB).
If the file has reached the set value, it is saved as filename.old and a new file with the name specified above is created. When this new file reaches the maximum size, it in turn is saved as "Filename.old". The information in the first protocol (log) file is lost.
5. Select the subscribers to be logged. Select either the *Monitor all subscribers* option or the select the subscribers you want to monitor with the *Monitor only selected subscribers* option. With this option, the subscribers that will be monitored appear below *Selected subscribers*.
6. Click *OK*.
7. Save your project.

A protocol file can be opened with SINAUT Diagnostics and Service. For more detailed information, refer to the SINAUT ST7 manual.

Note

Activating the message protocol (log) affects the throughput of the ST7cc server. If the message protocol (log) is no longer required during startup, the function can be disabled in this dialog so that it is not activated automatically during every restart.

3.4 Global settings

There are two ways of opening the *Global settings* dialog:

With *Edit > Global Settings*

Using the *F3* key

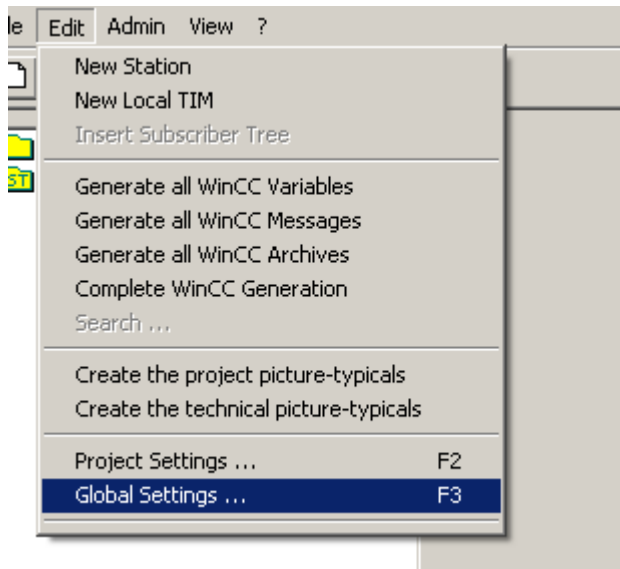


Figure 3-24 Selecting Global Settings from the Edit menu

The following sections describe the possible settings in the individual tabs.

Note

All global settings (computer settings) are saved project independent in various files in the directory "<Drive>\Siemens\ST7cc\base".

With other Windows operating systems (for example Windows Server 2008), the path can also be as follows: "<Drive> > Program Data > Siemens > ST7cc > base". Remember that the folder must be shared using the folder options so that it is visible in the Explorer.

3.4.1 Global settings: Computer

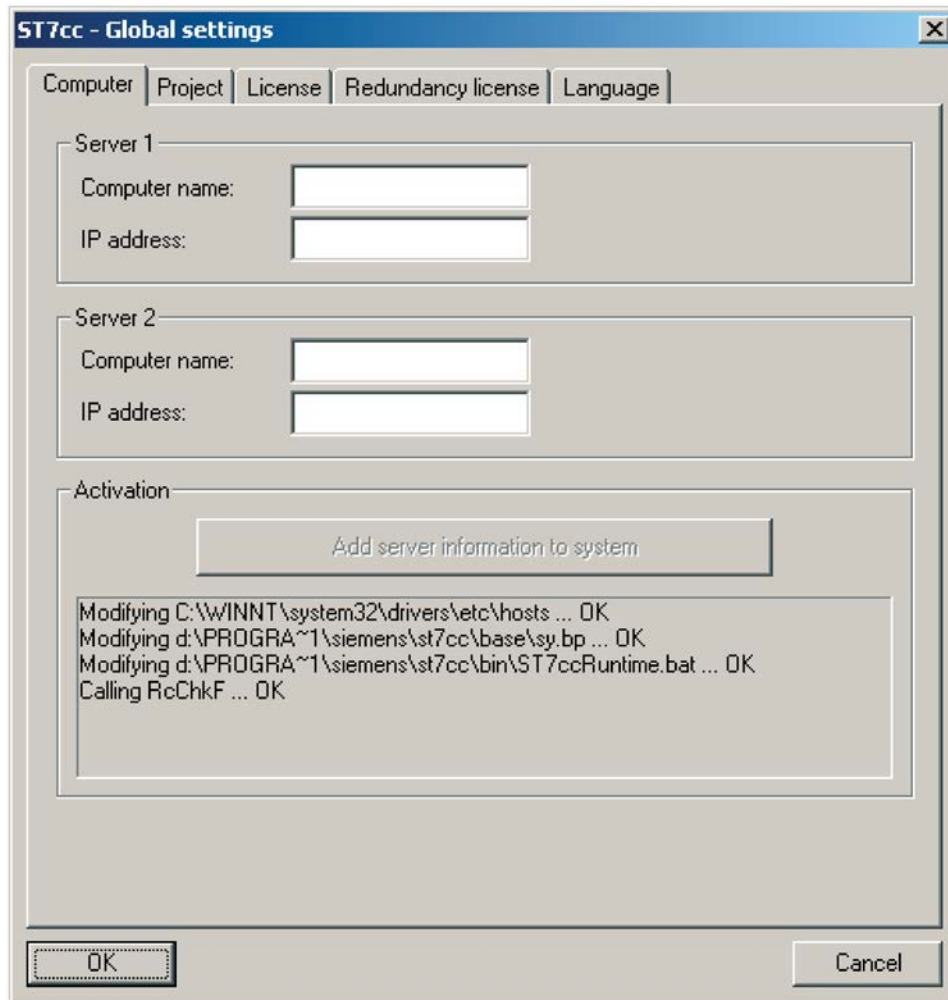


Figure 3-25 Global settings dialog, Computer tab

In the *Computer* tab, you enter the name and the IP address of the server and the redundant server (if it exists) into the system.

If you do not know the computer name and corresponding IP address of your PC, you can find these as follows:

1. Select *Start > Run*.
The *Run* dialog opens.
2. In the Open box, enter *cmd*.
3. Click *OK*.
The command prompt is opened.

4. In the command prompt, type:
`ipconfig /all`
You will then see all the information you require.
Repeat these steps on the redundant server (if it exists).
5. Under *Server 1*, enter the name of the server (host name) in the *Computer name* box.
6. Under *Server 1*, enter the *IP address* of the server in the IP address box.
7. Under *Server 2*, enter the computer name and IP address of the redundant server (if it exists).
8. Click the Add server information to system button.

Note

This button is only available when your entries match the actual computer names.

Note

After clicking the button, please check that OK is shown at the end of the message line for all the modifications activated by this action.

Note

If you expand your system later to set up a redundant system, do not forget to enter and activate the computer name and the IP address of the redundant computer here.

Make sure that the "Computer name" and the relevant "IP address" for server 1 and server 2 are identically configured on both computers and are entered in the same order.

3.4.2 Global settings: Project

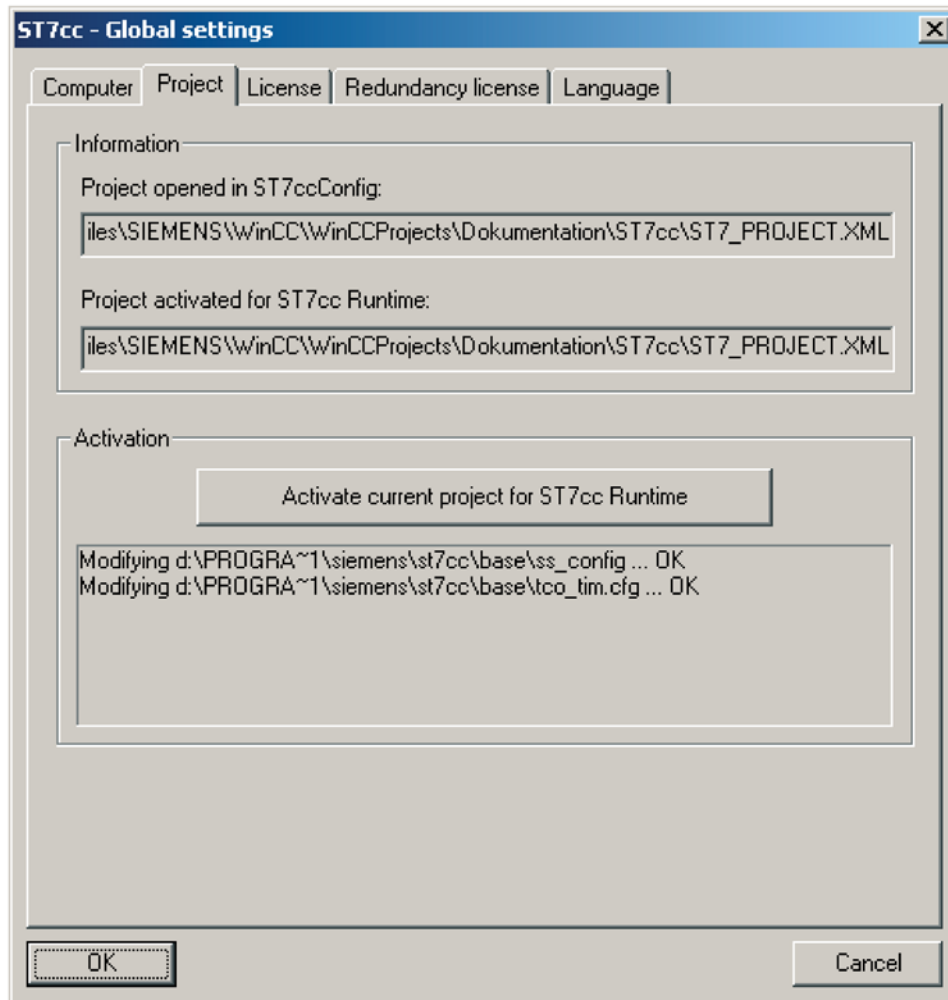


Figure 3-26 Tab for the global settings of the project

The *Project* tab allows you to activate projects created with ST7cc Config for ST7cc Runtime, i.e., when you start ST7cc Runtime (also starting several other programs including the ST7sc Server), it will read in the parameters and settings contained in the specified project file.

If you are managing several projects on your PC (for example in an engineering office), you can use this dialog to activate a specific project for online testing on your PC.

1. Click the *Activate current project for ST7cc Runtime* button.

The project currently opened in ST7cc Config is activated for ST7cc Runtime and started the next time the ST7cc server starts up.

Note

After clicking the button, please check that OK is shown at the end of the message line for all the modifications activated by this action (see figure).

Note

In some rare situations, it is necessary to repeat the project activation here in this tab:

- When you have made changes in the Communication tab of the Project Settings dialog.
 - If you want to activate a different project (engineering office).
 - If you have activated or deactivated the Configuration is redundant option in the Server tab of the Project settings dialog.
-

3.4.3 Global settings: Language

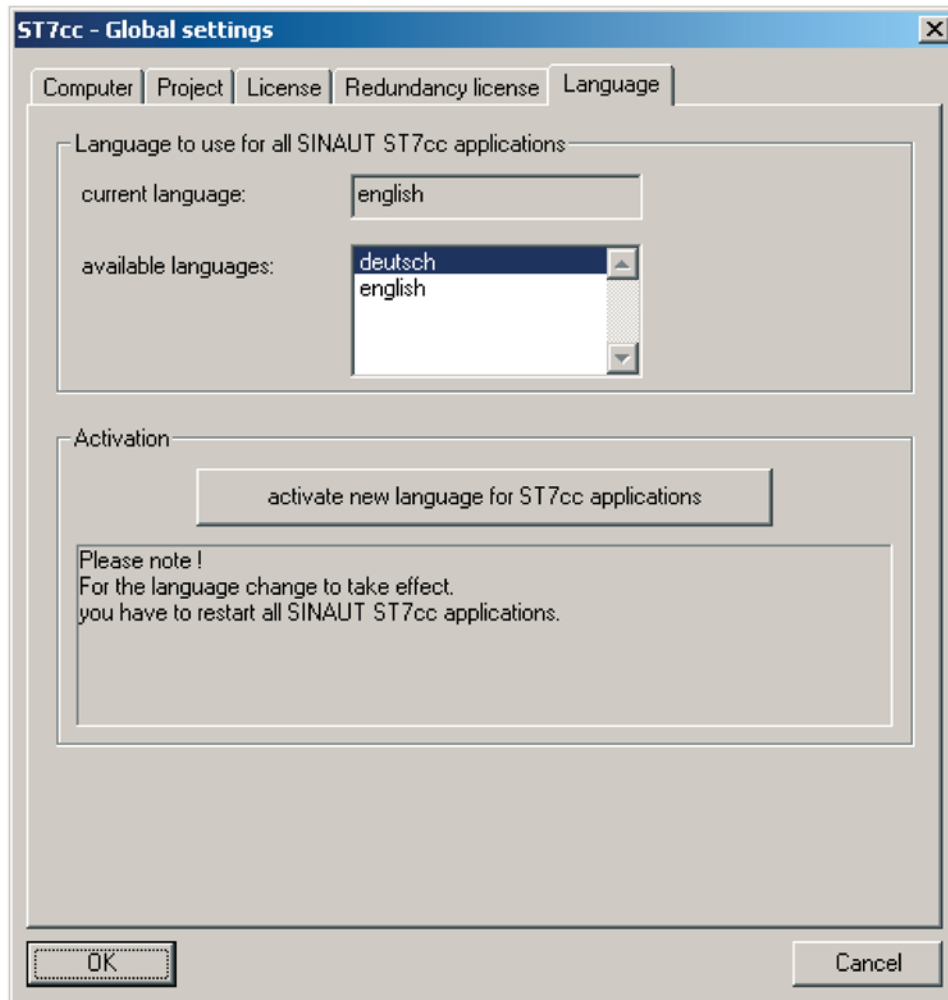


Figure 3-27 Tab for setting the ST7cc language

In the Language tab (see figure), you can set the language of all ST7cc applications. To set the language of ST7cc Config, follow the steps outlined below:

1. In the *available languages* box, select the language you want to set for the ST7cc applications.
2. Click activate new language for ST7cc applications.
3. Restart ST7cc Config so that the modification takes effect.

If you want to use an English HMI on an Asiatic Windows operating system, for example for the Chinese market, select the language setting English for Asia because the English characters are displayed in a non-proportional font in this system environment.

Configuring data with ST7cc Config

4.1 What is ST7cc Config?

The ST7cc Config component is configuration software that is fully compatible with WinCC. The figure shows the logical position of ST7cc Config in the SINAUT world.

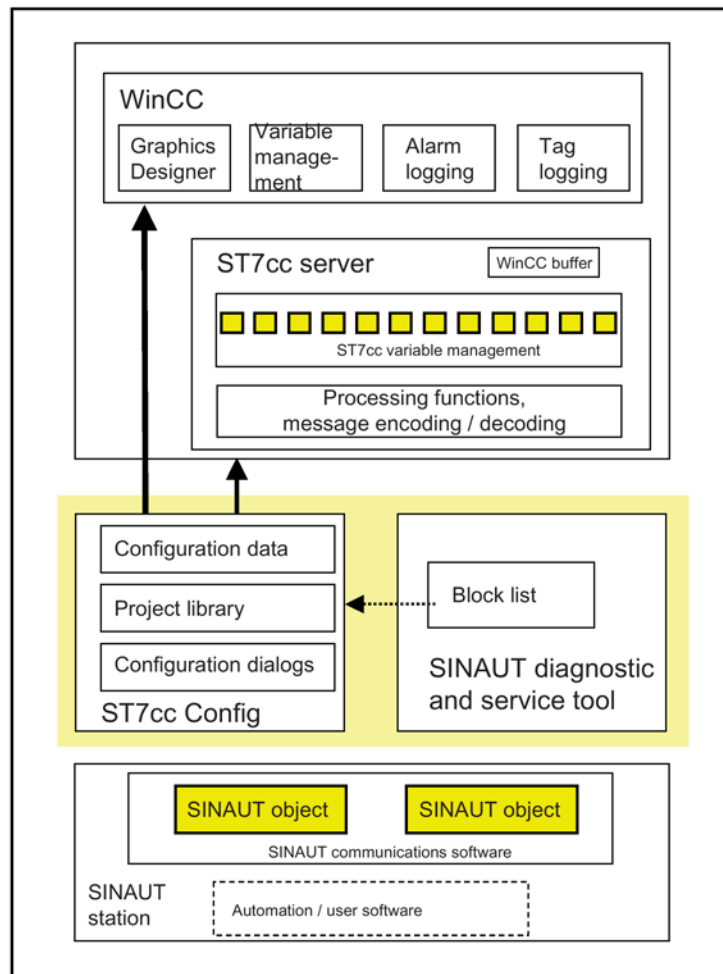


Figure 4-1 Logical position of ST7cc Config in the SINAUT world

The *ST7cc Config* component is the configuration tool that allows the following functions:

- Mapping of the ST7 data management (set of SINAUT objects) to ST7cc variables for the monitoring and control direction.
- Parameter assignment of processing that can be assigned to the ST7cc variables. Processing takes place in ST7cc and also in WinCC.
- The generation of the WinCC tag management, in which the parameter assignment for the WinCC processing is supplied to the WinCC components via the ODK interface.

This property, originating in the ST7 object world, allows efficient, cross-component configuration of the task in hand.

ST7cc Config is supported by the SINAUT Diagnostics and Service Tool. The TD7 block structure analysis allows the user to list the SINAUT objects of the ST7 project or individual stations and to transfer them to ST7cc Config. The transfer of the block list is not automatic and must be done by the user with the support of a dialog. Compare section SINAUT TD7 block structure in ST7cc Config (Page 259).

4.2 What does configuring mean?

Overview

Configuring with ST7cc Config means

- mapping the data of the SINAUT objects configured in the stations and
- mapping the most important status information of the SINAUT subscribers to ST7cc variables and setting the parameters for their processing in ST7cc and in WinCC.

To allow this mapping, the user must set up the SINAUT subscribers in ST7cc Config and create a decoding or coding for each SINAUT object. For ST7cc, subscribers are the CPUs in the stations and the TIMs connected to the ST7cc PC via the local MPI bus or Ethernet, but not the TIMs in the stations.

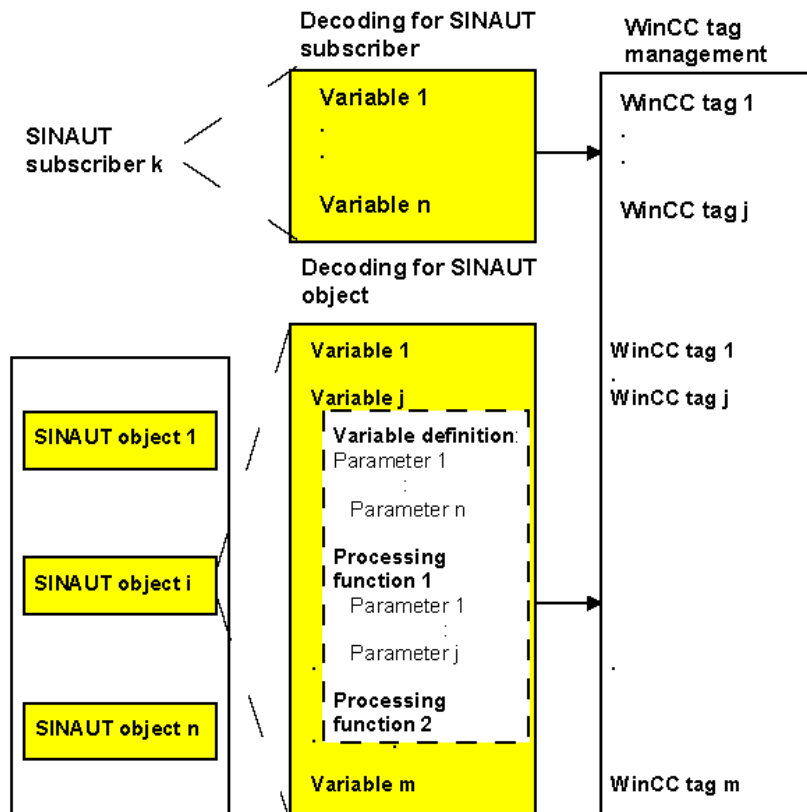


Figure 4-2 Relationship between SINAUT object, decoding, ST7cc and WinCC tag

In simple terms, "configuration" means creating decodings or codings. Only the term "decoding" will be used from now on.

Result

Generating the WinCC parameter assignments for ST7cc variables and the processing functions; in other words:

- Generating WinCC tags
- Generating WinCC messages
- Generating WinCC archives
- Creating the picture typicals of the ST7 subscribers

Based on the configuration data, the ST7cc server can decode the messages arriving from the SINAUT stations; in other words, map their data to ST7cc variables.

In the control direction (command and setpoint output), the ST7cc server can map the content of an ST7cc variable to the data structure of the SINAUT target object, create the SINAUT message for communication and transfer this to the TIM responsible for transmission.

4.3 Background knowledge on configuring

Note

Section Configuring (Page 178) describes the activity of configuring. Users have several options for decoding a SINAUT object.

To avoid needing to explain basic relationships several times in the descriptions of the various decoding options, the most important terms, relationships and dependencies are described below.

4.3.1 SINAUT subscriber

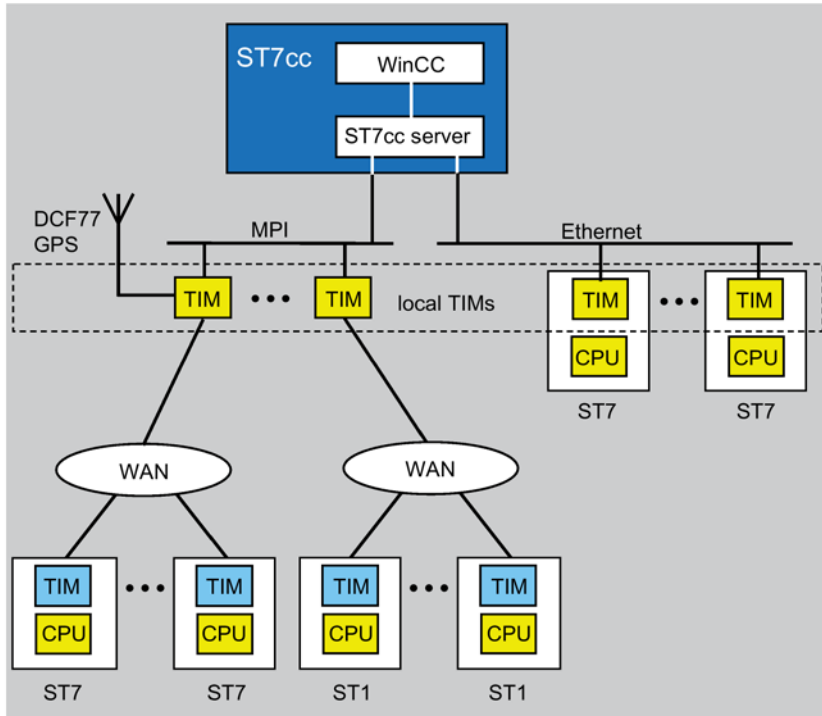


Figure 4-3 SINAUT ST7cc with connected ST7 and ST1 stations

For ST7cc, only the following subscribers are relevant:

- The CPUs in the stations.
- Local TIMs; in other words, the TIMs connected to the ST7cc PC locally over the MPI bus or Ethernet.

In ST7cc Config, ST7cc variables are created for each subscriber to contain their most important status information and to make them available to the WinCC tag management and other WinCC processing. Which status information is mapped on ST7cc variables is explained in detail in the section System typicals (Page 184).

4.3.2 SINAUT object

Overview

The term SINAUT object is used as a generic term for ST7 and ST1 objects.

There is a SINAUT object in the CPU of the station (when the TD7 software runs on the CPU) or the TIM (when the TD7 software runs on the TIM). Here, the SINAUT data point typicals, such as Bin04B, Ana04W, Cmd01B etc. are used for configuration.

A SINAUT object contains the data of one or more process variables, such as analog values, commands, calculated values, status information on motors, sliders etc.

On the station, type-specific processing and change checks are assigned to a SINAUT object to minimize the communication traffic in the WAN. Type-specific processing includes, for example, threshold value objects or calculating a mean value with the object type Ana04W. The change check is designed so that a message is generated only when the object data has changed compared with the last time its value was transferred or when the type-specific processing enables generation of a message because the object data is "worth" transferring.

Object identification

With its

- Subscriber number and
- Object number

each SINAUT object can be uniquely identified. The user specifies the object number during configuration of the SINAUT objects. The subscriber number and the object number together form the ID (identification) of each decoding. This allows a unique assignment between the SINAUT object and decoding.

Data subarea / variable

During ST7cc configuration, the SINAUT objects are considered primarily as information carriers. The ST7cc configuration engineer examines the user data area of an ST7 object that can include several information units and maps the information units to variables. The subarea of the object data area that represents an information unit is known as a data subarea. A data area can be mapped to several variables if the information unit needs to be processed more than once (see the section Configuring (Page 178)).

4.3.3 SINAUT object types

Overview

The SINAUT ST7 system manual describes all the ST7 and ST1 object types (typicals) along with their functions and data structures. Below, only the data objects for which decoding examples (known as object templates) are stored in the project library are described.

Note

Only the object types described below can be used in a station for communication with ST7cc. In particular, note the ending `_S` (for send object) and `_R` (for receive object) in the object names.

Object type Bin04B_S

The object type Bin04B_S contains four bytes of binary information in the stations such as messages, alarms etc. The object type has a data area of 32 bits.

Bin04B_S	Byte 1	Byte 2	Byte 3	Byte 4
Byte index	0	1	2	3
Bit index	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0
Length in bits	Variable between 1 and 32, length of the area in bits beginning with byte index and bit index in the direction of ascending indexes.			

The first bit of a Bin04B (least significant bit of the first byte) for example, is identified by byte index 0, bit index 0 and bit length 1.

The information contained in the 32-bit data area can differ widely and is alone the responsibility of the ST7 user (see figure).

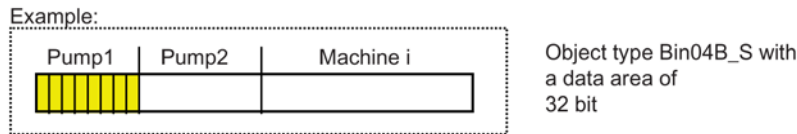


Figure 4-4 Possible divisions of a 32-bit object data area

Example:

In this case, the data area of 32 bits is divided into three information units. There are two information units with eight bits and one information unit with 16 bits that represent the status of pumps 1 and 2 and another piece of equipment.

Object type Ana04W_S

The object type Ana04W_S contains a 4-word (each 16 bits) data area of the SINAUT ST7 analog value processing. The bit assignment and the functionality of the object type are described in the ST7 system manual.

Ana04W_S	Word 1	Word 2	Word 3	Word 4
Byte index	0	2	4	6
Bit index	0	0	0	0
Length in bits	16	16	16	16

The four words of object type Ana04W are addressed by byte index 0, 2, 4 and 6, bit index 0 and length 16 (bits).

Object type Cnt01D_S

The object type Cnt01D_S contains a data area of 1 double word (32 bits) of the SINAUT ST7 counted value processing. The bit assignment and the functionality of the object type are described in the ST7 system manual.

Cnt01D_S	Double word 1
Byte index	0
Bit index	0
Length in bits	32

The counted value of the object type Cnt01D is addressed by byte index 0, bit index 0 and length 32 (bits).

Object type Cnt04D_S

The object type Cnt04D_S contains a data area of 4 double words (each 32 bits) of the SINAUT ST7 counted value processing. The bit assignment and the functionality of the object type are described in the ST7 system manual.

Cnt01D_S	Double word 1	Double word 2	Double word 3	Double word 4
Byte index	0	4	8	12
Bit index	0	0	0	0
Length in bits	32	32	32	32

The counted values of the object type Cnt04D are identified by byte index 0,4,8,12, the bit index 0 and length 32 (bits).

Object type Cmd01B_R

The object type Cmd01B_R contains a data area of 2 bytes of the SINAUT ST7 command output. The user sees only byte 1 that contains the command to be output. The second byte is used in the receiving station for additional plausibility checks before the command is output. Only bit 1 can ever be set in the command byte.

Cmd01B_R	Byte 1	Byte 2
Byte index	0	Is occupied by a copy of byte 1 during command processing.
Bit index	7 6 5 4 3 2 1 0	
Length in bits	1 or 8	

To address the bit to be set in the command byte, ST7cc Config provides two options with *single bit addressing* and *entire addressing*.

- *Single bit addressing* means that the 8 possible commands are mapped to 8 individual ST7cc variables in the decoding. Each of these 8 variables can be assigned the value 1. If a value higher than 1 is assigned by the WinCC application, the ST7cc interface automatically sets the value 1.
- *Entire addressing* means that the command byte is mapped on an individual ST7cc variable in the decoding. This variable can only be assigned the value 0 to 7. The value 0 to 7 specifies the number of the bit to be set in the byte. If a value higher than 7 is

4.3 Background knowledge on configuring

assigned by the WinCC application, the value is ignored by the ST7cc interface and a message is displayed in the ST7cc Log window.

Object type Set01W_R

The object type Set01W_R contains a 3-word (each 16 bits) data area of the SINAUT ST7 setpoint output. The bit assignment and the functionality are described in the SINAUT ST7 system manual.

Set01W_R	Local	Mirror value	Setpoint
Byte index	0	2	4
Bit index	0	0	0
Length in bits	1	16	16

In contrast to the other blocks, the setpoint block contains both the send and receive data. If appropriate, the first word (byte index 0, bit index 0, bit length 1) contains an identifier that indicates that the setpoint is set to local operation; in other words that it is not possible to enter the setpoint from within ST7cc. The second word (byte index 2, bit index 0, bit length 16) contains the mirror of the setpoint and can be processed like a measured value. The actual setpoint is addressed at byte index 4, bit index 0, bit length 16.

Object type Dat12D_S

The object type Dat12D_S contains a data area of 12 double words (each 32 bits). The information and structure assigned to the double words of the object type by the user can, in principle, differ from double word to double word.

Dat12D_S	Dword 1	Dword 2	Dword 3	...	Dword 12
Byte index	0	4	8		44
Bit index	0	0	0		0
Length in bits	32	32	32		32

Object type Par12D_R

The object type Par12D_R is oriented on a data area that transfers and returns 12 parameters/setpoints as double words. The first word (16 bits) is reserved for the Local message (byte index 0, bit index 0, length 1 (bit)) that indicates that the object is set to local operation; in other words that it is not possible to make an entry to this object from within ST7cc. Word 1 of the object type Para12D_R must not be modified by the user.

The information and structure contained in the double words of the object can vary from double word to double word.

Par12D_R	Word 1	Dword 1	...	Dword 12	Dword 13	...	Dword 24
Meaning:	Local	Mirror value 1		Mirror value 12	Setpoint / parameter 1		Setpoint / parameter 12
Byte index	0	2		46	50		94
Bit index	0	0		0	0		0
Length in bits	1	32		32	32		32

The data area that can be used by the user only begins at byte index 2 and then covers a maximum of twelve double words containing the mirror values (to ST7cc) and starting at index 50 over a maximum of twelve double words containing the setpoints / parameters to be entered locally or transferred by ST7cc to the automation level. If, for example, only 1 setpoint / parameter is used in the ST7 object, then only the data areas with byte index 0, 2 and 50 are occupied in this object.

Note

From the stations, all 12 double words (mirrored values) can be transferred in a block to the ST7cc target subscriber. In the opposite direction, the value change of an ST7cc variable triggers the immediate transfer of the individual double word.

4.3.4 ST7cc variable

Overview

An ST7cc variable is a data section from the data area of a SINAUT object that is managed and processed as a separate information unit on the ST7cc server. The variables are processed, however, in ST7cc and in WinCC. When the variables are defined, different processing functions can be assigned to them depending on their type. For more information on this topic, compare sections Type and subtype of a variable (Page 165) and Processing options for ST7cc variables (Page 168).

A variable can contain both a process value as well as status information from system components. System components are the SINAUT subscribers, see section SINAUT subscriber (Page 158).

For the purposes of describing ST7cc variables, it is sometimes an advantage to distinguish between process variables and system variables so that slight differences in the way they are created can be pointed out.

Process variable

An ST7cc variable that adopts information of a SINAUT object is known as a process variable.

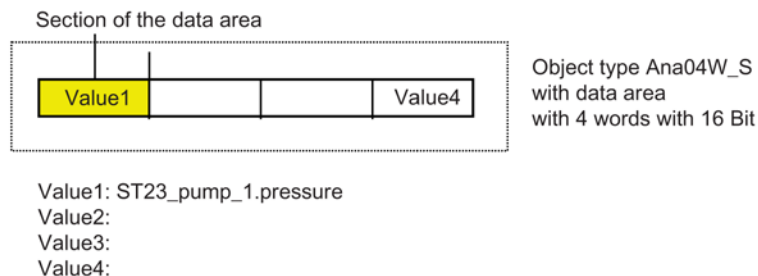


Figure 4-5 Object type Ana04W_S with four information units

System variable

An ST7cc variable that adopts information of a system component is known as a system variable.

To be able to adopt the status information of a system component, several ST7cc variables are generally necessary. The number of variables required depends on the complexity of the component.

The system variables are created automatically for every SINAUT subscriber. During the automatic creation of the system variables, the variable name is assigned automatically. System variables differ from process variables in their naming conventions.

4.3.5 Variable name

Overview

When defining an ST7cc variable, its name is entered by the user. The variable name must be unique. The ST7cc variable name is made up of the group name and the attribute name both for process and system variables.

WinCC tags are generated based on the ST7cc variables (see section Generating (Page 264)). The WinCC tag name is identical apart from the delimiter between group and attribute name.

It has already been mentioned in section ST7cc variable (Page 163) that there is no difference in the naming conventions for ST7cc process variables and system variables. The following paragraphs therefore explain the naming conventions for the following

- ST7cc process variables
- ST7cc system variables

in detail.

With version V2.7, you can also define variables in ST7cc Config that are created as internal tags in WinCC. The advantage of this is that all tags required for processing the SINAUT data supply in WinCC can be configured for WinCC using ST7cc Config.

ST7cc / WinCC variable/tag name for process variables/tags

The ST7cc variable name is made up of the group name and the attribute name. The ST7cc dialogs for the description of the variables demand a two-level name notation that has the following advantages:

- The user can create model partial decodings (typicals) in which, apart from other parameters, the attribute names of the variables defined in the typical have default entries. If these partial decodings (typicals) are used when creating decodings (instantiated), the user only then needs to enter the group name. By entering the group name once, the variable names of all the variables of the typical are completed. For more detailed information, refer to the section Object templates and typicals (Page 168).
- The two-level naming convention supports the user in grouping variables.

Example:

ST23_Pump_1.Pressure
ST23_Pump_1.Flow
ST23_Pump_1.Status
ST23_Pump_1.Alarm
etc.

Remember the following rules:

- The variable name must be unique.
- Group and attribute names must not contain the period or blanks.

In the ST7cc variable list, the two parts of the name are displayed separated by a period (.).

In the WinCC tag name, the parts of the name are separated by an underscore (_).

ST7cc / WinCC variable/tag name for system variables/tags

The two-level naming scheme applies to system variables just as to process variables.

The names of the system variables created automatically when the SINAUT subscriber is configured are made up of the subscriber name and the attribute name. The attribute name is taken automatically from the system typicals (see the section Object templates and typicals (Page 168)). The subscriber name is entered during configuration of the subscriber.

4.3.6 Type and subtype of a variable

Variable types

Now that the previous sections have described what you need to know about the ST7cc variable definition:

1. Specifying the data subarea from the ST7 object data area,
2. Naming rules

The last aspect to describe is how the data subarea is mapped. This is specified using the Type and Sub type parameters. The following table Variable types and sub types explains the possible types and sub types and the corresponding WinCC data types.

The type designations M, S, C, A and D stand for:

- M measured value
- S signal (status messages / alarms)
- C counted value
- A analog output (setpoint / parameter)
- D digital output (command)

4.3 Background knowledge on configuring

Table 4- 1 Variable types and subtypes

Type	Sub type	Permitted lengths	Explanation
M	1	16, 32 bits	The 16, 32 bits are interpreted as unsigned integers. WinCC data type: floating point 64-bit IEEE 754 ST7 source object type: Dat12D_S, Par12D_R
	2	16, 32 bits	The 16, 32 bits are interpreted as signed integers. WinCC data type: floating point 64-bit IEEE 754 ST7 source object type: Ana04W_S (16 bits) ST7 source object type: Dat12D_S, Par12D_R
	3	16 bits	The 16 bits are interpreted as ST1 measured value. WinCC data type: floating point 64-bit IEEE 754 ST1 source object type: ATZ01, ATZ03
	4	32 bits	The 32 bits are interpreted as a floating-point number. WinCC data type: floating point 64-bit IEEE 754 ST7 source object type: Dat12D_S, Par12D_R
S	1	1 to 32 bits	Data areas of 1 to a maximum of 32 bits can be defined as variables. Case 1: length = 1 bit -> WinCC data type: Binary tag Case 2: length = 2 to 32 bits -> WinCC data type: unsigned 32-bit value. ST7 source object type: Bin04B_S, Dat12D_S, Par12D_R ST1 source object type: MTZ01, MTZ02 (16 bits)
C	1	32 bits	The 32 bits represent an ST7 absolute counted value (28-bit value, 4-bit status). WinCC data type: floating point 64-bit IEEE 754 ST7 source object type: Cnt01D_S, Cnt04D_S ST1 source object type: ZTZ01, ZTZ02, ZTZ03
	2	32 bits	The 32 bits represent an ST7 absolute counted value (as in subtype 1). The ST7cc counted value processing, however, forms a difference value. WinCC data type: floating point 64-bit IEEE 754
	3	32 bits	The 32 bits represent an absolute counted value (32-bit value, no status). WinCC data type: floating point 64-bit IEEE 754 ST7 source object type: DAT12D_S
	4	32 bits	The 32 bits represent an absolute value (as in subtype 3). The ST7cc counted value processing, however, forms a difference value. WinCC data type: floating point 64-bit IEEE 754
A	1	16 bits	WinCC data type: floating point 64-bit IEEE 754 is mapped to 16 bits. ST7 target object type: Set01W_R
		32 bits	WinCC data type: floating point 64-bit IEEE 754 is mapped to 32 bits. ST7 target object type: Par12D_R

Type	Sub type	Permitted lengths	Explanation
	2	16 bits	WinCC data type: floating point 64-bit IEEE 754 is converted to a 16-bit ST1 setpoint (the three least significant bits are set to 0) and mapped to 16 bits. ST1 target object type: STA01
	3	-	not used
	4	32 bits	WinCC data type: floating point 64-bit IEEE 754 is mapped to a 32-bit floating-point number. ST7 target object type: Par12D_R
D	1	1, or 8 bits	If length = 1 bit: WinCC data type = binary tag (single bit addressing) If length = 8 bits: WinCC data type = unsigned 8-bit value; in other words by entering a value from 0 to 7, the command bit to be set is addressed and mapped to a command output (entire addressing). For more information on single bit and entire addressing, refer to section SINAUT object types (Page 159) Object type Cmd01B_R ST7 target object type: Cmd01B_R ST1 target object type: BTA01, BTA02
	2	8 bits	Only case 2: WinCC data type: If length = 8 bits unsigned 8-bit value: Command output: transferred as unmirrored organizational command.

Note

The formation of the quality code is explained in section Quality code of the WinCC tags supplied with values by ST7cc (Page 285).

- Example 1:
Variable with type definition *M*, 1, 16 bits:
The 16-bit data subarea is interpreted as an unsigned integer by the ST7cc message decoding and mapped to a WinCC data type *floating point 64-bit IEEE 754*. Whether these 16 bits originate from a Bin04B_S or Dat12D_S ST7 object type is unimportant for the decoding or variable definition.

- Example 2:
Variable with type definition *A*, 4, 32 bits:
Due to the length = 32 bits (in the ST7cc variable definition), the data type *floating-point number 64-bit IEEE 754* is created in WinCC. The 32 bits should be output to the automation level by ST7cc in floating-point format. The data target (ST7 object in the station) must be able to accept the 32 bits and must be of the Par12D_R object type. In this case, the target object type Set01W_R would not be possible because this can only process a 16-bit value.

4.3.7 Processing options for ST7cc variables

Depending on the object type, the following processing functions can be assigned to an ST7cc variable:

- Message processing (basic function)
- Entry of static text blocks as an expansion of message processing
- Archive processing
- Measured value processing
- Counted value processing

These functions are executed during message decoding by the ST7cc server and further processed WinCC. The processing by the ST7cc server can be seen as preprocessing for WinCC.

The parameter assignment for the processing is made in ST7cc Config. When generated in WinCC, the parameters for executing the processing functions are transferred to the WinCC components using ODK functions. As a result, a processing function only needs to be assigned parameters once. For a detailed description of the processing functions, refer to section Configuring processing functions (Page 231).

4.3.8 Object templates and typicals

Overview

Object templates and typicals are prepared decodings stored in the project library for the user.

Object templates

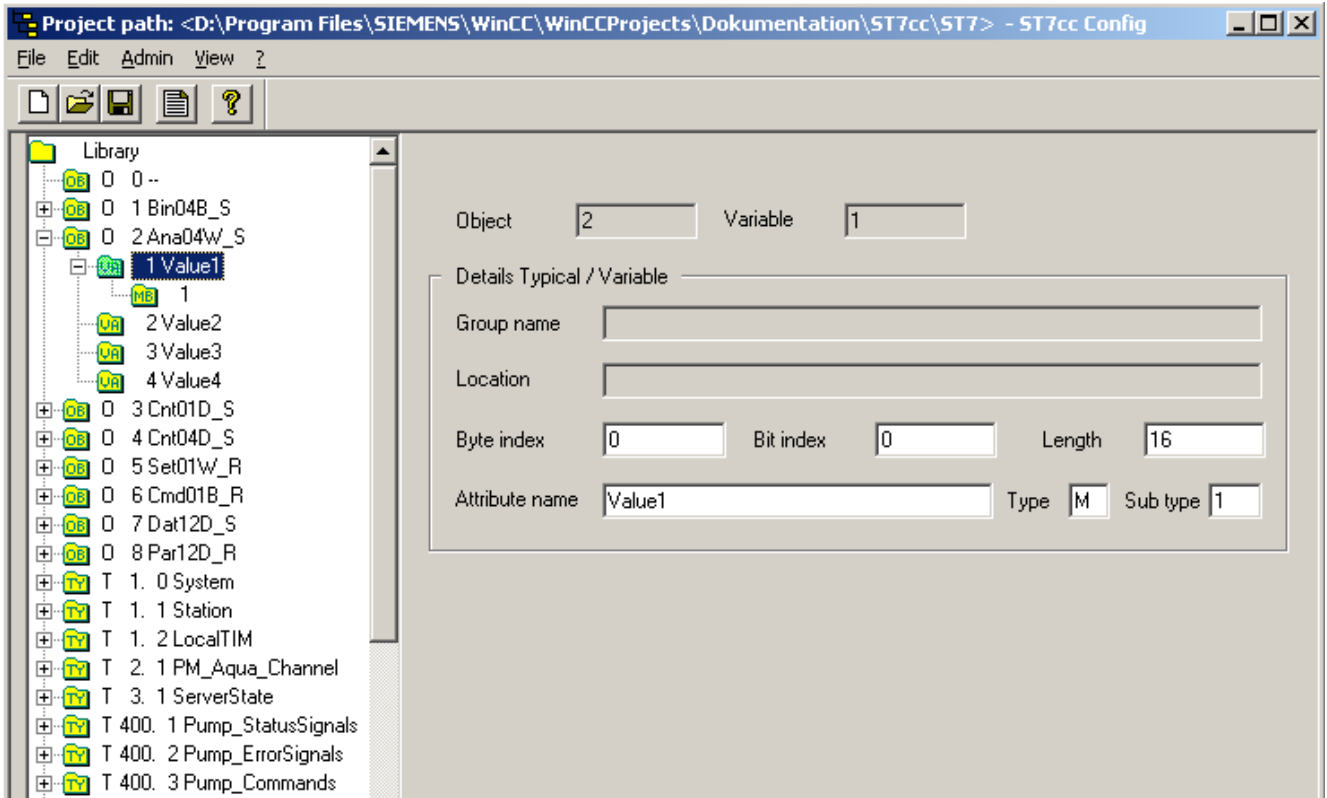




Figure 4-6 Project library with object templates (O) and typicals (T)


The project library contains object templates for the ST7 object types as examples of object type-specific decoding. The decoding always relates to the entire data area of a SINAUT object.

By modifying the supplied object templates and by inserting new object templates, the user can create model templates for specific projects.

By copying an object, the user can easily create the decoding of a SINAUT object.

The figure shows the *Library* project library. The object templates in the library can be recognized by the  icon and the letter *O* before the name of a (object) template.

The variables of an object can be recognized by the  icon. In the figure, the object template no. 2 *Ana04W_S* contains four variables with the attribute name *value1* to *value4*.


The  icon also makes it clear that a message processing function is assigned to variable 1.

Typicals

A typical is a model decoding with mechanisms allowing features to be inherited. The model decoding can relate to the entire data area of a SINAUT object. Generally, however, a typical usually relates only to a data subarea of an object data area that requires decoding as a commonly occurring data structure. The data subarea can have a maximum length of 32 bits; in other words, the maximum data that can be decoded as a typical is one double word of a data area of a SINAUT object. When decoding a SINAUT object with typicals, remember that positioning is only possible at byte addresses. For more detailed information, see section Principle of decoding using typicals (Page 171).

Creating a typical defines one or more variables including the defaults for their parameters resulting in minimum engineering effort during the later decoding.

The user can use typicals when creating a decoding of a SINAUT object (instantiation) with the advantage that any changes made to a typical (in the library) affect all decodings containing the typical as a partial decoding.

The figure shows an excerpt from the project library. The typicals in the library can be recognized by the  icon and the letter T before the name of a typical.

The use of typicals requires that an analysis is carried out during the planning stage of the plant concept to recognize repeating structures.

Terminologically, a distinction is made between system and user typicals. If no misunderstandings can be expected, the term typical is used alone.

The standard library contains three system typicals and several examples of user typicals, see figure.

System typical

The system typicals contain model partial decodings for generating the system variables for the ST7cc server (system) and the SINAUT subscribers. ST7cc Config uses the following system typicals for automatic generation of the ST7cc system variables:

- T 1.0 System (ST7cc server)
- T 1.1 Station (CPU in the station)
- T 1.2 Local TIM (for local TIMs connected over MPI or Ethernet to the ST7cc PC)
- T 2.1 PM_Aqua_channel
- T 3.1 ServerStatus

Note

System typicals have a typical no. lower than 100. System typicals must not be modified without consulting the system supplier!









User typicals

All typicals created by the user to decode SINAUT objects must have a typical number \geq 100.

Addressing / managing typicals

The typicals are managed using two-level addressing (type no., subtype no.). The figure shows not only system typicals but also user typicals (type no. 100) as model decodings for a number of type A. Typical 100.1 represents the model decoding of the status codes. Typical 100.2 represents the model decoding of the information representing a fault. Typical 100.3 represents the model decoding of the commands for the control direction. Type no. 100 forms the logical parenthesis over the three typicals which is required for typical-based decoding of the pump technological object of type A.

Table 4- 2 Typicals of the project library for a pump type A

	T	1.	0	System
	T	1.	1	Station
	T	1.	2	LocalTIM
	T	2.	1	PM_Aqua_channel
	T	3.	1	ServerStatus
	T	100.	1	Pump_Type_A_Status
	T	100.	2	Pump_Type_A_Error
	T	100.	3	Pump_Type_A_Commands

4.3.9 Principle of decoding using typicals

Overview

As already mentioned in section Object templates and typicals (Page 168), a typical is a partial decoding for a commonly occurring data structure that can occur repeatedly within a SINAUT object.

Since the SINAUT object types (data point typicals) available today allow only one transfer direction (monitoring or control direction) at least two SINAUT objects are required to control and monitor technological objects such as a pump.

The following examples illustrate how the creation of repeating data structures and use of typicals minimizes engineering effort and how, at the same time, optimum memory allocation on the CPU can be achieved.

Instantiating a typical

If a typical is used within a decoding, this is known as instantiation.

Offset of a typical

The beginning of the data structure of a typical can only be located at byte boundaries. Offset 0 positions the start of the typical at byte address 0 of the user data area of the SINAUT object. Offset 1 positions the start of the typical at byte address 1 etc.

Offset example 1

In example 1, three typicals as repeating data structures represent the information units for the operating status codes (automatic, manual, off, on, revision, local, disabled), the error codes (control error, not controllable, access violation, overtemperature) and the commands (automatic, manual, off, on) of a pump of type A. The data structure has a length of 8 (bits).

A SINAUT object of the type Bin04B_S can, for example, include status information for four pumps of type A or the status and error codes of two pumps. In the first situation, this means that when decoding the SINAUT object, the typical 100.1 (see figure) is used or instantiated four times. In the second situation, this means that when decoding the SINAUT object, the typicals 100.1 and 100.2 are each instantiated twice.

Using the SINAUT objects 52, 53, 54, and 55 of the type Cmd01B_R, pumps 1 through 4 will be controlled. To control a pump of type A, the typical 100.3 is created for the repeating data structure (automatic, manual, off, on).

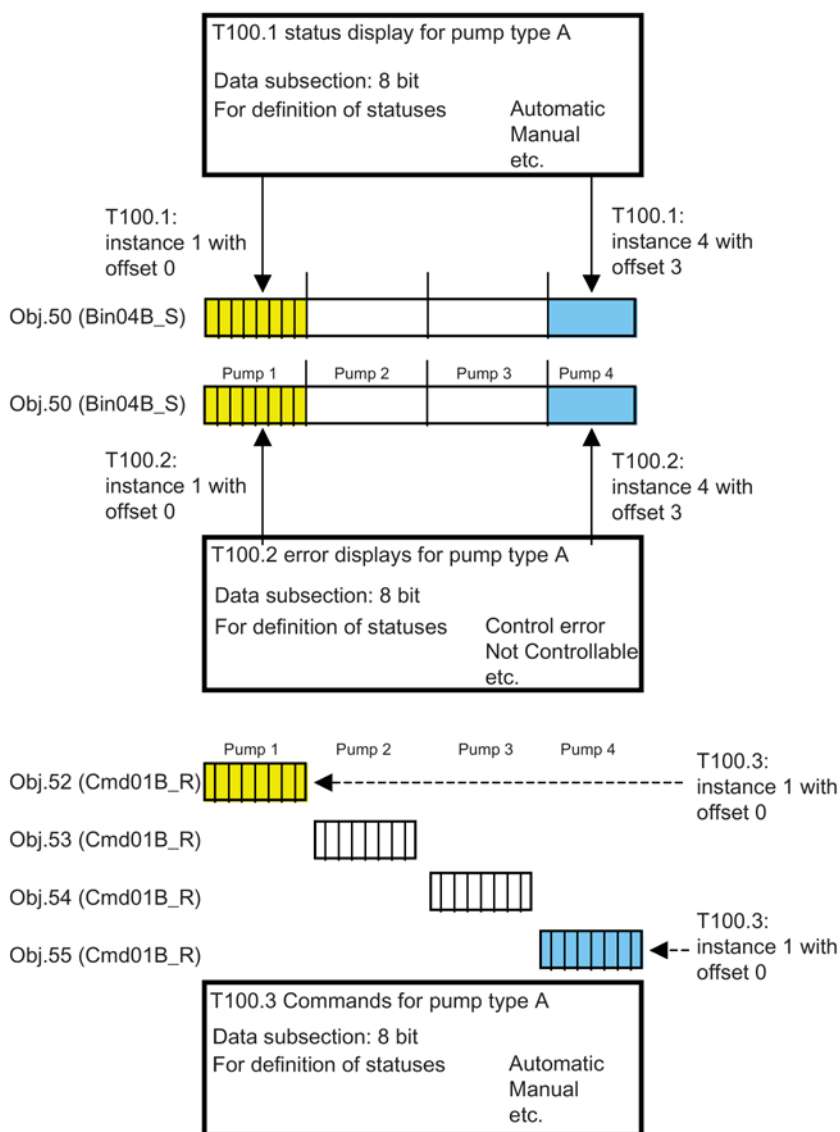


Figure 4-7 Decoding with typicals

Object 50 in the figure is decoded by the user repeatedly instantiating typical 100.1 with (instance 1 / offset 0), (instance 2 / offset 1), (instance 3 / offset 2) and (instance 4 / offset 3) and using the offset information to position on the data subarea to be decoded. With each instance, only the group name needs to be entered as part of the ST7cc variable name. This completes all the variable names of the typical.

The SINAUT objects 52 to 55 are each decoded by instantiating typical 104.2 with the parameter information instance 1, offset 0.

Offset example 2

Positioning on byte boundaries (specified offset) gives the impression that the coding using typicals can only be used optimally in terms of memory when a data area of at least 8 bits exists as a repeating data structure. When monitoring simple objects, such as windows,

three or four bits may well be enough to map their states such as closed, open, tilted and broken. The figure shows how typicalals can also be used in this application while making optimal utilization of memory in the station.

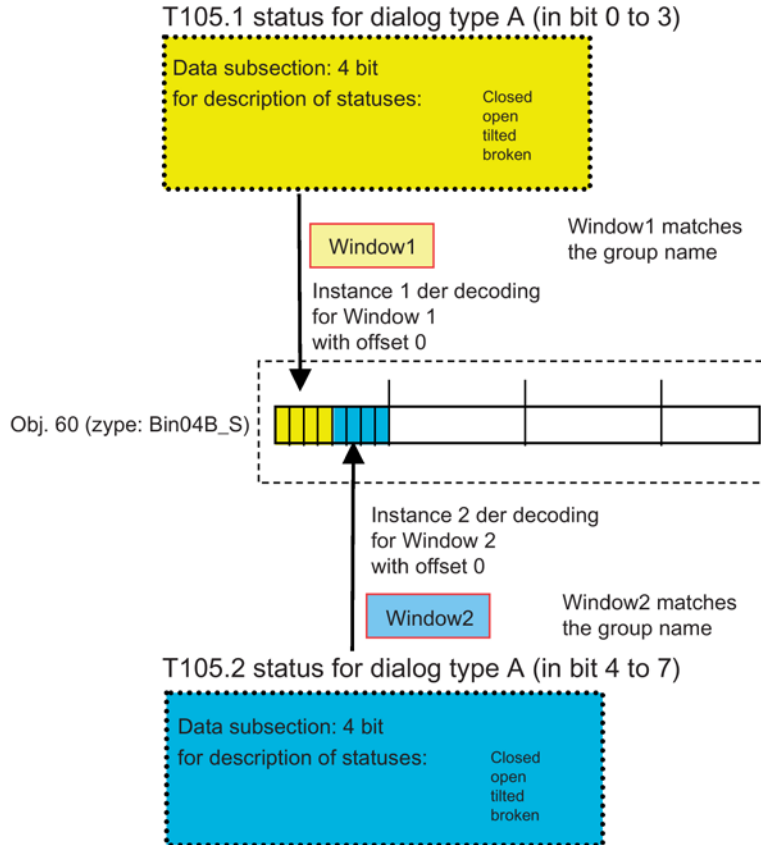


Figure 4-8 Decoding with typicalals

4.3.10 Group display

Group display

Note

The "Basic Process Control" package is required if you want to use group displays in WinCC. As of V6.0 SP3, this package is an integral part of WinCC. Prior to this version, it was available as an optional package.

Group displays for displaying alarms and warnings in any plant areas allow the operator to react quickly and to intervene in specific parts of a process (refer to the description PCS 7 runtime).

Group displays are used for the compressed representation of process status displays in graphic form. The group display object must be controlled by a tag (EventState variable) that represents the message status. You can use this tag in the other WinCC components if you want to represent statuses of group displays there.

Basically, this means that we are dealing with three terms:

- **Group display:**
The group display is a functionality found in PCS 7 that defines the assignment of the group display variable (EventState variable), allows the linking of group display variables and makes a defined representation of states of the group display variables possible by using a group display object.
- **Group display variable (EventState variable):**
The group display variable is also known as the "EventState" variable. This is the name under which you will find the variable in the typical definitions of the technological objects. The bit assignment of the "EventState" variable is defined precisely and can be found in the PCS 7 descriptions or online helps.
- **Group display object:**
The group display object is a WinCC picture element that displays the states of the group display tags graphically.

To be able to use the group display functionality, the following requirements must be met in ST7cc:

- The typical file `st7_typicals_pcs7.txt` or `st7_typicals_pcs7_english.txt` must be linked into PCS 7 and included in the message classes and message types used by the group display.
- The modified picture typicals for technological objects, stations, TIMs and ST7cc server objects must be used.
- The number assignment for PCS 7 user text blocks must be configured for the user message text blocks.

Typical file

The new typical file `st7_typicals_pcs7.txt` or `st7_typicals_pcs7_english.txt` is intended for use of the group display.

If you convert an existing project for this typical file, you must also use the appropriate picture and faceplate files as of version V2.7.

Description of the typical for the "EventState" variable

You will find descriptions of the typicals for the technological objects in the section Typicals in ST7cc (Page 341). At this point, we will look more closely at the part of the description of the typical involved in the group display.

4.3 Background knowledge on configuring

Based on the Alarm and Alarm_S variables of the typical description, the ST7cc server updates the following bits in the EventState variable:

- Bit 31 (Alarm High)
Bit 15 (acknowledgment bit belonging to bit 31)
- Bit 29 (Warning High)
Bit 13 (acknowledgment bit belonging to bit 29)

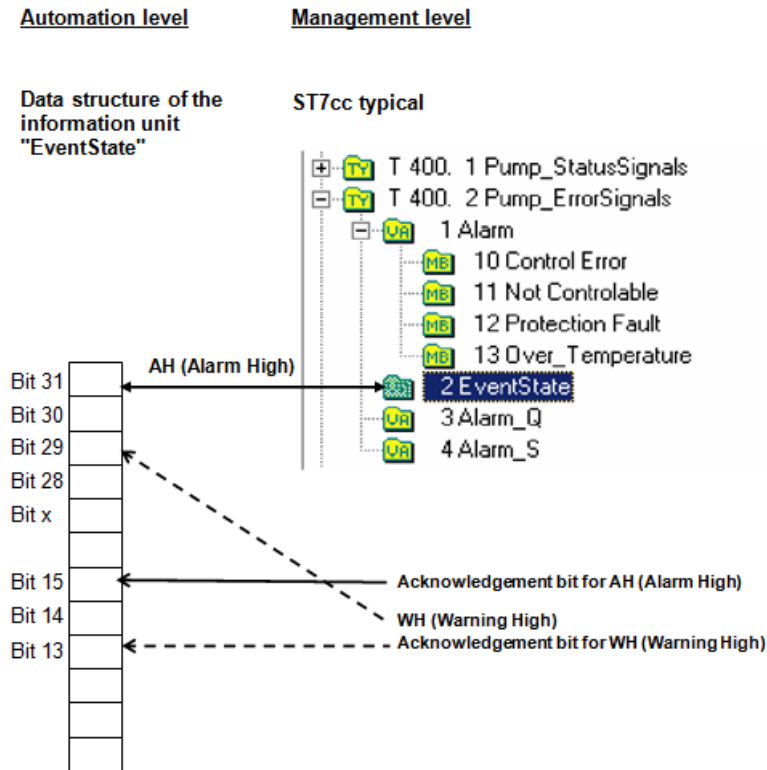


Figure 4-9 Pump1_Errorsignals

The Alarm variable of the typical T400.2 contains the defined alarms of the technological object. If the ST7cc server recognizes that an alarm is coming, bit 31 (alarm) is set and bit 15 (unacknowledged) is reset in the EventState variable. The bit assignment of the EventState variable is described in the PCS 7 description and in the online help. Once the alarm is cleared again, bit 31 is reset by the ST7cc server. The corresponding acknowledgment bit (bit 15) is set or reset depending on the acknowledgment status (bit 15 = 1 = acknowledged, bit 15 = 0 = unacknowledged).

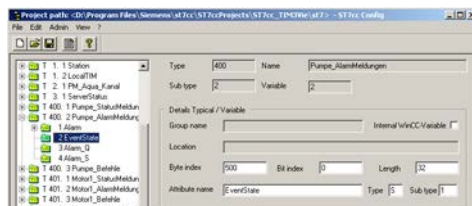


Figure 4-10 Definition of the EventState tag in ST7cc Config

The figure shows the definition of the EventState variable. The EventState variable begins at byte index 500 and this must not be modified.

Acknowledgment variables in the description of the typical

Variables 3 and 4 are the acknowledgment variable Alarm_Q and the status variable Alarm_S. The variables are defined as starting at byte index 504 or 508 and once again, these must not be modified by the user.

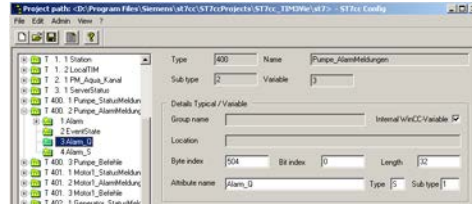


Figure 4-11 Definition of the Alarm_Q and Alarm_S tags in ST7cc Config

The Alarm_Q tag is an internal WinCC tag. Using this tag, a WinCC application can acknowledge an alarm state.

When an alarm state is acknowledged by the WinCC user, the ST7cc server is informed of this by tag 4 (Alarm_S). If the acknowledgment relates to a disturbance that was mapped to bit 31 of the EventState variable, the server now sets acknowledgment bit 15 of the EventState variable.

Note

The byte index of the typical variables EventState, Alarm_Q and Alarm_S must not be modified by the user. The ST7cc server expects to find the data areas described above starting at these indexes.

Note

The ST7cc server program was also prepared so that warnings (bits 29 and 13) can be processed in the EventState variable. You will then need to expand the description of the typical accordingly. The typical variables for the warnings must begin at byte address 512 for the acknowledgment variable (Warning_Q) and at byte address 516 for the status variable (Warning_S). You will also need to modify the supplied picture typicals and faceplates.

Picture typicals

The picture typicals as of version V2.7 are designed for the use of the group display.

Note

The supplied picture typicals for technological objects show only the AH (Alarm High) in the group display. The definitions of the typicals for the technological objects available in the typical file do not include any variables for warnings.

4.4 Configuring

This section describes the activities involved in configuration in other words, how to

- set the parameters for the data of the SINAUT objects configured in the stations and
- map the most important status information of the SINAUT subscribers to ST7cc variables and to set the parameters for their processing in ST7cc and in WinCC.

Overview

ST7cc configuration consists essentially of the following activities supported by suitable tools and dialogs.

Listing SINAUT objects with the SINAUT Diagnostics and Service Tool:

With the *SINAUT Diagnostics and Service Tool*, the user can display the SINAUT objects and their essential parameters for all or selected subscribers and store them in a file for further use in ST7sc Config.

Setting up SINAUT subscribers:

In a later step, the SINAUT subscribers are set up in ST7cc Config. This is supported by the *New Subscriber* dialog. Setting up a SINAUT subscriber is necessary so that

- the ST7cc system variables containing the status displays of the subscriber are created automatically for the subscriber and
- the decodings can be created for the SINAUT objects of the subscriber.

Creating object templates:

The supplied object templates should only be considered as simple examples. Users themselves must decide whether they want to create model object templates.

Creating the decodings for SINAUT objects:

In this step, the SINAUT object data area is mapped to one or more variables and, as an option, processing functions can be assigned and configured. As an alternative, this can also be done as follows.

- Creating the decoding and defining the individual variables and their processing
- Creating the decoding and its variables with typicals
- Creating the decoding by copying an object template
- Creating the decoding by copying an existing decoding
- Creating several decodings by copying the decodings of a subscriber.

The figure shows how decodings can be created in various dialogs after setting up a subscriber.

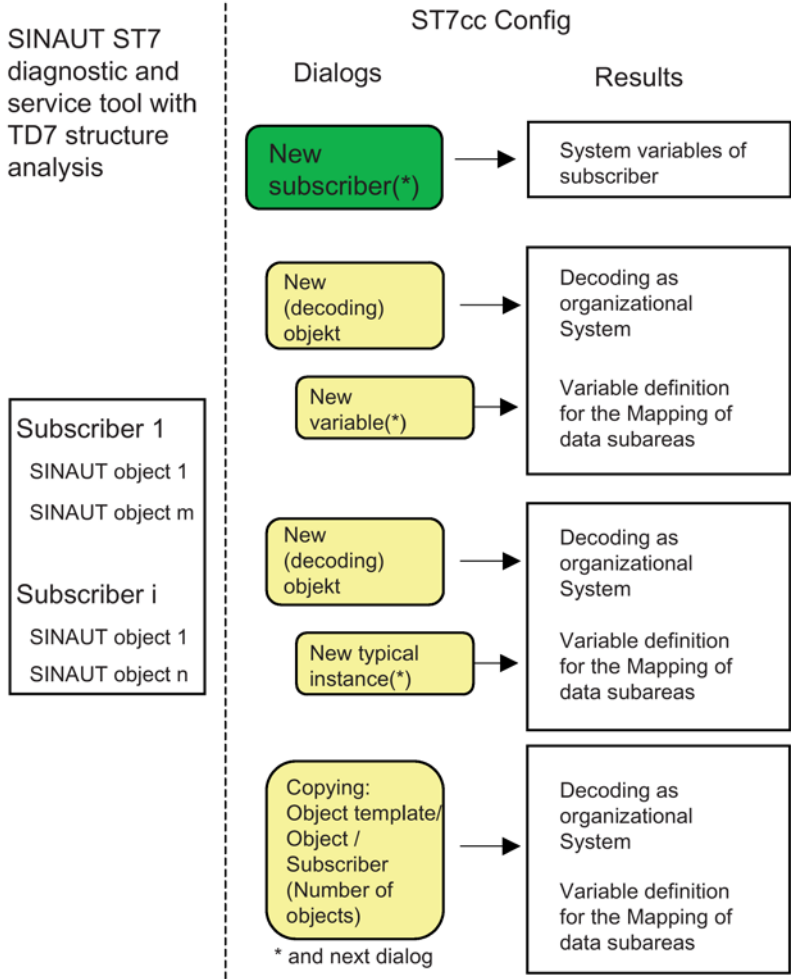


Figure 4-12 Options for creating a decoding

When creating a decoding, data subareas with repeating data structures can be decoded using typical and the information units of the remaining data area by step-by-step creation of the variables. When decoding object types such as Dat12D_S, Par12D_R or Bin04B_S, it is often useful to use both methods.

Note

You can make configuration more efficient by noting repeating data structures when configuring the SINAUT objects so that they can be decoded with typical. Only then can you use inherit mechanisms to make subsequent modifications simpler.

Variables in object templates and in typical can only be assigned message processing functions (basic function). Further processing (archive processing etc) can only be assigned to a variable within the actual decoding. When you copy a decoding, however, you also copy all the processing functions of a decoding. From this perspective, copying objects that contain typical is an efficient procedure with advantages.

4.4.1 Starting ST7cc Config

Creating and opening an ST7cc project is described in detail in the section Creating an ST7cc project (Page 107). There, you will find the information you require on:

- Starting ST7cc Config
- Creating a new ST7cc project

After you have successfully created a project, the ST7cc Config screen (see figure) contains:

- The library with the object templates and typicals
 - The subscriber 0 system, that contains the "non visible" and "visible" decodings via the ST7cc system. Subscriber 0 represents the ST7cc server. The ST7cc system variables of the "non visible" decodings are created automatically to be able to receive the main status information from the server. Along with the ServerStatus and PM-AQUA decodings that are "visible" to the user, further ST7cc variables are normally created, to be able to receive additional system information for single or redundant ST7cc servers and PM-AQUA channels. If this information is not required by the user, the decodings can be deleted.

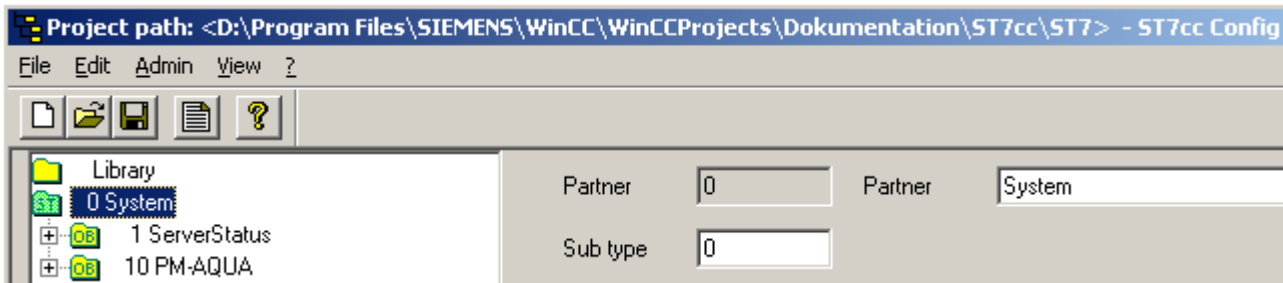


Figure 4-13 Content of the project file

For more detailed information on the terms used, refer to the relevant sections:

- Object template: Section Object templates and typicals (Page 168), Object templates (Page 182)
- System variable: Section ST7cc variable (Page 163)
- Subscribers: Section SINAUT subscriber (Page 158)
- Typical: Section Object templates and typicals (Page 168), System typicals (Page 184), Creating a user typical (Page 199)

4.4.2 ST7cc object tree

The figure shows the structure of the ST7cc object tree in the ST7cc Config screen and the icons used for its objects.

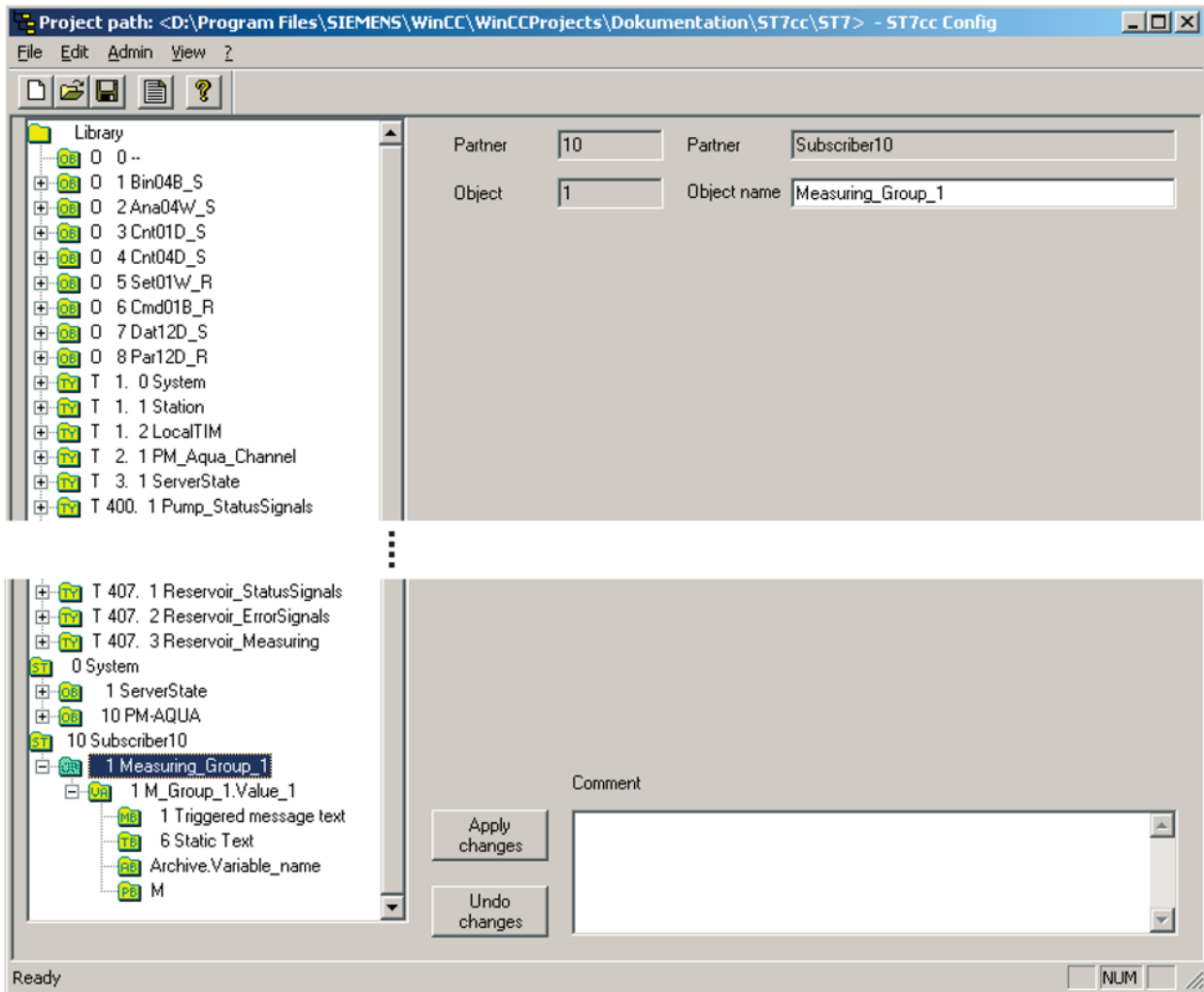











Figure 4-14 ST7cc object tree

The object tree contains the following objects:

- Object templates of the library; these are indicated by the icon  and the letter O before the name.
- Typicals of the library; these are indicated by the icon  and the letter O before the name.
- SINAUT subscribers; these are indicated by the icon .
- Decodings of the SINAUT objects of the subscribers; these are indicated by the icon .

4.4 Configuring

- Variables of the decodings, These are indicated by the icon .
- Processing functions of the variables are indicated by the icons  (message processing),  (text block entry),  (archive processing) and  (process value processing).

4.4.3 Object templates

ST7cc object tree

The figure shows the object tree in the ST7cc Config screen. In the library, for example, an object template for the ST7 object type Ana04W_S is "expanded". This object template contains the definitions of four variables. Variable 1 with the attribute name Value 1 decodes a 16-bit long data server area of an ST7 object data area beginning at byte and bit index 0. Due to the type and subtype information, the data subarea is interpreted as a 16-bit long unsigned integer.

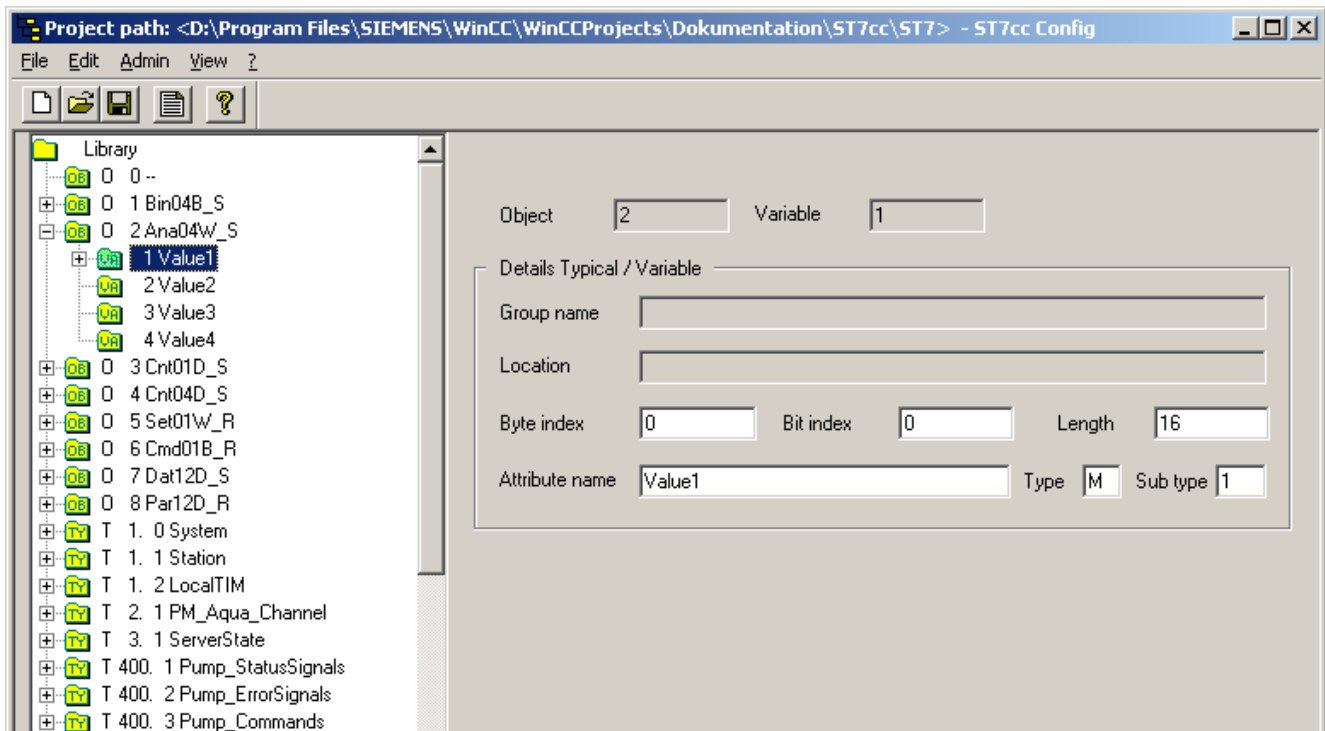


Figure 4-15 ST7cc object tree

By selecting variable 1 with the attribute name Value1, the parameters of the variable Value1 are displayed in the Details Typical / Variable box.

Creating an object template

To create a new object template in the library, only the first two steps are described. The subsequent steps are identical to those when creating a decoding (see section Creating a decoding (Page 210)).

To create an object template, follow the steps outlined below:

1. Select the library with the right mouse button and open the context menu (see figure).
2. Select the New Object option.

The *Add* dialog opens.

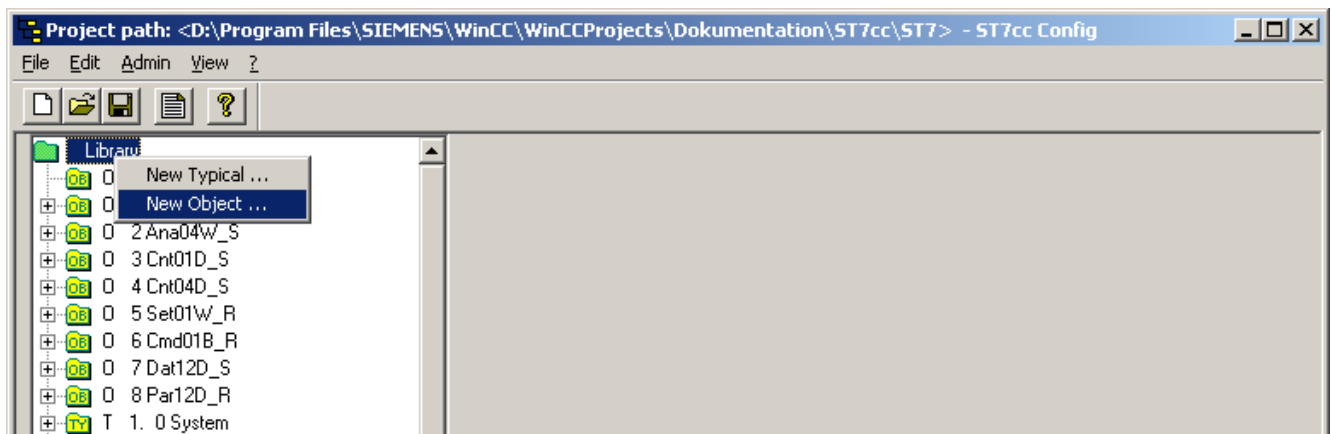



Figure 4-16 Selecting a subscriber to create a decoding.

3. In the Add dialog, enter the number under which the new object template will be created (see figure).



Figure 4-17 Entering the number of the object template

The subsequent steps are identical to those when creating a decoding. In principle, you must make the following entries for each variable:

- Attribute name: Enter the attribute name of the variable. You can only enter the group name when you configure the decoding of a SINAUT object by copying an object template.
- Byte index, bit index and length (in bits): When you enter these parameters, you specify the data subarea within the object data area that will be mapped to the variable.
- Type, sub type: These entries specify how the data of the data subarea will be converted.
- In the comment field, you can, for example, enter a note.
- Object templates and typicals can only be assigned message processing functions ( parameter fields for the basic function).

For further information, refer to sections: Variable name (Page 164), Type and subtype of a variable (Page 165), Object templates and typicals (Page 168) and Principle of decoding using typicals (Page 171).

4.4.4 System typicals

Overview

The system typicals contain model partial decodings for generating the system variables for the ST7cc server (system) and for the SINAUT subscribers

1. to be able to transfer the essential status information (operating and error messages) to WinCC, and
2. to be able to trigger organizational commands, such as a general request (GR).

ST7cc Config uses the following system typicals for automatic generation of the ST7cc system variables:

System typicals for basic information


1. T 1.0 System (ST7cc server)
2. T 1.1 Station (CPU in the station)
3. T 1.2 Local TIM (for local TIMs connected over MPI or Ethernet to the ST7cc PC)

When you create a SINAUT subscriber (local TIM, CPU in the station), the corresponding ST7cc variables and WinCC tags are created based on the system typical shown above. In the typical definitions, you only ever see the attribute name of the variable. The full name of the system variables is made up of the subscriber name and the attribute name. You specify the subscriber name when you set up the subscriber. In the *System* decoding and the decodings of the subscribers, the generated variables are not visible.

System typicals for additional information

On one hand to allow different system configurations, and on the other hand to meet the information requirements of the user, the following system typicals are available to the user:

1. T 2.1 PM_Aqua_channel
2. T 3.1 ServerStatus


Instances of these typicals are instantiated automatically in  System when the project is created to be able to generate the system variables and to display the expanded status information in faceplates.

Based on the system typicals, the ST7cc variables are described below so that you can also evaluate them in your WinCC application.

The full name of these system variables is made up of the group name and the attribute name. The attribute names of the system typicals must not be modified by the user since the faceplates use them to display system information and to execute organizational commands.

You will find more detailed information in Sections Variable name (Page 164) and Object templates and typicals (Page 168).

System typical System

Under typical description *T 1.0 System*, the figure shows the variables of the system typical *System*. The system variables are created automatically based on this typical. They contain the essential status information of the ST7cc server. Under the decoding  0 *System*, this typical is not visible as an instance.

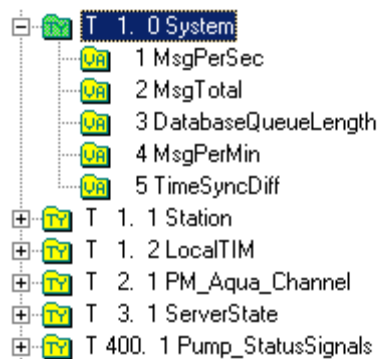



Figure 4-18 Variables of the System typical

Attribute name	Explanation
MsgPerSec	Number of messages received in the last second
MsgTotal	Total number of messages received since the server started
DatabaseQueueLength	Number of process values still to be transferred to the WinCC archive.
MsgPerMin	Number of messages received in the last minute
TimeSyncDiff	Difference in seconds between the computer time and the time of the time master TIM

The values of the listed variables are made available in WinCC.

ServerStatus system typical

Under typical description *T 3.1 ServerStatus*, the figure shows the variables of the system typical *System*.

To be able to accommodate the expanded status information of an ST7cc server, a decoding known as *ServerStatus* is created under the subscriber  0 *System*. The decoding is designed to be able to hold the expanded status information of a redundant ST7cc system. There are two instances of the ServerStatus system typical. For typical instance 1, the group name *Server1* is used, and for typical instance 2 the group name *Server2*.

If you do not have a redundant ST7cc configuration, you can delete typical instance 2 in the ServerStatus decoding. If you want to expand to a redundant system later, you will need to create the typical instance 2 again so that the ST7cc and WinCC tags can be created to accommodate the status information of the second (redundant) ST7cc server.

The figure shows the attribute names defined in the system typical as supplied as of ST7cc version V 2.4.0.2.

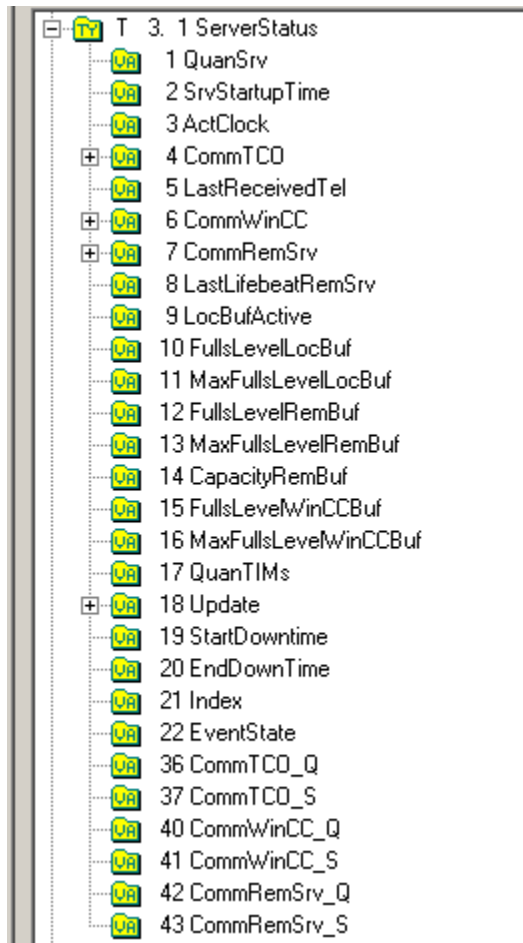


Figure 4-19 Variables of the ServerStatus typical

The status information contained in the den ST7cc / WinCC tags is displayed to the user in appropriate picture typicals and faceplates (see section Diagnostics: Subscriber typicals and faceplates (Page 306)). The attribute and variable names of the system variables are only of interest to you as the user if you want to further process the status information in your WinCC application. To allow you to interpret and evaluate the content of the system variables in your WinCC application, the variables are described briefly below.

Table 4-3 Variables of the ServerStatus typical

Attribute name	I/O	B	C	Explanation
QuanSrv	I			Number of ST7cc servers. 1: Configuration is not redundant 2: Configuration is redundant
SrvStartupTime	I			Indicates when (date, time) the relevant ST7cc server was started.

Attribute name	I/O	B	C	Explanation
ActClock	I			Current time of the relevant server. In redundant systems, discrepancies in the time information due to an error can be recognized by the user.
CommTCO				The TCO system component communicates with the TIMs connected over MPI or Ethernet.
	I	0	O	TCO communication unknown.
	I	1	I	TCO communication problem. ST7cc server has no contact with TCO.
	I	2	O	TCO communication OK
LastReceivedTel	I			Time at which a message was last received.
CommWinCC	I	0	O	WinCC communication unknown
	I	1	I	WinCC communication problem. The ST7cc server has no connection to the WinCC runtime system
	I	2	O	WinCC communication OK
CommRemSrv				Remote (redundant) ST7cc server in redundant ST7cc system.
	I	0	O	Remote server communication unknown
	I	1	I	Remote server communication problem
	I	2	O	Remote server communication OK
LastLifebeatRemSev	I			Time information: Last lifebeat received from redundant partner.
LocBufActive	I			Indicates that the ST7 messages are being buffered locally because the (local) WinCC runtime system cannot be reached.
FullsLevelLocBuf	I			Number of messages in the local buffer to indicate the current fill level.
MaxFullsLevelLocBuf	I			Maximum number of messages that can be stored in the local buffer (system parameter).
FullsLevelRemBuf	I			Number of messages in the remote buffer to indicate the current fill level.
MaxFullsLevelRemBuf	I			Maximum number of messages that can be stored in the remote buffer (system parameter).
CapacityRemBuf	I			Constantly recalculated value. On the basis of the messages already entered in the buffer, a calculation is made to predict how long the remote buffer can continue to store messages if they continue to occur at the same rate. If, for example, 10% of the buffer capacity has been used in the last 2 hours, the prediction will be 20 hours before the buffer is completely full (MaxFullsLevelRemBuf).
FullsLevelWinCCBuf	I			Number of jobs in the WinCC buffer to indicate the current fill level.
MaxFullsLevelWinCCBuf	I			Maximum number of jobs that can be stored in the WinCC buffer.
QuantIMs	I			Number of local TIMs connected over the MPI bus or Ethernet.
Update	I			In a redundant system, this indicates that a synchronization is currently taking place.

4.4 Configuring

Attribute name	I/O	B	C	Explanation
StartDowntime	I			Start of the synchronization period for a redundant ST7cc system.
EndDowntime	I			End of the synchronization period for a redundant ST7cc system.
Index	I			Message counter from which a still active data synchronization can be recognized.
EventState	I			Group display variable. The bit assignment matches that specified by PCS 7 (see section Group display (Page 174))
CommTCO _Q	I			Acknowledgment variable for the status variable CommTCO (see above); is created in WinCC as an internal tag.
CommTCO _S	I			Acknowledgment status variable for status variable CommTCO (see above)
CommWinCC _Q	I			Acknowledgment variable for the status variable CommWinCC (see above); is created in WinCC as an internal tag.
CommWinCC _S	I			Acknowledgment status variable for status variable CommWinCC (see above)
CommRemSrv _Q	I			Acknowledgment variable for the status variable CommRemSrv (see above); is created in WinCC as an internal tag.
CommRemSrv _S	I			Acknowledgment status variable for status variable CommRemSrv (see above)

The following abbreviations are used in the table:

- 1. Column I/O:
 - I: Input. The SINAUT subscriber or the ST7cc server generates the information
 - O: Output. The WinCC application supplies the ST7cc variable. From this, the ST7cc server forms a message to the target component.
- 2. Column B (bit number):
 - Number of the bit set for status display.
- 3. Column C (class):
 - E: Error message. The transferred value represents a problem.
 - O: Operation message. The transferred value represents a correct operating state.

PM_Aqua_channel system typical

The WinCC add-on PM-AQUA for archiving and processing of process data is described in detail in section PM-AQUA link (Page 321). Section PM-Aqua configuration with ST7cc Config (Page 322) describes the configuration of the PM-Aqua process connections with ST7cc Config.

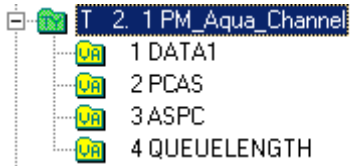


Figure 4-20 of the PM_Aqua_channel typical

PM_Aqua_channel variables are described in the section PM-AQUA process links (Page 321).

Note
Stipulation

The decoding (object 10) under the System subscriber is reserved to allow system variables to be created for the PM-AQUA process connections. An instance of the PM_Aqua_channel typical must be created for each process connection. The group name that then needs to be entered must be PM-AQUA0x where x is the number of the process connection.

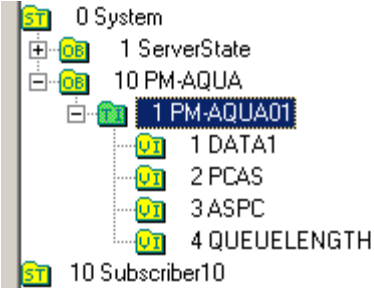


Figure 4-21 Instantiation of the PM_Aqua_channel typical under object 10

System typicals for ST7cc subscribers

For ST7cc, subscribers are:

1. The CPUs in the stations,
2. The local TIMs; in other words, the TIMs connected to the ST7cc PC locally over the MPI bus or Ethernet.

ST7cc uses the following system typicals to create the system variables:

- T 1.1 Station (CPU in the station)
- T 1.2 LocalTIM for local TIMs connected over MPI or Ethernet to the ST7cc PC

These typicals are intended for the TIM modules:

- TIM 3x and 4x as of firmware version V4.3
- TIM 3V-IE as of firmware version V1.0

Note

Functional expansions in the TIM firmware and the development of new TIM modules made further system variables necessary to be able to display the expanded status information to the user in picture typicals and faceplates. These requirements meant a modified or new system typical.

To make sure that you use the correct system typicals, read the notes below.

Update scenarios for ST7cc system typical

Upgrade stages for the use of picture typicals and faceplates in existing and new projects

ST7cc version used	Characteristics of the system typicals	TIM modules and TIM firmware version in the project	Picture typicals / faceplates in the WinCC project created with ST7cc version	Case (see below)
V2.4.x	<ul style="list-style-type: none"> Expansion of the range of information Distinction between local TIM and station (CPU) 	TIM 3x/4x as of firmware version V4.x	V2.4.x	A
V2.5	<ul style="list-style-type: none"> As V2.4 Expanded scope of information for TIM 3V-IE variants 	TIM 3x/4x as of firmware version V4.x	V2.4.x V2.5	C A
		TIM 3V-IE variants	V2.4.x > V2.5 *) V2.5	D A
V2.6	<ul style="list-style-type: none"> As V2.5 Enhanced function in the station faceplate 	TIM 3x/4x as of firmware version V4.x	V2.4.x / V2.5 V2.6	C1 A
		TIM 3V-IE variants	V2.4.x > V2.6 *) V2.5 V2.6	D E A
V2.7 to V3.0	<ul style="list-style-type: none"> As V2.6 Expanded scope of information for TIM 4R-IE Expansion of the group display function 	TIM 3x/4x as of firmware version V4.x	V2.4.x / V2.5 / V2.6 V2.7	C1 A
		TIM 3V-IE variants	V2.4.x > V2.7 *) V2.5 V2.6 V2.7	B E E A
		TIM 4R-IE	V2.7	A

*) You need to use the typicals/faceplates of the newer version in your existing project.

Case A:

If you start a new WinCC project with ST7cc as of Version V2.4, you will normally also use the supplied system typicals, picture typicals, and faceplates of the current ST7cc version. If you select this strategy, enable the New Faceplates (from ST7cc V2.4) option in the Server tab of the ST7cc Project Settings dialog (see figure).

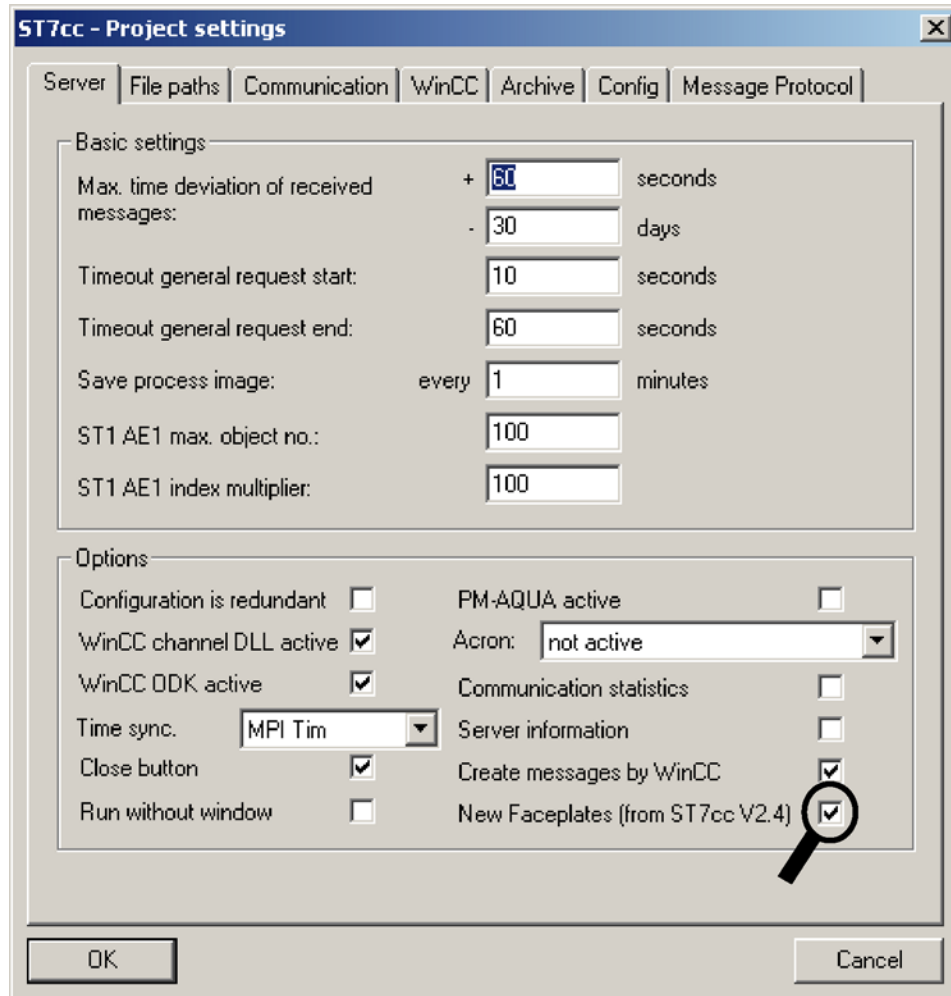


Figure 4-22 Enabling the use of the picture faceplates of version V 2.4 or higher (no check mark = disabled)

Case B:

You already have a WinCC project with pictures and picture typicals for the ST7cc subscribers and want to use ST7cc version V2.4.x / V2.5 / V2.6 (system typicals, picture typicals, and faceplates).

In this case, the following actions are necessary:

1. Delete the ST7cc system variables in WinCC.
2. If you select this strategy, enable the New Faceplates (from ST7cc V2.4) option in the Server tab of the ST7cc Project Settings dialog (see figure).

3. Replace the old system typicals in our existing library with the system typicals of ST7cc version V2.4.x, V2.5 or V2.6.
4. Update the Subtype parameter for every subscriber in your ST7cc object tree using ST7cc Config to indicate whether this is a local TIM or a station subscriber (CPU).
5. Replace the old picture typicals in your process pictures with the new picture typicals of ST7cc version V2.4.x, V2.5 or V2.6 (see also section Generating technological picture objects (Page 270)), since the old picture typicals will no longer be correctly supplied with values.
6. Re-generate your WinCC tag management so that the WinCC tags are created for the new system variables.

Note

If you want to display expanded information for stations and local TIMs, all the local TIMs must be upgraded to firmware V4.x. Otherwise, only a subset of the information will be displayed.

Case C:

If you upgrade your existing project with system typicals of ST7cc version V2.4 to ST7cc version V2.5 without wanting to connect a TIM 3V-IE / TIM 4R-IE as local TIM over Ethernet, you do not need to do anything. ST7cc V2.5 continues to supply the picture typicals of version V2.4 with values correctly.

Case D:

You already have an existing project with ST7cc version V2.4.x and want to connect a TIM 3V IE / TIM 3V IE Advanced / TIM 4R-IE as local TIM over Ethernet to your SINAUT system. To obtain the new status information from a TIM 3V-IE, you need to use the system typicals and faceplates of ST7cc version V2.5 or higher. In this case, the following actions are necessary:

Replace the system typicals of version V2.4.x in your existing library with the system typicals of ST7cc version V2.5, V2.6 or V2.7 and use the new picture typicals and faceplates for the new SINAUT subscribers you introduce into your project. You only need to replace the picture typical existing in the process picture with the new picture typical if you replace a TIM 3x with a TIM 3V-IE / TIM 4R-IE.

Case E:

You already have a project with ST7cc version V2.5. If you do not want to use the new functions of version V2.6 or V2.7, you do not need to do anything.

If you want to use the new functions of version V2.6 or V2.7, you will need to take the following action:

1. Copy the new faceplates to your project.
2. Replace the previously used system typicals in your existing library with the system typicals of ST7cc version V2.6 or V2.7.
3. Re-generate your WinCC tag management so that the WinCC tags are created for the new system variables.

System typicals for stations (CPU in the station)

The figure shows the variables of the Station system typical. The system variables are created automatically based on this typical. They contain the essential status information of the SINAUT Station subscriber and allow the output of organizational commands.

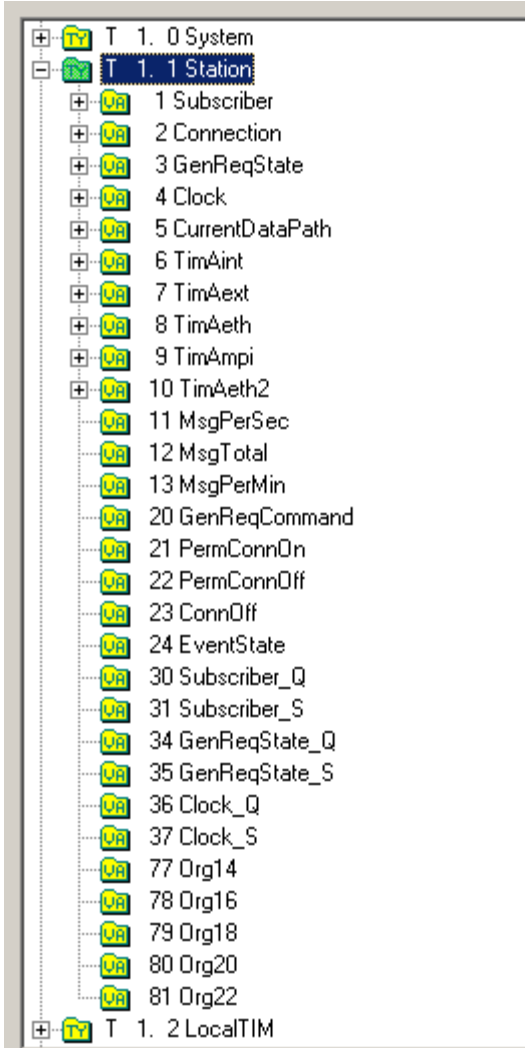


Figure 4-23 Variables of the Station typical

The following abbreviations are used in the table:

- 1. Column I/O:
 - I: Input. The SINAUT subscriber or the ST7cc server generates the information
 - O: Output. The WinCC application supplies the ST7cc variable. From this, the ST7cc server forms a message to the target component.
- 2. Column B (bit number):
 - Number of the bit set for status display.
- 3. Column C (class):
 - E: Error message. The transferred value represents a problem.
 - O: Operation message. The transferred value represents a correct operating state.

Table 4- 4 Variables of the Station typical

Attribute name	I/O	B	C	Explanation
Subscriber	I	0	I	Station not accessible
	I	1	I	Station not accessible over all paths
	I	3	O	Station accessible over all paths
Connection	I	1	O	Dial-up network: Connection establishment requested; dialing active, an attempt is being made to establish a connection (*).
	I	2	O	Dial-up network: Offline: There is currently no active connection to the station; in other words, the ST7cc variables of this station will not be updated (*).
	I	3	O	Dial-up network: Online: There is currently an active connection to the station; in other words, the ST7cc variables of this station will be updated (*).
	I	4	O	Permanent connection active:
				Dedicated line: The station is calling permanently.
				Dial-up network: There is a permanent connection to the station
GenReqState	I	1	O	General request (GR) was requested, reply message to GenReqCommand
	I	2	O	GR start: as the first response of the station to GenReqCommand
	I	3	O	GR end: Final message of the station, ST7cc starts to check whether all messages have arrived.
	I	4	I	GR start timeout: The ST7cc server did not receive a "GR Start" from the station during a configured time
	I	5	I	GR end timeout: The ST7cc server did not receive a GR end from the station during a configured time
	I	6	I	GR incomplete: On receiving GR end, the ST7cc Server checks whether it has received the data of all SINAUT objects. If this is not the case, it sets GR incomplete.
Clock	I	0	I	Time invalid: Station does not have SINAUT time.

Attribute name	I/O	B	C	Explanation
	I	1	O	Standard time
	I	2	I	Time invalid: Station does not have SINAUT time (identical to value 0).
	I	3	O	Daylight saving time
CurrentDataPath				The following displays are only supplied with the TIM firmware version V4.0. The values show the TIM port over which the current data path runs.
	I	1	O	Internal WAN interface:
	I	2	O	External WAN interface
	I	4	O	Ethernet interface
	I	5	O	Internal WAN interface / Ethernet interface
	I	6	O	External WAN interface / Ethernet interface
	I	8	O	MPI interface
	I	9	O	Internal WAN interface / MPI interface
	I	10	O	External WAN interface / MPI interface
TimAint				Detailed code of the connection statuses over the internal WAN interface.
	I	0	O	Connection terminated, no dial-up function active (dial-up network)
	I	1	O	Outgoing call initialized (dial-up network)
	I	2	O	Incoming call established (dial-up network)
	I	3	O	Outgoing call established (dial-up network)
	I	4	O	Permanent connection registered (dial-up network)
	I	5	O	Permanent connection registered and outgoing call initiated (dial-up network)
	I	6	O	Connection established and permanent connection canceled (dial-up network)
	I	7	O	Permanent connection established (dial-up network)
	I	8	O	free
	I	9	O	free
	I	10	O	No driver-specific status available
	I	11	O	free
	I	12	O	Call in main cycle (dedicated line)
	I	13	O	Call in subcycle (dedicated line)
	I	14	O	Permanent call in main cycle (dedicated line)
	I	15	O	Permanent call in subcycle (dedicated line)
	I	16	O	No driver-specific status available
	I	17	O	No driver-specific status available
TimAext	I			As for TimAint however for the external WAN interface
TimAeth	I			As TimAint
TimAmpi	I			As TimAint
MsgPerSec	I			Number of messages received in the last second (from station to ST7cc)

4.4 Configuring

Attribute name	I/O	B	C	Explanation
MsgTotal	I			Total number of messages received since the server started up (from station to ST7cc)
MsgPerMin	I			Number of messages received in the last minute (from station to ST7cc)
GenReqCommand	O	1		Trigger a general request
PermConnOn	O	1		Request a <i>permanent call</i> or a <i>permanent connection</i> to station
PermConnOff	O	1		Cancel a <i>permanent call</i> or a <i>permanent connection</i> to station
ConnOff	O	1		Immediate abort of an existing dial-up connection
EventState	I			Group display variable. The bit assignment matches that specified by PCS 7 (see Section Group display (Page 174))
Subscriber_Q	I			Acknowledgment variable for Subscriber variable (see above), created in WinCC as an internal tag.
Subscriber_S	I			Acknowledgment status variable for Subscriber variable (see above)
GenReqState_Q	I			Acknowledgment variable for GenReqState variable (see above), created in WinCC as an internal tag.
GenReqState_S	I			Acknowledgment status variable for GenReqState variable (see above)
Clock_Q	I			Acknowledgment variable for the Clock variable (see above) is created in WinCC as an internal tag.
Clock_S	I			Acknowledgment status variable for Clock variable (see above),
Org14	I			For internal processing
Org16	I			For internal processing
Org18	I			For internal processing
Org20	I			For internal processing
Org22	I			For internal processing

(*) As of TIM firmware version V4.3

System typical for local TIMs

These are the TIMs connected locally over the MPI bus or Ethernet to the ST7cc PC. The figure shows the variables defined in the typical.

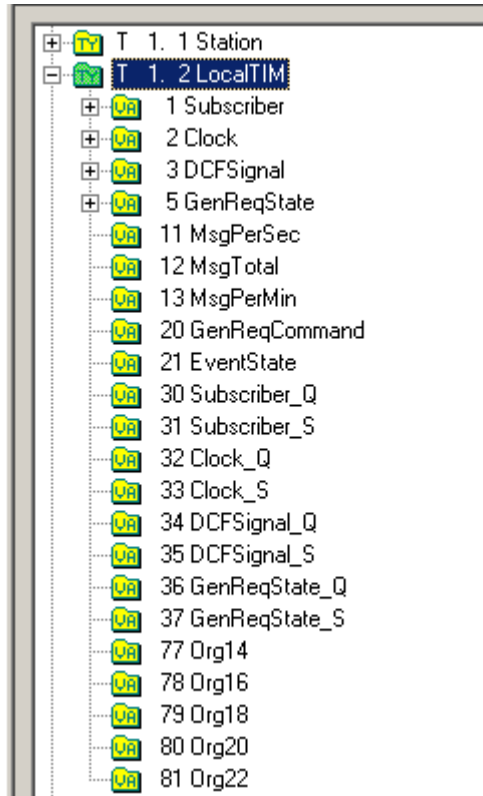


Figure 4-24 System typical for TIM in the control center

To allow you to interpret and evaluate the content of the system variables in your client application, the variables are described briefly below. The following abbreviations are used in the table:

- 1. Column I/O:
 - I: Input. The SINAUT subscriber or the ST7cc server generates the information
 - O: Output. The WinCC application supplies the ST7cc variable. From this, the ST7cc server forms a message to the target component.
- 2. Column B (bit number):
 - Number of the bit set for status display.
- 3. Column C (class):
 - E: Error message. The transferred value represents a problem.
 - O: Operation message. The transferred value represents a correct operating state.

Table 4- 5 System typical for TIM in the control center

Attribute name	I/O	B	C	Explanation
Subscriber	I	0	I	TIM not accessible
	I	1	I	TIM not accessible over all paths
	I	3	O	TIM accessible over all paths
Clock	I	0	I	Time invalid: TIM does not have SINAUT time.
	I	1	O	Standard time
	I	2	I	Time invalid: TIM does not have SINAUT time (same as with value 0).
	I	3	O	Daylight saving time
DCF signal	I	0	O	TIM has no radio clock interface
	I	1	I	No time signal received, validation not yet completed.
	I	2	O	TIM has no radio clock interface
	I	3	I	No time signal received, validation not yet completed.
	I	4	O	TIM has no radio clock interface
	I	5	O	Time signal correctly received. Validation OK.
	I	6	O	TIM has no radio clock interface
	I	7	O	Time signal correctly received, validation OK
TimBus	I			Not currently available.
MsgPerSec	I			Number of messages received in the last second (from TIM to ST7cc)
MsgTotal	I			Total number of messages received since the server started up (from TIM to ST7cc).
MsgPerMin	I			Number of messages received in the last minute (from TIM to ST7cc)
GenReqCommand	O	1		Trigger a general request
EventState	I			Group display variable. The bit assignment matches that specified by PCS 7 (see Section Group display (Page 174))
Subscriber_Q	I			Acknowledgment variable for the Subscriber variable (see above) is created in WinCC as an internal tag.
Subscriber_S	O			Acknowledgment status variable for Subscriber variable (see above)
Clock_Q	I			Acknowledgment variable for the Clock variable (see above) is created in WinCC as an internal tag.
Clock_S	O			Acknowledgment status variable for Clock variable (see above),
DCFSignal_Q	I			Acknowledgment variable for the DCFSignal variable (see above) is created in WinCC as an internal tag.
DCFSignal_S	O			Acknowledgment status variable for DCFSignal variable (see above)

Attribute name	I/O	B	C	Explanation
GenReqState_Q	I			Acknowledgment variable for the GenReqState variable (see above) is created in WinCC as an internal tag.
GenReqState_S	O			Acknowledgment status variable for GenReqState variable (see above)
Org14	I			For internal processing
Org16	I			For internal processing
Org18	I			For internal processing
Org20	I			For internal processing
Org22	I			For internal processing

4.4.5 Creating a user typical

Overview

A typical is a model decoding with mechanisms allowing features to be inherited. The model decoding can relate to the entire data area of a SINAUT object. Generally, however, a typical usually relates only to a data subarea of an object data area that requires decoding as a commonly occurring data structure.

Typicals are created and managed in the ST7cc library.

For detailed information on the topic of typicals, refer to sections Object templates and typicals (Page 168) and Principle of decoding using typicals (Page 171).

Note

The following example is designed for a pump of any type A. All operating states that represent a normal pump state should be stored in a data subarea with a length of 8. The data subarea is mapped to a variable WinCC attribute name *Status*. The meaning of the individual bits is described in the message blocks of the variables. This strategy was selected to minimize the number of WinCC tags. If every bit (status information) of a data subarea was mapped to a variable, this would lead to an unnecessarily large number of WinCC tags (external variables).

In the form in which the ST7cc library is supplied, other user typicals can be included and do not need to match the example described here.

Creating a typical

To create a typical, first create the typical object, in which you later create the variables. Follow the steps outlined below:

1. Select the ST7cc library with the right mouse button and open the context menu (see figure).
2. Select the *New Typical* option.
The *Add Typical* dialog opens

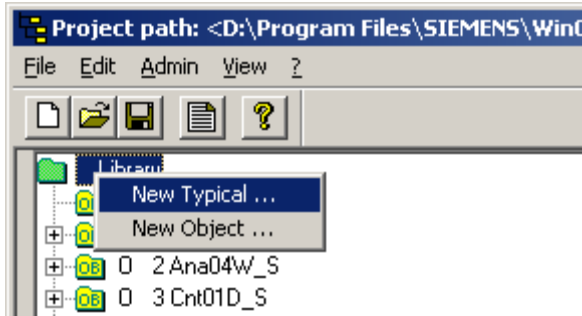


Figure 4-25 Generating a typical frame

3. Enter the typical ID consisting of *type* and *sub type* in the input boxes of the *Add Typical* dialog. Both entries are numbers.
The number of the type must be > 100 and the number of the sub type must be < 9.

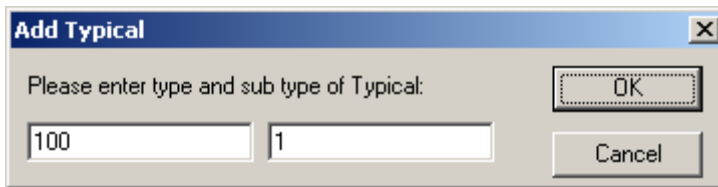


Figure 4-26 Entering the typical ID

4. Confirm the entered typical ID with *OK*.
The new typical is created as a data object and is displayed in the object tree (see figure).

5. Select the newly created typical in the object tree.
The *Name* input field is displayed in the window where you enter the name of the typical.

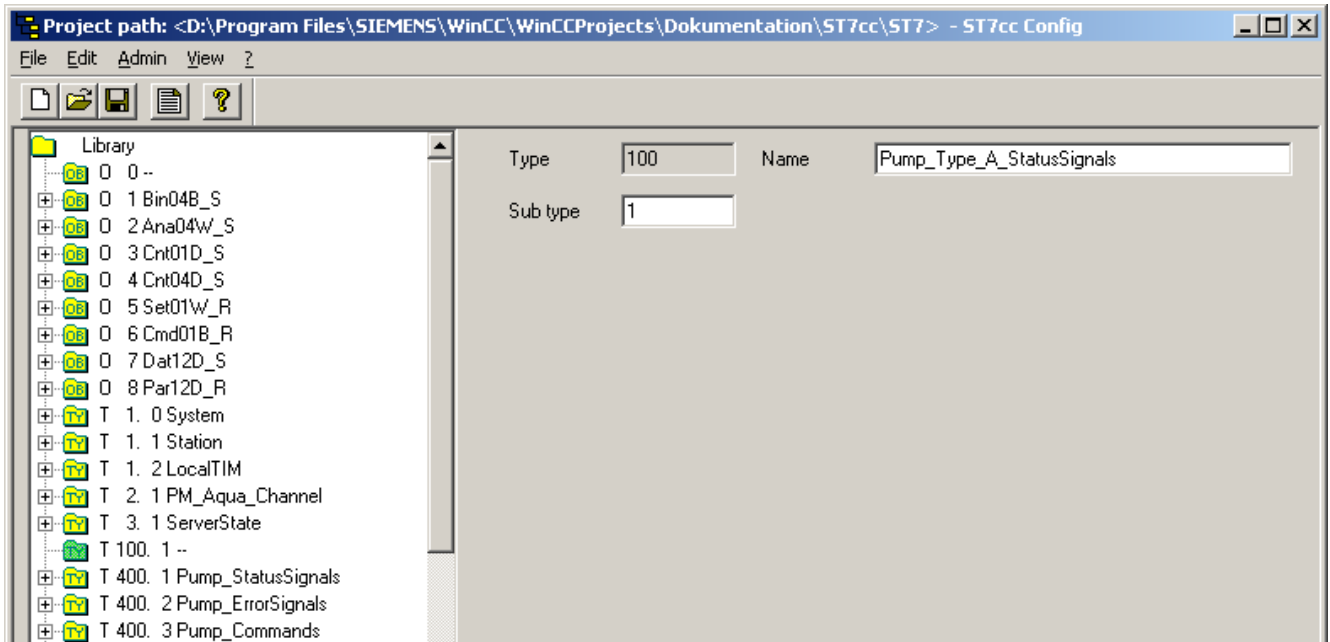


Figure 4-27 ST7cc Config dialog

6. Enter the typical name in the *Name* input field. The name of the typical is not required in any other processing. Enter a name that indicates the application of the typical.
7. Click *Apply changes*. This enters your data. The typical name you entered is visible in the object tree.

Creating a variable in the typical

Once the typical has been created in the object tree, you can create the variables of the typical and its message processing in later steps.

1. Select the typical to be edited with the right mouse button and open the context menu (see figure).



Figure 4-28 Creating a typical variable

2. Select the *New Variable* option. The *Add* dialog opens.

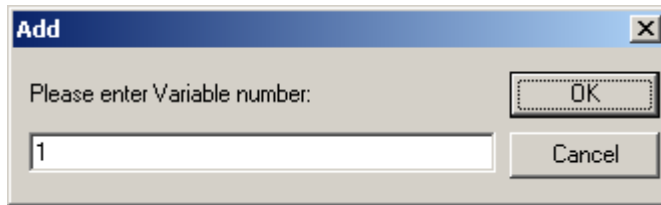


Figure 4-29 Entering the variable number

3. Enter the number of the variable (simply a management number, 1 to 9 999) in the input box of the *Add* dialog, and confirm with *OK*. The variable becomes visible in the object tree (see figure).

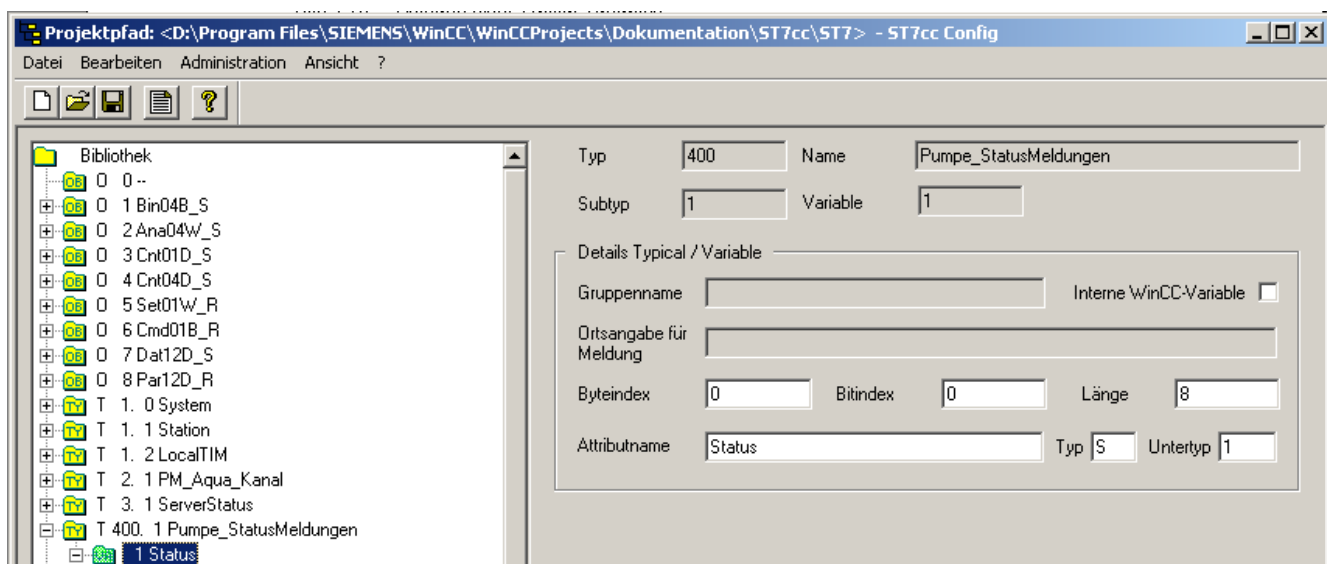


Figure 4-30 Input boxes for defining a variable

4. Left-click on the variable you want to edit in the object tree (see figure). Once you have selected a variable, the input fields for full definition of the variable are displayed.
5. With the *Internal WinCC tag* check box, you can decide whether this tag is created as an internal tag in WinCC when the WinCC tags are generated. If you do not select the check box, the tag is created as an external tag in WinCC.
6. Enter the attribute name as part of the variable name. The attribute name must not contain a period or blanks.
7. Enter the byte index, the bit index and the length to define the number of bits in the data subarea of the object data area that will be mapped to this variable. The length of a data subarea must be greater than 0, but a maximum of 32 bits.

Section SINAUT object types (Page 159) contains a description of the object types that gives you an overview of the data structures of the ST7 object types.

8. Complete the variable definition by entering the type in the Type field and the sub type in the Sub type field of the previously defined data subarea. By entering the type and sub type, you specify how the data subarea will be converted during decoding.

For a detailed description of the type and sub type information, refer to section Type and subtype of a variable (Page 165).

9. Click Apply changes. This enters your data.

Creating a message processing function for a variable


With ST7cc version V2.7, you have two methods (old and new) available for generating a message number. Variables can be generated directly within a typical and within an object.

Case 1, variable within a typical:

When you create the message processing for a variable of a typical, you always enter the number of the message block explicitly.

Case 2, variable within an object:

When you enter the message processing for an object variable, you also enter a number for the message block when working with the old method (structure-oriented). With the new method (offset-oriented), the consecutive message number is assigned automatically when you create the message block. Step (3) of the sequence described below is then omitted.

Once the variable is defined, you can create one or more message processing functions (message blocks ) for the variables.

1. Select the variable to be edited with the right mouse button and open the context menu. (see figure)

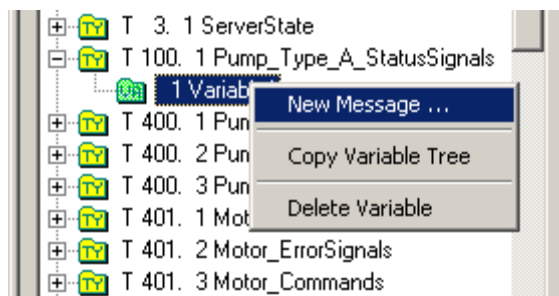


Figure 4-31 Creating a message processing function

2. Select the *New Message* option. The *Add* dialog opens.

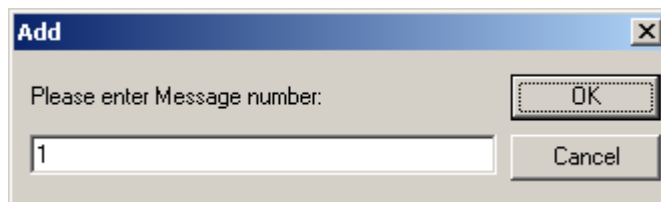


Figure 4-32 Entering in the number of the message processing function

4.4 Configuring

3. Enter the variable-specific number (1 to 99) of the message processing function / message block in the input box of the *Add* dialog and confirm with *OK*. With this confirmation, the parameter block for the message processing function is created organizationally. (see figure)
4. Select the message block you want to edit with the left mouse button. The *Message block details* dialog area displays the input boxes for assigning parameters to the message processing function - basic function. (see figure)

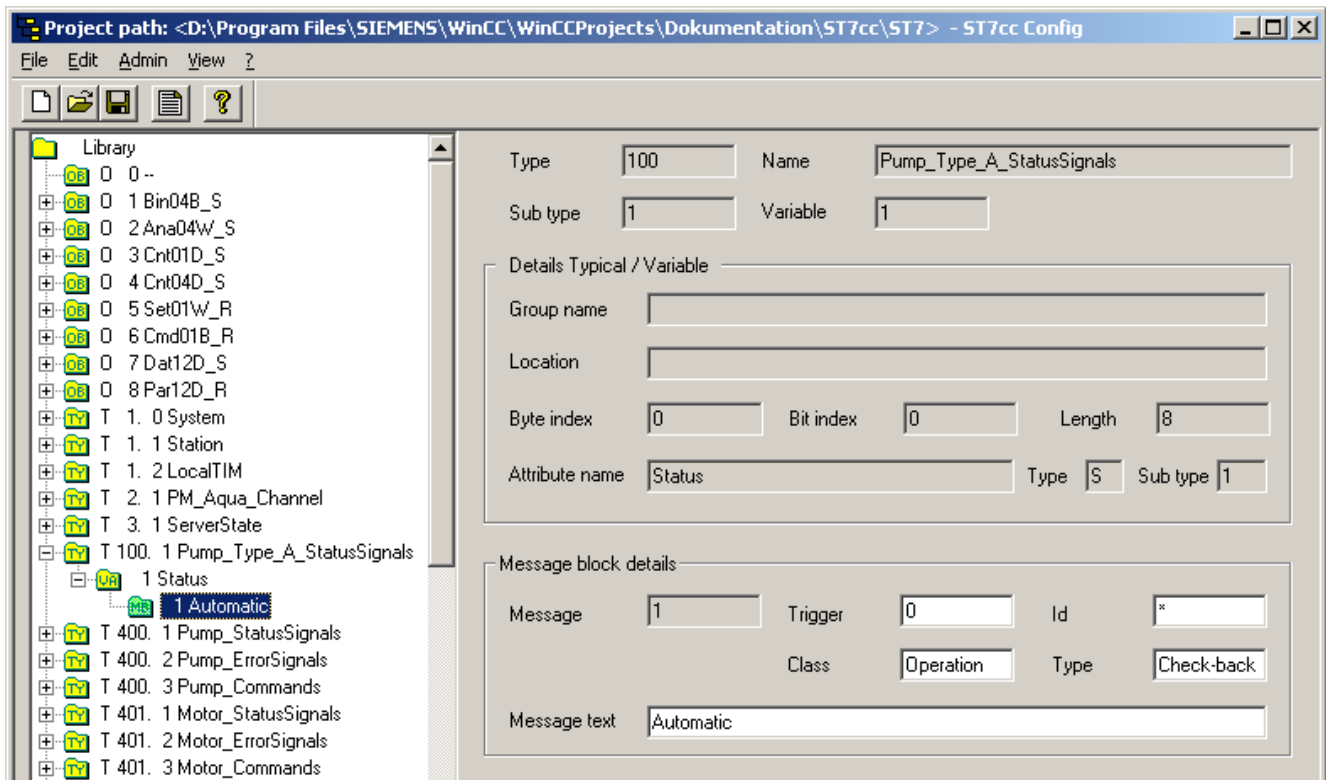


Figure 4-33 Input boxes for parameter assignment of the message processing function (message block)

Assigning parameters for the message processing function is described in the section Message processing (Page 235).

Converting the message block number of a typical into message numbers

With the old (structure-oriented) method for generating message numbers, follow the procedure as described in the section Project settings: Config (Page 142).

With the new (offset-oriented) method for generating message numbers, 99 numbers are reserved starting at the first free number so that the numbers calculated from the typical instance + message block number can be mapped to this band of numbers. With this method, the message numbers are continuous but not without gaps. This is an important point to remember so that gaps in the otherwise consecutive band of message numbers are not interpreted as "lost message numbers".

Note on using a typical

Remember that a typical is a partial decoding and can be used more than once within a decoding of an object data area. For example, a Bin04B_S can contain the status information of several pumps or an Ana04W_S can contain four analog values.

The byte index entered in the typical is a relative address specification of the data subarea to be processed for the subsequent application (instantiation) within a decoding. The exact position in the decoding is decided by the *Offset* parameter that, along with the byte and bit index, identifies the data subarea to be decoded.

Based on two examples, the paragraphs below describe how the content of a data subarea of a SINAUT object is mapped on variables. In example 1, each bit of the data subarea in question is mapped to a variable. In example 2, a data subarea is mapped to a single variable. The meaning of the individual bits of the data subarea can then only be recognized in ST7cc Config in the message processing functions of the variable.

Example 1:

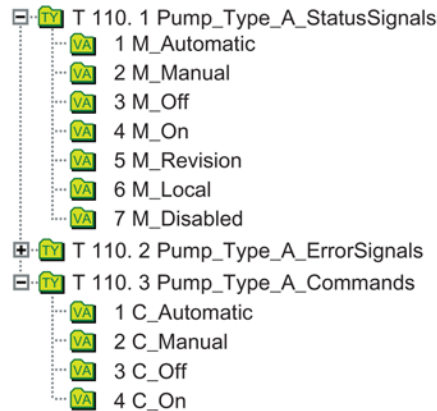


Figure 4-34 User typical in the project library

The figure shows two typicals (of the ST7cc library) with the variable descriptions for a pump of type A. Typical no. 110 forms the logical parenthesis. The typical 110.1 contains the variable definitions for the statuses that represent regular operation, the typical 110.2 contains the variable definitions for the statuses presenting a problem, the typical 110.3 contains the variable definitions for the commands.

The attribute names were selected so that they are distinguished by their prefixes M_ and B_. When the typicals are used later (instantiation) to decode a pump, for example ST23_Pump1, the same group name ST23_Pump1 can be used for decoding in the monitoring direction (Bin04B_S) and the control direction (Cmd01B_R). Selecting different attribute names guarantees the uniqueness of the ST7cc / WinCC variable name in this case.

The disadvantage of this procedure is that a variable (external variable) is required for each pump state.

Example 2:

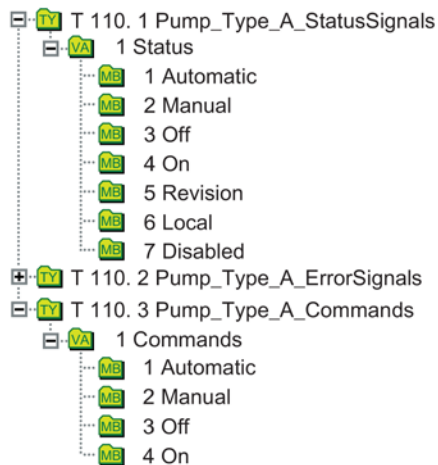


Figure 4-35 User typical in the project library

In this case, only three variables are required to describe all the information (statuses and commands). The significance of the individual bits of the data subareas of the SINAUT object is stored in ST7cc Config only in the message texts of the message processing functions of the variables. In a decoding with the typicals 100.1 and 100.3, a message will be generated and displayed in WinCC if the command to turn on a pump is output and when the pump state has changed to "leaving state".

If the user does not want to generate messages and does not create any message processing functions, the meaning of the individual bits of the decoded data subareas does not exist in ST7cc Config.

4.4.6 Setting up a subscriber

Setting up a subscriber

This section describes how to set up SINAUT subscribers in ST7cc Config.

SINAUT subscriber

SINAUT subscribers for ST7cc are:

- The ST7cc server (subscriber 0 System)
The *System* subscriber already exists in the project after you create the ST7cc project and you do not need to set it up. The system variables defined in the *System* system typical (see Section System typicals (Page 184)) are generated automatically.
- The CPUs of the stations:
You need to set up every SINAUT station. When you set up a station, the system variables defined in the *Station* system typical are created automatically (see section System typicals (Page 184)).
- The TIMs connected locally over the MPI bus or Ethernet to the ST7cc PC:
Each of these TIMs must be set up as a subscriber. When you set up a TIM, the system variables defined in the *LocalTIM* system typical are created automatically (see section System typicals (Page 184)).

Sub type

To inform the system of the subscriber type (CPU, TIM), you specify this when setting up the TIMs using the *Sub type* parameter. The following system typicals are relevant for ST7cc:

- Sub type 0: stands for system (ST7cc server)
- Sub type 1: stands for the Station subscriber
- Sub type 2: stands for the LocalTIM subscriber on the MPI bus or Ethernet.

Setting up a subscriber

To set up a new subscriber, follow the steps below:

1. Select the *Edit* button with the left mouse button and open the menu (see figure).
2. Select the *New Station* option if a CPU subscriber is involved or *New Local TIM* if a TIM subscriber is involved.
The *Add* dialog opens.

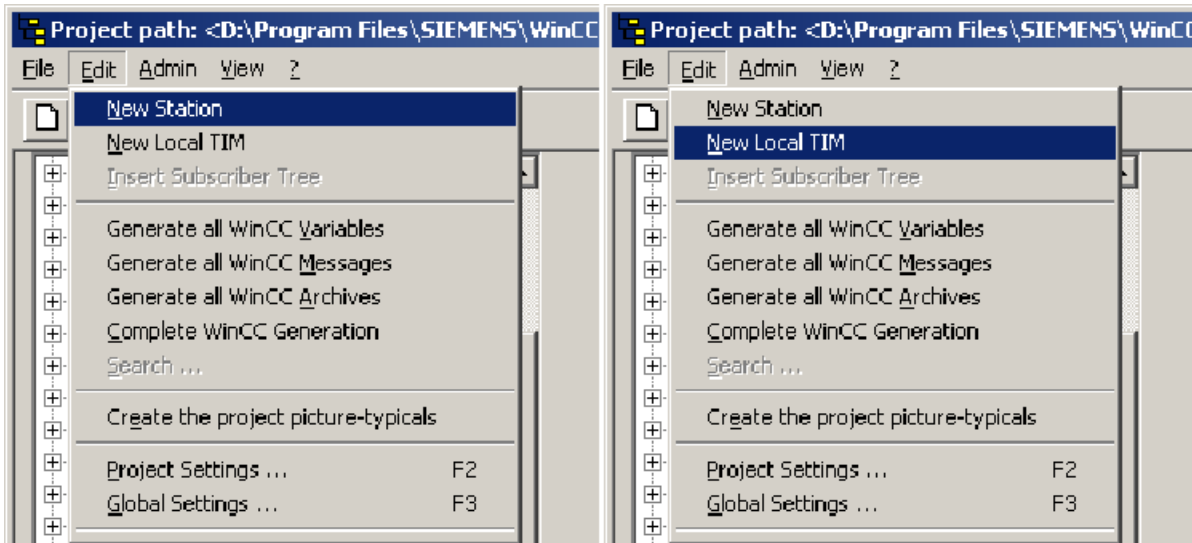


Figure 4-36 Setting up a SINAUT subscriber

3. Enter the subscriber number of the SINAUT subscriber you are setting up in the input field of the *Add* dialog and confirm your entry with *OK* (see figure).

Note

The SINAUT subscriber numbers of the TIMs and the stations (CPU) can be found in the corresponding STEP 7 project. You will also find further information in the TD7 block list (see section SINAUT TD7 block structure in ST7cc Config (Page 259)).

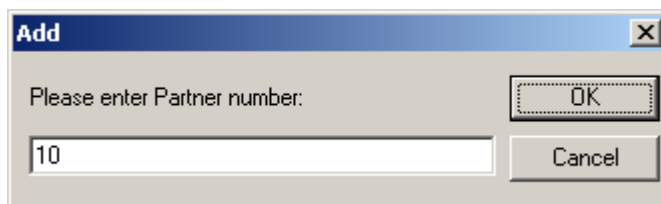


Figure 4-37 Dialog for entering the subscriber number

After entering the subscriber successfully, the new subscriber is displayed in the object tree (see figure).

1. Select the subscriber. With the selection, input boxes are displayed in which you can enter the subscriber name and the subtype (see figure). The input boxes have default entries.
2. Enter the subscriber name.

Note

Remember that the subscriber name will be used as the group name in the ST7cc system variable name. You should select the combination of subscriber name and attribute name of the variables in the system typical so that the variable name complies with your conventions.

3. Enter the type of your subscriber in the *Sub type* input field. Subtype 1 stands for station, subtype 2 for local TIM.

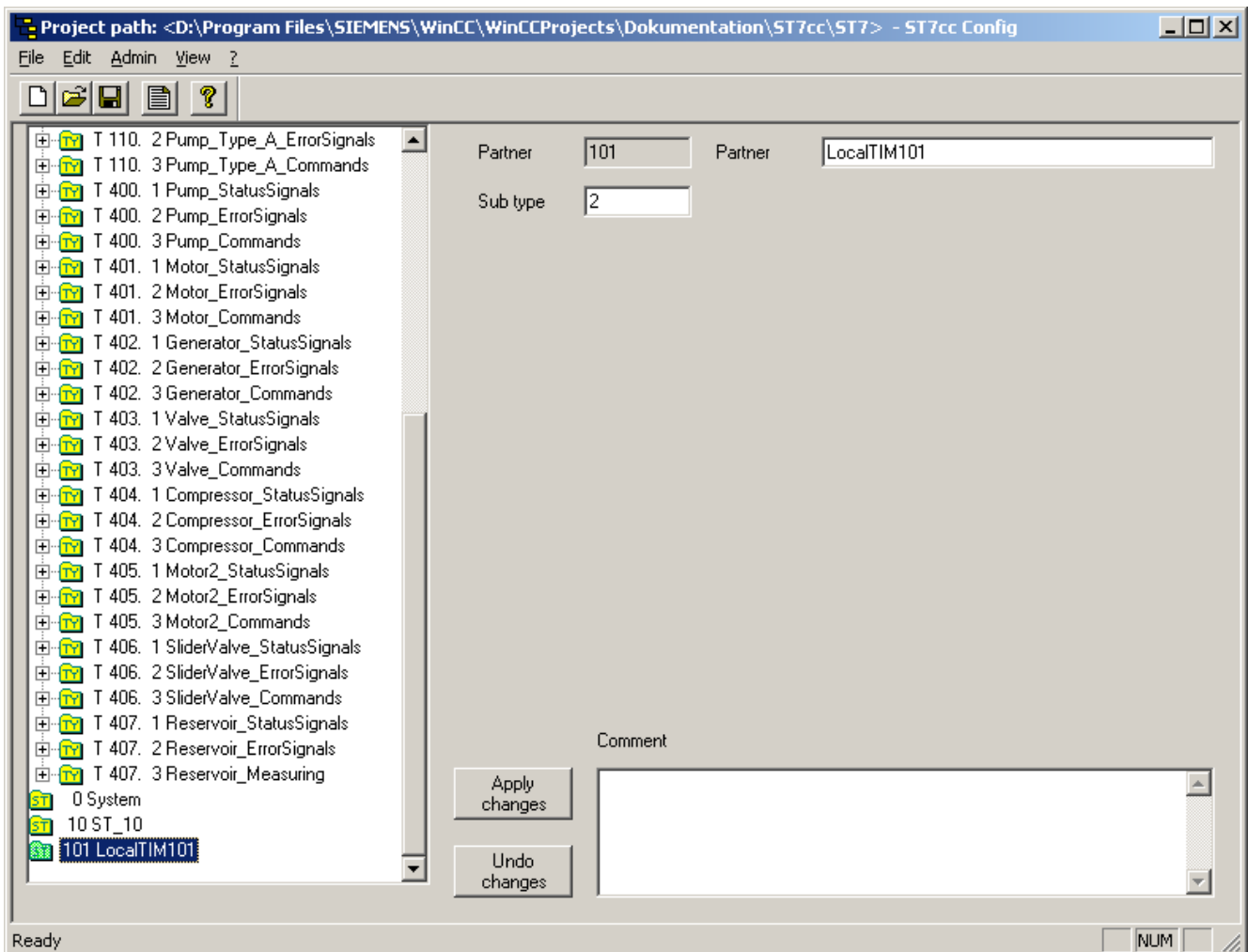


Figure 4-38 ST7cc object tree with newly set up subscriber

4.4 Configuring

- In the input fields internal WAN IF, external WAN IF, Ethernet IF and MPI bus IF enter the attribute names you want to use in the station faceplate (see section Picture typicals and faceplates for a station (Page 306)) to identify the interfaces. These parameters can only be entered for stations (subtype 1).

Attribute name	Explanation
Internal WAN IF	Internal modem interface of the TIM
External WAN IF	External modem interface of the TIM
Ethernet IF	Ethernet interface of the TIM
MPI bus IF	MPI bus interface of the TIM

The attribute names relate to the interfaces of the master TIM over which ST7cc communicates with the station. Depending on which transmission network is connected to these interfaces of the master TIM, the attribute names could be changed, for example to "Dedicated line S34-1", "Phone network", "Wireless network south" or similar, in other words to names that have more meaning for the user than the standard attribute names. These network-specific names are intended to make the information displayed in the station faceplate more readable for the operator. Particularly when a station is connected over redundant paths, it is easier for the operator to recognize which network the station is currently using for communication and, in the case of a disruption, which of the two networks can no longer be used to reach the station.

Although there are input fields for 4 attribute names, only the name of the interface over which the station is actually connected to the master TIM needs to be changed. Two interface names would only need to be changed when redundancy is being used.

- Click on Apply changes to save your data.

After successfully setting up a CPU subscriber, you can start to create the decodings for this subscriber.

4.4.7 Creating a decoding

Creating a decoding

This section shows you how to create a decoding in ST7cc Config.

Note

The decodings for SINAUT objects are created under their subscribers whereas new object templates are created in the library. Apart from minor differences that will be mentioned in the individual dialog steps, creating a decoding for an object template is the same as creating a decoding of a SINAUT object.

Follow the steps below for decoding:

1. Right-click on the subscriber under which you want to create a decoding and open the context menu (see figure).
2. Select the *New Object* option. The *Add* dialog opens.

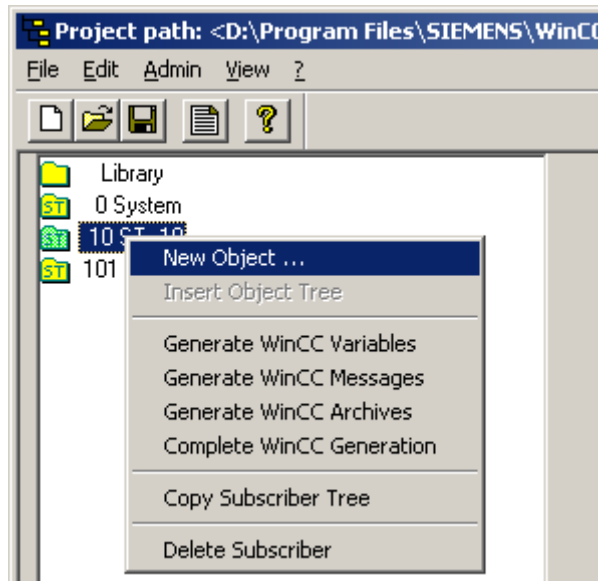


Figure 4-39 Selecting a subscriber to create a decoding.

3. Enter the object number of the SINAUT object for which you want to create the decoding in the *Add* dialog (see figure).

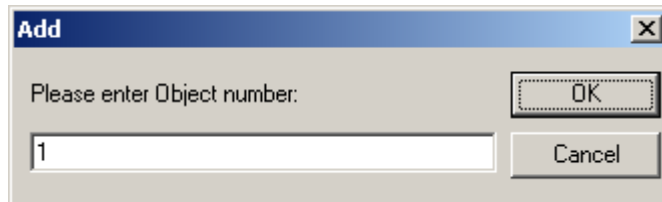


Figure 4-40 Entering the object number

Note

The SINAUT object number is identical to the number of the instance DB that was configured for the object to be decoded in the CPU program of the station. You will also find further information in the TD7 block list (see section SINAUT TD7 block structure in ST7cc Config (Page 259)).

4. Confirm the entered object number with OK. After it has been entered, the decoding is displayed in the object tree (see figure).

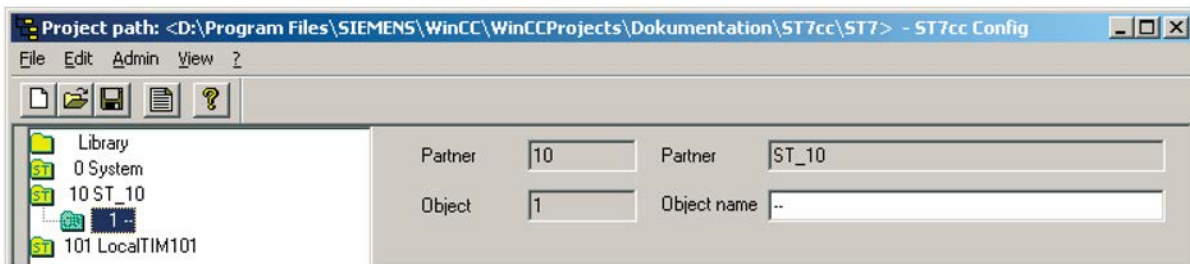


Figure 4-41 New decoding in the object tree

5. Select the decoding. The input field for entering the object name is displayed. The input box has the default entry -- (see figure). The entry of the object name is optional and is not included in any processing.

Creating a variable

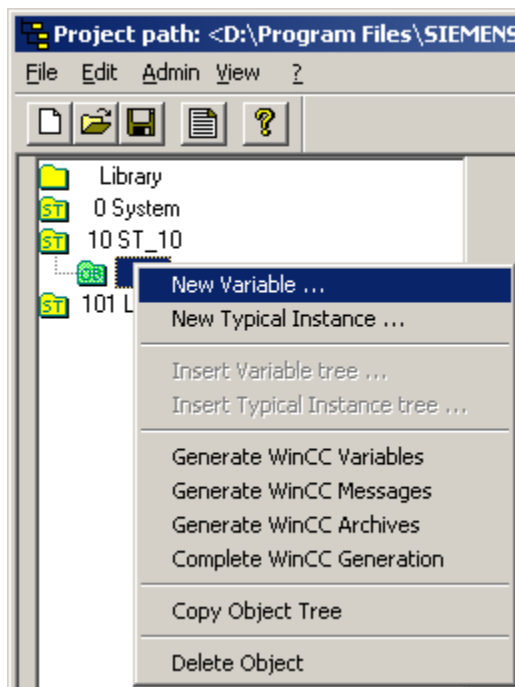


Figure 4-42 Creating a new variable

1. Right-click on the decoding and open the context menu (see figure).
2. Select the *New Variable* option. The *Add* dialog opens in which you can enter the variable number (see figure).

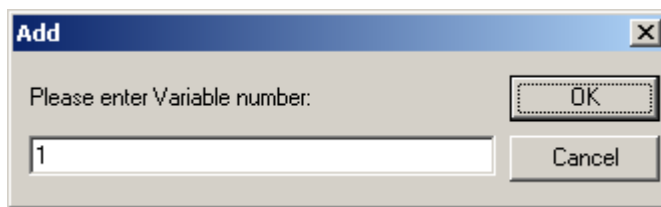


Figure 4-43 Input box for the variable number

3. Enter the variable number (see figure). The variable number is simply a management number within a decoding in the range from 1 through 9 999. It must be unique within the object to be decoded.
4. Confirm the entered variable number with *OK*.

On completion of the entry, the new variable is displayed within the decoding with the variable name Group.Variable1 (default). When you select the variable, the input boxes for the variable definition with the defaults is displayed (dialog area Details Typical / Variable).

Entering the Variable Definition

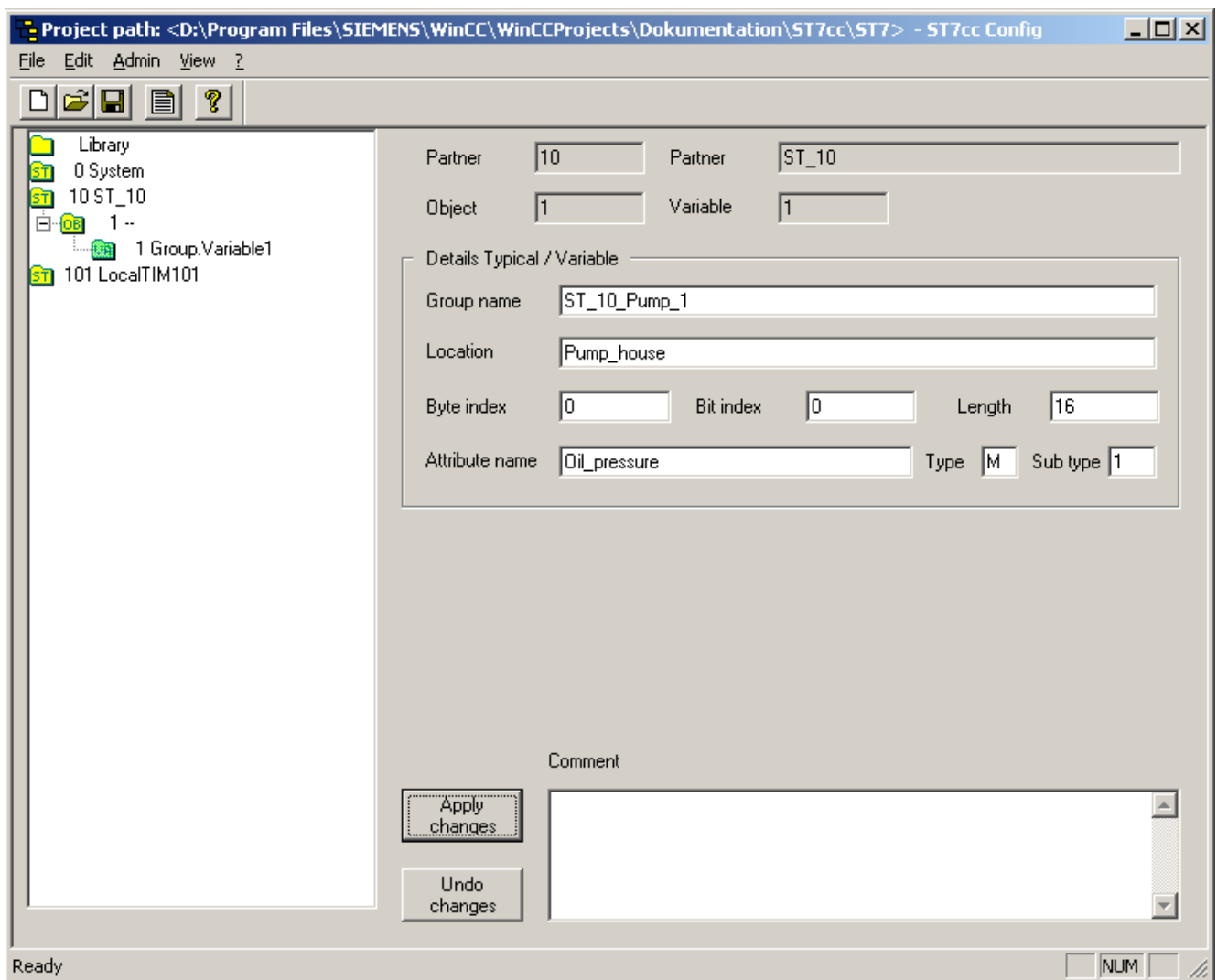


Figure 4-44 Input boxes for defining variables

The variable definition involves the entry of all parameters required to map a data subarea of an object data area on the variable.

When you select the variable, you will see the input fields for entering the parameters for the variable definition (see figure).

4.4 Configuring

To define the variables, follow the steps outlined below:

1. Enter the group name as part of the variable name. The group name must not contain the period or blanks (see section Variable name (Page 164)).
2. Enter the attribute name as part of the variable name. The attribute name must not contain the period or blanks (see also section Variable name (Page 164)).
3. With the Internal WinCC tag check box, you can decide whether this tag is created as an internal tag in WinCC when the WinCC tags are generated. If you do not select the check box, the tag is created as an external tag in WinCC.
4. Enter the byte index, the bit index and the length to define the number of bits in the data subarea of the object data area that will be mapped to this variable. The length of a data subarea must be greater than 0, but a maximum of 32 bits. Section SINAUT object types (Page 159) contains an overview of the data structures of the SINAUT object types.
5. Complete the variable definition by entering the type in the Type field and the sub type in the Sub type field of the previously defined data subarea. By entering the type and sub type, you specify how the data subarea will be converted during decoding. For a detailed description of the type and sub type information, refer to section Type and subtype of a variable (Page 165).
6. Click on Apply changes if the default Apply changes automatically was not activated (see section Project settings: Config (Page 142)). This enters your data.

Entering processing functions

One or more processing functions can be assigned to a variable. A variable of the type M was created in the figure. In the next steps, the variable is assigned, as an example, to a measured value processing function.

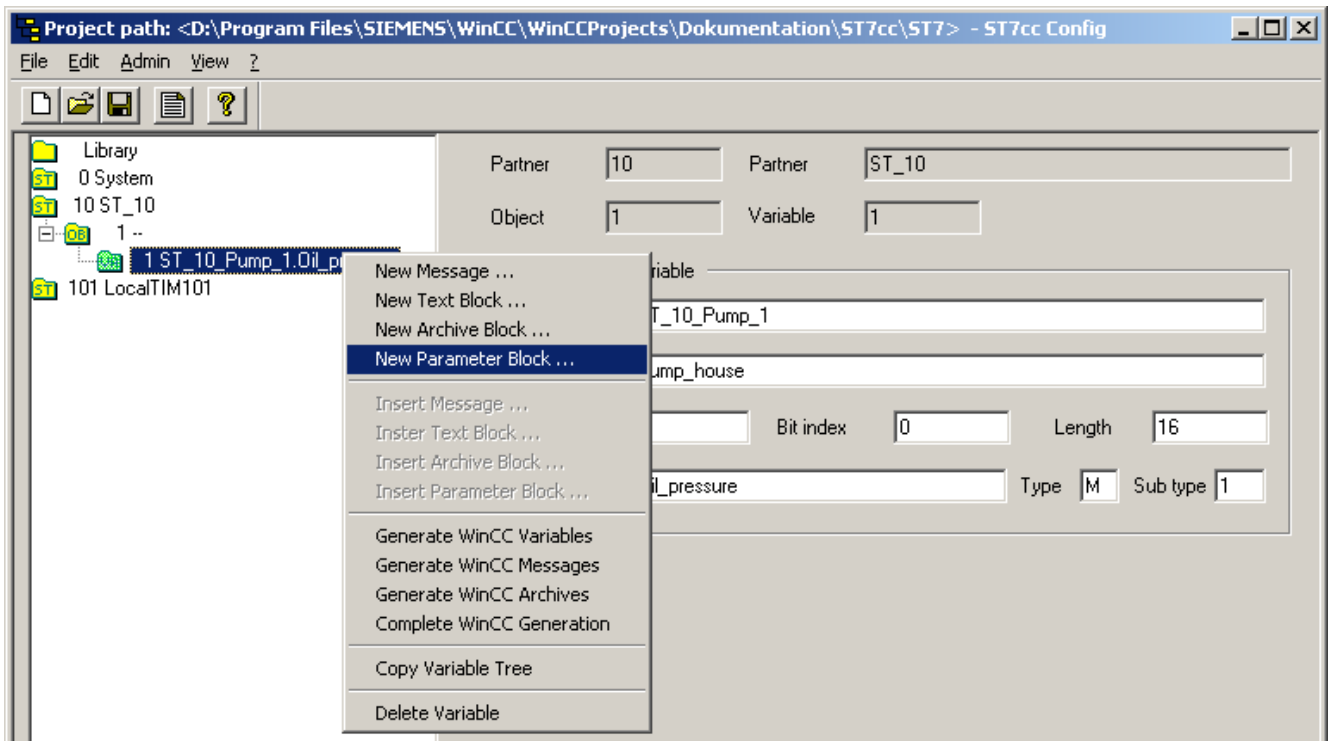


Figure 4-45 Creating a processing function

1. Select the required variable with the right mouse button and open the context menu (see figure). To create processing functions, the following options are available;

New Message: for message processing (basic function)

New Text Block: for entry of additional static texts for message processing.

New Archive Block: for archive processing

New Parameter Block: for measured value or counted value processing.

Depending on the type of variable selected, a measured value or counted value processing function is created.

2. Select the *New Parameter Block* option (see figure).

Due to its type *M*, measured value processing is automatically assigned to the variable. Since only one measured value processing function can be assigned to a variable, the parameter block is created immediately and is visible in the object tree.

When you select the processing function, you will see the input fields for entering the parameters (see figure). The parameters are described in section Measured value processing (Page 250).

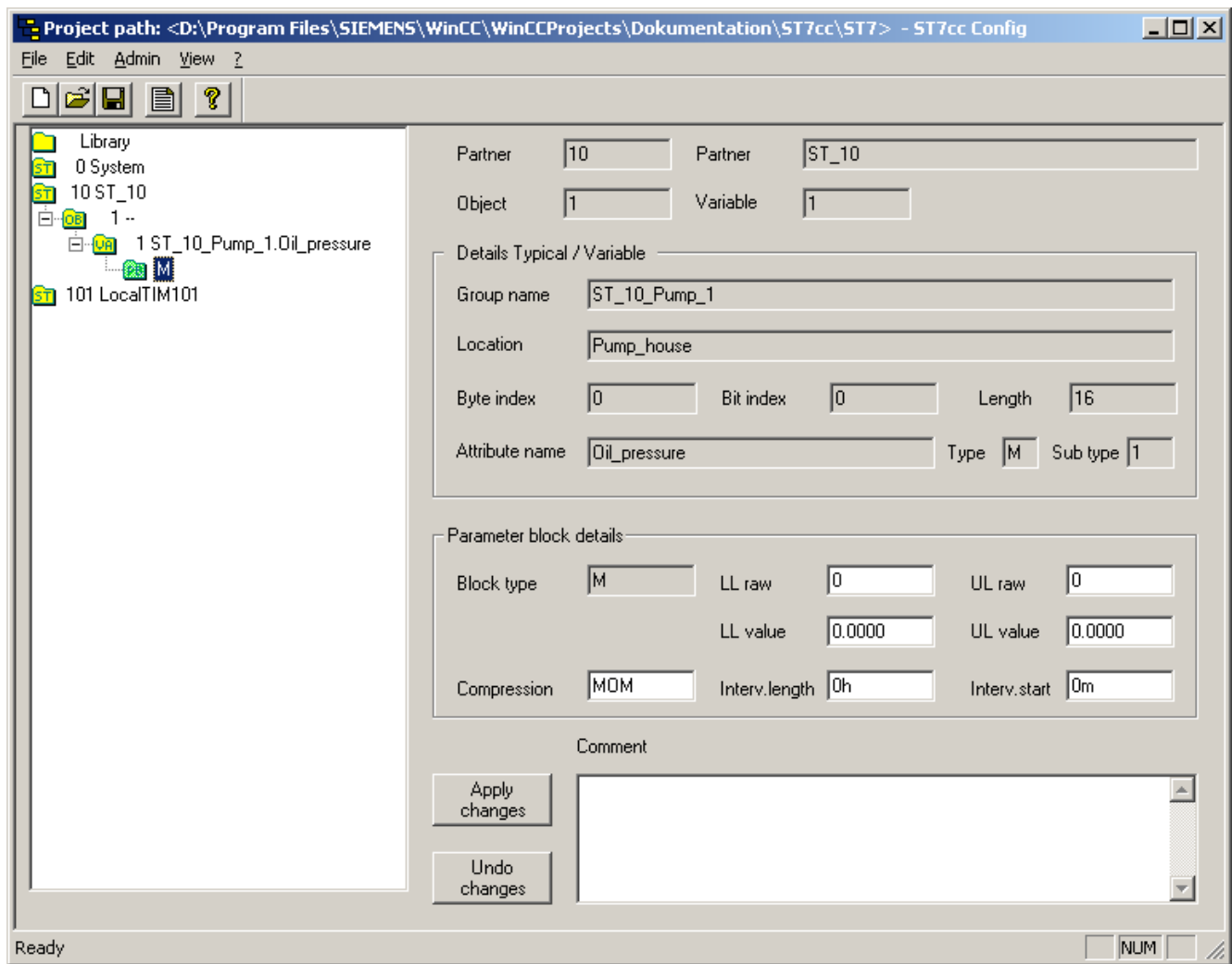


Figure 4-46 Parameter box of the measured value processing function

Copy functions

Copying decodings makes efficient engineering possible since all editing is included.

Note

When you copy, the name information is also copied 1:1. The user must update the name information after copying so that the names are once again unique.

Using typicals

The main feature of the method of creating decodings described above is that the variables are defined for each specific decoding. Each time a variable is changed, the user must select the variable via the decoding and enter the change.

By using typicals, however, a modification of the typical in the library is passed on to the objects affected.

4.4.8 Creating a decoding with typicals

In this section, we will show you how to use a typical in a decoding.

The status codes of two pumps will be included in a new decoding (decoding object no. 2). User typical 100.1 will be used and is instantiated twice. For further information, refer to sections Object templates and typicals (Page 168) and Principle of decoding using typicals (Page 171).

Instantiating from the library

1. Right-click on the decoding you want to edit and open the context menu (see figure).
2. Select the *New Typical Instance* option.

The Add Typical Instance dialog opens. Instantiation means that you use a typical created in the project library in a decoding. The typical instance is therefore a typical called in the decoding.

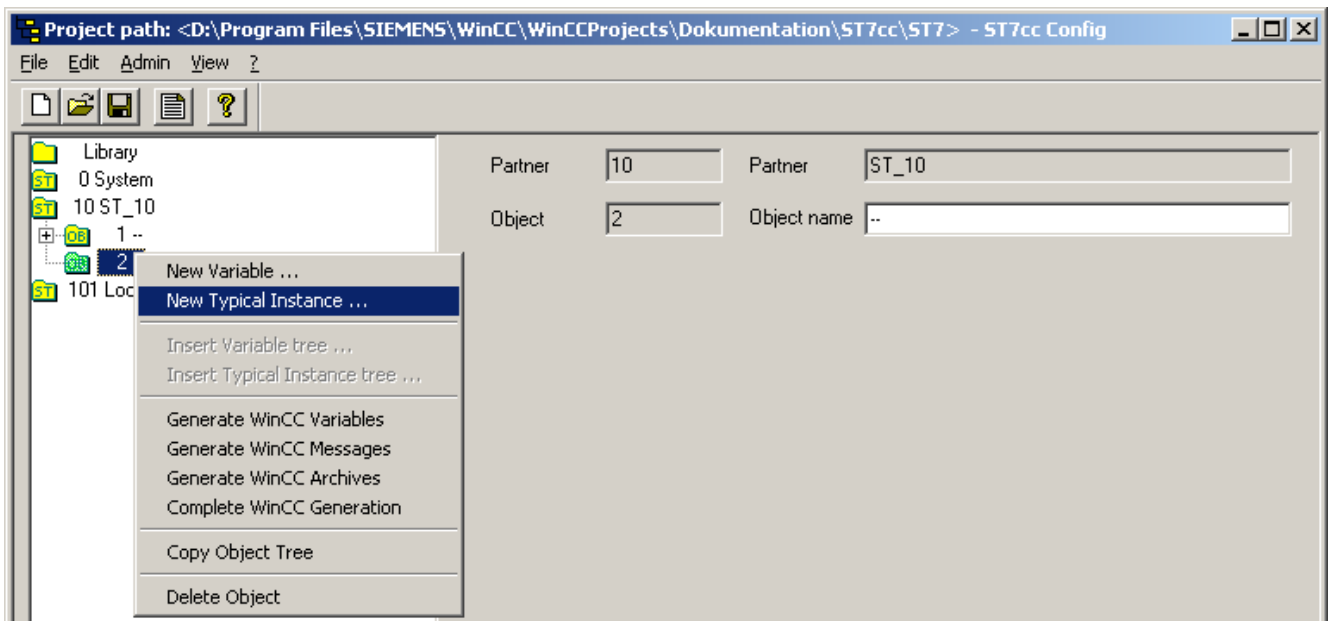


Figure 4-47 ST7cc Config dialog

Within a decoding, you can use typicals repeatedly. For each typical instance, you must assign a number that is unique within the decoding. Numbers from 1 through 99 are permitted.

3. Enter the instance number in the Add Typical Instance dialog (see figure).

4. Enter the type and sub type number with which you specify which typical you want to use and confirm your entry with OK.

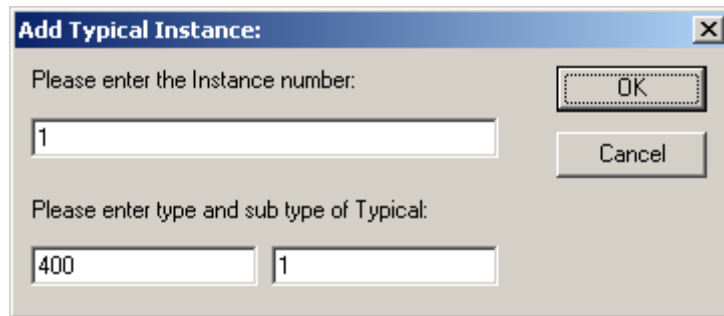


Figure 4-48 Dialog for selecting typicals

After confirming with OK, the instantiated typical is displayed in the ST7cc Config dialog (see figure) with the default group name Instance1.

To continue editing, select the typical instance you created. The ST7cc Config dialog displays the input boxes for the other parameters (see figure).

5. Enter the group name in the *Group name* input field.

By entering the group name, the variable names of all the variables of the typical are completed. In the example (see figure), the group name ST_10_Pump1 was selected.

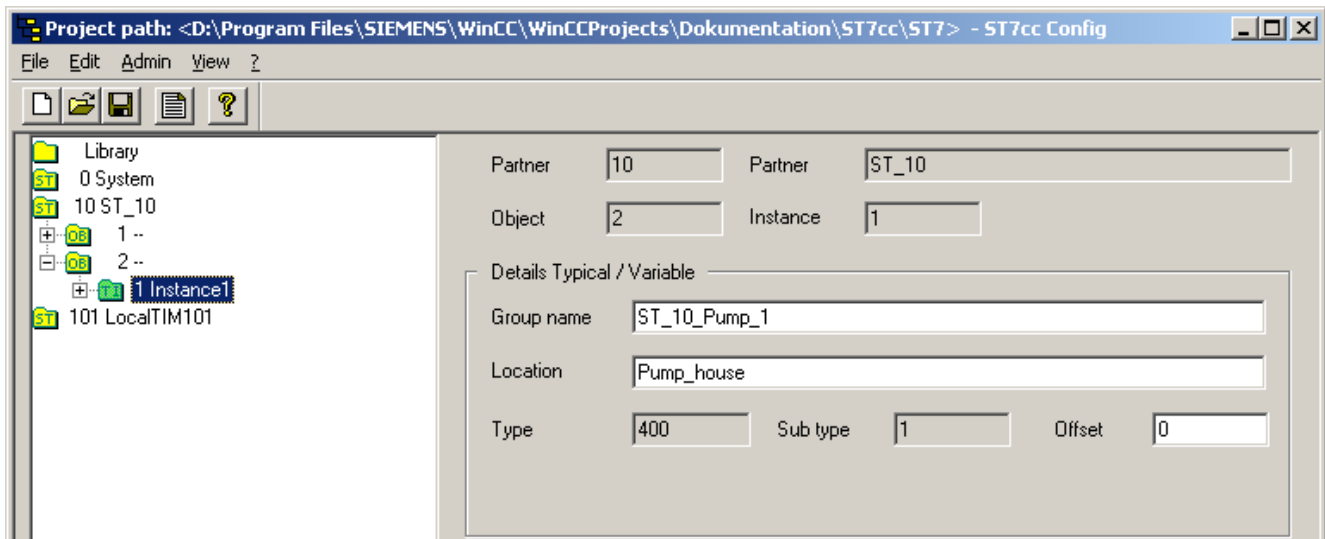


Figure 4-49 ST7cc Config dialog

6. Enter the parameter offset in the Offset input box. The parameter is used to specify the byte index at which the data section to be decoded by the typical begins. Compare the explanations in sections Object templates and typicals (Page 168) and Principle of decoding using typicals (Page 171).

If you select a typical instance by double-clicking on it, you obtain further detailed information on the instantiated typical.

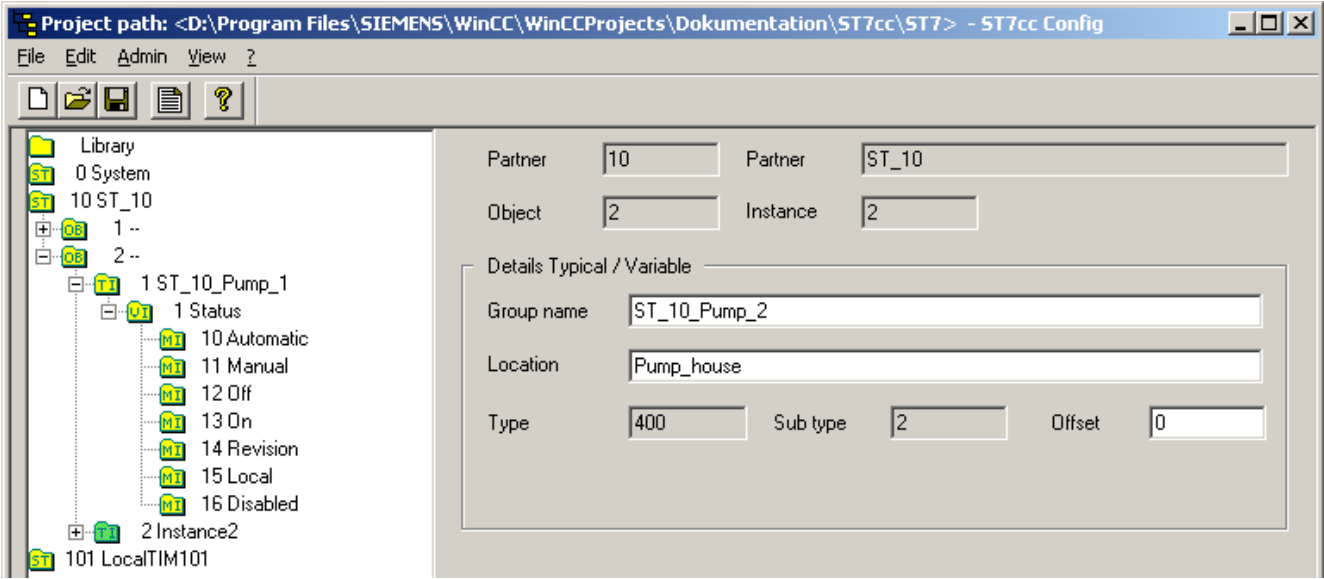




Figure 4-50 Structure information for a typical instance

Within the instantiated typical, the variables are identified by the  icon. The message processing functions of the variables are indicated by the  icon.

Instantiating by copying

When you create a typical, you can only assign one message processing function to a variable. Within a decoding, you can, however, assign further processing functions to a typical variable. To do this, right-click on the relevant variable and select the required option.

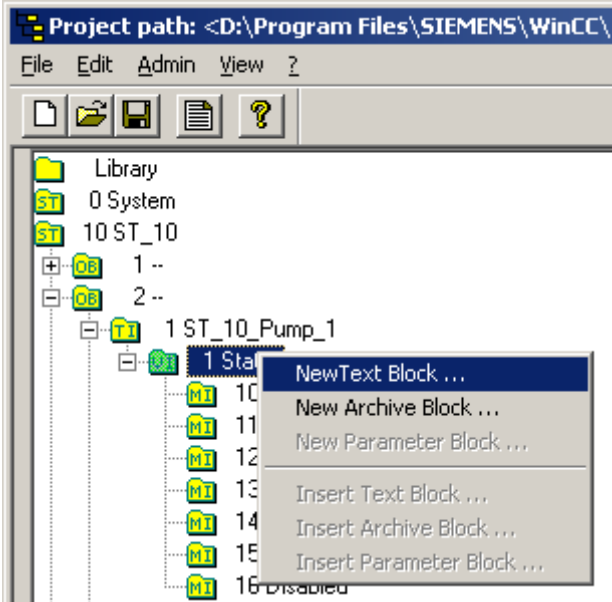


Figure 4-51 Creating a processing function in a typical instance

4.4 Configuring

After you have expanded instances of typicals by adding processing functions, you can copy them and paste them into an object decoding.

To copy a typical instance, follow the steps below:

1. Right-click on the typical instance you want to copy and open the context menu (see figure).
2. Select Copy Typical Instance Tree (see figure).

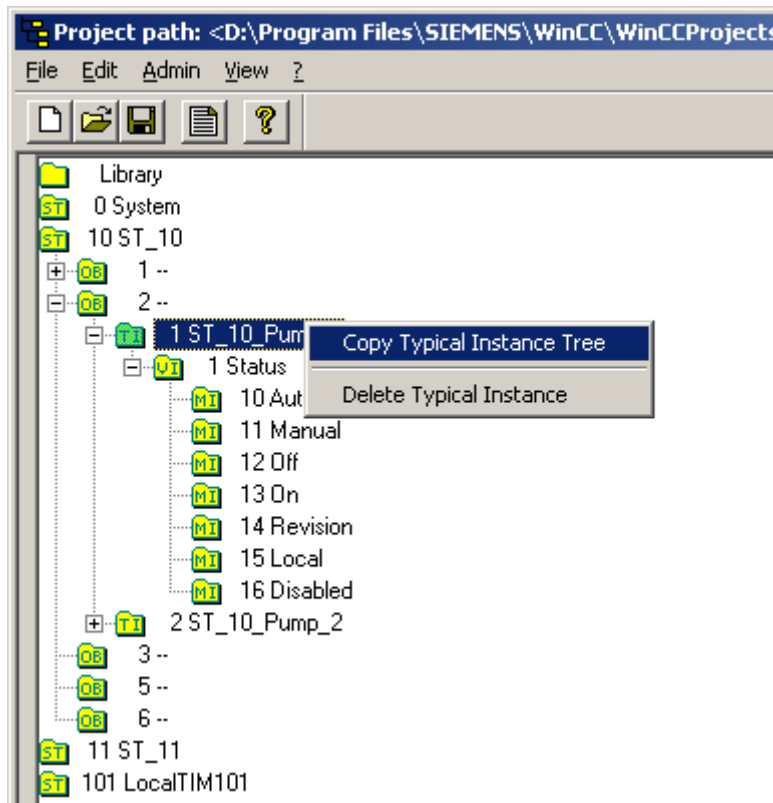


Figure 4-52 Copying a typical instance

To insert a copied typical instance into a decoding, follow the steps below:

- 3. Right-click on the required decoding, open the context menu and select the option Insert Typical Instance tree (see figure).

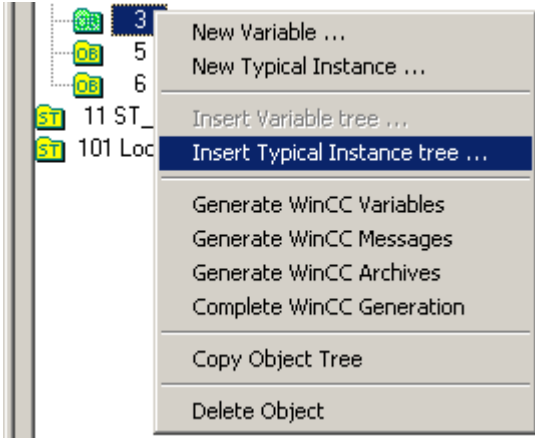


Figure 4-53 Inserting a typical instance

- 4. Then enter the instance number and click *OK*.

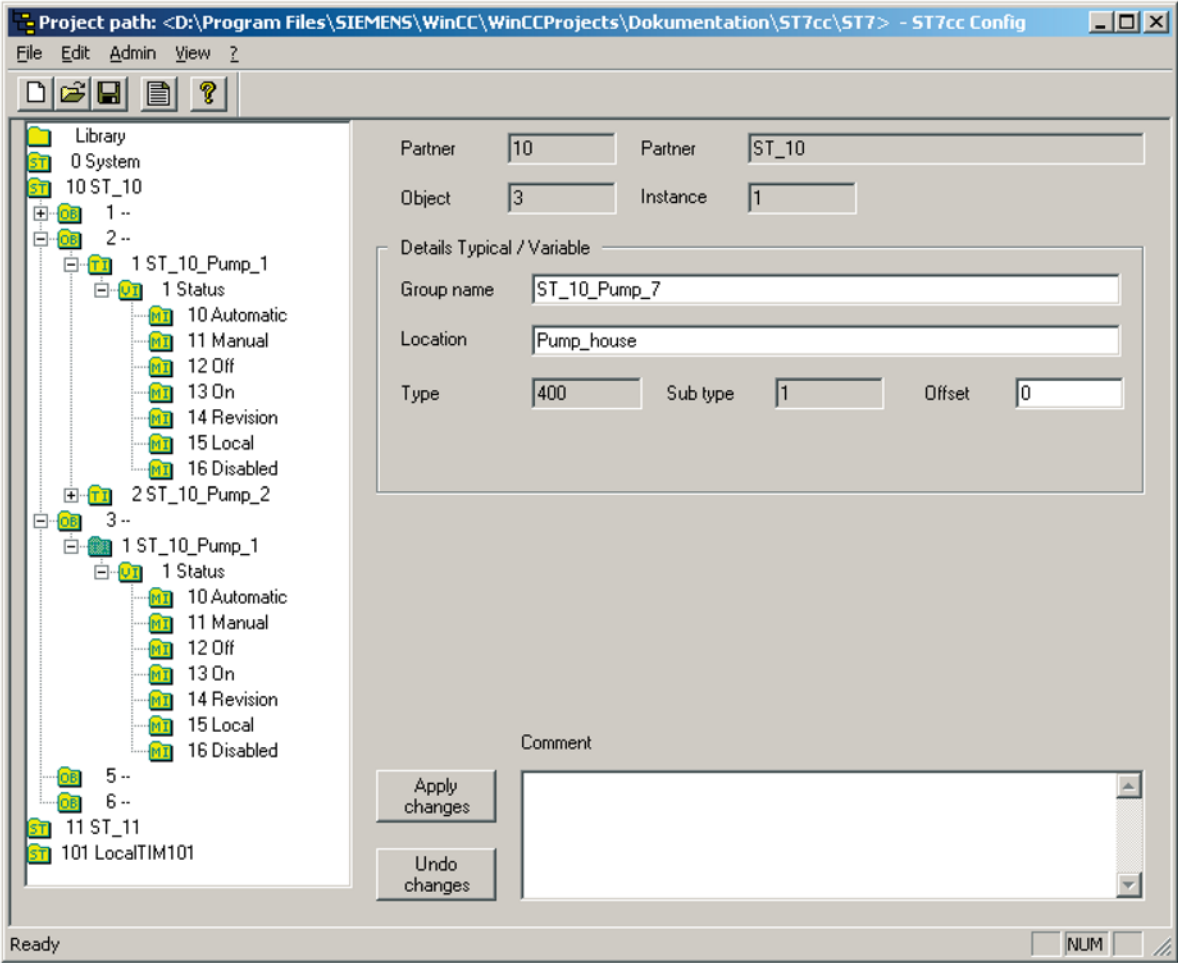


Figure 4-54 Inserting a typical instance

4.4 Configuring

After confirming with OK, the new typical instance exists in the object tree (see figure).

Note

When you copy, all information is copied 1:1. After copying, the user must change the group name and possibly the offset information to obtain a unique name or the correct position within the data area.

4.4.9 Copying and deleting decodings

Overview

With the functions for copying and pasting parts of object trees (see figures below), you can copy a decoding or object template and paste it as a new decoding for a SINAUT object.

With the library, it is not possible to create new object templates by copying object templates. These must always be created new.

If you want the copy to decode a SINAUT object, when you paste it, specify the object number of the SINAUT object to be decoded.

Copying a decoding

To copy a decoding, follow the steps outlined below:

1. Right-click on the object you want to copy (decoding, object template) and open the context menu (see figure).
2. Select the Copy Object Tree option.

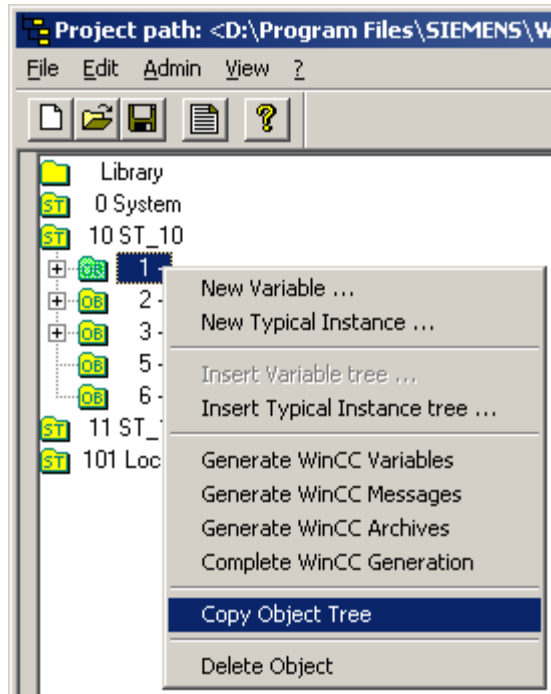


Figure 4-55 ST7cc Config dialog

3. Then right-click on the subscriber below which you want to paste the copy and open the context menu (see figure).

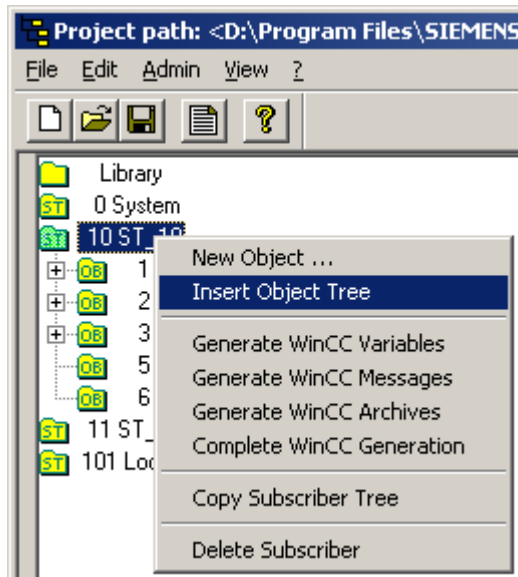


Figure 4-56 ST7cc Config dialog

4. Select the *Insert Object Tree*. The *Add* dialog opens.
5. In the *Add* dialog, enter the number of the SINAUT object that will be decoded by the copy and confirm with *OK*.

Note

The SINAUT object number is identical to the number of the instance DB that was configured for the object to be decoded in the CPU program of the station. You will also find further information in the TD7 block list (see section SINAUT TD7 block structure in ST7cc Config (Page 259)).

As a result, the inserted object is displayed in the object tree, in this example, object 3 of subscriber 10 (see figure).

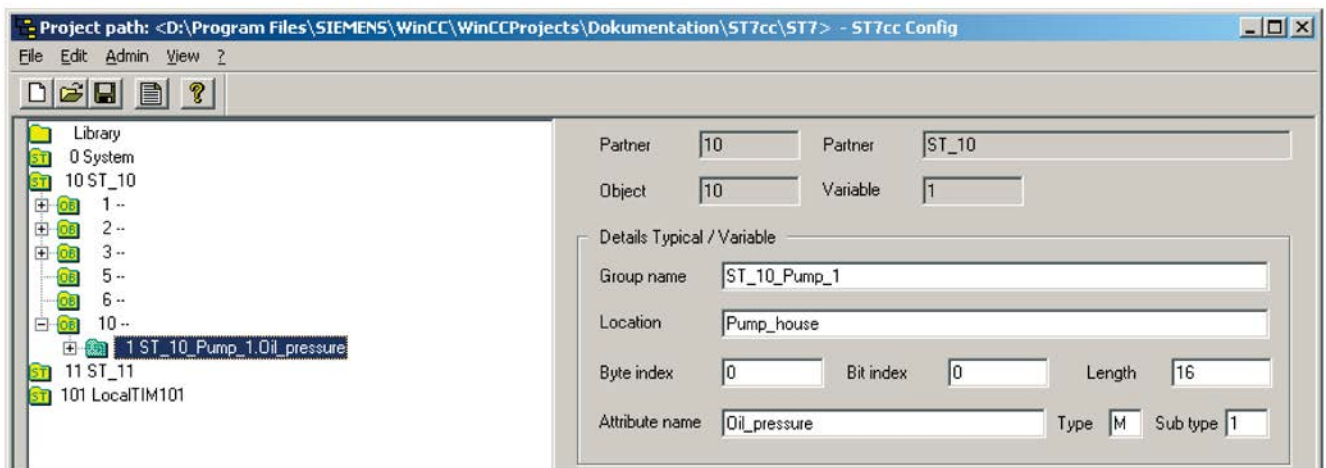


Figure 4-57 Section of the ST7cc Config dialog

Note

When copying, variable names are adopted 1:1 and must then be modified. If you forget to change the names, the variable names occurring two or more times are marked in the variable list. Compare section Variable list (Page 257). When you generate the WinCC tags, if there are ST7cc variables with the same name, only those first accessed by the generator are created.

Deleting decodings, object templates and typicals

To delete a decoding, an object template, or typicals, follow the steps outlined below:

1. Right-click on the object you want to delete (decoding, object template, typical) and open the context menu (see figure).
2. Select the *Delete Object* option.

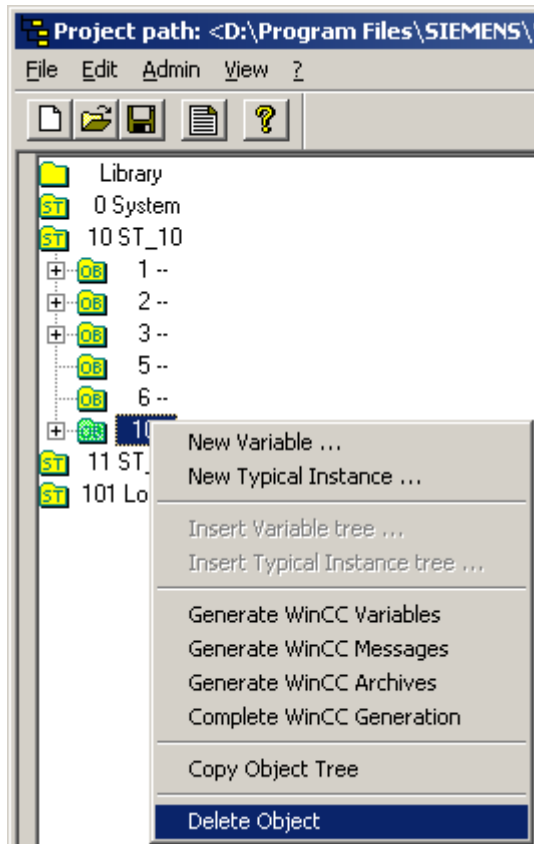


Figure 4-58 ST7cc Config dialog

4.4.10 Copying and deleting subscribers

When you copy a subscriber, you copy all the decodings of a subscriber to create them under a new subscriber number. When you paste, you therefore set up a new subscriber.

Follow the steps outlined below:

1. Select the subscriber to be copied with the right mouse button and open the context menu (see figure).
2. Select the Copy Subscriber Tree option.

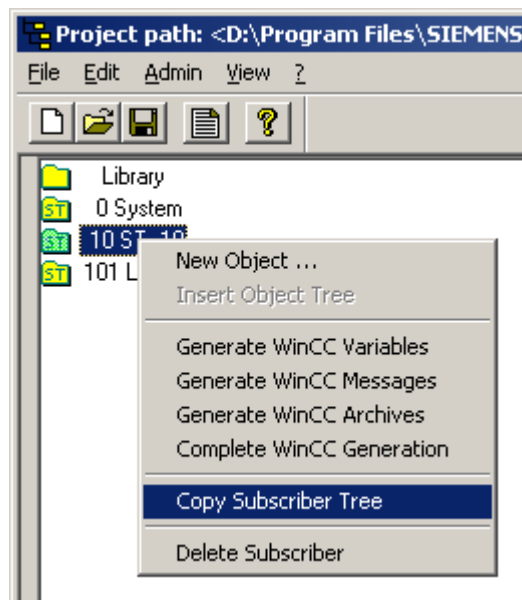


Figure 4-59 ST7cc Config dialog

3. Click on Edit in the menu bar to open the menu (see figure).
4. Select the *Insert Subscriber Tree* option. The *Add* dialog opens.

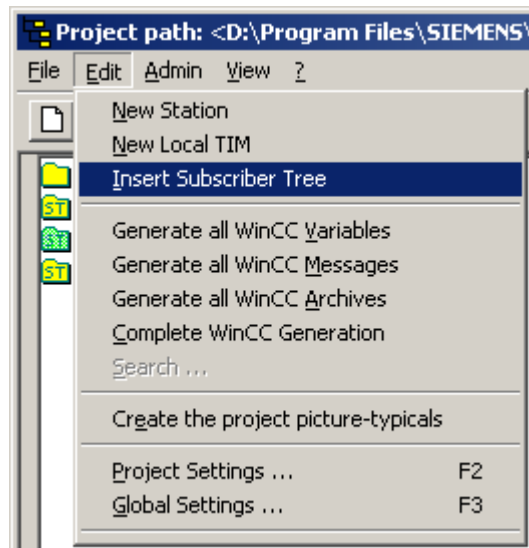


Figure 4-60 ST7cc Config dialog

5. Enter the subscriber number of the SINAUT subscriber you are setting up in the input field of the Add dialog and confirm your entry with OK. (see figure)

Note

The SINAUT subscriber numbers of the TIMs and the stations (CPU) can be found in the corresponding STEP 7 project. You will also find further information in the TD7 block list (see section SINAUT TD7 block structure in ST7cc Config (Page 259)).

4.4 Configuring

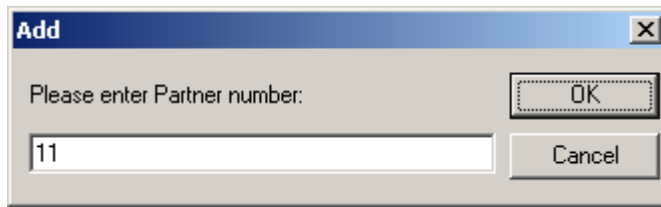


Figure 4-61 Dialog for entering the subscriber number

After entering the subscriber successfully, the new subscriber is displayed in the object tree.

Note

When copying, the subscriber name and the variable names are adopted 1:1 and must be updated. If you forget to change the names, the variable names occurring two or more times are marked in the variable list. Compare section Variable list (Page 257). If you do not modify the subscriber name this means that the system variable names of the copied subscriber also exist twice and are marked accordingly.

4.4.11 Update scenarios for system typicals

Update scenarios for ST7cc system typical

Table 4- 6 Upgrade stages for the use of picture typicals and faceplates

ST7cc version	Characteristics of the system typicals	TIM modules and version	Use / continued use of picture typicals and faceplates of the ST7cc version	Note
V1.x to V2.0.4	Simple range of information No distinction between local TIM and station (CPU)	TIM 3x, 4x with firmware version V3.x	V1.x	
V2.4.x	Expansion of the range of information Distinction between local TIM and station (CPU)	TIM 3x, 4x as of firmware version V4.x	V1.x / V2.0.4 V2.4.x V1.x / V2.0.4 → V2.4.x	A B C
V2.5	As V2.x Expanded of the range of information for TIM 3V-IE	TIM 3x, 4x as of firmware version V4.x and TIM 3V-IE	V1.x / V2.0.4 V2.4.x V2.5 V1.x / V2.04 → V2.5 V2.4.x → V2.5	A1 D I C F

Case A:

You already have a WinCC project with numerous pictures and picture typicals for the ST7cc subscribers and want to use ST7cc version V2.4 or higher. If you can do without the extended range of information, you should continue to use system typicals of the ST7cc version V1.x to V2.0.4 to avoid the extra effort involved.

To do this, deselect the *New Faceplates (from ST7cc V2.4)* check box in the *Server* tab of the *ST7cc Project Settings* dialog. The new ST7cc version as of V2.4.x then works with the old system typicals.

Case A1:

Case A still applies if no TIM 3V-IE is connected to ST7cc via Ethernet. Otherwise Case C applies.

Case B:

You start a new WinCC project with ST7cc version V2.4.x. You then also need to use the picture typicals and faceplates of ST7cc version V2.4.x. To do this, select the *New Faceplates (from ST7cc V2.4)* check box in the *Server* tab of the *ST7cc Project Settings* dialog.

Case C:

You already have a WinCC project with pictures and picture typicals for the ST7cc subscribers and want to use ST7cc version V2.4.x / V2.5 (system typicals, picture typicals, and faceplates).

In this case, the following actions are necessary:

1. Delete the ST7cc system variables in WinCC.
2. Enable the *New Faceplates (from ST7cc V2.4)* option in the *Server* tab of the *ST7cc Project Settings* dialog.
3. Replace the old system typicals in our existing library with the system typicals of ST7cc version V2.4.x or V2.5.
4. Update the *Sub type* parameter for every subscriber in your ST7cc object tree using ST7cc Config to indicate whether this is a local TIM or a station subscriber (CPU).
5. Replace all old picture typicals in your process pictures with the new picture typicals of ST7cc version V2.4.x or V2.5 since the old picture typicals will no longer be correctly supplied.
6. Re-generate your WinCC tag management so that the WinCC tags are created for the new system variables.

Note

If you want to display expanded information for stations and local TIMs, all the local TIMs must be upgraded to firmware V4.x. Otherwise, only a subset of the information will be displayed.

Case D:

If you upgrade your existing project with system typicals of ST7cc version V2.4 to ST7cc version V2.5 without wanting to connect a TIM 3V-IE as local TIM over Ethernet, you do not need to do anything. ST7cc V2.5 continues to supply the picture typicals of version V2.4.x with values correctly.

Case E:

If you start a new WinCC project with ST7cc version V2.5, you will normally also use the supplied system typical, picture typical, and faceplate of ST7cc version V2.5. To do this, select the *New Faceplates (from ST7cc V2.4)* check box in the *Server* tab of the *ST7cc Project Settings* dialog.

Case F:

You already have an existing project with ST7cc version V2.4.x and want to connect a TIM 3V-IE as local TIM over Ethernet to your SINAUT system. To obtain the new status information from a TIM 3V-IE, you need to use the system typical and faceplate of ST7cc version V2.5.

In this case, the following actions are necessary:

1. Replace the system typical of version V2.4.x in your existing library with the system typical of ST7cc version V2.5 and use the new picture typical and faceplate for the new SINAUT subscribers you introduce into your project. You only need to replace the picture typical existing in the process picture with the new picture typical if you replace a TIM 3x with a TIM 3V-IE.

4.5 Configuring processing functions

Overview

As already described in section Processing options for ST7cc variables (Page 168), one or more processing functions can be assigned to a variable (see figure).

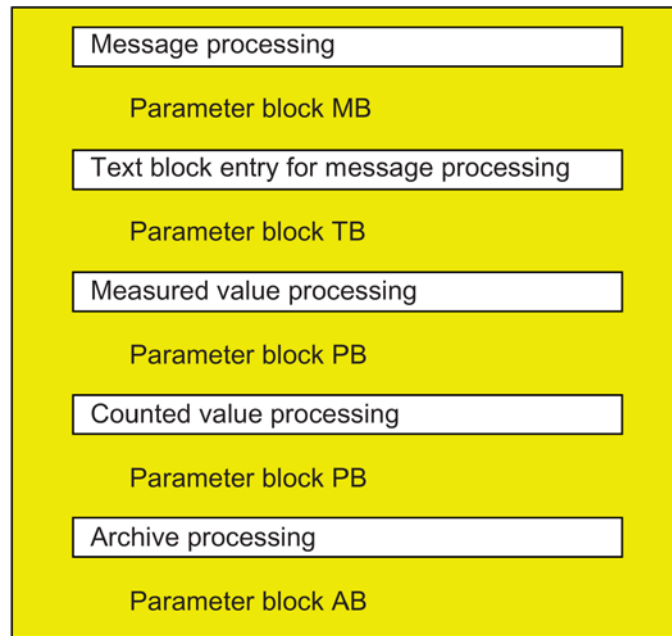


Figure 4-62 Processing functions with their parameter blocks

The processing functions fulfill two tasks:

1. They execute functions that logically expand the ST 7 object processing of the automation level in the management level and that are expected in SINAUT applications. Examples include counted value and measured value processing.
2. They execute functions that cannot be executed by WinCC due to the staggered supply of the data, for example chronologically correct archiving of data that arrives staggered over time.

It is both efficient and convenient to configure WinCC processing at the point where the variable is defined and then transfer the required parameter information to the target processing function. Passing parameter information to WinCC is implemented by the ST7cc generating functions for WinCC, see section Generating for WinCC (Page 263).

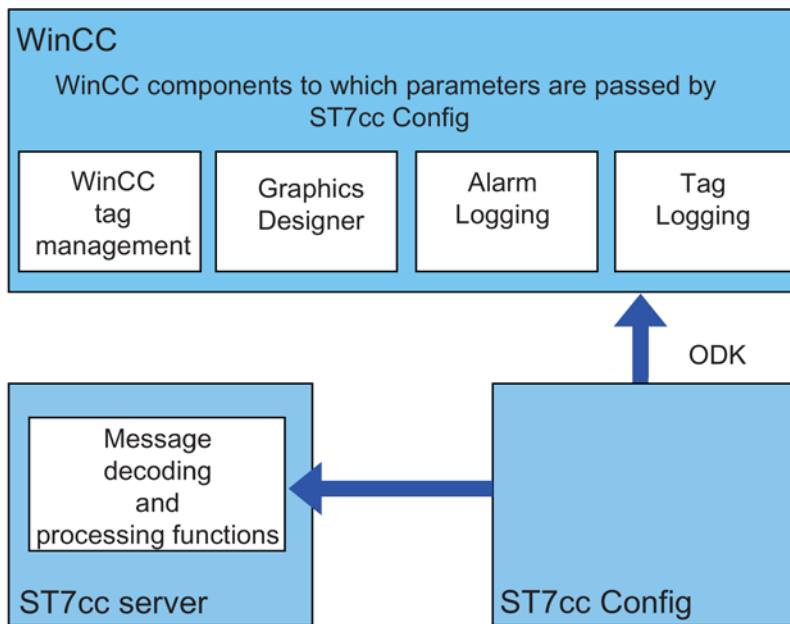


Figure 4-63 Transferring parameters to the processing components

The figure shows the system components to which ST7cc Config transfers generated parameter data. With the WinCC generation, the parameters are transferred to the processing functions that will be executed by WinCC.

Permitted processing functions dependent on the variable type

Type	Sub type	Permitted lengths	Permitted processing functions
M	1	16, 32 bits	MB, TB, AB, PB (measured value processing)
	2	16, 32 bits	MB, TB, AB, PB (measured value processing)
	3	16 bits	MB, TB, AB, PB (measured value processing)
	4	32 bits	MB, TB, AB, PB (measured value processing)
S	1	1 to 32 bits	MB, TB, AB
C	1	32 bits	MB, TB, AB, PB (counted value processing)
	2	32 bits	MB, TB, AB, PB (counted value processing)
	3	32 bits	MB, TB, AB, PB (counted value processing)
	4	32 bits	MB, TB, AB, PB (counted value processing)
A	1	16 bits	MB, TB, AB, PB (format conversion)
		32 bits	MB, TB, AB, PB (format conversion)
	2	16 bits	MB, TB, AB, PB (format conversion)
	3	-	not used
	4	32 bits	MB, TB, AB, PB (format conversion)
D	1	1, or 8 bits	MB, TB, AB
D	2	8 bits	MB, TB, AB

Note

A message processing functions (MB parameter block) can be assigned to the variables in object templates and typical in the library and in decodings.

The parameter blocks TB, AB, and PB can only be assigned to variables in decodings.

Several message processing functions and text block entries can be assigned to a variable, however only one measured value or counted value processing function.

4.5.1 Working with a processing function

Creating a processing function

To create a processing function for a variable, follow the steps outlined below:

- Select the required variable with the right mouse button and open the context menu (see figure). To create processing functions, the following options are available:

New Message: for message processing (basic function)

New Text Block: for entry of additional static texts for message processing.

New Archive Block: for archive processing

New Parameter Block: for measured value or counted value processing. Depending on the type of variable selected, a measured value or counted value processing function is created.

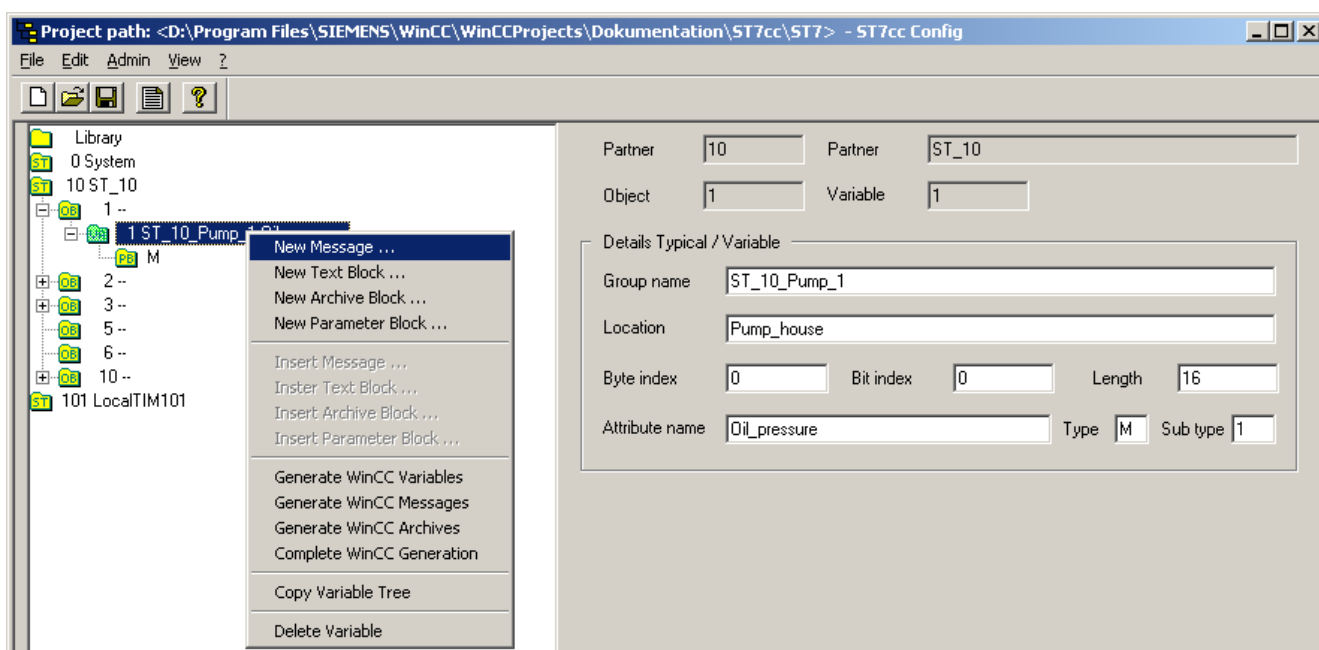


Figure 4-64 Creating a processing function in the ST7cc Config window

Copying and pasting a processing function

1. Select the required processing function with the right mouse button and open the context menu (see figure).
2. To copy, select the option *Message / Text Block / Archive Block / Copy Parameter Block*. The suitable option is available for each processing function.

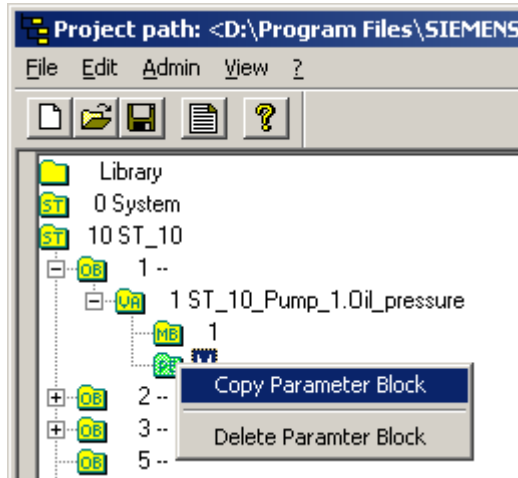


Figure 4-65 Copying a processing function

3. Select the required variable with the right mouse button and open the context menu (see figure).

4. To paste, select, for example, the option Insert Parameter Block if you are pasting a measured value processing function.

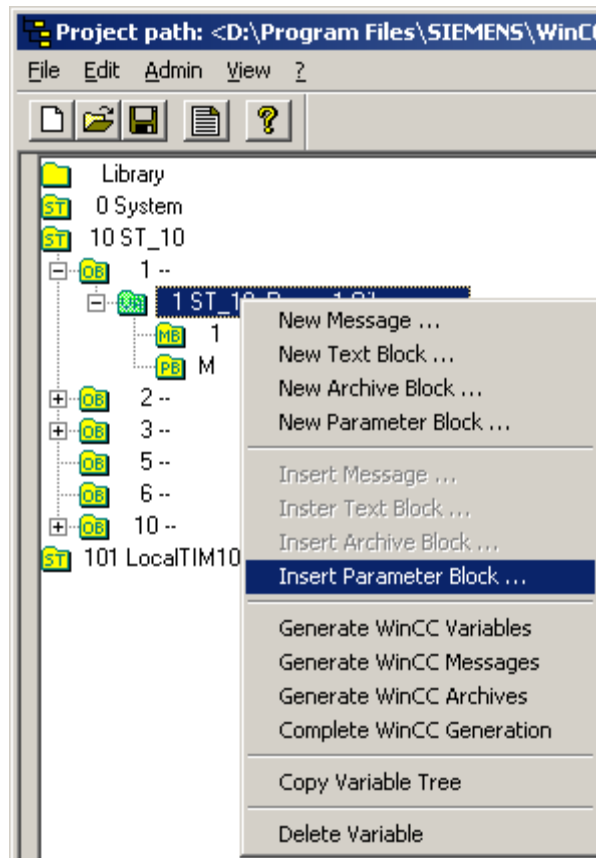


Figure 4-66 Inserting a processing function

Deleting a processing function

1. Select the required processing function with the right mouse button and open the context menu (see figure).
2. To delete the parameter block, select the Delete Parameter Block option.

4.5.2 Message processing

Message processing

The message processing of the variables involves interaction between the ST7cc server and ST7cc Config and the WinCC message system.

The WinCC message system processes events of functions that monitor activities in the process, at the automation level, and in the WinCC system. It displays acquired events both optically and acoustically and archives them electronically and on paper.

The task of Alarm Logging Runtime is to acquire the messages and to handle the acknowledgments.

By using SINAUT, two features emerge compared with pure WinCC applications:

- Messages arrive from the automation level staggered over time. The time stamp of the message is event-related, the transfer of the message to the management level can be delayed by hours in dial-up networks subject to call charges depending on the connection configuration.
- In various applications, particularly in older systems, process states are transferred as multi-state signals. This means that a bit area / value range must be evaluated. A two-state signal (range of values of 2 bits) with the set of values 0 to 3 can therefore represent four process states whose occurrence must be signaled and archived.

Due to the special features of SINAUT and the WinCC functionality, this results in the situation as shown in the table below.

Table 4- 7 WinCC functionality and SINAUT requirements

WinCC version	Arrival of events over SINAUT staggered over time	Evaluation of value ranges / multi-state signals.
Less than V5.1	Processing in WinCC not possible, Concrete time relationship is lost	Processing in WinCC not possible
As of V5.1	Processing in WinCC possible	Processing in WinCC not possible

The functionality of ST7cc message processing is designed for this situation. If the user has a WinCC version \geq V5.1 and no multi-state signals, message processing can be handled completely in WinCC.

With both methods of message generation, the parameters for message processing are set with ST7cc Config regardless of the executing component. When the message management is generated, the parameters are transferred to WinCC (Alarm Logging).

The figures show the differences between message generation in ST7cc and in WinCC.

Note

Please note the advantages and disadvantages of message generation in ST7cc / WinCC as explained in the following sections.

Control of message processing

Using the configuration parameter Create messages by WinCC, the configuration engineer can decide whether the ST7cc server generates the message jobs (change of state information causing the message) or the WinCC data manager. For more detailed information, refer to section Project settings: Server (Page 118).

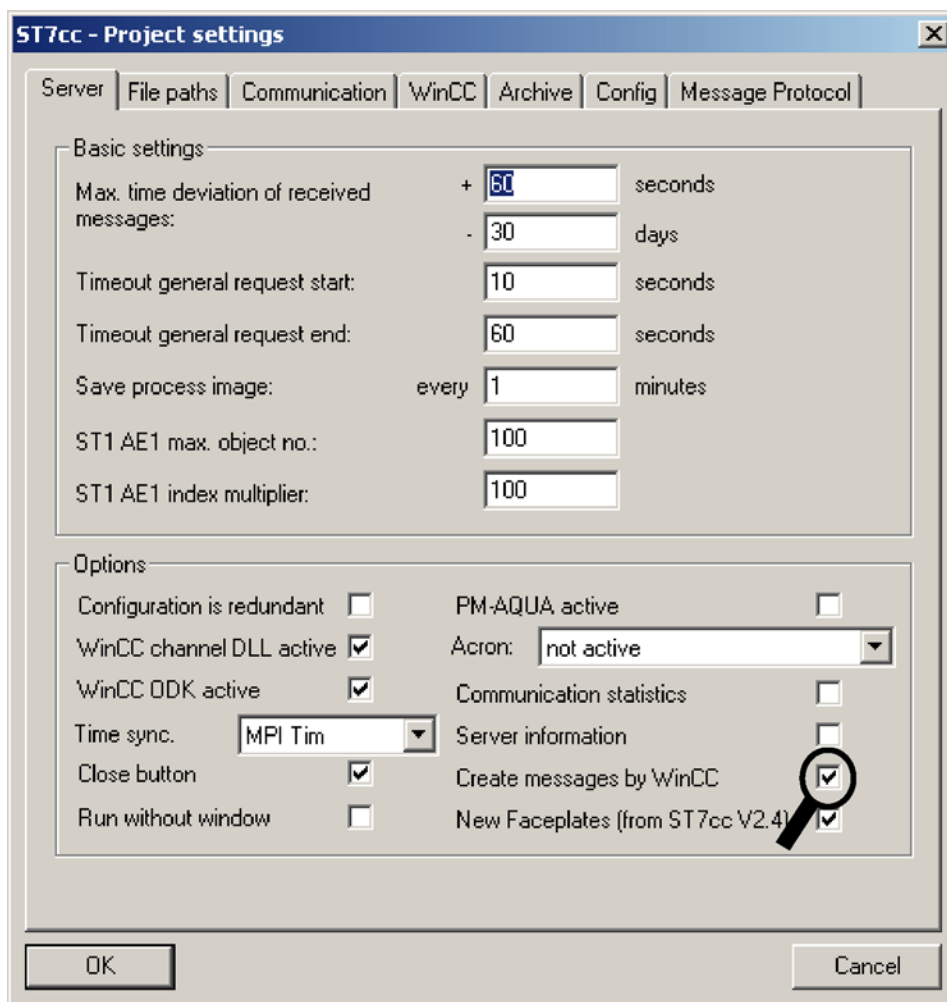


Figure 4-67 ST7cc Project Settings dialog

Generating messages in ST7cc

The ST7cc server checks, among other things, whether a message should be generated for the incoming process information. If this is the case, the ST7cc server transfers a message job to Alarm Logging. Alarm Logging puts together the actual individual message and is responsible for displaying and archiving the message. With this strategy, up to WinCC version 5.0, it is guaranteed that the event-related time stamp supplied by ST7 was included in the WinCC messages. This strategy is also possible in WinCC versions > V5.0.

Since, in this application, the message jobs are generated by the ST7cc server, value ranges can be evaluated and a message trigger generated depending on selectable change of state information.

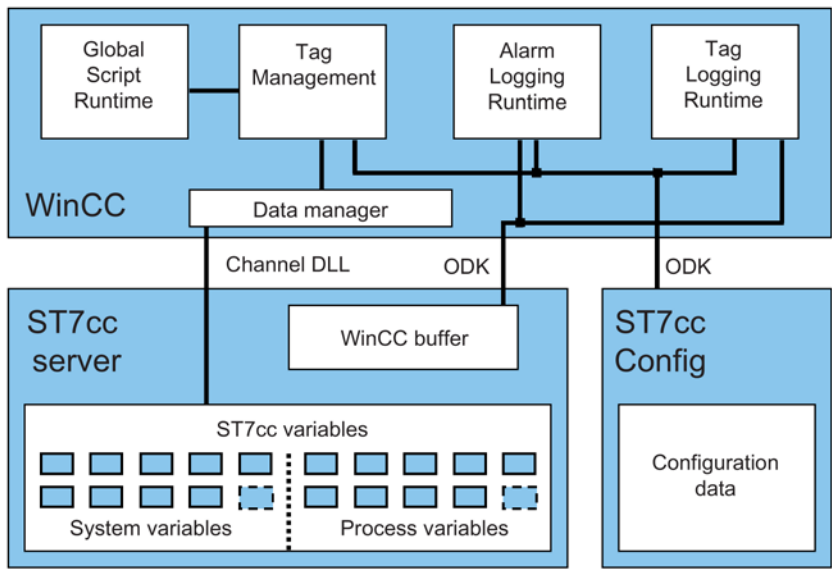


Figure 4-68 Data paths with messages generated by the ST7cc server

Note

Disadvantage

The disadvantage of generating messages in ST7cc is that when using a redundant WinCC system (WinCC Redundancy), the message must be acknowledged by the operator on each of the redundant partners. The consistency of the acknowledgments cannot be supported by the system.

Generating messages in WinCC

As of WinCC version V5.1, the WinCC data manager can generate the message job and accept and process the event-related time stamp supplied by ST7cc. This means that the messages are generated entirely in WinCC.

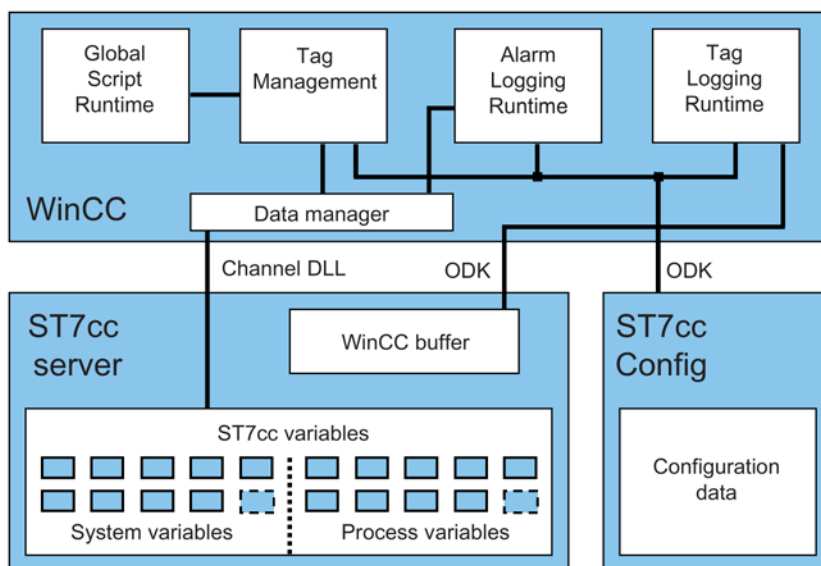


Figure 4-69 Data paths with messages generated by WinCC

Note

Disadvantage

Messages are generated in WinCC bit-oriented. It cannot analyze value ranges or generate messages dependent on trigger values.

Note

Advantage

The advantage of generating messages in WinCC is that when using a redundant WinCC system, an event can be acknowledged with system support on both redundancy partners (acknowledgment consistency). The user has all WinCC acknowledgment options available.

Message number

In WinCC, the configured messages are identified uniquely by their numbers. This makes the message number a central factor in the WinCC message system.

As described in section Project settings: Config (Page 142), message numbers can be generated in two ways in ST7cc:

- Old method (structure-oriented)
- New method (offset-oriented)

Message number according to the old method

Note

Since WinCC reserves various number ranges for message numbers and ST7cc relieves the user of the need to specify a message number with an algorithm, configured numbers are available in a range between 2 million and approximately 500 million for the message numbers configured by ST7cc.

In WinCC, a message is put together from:

1. Subscriber number of the communication subscriber
2. Number of the communication object
3. Number of the subtypical instance (or 0 for variables not associated with an instance, i.e. when decoding using object templates)
4. ST7 message number (consecutive numbering of the messages of a variable)

and forms a WinCC message number that is unique throughout the project. The precise composition of this number can be set globally using the message format.

For possible settings, refer to section Project settings: Config (Page 142).

Message number according to the new method

If the message numbers are created using the new (offset-oriented) method, consecutive numbers are generated automatically starting at the offset. Apart from setting the offset, you cannot influence the generation of the numbers.

Parameter assignment for message processing

How to create a processing function is described in Working with a processing function (Page 233). The figure shows the ST7cc Config window with the input boxes for assigning parameters to the processing function (window area: Message block details). In the example selected, a message is to be generated when there is a change to the individual states. The figure shows the message blocks of a variable of a typical. How to create message blocks within a user typical is described in section Creating a user typical (Page 199).

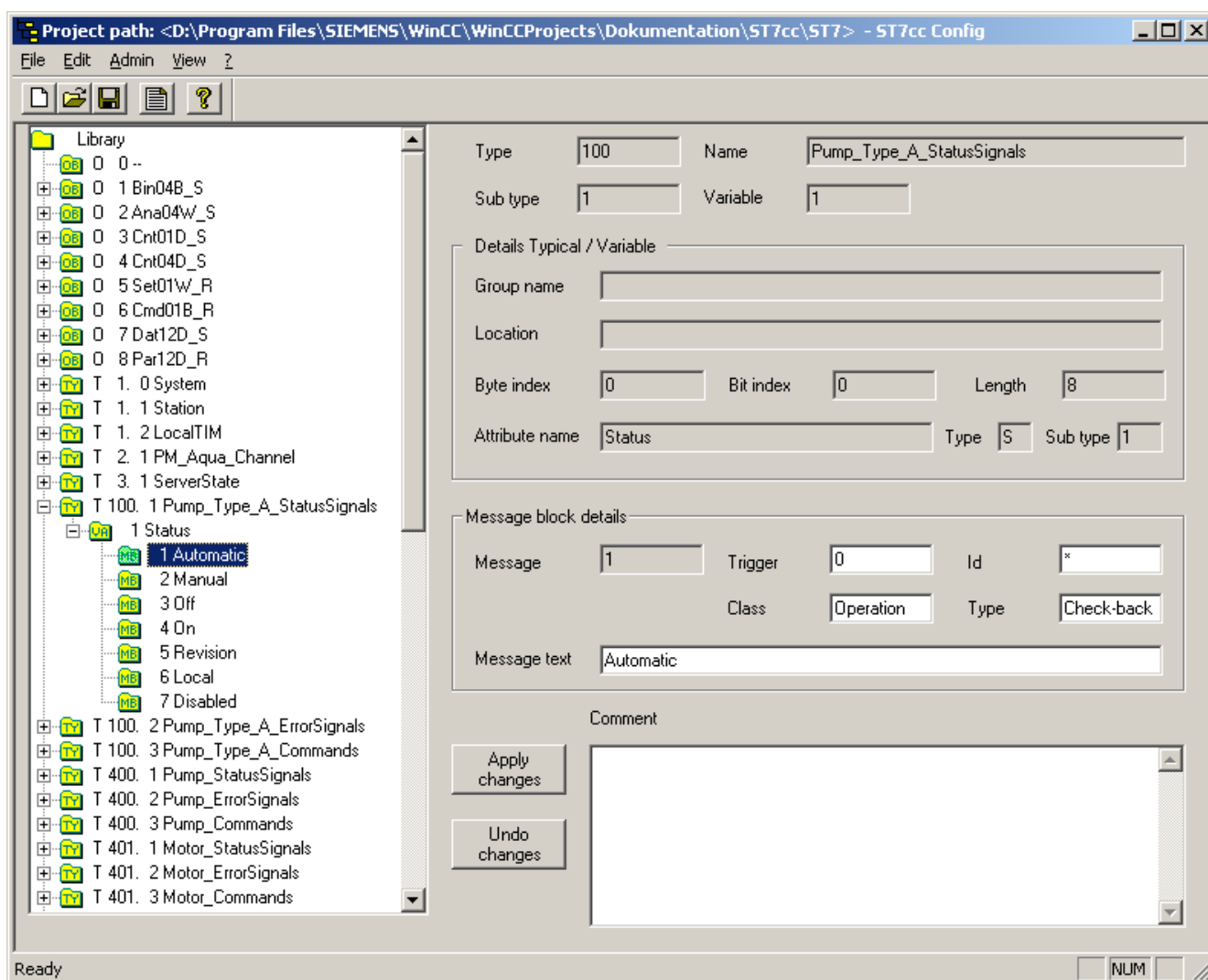


Figure 4-70 Message blocks of an ST7cc variable

Seven message blocks with numbers 1 through 7 were created for the seven states of a variable. The message number is the number of the message block of a variable. It is assigned when the message block is created. It is included in the automatic generation of the WinCC message number.

If you created the message numbers using the new (offset-oriented) method, the message number displayed in the Message number output box is a relative message number. The relative message number is the message number generated by ST7cc minus the offset, see section Project settings: Config (Page 142).

Note

Please note that you can assign a total of maximum 99 message processing functions for all variables of an object decoding.

Message class

Each message is assigned to a WinCC message class. Using message classes, the following is specified in WinCC for all message types of this message class:

1. The acknowledgment philosophy
2. The texts for entered state, left state, acknowledged and entered/left state
3. The output of optical / acoustic signals

The message class entered in ST7cc Config must already have been defined in WinCC.

Message type

In WinCC, message types are subgroups of the message class and can adopt different colors according to the message status.

The message type entered in ST7cc Config must already have been defined in WinCC.

Message ID

The message ID specifies whether a message (more correctly: change of state information causing the ST7cc message) is sent only when a state is entered or left (adopting a value from the set of values or leaving the set of values) or whether it is also sent when the state is signaled again and the value is within the set of values.

Identifier	Meaning
*	The message is sent as entering state when the signal adopts one of the values of the set; it is sent as leaving state when it no longer has a value from the set of values.
!	The message is then sent again as entering state when the next signaled value remains within the set of values.

Set of values / trigger value

Generating messages in WinCC:

Messages are generated in WinCC bit-oriented. The parameter specifies the bit number within the WinCC data area (bit field) that causes a message to be generated when it changes.

Generating messages in ST7cc:

Message generation in ST7cc is value range-oriented. A message is sent as 'entering state' when the signaled value adopts a certain state characterized by a specified set of possible values. It is sent as "leaving state" when the signal leaves the state; in other words, it adopts a value outside this set of values.

The set of values can be defined as follows:

- **Single value** (single number)
- **Value range** (number1 – number2)
- **Listing** of single values and value ranges (separated by comma)

Example of sets of values of a variable of the type *M*.

The set of values 80-100 could, for example, describe the temperature range of a medium that causes a message to be sent when the medium comes into this temperature range. A second set of values (second MB parameter block) 90-100 could be used to define a narrower monitoring range that stimulates a further message from ST7cc when the temperature enters this range.

This method could, for example, be used to monitor the temperature reaching a first upper limit with the value range 80-100 and a second upper limit with the value range 90-100.

Message text

The actual text of the message. In addition to this text, the context information resulting from the assignment of the message to a variable (device name, group name and variable name) is also made available in WinCC. Further static additional texts can also be defined.

4.5.3 Static additional texts

User texts

During configuration of the message system in WinCC, user text blocks can be deleted or added from a predefined list to be able to display additional static texts in a message. User texts (static additional texts) include, for example the plant designation, point of error etc.

The maximum length of a user text block is 254 characters. They are displayed, however, in one line and are limited to the screen width. Longer text is truncated in the display and cannot be shown.

Note

The defaults of the user text blocks differ in WinCC and PCS 7. With the WinCC User Text Blocks or PCS 7 User Text Blocks buttons, you can select the defaults you want to use.

A message in WinCC can contain up to 10 user text blocks and an information text (text block 11). Of these, the following text blocks are used as follows:

WinCC / PCS7	Meaning
Block 1	Message text. The message text is entered during assignment of parameters for the message processing function (message block).
Block 2	Subscriber name. In many applications, the subscriber name indicates the location / installation location of the subscriber.
Block 3	Group name
Block 4	Attribute name
Block 5	Location: In principle, this is a free text entry to provide the operator with more information. If the content of text block 2 provides enough information on the location, repeating the location information is unnecessary.
Block 6	Free for user
Block 7	Free for user

WinCC / PCS7	Meaning
Block 8	Free for user
Block 9	Free for user
Block 10	Free for user
Block 11	Information text (used by WinCC)

While the message text of the message processing (message block) is event-related, for example, Pump_1 changes to the ON, OFF, AUTOMATIC state etc., the additional static text of the text block, for example the plant designation is purely variable-related and has no relationship with the concrete event.

Entering text blocks

How to create a processing function is described in Working with a processing function (Page 233). The figure shows the ST7cc Config window with the input boxes for assigning parameters to the processing function (window area: Text block details).

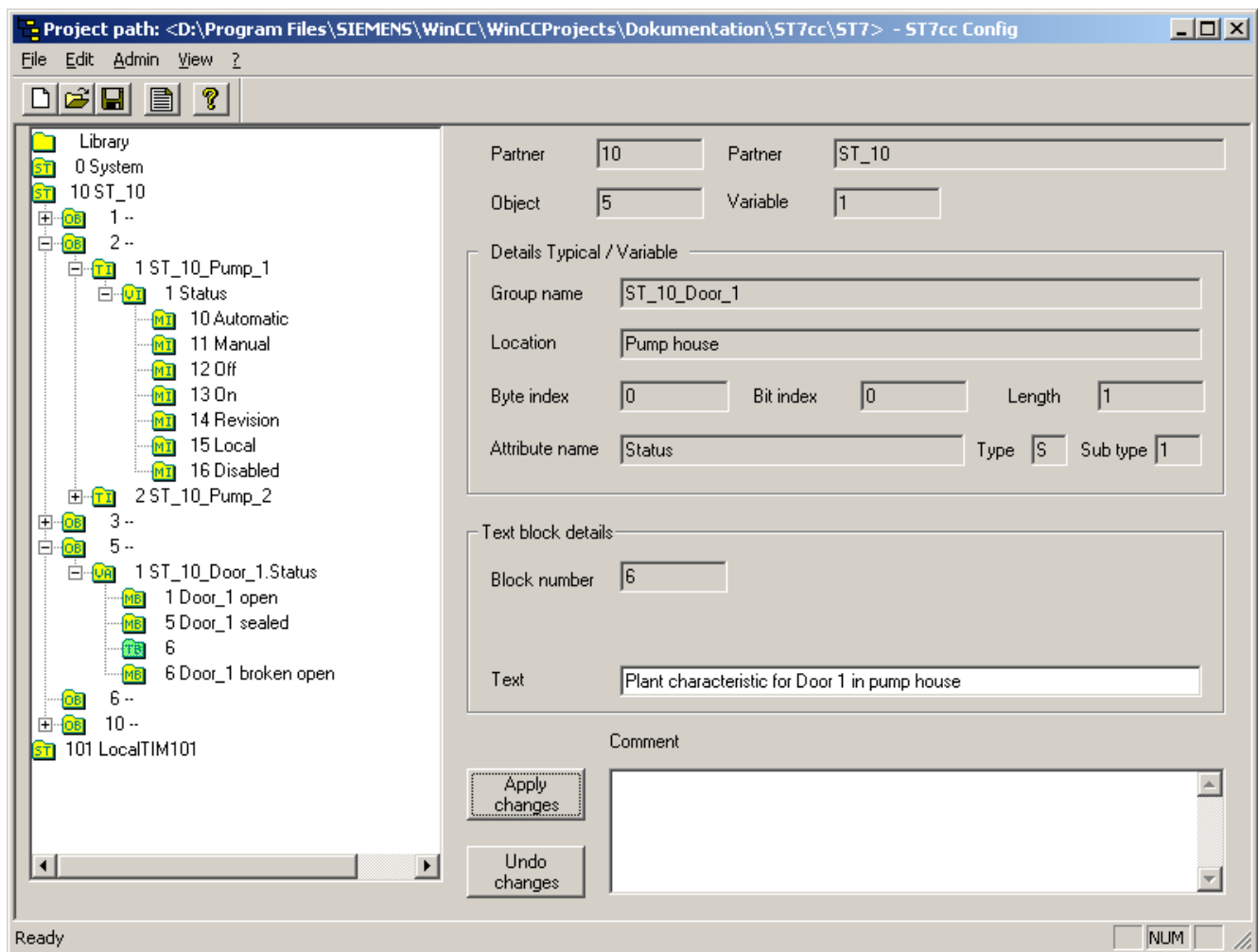


Figure 4-71 Inserting a processing function

Entering an additional static text can be considered as an expansion of the message processing function. Static text blocks with numbers 6 through 10 can be assigned to a variable.

The figure shows TB 6 after it was created in the object tree. The fact that text block 6 (TB 6) is beside message block 6 (MB 6) with the ST_10_Door_1.Status variable does not indicate a logical relationship between message processing (MB 6) and the additional static text. The additional static text of the text block (TB6) is output in all messages of the variable ST_10_Pump1.Status.

4.5.4 Counted value processing

In counted value processing, a distinction is made between basic processing and the calculation of the interval quantity as part of further processing.

Basic processing

A counted value is transferred as an absolute counter reading and multiplied by a factor to convert it into a physical value. If this value is displayed and/or saved as an absolute value, counted value processing is completed. Counted value processing is restricted to basic processing by setting the interval length or duration to 0 seconds.

Processing the end of the interval

The definition of intervals and corresponding interval-based calculation functions are an absolute requirement for calculating interval quantities (interval-related differential quantities). ST7cc interval end processing is designed for incoming ST7 messages containing event-oriented time-stamped process values received staggered over time. An interval quantity over a processing interval can only be completed when a process value is received with a time stamp that falls logically into the next interval. In the figure, the interval 44 is completed by the value received at 10:20.

Depending on the subtype of the variable, the interval end processing calculates

- the absolute value at the end of the interval. The calculation is made using the method of equal distribution (variable type Z, subtype 1 or 3).
- the differential quantity (interval quantity) between the calculated absolute values at the start and end of the interval (variable type Z, subtype 2 or 4).

Calculating the interval quantity

Interval quantity calculation means that the user defines a time interval, for example, 15 minute interval and wants the differential quantity calculated over this period. The differential quantity is obtained from the absolute counter reading at the end of the interval minus the absolute counter reading at the start of the interval. Since it would only be coincidence for the time stamp of a counted value to have a time value matching the end of the interval, the interval end processing calculates the value at the end of the interval based on equal distribution. In the sample calculations in the figure, it is assumed that the absolute counter starts at 0. The difference between the second received value (950) and the first received value (400) is distributed evenly over the intervals 44 and 45.

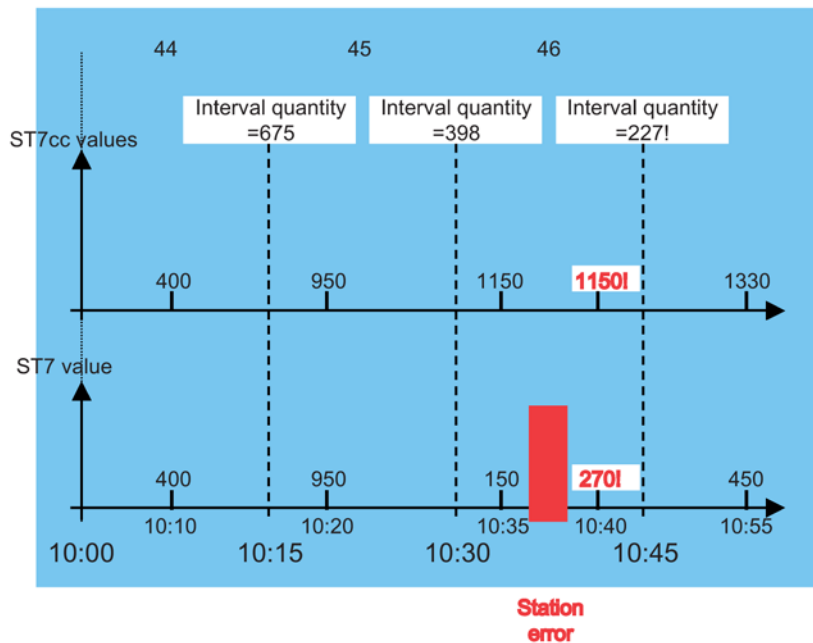


Figure 4-72 Interval-related counted value processing

Counter overflow

Counter overflow means that a counter at the automation level reaches its upper limit (highest counted value) and then starts at 0 again. When setting the parameters for counted value processing, you specify the upper limit (overflow).

In the example (see figure), the value 1000 was selected as the overflow value of the absolute counter. In interval 46, a value of 150 is received, (time stamp = 10:35). The ST7cc counted value processing function recognizes that there was an overflow and takes this into account when calculating the absolute value.

Disturbance in counted value acquisition

If counted value acquisition is interrupted by a problem on the CPU, the first counted value after the restart is transferred with a *first value identifier* that indicates to ST7cc that this is a new starting point for absolute value calculation. The value transferred with this identifier is not included in the absolute value calculation, in other words, the last absolute value with correct values is retained. Only the next SINAUT counted value is included in the continued absolute value calculation.

Parameter assignment for counted value processing

How to create a processing function is described in Working with a processing function (Page 233). The figure shows the ST7cc Config window with the input boxes for assigning parameters to the counted value processing function (window area: Parameter block details). Since the variable is of the counted value type, the parameter block for counted value processing is assigned to it automatically.

Note

One counted value processing function can be assigned to a variable. If the counted value transferred by SINAUT is required as an absolute and differential value for further processing, for example archiving, the transferred data subarea must be mapped to two variables.

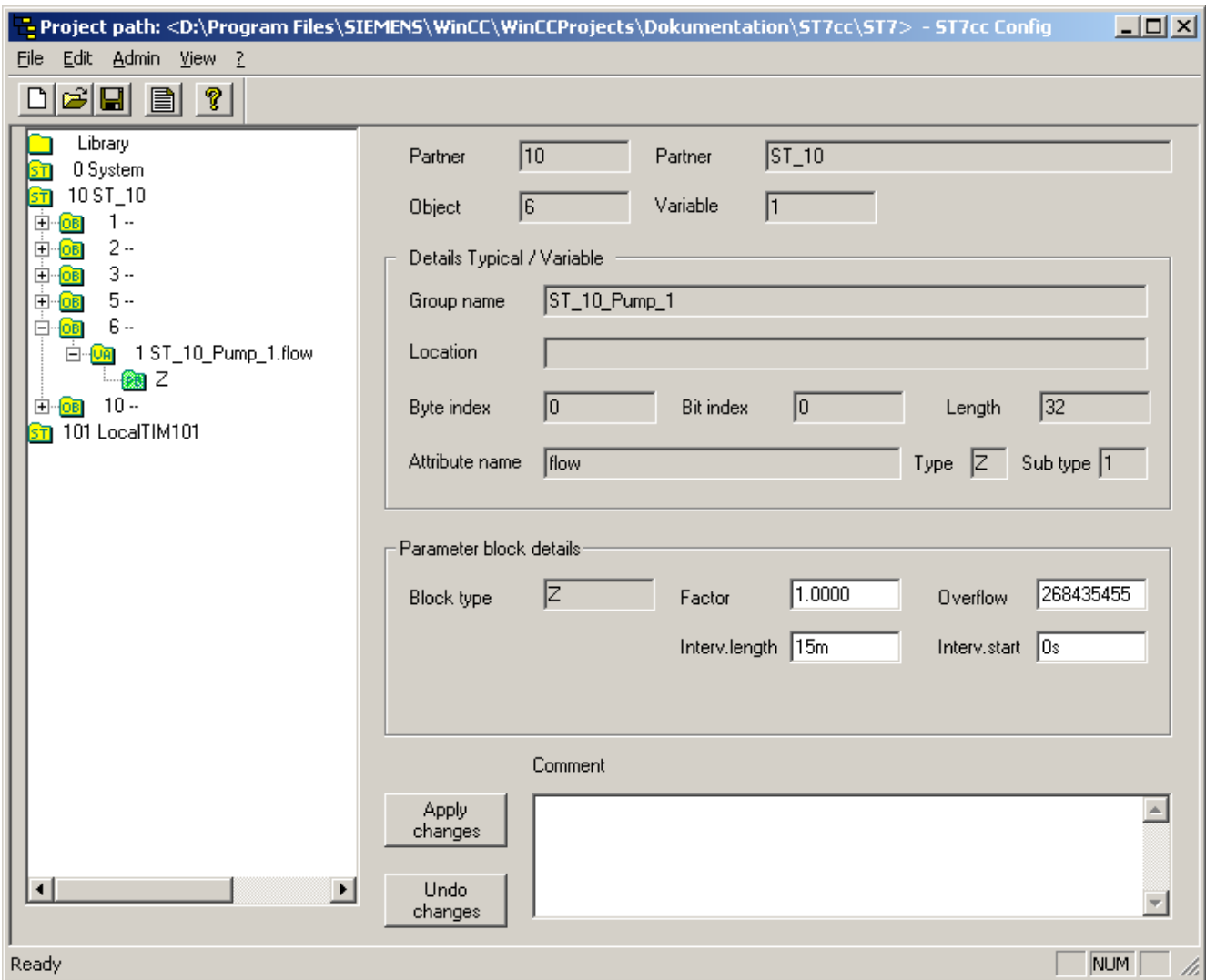


Figure 4-73 Inserting a processing function

Factor:

The absolute counter reading is converted to a physical value by multiplying it by the factor.

Overflow:

The overflow is the maximum counter reading that can be transferred to ST7cc from the automation level (programmable controller, calculation). When this value is exceeded, absolute value counting starts again at zero.

Time units for interval length and interval start:

The intervals for interval quantity calculation are defined by the interval length and a starting point within the interval. The time units used are:

*W*Week, *D*Day, *h* hour, *m* minute, *s* second.

Interval length:

The following values are permitted as the interval length:

1W, 1D, 1h, 30m, 20m, 15m, 12m, 10m, 6m, 5m, 4m, 3m, 2m, 1m, 0sec

Note

If you select a length of 0 seconds, the current value is passed on for archiving immediately without calculating an interval quantity

Interval start:

Lower / upper data point of the raw value for conversion to a physical value. The unit for the start time must be less than that for the interval length and must be within the interval length.

Example:

An interval 1T with a start time of 6h defines a daily interval starting at 6 a.m.

Note

The changeover from daylight saving time to stand time is handled as follows: The intervals 1W or 1D are increased or reduced by one hour. For all other intervals, there are additional interval entries or the entries are omitted.

4.5.5 Measured value processing

Overview

ST7cc measured value processing is designed to handle two tasks:

1. The transfer of measured values (raw values, physical values). In applications in which the automation level does not convert a raw value to a physical value, the conversion to the physical value is handled in ST7cc.
2. On one hand, the interaction of the event-oriented measured value acquisition and transfer staggered over time and on the other, the WinCC archive functions. The event-oriented and the time-staggered data supply require interval calculation and processing for the calculation of mean, minimum and maximum values that calculate the required values and supply them to the WinCC archives with the correct timing.

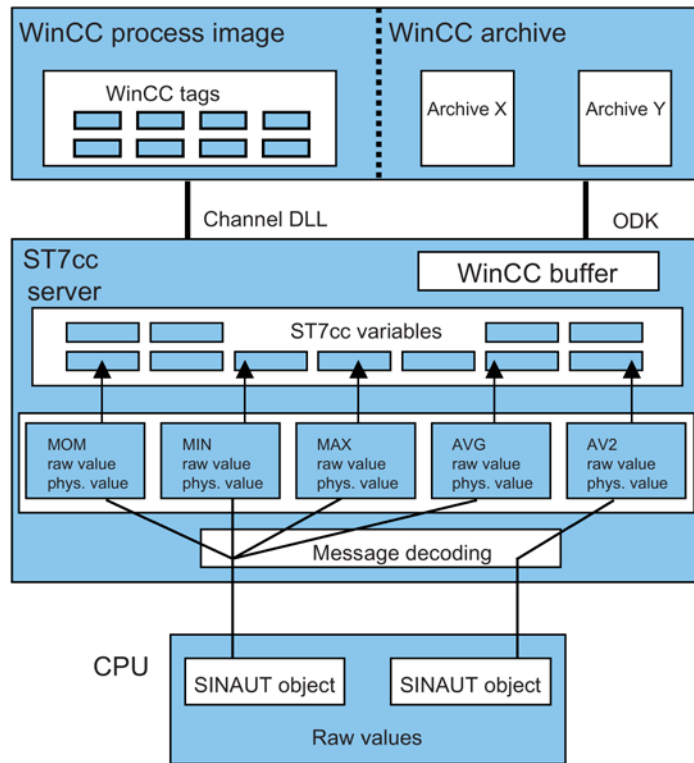


Figure 4-74 Measured value processing

Measured value processing consists of basic processing that converts a value that is supplied as a raw value into a physical value and further processing functions that calculate final values (results):

1. MIN: Interval-related calculation of a minimum (result calculation)
2. MAX: Interval-related calculation of a maximum (result calculation)
3. MOM: Instantaneous value acquisition at interval end (result calculation)

4. AVG: Calculation of a mean value based on incoming instantaneous values arriving at different times (result calculation)
5. AV2: Adoption of a mean value calculated automation level.

Basic processing / raw value scaling

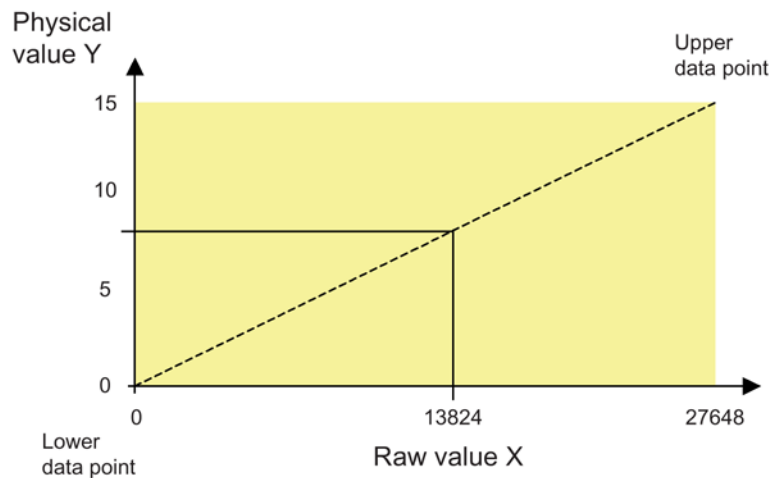


Figure 4-75 Linearization of a raw value into a physical value

Raw value scaling involves linear mapping of a range of values X to a range of values Y. In analog value acquisition, the mA range (for example 0 to 20 mA) of a process device is mapped to a raw value range (for example 0 to 27648) that in turn is converted to a physical value (for example 0 to 15 bar). The raw value ranges of the SIMATIC S5/S7 modules are described in the SIMATIC S7 manuals.

Note

Based on the variable type / subtype (for example M2), the S5/S7 raw value implicitly contains status information (for example 8000 hex or 7FFF hex). These are ignored in the conversion to the physical value by ST7cc. After calculating a raw value to obtain its physical value, the status information in WinCC must also be 'recalculated' filtered out / evaluated by the user.

When scaling in the message direction (measured value, returned setpoint), prior to display or archiving, the raw value integer is converted to a physical value in floating-point format bilinear interpolation between two reference points.

In the command direction (setpoint), the scaling is in the reverse direction.

Processing the end of the interval

The definition of intervals and corresponding interval-based calculation functions are an absolute requirement for calculating mean, minimum and maximum values. ST7cc interval end processing is designed for incoming ST7 messages containing event-oriented time-stamped process values received staggered over time. A mean, minimum and maximum value calculation over a processing interval can only be completed when a process value is received with a time stamp that falls logically into the next interval. This is illustrated in the figure below.

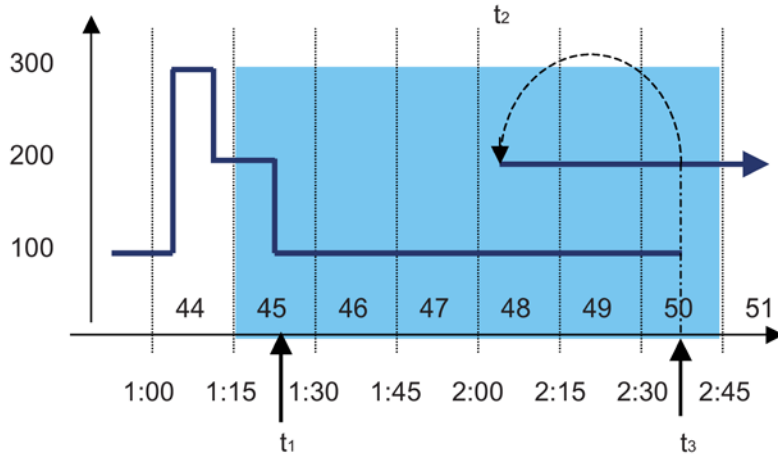


Figure 4-76 Interval-related measured value processing

The processing interval 44 is completed by the value received at time t_1 . A process value is received late compared with event-oriented process value acquisition (t_2) only at t_3 . With this supplied value, the ST7cc interval end processing function can process intervals 45, 46 and 47; in other words calculate the required values. Interval 48 can only be completed when a value is received whose time stamp is logically located in interval 49.

AVG

On the basis of time-stamped values that arrive time-staggered, the mean value can be calculated over a selectable time unit (interval). Mean value calculation is achieved by weighting individual values proportional to their duration in the interval.

Due to the event-oriented data acquisition, the event time of an acquired measured value in the ST7 object processing does not normally match either the interval start or interval end. This means that start and end values of an interval are calculated values. The calculations (MIN, MAX, MOM) are made during the interval end of processing that initiates the mean value calculation.

As a result of the selected hysteresis, with values that change only slightly, it is possible that several intervals are completed by a measured value supplied by ST7cc. In this case, several result values arise in a time lapse.

MOM

When the interval end is processed, the instantaneous value that matches the interval end is calculated, for example the value 200 at the completion of interval 44 in the figure. If the currently received measured value needs to be entered in the archive immediately along with its time stamp, the interval length must be set to a value of 0 seconds.

MIN

If the interval end is process, the minimum value is calculated from the set of values belonging to the interval including the MOM, for example the value 100 at the completion of interval 44 in the figure.

MAX

If the interval end is process, the maximum value is calculated from the set of values belonging to the interval including the MOM, for example the value 300 at the completion of interval 44 in the figure.

Mean value calculation is achieved by weighting individual values proportional to their duration in the interval.

AV2

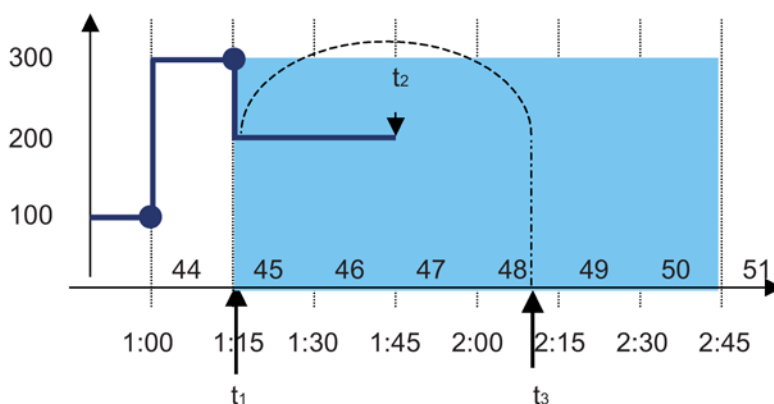


Figure 4-77 Interval-related measured value processing (AV2)

In AV2 mean value calculation, the mean value is already calculated at the automation level. The mean value is transferred as an instantaneous value. In the mean value calculation in ST7cc (AVG), the instantaneous value is future-oriented as of the time stamp until a new instantaneous value is received. In AV2 mean value calculation, for ST7cc the instantaneous value is past-oriented in this case and is valid from the last value supplied until the time stamp of the current received value. This is illustrated in the figure (t_1 = time stamp of the last received value, t_2 = time stamp of the time-staggered value current at time t_3).

In practice, the time stamp at the automation level cannot be calculated exactly at the set interval end times. It will deviate in the order of milliseconds. ST7cc interval processing recognizes this even if the difference is very slight and processes the transferred mean value

(even distribution of the value according to the proportion of the interval over time). Due to this procedure, the mean values calculated at the automation level will normally change minimally after they are received by ST7cc.

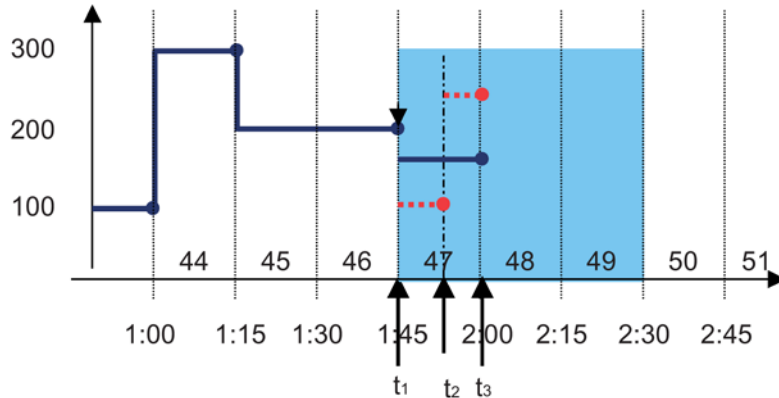


Figure 4-78 Interval-related measured value processing (AV2)

The figure shows how mean value calculation at the automation level is interrupted by a general request. Due to the general request, a partly complete main value is calculated at the time t_2 , transferred to the target subscriber, and internally the value is reset to zero. When the interval end time is reached, the mean value calculation is repeated and transferred and this represents the second portion for the monitored interval. Based on the even distribution, the ST7cc mean value calculation calculates the interval-related "complete" mean value in the interval end processing.

Setting parameters for measured value processing

How to create a processing function is described in Working with a processing function (Page 233). The figure shows the ST7cc Config window with the input boxes for assigning parameters to the measured value processing function (window area: Parameter block details). Since the variable is of the measured value type, the parameter block for measured value processing is assigned to it automatically.

Note

One measured value processing function can be assigned to a variable. If the measured value transferred by SINAUT needs to be handled by several measured value processing functions (MIN, MAX, AVG etc.), the transferred data subarea must be mapped to several variables.

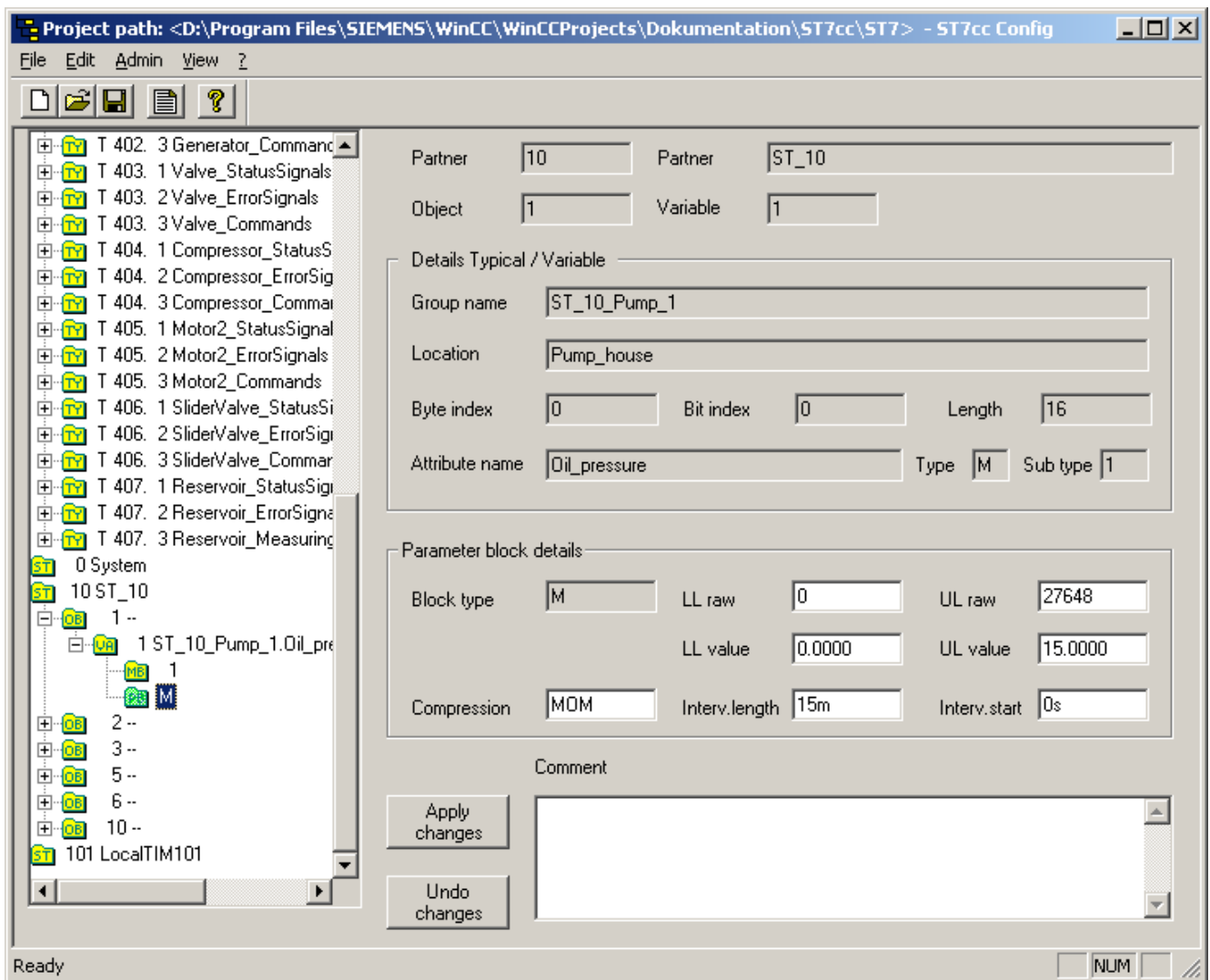


Figure 4-79 Inserting a processing function

LL raw / UL raw

Lower / upper data point of the raw value for conversion to a physical value.

LL value / UL value

Lower / upper data point of the physical value for conversion of the raw value to its physical value.

Compression

The Compression parameter allows the following entries:

- **MOM** Calculation of the instantaneous value at the end of the interval
- **MIN** Calculation of the minimum value within the interval
- **MAX** Calculation of the maximum value within the interval

- **AVG** Calculation of the mean value over the interval
- **AV2** Acceptance of the mean value calculated at the automation level

Time units for interval length and interval start:

The intervals for interval quantity calculation are defined by the interval length and a starting point within the interval. The time units used are:
W week, D day, h hour, m minute, s second.

Interval length:

The following values are permitted as the interval length:

1W, 1D, 1h, 30m, 20m, 15m, 12m, 10m, 6m, 5m, 4m, 3m, 2m, 1m, 0sec

Note

If you select a length of 0 seconds, the current value is passed on for archiving immediately without calculating an interval quantity.

Interval start:

Lower / upper data point of the raw value for conversion to a physical value. The unit for the start time must be less than that for the interval length and must be within the interval length.

Example:

An interval 1T with a start time of 6h defines a daily interval starting at 6 a.m.

Note

The changeover from daylight saving time to stand time is handled as follows: The intervals 1W or 1D are increased or reduced by one hour. For all other intervals, there are additional interval entries or the entries are omitted.

4.5.6 Archive

Archiving can be used for all variable types. The result of a processing function or (with signals) the signal value is archived.

To allow archiving, the variable must be assigned to a WinCC archive using an archive name and an archive variable name.

- **Archive name:** In preparation, an archive with the required archive name must be created in WinCC.
- **(Archive) variable name:** The variable name is used as the archive variable name as default by ST7cc and can be changed by the configuration engineer.

Additional parameters that will be used by the WinCC trends are as follows:

- Name of the physical unit of the scaled value
- Lower scale limit
- Upper scale limit

The figure shows how this data is entered in ST7cc Config:

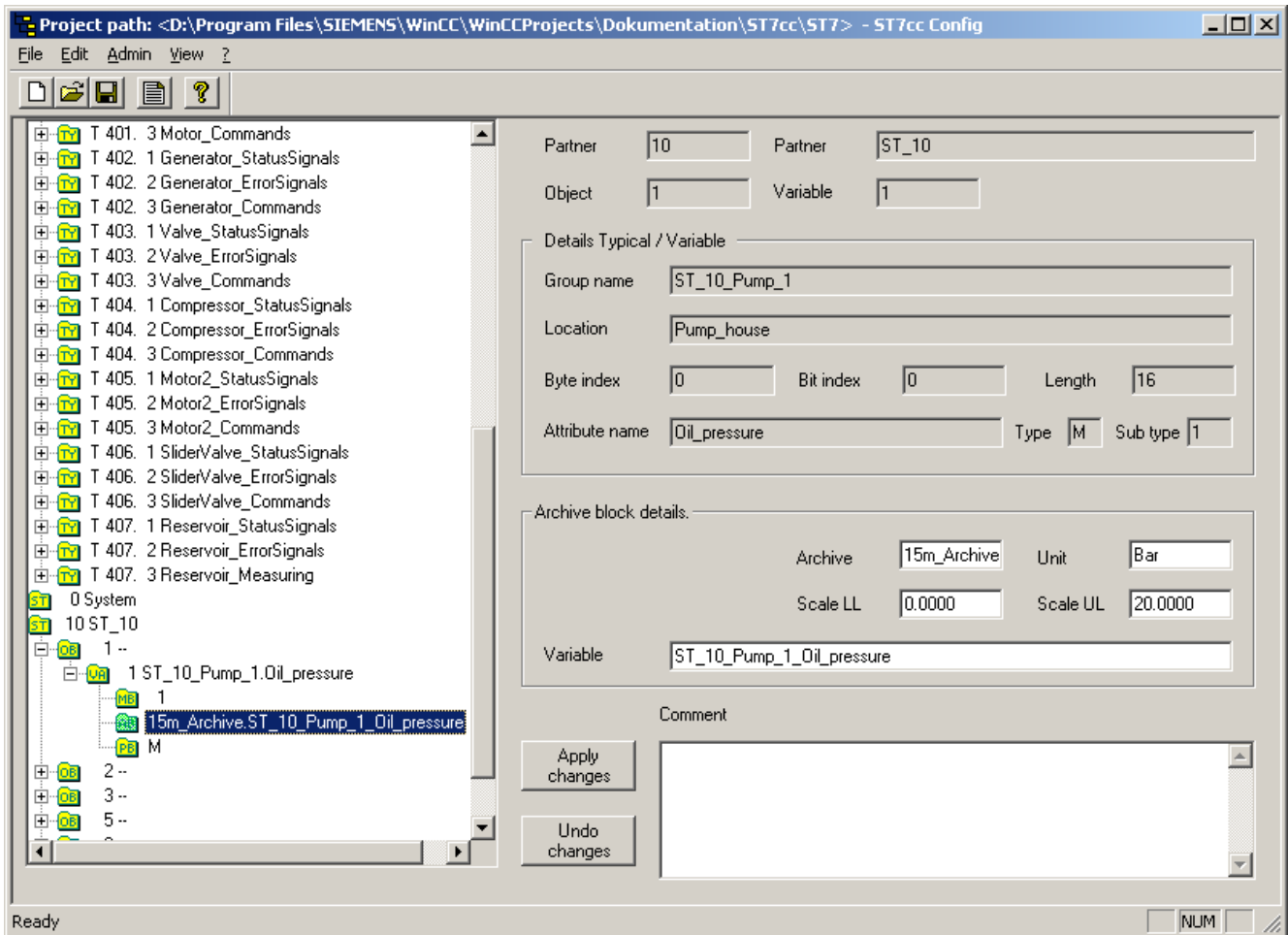


Figure 4-80 Archive processing in ST7cc

4.6 Variable list

Overview

In the object tree (see section ST7cc object tree (Page 181)), you can navigate to the level of the variables by selecting a subscriber and selecting an object (decoding, object template, typical).

To get an overview of all the variables, you can display the variable list.

Variable list

You can select the variable list in two ways.

1. Select the View button in the toolbar and open the menu (see figure).
2. Select the *Variable List* option.

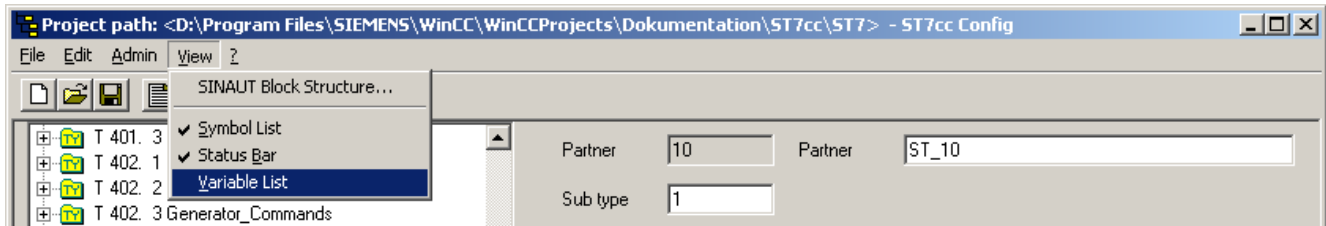



Figure 4-81 ST7cc Config dialog

As an alternative, you can select the variable list from the toolbar by clicking the  button.

Whichever method you choose, the variable list is displayed in the ST7cc Config screen (see figure). The group name and the attribute name are separated by (.) as the delimiter.

The variables are listed in alphabetical order. Variable names are retained 1:1 in the copy when you copy decodings and subscribers and it is possible for the configuration engineer to forget to update the subscriber or group names when inserting the copies. In this case, all the variables that occur more than once are listed at the beginning of the variable list and each of the variable names preceded by (!!). When you select the variable, the detailed parameters are displayed in the ST7cc Config screen so that you can make the necessary correction.

Note

The algorithm only detects that variable names occur twice or more. It cannot distinguish between the original and copy or copies. You yourself must check whether the variable name marked with (!!)

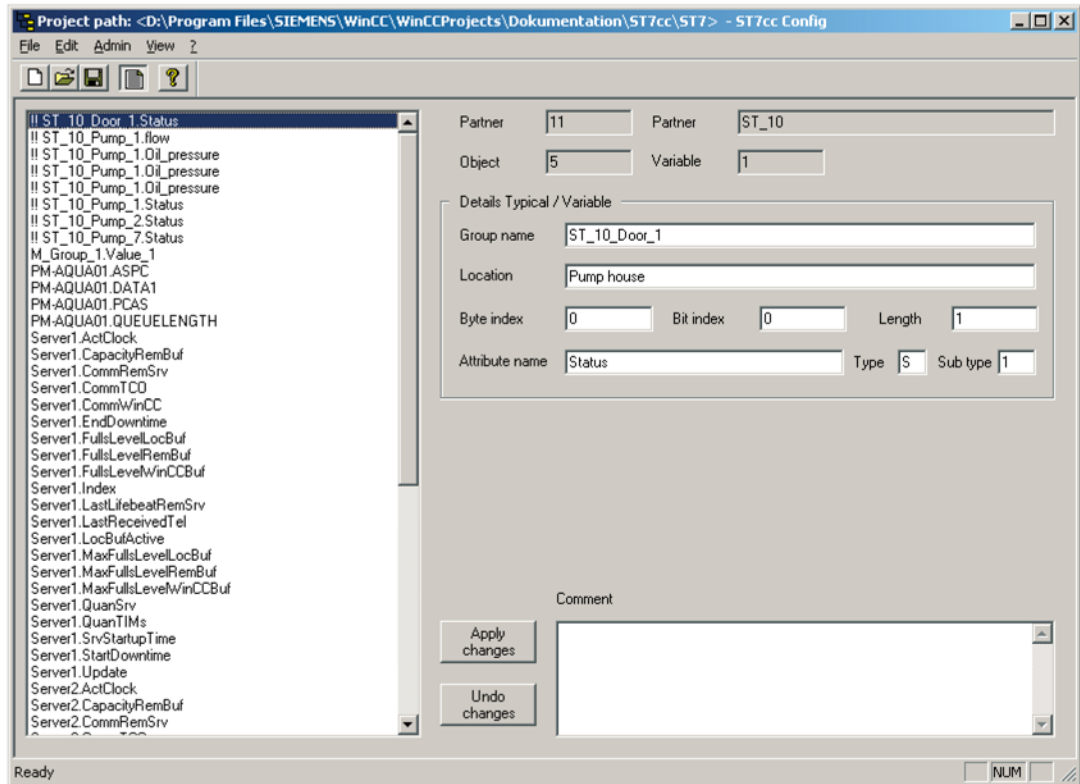


Figure 4-82 Variable list

4.7 SINAUT TD7 block structure in ST7cc Config

The following steps describe selecting the SINAUT TD7 block structure in ST7cc Config. The database is the SINAUT TD7 block structure that you saved in a file earlier with the *SINAUT ST7: Diagnostics and Service* tool. For the required steps, please refer to the description: SINAUT ST7 System Manual, 'Diagnostics and Service Tool'.

Note

The current SINAUT TD7 block structure is saved manually in the *SINAUT ST7: Diagnostics and Service* tool and you should make sure that you have saved the latest version of the SINAUT TD7 block structure for ST7cc Config.

Selecting the SINAUT block structure

1. Open the SINAUT TD7 block structure dialog by pointing to the View menu in the ST7cc Config dialog and selecting the SINAUT TD7 Block Structure... menu command (see figure).

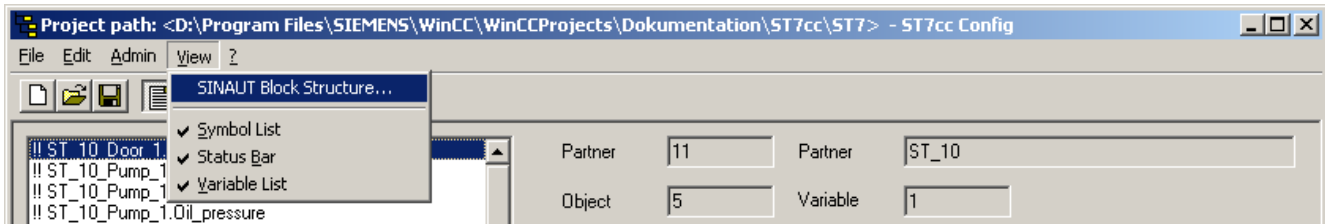


Figure 4-83 Selecting the SINAUT TD7 block structure

2. In the *Select SINAUT TD7 block structure XML file...* dialog, enter the path and file name under which you saved the relevant file and click *Open*.

The *SINAUT TD7 block structure* dialog opens.

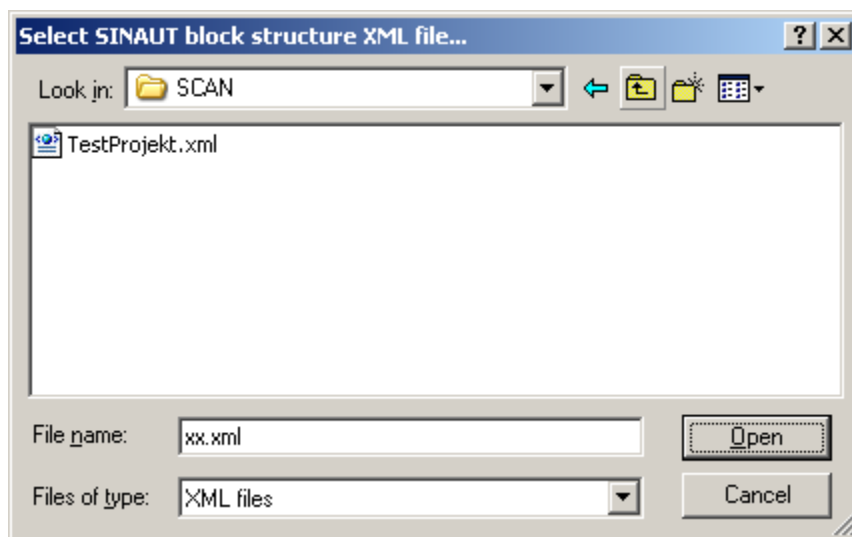


Figure 4-84 Dialog for selecting the SINAUT TD7 block structure

SINAUT TD7 block structure dialog

The SINAUT TD7 block structure dialog contains the TD7 Statistics, TD7 Overview and Send/ Receive Block List tabs (see figure).

Send/Receive Block List tab

In the *Send/ Receive Block List* tab, you can select the list of send and receive blocks. This provides information on which SINAUT subscribers and SINAUT objects exist in the selected *TD7 block structure* and which decodings have already been created in ST7cc Config.

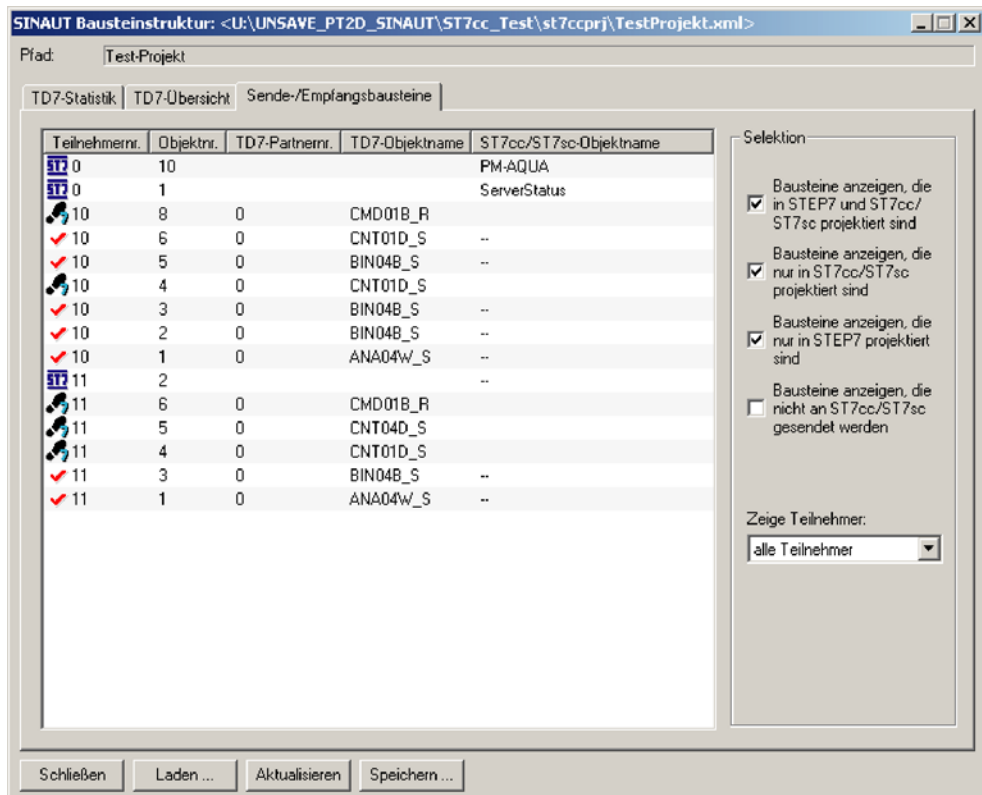


Figure 4-85 Send/Receive Block List tab

The *Send/Receive Block List* tab contains a table with object entries that describe whether

- there is only one SINAUT object in the TD7 block list (icon: 📡), or
- there is only one decoding in ST7cc Config (icon: ST7), or
- there is a SINAUT object with its decoding (icon: ✓).

The use of icons makes it immediately clear to the user which SINAUT objects have already been decoded and which decodings do not yet have any SINAUT objects.

- Description of the table columns:
 - Subscriber no.: The subscriber number of the CPU in which the listed SINAUT objects exist.
 - Object no.: The object number of the listed SINAUT object; in other words, the number of the instance DB on the CPU.
 - TD7 partner no. The TD7 partner number is the subscriber number to which the SINAUT object sends or from which it receives data. In ST7cc Config, only the SINAUT objects for which the partner number is identical with the SINAUT subscriber number of ST7cc need to be decoded. If 0 (multicast address) is displayed as the partner number, you will need to make a separate check to find out whether the object is relevant for ST7cc.
 - TD7 object name: Name of the SINAUT object types
 - ST7cc / ST7cc object name: Optional name that can be assigned to a decoding of a SINAUT object by the user. Compare section Creating a decoding (Page 210).
- Selection options
Various selection options allow you to filter the table entries displayed so that subsets of the total entries can be selected. These options are as follows:
 - Show blocks configured in STEP 7 and ST7cc / ST7sc: The only table entries displayed are those in which every SINAUT object also has a decoding.
 - Show blocks configured only in ST7cc / ST7sc: The only table entries displayed are those that only contain decodings.
 - Show blocks configured only in STEP 7: The only table entries displayed are SINAUT objects that do not yet have a decoding.
 - Show blocks that do not communicate with ST7cc / ST7sc: The only table entries displayed are SINAUT objects that communicate only with other SINAUT subscribers and not with this ST7cc. No decodings are necessary for these objects in ST7cc.
 - With the Show subscribers list box, you can display all the subscribers that occur in the send/receive block list. At the same time, you can select a subscriber so that the table entries are limited to the selected subscriber.
- Load button
 - Load means that you can select and open another TD7 block list without closing the dialog.

4.8 Generating for WinCC

Overview

Generating WinCC parameter assignments means that the parameters of processing functions executed in WinCC are transferred via the ODK interface to the WinCC target components (see figure). ST7cc Config provides the following generating options:

- WinCC generation. This includes:
 - WinCC tag management
 - WinCC messages
 - WinCC archive variables
- Generating subscriber picture typicals

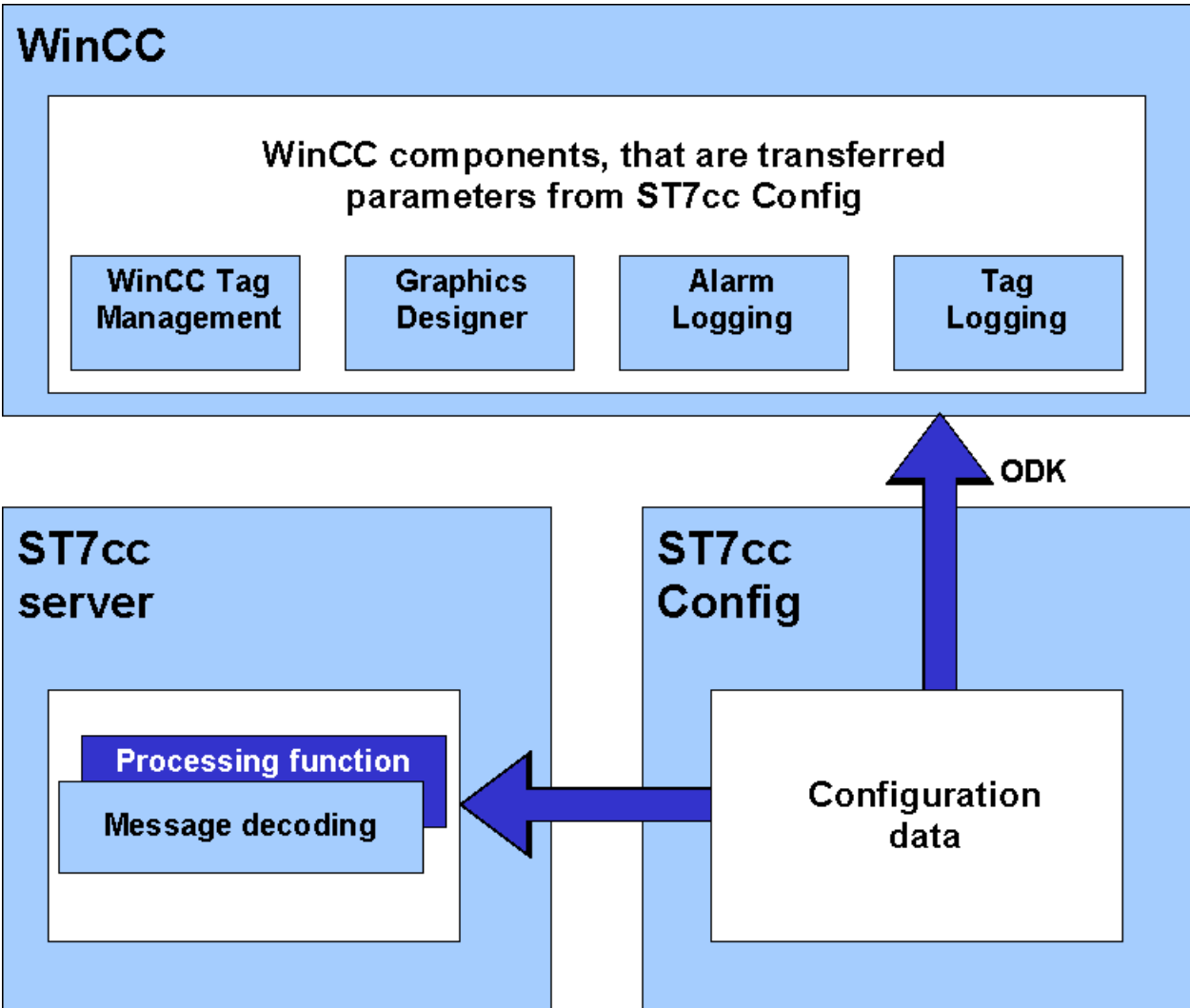


Figure 4-86 Transferring parameters to the processing components

Prerequisites for generation

Before generating variables, messages or archives, always make sure that the following prerequisites are met:

- The project in which you want to generate is set as the current WinCC project in configuration mode (must not be in runtime mode).
- The standard language of the project is activated.
- The channel DLL for the ST7 server (ST7.DLL) is declared in the project.
- The message classes and message types have been created in WinCC for the system and user variables.
- The user archives have been created in WinCC.
- Prior to generating, close the editors "Alarm Logging" and "Tag Logging".

During the first generation run, ST7cc Config registers with all WinCC components.

It deregisters only when you close the generation dialog.

4.8.1 Generating


Generating

You can trigger the full WinCC generation in the following menus:

- In the main menu
- In the subscriber shortcut menu, object shortcut menu or variable shortcut menu

4.8.2 The tag management

Checking the variable names

By clicking on the *Variable list* icon  or selecting the menu command *View > Variable list*, you can check which variables will be created and whether duplicate names exist before you generate.

Duplicate names are indicated in the variable list by two preceding exclamation points (!!). During generation, only the first variable with this name would be created and you must therefore resolve the naming conflict first by renaming one of the variables.

The variable list can be useful as an aid to navigation helping you to locate logically related variables in different communication objects. When you change between the variables and the object tree, the current variable selection is retained.

Generation of the system and user variables

Prior to generating the variables of a subscriber, a logical connection within the channel unit *ST7 server* is created under the name of this subscriber and the system variables are created that contain the essential status information of this subscriber.

The newly created user variables are then assigned to this logical connection. The logical connection receives the subscriber address of the subscriber as an address parameter.

Note the following situations from the ST7cc perspective when generating variables:

1. A new ST7cc variable is newly generated in WinCC.
2. If the variable name of an ST7cc variable is changed, the 'old' variable remains in WinCC, the changed ST7cc variable is created as a new variable under its modified name. This mechanism can lead to an accumulation of WinCC variables that are no longer required.
3. It is not possible to delete a WinCC variable from within ST7cc.

Generation of the variable groups

The group name is used both to form the WinCC tag groups and as the name prefix for the WinCC tag name. With typical-related variables, the name of the typical instance defines the group name.

Note

If a variable group in WinCC is already declared as an internal group or declared within a different logical connection, the variables can be created even if the variable name itself has not yet been assigned in WinCC.

Generation of the individual variables

The WinCC variable name is obtained from the combination of group name and attribute name in ST7cc Config.

The data type of the variable is obtained from the processing type on the ST7 server: See also section Type and subtype of a variable (Page 165).

The channel-related address information of the variable is put together based on the number of the communication object, number of the typical instance (or 0 for variables that are not typical-related) and the consecutive number of the variable.

Note

Duplicate addresses can result from forgetting to delete variables with the same address information in WinCC. As a result, only the variable last registered by the data manager with the ST7 server is updated later by the data manager.

Working on generated variables later

You can edit generated variables, for example to declare a start value or a limit value check within the WinCC data manager.

If necessary, simply delete all the variables of the channel and generate everything again.

The generated tag management appears in WinCC, for example, as follows:

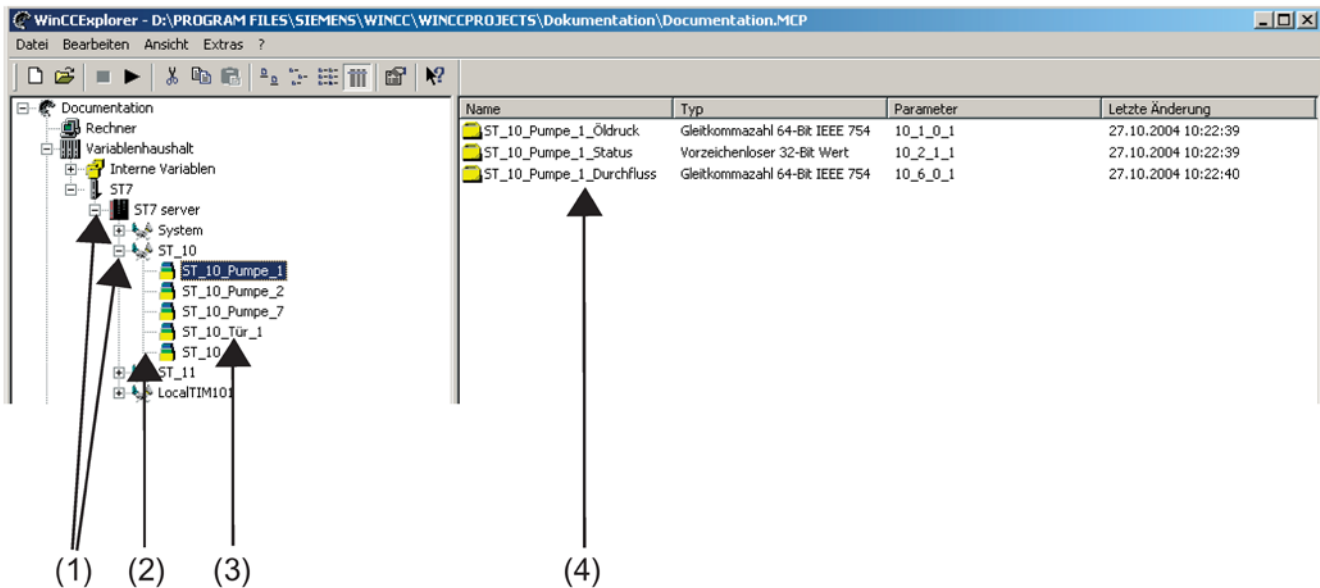


Figure 4-87 WinCC tag management

No.	Description
1	ST7 channel DLL + unit
2	Logical connection
3	Variable group
4	Variables

4.8.3 Message management

Before you generate messages, make sure that all the message classes and message types used in the project are defined.

Note

In WinCC, make sure that the Error message class is created with the message type Alarm and the Operation message class with the message type Check-back message, to allow the ST7cc system variables to output their messages.

In WinCC, enter the text "Check back message" in the "Name of message type" box.

Message numbers

The message number format selected in the settings specifies the composition of the message numbers for the messages to be generated.

This system ensures the following:

- that the message numbers are unique throughout the project in WinCC,
- that the WinCC message can be assigned to the object in the object tree again

For more detailed information on message number format and the use of the text blocks, refer to section Message processing (Page 235).

Note

Please make sure that in the initialization file of the ST7cc server, the same message number format is entered that you used to generate the messages.

Avoid possible conflicts in the message number assignment if you generate your own messages directly in WinCC.

Allocation of message blocks

The following configurable texts can be assigned to the available message text blocks: subscriber name, group name, variable name, location, message text.

It is advisable to set the message text to text block 1 (WinCC default). Text block 2 (point of error) can either contain the location or the cause of the problem.

If several texts are assigned to the same text block, they are appended in the order shown above.

Special features

The generated messages are not assigned to any data manager variables because the message system is supplied separately from the data manager and the acknowledgment is made locally in WinCC.

The following example shows an excerpt from the generated message management:

...	Nummer	Klasse	Art	MeldeVariable	MeldeBit	Zustandsvariable	Zustandsbit	Meldetext	Störort	Block: 3	Block: 4	Block: 5	Block: 6
	10002111	Betrieb	Rückmeldung	ST_10_Pumpe_1_St:1		ST_10_Pumpe_1_Status_5	1	Hand	ST_10	ST_10_Pumpe_1	Status	Pumpenhaus	statischer Text für 5
	10002112	Betrieb	Rückmeldung	ST_10_Pumpe_1_St:2		ST_10_Pumpe_1_Status_5	2	Aus	ST_10	ST_10_Pumpe_1	Status	Pumpenhaus	statischer Text für 5
	10002113	Betrieb	Rückmeldung	ST_10_Pumpe_1_St:3		ST_10_Pumpe_1_Status_5	3	Ein	ST_10	ST_10_Pumpe_1	Status	Pumpenhaus	statischer Text für 5
	10002114	Betrieb	Rückmeldung	ST_10_Pumpe_1_St:4		ST_10_Pumpe_1_Status_5	4	Revision	ST_10	ST_10_Pumpe_1	Status	Pumpenhaus	statischer Text für 5
	10002115	Betrieb	Rückmeldung	ST_10_Pumpe_1_St:5		ST_10_Pumpe_1_Status_5	5	Örtlich	ST_10	ST_10_Pumpe_1	Status	Pumpenhaus	statischer Text für 5
	10002116	Betrieb	Rückmeldung	ST_10_Pumpe_1_St:6		ST_10_Pumpe_1_Status_5	6	Gesperrt	ST_10	ST_10_Pumpe_1	Status	Pumpenhaus	statischer Text für 5
	10002210	Betrieb	Rückmeldung	ST_10_Pumpe_2_St:0		ST_10_Pumpe_2_Status_5	0	Automatik	ST_10	ST_10_Pumpe_2	Status	Pumpenhaus	
	10002211	Betrieb	Rückmeldung	ST_10_Pumpe_2_St:1		ST_10_Pumpe_2_Status_5	1	Hand	ST_10	ST_10_Pumpe_2	Status	Pumpenhaus	
	10002212	Betrieb	Rückmeldung	ST_10_Pumpe_2_St:2		ST_10_Pumpe_2_Status_5	2	Aus	ST_10	ST_10_Pumpe_2	Status	Pumpenhaus	

Figure 4-88 WinCC message management

No.	Description
1	Subscriber name
2	Group name
3	Attribute name

4.8.4 Archive tags

Before you generate archive tags, make sure that all the archives used in the project are defined.

Generated archive tags

The following fields of the archive tags are used during creation and specified in the processing rule:

- Name
- Unit
- Lower / upper limit of the scaling
- The name of the data manager tag is entered as a comment.

The following example shows an excerpt from the generated archive assignment:

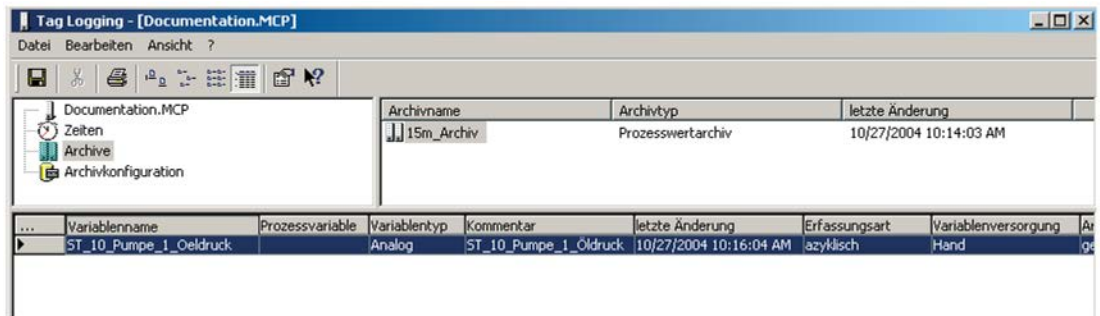


Figure 4-89 Archive assignment

Special features

An archive variable is not assigned to a data manager variable since the archive system is supplied separately from the data manager. The type of archiving is always *acyclic* and the variable type *analog*.

In archives filled by the ST7cc server, the trend display is not currently updated online. To obtain the most recent values, therefore, the dynamic updating of the trend must be turned off and the trend manually updated by scrolling to the end of the display area.

4.8.5 Generating subscriber picture typicals

Basic function

The picture typicals for SINAUT subscribers (local TIMs and stations) created in ST7cc Config are generated automatically. This means that picture typicals are created in the `st7_project.pdl` file for every subscriber based on the standard picture typicals in the `st7_typical.pdl` file in the WinCC project. When they are created, the relationship to the name (subscriber name) is established at the same time.

Note

When you install ST7cc, the picture typical and faceplate files are stored in the ST7cc directory.

With the Copy faceplates to a WinCC project function (see section Copy faceplates to a WinCC project (Page 116)), you must decide whether you want to use the latest picture typicals and faceplates or want to continue working with those already stored in the WinCC directory.

Please note the update scenarios in section System typicals (Page 184) section on update scenarios.

Inserting the subscriber picture typicals in process pictures

To be able to use the picture typicals of the subscribers in your process picture, follow the steps below in the WinCC Graphics Designer:

1. Copy the picture object *FPL* from the *st7_project.pdl* file into your process picture.
2. Select the Create project picture typicals option. This generates the picture typicals for all the SINAUT subscribers in the project and stores them in the `project_typical.pdl` file.

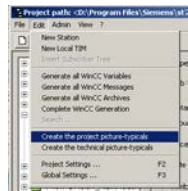


Figure 4-90 Generating subscriber picture typicals

3. Copy the subscriber picture typicals from the `project_typical.pdl` file into your process picture.

4.8.6 Generating technological picture objects

Basic function

To simplify configuration, you will find technological typicals (ST7cc typicals and picture typicals for WinCC) in section Technological typicals (Page 333). If you create your technological objects based on these typicals, you can generate the corresponding picture objects. This means that based on the picture typicals in the st7_technicalobjects.pdl file, a picture object is created in the project_technicalobjects.pdl file in the WinCC project for every technological object. When these are created, the relationships of the names to the WinCC tags are also established.

Note

When you install ST7cc, the picture typical and faceplate files are stored in the ST7cc directory.

With the Copy faceplates to a WinCC project function (see section Copy faceplates to a WinCC project (Page 116)), you must decide whether you want to use the latest picture typicals and faceplates or want to continue working with those already stored in the WinCC directory.

Note the update scenarios in section System typicals (Page 184) section on update scenarios.

Inserting the technological picture objects in process pictures

To be able to use the picture typicals of the technological objects in your process picture, follow the steps below in the WinCC Graphics Designer:

1. Copy the picture object FPL from the project_technicalobjects.pdl file to your process picture.
2. Select the Create the technical picture typicals option. This generates the picture typicals for all the technological objects in the project and stores them in the project_technicalobjects.pdl file.

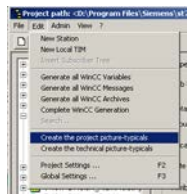


Figure 4-91 Generating picture typicals for technological objects

3. Copy the technological picture objects from the project_technicalobjects.pdl file to your process picture.

4.8.7 Inserting picture typicals and faceplates in process pictures

When you install ST7cc, the picture typicals and faceplates described above are copied to the GraCS subdirectory of your ST7cc installation directory.

ST7cc server

5.1 ST7cc server

This chapter is intended to help the user to correctly evaluate the dynamic response of the ST7cc server in conjunction with WinCC when commissioning a plant.

The ST7cc server is the runtime component of ST7cc. The ST7cc server communicates with and monitors the TIMs connected locally over the MPI bus or Ethernet. It receives the incoming messages and maps the data to the ST7cc variables. The ST7cc variables represent the process image. This contains all the process data and all the status data of the SINAUT subscribers in the network.

To be able to supply WinCC with events arriving staggered over time chronologically correctly according to the time of the events, further processing is necessary (see section Configuring processing functions (Page 231)). From a WinCC perspective, this processing represents preprocessing.

5.1.1 Components and functions

Essential program components

shows the essential program components of the ST7cc server on the basis of which the system response in interaction with WinCC can be described in a single and a redundant system installation. Terms repeated often such as WinCC tag, channel DLL, ODK, ST7cc variable are described in the Glossary.

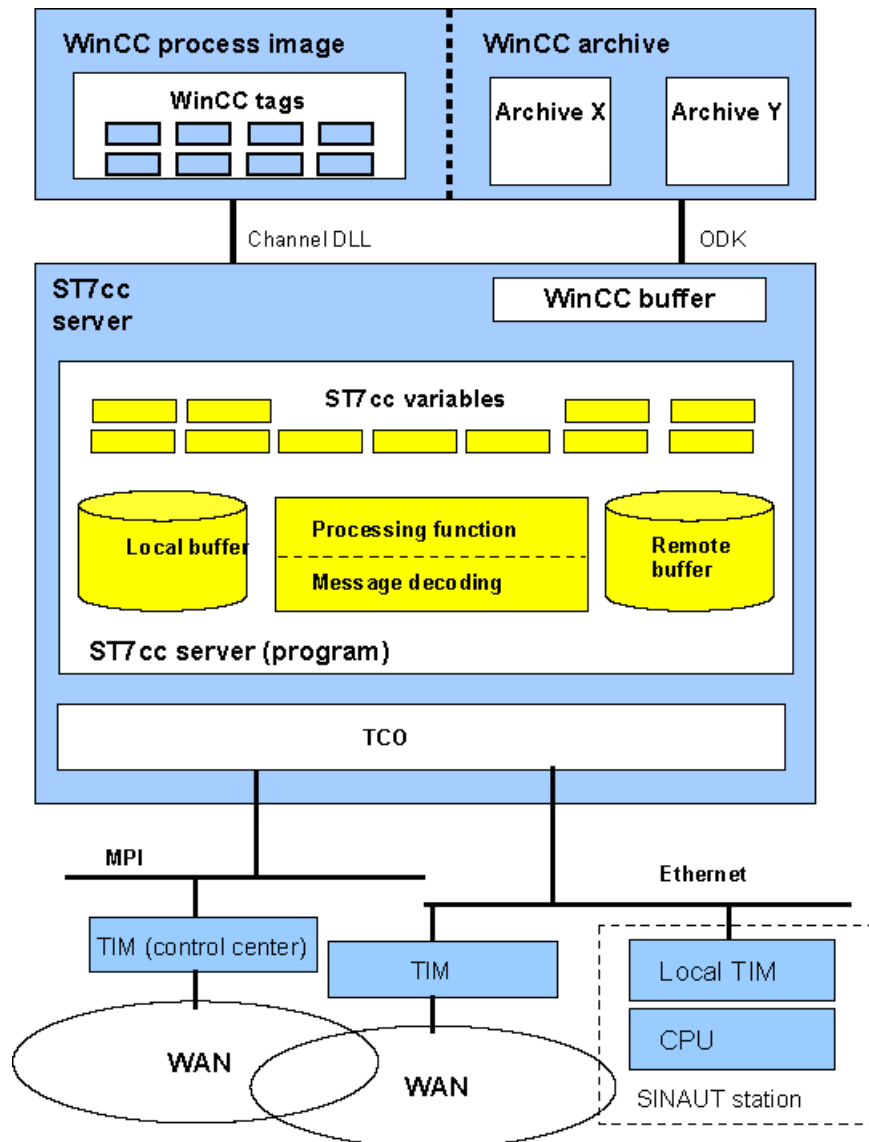


Figure 5-1 Essential program components

ST7cc server

The ST7cc server is the runtime component of ST7cc. The ST7cc server communicates with and monitors the TIMs connected locally over the MPI bus. It receives the incoming messages and maps the data to the ST7cc variables. The ST7cc variables represent the process image. This contains all the process data and all the status data of the SINAUT subscribers in the network.

WinCC buffer

Message and archive processing can be assigned to the ST7cc variables. If this is the case, individual messages or archive data are generated in ST7cc, that are transferred over the ODK interface to Alarm Logging or Tag Logging for further processing. As of WinCC version V5.2, individual messages are generated as default by WinCC.

The results of processing an ST7cc variable can, however, accrue faster than they can be accepted by WinCC. The WinCC buffer therefore takes the WinCC jobs from the ST7cc processing and in doing so separates the asynchronous procedures of job creation and job processing.

TCO (TIM Connect)

The TCO component monitors the local TIMs connected over the MPI bus or Ethernet. The TCO component maps the most important status information on ST7cc variables, passes received messages to *message decoding* or passes messages for transmission to the TIM for WAN communication or to the connected SINAUT stations over Ethernet. When the ST7cc server starts up or restarts, the TCO component determines which local TIMs can be reached.

Local buffer

If the ST7cc server cannot pass on its data to WinCC, all messages (ST7 data messages and organizational messages) are stored in the local buffer. Once WinCC becomes available again, the buffered messages are processed. This mechanism achieves two aims:

- That the master station is accessible from the perspective of the stations even when WinCC is not available.
- That general requests as a result of temporary deactivation of WinCC can be avoided.

Remote buffer

The remote buffer is necessary to ensure data consistency when using a redundant ST7cc system.

The remote buffer is set up only for redundant ST7cc. Whether or not redundant operation is required is recognized by the ST7cc server based on the configuration data (see section Project settings: Server (Page 118)).

The remote buffer is organized as a ring buffer and records all incoming messages so that it can be used as a data source for the redundant partner during a restart. If the partner of a redundant ST7cc system starts up again, it can recognize the period for which messages are missing and can request these from the redundancy partner.

5.2 Process image of the ST7cc server

The ST7cc server maintains a persistent process image; in other words, the values of the ST7cc variables are retained even after the server restarts.

5.3 ST7cc redundancy package:

When the configuration is changed, the process image is regenerated. Values of variables are then taken from the old process image if their type and address information have not been changed.

This means that after a restart:

- The measured value and counted value processing can be continued correctly
- Only the messages are triggered whose output value has really changed

5.3 ST7cc redundancy package:

Redundant ST7cc

The ST7cc redundancy functionality is described below. The figure shows the redundant systems A and B.

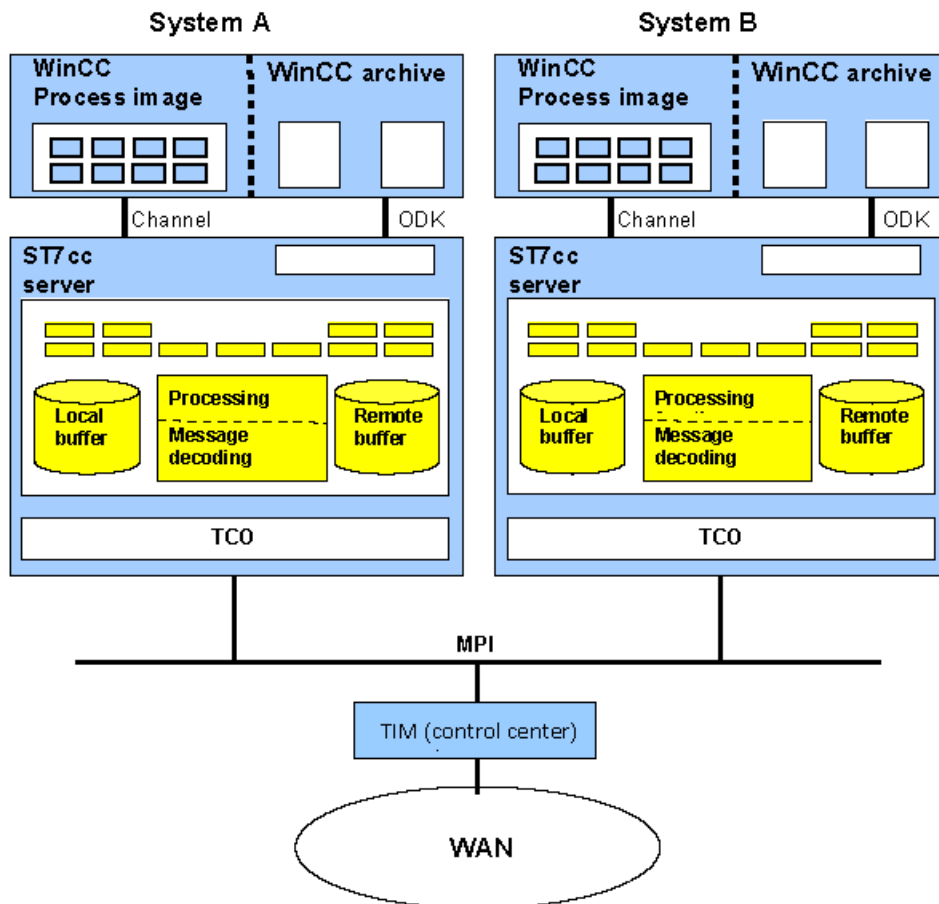


Figure 5-2 Redundant ST7cc

Basic concept

Systems A and B run parallel to each other. The TIMs connected to the local MPI bus or Ethernet always supply both ST7cc target systems.

Note

If the local TIM has two Ethernet ports (TIM 4R-IE), only one of these ports can be used to connect to the redundant ST7cc system over a common bus.

For a TIM, the redundant partner has been reached successfully when one partner can be reached.

Each of the systems in the redundant arrangement can handle the non-availability of its WinCC partner using its WinCC buffer.

If an ST7cc server fails and then returns, it can request the ST7 messages it has "missed" from the *remote buffer* of its redundancy partner and process them just like the messages from its local buffer in "time lapse".

System status vector

The WinCC-A, ST7cc-A, WinCC-B, and ST7cc-B components represent the redundant system. To be able to describe the system statuses briefly, a system status vector (WinCC-A, ST7cc-A, WinCC-B, ST7ccB) is defined. The status (1,1,1,1) means that all four components are available. The system status (0,1,1,1) means that WinCC-A is not available but the remaining components are.

The following situations are distinguished:

1. System status: (1,1,1,1)
2. System status: (0,1,1,1), analogously also (1,1,0,1)
3. System status: (0,0,1,1), analogously also (1,1,0,0)
4. System status: (1,0,1,0)
5. System status: (1,0,0,0), analogously also (0,0,1,0)
6. System status: (1,1,1,0), analogously also (1,0,1,1)
7. System status: (0,1,0,1)
8. System status: (0,0,0,1), analogously also (0,1,0,0)
9. System status: (1,0,0,1), analogously also (0,1,1,0)
10. System status: (0,0,0,0)

Communicating the system statuses

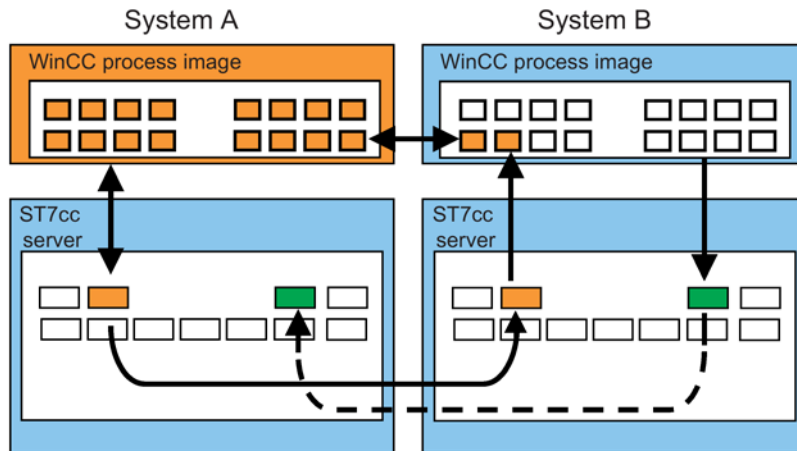


Figure 5-3 Communicating the system statuses

shows that the exchange of system statuses is duplicated:

- The WinCC redundancy functions (WinCC redundancy) recognize when their redundancy partner is not available. The WinCC mechanisms for ensuring WinCC data consistency (WinCC Archive) also require this system performance.
- The non-availability of WinCC A is also detected by the ST7cc Server and mapped to an ST7cc variable. This information is also passed on by the ST7cc server to its redundant partner where it is mapped to a WinCC tag.

With the data exchange over two paths, we therefore have the WinCC view on the redundant partner that does not know ST7cc and the view how ST7cc sees its local WinCC partner. Both views are displayed to the user in the system faceplate. Generally, both views show identical system statuses. If two faults occurred, different results may, however, arise. Example: WinCC-A is available at the time when the ST7cc server A fails and then fails itself. In such a situation, WinCC-B would indicate the non-availability of its redundancy partner, the ST7cc view of WinCC-A would be incorrect in this case because it is not up-to-date. From the information that ST7cc-A is not available, the user must conclude that the ST7cc information is out of date because of a double fault.

5.3.1 General requests (GR) when starting up the redundant system

General requests (GR) when starting up the redundant system

In a redundant ST7cc system, server 1 (for example, system A) is specified as the default master and server 2 (for example, system B) as the default slave.

When an ST7cc system starts up, it always runs a general request (GR) on its local TIMs to obtain information on the reachability of all subscribers by means of organizational messages. These general requests are always necessary and are not what is meant in the following paragraph .

Startup scenarios and corresponding GR processing

Three startup scenarios can be distinguished:

- System A is started while system B is running.
There is no GR because system A can update over system B.
- System B is started while system A is running.
There is no GR because system B can update over system A.
- Both ST7cc systems (A and B) start up at the same time. The initiated GR is first send to the ST7cc partner system for every reachable station to check whether or not the partner has already sent a GR to this station.
The partner system then checks whether or not it has the corresponding station marked as being reachable and, if it has, whether or not it has already sent a GR to this station. If this is the case, no further GR is triggered because the ST7cc server can update the querying partner.
If the relevant station is reachable, but no GR has been triggered since the restart, a GR is sent.
If the station is not reachable, the GR is returned to the original system that now sends the GR to the station itself (assuming that the station can be reached by the original system, which should normally be the case).

5.3.2 Description of the system statuses (redundant system)

System status (1,1,1,1)

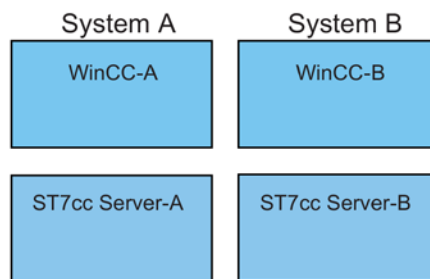


Figure 5-4 System status (1,1,1,1)

Systems A and B receive and process the incoming ST7 messages. The messages are entered in the remote buffer (ring buffer organization) to be able to close gaps in the data of a partner restarting following a down time.

The system and process variables of the ST7cc server and the WinCC systems do not indicate a problem.

System status: (0,1,1,1), analogously also (1,1,0,1)

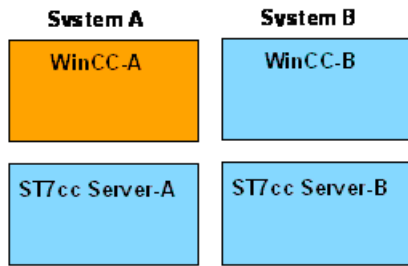


Figure 5-5 System status (0,1,1,1) / (1,1,0,1)

WinCC A is down, the other components ST7cc-A, WinCC-B, and ST7cc-B are available:

WinCC level:

Due to its redundancy mechanisms (WinCC Redundancy), WinCC-B recognizes that its WinCC partner is not available. This information is also made available to WinCC-B by ST7cc-B.

ST7cc level:

ST7cc server A detects the non-availability of WinCC-A. This is displayed in the appropriate system variable. The ST7 messages are redirected to the local buffer. System monitoring informs ST7cc server B of the non-availability of WinCC A and sets the appropriate system variable in the ST7cc process image.

WinCC A available again:

Since the ST7cc server was continuously available during the non-availability of WinCC-A, the WinCC data can be updated entirely from the local ST7cc buffer. Access to the remote buffer of ST7cc-B is not necessary.

System status: (0,0,1,1), analogously also (1,1,0,0)

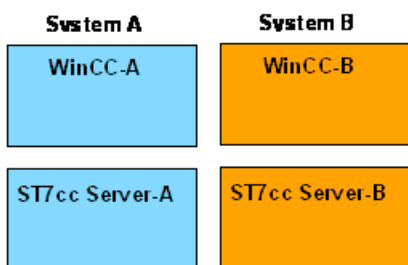


Figure 5-6 System status (0,0,1,1) / (1,1,0,0)

System B is not available, system A is functional.

WinCC level:

Due to its monitoring functions (WinCC Redundancy), WinCC A recognizes that its WinCC partner is not available. In this case, this information is also provided to WinCC B by ST7cc B, if WinCC A failed before ST7cc server A and ST7cc server A was able to signal the information to ST7cc server B. If the systems fail in the opposite order, this information can no longer be forwarded to the available redundant ST7cc server.

This means that depending on the order in which problems occur, there may be discrepancies in the detailed display in the system faceplate regarding the situation as seen by WinCC and ST7cc that the operator must be able to interpret based on background knowledge.

ST7cc level:

If WinCC B fails before ST7cc server B, this can be signaled to the ST7cc redundant partner by system monitoring. If the failure is in the opposite order, this "second path" is not possible.

Renewed availability of the failed components:

As soon as the failed ST7cc server A is available again, it can receive the ST7 messages and stores them in its local buffer until WinCC is available again. The WinCC redundancy mechanisms come into play when WinCC A becomes available again (synchronization of the WinCC archive data). At the same time, the processing of the frames from the local buffer begins and the gaps in the data are filled from the remote buffer of the redundant partner.

To ensure the chronological sequence of message decoding and processing, all incoming messages for the restarting ST7cc server are redirected over the local buffer until the present time is reached.

System status (1,0,1,0)

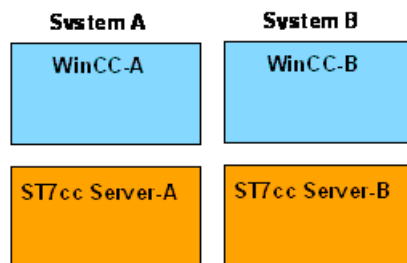


Figure 5-7 System status (1,0,1,0)

In this case, both WinCC systems are available but ST7cc servers A and B are not:

WinCC level:

Both WinCC servers detect the non-availability of their respective ST7cc server. Each WinCC system marks its process values as not up-to-date (no connection to the process). For WinCC itself, there is no failure because the WinCC system monitoring is designed to monitor its own availability.

ST7cc level:

As soon as the TIMs connected to the local MPI bus recognize that both redundant partners are unavailable, this is signaled to the stations. The station TIMs buffer their messages locally. The same applies to the SINAUT stations connected to Ethernet.

Renewed availability of the failed components:

As soon as one of the ST7cc servers (A or B) becomes available, it checks the accessibility of the stations and triggers a general request for them. All incoming ST7 messages are stored in the remote buffer and are available for the redundant ST7cc partner when it starts up to allow it to fill the gaps in its data.

System status: (1,0,0,0), analogously also (0,0,1,0)

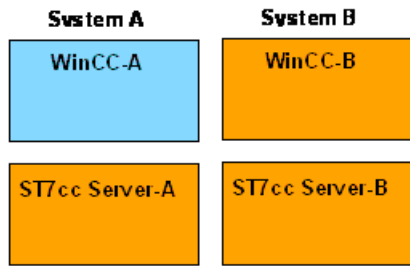


Figure 5-8 System status (1,0,0,0) / (0,0,1,0)

In this case, only one WinCC system is available but ST7cc servers A and B are not:

WinCC level:

The available WinCC server detects the non-availability of its WinCC partner and its ST7cc server. The available WinCC system marks its process values as not up-to-date (no connection to the process).

ST7cc level:

As soon as the TIMs connected to the local MPI bus recognize that both redundant partners are unavailable, this is signaled to the stations. The station TIMs buffer their messages locally. The same applies to the SINAUT stations connected to Ethernet.

Renewed availability of the failed components:

As soon as one of the ST7cc servers (A or B) becomes available, it checks the accessibility of the stations and triggers a general request for them. All incoming ST7 messages are stored in the remote buffer and are available for the redundant ST7cc partner when it starts up to allow it to fill the gaps in its data.

If the returned ST7cc server can communicate with its WinCC, the ST7 messages are processed and the information passed on to WinCC. If this is not the case, the messages are buffered in the local and remote buffer.

If the failed WinCC component becomes available again, the WinCC redundancy functions start synchronization of the WinCC data management. ST7 or the ST7cc redundancy mechanisms are responsible for the updating of the data stored in the stations or on the ST7cc server.

System status: (1,1,1,0), analogously also (1,0,1,1)

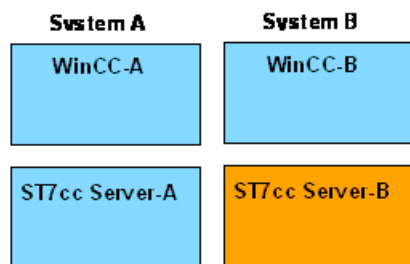


Figure 5-9 System status (1,1,1,0) / (1,0,1,1)

In this case, both WinCC systems are available but ST7cc server A or server B is not:

WinCC level:

Each WinCC server detects the non-availability of its ST7cc server. A WinCC system marks its process values as being not up-to-date if its ST7cc server is not available (no connection to the process). For WinCC itself, there is no failure because the WinCC system monitoring is designed to monitor its own availability.

ST7cc level:

The failed ST7 server does not receive any messages during the time of the failure and must fetch these from the remote buffer of its partner when it restarts.

System status (0,1,0,1)

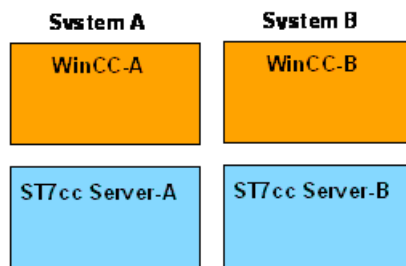


Figure 5-10 System status (0,1,0,1)

In this case, both WinCC systems are unavailable but ST7cc servers A and B are available:

WinCC level:

The user has no access to the technological process.

ST7cc level:

Each ST7cc server detects the non-availability of its WinCC partner. The incoming ST7 messages are stored in the respective local buffers. For the ST7 stations, the target nodes (ST7cc servers A, B) can be reached.

Renewed availability of the failed components:

Normally, both WinCC systems will not become available at the same time. The first available WinCC is updated by its ST7cc server based on the local buffer. As soon as the redundant partner is available, the WinCC redundancy functions synchronize the archive data that has gaps compared with the real process events. These gaps are closed by ST7cc.

If both ST7cc servers were continuously available during the WinCC downtime, each WinCC system is updated based on the respective local ST7cc buffer. The services of the remote buffer are not required.

System status: (0,0,0,1), analogously also (0,1,0,0)

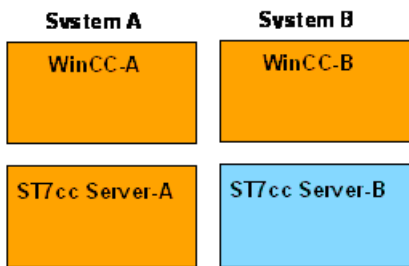


Figure 5-11 System status (0,0,0,1) / (0,1,0,0)

In this case, both WinCC systems are unavailable but ST7cc server A or server B is available:

WinCC level:

The user has no access to the technological process.

ST7cc level:

Each ST7cc server detects the non-availability of its WinCC partner. The incoming ST7 messages are stored in the respective local buffers. For the ST7 stations, the target nodes (ST7cc servers A, B) can be reached.

Renewed availability of the failed components:

Normally, both WinCC systems will not become available at the same time. The first available WinCC is updated by its ST7cc server based on the local buffer. As soon as the redundant partner is available, the WinCC redundancy functions synchronize the archive data that has gaps compared with the real process events. These gaps are closed by ST7cc.

Since during the non-availability of the WinCC servers one of the ST7cc servers was also unavailable, the data gaps resulting in the affected ST7cc server are filled from the remote buffer of the partner.

System status: (1,0,0,1), analogously also (0,1,1,0)

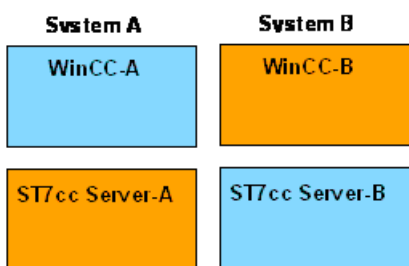


Figure 5-12 System status (1,0,0,1) / (0,1,1,0)

In this case one WinCC system and one ST7cc server is available and one of each failed (but crossed over):

WinCC level:

The available WinCC system recognizes the non-availability of its WinCC and its ST7cc

partner. Since no communication is possible with the technological process, the variables affected are indicated as such in the display (grayed out).

ST7cc level:

Each ST7cc server detects the non-availability of its WinCC and ST7cc partner. The incoming ST7 messages are stored in the respective local buffers. For the ST7 stations, the target nodes (ST7cc servers A, B) can be reached.

Renewed availability of the failed components:

If WinCC becomes available again, the WinCC archive data is updated by the redundancy functions for the period of the failure. The data supplied to the available WinCC system staggered over time prior to the failure are not included. This data update is an ST7cc redundancy function.

If ST7cc becomes available again, the incoming ST7 messages are received and redirected to the local buffer until the WinCC data gap is closed through the interaction with the remote buffer of the ST7cc partner. Following this, the local buffer is processed and normal operation resumed.

System status (0,0,0,0)

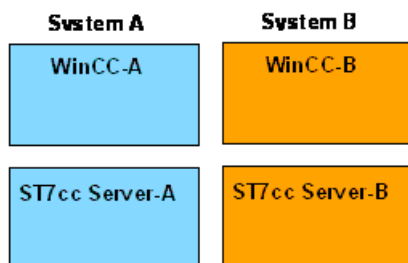
System A and B not available.

For the ST7 stations, the situation is the same as the non-availability of both ST7cc servers.

5.3.3 ST7cc functions to ensure data consistency

This section illustrates the necessity of the ST7cc redundancy function. Only with ST7cc support can the data consistency of the WinCC data be guaranteed even with data received staggered over time.

System status (1,1,0,0)



With the non-availability of the hardware unit B, the WinCC and ST7cc server B components are also unavailable.

For WinCC redundancy, the downtime is the time when the redundant partner was last able to detect the availability of its partner. When establishing data consistency when a failed partner becomes available again, WinCC updates only the data missing since the *time of the failure*.

WinCC gap

If, during the failure of the hardware unit B, SINAUT sends data staggered over time with a time stamp (t_1 in the figure) before the failure time t_2 , this is included correctly in the data management of WinCC server A, but no longer in the data management of WinCC server B. This gap is closed by ST7cc.

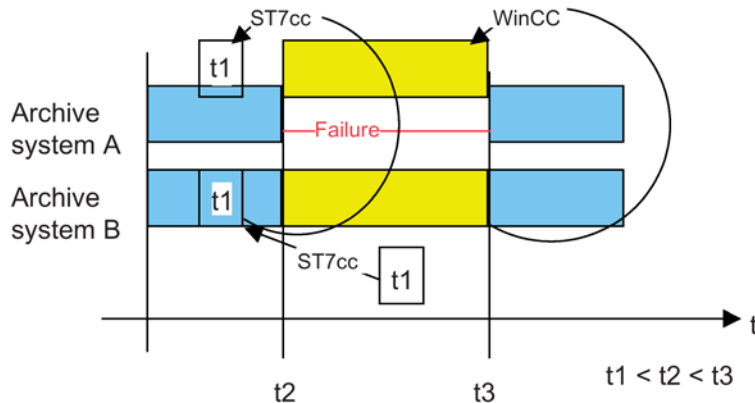


Figure 5-13 Guarantee of data consistency

- Situation up to time t_2 :
 Up to time t_2 , both hardware components (HW A and HW B) can be reached by SINAUT. ST7cc servers A and B can also reach the WinCC servers A and B. Both WinCC data managements and archives are supplied correctly at the same time.
- Situation from time t_2 to time t_3 :
 As of time t_2 , only ST7cc server A can be reached by SINAUT. Only the WinCC data management of WinCC server A can be reached. The non-availability of hardware B lasts until time t_3 .
 While hardware B is not available, frames with the time stamps t_1 and $<t_2$ are sent staggered over time by SINAUT. This data is entered chronologically correctly in WinCC archive A.
- Situation as of time t_3 :
 The ST7cc and WinCC server B are available again. WinCC redundancy restores data consistency (from the point of view of WinCC) by updating the WinCC archive data from time t_2 onwards.
 Following this, ST7cc restores full data consistency by updating the messages collected in the remote buffer during the time of the failure.

Archive A and B:

The archives contain the data (Tag Logging) of WinCC servers A and B. WinCC Tag Logging operates computer time-oriented; in other words, WinCC archive data (Tag Logging) is entered in the WinCC Tag Logging with a time stamp indicating the time when it was processed in WinCC. The time information of the SINAUT time stamp is ignored by WinCC in this case. This method is not compatible with SINAUT functionality, which allows ST7 messages to be supplied staggered over time. ST7cc supplies the data to be archived to Tag Logging with the ODK functions. WinCC has no organizational knowledge of the data whatsoever that are supplied staggered over time during the period of a failure.

WinCC redundancy package:

The WinCC redundancy package (WinCC Redundancy) is responsible for restoring WinCC data consistency of both WinCC servers following the return of a previously failed partner.

ST7cc redundancy package:

The ST7cc redundancy package (ST7cc Redundancy) as a WinCC add-on is also responsible for restoring the WinCC data consistency of SINAUT data supplied staggered over time during the period of failure.

5.4 Quality code of the WinCC tags supplied with values by ST7cc

WinCC tag management

The tag management of the WinCC project can be displayed with the WinCC Explorer (see figure). By selecting a tag with the mouse pointer, the current process value, its WinCC quality code and the time of the last change to the tag is displayed.

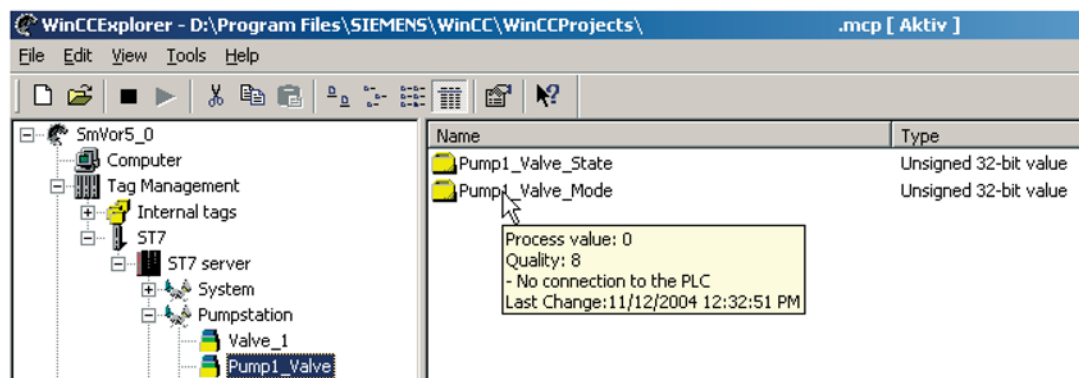


Figure 5-14 WinCC tag management with quality code information for selected tags

The status information transferred to WinCC by the ST7cc server causes the following three WinCC quality information displays:

1. Quality 80: OK
2. Quality 18: Connection to partner not established.
3. Quality 4c: Initialization value of the tag.

The figure shows which input variables are logically included in the quality.

Forming the WinCC quality code based on the ST7cc system monitoring

The TCO component monitors the local TIMs connected over MPI and the TIMS connected over Ethernet (local stations). The TCO maps the most important status information to ST7cc variables.

WinCC quality 4c (ST7cc server startup):

This is the case when the ST7cc server starts up or restarts and does not yet have a connection to the stations and therefore to the SINAUT objects. In this case, the ST7cc variables are given a status from which WinCC derives the quality 4c. If a variable already exists on the ST7cc server but the corresponding SINAUT object is not yet configured, this state is not exited.

WinCC quality 18 (station no longer accessible):

In this case, there were already valid value is on the ST7cc server following successful communication. If a problem occurs on the line afterwards, the last valid acquired value (value, time stamp) should not be changed, the operator must, however, be informed that there is temporarily no communication between the variable and the origin of its value. In this case, the ST7cc variables are given a status from which WinCC derives the quality 18.

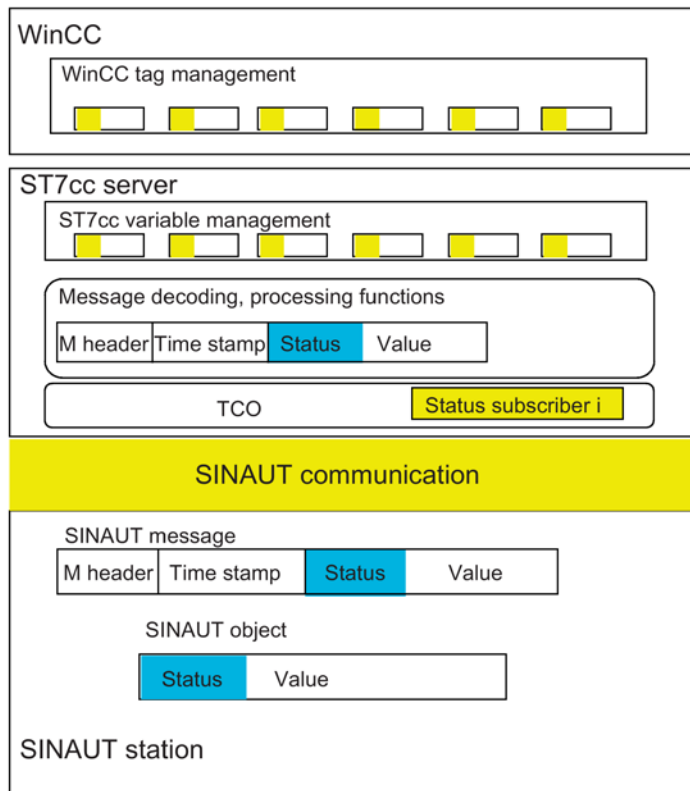


Figure 5-15 Forming WinCC quality for ST7cc variables

WinCC quality code formation based on status information

The SINAUT object types ST1 ATZ01 and ST7 Cnt01D / Cnt04D, ST1 ZTZ01 contain a status and value information in their object data areas. With ST7 object type Ana04W, conclusions can be made about its status based on the value due to the defined range limits. You will find a more detailed description of the status information in the SINAUT ST7 System Manual.

This status information integrated in the object data area is shown as the status in the figure. To support the user when decoding these object data areas, the type and subtype options M3, Z1 and Z2 are available in ST7cc Config to define the variables. The evaluation of the status information during message decoding makes use of the type and subtype parameters of the variable definition. The status information of object types ST1 ATZ01 and ST7 Cnt01D / Cnt04D, ST1 ZTZ01 is used primarily in the ST7cc processing of the variables and has only a limited influence on the WinCC quality code.

Selecting variable types and subtypes for the decoding

Type	Sub type	Permitted lengths	Explanation
M	1	16, 32 bits	16, 32 bits are interpreted as unsigned integers. WinCC data type: floating point 64-bit IEEE 754
	2	16, 32 bits	16, 32 bits are interpreted as signed integers. WinCC data type: floating point 64-bit IEEE 754
	3	16 bits	The 16 bits are interpreted as ST1 measured value. WinCC data type: floating point 64-bit IEEE 754
	4	32 bits	32 bits as floating-point number. WinCC data type: floating point 64-bit IEEE 754
S	1	1 - 32 bits	Data areas of 1 to a maximum of 32 bits can be defined as variables. Case 1: length = 1 bit -> WinCC data type: Binary tag Case 2: length = 2 to 32 bits -> WinCC data type: unsigned 32-bit value.
C	1	32 bits	32 bits represent an ST7 absolute counted value (28-bit value, 4-bit status). WinCC data type: floating point 64-bit IEEE 754
	2	32 bits	32 bits represent an ST7 difference counted value (28-bit value, 4-bit status). WinCC data type: floating point 64-bit IEEE 754
	3	32 bits	32 bits represent an absolute counted value (32-bit value, no status). WinCC data type: floating point 64-bit IEEE 754
	4	32 bits	32 bits represent a difference counted value (32-bit value, no status). WinCC data type: floating point 64-bit IEEE 754

Quality code resulting from status codes of the ST1 / ST7 objects

Type M3:

If the *overflow bit* or the *wire-break bit* of the ST1 data format is set, this is detected on the ST7cc server and leads to quality 18 in WinCC.

Type M2:

If users know that the data source of the monitored value is of the SINAUT object type Ana04W, they can filter out status information by comparing the process value with various range limits.

Type Z1, Z2:

Status codes of the absolute or differential counter:

The A bit (up-to-date bit) of the SINAUT object value specifies whether the value is up-to-date; in other words, valid or whether the value is an "initial value". The significance of the bits is included only in the counted value processing of the variables and has no influence on the WinCC quality.

5.5 Adopting the configuration data

Adopting the configuration data describes when and how the configuration data of the user is adopted by the ST7cc server.

5.5.1 Adopting the configuration data on a single system

The procedure starts with the project engineering / configuration with ST7cc Config. Due to the integration in WinCC of the ST7cc configuration, parameter data for ST7cc and WinCC usually result. A configuration is rarely restricted only to the functional range of the ST7cc server.

The plausibility checks of ST7cc Config do not allow a "half-finished" configuration. Whether or not a configuration is complete in terms of its logic / task, cannot be checked by the configuration tool.

ST7cc Config:

It is possible to configure / reconfigure in ST7cc Config during operation of the ST7cc and/or WinCC server. Since the ST7cc configuration extends to WinCC as already mentioned, to ensure the consistency of the parameter data that involves more than one component, the ST7cc and WinCC servers must be deactivated before the WinCC data (tag management, messages, archive tags) are generated.

ST7cc server:

When it restarts, the ST7cc server adopts the ST7cc configuration data. The ST7cc server runs its processing functions regardless of whether the WinCC data is currently being generated or not. From this point of view, assurance of parameter data consistency with WinCC remains the responsibility of the configuration engineer.

Following a restart, the server checks the availability of the ST7 stations and activates a general request for the available stations to update its process image.

WinCC Runtime:

During a restart, the WinCC Runtime system adopts the current parameter information. Since ST7cc monitors the availability of WinCC and stores the ST7 messages if it is not available, no data intended for WinCC can be lost even if WinCC is not available.

5.5.2 Adopting the configuration data on a redundant system

Knowledge of the steps in reconfiguration of ST7cc with an ST7 single system is assumed (see section Adopting the configuration data on a single system (Page 288)).

The two packages WinCC Redundancy and ST7cc Redundancy ensure that no data is lost if one of the redundancy partners is not available.

This system property is made use of when changing the parameter settings of a redundant ST7cc system.

Note**Conventions:**

The redundant partners are simply known as system *A* and system *B*.

Step 1:

After including all parameter changes, a redundancy partner, for example system A (WinCC A and ST7cc server A) is shut down. The WinCC data management is then generated (tag management, archive tags, message management). After the system has been generated, *system A* can be restarted. When the ST7cc server is restarted successfully, redundancy is restored since the ST7cc redundancy mechanisms avoid the loss of process data.

Step 2:

Step 2 is possible only if step 1 was completed successfully. If this is the case, redundancy partner B (WinCC B and ST7cc server B) is deactivated.

Step 3:

Step 3 is possible only if steps 1 and 2 work completed successfully and WinCC system A has started up correctly. If this is the case, the WinCC DUPLICATOR copies the parameter data to *system B*. After successful duplication of the data, *system B* can be restarted. When *system B* has started up successfully, redundancy is fully restored.

5.6 Startup behavior and start order

During startup, the ST7cc server performs the following steps:

- Evaluation of the configuration settings
- If applicable, waiting for a default time until Windows has started up completely
- If applicable, starting WinCC

- Dynamic linking of the active libraries (channel DLL, ODK, SAPI-S7)
- Reading in the process image
- Deleting all old WinCC variable registrations in the process image
- Enabling channel DLL for new WinCC variable registrations
- Waiting for successful registration with WinCC Tag Logging and Alarm Logging
- Enabling communication
- Sending lifebeat messages to the locally connected TIMs
- General request of the locally connected TIMs on confirmation of the lifebeat message
- Detection of the subscribers reachable over the locally connected TIMs
- General request of the reachable subscribers
- After timeout, fault message for the subscribers for which no TIM has declared itself to be responsible

Note

Correct functioning of the channel DLL can only be guaranteed when WinCC is started by the server.

5.7 Exiting ST7cc server and WinCC

During shutdown, the ST7cc server performs the following steps:

1. Terminating communication and deleting network management structures
2. Notifying WinCC that the server is no longer active
3. Enabling the dynamically linked program libraries
4. Closing the logging system
5. Saving the process image

If the ST7cc server window is not open, the file <st7cccshutdown.bat> is available for exiting ST7cc in the folder "...\\Siemens\\ST7cc\\bin".

5.8 Restarting WinCC with the ST7cc server active

It is possible to shut down and restart WinCC while the ST7cc server is operating.

5.9 ST7cc server status

ST7cc server status

In the ST7cc server, it is possible to display a status. Select the menu sequence *SINAUT* > *ST7cc Server Status* in the server window. This status display shows all the important information on the local and remote servers (see figure). All status displays relevant only for redundant operation are hidden in single operation.

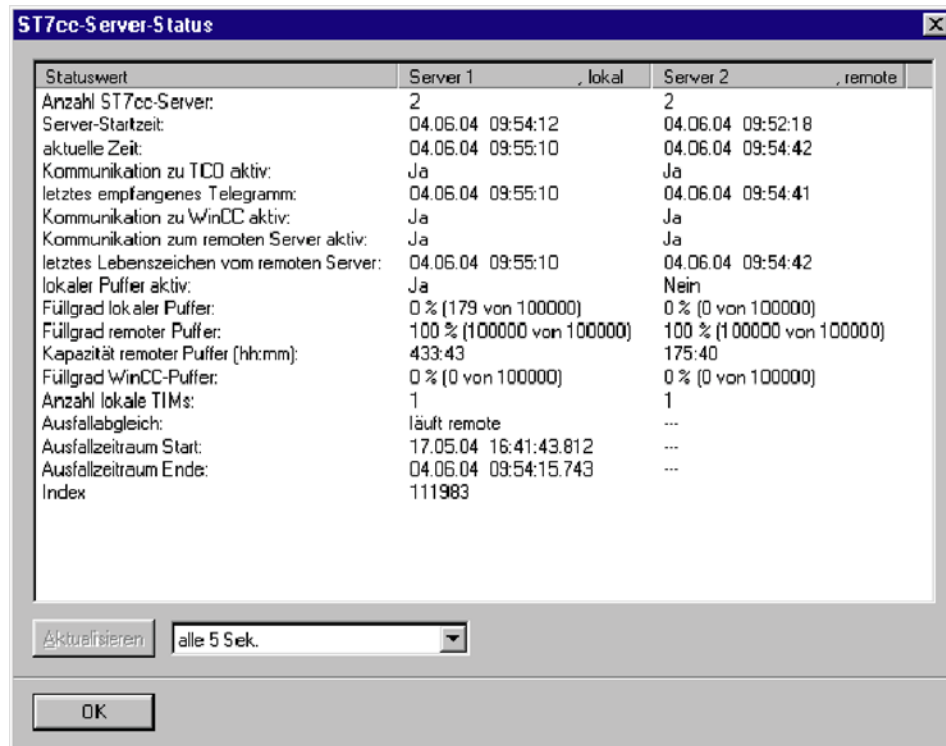


Figure 5-16 Display of the ST7cc server status

Status attribute	Explanation
Number of ST7cc servers	Number of configured <i>ST7cc servers</i> : 1: Single system 2: Redundant system
Server start time	Time when the <i>ST7cc server</i> program started
Current time:	Current time on the PC
TCO communication active:	Possible display: YES/NO Display indicating whether or not the <i>tco server</i> program has started. This program precedes the <i>ST7cc server</i> and handles communication between the <i>central TIM</i> and the <i>ST7cc server</i> program.
Last received message	Time stamp of the message last received from the control center TIM.

5.9 ST7cc server status

Status attribute	Explanation
WinCC communication active:	Possible display: YES/NO Displays whether WinCC Runtime is active.
Remote server communication active:	Possible display: YES/NO Displays whether the redundant partner PC can be reached and whether the <i>ST7cc server</i> has started on this computer.
Last life beat from remote server:	Time stamp of the last life beat message sent from the remote partner PC.
Local buffer active:	Possible display: YES/NO If WinCC runtime is not active, all messages are stored temporarily in the local buffer. Buffer is activated only when WinCC runtime is deactivated.
Fill level of local buffer:	Default: 0 % (0 of 100000) Display of the number of messages stored in the local buffer as a percentage and absolute number.
Fill level of remote buffer:	Default: 100 % (100000 of 100000) Display of the number of transferred messages stored in the remote buffer as a percentage and absolute number. All the messages from the control center TIM are stored in this buffer in case they are required for redundancy synchronization between the two server PCs. After the first fill phase, this buffer is always filled to 100%.
Capacity of remote buffer (hh:mm):	Elapsed time in hours and minutes since the oldest message in the remote buffer. This indicates the period of a failure that can be covered by the remote buffer.
Fill level of WinCC buffer:	Default: 0 % (0 of 100000) Display of the number of messages stored in the WinCC buffer as a percentage and absolute number. If messages are transferred by the control center TIM faster than they can be accepted by WinCC runtime, they are stored temporarily in the WinCC buffer.
Number of local TIMs:	Number of configured <i>TIMs</i> on the MPI bus or on Ethernet.
Update after downtime:	Possible displays: <ul style="list-style-type: none"> • Starting • Running remote • Pending • Preparing • Running local • Unknown Instantaneous status display if there is a failure synchronization following an <i>ST7cc</i> runtime restart.
Downtime start:	Time at which the partner downtime started. This is displayed only when the <i>ST7cc</i> runtime of the failed computer restarts.
Downtime end:	Time at which the remote partner restarted. This is displayed only when the <i>ST7cc</i> runtime of the failed computer restarts.
Index:	Displays the current data record number to be synchronized in the remote buffer.

5.10 Standard general request and accelerated general request

General request (GR)

If a station connected to ST7cc returns following a disruption, ST7cc sends a general request (GR) to the station to update the current process status of the station. If ST7cc itself fails and starts up again, it sends a general request to all connected stations to bring its process data up to date.

Apart from these GRs triggered automatically by ST7cc due to disturbances, the operator can also send a GR to one station when necessary. The operator can trigger sending of the GR in the faceplate.

In response to a standard GR, the queried station enters all its data messages along with the current process values in the send buffer of the station TIM. The data messages are preceded by a message indicating "GR start" and the requested messages are followed by a message indicating "GR end". If there are still messages stored in the buffer of the station TIM that are waiting for transmission, the requested messages and the "GR end" message are entered after the buffered messages. Only the "GR start" message is entered at the start of the send buffer.

The messages are then transmitted in the order in which they occur, in other words, first the "GR start" message shown in the picture typical of the queried station by the GR LED flashing yellow. Any stored messages are then transmitted followed by the requested messages with the current process image and finally the "GR end" message. With the "GR end" message, the GR LED is permanently lit green in the picture typical. If the operator triggered a GR manually, and the GR LED is lit green, the operator knows that all archives and the process image of the queried station are up to date.

The time required for a standard GR depends on the performance of the transmission network, the number of messages triggered by the GR and the number of messages still stored in the buffer of the station TIM. If the connection to a station is down but the station is still functioning correctly, the number of messages stored in the send buffer of the TIM while the connection is down depends on the process characteristics and the duration of the down time (depending on the TIM type, there is capacity for between 10,000 and 95,000 messages). When the connection is re-established and ST7cc has automatically sent a GR, it may take some time before the archive and then the process image are once again up to date.

Stations connected over dial-up networks generally have longer connection pauses because the connections are established as seldom as possible to save costs. When there is a GR, it must always be assumed that there are buffered messages to be dealt with and that some time will be needed before there is a response to the general request.

The advantage of the standard GR based on the first-in/first-out principle is that the data is transferred in the correct chronological order. This allows normal process control systems to process the data without any problems; in other words, they are capable of supplying their process image and their archives correctly with data.

Accelerated general request

If the user requires the current process image more quickly, an "accelerated GR" can be used instead of the standard GR. This is possible with ST7cc as if V2.6 in conjunction with the Ethernet TIMs (TIM 3V-IE types, TIM 4R-IE).

Note

An accelerated general request can only be responded to by stations on which the SINAUT program runs on the TIM (program configured with TD7onTIM). SINAUT programs running on the CPU (created with TD7onCPU) cannot respond to the accelerated GR. They react to such requests as they would react to a standard general request.

In an accelerated general request, the messages with the requested process image and the "GR start" and "GR end" messages are entered at the start of the send buffer of the stations TIM; in other words, before any messages still buffered on the TIM. The requested messages and the current process image are therefore transmitted first. These are then followed by any buffered messages that are entered in ST7cc in the archives (and not in the process image) if they are older than the process image transferred with the GR.

In ST7cc, it is possible to make a setting so that the accelerated GR is used automatically instead of the standard GR following disruptions (specified in Project Settings in the Server tab, see section Project settings: Server (Page 118)). This makes the current process image available more quickly following a disruption.

Although manual triggering of a GR by the operator is possible for stations connected over dedicated line or wireless, it does not generally achieve a lot because the ST7cc control center is normally kept permanently up to date with these network types.

If, on the other hand, stations are connected over a dial-up network, it makes sense for the operator to first check the current status of the station before entering a command or changing a setpoint. In this situation, the accelerated general request has advantages. The operator can now establish a permanent connection with the station using the station faceplate and fetch the current process image immediately by starting an accelerated GR (also initiated in the station faceplate) without having to wait for the transmission of the locally buffered messages. It is then possible to enter a command or setpoint more quickly and by repeating the accelerated GR at any time afterwards, the operator can check the reaction of the process if the buffered messages are still being transmitted and are delaying the transmission of the current process changes.

Processing functions in ST7cc and interaction with WinCC

Various processing functions (message processing, archive processing etc.) can be configured for ST7cc variables (see section Configuring processing functions (Page 231)). The creation, acknowledgment and archiving of messages as well as the archiving of process values takes place in WinCC.

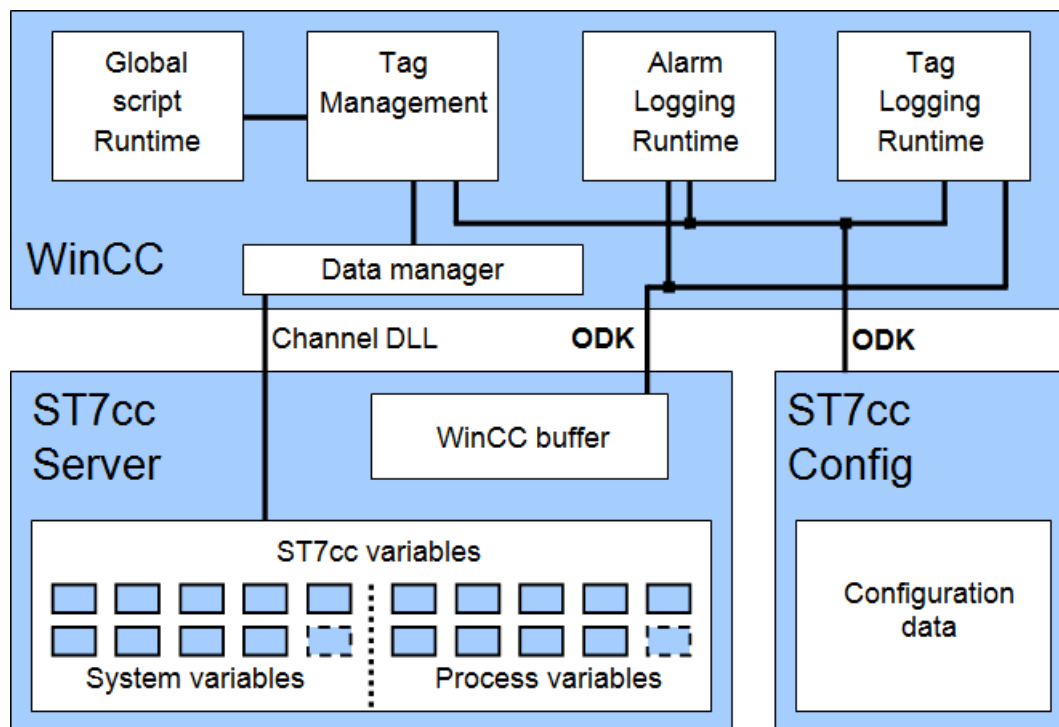


Figure 5-17 Interaction between ST7cc server status and WinCC

When an accelerated GR is triggered, ST7cc reacts as follows:

Case1:

A value from a message obtained through the accelerated GR is transferred to the ST7cc server.

ST7cc recognizes that this is a value from the process image obtained with an accelerated GR and forwards this to the WinCC data manager. Processing functions are not performed in ST7cc because this could lead to incorrect results due to ignoring older values still buffered on the station. Since WinCC, on the other hand, receives a correct value, all the processing functions configured for the WinCC tag are performed in WinCC, for example, generating messages.

Case 2

After the transmission of the accelerated messages, a historical value (message from the send buffer of the station TIM) is supplied to ST7cc.

5.10 Standard general request and accelerated general request

ST7cc recognizes that this is a historical value (older than the current value in the process image) and no longer transfers this to the WinCC data manager. ST7cc runs the following processing functions for the historical value:

- All ST7cc processing functions.
On the ST7cc server, the measured value, counted value and archive processing functions are performed and the results forwarded to WinCC.
- The message for the historical values is created on the ST7cc server and the message entered in the appropriate WinCC archive (short- or long-term archive). If this is a message that must be acknowledged, the message is acknowledged automatically by WinCC if the mandatory message state has been exited based on the current image information that is already available to WinCC. This automatic acknowledgment is displayed to the operator as "acknowledged by system".

Diagnostics and trace options

6.1 Diagnostics: Log server messages

The log messages are displayed in a separate program, the *SINAUT LOG Server*. This means that the LOG messages remain visible even when the ST7cc server is started in the background.

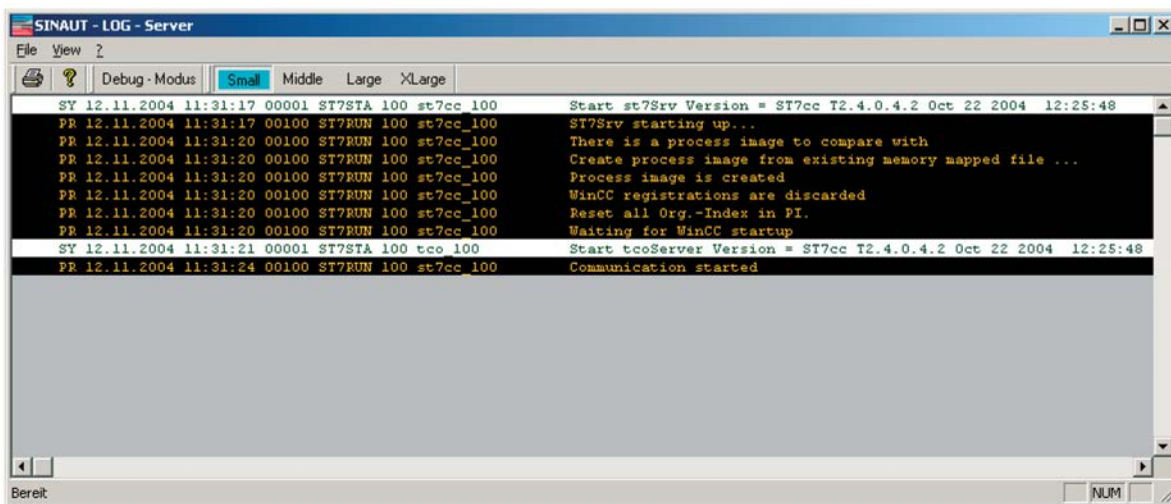


Figure 6-1 Display of messages in the SINAUT LOG server

The log messages are used to display events during server startup and for diagnostics of connection statuses in normal operation. Typical error situations can be recognized easily and remedied based on the log messages.

In normal server operation, no popup messages are created unless it is attempted to send commands to an unobtainable subscriber.

Fatal errors when the server starts up (lack of program libraries, incomplete installation) are generated as popup messages and the server shuts down again after the message is acknowledged.

6.1.1 Messages relating to the process image

The process image is kept in a memory image file. The following messages relating to the process image can be generated as follows:

Sharing violation

The memory image file is blocked by another program and could not be opened.

Create process image from existing memory image file

During startup, the server has found an existing valid memory image file and will use this for the process image.

At least one client is still attached to the renamed old shared file --> sending a detach request now !!!

A different program was connected to the existing memory image file when the server started. The server has set up a new process image due to a change and the program must register for the currently valid memory image file.

No process image to compare with

No memory image file has been created for the process image or the existing file was deleted by the user.

No object list available

No process image can be created because there is no object list. Please check the server settings for the object list.

No typical list available

No process image can be created because there is no typical list (library). Please check the server settings for the library.

Copy present process image as reference image

A memory image file was found that is no longer valid. The server sets up a new process image but uses the values from the old memory file whose addressing and processing has not changed.

Object list is newer than the process image

The object list is newer than the last saved process image. The process image must be set up again based on the object list.

Configuration change detected - Server will be restarted in 5 seconds

When checking the object list, the server detected configuration changes. To set up a new process image, it will run a restart.

Process image is created

The process image was set up again successfully.

Create completely new process image ...

The process image is recreated and filled with zeros as initial values.

There is a process image to compare with

An existing process image was found that will now be checked for validity.

Server is newer than process image

The process image found is invalid because a new program version has been loaded. A new process image must be set up based on the object list.

Typical list is newer than the process image

The object list is newer than the last saved process image. The process image must be set up again based on the object list.

WinCC registrations are discarded

WinCC tag registrations that still existed in the process image were deleted after the server restarted.

WinCC registrations will be kept

WinCC tag registrations that still exist in the process image are adopted after the server is restarted.

WinCC registrations will be discarded now

WinCC tag registrations that still existed in the process image will be deleted after the server restarts.

6.1.2 Error messages on communication

Communication error messages are enclosed in the double hash characters.

Wrong time stamp for message %d/%d (%s)

A subscriber has sent a message with an implausible time stamp (too far in future or in the past). You can specify the limits for plausibility checks in the server settings.

Partner %d: not present in process image

The server has received a message from a subscriber that does not exist in the process image. The configuration should be updated.

Subscriber %d: GR End Timeout

The ready message for the general request to a subscriber did not arrive in time; in other words, the general request took too long. The time for the timeout can be set in the service settings.

Subscriber %d: GR Start Timeout

The start message for the general request to a subscriber did not arrive in time after sending the request. The time for the timeout can be set in the service settings.

Subscriber %d: not available

A subscriber on the WAN cannot be reached because it has been turned off or all the connections are down.

TIM %d: GR End Timeout

The ready message for the general request to a local TIM subscriber did not arrive in time; in other words, the general request took too long. The time for the timeout can be set in the service settings.

TIM %d: GR Start Timeout

The start message for the general request to a subscriber did not arrive in time after sending the request. The time for the timeout can be set in the service settings.

TIM %d: not available

A TIM subscriber on the MPI bus or on Ethernet can no longer be reached, for example, because it has been turned off.

Time stamp marked invalid for message %d/%d

Following a restart, a subscriber could not yet synchronize and its data cannot be accepted although the connection was correctly established and data transferred.

dll_S7_get_brcv_ind: Error

This is a fatal internal error. Please contact support.

FiFo backlog

An internal queue has overflowed. This might be an overload problem. Please contact support.

6.1.3 Status messages on communication

No connection to subscriber %d

A subscriber cannot currently be reached.

Communication started

Communication was activated after a server restart. This normally happens following successful startup of WinCC Runtime.

Subscriber %d: available via TIM %d

A WAN subscriber was signaled as being available by a local TIM.

Subscriber %d: GR end

A WAN subscriber has signaled the end of a general request.

Subscriber %d: GR start

A WAN subscriber has signaled the start of a general request.

Subscriber %d: General request sent

A WAN subscriber was requested to transfer its data within the framework of a general request.

TIM %d (MPI %d): Transport connection established

The configured S7 connection to a local TIM subscriber was established.

TIM %d (MPI %d): Transport connection not established

The configured S7 connection to a local TIM subscriber could not be established.

TIM %d (MPI %d): Link closed

The configured S7 connection to a local TIM subscriber was terminated.

TIM %d (MPI %d): Link busy

Send jobs are being processed on the connection to the local TIM subscriber.

TIM %d (MPI %d): Link valid

The ST7 connection to a local TIM subscriber is valid (in other words, the transport connection is established and the TIM subscriber is answering the libebeat messages).

TIM %d: available

A local TIM subscriber is signaling as being available.

TIM %d: GR end

A local TIM subscriber has signaled the end of a general request.

TIM %d: GR start

A local TIM subscriber has signaled the start of a general request.

TIM %d: General request sent

A local TIM subscriber was requested to transfer its subscriber records.

Received unhandled S7 message

An S7 message with a type not expected in ST7cc was received. It is ignored.

Waiting for WinCC startup

ST7cc is waiting for the start of WinCC runtime to start communication.

6.1.4 Messages on time-of-day synchronization

Time sync message missing!

The time-of-day synchronization was not received in the expected minute cycle.

Time synchronization active

A valid time synchronization message was received. The server automatically starts time-of-day synchronization. The system time of the WinCC PC is adjusted by up to 5 seconds per minute by going slower or faster to avoid a time jump.

Time for synchronization differs by %d seconds from current time! Please exit WinCC and correct the computer clock !!

The difference between the received time message and the system time is too great to make a gradual adjustment with the synchronization function practicable. There is also the risk that messages are rejected due to an implausible time stamp. The time of the computer should be adjusted manually.

The maximum time difference for time synchronization corresponds to the parameter *Maximum time deviation in the future* in the server settings.

6.1.5 Diagnostic messages on parameter assignment errors

The following errors can be found by evaluating the results of a general request:

1. Objects that exist in the general request and are transferred but that were not configured in ST7cc.
2. Variables configured in ST7cc but that are not transferred by the general request (either because the object does not exist or does not include the configured data area).

This allows you to check the consistency of the configuration in the general request and in ST7cc in both directions.

Subscriber %d: GR complete

All the values configured for this subscriber were transferred within the framework of the general request.

Subscriber %d: GR not complete

Not all the values configured for this subscriber were transferred within the framework of the general request. This is followed by a list of objects / variables not transferred.

Object %d Variable %d (%s.%s) not transferred during GR

Message, which objects / data areas are not in the GR.

Partner %d, Object %d: not present in process image

Data was received from the specified subscriber that could not be assigned to any configured object. Set up the relevant object for this subscriber. If you have set up the object and this error nevertheless appears, check whether the ST7 server is working with the correct project file (object list).

6.1.6 Messages on WinCC Tag Logging / Alarm Logging

**The job queue for archiving is full.
New archive requests will be rejected.**

This message appears as a popup. Either the ST7cc server is not currently processing archiving requests, for example, because runtime is not started or too many requests were sent in a short time.

If necessary, increase the length of the archive buffer.

Tag Logging Archive %s Variable %s: Archiving failed with error code %d (%s)

Tag Logging could not execute the archive job for the relevant tag.

The relevant archive variable was probably not generated.

Remedy the situation.

Message number format code: s = %d o = %d i = %d v = %d m = %d

Part of the message number (s = subscriber number, o = object number, i = typical instance number, v = variable number, m = consecutive message number) does not match your message number system. To avoid an incorrect message being generated, the message was not displayed. Check that the object keeps to the number system.

Alarm Logging: Message %d could not be created: Error code %d (%s)

Alarm Logging could not create the message. The relevant message was probably not generated. Remedy the situation.

6.1.7 Messages on the PM-AQUA interface

**The queue for archiving with PM-AQUA is full.
New archive requests will be rejected.**

This message appears as a popup. Either PM-AQUA is currently not currently processing archiving requests, for example, because runtime is not started or too many requests were sent in a short time.

If necessary, increase the length of the archive buffer.

PM-AQUA connection: %d Wrong index: %d

An attempt was made to address an index that was not configured in PM-AQUA within a process connection.

PM-AQUA connection: %d Index: %d Time: %s already processed

PM-AQUA rejected an archive request because this request or a newer value is already being processed. This message can occur following a restart if the first value in the PM-AQUA queue was accepted but not acknowledged prior to shut down and was therefore transferred again.

PM-AQUA connection: %d Index: %d Time: %s invalid (e.g. in the future)

PM-AQUA rejected an archiving request because the time stamp was invalid.

PM-AQUA connection: %d Index: %d PM-AQUA indicated error %d for raw data interface

An unexpected error code was returned by PM-AQUA. Please refer to the error code in the PM-AQUA manual.

6.2 Diagnostics: Message protocol of the ST7cc server

The message protocol of the ST7cc server is used for diagnostics of the message traffic.

The ST7cc server and ST7cc Config both provide activation and deactivation allowing the creation of a message protocol. If the message protocol is activated by an operation over the ST7cc server, the startup behavior of the servers cannot be recorded. This is possible only when the message protocol is activated in the dialogs of ST7cc Config (see section Project settings: Message protocol (Page 146)). Regardless of where you activate the message protocol, it will run until you stop it with Stop trace or you exit ST7cc runtime. To create a message trace in online operation, follow the steps below:



Figure 6-2 Starting the message protocol in the ST7cc server

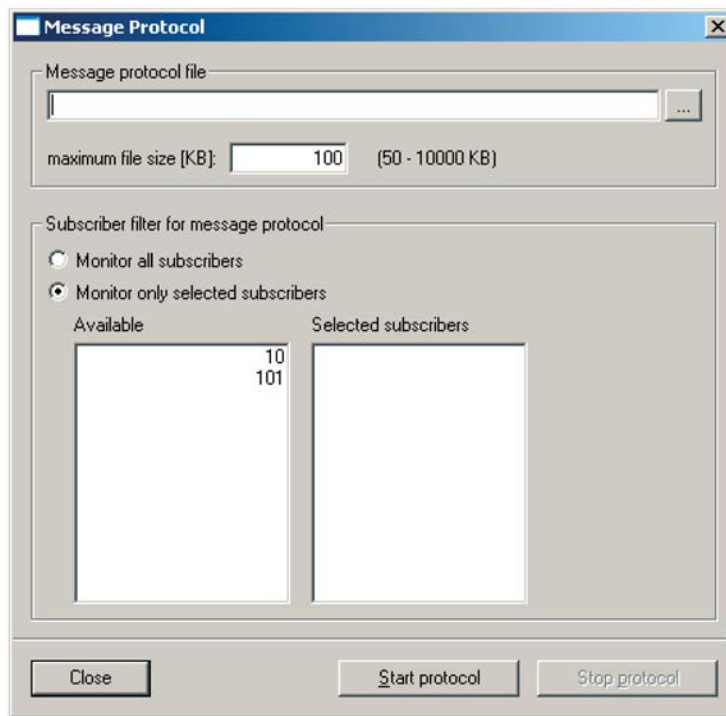


Figure 6-3 Message protocol dialog of the ST7cc server

1. Click on SINAUT > Message protocol in the window of the ST7cc server (see figure).
The Message Protocol dialog opens (see figure).
2. Click on the button (...) beside *Message protocol file*.
3. In the dialog that now opens, select the file and path (default: ...Siemens\Step7\S7Temp\) in which you want to save the message protocol (log) and click *Open*.
4. Under maximum file size, enter the maximum size of the protocol file. The optimum size is largely dependent on how many subscribers are being logged and the size the messages. As an average value, 35 bytes per message can be assumed. This results in a selectable number of messages to be logged of approximately 1,400 (50 KB) – 280,000 (10,000 KB).
If the file has reached the set value, it is saved as filename.old and a new file with the name specified above is created. When this new file reaches the maximum size, it in turn is saved as "Filename.old". The information in the first protocol (log) file is lost.
5. Select the subscribers to be logged. Select either the Monitor all subscribers option or the select the subscribers you want to monitor with the Monitor only selected subscribers option. With this option, the subscribers that will be monitored appear below Selected subscribers.
6. Click on Stop protocol.

The message protocol continues until you stop it with *Stop protocol* or exit ST7cc runtime.

For diagnostics purposes, the trace file can be opened with *SINAUT Diagnostics and Service*. For more detailed information, refer to the SINAUT ST7 System Manual.

6.3 Diagnostics: Subscriber typicals and faceplates

Subscriber typicals are used to visualize the most important communication statuses and allow manual control of the communication functions, such as triggering a general request or activating a permanent connection.

6.3.1 Picture typicals and faceplates for a station

Picture typical

The picture typical for a station (see figure) is used to display status information of a station.



Figure 6-4 Picture typical for a station

The significance of the individual LEDs is explained below.

LED name	LED display	Meaning
Subscriber	Red	Station problem
	Yellow	Some paths not OK
	Green	All paths OK
	Gray	Undefined status during startup phase
Connection	Flashing green	Requested
	Flashing yellow	Permanent connection
	Green	Online
	Gray	Offline
GR	Flashing yellow	GR start
	Green	GR end
	Red	GR incomplete
	Gray	Undefined status during startup phase
Time of day	Green	Standard / daylight-saving time (time valid)
	Red	Time invalid
	Yellow	Synchronization problem
	Gray	Undefined status during startup phase

Station faceplate

The faceplate contains text displays on the status information of the status and allows operator control of stations (see figure). To display the faceplate of the station, follow the steps below:

1. Left-click on the picture typical of the required station.

The figure shows a dialog box titled "Name" with a table of variables and their status. Below the table are three radio buttons for connection options and four buttons: OK, Apply, Details, and Cancel.

Name	
Subscriber	***
Connection	***
GR	***
Clock	***
Current data path	***

General request
 Permanent connection on
 Permanent connection off

Figure 6-5 Station faceplate

Variable name	Possible displays
Subscriber	disturbed
	Some paths not OK
	All paths OK
Connection	Requested
	Offline
	Online
	Permanent connection active
GR	GR requested
	GR start
	GR end
	GR start timeout
	GA end timeout
	GR incomplete
Time of day	Invalid time
	Standard time

Variable name	Possible displays
	Daylight saving time
	Synchronization disrupted on CPU
Current data path *)	Internal IF
	External IF
	Ethernet 1
	Ethernet 2

*) The variable names used here are standard names. The user can change these to names with more meaning, such as "Dedicated line S34-1", "Phone network", "Wireless network south", or similar.
 For more information on changing names, see Section Setting up a subscriber (Page 206).

What do GR start, GR end, GR not completed, GR start timeout, GR end timeout mean?

GR start means that the station reports to ST7cc that it has started to process a general request (GR). The text *GR start* is displayed in the faceplate.

GR end means that the station reports to ST7cc that it has completed processing a general request (GR). The text *GR end* is displayed in the faceplate.

After receiving the GR end message, the ST7cc server checks that nothing is missing. If the result of the checks shows the ST7cc server that the general request was correct; in other words was executed fully for all SINAUT objects, the text *GR end* remains displayed in the faceplate. If the result of the checks is negative, the text *GR not completed* is displayed. In this case, the text display *GR end* was only a temporary display.

GR start timeout means that following a general request by the ST7cc server, the station did not report that it started processing the general request within the monitoring time.

GR end timeout means that the station has reported starting to execute a general request with GR start, however the monitoring time within which a general request must be executed has elapsed.

To control a station over the faceplate, follow the steps below:

1. Select the required command (for example *Start general request*).
2. Click on the *Apply* or *OK* button.

Note

If you close the faceplate with *OK*, the selected command is executed at the same time. If you want to make sure that you do not send a command accidentally, close the faceplate with *Cancel*.

Command	Meaning
Start general request	Start of a standard general request to the selected station
Start accelerated general request	Start of an accelerated general request to the selected station
Connection off	In dial-up networks, a connection currently established to the station is forced to terminate even when there is still data to transfer.

Command	Meaning
Permanent connection on	A connection is established to a dial-up station and maintained until it is terminated again by the permanent connection off command.
Permanent connection off	See Permanent connection on.

Displaying further station details

1. To display further station details, click on *Details* in the station faceplate.

The screenshot shows a dialog box titled "Station25" with the following details:

Subscriber	all paths o.k.
Connection	online
GR	GR end
Clock	daylight saving time
Current data path	internal WAN IF

Below the table are five radio button options:

- General request
- Xcelerated general request
- Connection off
- Permanent connection on
- Permanent connection off

Under the heading "Details of connection:", there is another table:

internal WAN IF	polling in main cycle
external WAN IF	no connection
Ethernet	no connection
MPI-Bus	no connection

At the bottom of the dialog are three buttons: "OK", "Apply", and "Cancel".

Figure 6-6 Faceplate station details

The possible displays are the same for the two variables and they are therefore shown only once below.

Variable name	Possible displays
TIM A: internal WAN IF external WAN IF	No connection
	Outgoing call initialized
	Incoming call established
	Outgoing call established
	Permanent connection registered

Variable name	Possible displays
	Permanent connection registered and outgoing call initiated
	Connection established and permanent connection deregistered
	Permanent connection established
	Connection establishment attempt active in background
	Call number list blocked
	No driver-specific connection status available
	Call in main cycle
	Call in sub cycle
	Permanent call in main cycle
	Permanent call in sub cycle
Ethernet 1 *)	No connection
	Ethernet
Ethernet 2 *)	No connection
	Ethernet
MPI bus *)	No connection
	MPI bus

*) The variable names used here are standard names. The user can change these to names with more meaning, such as "Dedicated line S34-1", "Phone network", "Wireless network south", or similar.

For more information on changing names, see Section Setting up a subscriber (Page 206).

Faceplate for station statistics

To display the faceplate for the statistics of the station, right-click on the picture typical of the required station.

Three statistical values can be read from the faceplate (see figure).

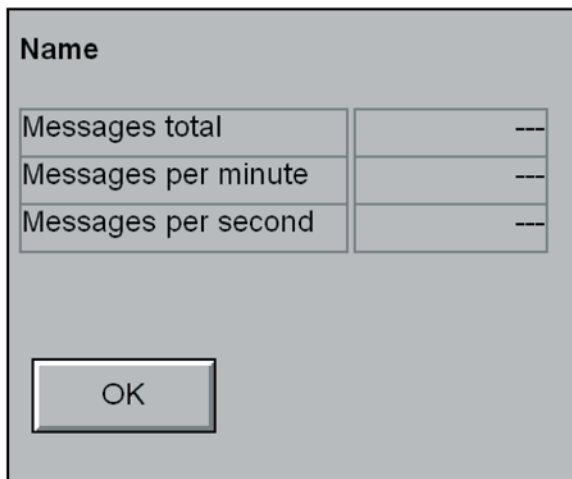


Figure 6-7 Faceplate for station statistics

The number of messages that ST7cc has received from the relevant station or the relevant TIM is displayed. Messages sent by ST7cc to a station or TIM are not included in these statistics.

Variable name	Value
Messages total	<i>Messages total</i> shows the total number of messages that ST7cc has received from the relevant station/TIM since the start of ST7cc runtime.
Messages per minute	<i>Messages per minute</i> shows how many messages ST7cc has received from the relevant station/TIM in the last minute.
Messages per second	<i>Messages per second</i> shows how many messages ST7cc has received from the relevant station/TIM in the last second.

The messages received by ST7cc from a TIM are only the messages created by this TIM itself; in other words, this involves only organizational messages, mainly organizational records.

Note on stations with TD7onTIM:

Messages created by the TIM for 'TD7onTIM' retain the subscriber number of the station CPU as the source address and are then assigned to this CPU by ST7cc (in terms of message statistics).

6.3.2 Picture typicals and faceplates for a local TIM

Picture typical for a TIM

The picture typical for a local TIM (see figure) is used to display status information of a local TIM.

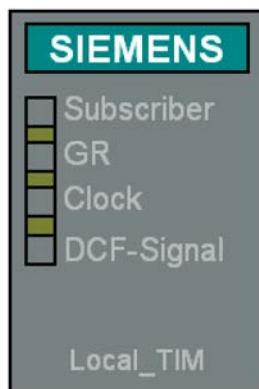


Figure 6-8 Picture typical for a TIM

The significance of the individual LEDs is explained below.

LED name	LED display	Meaning
Subscriber	Red	Local TIM disrupted
	Green	Connection to local TIM OK
	Gray	Undefined status during startup phase
GR	Flashing yellow	GR start
	Green	GR end
	Red	GR incomplete
	Gray	Undefined status during startup phase
Time of day	Green	Standard / daylight-saving time (time valid)
	Red	Time disrupted
	Gray	Undefined status during startup phase
DCF signal	Green	Radio signal OK
	Gray	Radio clock not present
	Red	No radio signal

Faceplate for local TIM

The faceplate contains more details on the status information of the local TIM (see figure) and allows a general request to be sent to the TIM. To display the faceplate of the local TIM, left -click on the picture typical of the required TIM.

Name

Subscriber	***
GR	***
Clock	***
DCF-Signal	***

General request

OK

Apply

Cancel

Figure 6-9 Faceplate for local TIM

The possible displays in the faceplate are described below.

Variable name	Possible displays
Subscriber	disturbed
	All paths OK
GR	GR requested
	GR start
	GR end
	GR start timeout
	GA end timeout
	GR incomplete
Time of day	Invalid
	Standard time
	Daylight saving time
DCF signal	not found
	No radio signal
	Radio signal OK

To trigger a general request, follow the steps below:

1. Select *Start General Request*.
2. Click on the *Apply* or *OK* button.

Note

If you close the faceplate with *OK*, the selected command is executed at the same time. If you want to make sure that you do not send a command accidentally, close the faceplate with *Cancel*

Faceplate for TIM statistics

To display the faceplate for the statistics of the TIM, right-click on the picture typical of the required TIM.

Three statistical values can be read from the faceplate (see figure).

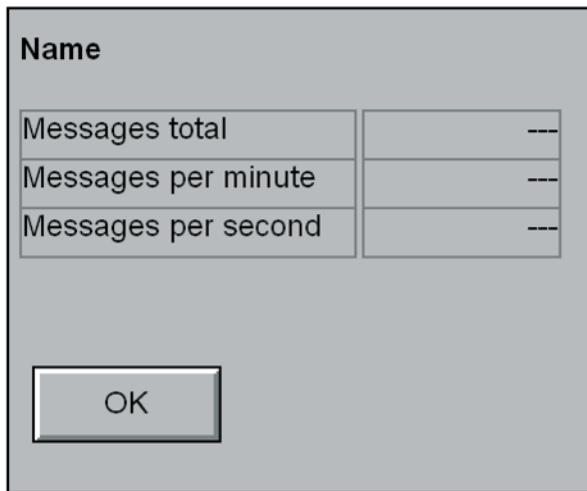


Figure 6-10 Faceplate for station statistics

The number of messages received by ST7cc from the relevant TIM is displayed. Messages sent by ST7cc to a TIM are not included in these statistics. For more detailed information, refer to the section Picture typicals and faceplates for a station (Page 306).

6.3.3 Picture typical and faceplate for a server

The server typical is used to visualize the most important server statuses and provides information on any downtime synchronization that may have taken place.

Server picture typical

The picture typical (see figure) is used to display server status information.

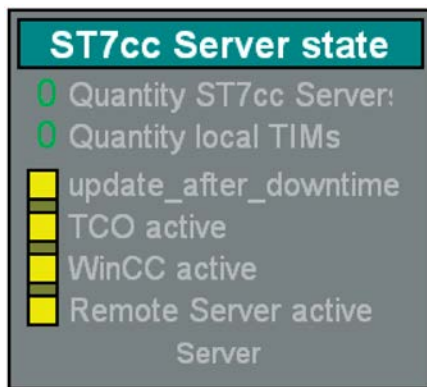


Figure 6-11 Picture typical for ST7cc server status information

The significance of the individual LEDs is explained below.

LED name	LED display	Meaning
Downtime synchronization	Green	Inactive
	Flashing yellow	Started / waiting / aborted / local
	Flashing green	Running
	Gray	Undefined status during startup phase
TCO active	Yellow	TCO communication unknown
	Green	TCO communication OK
	Red	TCO communication problem
	Gray	Undefined status during startup phase
WinCC active	Yellow	WinCC communication unknown
	Red	WinCC communication problem
	Green	WinCC communication OK
	Gray	Undefined status during startup phase
Remote server active	Yellow	Remote server communication unknown
	Red	Remote server communication problem
	Green	Remote server communication OK
	Gray	Undefined status during startup phase

Server faceplate

The faceplate contains further details on the status information of the server (see figure).

To display the faceplate of the server, left -click on the picture typical of the required server.

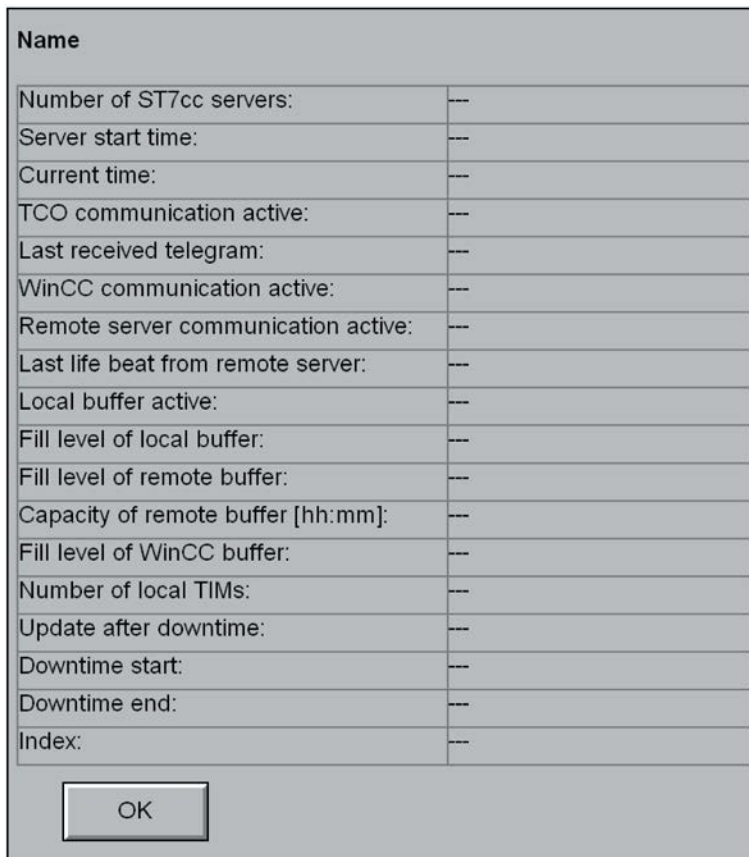


Figure 6-12 ST7cc server faceplate

Variables	Meaning
Number of ST7cc servers	Possible displays: 1: For single system 2: For redundant system
Server start time	Time when the <i>ST7cc server</i> program started
Current time:	Current time on the PC
TCO communication active:	Possible displays: <ul style="list-style-type: none"> • no • yes Display indicating whether or not the <i>tco server</i> program has started. This program precedes the <i>ST7cc server</i> and handles communication between the local TIM and the <i>ST7cc server</i> program.
Last received message	Time stamp of the message last received from a local TIM.
WinCC communication active:	Possible displays: <ul style="list-style-type: none"> • no • yes Displays whether WinCC Runtime is active.

Variables	Meaning
Remote server communication active:	<p>Possible displays:</p> <ul style="list-style-type: none"> no yes <p>Displays whether the redundant partner PC can be reached and whether the <i>ST7cc server</i> has started on this computer.</p>
Last life beat from remote server:	Time stamp of the last life beat message sent from the remote partner PC.
Local buffer active:	<p>Possible displays:</p> <ul style="list-style-type: none"> no yes <p>If WinCC runtime is not active, all messages are stored temporarily in the local buffer. The buffer is activated only when WinCC runtime is deactivated.</p>
Fill level of local buffer:	<p>Default: 0 % (0 of 100000)</p> <p>Display of the number of messages stored in the local buffer as a percentage and absolute number.</p>
Fill level of remote buffer:	<p>Default: 100 % (100000 of 100000)</p> <p>Display of the number of transferred messages stored in the remote buffer as a percentage and absolute number. All the messages from the control center TIM are stored in this buffer in case they are required for redundancy synchronization between the two server PCs. After the first fill phase, this buffer is always filled to 100%.</p>
Capacity of remote buffer (hh:mm):	Elapsed time in hours and minutes since the oldest message in the remote buffer. This indicates the period of a failure that can be covered by the remote buffer.
Fill level of WinCC buffer:	<p>Default: 0 % (0 of 100000)</p> <p>Display of the number of messages stored in the WinCC buffer as a percentage and absolute number.</p>
Number of local TIMs:	Number of configured <i>TIMs</i> on the MPI bus or on Ethernet.
Update after downtime:	<p>Possible displays:</p> <ul style="list-style-type: none"> --- (=inactive) starting running remote pending preparing running local <p>Instantaneous status display if there is a failure synchronization following an <i>ST7cc</i> runtime restart.</p>
Downtime start:	Time at which the partner downtime started. This is displayed only when the <i>ST7cc</i> runtime of the failed computer restarts.
Downtime end:	Time at which the remote partner restarted. This is displayed only when the <i>ST7cc</i> runtime of the failed computer restarts.
Index:	Displays the current data record number to be synchronized in the remote buffer.

6.4 Diagnostics: Trace

The trace function is started in a separate program called *trhi.exe*. To start the trace program, follow the steps outlined below:

1. Select the menu sequence *Start > Simatic > ST7cc > ST7cc Trace*.

The *trhi* window opens.

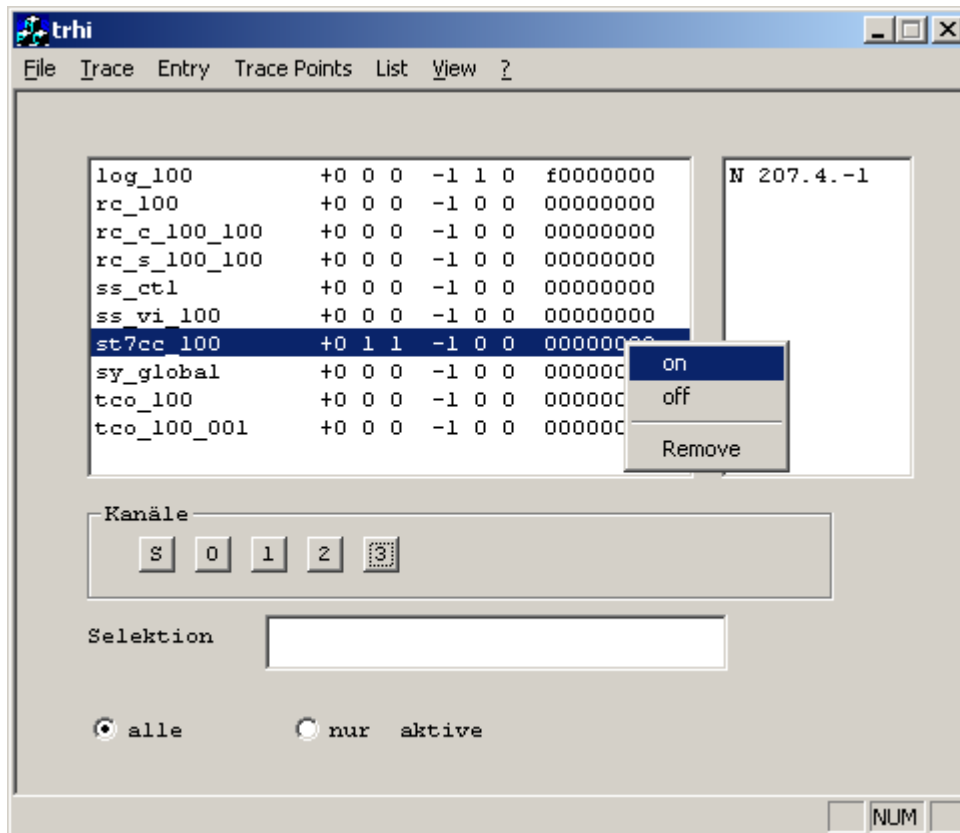


Figure 6-13 Window of the trace program

The basic trace window displays all programs that have started. Up to four programs can be written to different trace output files and the same time.

To activate the trace for a program, follow the steps outlined below:

1. Select the program by left-clicking on it.
2. Right-click to open the context menu and select *on*.
3. In the Channels box, click the button (0, 1, 2 or 3) that corresponds to the open "Trace channel 0, 1, 2 or 3".

Note

Once trace diagnostics is completed, the trace output for this program must be turned off again

6.4.1 Trace output dialog

Trace dialog with output

To open the output dialog (see figure), select the menu sequence List > Channel 0.

The output dialog contains a list box with the trace output and the following check boxes:

- (De)activate automatic scrolling of the output of the messages
- Clear or messages from the output dialog.

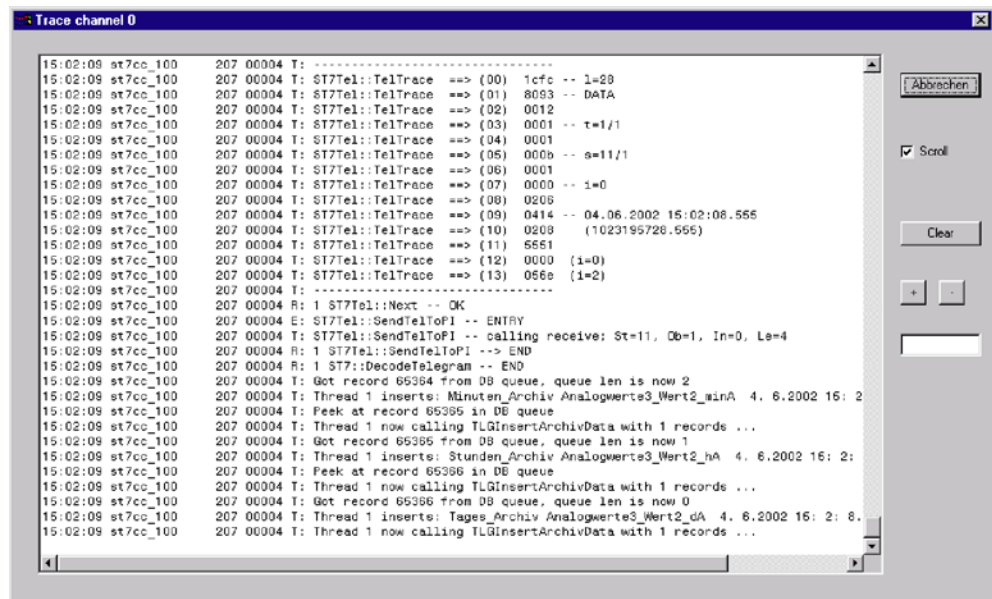


Figure 6-14 Trace dialog with output

Structure of the trace rows

Each trace row contains:

- time information,
- information on the triggering trace point:
207.004 = ST7 message trace
and other trace points for internal diagnostics
- a function call ID:
E = entry,
T = text,
R = return
- an English trace text beginning with the name of the calling function.

The trace example (see figure) shows the analysis of the content and processing of a time synchronization message.

6.5 Diagnostics: System typical

The system typical allows visualization of some of the main performance data of the system. To use the system typical, the subscriber 0 (System) must be set up with sub type 0. To regenerate the variables of the subscriber, the following variables are configured:

Messages_per_second

Number of messages received in the last second (see also section Picture typicals and faceplates for a station (Page 306)).

Messages_per_minute

Number of messages received in the last minute (see also section Picture typicals and faceplates for a station (Page 306)).

Messages_total

Total number of messages received since the server started up (see also Section Picture typicals and faceplates for a station (Page 306)).

Length_database_queue

Length of the queue for values still to be archived in Tag Logging.

Time_difference

Difference in seconds between the computer time and the time of the time master TIM.

PM-AQUA link

7.1 PM-AQUA process links

ST7cc is linked to PM-AQUA over PM-AQUA process links (see PM-AQUA configuration manual and description of runtime module V3.0, issue February 1999, Chapter ACRON link (Page 327) Process Links).

Below, there is a brief outline of this chapter that is not intended as a full description.

Process link

A process link uses raw data variables to transfer data from WinCC to PM-AQUA process-controlled. Process-controlled, in this context, means that ST7cc determines the point in time when the data is transferred.

Three WinCC tags for connection are required to allow data transmission: One raw data tag over which the data message is sent, and two 16-bit tags used for the handshake between the PC program and the automation system (AS).

The conventions for the WinCC tags are as follows:

PM-AQUAxxASPC: 16-bit word variable for handshake from ST7cc

PM-AQUAxxPCAS: 16-bit word variable for handshake from PM-AQUA

PM-AQUAxxDATA1: Raw data variable (filled by ST7cc with a value + time stamp)

In the variable name, xx is a placeholder for the process link number.

Process link number

A process link is identified by a process link number. ST7cc Config support process links 1 – 9.

Index

The data transmitted over a process link are identified by an index. A process-controlled measured value or maintenance counter is assigned exactly one index within a process link. Any index can be assigned, however it is better to use consecutive indexes within a station to preserve clarity.

Sequence of data transmission

- ST7cc fills the raw data variable *PM-AQUAXXDATA1* with data.
- ST7cc writes the bits to *PM-AQUAXXASPC*.
- Bit 1 is set to indicate that the data is intended for archiving.
- Bit 0 is set to indicate that the data is complete.

- PM-AQUA accepts data.
- PM-AQUA sets *PM-AQUAXXPCAS* with the error number.
- PM-AQUA sets *PM-AQUAXXPCAS* bit 0 to indicate that the data was accepted.
- ST7cc evaluates the error number and deletes *PM-AQUAXXASPC* bit 0/1.
- PM-AQUA deletes *PM-AQUAXXPCAS* bit 0.

7.2 PM-Aqua configuration with ST7cc Config

Project settings

To archive process values in PM-Aqua, you will need to activate the function in the ST7cc project settings (see section Project settings: Server (Page 118))

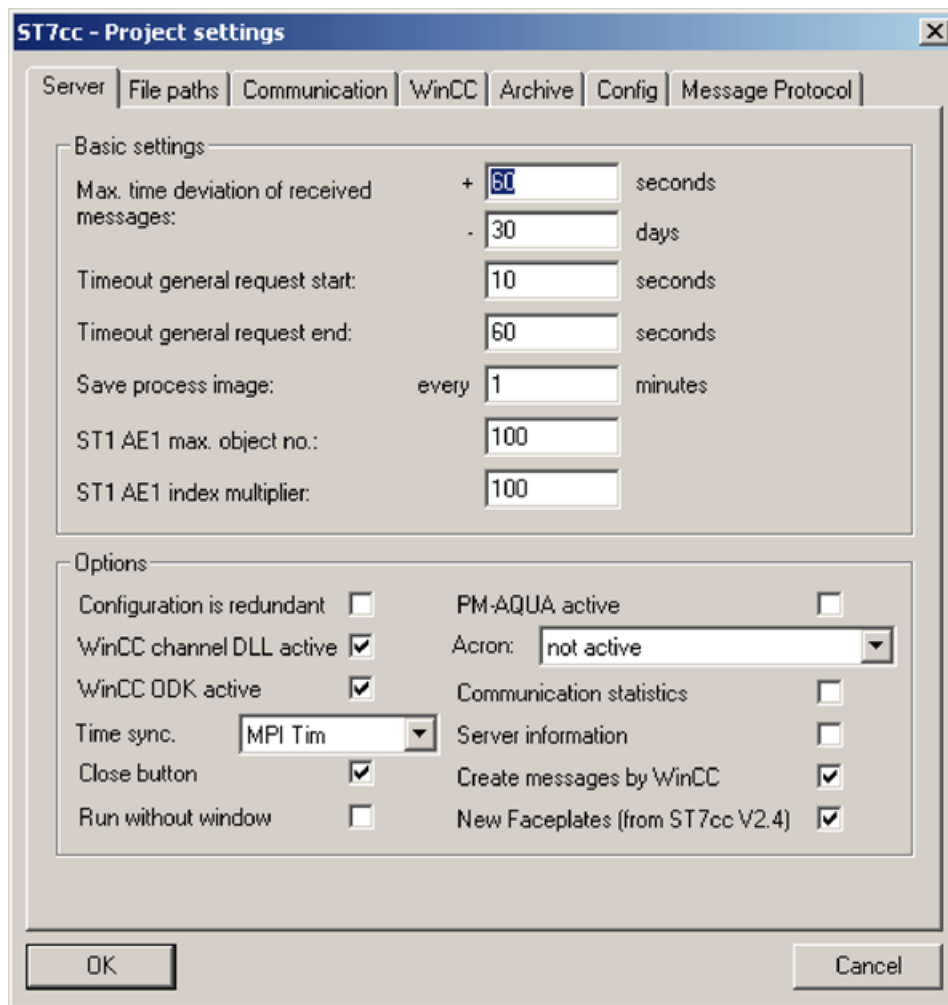


Figure 7-1 Activating data transfer to PM-Aqua in the project settings of the ST7cc server

Configuring the variables for PM-AQUA process links

A subscriber with subscriber number *0* (system subscriber as container for the system objects) must be created. Within this subscriber, object *10* is reserved for PM-AQUA process links.

For the process links, you use typical type *2* subtype *1* and an instance of this must be created once for each connection used within object *10*. The number of the instance must be identical to the number of the process link, the name of the instance must be *PM-AQUA0x*, where *x* stands for the number of the process link:

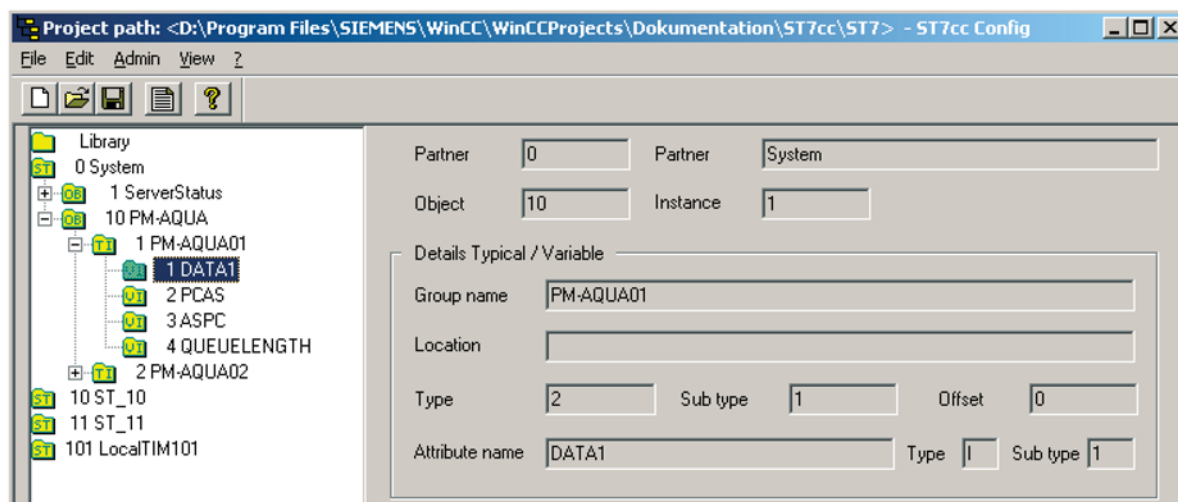


Figure 7-2 Configuring the variables for PM-AQUA process links

The variables for the PM-AQUA process link can then be generated just like any other variable.

Archiving instructions for PM-AQUA

If an archiving instruction references the archive name *PM-AQUA0x* and, instead of a variable name, contains a valid index number within the PM-AQUA process link, archiving will use PM-AQUA instead of WinCC.

When you generate, no WinCC archive tag is then created.

The unit field of the archiving instruction can specify the duration of an interval. This consists of a number followed by one of the time units *s*, *m*, *h*, *D* for *seconds*, *minutes*, *hours* and *days*. Examples of valid intervals might be *15s*, *1m*, *5m*, *1h*, *1D* etc.

In the case of a measured value, the interval is interpreted as an averaging period. Since PM-AQUA interprets the time stamp as the start of the interval for the mean value transfer, but the archiving functions of ST7cc, on the other hand, normally use the transfer time stamp (= time stamp of the end of the interval), this information is used by ST7cc to connect the time stamp for PM-AQUA.

In the case of a counted value, the interval is transferred as value resolution information to PM-AQUA. The other parameters of the archiving instruction (scale information) are ignored.

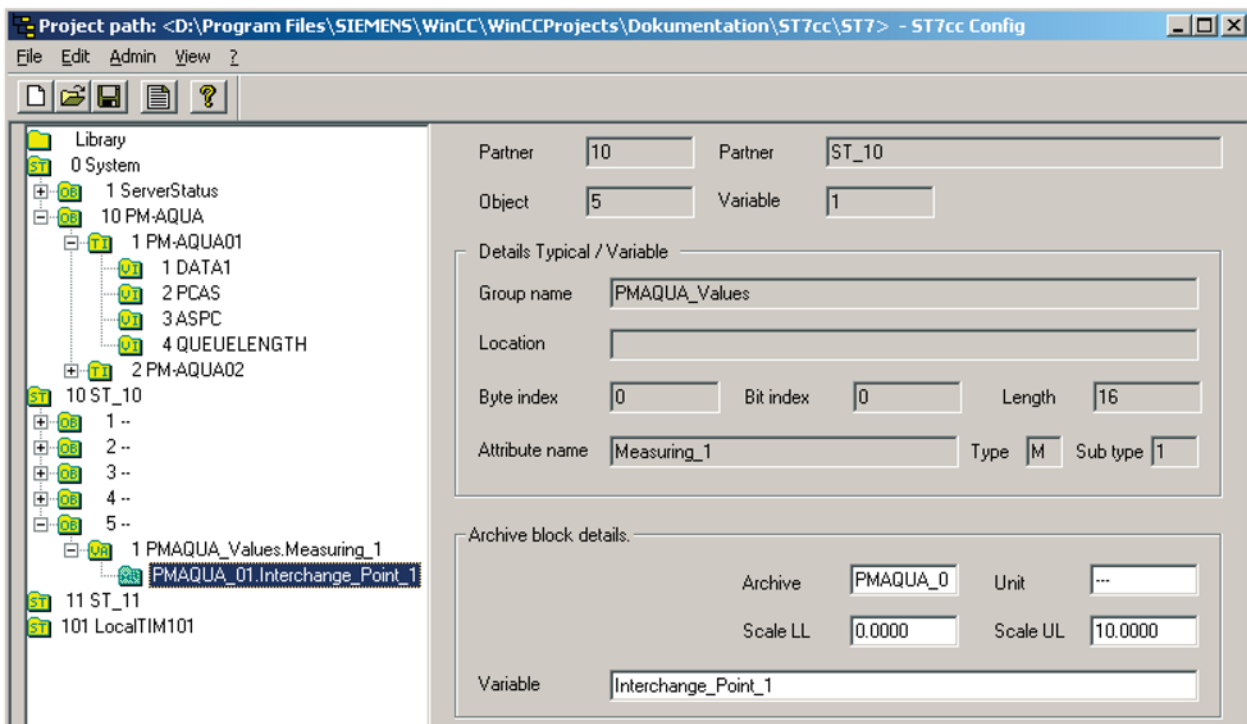


Figure 7-3 Archiving instructions for PM-AQUA

The preprocessing functions of ST7cc (raw value scaling, interval-related compression) can be used as options. It is advisable to precompress the process data in the PM-AQUA basic cycle (15 minutes) to reduce the archiving load on the system (see also section Optimization of the handshake procedure (Page 324)).

Message types and data formats used

Measured values and signals are forwarded to PM-AQUA as measured values (message type 0x02), counted values as counted values (message type 0x23). If the unit field contains a valid interval, a counted value as operating hours counter reading using the specified interval as value resolution information is sent to PM-AQUA (message type 0x21).

One value per message without status is sent as an S7 float value. The S7 time format is used as the time stamp format.

All the values transferred to PM-AQUA always have the archiving ID added to them.

7.3 Optimization of the handshake procedure

As default, PM-AQUA is configured so that the handshake variables are monitored in a 2-second cycle. This results in a total duration for the handshake procedure of 4 seconds for value.

To optimize the handshake in conjunction with ST7cc, the following registry entries of the type DWORD should be made for PM-AQUA and set to the value 0:

HKEY_LOCAL_MACHINE\SOFTWARE\SIEMENS\PM\DMSYSTEM\ChangeCycle

HKEY_LOCAL_MACHINE\SOFTWARE\SIEMENS\PM\DMSYSTEM\CycleCycle

Valid values for these entries for monitoring the handshake variables are as follows:

0 = on change

1 = 250 ms

2 = 500 ms

3 = 1 s

4 = 2 s

Setting the value 0 achieves optimum reaction times.

As long as the PM-AQUA raw data interface is operated only in conjunction with ST7cc, no side-effects should be expected.

Nevertheless, on a system (Pentium II-300, 128 MB RAM), a throughput of an order not exceeding 1 - 2 values / second should be expected.

ACRON link

8.1 Importing historical files (CSV, DBASE)

Background

The ACRON provider is used for all interfaces to receive data from the process. This also applies to the file interface. The received files are stored in a configurable path (import path). The provider checks cyclically whether a new file has been created in this path. If this is the case, the data from this file is automatically entered in the ACRON database. Following this, the file is then deleted by the ACRON provider. Any errors that occur during entry of the data are logged in a log file (file_name.CS_). This log file is in *.CSV format and can be imported again by renaming to file_name.CSV.

Measured values, maintenance counter readings and alarms can be imported.

Maintenance pulses cannot be imported! To allow this, maintenance counters must be implemented in the PLC.

Link for importing historical values

All data is always imported. Existing data with the same time stamp is overwritten.

To import data, two formats are supported:

- DBase format with the file extension **.DBF*.

and

- CSV format with the file extension **.CSV*.

To be able to distinguish value files and historical alarm files, the file names must begin with the letters *A* or *B*.

The table structure is the same for both formats and is interpreted as follows.

Field name	Field type	Remarks
DATETIME	Char, max. 30 characters	The time and date is set here as a character string the format: YYYY-MM-DD HH:MM:SS
VARIABLE	Char, max. 64 characters	This specifies the name of the external variable configured in ACRON.
VALUE	Char, max. 64 characters	The value of the currently specified value of the variable is set here as a character string. Both a period and a comma are accepted as the decimal delimiter. Scientific notation is also accepted. The valid value range is between -10^{34} and $+10^{34}$

8.1 Importing historical files (CSV, DBASE)

Table 8- 1 Other optional fields

Field name	Field type	Remarks
STATUS	Numeric, 1 character	0:
MEANTIME	Char, max. 32 characters	If this entry does not exist or is set to 0, ACRON assumes that this is a single value. If the entry is > 0, the value applies for a period of seconds counting backwards from the time sent.
TYPE	Char, 1 character	Indicates the type variant. A: Mean value or instantaneous value M: Minimum X: Maximum

A table can contain any number of records. Each entry that has been successfully adopted by ACRON is deleted. Once the entire table has been adopted successfully, it is completely deleted. The name of the table must begin with the letters A or B, the file extension must, however, be *.DBF or *.CSV.

A semicolon is used as the separator between the fields.

Example:

```
1996-03-25 17:23:00;Inflow;100.5
1996-03-25 17:23:00;Outflow;223.5
1996-03-25 18:10:00;Inflow;110.5
1996-03-25 18:10:00;Outflow;202.5
1996-03-25 18:12:00;Inflow;110.5;1;0;X
1996-03-25 18:12:00;Outflow;202.5;1;86400;A
```

All the rows in the file that do not begin with a number are ignored. Fields enclosed in characters such as ' or " are accepted.

Syntax of the external variables

Enter the name of variables in the ACRON Designer just as they are used in the DBASE or CSV file.

Provider settings

- Write interval:
This setting has no relevance for acceptance of historical data.
- First value:
Recommended setting: 0 sec.

- Measurement interval:
Recommended setting: 60 sec.
- Max / Min evaluation:
The option is not supported by this link.

Driver parameters

1. Parameter:
Path for saving the files.
2. Parameter:
Time tolerance: If the historical data was measured at different points in time compared with the automatic data, a time tolerance in seconds can be specified here to allow the data to be sorted along with the existing data. This saves disk space since no new records need to be created for the historical data if data records already exist within this tolerance range. If, however, you only want to log historical data or require the time of the measurement exact to a second, enter zero seconds here.
In most situations, 30 seconds is a practical value. If you are already using a provider with a different link to record data, you should enter half the write interval of this provider.

8.2 ACRON project settings with ST7cc Config

To allow ST7cc in conjunction with ACRON to generate suitable files in CSV format, the following project settings are required in ST7cc Config:

1. Open the Windows Explorer and create a directory for saving the CSV files. You can also use the default directory *c:\siemens\st7cc\lacron* that is created automatically when you install ST7cc.
2. In the "Server" tab, under "Acron" enable the CSV archiving active function. Here, you can also decide whether you want to write only the configured ACRON archive blocks or the entire message traffic to the CSV file. You will find a description of the functions CSV archiving active, CSV data logger active and WinCC Tag Logging active in section Project settings: Server (Page 118).

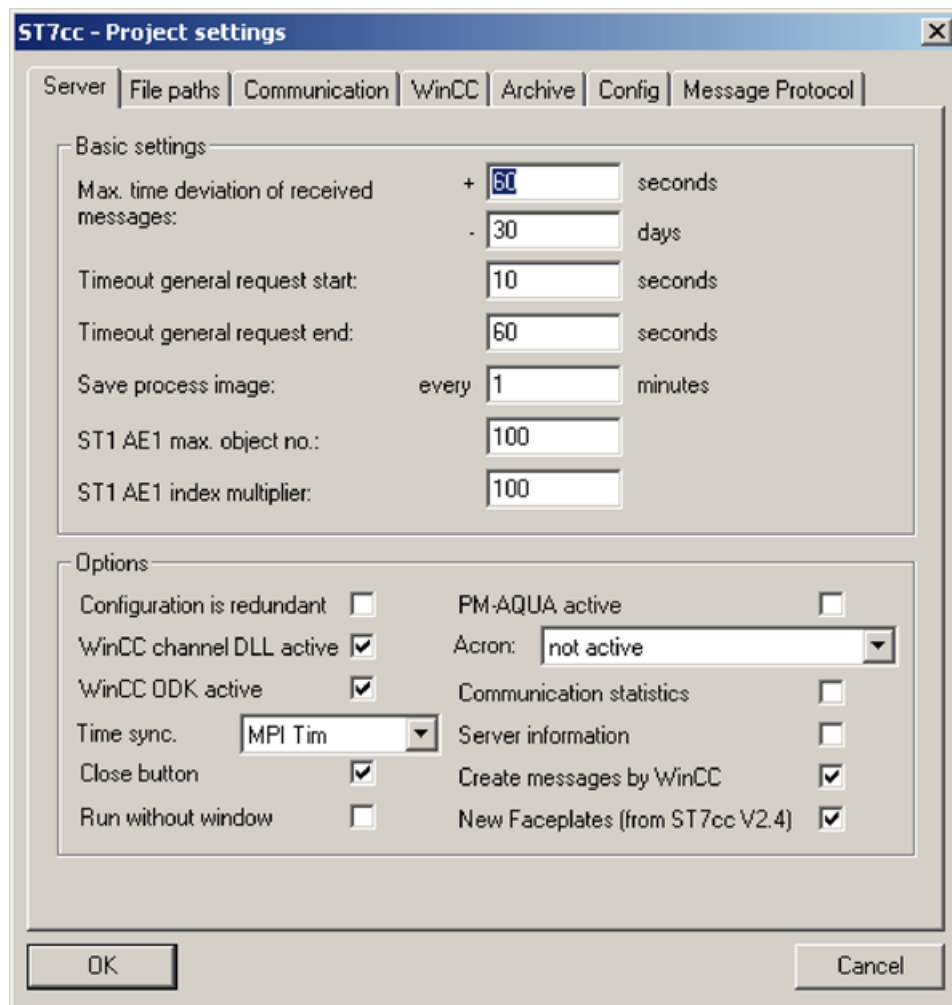


Figure 8-1 Selecting the *CSV archiving active* function

1. Select the directory you have just created under *Acron* by clicking on (...).
2. Enter the restore time for the current CSV file in seconds.

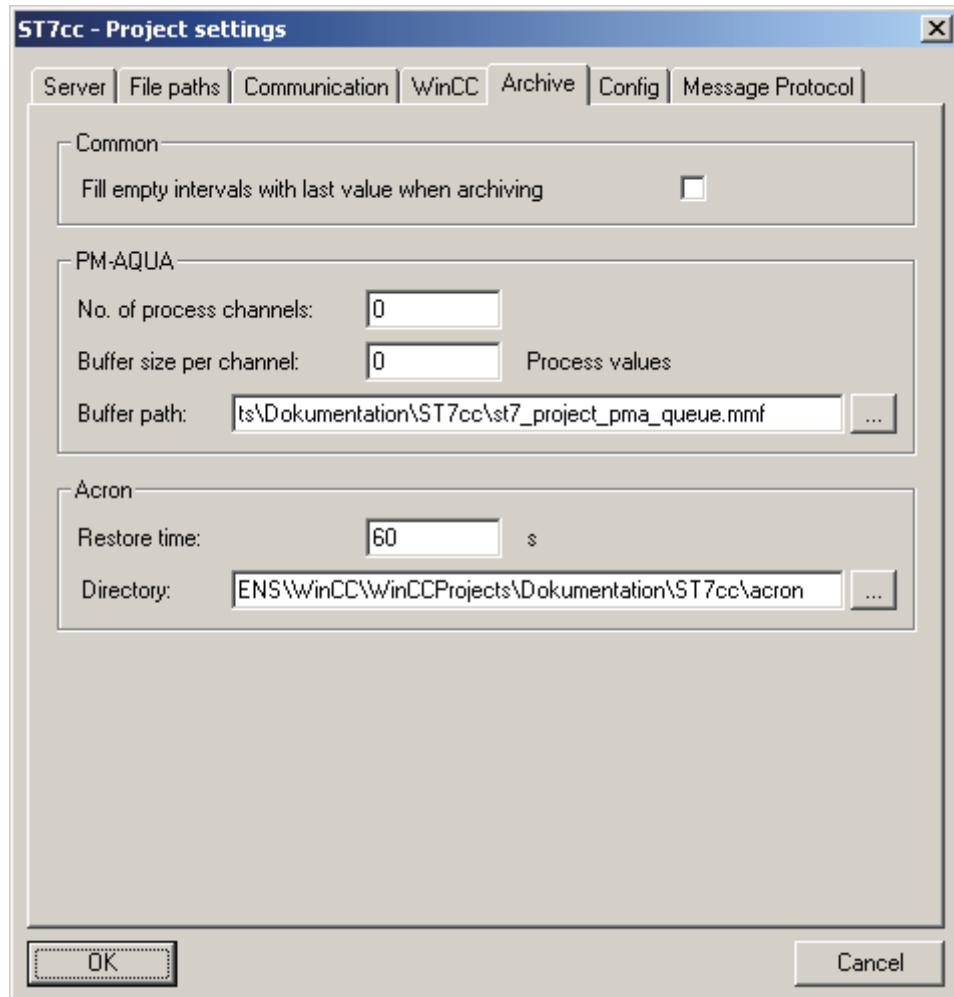


Figure 8-2 Setting the ACRON directory and the restore time

8.3 ACRON configuration with ST7cc Config

Archiving instructions for ACRON

If an archiving instruction references the ACRON archive name, instead of archiving over WinCC, the data is written to a file in CSV format.

When you generate, no WinCC archive tag is then created.

The unit field of the archiving instruction can specify the duration of an interval in seconds.

If the entry is > 0, the value applies for a period of seconds counting backwards from the time sent.

If this entry does not exist or is set to 0, ACRON assumes that this is a single value.

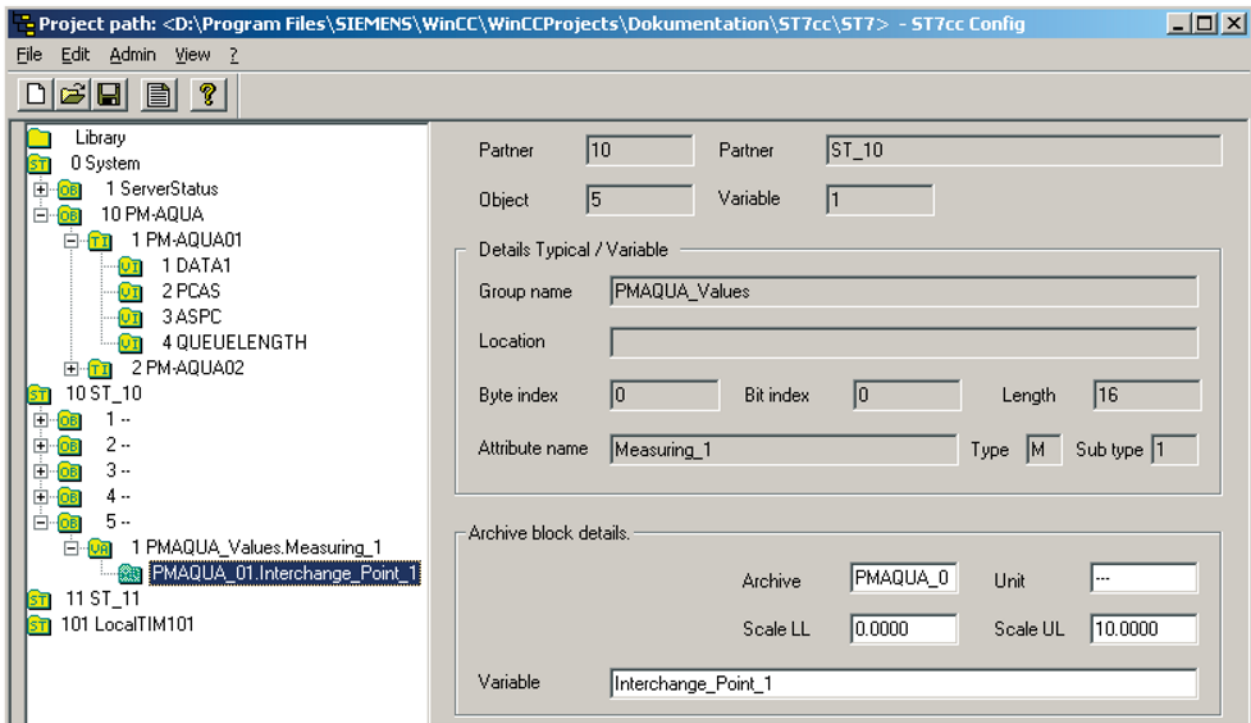


Figure 8-3 Archiving instructions for ACRON

The preprocessing functions of ST7cc (raw value scaling, interval-related compression) can be used as options. It is advisable to precompress the process data for ACRON to reduce the archiving load in the system.

Technological typicals

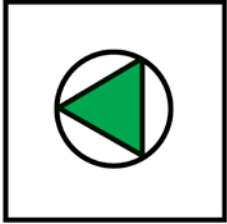


9.1 Aims


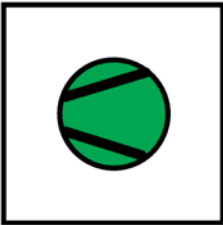
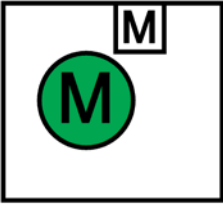

Picture typicals

To support users when engineering their plants, model engineering templates are available for commonly used technological objects. An engineering template for a technological object includes one or more picture typicals, a faceplate, and one or more ST7cc typicals. The basis of the engineering templates is formed by information units whose data structures and content was specified in the form of a model for specific objects. The data structures were specified based on an analysis of several projects.

If the available templates meet the requirements of your specific project, they can be adopted unchanged. If this is not the case, the user can modify data structures and edit ST7cc typicals, picture typicals, and faceplates to suit the situation in hand. Section Overview (Page 335) provides you with an overview of the use of engineering templates in the overall configuration. The section also describes the general structure of the configuration templates.

Templates have been created for the following technological objects:

Picture typical	Technological object
	Pump (see section Templates for the pump technological object (Page 346))
	Motor1 (see section Templates for the 1Motor technological object (Page 353))
	Generator (see section Templates for the generator technological object (Page 360))

Picture typical	Technological object
	<p>Valve (see section Templates for the valve technological object (Page 367))</p>
	<p>Compressor (see section Templates for the compressor technological object (Page 374))</p>
	<p>Motor2 (Motor with two 2 forward and 2 reverse gears) (see section Templates for the Motor2 technological object (Page 381))</p>
	<p>Slider (see section Templates for the slider valve technological object (Page 389))</p>

9.2 Overview

The configuration of technological objects when working with SINAUT involves the management, communication, and automation levels (see figure).

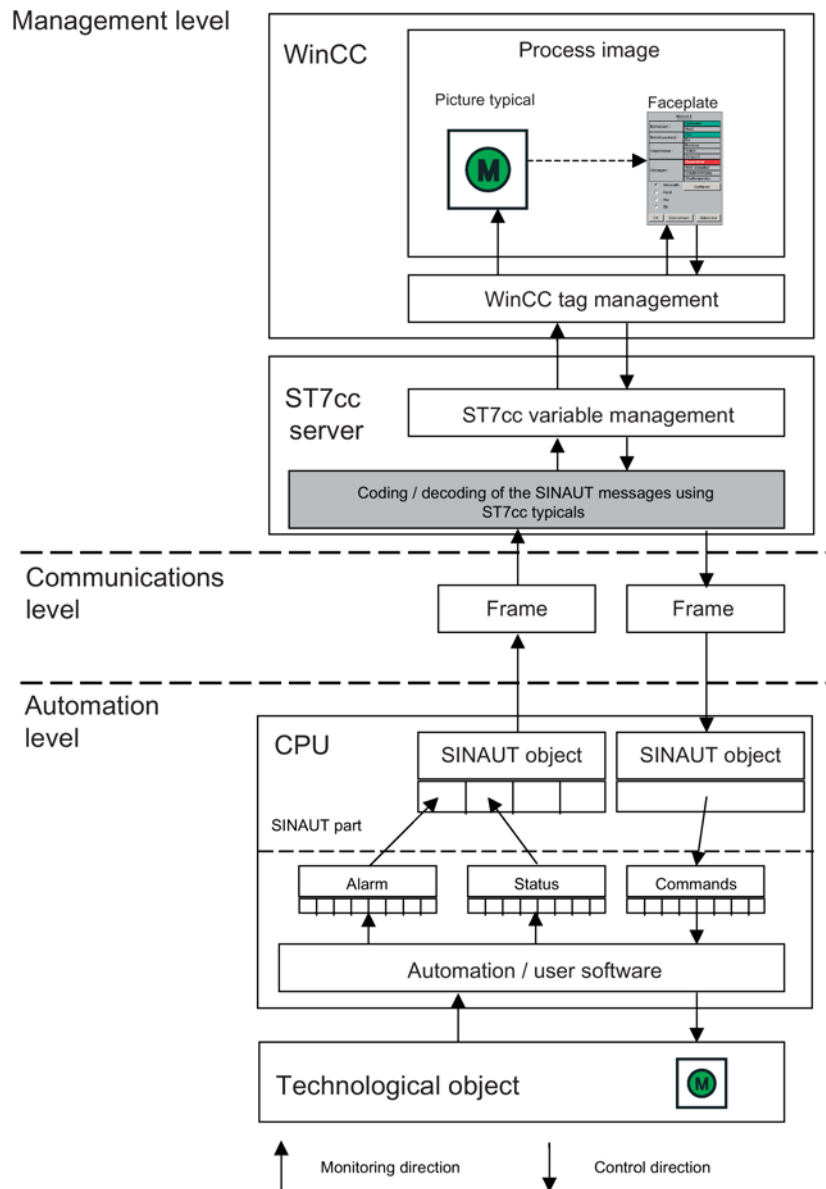


Figure 9-1 Structure of the 3 SINAUT levels

As mentioned above, the specified model data structures of the information units (data subarea of an object data area) are a necessary prerequisite for using the engineering templates.

The figure shows only the engineering units (picture typical, faceplate, ST7cc typical, and SINAUT object with its information units) relevant for engineering.

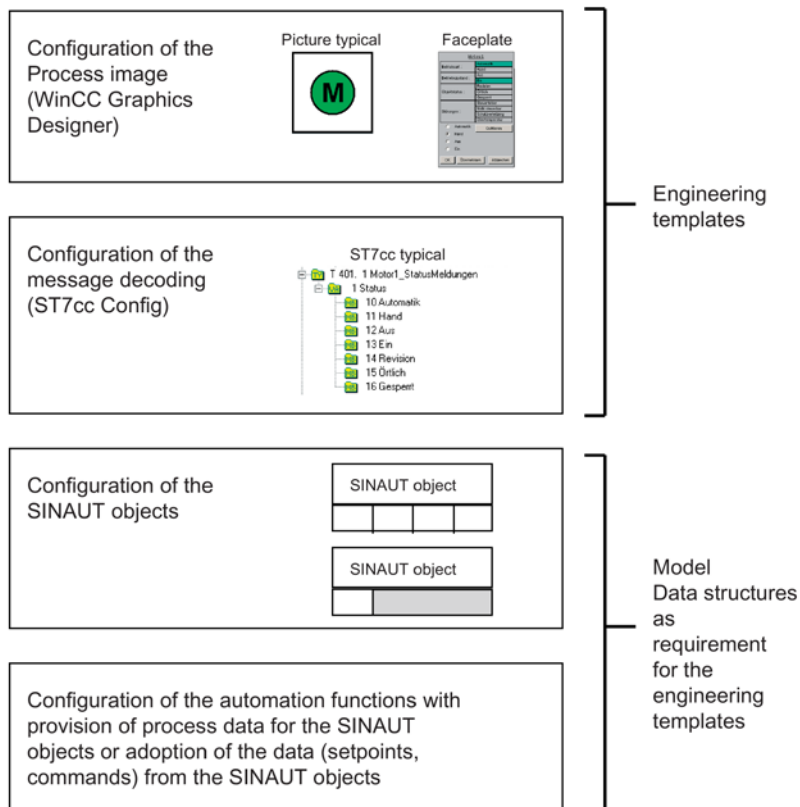


Figure 9-2 Engineering view

To decode SINAUT objects in ST7cc, the user has ST7cc typicals available. When the WinCC tag management is generated, all the WinCC tags required for further processing of the process values transferred by SINAUT are generated. The preconfigured picture typicals and faceplates then access these WinCC tags. These represent the statuses of the technological objects in the WinCC process picture and/or allow control of the technological objects.

The ST7cc typicals are in the ST7cc library, the picture typicals and faceplates are in the ST7cc installation directory in the GraCS subdirectory. Using the function already described in Section Copy faceplates to a WinCC project (Page 116) Copying faceplates to a WinCC project, you can copy the picture typicals and faceplates to your WinCC project directory.

9.2.1 ST7cc typical and data structure of an information unit

Section SINAUT object (Page 158) describes the relationship between the data area of a SINAUT object and the data subarea from this data area. The data subarea represents the information unit that will be mapped to an ST7cc variable and then to a WinCC tag. How the information unit is mapped to an ST7cc variable is defined in ST7cc Config by an ST7cc typical. The WinCC tags, on the other hand, are the information carriers for the picture typicals and faceplates.

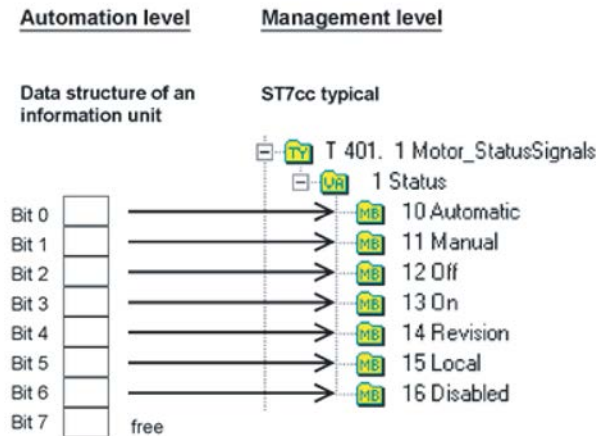


Figure 9-3 Decoding an information unit with an ST7cc typical

The figure shows how the data structure (bit assignment) describing the status of an object is mapped to a variable. The detailed relationships can be found in section Background knowledge on configuring (Page 157).

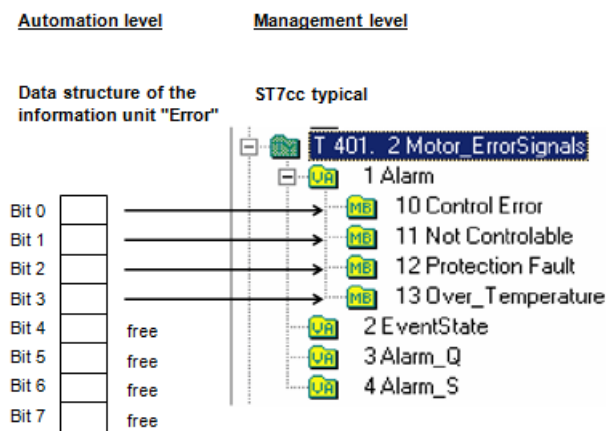


Figure 9-4 Decoding an information unit (alarm messages) with an ST7cc typical

Note

To be able to use the group display function, the EventState, Alarm_Q and Alarm_S variables are defined in the technological typicals. See also section Group display (Page 174).

These examples make it clear that the use of our engineering templates implicitly defines the data structure of the information unit and a modification of the data structure at the automation level causes a modification of the engineering templates.

Note

With all engineering templates, messages must be generated by WinCC.

9.2.2 Definition of the information units

Due to the analysis that was made and the requirement of making optimum use of SINAUT communication possibilities, the number of statuses that can be adopted by a technological object was spread over three information units. These are:

- The status information unit for all statuses of a technological object that do not indicate a problem. These include, for example, the states On, Off, Automatic etc.
- The Alarm information unit for all statuses of a technological object that indicate a problem or a status to be transferred under certain conditions. These include, for example, the statuses control error, overtemperature etc.
- The Command information unit for outputting commands to a technological object, for example On, Off etc.

The statuses were split into the information units Alarm and Status so that it is possible when using dial-up connections to decide which statuses would cause a connection (incurring charges) to be established for the transfer of process data. In this case, the Status and Alarm information units must be stored in separate SINAUT objects. In practice, a station connected over a dial-up network only initiates establishment of a connection when there is a status change in the Alarm information unit. The control statuses of the Status information unit are then only transferred when the control center establishes a connection time-driven or to output commands or when the station itself establishes a connection due to an alarm.

In contrast to dial-up connections, stations connected over dedicated lines are always connected with the target subscriber. A data change is therefore always transferred immediately. The *Status* and *Alarm* information units can therefore be configured in a SINAUT object.

Complex technological objects may make further information units necessary and these can be defined and included in the engineering templates by the user.

9.2.3 Assigning information units to SINAUT objects

To transfer the Status and Alarm information units, ST7 object types Bin04 and Dat12D are suitable. If you want to transfer several technological objects together, for example to minimize message traffic, a Dat12D may be the suitable object type. In the following examples, the SINAUT object type Bin04 is used to transfer the information units.

For commands, the Cmd01B object type is used. A maximum of eight commands are available per object of this type.

Example of a dial-up connection

With this type of connection, status and alarm information is stored in separate objects so that a connection incurring charges is established by the station only when alarm events occur that make such a connection worthwhile.

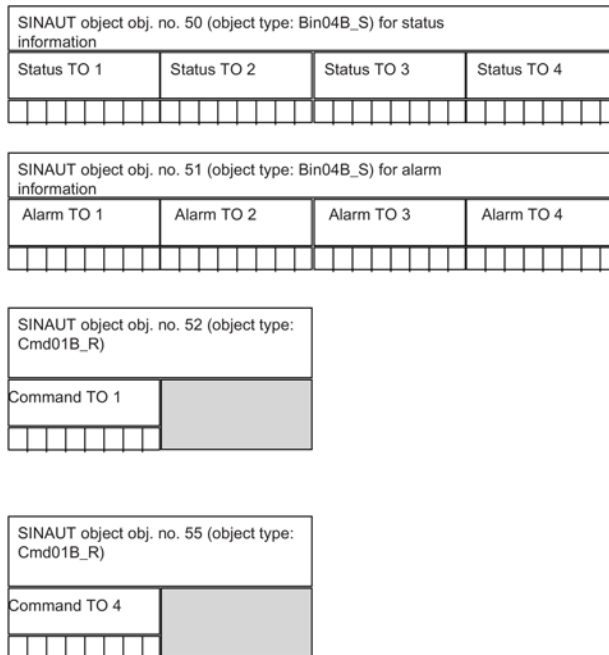


Figure 9-5 Possible assignment of the SINAUT objects for a dial-up connection (TO technological object)

In this case, the data area of a SINAUT object of the type Bin04B holds four information units of the type status or alarm. Due to the specified offset of the typicals, these are positioned on the corresponding data subarea within the object data area during decoding (compare section ST7cc typical and data structure of an information unit (Page 337)). To transfer the commands to a technological object (TO), a SINAUT object must be used for every technological object.

Example of a dedicated line

With this type of connection, alarm and status messages of a technological object can be stored in one SINAUT object because the station is constantly polled anyway and no alarms are generated in the same way as with a dial-up connection. The advantage of this is that all status and alarm messages of a technological object can be stored in one SINAUT object as long as they are not too long. The commands are stored in a separate SINAUT object.

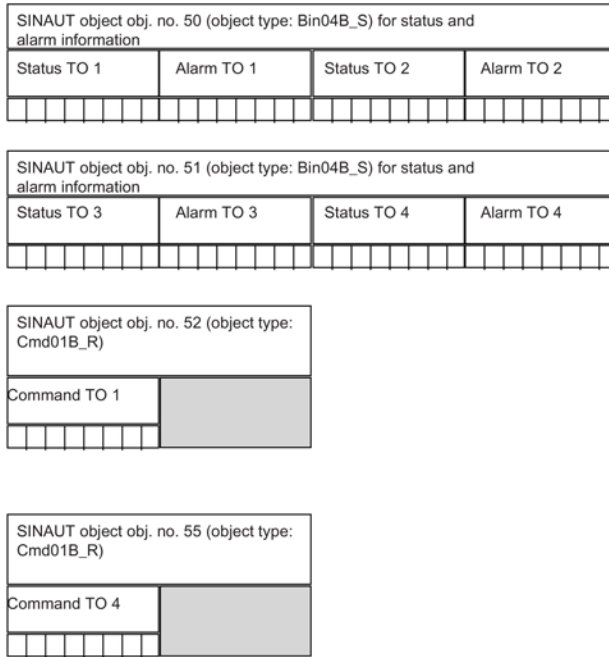


Figure 9-6 Possible assignment of the SINAUT objects for a dedicated line (TO technological object)

9.2.4 Typicals in ST7cc

Overview

An ST7cc typical is available for each information unit of a technological object. The meaning and handling of typicals is described in Sections Object templates and typicals (Page 168) and Principle of decoding using typicals (Page 171). The figure shows typicals for the engineering templates Pump, Motor1 etc.

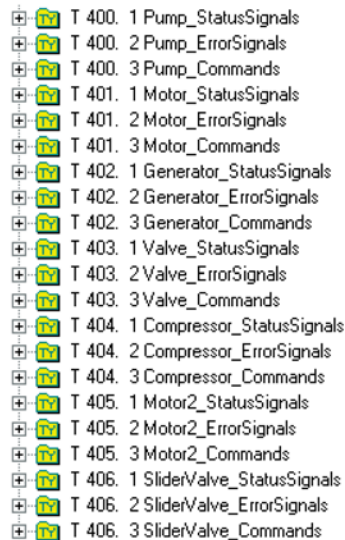


Figure 9-7 All ST7cc engineering templates (excerpt from the ST7cc library)

The typicals for each technological object are described individually in sections Templates for the pump technological object (Page 346) to Templates for the slider valve technological object (Page 389).

9.2.5 Picture typicals in WinCC

Display types

There is at least one picture typical for each technological object. Several variants of many technological objects are supplied, for example to be able to display different flow directions.

The figure shows the minimum display and the total display of the picture typical. This total display was selected to show all the symbols (parts) that a picture typical makes available. In practice, not all symbols will be displayed at the same time; for example, a technological object can be either in the manual or revision status. Which statuses can be active at the same time, however, depends only on the configuration in the automation software. The symbols of the picture typical show the bit assignment of the information unit 1:1; in other words, the WinCC tags (except for: Error status E).

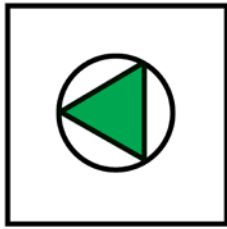


Figure 9-8 Minimum display

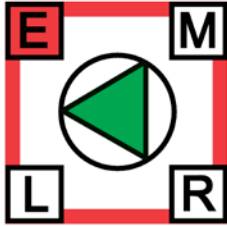


Figure 9-9 Total display

Since several picture typicals can be displayed within a process picture, the minimum display was selected so that only the type of technological object and the statuses On or Off are displayed. The minimum information of the default display only has an additional information added when attention should be drawn to a status that does not correspond to normal operation. This display concept is intended to make it easier for the operator to recognize problems at a glance.

The display of the additional symbols means that the assigned status is active or the opposite status is inactive. For example, displaying the symbol M for manual mode, indicates that the manual mode is active. If the M is not visible, the technological object is in automatic mode. The exact meaning and dynamics of the symbols are explained in sections Templates for the pump technological object (Page 346) to Templates for the slider valve technological object (Page 389) for each individual technological object.

The configuration engineer is free to modify existing picture typicals and faceplates. However, the logical consistency of the engineering components ST7cc typical, picture typical, and faceplate must be maintained.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typicals.

The figure shows the object symbol and the symbol M on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol M means that the bits for manual and automatic mode are both set to 1 or 0 indicating an error. The same thing applies to the object symbol for the status information On and Off.

The implausible bit settings can be caused by the following:

- WinCC has just started up and has not yet been supplied with process data.
- An object name was assigned to the picture typical that does not match the group name of the typical instance.
- The user program at the automation level is not supplying the ST7 objects correctly.

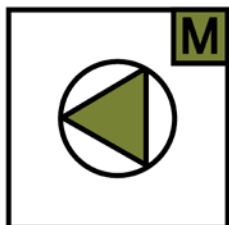


Figure 9-10 Display when data is missing or incorrect

Relationship of a picture typical to a variable

When an instance of an ST7cc typical is created, the instance is given a group name (for example *Pump1*) that references the technological object uniquely throughout the project. The group name and attribute name of a variable (for example Status) together form a unique variable name.

The dynamic display of a picture typical is only possible if the object name of the picture typical corresponds to the group name of the typical instance that references the technological object.

Note

When using picture typicals, please note that the object name of a picture typical must correspond to the group name of the typical instance and therefore to the prefix of the WinCC tag required by the picture typical.

Inserting picture typicals in process pictures

The *st7_technicalObjects.pdl* file provides picture typicals for technological objects such as motors, pumps etc. that the user can use in a project.

While the picture typicals can be generated automatically for the SINAUT subscribers (local TIMs and stations) created in ST7cc Config, this is not possible for the technological objects.

To use the picture typicals and faceplates in your WinCC project, first follow the steps described in section Copy faceplates to a WinCC project (Page 116). Then follow the steps below:

1. Using the Graphics Designer, open the process picture in which you want to place one of the picture typicals.
2. Open the *st7_technicalObjects.pdl* file using the Graphics Designer and copy the required picture typical.

3. Insert the picture typical in your target picture and change the object name to the name of the technological object you want to visualize (= group name of the tag in the WinCC tag management).
4. Insert a copy of the *BInformation1* object into your process picture from the *st7_technicalObjects.pdl* file. One copy is adequate even if you use several technological objects in this process picture.
5. If the tags have already been generated, you can now activate WinCC Runtime and test your picture.

9.2.6 Faceplates in WinCC

Structure and dynamics

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Pump) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

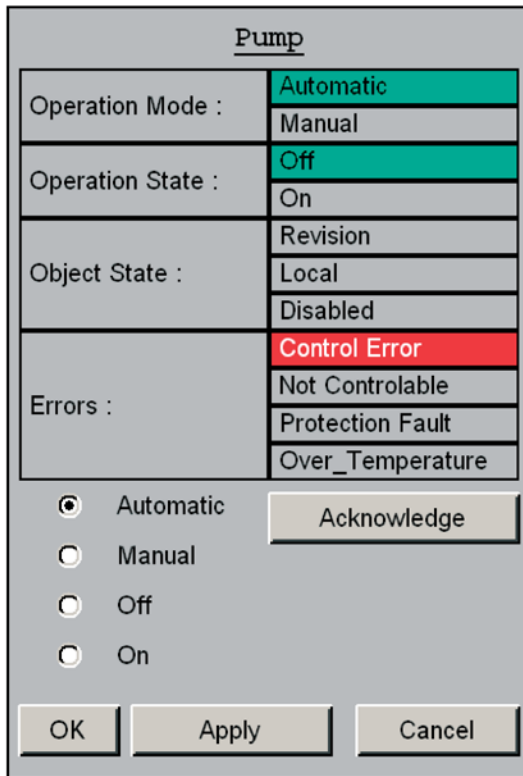


Figure 9-11 Pump faceplate

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the faceplates.

If all the fields beside operation mode and / operation state have a dark yellow background, this means that the bit assignment for Manual / Automatic or On / Off is 1 for both bits. Since a technological object cannot be, for example On and Off at the same time, this display indicates either a transfer error or a configuration error in the automation software. The same applies to technological objects with more than two operating states (the example Motor2). Here, all states whose bit is set to 1 are shown on a dark yellow background if more than one state is displayed as being active.

If all the fields beside operation mode and operation state are shown on a dark gray background, this means that either WinCC has not yet received any data after startup or that the link to the WinCC tags over the object name of the picture typical is incorrect.

Note

When using picture typicals, please note that the object name of a picture typical must correspond to the group name of the typical instance and therefore to the prefix of the WinCC tag required by the picture typical.

Meaning of the buttons

Clicking on *Acknowledge* acknowledges an error.

For more details on acknowledging an error, see below.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.

Clicking on *Apply* executes the selected command without closing the faceplate.

Clicking on *Cancel* closes the faceplate without executing the selected command.

For more details on executing commands, see below.

Sending a command

To send a command to the technological object from the process picture, follow the steps below:

1. Click on the required picture typical.
The faceplate of the picture typical is displayed.
2. Select the required command (for example *Manual*).

3. Click the *Apply* button.

The command is sent to the technological object using SINAUT communication. If the command can be executed at the automation level (automation software), this leads to a status change in the Status information unit. This status change, in turn, is displayed in the picture typical and in the faceplate by a change to the required status or mode. If the command cannot be executed, this is indicated by a suitable error, for example Control Error. The exact meaning of the error messages is explained in sections Templates for the pump technological object (Page 346) to Templates for the slider valve technological object (Page 389) for each individual technological object. The necessary logic must be implemented in the automation software.

4. Close the faceplate with *Cancel*.

Note

If you close the faceplate with *OK*, the selected command is executed at the same time. If you want to make sure that you do not send a command accidentally, close the faceplate with *Cancel*.

Acknowledging an error

To acknowledge an error, follow the steps below:

1. Click on the picture typical containing the error.

The faceplate of the picture typical is displayed.

2. Click the Acknowledge button

Once the error has been acknowledged, the display of the picture typical changes to the display for the error and acknowledged status.

3. Close the faceplate with *Cancel*.

The exact meaning and dynamics of the symbols are explained in sections Templates for the pump technological object (Page 346) to Templates for the slider valve technological object (Page 389) for each individual technological object.

9.3 Templates for the pump technological object

This section describes the components of the engineering template for the pump technological object.

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.3.1 ST7cc typicals

Status messages

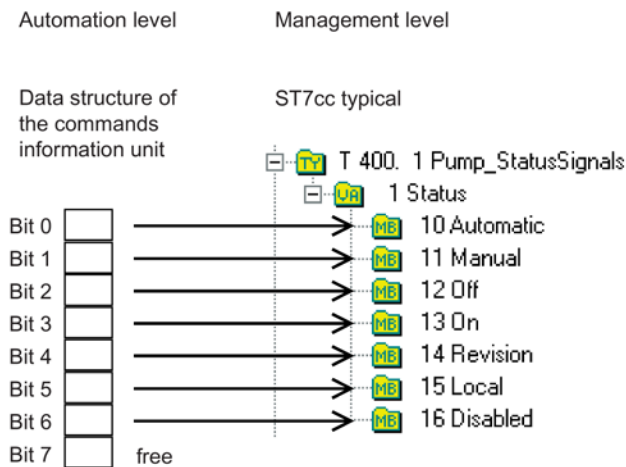


Figure 9-12 Status messages of a pump

This decoding template contains the status messages for a pump. The typical contains a status variable with a length of 8 bits. The variable contains 7 message blocks for the 7 different status messages of the pump. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Off

The technological object is turned off.

On

The technological object is turned on.

Revision


The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is

performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The *Local* mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the *Not Controllable* error.

 WARNING
If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages

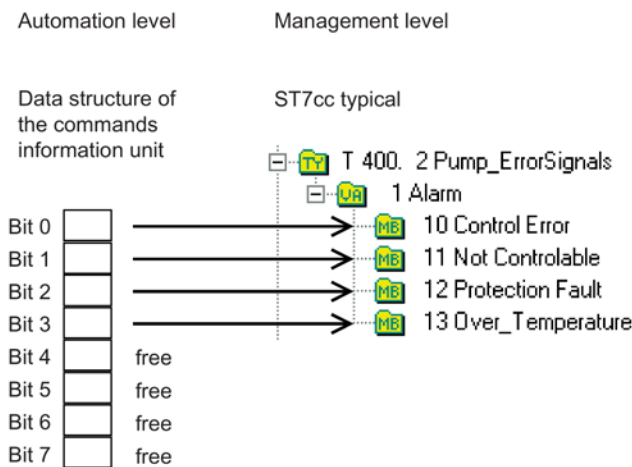


Figure 9-13 Alarm messages of a pump

This decoding template contains the alarm messages for a pump. The typical contains an *Alarm* variable with a length of 8 bits. The variable contains 4 message blocks for the 4 different pump errors. The meaning of these errors is explained below.

Control error

The *Control Error* occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time.

Whether the relay or the technological object to be controlled is defective in such a situation can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The *Not Controllable* error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the *On* command in the *Disabled* mode causes this error display.

Protection Fault

The *Protection Fault* indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The *Over_Temperature* error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Commands

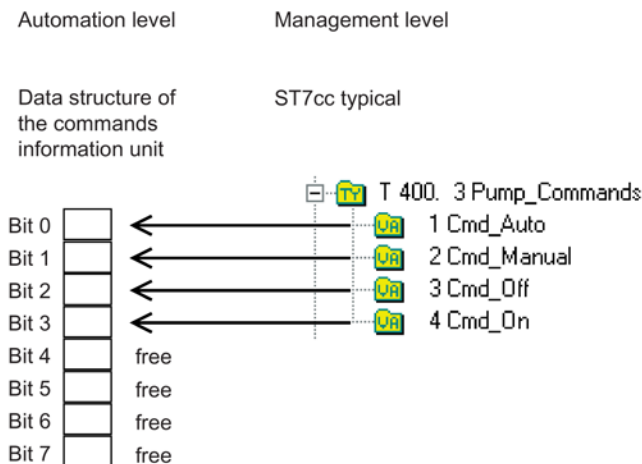


Figure 9-14 Commands of a pump

This coding template contains the commands for a pump. The typical contains a variable for each of the 4 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_Off

The *Cmd_Off* command turns off the technological object.

Cmd_On

The *Cmd_On* command turns on the technological object.

9.3.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical. 4 variants of this picture typical are supplied to allow the conveyor direction to be displayed. The variant shown in the figure shows the conveyor direction left.

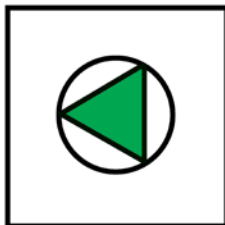


Figure 9-15 Minimum display

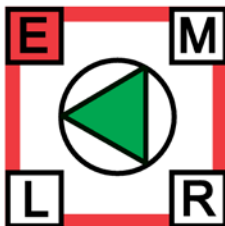


Figure 9-16 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol *M* on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol *M* means that the bits for manual and automatic mode are both set to *1* or *0* indicating an error. The same thing applies to the object symbol for the status information *On* and *Off*.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

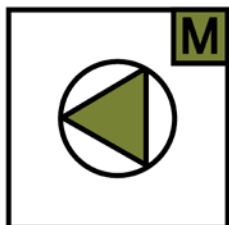





Figure 9-17 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = On
	Light gray background	Operation State = Off
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
	 Flashing red	Error, unacknowledged
	 Red constant	Error, acknowledged
	 Flashing red	No error, unacknowledged (error occurred and was corrected before it was acknowledged)

Note on the no error, unacknowledged status:

If a pump, for example, has the overtemperature error, it is normally turned off by the automatic functions. The pump cools down, the overtemperature error is cleared and the

pump can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.3.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Pump) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

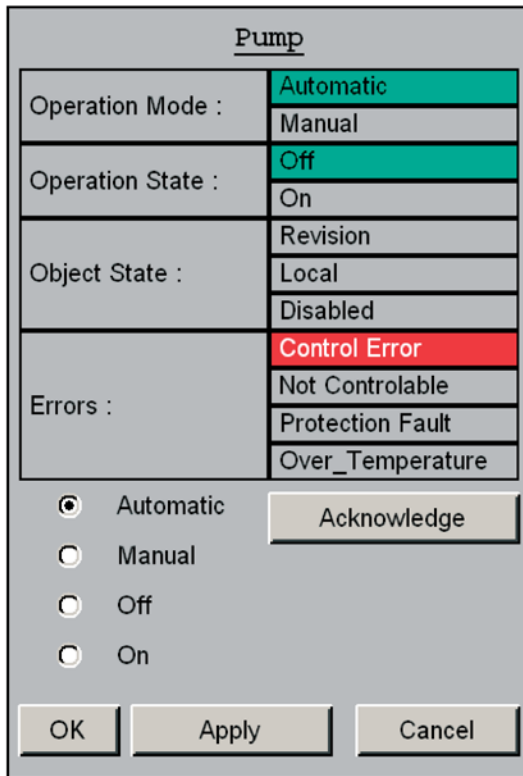


Figure 9-18 Pump faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.
 Clicking on *Apply* executes the selected command without closing the faceplate.
 Clicking on *Cancel* closes the faceplate without executing the selected command.

9.4 Templates for the 1Motor technological object

This section describes the components of the engineering template for the motor technological object.

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.4.1 ST7cc typicals

Status messages

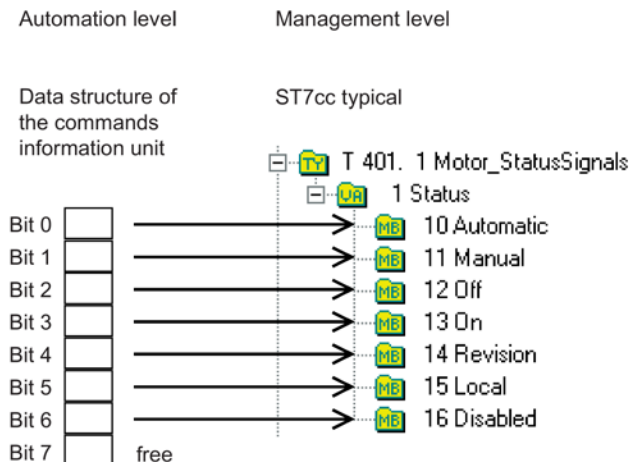


Figure 9-19 Status messages of a motor

This decoding template contains the status messages for a motor. The typical contains a status variable with a length of 8 bits. The variable contains 7 message blocks for the 7 different status messages of the motor. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Off

The technological object is turned off.

On

The technological object is turned on.

Revision

The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The Local mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

 **WARNING**

If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages (errors)

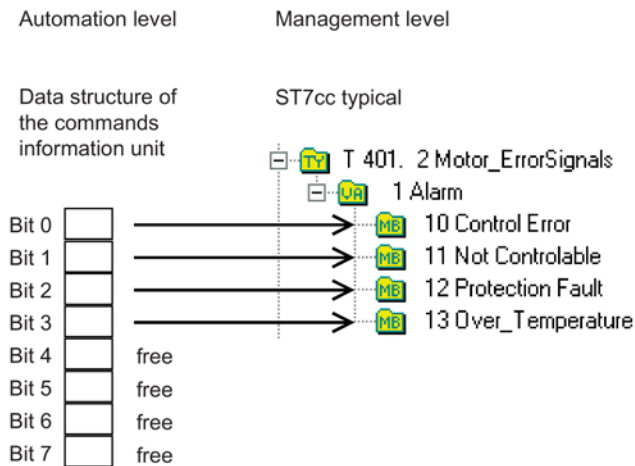


Figure 9-20 Alarm messages of a motor

This decoding template contains the alarm messages for a motor. The typical contains an *Alarm* variable with a length of 8 bits. The variable contains 4 message blocks for the 4 different errors of the motor. The meaning of these errors is explained below.

Control error

The Control Error occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time. Whether the relay or the technological object to be controlled is defective in such a situation can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The Not Controllable error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the *On* command in the *Disabled* mode causes this error display.

Protection Fault

The *Protection Fault* indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The *Over_Temperature* error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Commands

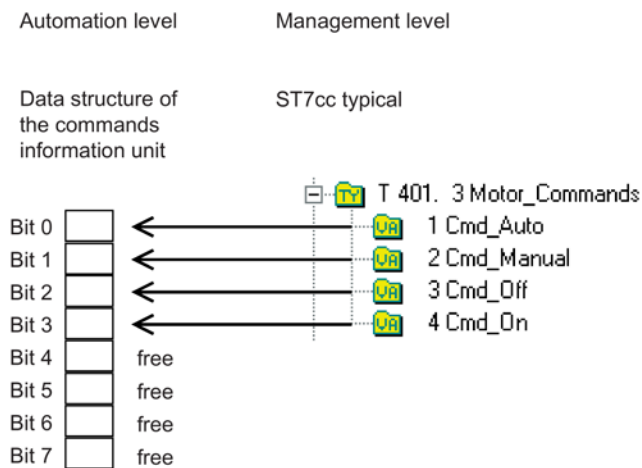


Figure 9-21 Commands of a motor

This coding template contains the commands for a motor. The typical contains a variable for each of the 4 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_Off

The *Cmd_Off* command turns off the technological object.

Cmd_On

The *Cmd_On* command turns on the technological object.

9.4.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical.



Figure 9-22 Minimum display

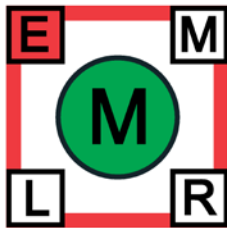


Figure 9-23 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol M on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol M means that the bits for manual and automatic mode are both set to 1 or 0 indicating an error. The same thing applies to the object symbol for the status information On and Off.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

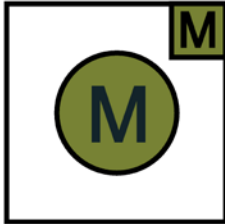





Figure 9-24 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = On
	Light gray background	Operation State = Off
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
		Error, unacknowledged
	Flashing red	
		Error, acknowledged
Red constant		
	No error, unacknowledged (error occurred and was corrected before it was acknowledged)	
Flashing red		

Note on the no error, unacknowledged status:

If a motor, for example, has the overtemperature error, it is normally turned off by the automatic functions. The motor cools down, the overtemperature error is cleared and the

motor can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.4.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Motor1) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

Motor1		
Operation Mode :	Automatic Manual	
Operation State :	Off On	
Object State :	Revision Local Disabled	
Errors :	Control Error Not Controlable Protection Fault Over_Temperature	
<input checked="" type="radio"/> Automatic	Acknowledge	
<input type="radio"/> Manual		
<input type="radio"/> Off		
<input type="radio"/> On		
OK	Apply	Cancel

Figure 9-25 Motor1 faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.

Clicking on *Apply* executes the selected command without closing the faceplate.

Clicking on *Cancel* closes the faceplate without executing the selected command.

9.5 Templates for the generator technological object

This section describes the components of the engineering template for the generator technological object.

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.5.1 ST7cc typicals

Status messages

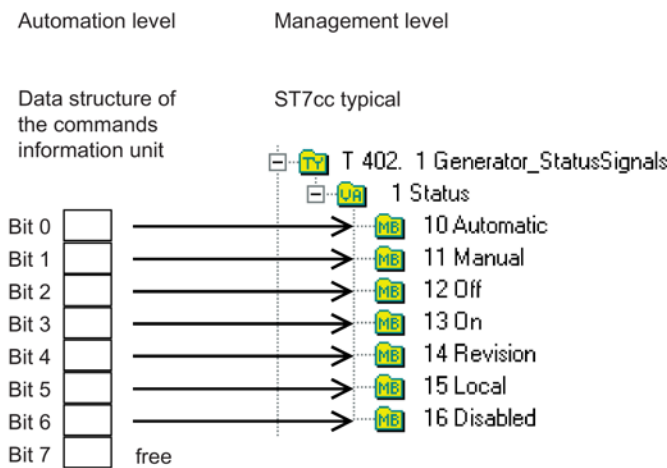


Figure 9-26 Status messages of a generator

This decoding template contains the status messages for a generator. The typical contains a status variable with a length of 8 bits. The variable contains 7 message blocks for the 7

different status messages of the generator. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Off

The technological object is turned off.

On

The technological object is turned on.

Revision

The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The Local mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

 WARNING
--

If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages (errors)

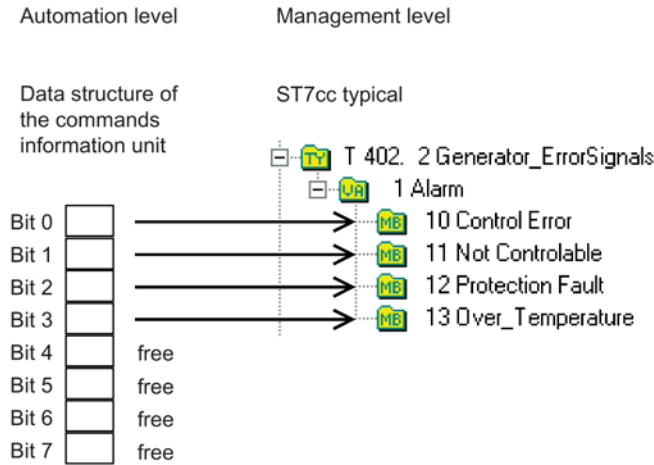


Figure 9-27 Alarm messages of a generator

This decoding template contains the alarm messages for a generator. The typical contains an alarm variable with a length of 8 bits. The variable contains 4 message blocks for the 4 different errors of the generator. The meaning of these errors is explained below.

Control error

The Control Error occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time. Whether the relay or the technological object to be controlled is defective in such a situation can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The Not Controllable error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the On command in the Disabled mode causes this error display.

Protection Fault

The Protection Fault indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The Over_Temperature error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Commands

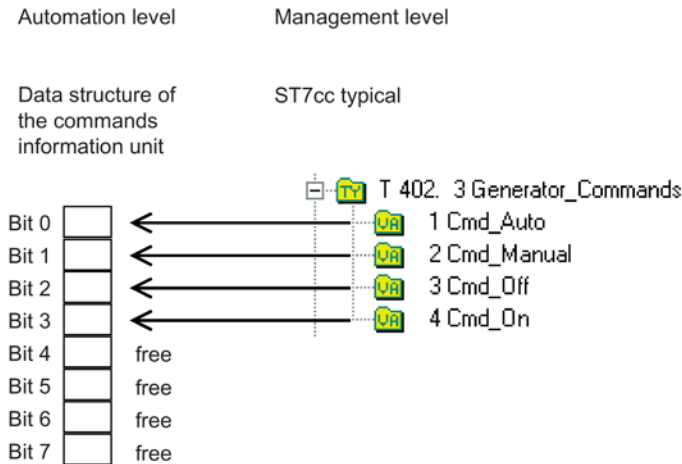


Figure 9-28 Commands for a generator

This coding template contains the commands for a generator. The typical contains a variable for each of the 4 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_Off

The *Cmd_Off* command turns off the technological object.

Cmd_On

The *Cmd_On* command turns on the technological object.

9.5.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical.



Figure 9-29 Minimum display

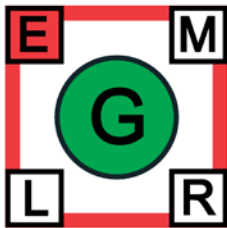


Figure 9-30 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol *M* on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol *M* means that the bits for manual and automatic mode are both set to *1* or *0* indicating an error. The same thing applies to the object symbol for the status information *On* and *Off*.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

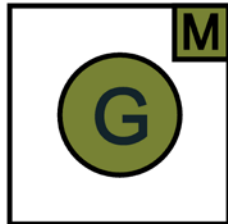





Figure 9-31 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = On
	Light gray background	Operation State = Off
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
	 Flashing red	Error, unacknowledged
	 Red constant	Error, acknowledged
	 Flashing red	No error, unacknowledged (error occurred and was corrected before it was acknowledged)

Note on the no error, unacknowledged status:

If a generator, for example, has the overtemperature error, it is normally turned off by the automatic functions. The generator cools down, the overtemperature error is cleared and the

generator can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.5.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Generator) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

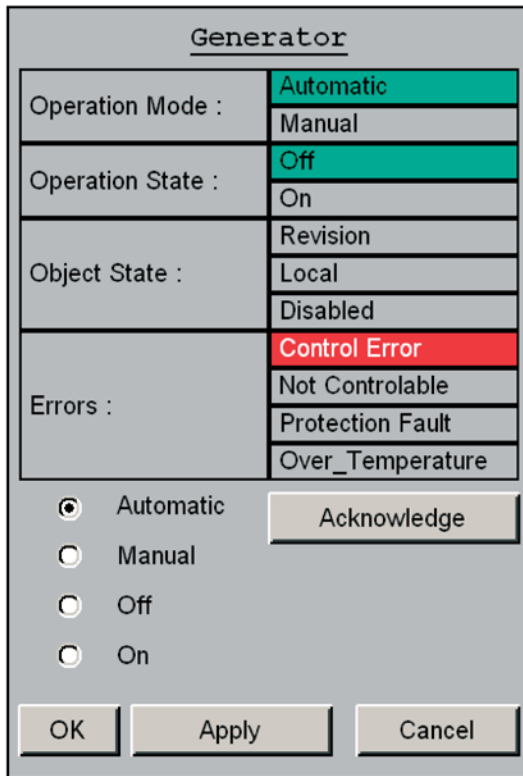


Figure 9-32 Generator faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.
 Clicking on *Apply* executes the selected command without closing the faceplate.
 Clicking on *Cancel* closes the faceplate without executing the selected command.

9.6 Templates for the valve technological object

This section describes the components of the engineering template for the valve technological object.

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.6.1 ST7cc typicals

Status messages

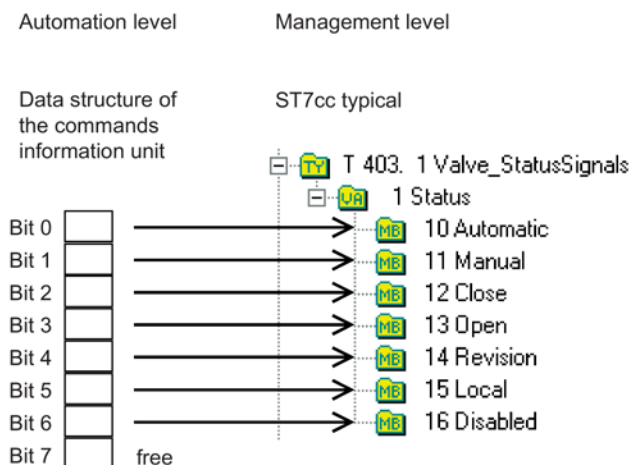


Figure 9-33 Status messages of a valve

This decoding template contains the status messages for a valve. The typical contains a status variable with a length of 8 bits. The variable contains 7 message blocks for the 7 different status messages of the valve. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Closed

The technological object is turned closed.

Open

The technological object is open.

Revision

The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The Local mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

 **WARNING**

If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages (errors)

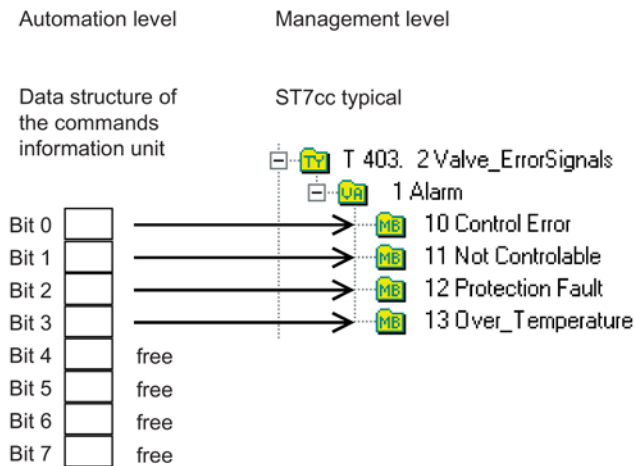


Figure 9-34 Alarm messages of a valve

This decoding template contains the error messages for a valve. The typical contains an alarm variable with a length of 8 bits. The variable contains 4 message blocks for the 4 different errors of the valve. The meaning of these errors is explained below.

Control error

The Control Error occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time. Whether the relay or the technological object to be controlled is defective in such a situation can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The Not Controllable error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the Open command in the Disabled mode causes this error display.

Protection Fault

The Protection Fault indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The Over_Temperature error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Commands

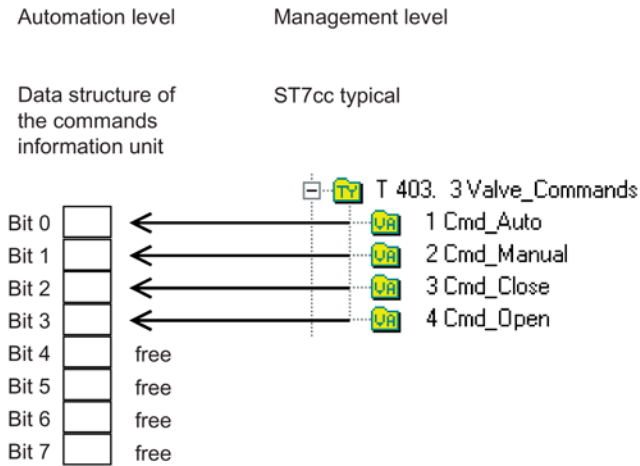


Figure 9-35 Commands of a valve

This coding template contains the commands for a valve. The typical contains a variable for each of the 4 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_Close

The *Cmd_Close* command closes the technological object.

Cmd_Open

The *Cmd_Open* command opens the technological object.

9.6.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical. 2 variants of this picture typical are supplied to allow the flow direction to be displayed. The variant shown in the figure shows the vertical flow direction.



Figure 9-36 Minimum display



Figure 9-37 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol M on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol M means that the bits for manual and automatic mode are both set to 1 or 0 indicating an error. The same thing applies to the object symbol for the status information Open and Closed.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

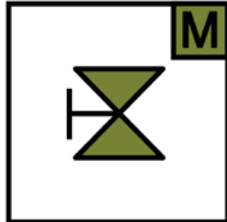


Figure 9-38 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = Open
	Light gray background	Operation State = Closed
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
		Error, unacknowledged
	Flashing red	
		Error, acknowledged
	Red constant	
		No error, unacknowledged (error occurred and was corrected before it was acknowledged)
	Flashing red	

Note on the no error, unacknowledged status:

If a valve, for example, has the overtemperature error, it is normally turned off by the automatic functions. The valve cools down, the overtemperature error is cleared and the

valve can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.6.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Valve) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Closed).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

Valve					
Operation Mode :	<table border="1"> <tr><td>Automatic</td></tr> <tr><td>Manual</td></tr> </table>	Automatic	Manual		
Automatic					
Manual					
Operation State :	<table border="1"> <tr><td>Close</td></tr> <tr><td>Open</td></tr> </table>	Close	Open		
Close					
Open					
Object State :	<table border="1"> <tr><td>Revision</td></tr> <tr><td>Local</td></tr> <tr><td>Disabled</td></tr> </table>	Revision	Local	Disabled	
Revision					
Local					
Disabled					
Errors :	<table border="1"> <tr><td>Control Error</td></tr> <tr><td>Not Controlable</td></tr> <tr><td>Protection Fault</td></tr> <tr><td>Over_Temperature</td></tr> </table>	Control Error	Not Controlable	Protection Fault	Over_Temperature
Control Error					
Not Controlable					
Protection Fault					
Over_Temperature					
<input checked="" type="radio"/> Automatic <input type="radio"/> Manual <input type="radio"/> Close <input type="radio"/> Open	<input type="button" value="Acknowledge"/>				
<input type="button" value="OK"/> <input type="button" value="Apply"/> <input type="button" value="Cancel"/>					

Figure 9-39 Valve faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.
 Clicking on *Apply* executes the selected command without closing the faceplate.
 Clicking on *Cancel* closes the faceplate without executing the selected command.

9.7 Templates for the compressor technological object

This section describes the components of the engineering template for the compressor technological object.

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.7.1 ST7cc typicals

Status messages

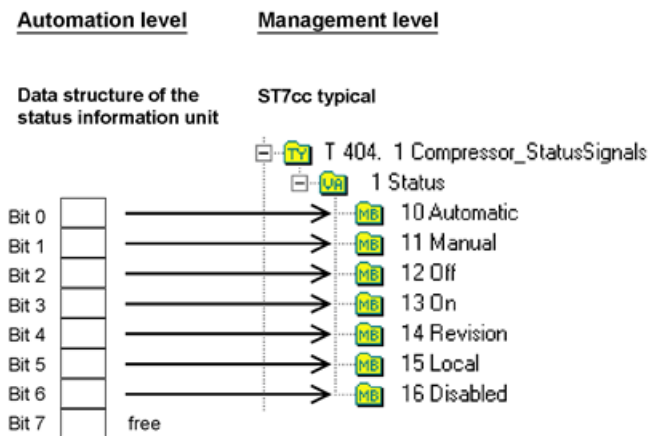


Figure 9-40 Status messages of a compressor

This decoding template contains the status messages for a compressor. The typical contains a status variable with a length of 8 bits. The variable contains 7 message blocks for the 7 different status messages of the compressor. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Off

The technological object is turned off.

On

The technological object is turned on.

Revision

The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The Local mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

 **WARNING**

If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages (errors)

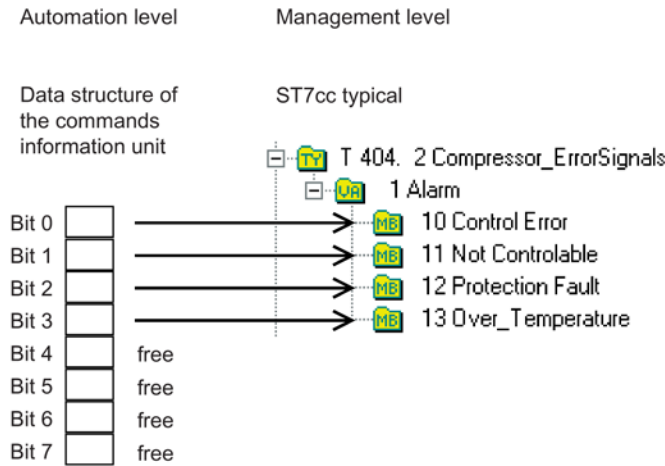


Figure 9-41 Alarm messages of a compressor

This decoding template contains the alarm messages for a compressor. The typical contains an alarm variable with a length of 8 bits. The variable contains 4 message blocks for the 4 different errors of the compressor. The meaning of these errors is explained below.

Control error

The Control Error occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time. Whether the relay or the technological object to be controlled is defective in such a situation can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The Not Controllable error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the On command in the Disabled mode causes this error display.

Protection Fault

The Protection Fault indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The Over_Temperature error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Commands

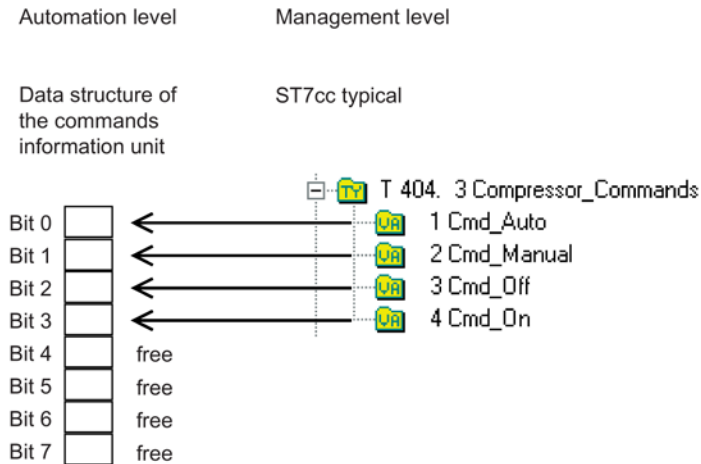


Figure 9-42 Commands of a compressor

This coding template contains the commands for a compressor. The typical contains a variable for each of the 4 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_Off

The *Cmd_Off* command turns off the technological object.

Cmd_On

The *Cmd_On* command turns on the technological object.

9.7.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical. 4 variants of this picture typical are supplied to allow the conveyor direction to be displayed. The variant shown in the figure shows the conveyor direction left.

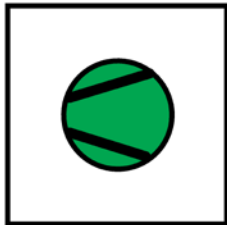


Figure 9-43 Minimum display

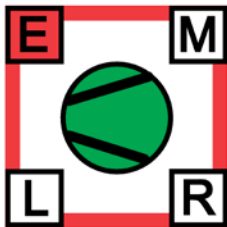


Figure 9-44 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol M on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol M means that the bits for manual and automatic mode are both set to 1 or 0 indicating an error. The same thing applies to the object symbol for the status information On and Off.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

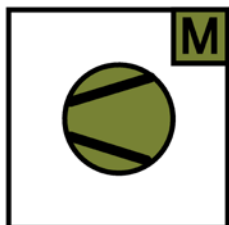





Figure 9-45 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = On
	Light gray background	Operation State = Off
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
	 Flashing red	Error, unacknowledged
	 Red constant	Error, acknowledged
	 Flashing red	No error, unacknowledged (error occurred and was corrected before it was acknowledged)

Note on the no error, unacknowledged status:

If a compressor, for example, has the overtemperature error, it is normally turned off by the automatic functions. The compressor cools down, the overtemperature error is cleared and

the compressor can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.7.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Compressor) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

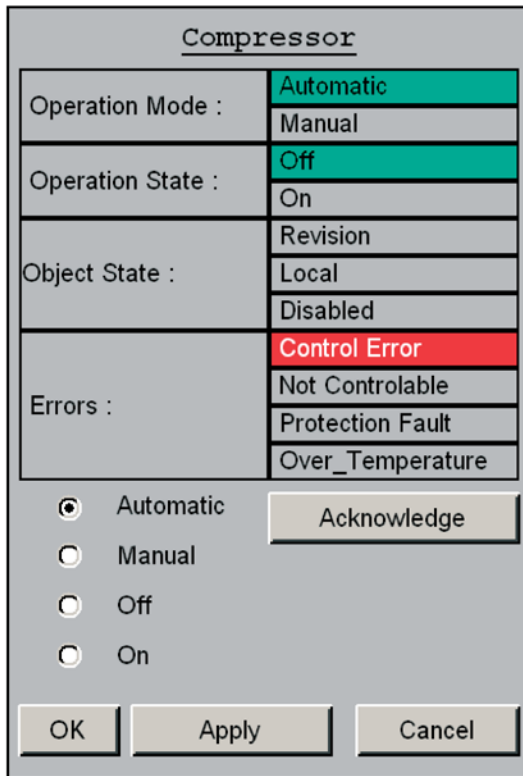


Figure 9-46 Compressor faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.
Clicking on *Apply* executes the selected command without closing the faceplate.
Clicking on *Cancel* closes the faceplate without executing the selected command.

9.8 Templates for the Motor2 technological object

This section describes the components of the engineering template for the Motor2 technological object (motor with two forwards and two reverse gears).

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.8.1 ST7cc typicals

Status messages

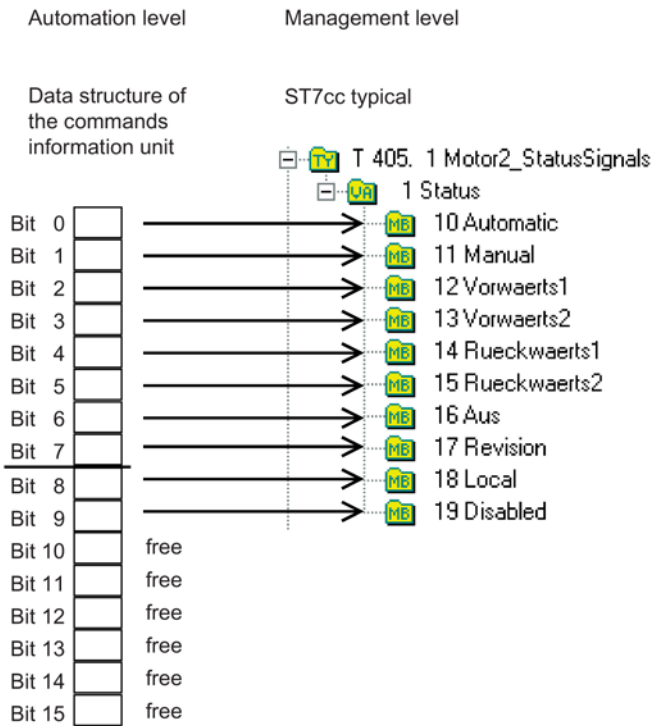


Figure 9-47 Motor2 status messages

This decoding template contains the status messages for a motor with two forwards and two reverse gears. The typical contains a status variable with a length of 16 bits. The variable contains 10 message blocks for the 10 different status messages of the motor. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Forwards1

The technological object is turned on and is running in the 1st forwards gear.

Forwards2

The technological object is turned on and is running in the 2nd forwards gear.

Reverse1

The technological object is turned on and is running in the 1st reverse gear.

Reverse2

The technological object is turned on and is running in the 2nd reverse gear.

Off

The technological object is turned off.

Revision

The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The Local mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

 WARNING

If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages (errors)

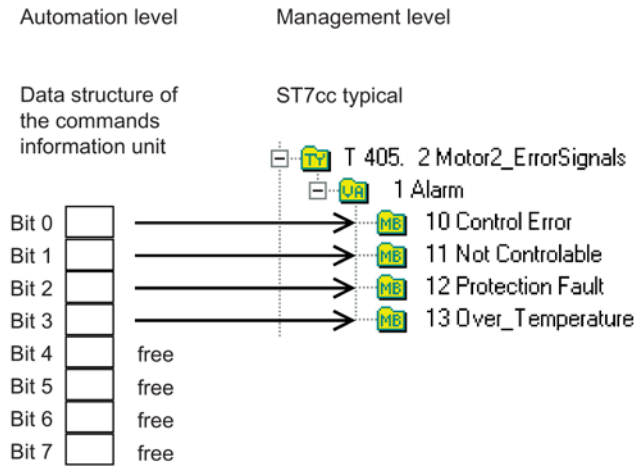


Figure 9-48 Alarm messages of Motor2

This decoding template contains the error messages for a motor with two forwards and two reverse gears. The typical contains an alarm variable with a length of 8 bits. The variable contains 4 message blocks for the 4 different errors of the motor. The meaning of these errors is explained below.

Control error

The Control Error occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time. Whether the relay or the technological object to be controlled is defective in such a situation can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The Not Controllable error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the F1On command in the Disabled mode causes this error display.

Protection Fault

The Protection Fault indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The Over_Temperature error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Commands

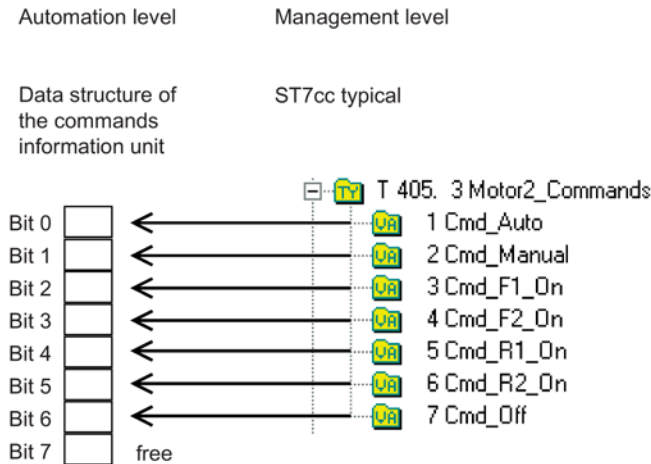


Figure 9-49 Commands of Motor2

This coding template contains the commands for a motor. The typical contains a variable for each of the 7 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_F1_On

The technological object is switched to the 1st forwards gear.

Cmd_F2_On

The technological object is switched to the 2nd forwards gear.

Cmd_R1_On

The technological object is switched to the 1st reverse gear.

Cmd_R2_On

The technological object is switched to the 2nd reverse gear.

Cmd_Off

The *Cmd_Off* command turns off the technological object.

9.8.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical.

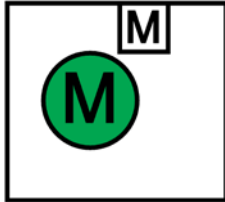


Figure 9-50 Minimum display

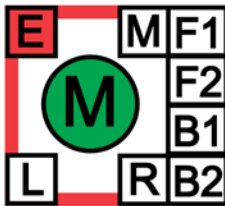


Figure 9-51 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol M on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol M means that the bits for manual and automatic mode are both set to 1 or 0 indicating an error. The same applies analogously to the status information Forwards1, Forwards2, Reverse1, Reverse2 and Off.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

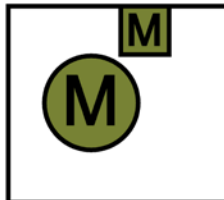





Figure 9-52 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = On
	Light gray background	Operation State = Off
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol F1	Visible	Operation State = Forwards1
	Not visible	Operation State ≠ Forwards1
Symbol F2	Visible	Operation State = Forwards2
	Not visible	Operation State ≠ Forwards2
Symbol R1	Visible	Operation State = Reverse1
	Not visible	Operation State ≠ Reverse1
Symbol R2	Visible	Operation State = Reverse2
	Not visible	Operation State ≠ Reverse2
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
	 Flashing red	Error, unacknowledged
	 Red constant	Error, acknowledged
	 Flashing red	No error, unacknowledged (error occurred and was corrected before it was acknowledged)

Note on the no error, unacknowledged status:

If a motor, for example, has the overtemperature error, it is normally turned off by the automatic functions. The motor cools down, the overtemperature error is cleared and the motor can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.8.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Motor2) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

Motor2		
Operation Mode :	Automatic	
	Manual	
Operation State :	Forward 1	
	Forward 2	
	Backward 1	
	Backward 2	
	Off	
Object State:	Revision	
	Local	
	Disabled	
Errors :	Control Error	
	Not Controlable	
	Protection Fault	
	Over_Temperature	
<input checked="" type="radio"/>	Automatic	Acknowledge
<input type="radio"/>	Manual	
<input type="radio"/>	Forward 1	
<input type="radio"/>	Forward 2	
<input type="radio"/>	Backward 1	
<input type="radio"/>	Backward 2	
<input type="radio"/>	Off	
OK Apply Cancel		

Figure 9-53 Motor2 faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.
 Clicking on *Apply* executes the selected command without closing the faceplate.
 Clicking on *Cancel* closes the faceplate without executing the selected command.

9.9 Templates for the slider valve technological object

This section describes the components of the engineering template for the slider valve technological object.

Note

Please note that the components of the engineering templates do not include any control logic that prevents implausible switch settings or status displays. The templates are designed to display the statuses of a technological object and to output commands to a technological object. The necessary plausibility checks and interlocks must be included in the automation software created by the user.

9.9.1 ST7cc typicals

Status messages

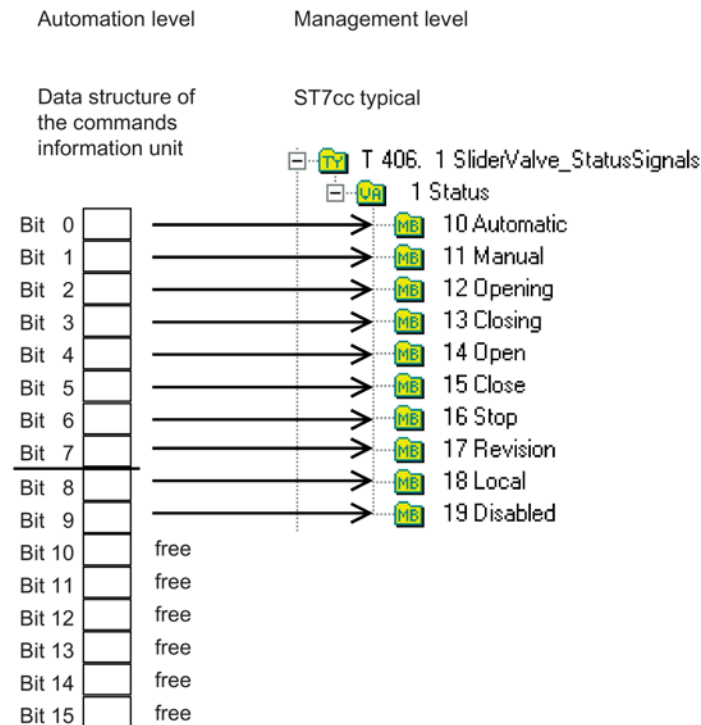


Figure 9-54 Status messages of a slider valve

This decoding template contains the status messages for a slider valve. The typical contains a status variable with a length of 16 bits. The variable contains 10 message blocks for the 10 different status messages of the slider valve. The meaning of these status messages is explained below.

Automatic

The object is controlled automatically by a user program / algorithm on the CPU. It is activated or deactivated depending on the control logic programmed in the automation software.

Manual

The technological object is controlled by input by the operator. Automatic functions (algorithm) that may exist, cannot control the technological object in this case.

Opening

The technological object is opening.

Closing

The technological object is closing.

Open

The technological object is open.

Closed

The technological object is closed.

Stop

The technological object is stopped at its current position.

Revision

The revision mode indicates that the object is set to a special mode for maintenance or servicing. This mode can only be set by an operator on-site. During a revision, the object can be turned on and off by maintenance personnel and can be checked for protection violations. Maintenance personnel can also deliberately cause an error for test purposes. The Revision status indicates to the operator in the control center that the personnel responsible is performing maintenance or test activities locally. The running of the technological process takes into account this work. As a result, the operator does not need to keep track of the statuses of the object during the revision.

In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

Local

The Local mode means that the object is controlled by an operator on-site from the switching cabinet. Turning the object on or off does, however, have an influence on the technological process. This means that the local operator is responsible for the running of the process. In general, the automation software rejects remote manual input and automatic commands when in this status. This leads to the Not Controllable error.

WARNING

If problems occur in Local mode, these have not been caused deliberately for test purposes as in Revision. The operator must therefore react immediately.

Disabled

The Disabled mode means that the existing object cannot be used temporarily or that it exists in the configuration but is not yet physically ready for use. This mode can be set by configuration or by local operator action.

Generally, the automation software rejects all input and commands in this mode. This leads to the Not Controllable error.

Alarm messages (errors)

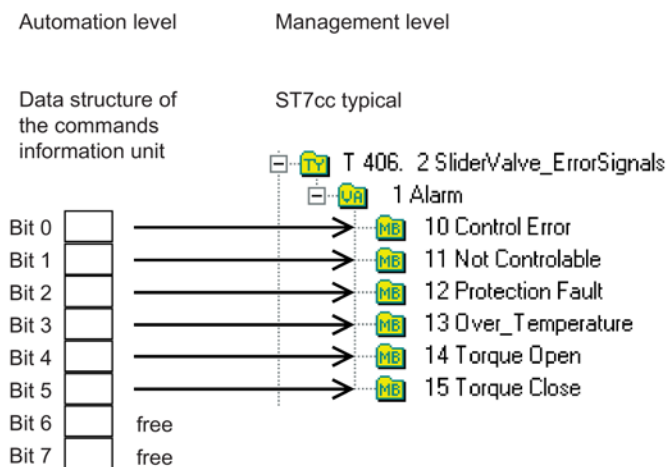


Figure 9-55 Alarm messages of a slider valve

This decoding template contains the alarm messages for a slider valve. The typical contains an *Alarm* variable with a length of 8 bits. The variable contains 6 message blocks for the 6 different errors of the slider valve. The meaning of these errors is explained below.

Control error

The Control Error occurs when a command is output to the technological object by the automation software and there is no reaction within a configurable time.

Whether the relay or the technological object to be controlled is defective in such a situation

can only be decided by additional plausibility checks and error displays not included in the supplied typicals.

Not Controllable

The Not Controllable error indicates that a control instruction can be executed due to rules specified in the automation software. Normally, for example, the Cmd_Opening command in the Disabled mode causes this error display.

Protection Fault

The Protection Fault indicates that a protective mechanism has been violated or has failed. The technological object normally turns itself off. This is implemented in the automation software.

Over_Temperature

The Over_Temperature error indicates that the temperature monitoring of the object has responded. The technological object normally turns itself off. This is implemented in the automation software.

Torque Open

The slider valve is stuck due to contamination, blockage or similar and cannot react to the *Cmd_Opening* command.

Torque Close

The slider valve is stuck due to contamination, blockage or similar and cannot react to the *Cmd_Closing* command.

Commands

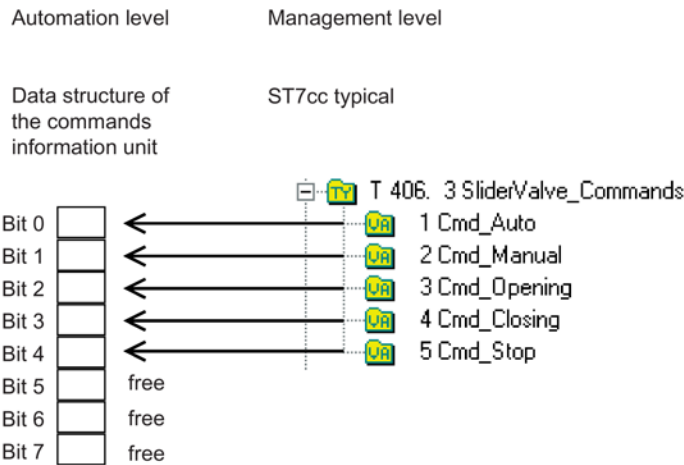


Figure 9-56 Commands of a slider valve

This coding template contains the commands for a slider valve. The typical contains a variable for each of the 5 commands each with a length of 1 bit.

Cmd_Auto

With the *Cmd_Auto* command, the user enables automatic mode for the automation software.

Cmd_Manual

The *Cmd_Manual* command disables the object for automatic mode. The object is controlled solely by operator input.

Cmd_Opening

The *Cmd_Opening* command sets the slider valve to the *Opening* state.

Cmd_Closing

The *Cmd_Closing* command sets the slider valve to the *Closing* state.

Cmd_Stop

The *Cmd_Stop* command stops the *Opening* or *Closing* states. In general, a slider valve movement is brought about by a flow control. This means that the three commands *Opening*, *Closing* and *Stop* can set a variable flow.

9.9.2 Corresponding picture typical

The figure shows the minimum display and the total display of the picture typical. This display was selected to show all the symbols that make up the picture typical. 2 variants of this picture typical are supplied to allow the flow direction to be displayed. The variant shown in the figure shows the vertical flow direction.



Figure 9-57 Minimum display



Figure 9-58 Total display

The meaning and dynamics of the symbols in the picture typical are explained below.

Plausibility checks

To be able to indicate incorrect configurations and bad links to the user, simple plausibility checks are integrated in the picture typical.

The figure shows the object symbol and the symbol M on a dark yellow background. This indicates that WinCC has no or incomplete data as far as this can be determined by simple plausibility checks.

The dark yellow display of the symbol M means that the bits for manual and automatic mode are both set to 1 or 0 indicating an error. The same applies analogously to the status information Opening, Closing, Open, Closed and Stop.

The implausible bit settings can be caused by the following:

1. WinCC has just started up and has not yet been supplied with process data.
2. An object name was assigned to the picture typical that does not match the group name of the typical instance.
3. The user program at the automation level is not supplying the ST7 objects correctly.

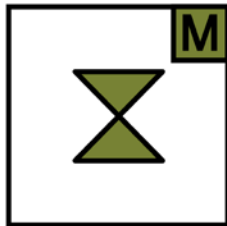





Figure 9-59 Display when data is missing or incorrect

Symbol	Appearance	Meaning
Object symbol	Green background	Operation State = Open
	Light gray background	Operation State = Closed
	Flashing green	Operation State = Opening
	Flashing red	Operation State = Closing
	Gray background	Operation State = Stop
Symbol M	Visible	Operation Mode = Manual
	Not visible	Operation Mode = Automatic
Symbol R	Visible	Object State = Revision
	Not visible	Object State ≠ Revision
Symbol L	Visible	Object State = Local
	Not visible	Object State ≠ Local
Frame bold and red	Visible	Object State = Disabled
	Not visible	Object State ≠ Disabled
Symbol E (group error)	Not visible	No error
	 Flashing red	Error, unacknowledged
	 Red constant	Error, acknowledged
	 Flashing red	No error, unacknowledged (error occurred and was corrected before it was acknowledged)

Note on the no error, unacknowledged status:

If a slider valve, for example, has the overtemperature error, it is normally turned off by the automatic functions. The slider valve cools down, the overtemperature error is cleared and the slider valve can be turned on again. The operator must therefore acknowledge that there had been an error.

Note on the E symbol (group error):

To avoid overloading the picture typical with too much detailed information, all possible errors are simply indicated in the picture typical by E. The detailed information is indicated to the user in the faceplate.

9.9.3 Corresponding faceplate

To establish an address relationship to the WinCC tags, the faceplate automatically adopts the object name (in the figure Slider) from the object name of the picture typical you have clicked on.

Beside operation mode, operation state, and object state, active statuses are indicated by a green background (in the figure Automatic and Off).

An active error (disturbance) is indicated by a white font on a red background (in the figure Control Error). The commands are implemented as option buttons arranged one above the other.

The displays of implausible states resulting from incorrect configuration or lack of data supply following a WinCC restart are described in section Faceplates in WinCC (Page 344).

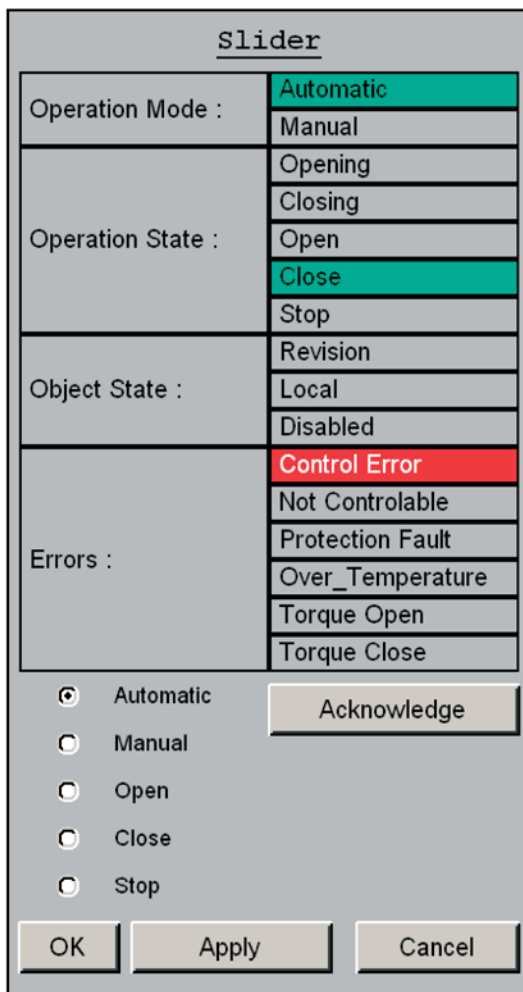


Figure 9-60 Slider faceplate

Clicking on *Acknowledge* acknowledges an error.

Clicking on *OK* closes the faceplate; at the same time the selected command is executed.

Clicking on *Apply* executes the selected command without closing the faceplate.

Clicking on *Cancel* closes the faceplate without executing the selected command.

Glossary

Channel DLL

To allow WinCC to communicate with the widest variety of data sources (programmable controllers, ST7cc servers etc.), various communications drivers are used.

A communications driver is a C++ DLL that communicates with the data manager over an interface known as the channel API that is specified by the data manager. The WinCC tags are supplied with process values via the communications driver.

Data manager

The accrued data is managed by the data manager in WinCC. The data manager works with data created in the WinCC project and data stored in the project database. It handles the entire management of WinCC tags while WinCC is in runtime mode. All WinCC applications request the data from the data manager in the form of WinCC tags. These applications include Graphics Runtime, Alarm Logging Runtime and Tag Logging Runtime.

To allow WinCC to communicate with the widest variety of data sources (programmable controllers, ST7cc servers etc.), various communications drivers are used.

A communications driver is a C++ DLL that communicates with the data manager over a specified interface known as the channel API. The WinCC tags are supplied with process values via the communications driver.

Global Script (Runtime)

The term Global Script means all the C functions and actions that can be used throughout the project or over several projects. C actions are used in process control during runtime.

Graphics Designer

The Graphics Designer is a vector-oriented drawing program for creating process pictures. Complex process pictures can be created using a wide variety of graphic objects from an object and style palette.

ST7cc Config uses the ODK interface of the Graphics Designer to create picture typicals for the SINAUT subscribers and to make these available to the WinCC configuration engineer in a sample picture for use in further process pictures.

Local buffer

If the ST7cc server cannot forward its data to WinCC, all messages (ST7 data messages and organizational messages) are stored in the local buffer. Once WinCC becomes available again, the buffered messages are processed. This mechanism achieves two aims:

- That the master station is accessible from the perspective of the stations even when WinCC is not available.
- That general requests as a result of temporary deactivation of WinCC can be avoided.

Message decoding, message encoding

Message decoding has the task of mapping the data of a received SINAUT message to ST7cc variables (monitoring direction). The basis for this mapping is the decoding created for each SINAUT object with ST7sc Config. In the control/command direction, the content of an ST7sc variable is entered in a SINAUT message. This is known as message encoding. To simplify matters, only the term message decoding is used regardless of the transmission direction.

Remote buffer

The remote buffer is set up only for redundant ST7cc. The ST7cc server recognizes whether or not the redundant mode is required based on the existence of the redundancy license.

The remote buffer is organized as a ring buffer and records all incoming messages so that it can be used as a data source for the redundant partner during a restart. If the partner of a redundant ST7cc system starts up again, it can recognize the time for which messages are missing and can request these from the redundancy partner. The remote buffer is necessary to ensure data consistency when using a redundant ST7cc system.

SINAUT object

A SINAUT object contains the data of one or more process variables, such as analog values, commands, calculated values, status information on motors, sliders etc. On the station, type-specific processing and change checks are assigned to an ST7 object to minimize the communication traffic in the WAN. Type-specific processing includes, for example, the threshold value check or calculating a mean value with the object type for analog values Ana04W. The change check is designed so that a message is generated only when the object data has changed compared with the last time its value was transferred or when the type-specific processing enables generation of a message because the object data is "worth" transferring.

ST7 message

ST7 messages consist of a frame, an area for address and control fields (message header) and an area for net data (object data) with the time stamp. The ST7 messages are divided into organizational and user messages. For the ST7cc configuration engineer, only the user messages are relevant since these contain the ST7 object data (complete or a subarea) in their net data area.

The mapping of the most important status information of the ST7cc subscribers is a system function. The configuration engineer does not therefore need prior knowledge of the structure of organizational messages.

ST7 frame header:

The ST7 message header contains the address and control fields for the ST7 protocol. The control fields include the time status bits indicating whether correct time information is present and whether the time information is standard or daylight saving time etc.

ST7 object data:

In its net data area, an ST7 user message contains the object data or a contiguous subarea of the object data. In either case, a time stamp is also included in the object data area. The time stamp corresponds to the time at which the event was created. The event creation is a function of ST7 object processing.

ST7cc variable

An ST7cc variable is a data section from the data area of a SINAUT object that is managed and processed as a separate information unit on the ST7cc server. The variables are processed, however, in ST7cc and in WinCC. When the variables are defined, different processing functions can be assigned to them depending on their type. A variable can contain both a process value as well as status information from system components. System components are the SINAUT subscribers.

ST7cc variable management

ST7cc variable management covers all ST7cc variables. The content of the ST7cc variables represents the current process image. WinCC writes and reads the ST7cc variable.

Tag Logging (Runtime)

Alarm logging controls the acquisition and archiving of events and provides display and operator input options. Using the message blocks, message class and message type structure elements, the configuration engineer can class the events according to their significance and allow the operator a fast evaluation of the status of the system.

Notes on the interaction between ST7cc and Alarm Logging:

ST7cc server:

The ST7cc server allows two types of interaction with Alarm Logging.

1. The ST7cc server checks, among other things, whether a message should be generated for the incoming process information. If this is the case, the ST7cc server transfers a message job to Alarm Logging. Alarm Logging puts together the actual individual message and is responsible for displaying and archiving the message. With this strategy, up to WinCC version 5.0, it is guaranteed that the event-related time stamp supplied by ST7 was included in the WinCC messages. This strategy is also possible with the WinCC versions > V5.0, but has the disadvantage that when using a redundant WinCC system (WinCC Redundancy), the operator must acknowledge a message on each redundancy partner.
2. As of WinCC version V5.1, the WinCC data manager can generate the message job and accept and process the event-related time stamp supplied by ST7cc. This means that the messages are generated entirely in WinCC. The advantage of this strategy is that when

using a redundant WinCC system, an event can be acknowledged with system support on both redundancy partners (acknowledgment consistency).

ST7cc Config:

ST7cc Config allows the parameter values required for message processing to be specified. When the WinCC data framework is generated (in other words, message management), the individual WinCC messages are generated and imported into WinCC.

Tag Logging is used to receive data from active processes and to prepare it for display and archiving. The data formats of the archives and the acquisition times and archiving times can be freely selected.

WinCC Tag Logging is computer time-oriented and not intended for the arrival of data with a delay as is the case with SINAUT ST7. This means that the ST7cc server must make certain archiving preparations for WinCC. The ST7cc server transfers the values to be archived to Tag Logging via the ODK interface. This ensures the chronological arrangement of the archive values even if process data is delivered by the ST7 stations, for example, with an offset of an hour.

Tag Logging (Runtime)

Alarm logging controls the acquisition and archiving of events and provides display and operator input options. Using the message blocks, message class and message type structure elements, the configuration engineer can class the events according to their significance and allow the operator a fast evaluation of the status of the system.

Notes on the interaction between ST7cc and Alarm Logging:

ST7cc server:

The ST7cc server allows two types of interaction with Alarm Logging.

1. The ST7cc server checks, among other things, whether a message should be generated for the incoming process information. If this is the case, the ST7cc server transfers a message job to Alarm Logging. Alarm Logging puts together the actual individual message and is responsible for displaying and archiving the message. With this strategy, up to WinCC version 5.0, it is guaranteed that the event-related time stamp supplied by ST7 was included in the WinCC messages. This strategy is also possible with the WinCC versions > V5.0, but has the disadvantage that when using a redundant WinCC system (WinCC Redundancy), the operator must acknowledge a message on each redundancy partner.
2. As of WinCC version V5.1, the WinCC data manager can generate the message job and accept and process the event-related time stamp supplied by ST7cc. This means that the messages are generated entirely in WinCC. The advantage of this strategy is that when using a redundant WinCC system, an event can be acknowledged with system support on both redundancy partners (acknowledgment consistency).

ST7cc Config:

ST7cc Config allows the parameter values required for message processing to be specified. When the WinCC data framework is generated (in other words, message management), the individual WinCC messages are generated and imported into WinCC.

Tag Logging is used to receive data from active processes and to prepare it for display and archiving. The data formats of the archives and the acquisition times and archiving times can be freely selected.

WinCC Tag Logging is computer time-oriented and not intended for the arrival of data with a delay as is the case with SINAUT ST7. This means that the ST7cc server must make certain archiving preparations for WinCC. The ST7cc server transfers the values to be archived to Tag Logging via the ODK interface. This ensures the chronological arrangement of the archive values even if process data is delivered by the ST7 stations, for example, with an offset of an hour.

TCO (TIM Connect)

The TCO component monitors the local TIMs connected over MPI or Ethernet, maps their most important status displays to ST7sc variables, forwards received messages for 'message decoding' or transfers the messages to be sent to the relevant TIM for WAN communication.

WinCC API

As a completely open and expandable system, WinCC makes a comprehensive API (Application Program Interface) available. This involves an interface over which the user programs such as ST7cc Server and ST7cc Config can access WinCC. A comprehensive description is available with the WinCC ODK (Open Developers Kit).

WinCC buffer

Message and archive processing can be assigned to the ST7cc variables. If this is the case, individual messages or archive data are generated in ST7cc, that are transferred over the ODK interface to Alarm Logging or Tag Logging for further processing.

The results of processing an ST7cc variable can, however, accrue faster than they can be accepted by WinCC. The WinCC buffer takes the WinCC jobs from the ST7cc processing and therefore separates the asynchronous procedures of job creation and job processing.

WinCC tag

WinCC tags are the central elements that allow process values to be accessed. Within a WinCC project, they have a unique name and a data type. A WinCC tag is assigned to a logical connection that specifies which channel supplies the process values of the tags and over which connection.

For the WinCC tags whose data sources are the ST7cc variables, the channel DLL is the connection over which the ST7cc server supplies the process values.

The WinCC tags required for ST7cc can be generated with ST7cc Config.

Index

A

- Archive, 256
 - (Archive) variable name, 256
 - Archive name, 256
 - Project settings, 140

C

- Counted value processing
 - Basic processing, 245
 - Counter overflow, 246
 - Interval, 245
- Cross references (PDF), 4

D

- Decoding
 - Copy functions, 216
 - Copying a decoding, 223
 - Creating, 210
 - Creating a variable, 212
 - Decoding with typical, 217
 - Deleting decodings, 226
 - Entering the Variable Definition, 213
 - Using typical, 216

F

- Faceplate
 - Local TIM, 312
 - Server, 315
 - Station, 307
 - Station statistics, 310

G

- Generating for WinCC
 - Archive tags, 268
 - Message management, 266
 - Single variables, 265
 - Tag management, 264
- Glossary, 5

M

- Measured value processing, 250
 - AV2, 253
 - AVG, 252
 - Basic processing, 251
 - MAX, 253
 - MIN, 253
 - MOM, 253
- Message processing, 235
 - Allocation of message blocks, 267
 - Generating messages in ST7cc, 237
 - Generating messages in WinCC, 239
 - Message number, 239
 - Message number format, 144
 - Message numbers, 266
 - Message type, 242
 - Static additional texts, 138, 243
 - WinCC message blocks, 138, 243

O

- Object templates
 - Creating, 183
 - Texts for initializing, 145

P

- Picture typical
 - Local TIM, 311
 - Server, 314
 - Station, 306
- Process Historian, 18
- Processing functions, 231
 - Copying and pasting a processing function, 234, 235
 - Creating a processing function, 233
 - Variable type, 232

S

- Service & Support, 5
- SIMATIC NET glossary, 5
- SINAUT object, 158
 - Decoding, 261

- Object number, 159, 159
- Subscriber number, 159, 159
- TD7 block list, 261
- Variables, 159
- SINAUT object types, 159
 - Ana04W, 160
 - Bin04B, 160
 - Cmd01B, 161
 - Cnt01D, 161
 - Cnt04D, 161
 - Object templates, 169
 - Par12D, 162
 - Set01W, 162
- SINAUT subscriber, 158
 - Application access points, 132
 - CPU, 158
 - CPUs of the stations, 207
 - Local IDs, 132
 - Local TIM, 158
 - Local TIMs, 207
 - Setting up, 208
 - ST7cc server, 207
 - Sub type, 207
 - Subscriber number, 132
- SINAUT TD7 block structure, 259
- Software version, 3
- ST7cc - version, 3
- ST7cc object tree, 181
 - Decoding, 210
 - Decodings, 181
 - Object templates, 181
 - Processing functions, 182
 - SINAUT subscriber, 181
 - Typicals, 181
 - Variables, 182
- ST7cc server, 271
 - Configuration data in the redundant system, 289
 - Exit, 290
 - Local buffer, 273
 - Log server messages, 297
 - Message protocol, 146
 - Process image, 273
 - Remote buffer, 273
 - Restart, 290
 - Startup behavior, 289
 - Startup order, 289
 - TCO (TIM Connect), 273
 - WinCC buffer, 273
 - WinCC redundancy package, 285
- ST7cc variable
 - Attribute names, 164, 165
 - Entering the Variable Definition, 213

- Group name, 164
- Naming convention, 164, 165
- Process variable, 163
- Processing functions, 168
- Sub types, 165
- Subscriber name, 165
- System variable, 164
- Types, 165
- Variable list, 257
- Variable name, 164
- Variable type, 232

T

- Training, 5
- Typical, 170, 199
 - Addressing, 171
 - Creating, 200
 - Instance, 171
 - Local TIM, 197
 - Notes on applications, 205
 - Offset, 171
 - PM_Aqua_channel system typical, 188
 - ServerStatus system typical, 185
 - Station, 193
 - System typical, 170, 184
 - System typical for ST7cc subscribers, 189
 - System typical System, 185
 - Update scenarios for ST7cc system typical, 228
 - User typicals, 170

V

- Variable types, 165
- VMware, 18

W

- WinCC
 - Faceplates, 343
- WinCC tags
 - WinCC quality code, 285