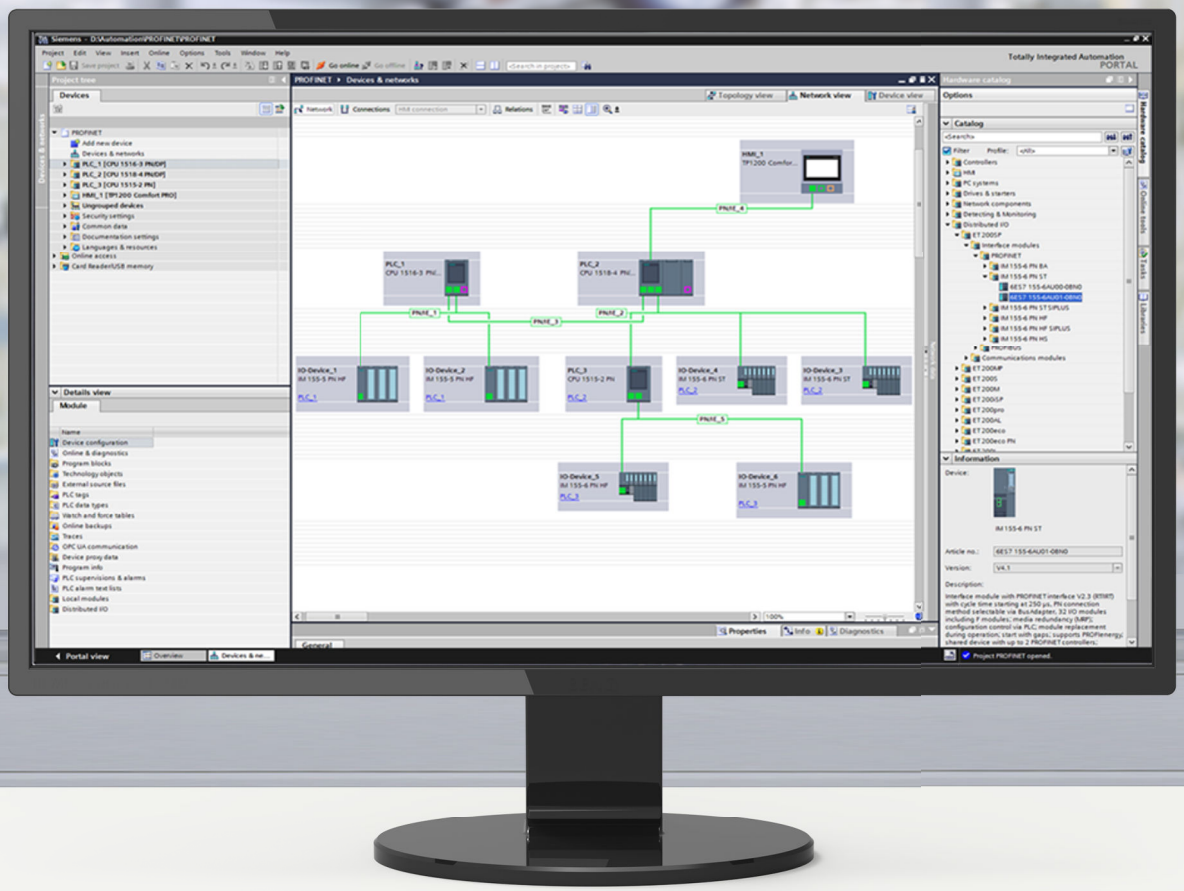
SIEMENS



Edition 11/2022

FUNCTION MANUAL

SIMATIC

S7-1500, ET 200MP, ET 200SP, ET 200AL, ET 200pro

Communication

support.industry.siemens.com

SIEMENS

SIMATIC

S7-1500, ET 200MP, ET 200SP, ET 200AL, ET 200pro, ET 200eco PN Communication

Function Manual

11/2022
A5E03735815-AK

Continued on next page

S7-1500, ET 200MP, ET 200SP, ET 200AL, ET 200pro, ET 200eco PN Communication

Function Manual

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.

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E03735815-AK
Ⓟ 10/2022 Subject to change

Table of contents

Introduction

1

Purpose of the documentation

This function manual provides you with an overview of the communication options, the CPUs, communication modules and processors and PC systems of the systems SIMATIC S7-1500, ET 200MP, ET 200SP, ET 200AL, ET 200pro and SIMATIC Drive Controller. This function manual describes the connection-oriented, asynchronous communication.

The documentation covers the following:

- Overview of the communication services
- Properties of the communication services
- Overview of the user activities for setting up the communication services

Basic knowledge required

The following knowledge is required in order to understand the Function manual:

- General knowledge of automation technology
- Knowledge of the industrial automation system SIMATIC
- Knowledge about how to use STEP 7 (TIA Portal)

Scope of the documentation

This documentation is the basic documentation for all products of the SIMATIC S7-1500, ET 200MP, ET 200SP, ET 200AL and ET 200pro systems. The product documentation is based on this documentation.

What's new in the Communication Function Manual, Edition 11/2022 as compared to Edition 05/2021?

What's new?	What are the customer benefits?	Where can I find the information?
Revision of the tables for communication protocols and port numbers used in Ethernet communication	Updated information on protocols and ports used. You can see at first glance which default settings apply. This allows you to specifically adjust only those settings that are relevant to your application.	Communications protocols and port numbers used for Ethernet communication (Page 30)
Activate / Deactivate the SNMP	Depending on the FW version of the S7-1500 CPUs, the SNMP is activated or deactivated in the default settings. You can change the default settings as required.	SNMP (Page 103)
Revision of the virtual interface for IP-based applications	With a CP 1543-1 with firmware version V3.0 or higher, the internal CP firewall is available. This is used to secure data traffic via the virtual interface.	Virtual interface for IP-based applications (Page 365)

What's new?	What are the customer benefits?	Where can I find the information?
OPC UA server: Reading the diagnostics status of the own address space	By using the OPC UA instruction for reading ("OPC_UA_ReadList"), the own namespace of the OPC UA Server can be accessed. This makes it possible to read out the status of the own OPC UA server as well as the connections of OPC UA clients, the session as well as the subscriptions and to react to them in the user program. This allows connection problems to be quickly detected, for example, and plant availability to be increased.	Running diagnostics for OPC UA servers in the program (Page 294)
OPC UA server: Time stamping of the source time of nodes	By using the OPC UA instruction for writing ("OPC_UA_WriteList"), it is possible to change the "SourceTimestamp" as well as the status code of an OPC UA variable (node). This makes it possible to distinguish between the "Source" and "Server" time as of V18.	Client accesses and local accesses to the OPC UA server (Page 203)
OPC UA GDS mechanism: Now also usable for Web server certificates	The Web server certificate for HTTPS communication can now also be managed via the OPC UA GDS mechanism, without separate download of the hardware configuration.	What you should know about the certificate management (Page 54) Automated certificate management with GDS (Page 180)

What's new in the Communication Function Manual, Edition 05/2021 as compared to Edition 11/2019?

What's new?	What are the customer benefits?	Where can I find the information?
Improved security for SIMATIC PG/HMI communication	• Allows unique identification of each PLC based on individual certificates • Provides additional confidentiality protection through encrypted communication • Protection of the configuration data through individual passwords	Secure PG/HMI communication (Page 93)
Security wizard for new PLC security mechanisms	• Quick and easy configuration of the new security mechanisms of the PLC in one operation • Supporting information to select suitable settings for own application	Protection of confidential configuration data (Page 65)
Certificate management via OPC UA Global Discovery Server (GDS)	• Certificate update during runtime • Support of CRLs • Access protection for certificate management	Certificate management via Global Discovery Server (GDS) (Page 180)
Transferring CPU alarms to OPC UA clients	• Subscriptions allow clients to subscribe to CPU alarms as "Alarms and Conditions" from the OPC UA server of the CPU. • Program messages including associated values are made available by the OPC UA server • Alarms requiring acknowledgement can be acknowledged by the OPC UA client (can be disabled) • An alarm burst is displayed as "overload" and clients can be reloaded with the refresh method	Providing alarms on the OPC UA server (Page 272)

What's new?	What are the customer benefits?	Where can I find the information?
Dynamic assignment of the network configuration with DHCP	Deployment of the CPU in IT managed networks using the following functions: • Connection of the CPU to an existing network without additional manual configuration of the network interface • Request for network parameters for the CPU according to RFC 2131 from a DHCPv4 server (IP address and subnet mask, default IP router address and further optional network parameters such as DNS and NTP server addresses)	Addressing via DHCP (Page 340)
Name-based addressing with DNS	• DNS server addresses can be obtained from the CPU via DHCP • The CPU can obtain host and domain names from a DHCP server for applications that are implemented with OPC UA or (Secure) OUC. • The CPU may transfer configured host or domain names to DHCP servers coupled with DNS servers for dynamic matching (Dynamic DNS). • The NTP client of the CPU can address NTP servers with names • Network parameters can be written with the new "CommConfig" instruction, for example, IP address parameters, DNS server, host and domain name	DHCP with DNS (Page 343)

What's new in the Communication Function Manual, Edition 11/2019 as compared to Edition 10/2018?

What's new?	What are the customer benefits?	Where can I find the information?
IP forwarding	Simple access from the control level to the field level for configuration and parameter assignment of devices, e.g. via PDM or Web browser.	IP forwarding (Page 356)
OPC UA server expansion	For S7-1500 CPUs as of firmware V2.8 and TIA Portal version 16, with a corresponding Runtime license, you can benefit from the following expansions of the integrated OPC UA server: • Improved diagnostics: The OPC UA user receives information on the status of the OPC UA server via alarms in the diagnostics buffer, an OPC UA category in the Online & Diagnostics area of TIA Portal as well as an improved connection resources display. • Download behavior: In RUN mode, the OPC UA server only performs a restart during download from the TIA Portal when the newly downloaded data has an effect on the data management of the OPC UA server. • Server interface modeling: It is now possible in the TIA Portal to model server interfaces or import OPC UA Companion Specifications and map them to the PLC data management.	Section OPC UA communication (Page 151)

What's new in the Communication Function Manual, Edition 10/2018 as compared to Edition 12/2017?

What's new?	What are the customer benefits?	Where can I find the information?
Description of communication with the S7-1500R/H redundant system	You receive information on the particularities of communication with the S7-1500R/H redundant system	Section Communication with the redundant system S7-1500R/H (Page 384)
Scope of the function manual expanded to include the S7-1500R/H redundant system	Functions with which you are familiar from the SIMATIC S7-1500 automation system are implemented for the S7-1500R/H redundant system.	S7-1500R/H redundant system System Manual (https://support.industry.siemens.com/cs/ww/en/view/109754833)

What's new in the Communication Function Manual, Edition 12/2017 compared to Edition 09/2016

What's new?	What are the customer benefits?	Where can I find the information?
OPC UA Companion Specification	Through OPC UA Companion Specification, methods can be specified in a uniform and manufacturer-neutral way. Using these specified methods, you can easily integrate devices from various manufacturers into the plant and the production processes.	Section OPC UA server interface configuration (Page 236)
Setting up a secure connection to a mail server over the CPU interface	You can set up a secure connection to a mail server without additional hardware.	Section Secure OUC via e-mail (Page 89)
Secure communication over Modbus TCP	You can establish secure TCP connections between a Modbus TCP client and a Modbus TCP server.	Section Secure OUC with Modbus TCP (Page 88)

What's new in the Communication Function Manual, Edition 09/2016 compared to Edition 12/2014

What's new?	What are the customer benefits?	Where can I find the information?
OPC UA server	OPC UA is a uniform standard for data communication and is independent of any particular operating system platforms. OPC UA uses integrated safety mechanisms on various automation systems, for example with data exchange, at application level, for the legitimation of the user. The OPC UA server provides a large amount of data: • Values of PLC tags that clients can access • Data types of these PLC tags • Information about the OPC UA server itself and the CPU In this way, clients can gain an overview of the tag management and can read and write values.	Section OPC UA communication (Page 151)
Secure Open User Communication	Secure data exchange with other devices.	Section Secure Open User Communication (Page 75)

What's new?	What are the customer benefits?	Where can I find the information?
Certificate handling in STEP 7	You can manage certificates for the following applications in STEP 7: • OPC UA server • Secure Open User Communication • Web server of the CPU	Section Certificate management with TIA Portal (Page 54)
Deactivating SNMP for the CPU	You can deactivate SNMP for the CPU. This can make sense under certain conditions, for example if the security guidelines in your network do not permit SNMP.	Section SNMP (Page 103)

Conventions

STEP 7: We refer to "STEP 7" in this documentation as a synonym for the configuration and programming software "STEP 7 as of V12 (TIA Portal)".
"S7-1500 CPUs" also refers to the CPU variants S7 1500F, S7 1500T, S7 1500TF, S7 1500C, S7-1500R/H, S7 1500pro, ET200S, S7 1500 Software Controller as well as SIMATIC Drive Controller.
This documentation contains pictures of the devices described. The figures may differ slightly from the device supplied.
You should also pay particular attention to notes such as the one shown below:

NOTE

A note contains important information on the product, on handling of the product and on the section of the documentation to which you should pay particular attention.

Industry Mall

The Industry Mall is the catalog and order system of Siemens AG for automation and drive solutions on the basis of Totally Integrated Automation (TIA) and Totally Integrated Power (TIP).
You can find catalogs for all automation and drive products on the Internet (https://mall.industry.siemens.com).

1.1 Function Manuals documentation guide

1.1.1 Information classes Function Manuals

The documentation for the SIMATIC S7-1500 automation system, for the 1513/1516pro-2 PN, SIMATIC Drive Controller CPUs based on SIMATIC S7-1500 and the SIMATIC ET 200MP, ET 200SP, ET 200AL and ET 200eco PN distributed I/O systems is arranged into three areas.
This arrangement enables you to access the specific content you require.
You can download the documentation free of charge from the Internet (https://support.industry.siemens.com/cs/ww/en/view/109742705).

Basic information

The system manuals and Getting Started describe in detail the configuration, installation, wiring and commissioning of the SIMATIC S7-1500, SIMATIC Drive Controller, ET 200MP, ET 200SP, ET 200AL and ET 200eco PN systems. Use the corresponding operating instructions for 1513/1516pro-2 PN CPUs.
The STEP 7 online help supports you in the configuration and programming.
Examples:

- Getting Started S7-1500
- System manuals
- Operating instructions ET 200pro and 1516pro-2 PN CPU
- Online help TIA Portal

Device information

Equipment manuals contain a compact description of the module-specific information, such as properties, wiring diagrams, characteristics and technical specifications.
Examples:

- Equipment manuals for CPUs
- Equipment manuals for interface modules
- Equipment manuals for digital modules
- Equipment manuals for analog modules
- Equipment manuals for communication modules
- Equipment manuals for technology modules
- Equipment manuals for power supply modules
- Equipment manuals for BaseUnits

General information

The function manuals contain detailed descriptions on general topics relating to the SIMATIC Drive Controller and the S7-1500 automation system.
Examples:

- Function Manual Diagnostics
- Function Manual Communication
- Function Manuals Motion Control
- Function Manual Web Server
- Function Manual Cycle and Response Times
- PROFINET Function Manual
- PROFIBUS Function Manual

Product Information

Changes and supplements to the manuals are documented in a Product Information. The Product Information takes precedence over the device and system manuals.
You will find the latest Product Information on the Internet:

- S7-1500/ET 200MP (https://support.industry.siemens.com/cs/de/en/view/68052815)
- SIMATIC Drive Controller (https://support.industry.siemens.com/cs/de/en/view/109772684/en)
- Motion Control (https://support.industry.siemens.com/cs/de/en/view/109794046/en)
- ET 200SP (https://support.industry.siemens.com/cs/de/en/view/73021864)
- ET 200eco PN (https://support.industry.siemens.com/cs/ww/en/view/109765611)

Manual Collections

The Manual Collections contain the complete documentation of the systems put together in one file.
You will find the Manual Collections on the Internet:

- S7-1500/ET 200MP/SIMATIC Drive Controller (https://support.industry.siemens.com/cs/ww/en/view/86140384)
- ET 200SP (https://support.industry.siemens.com/cs/ww/en/view/84133942)
- ET 200AL (https://support.industry.siemens.com/cs/ww/en/view/95242965)
- ET 200eco PN (https://support.industry.siemens.com/cs/ww/en/view/109781058)

1.1.2 Basic tools

The tools described below support you in all steps: from planning, over commissioning, all the way to analysis of your system.

TIA Selection Tool

The TIA Selection Tool tool supports you in the selection, configuration, and ordering of devices for Totally Integrated Automation (TIA).
As successor of the SIMATIC Selection Tools , it assembles the configuration editors for automation technology already familiar into a single tool.

With the TIA Selection Tool , you can generate a complete order list from your product selection or product configuration.
You can find the TIA Selection Tool on the Internet.
(https://support.industry.siemens.com/cs/ww/en/view/109767888)

SIMATIC Automation Tool

You can use the SIMATIC Automation Tool to perform commissioning and maintenance activities on various SIMATIC S7 stations as bulk operations independent of TIA Portal.
The SIMATIC Automation Tool offers a wide range of functions:

- Scanning of a PROFINET/Ethernet system network and identification of all connected CPUs
- Assignment of addresses (IP, subnet, Gateway) and device name (PROFINET device) to a CPU
- Transfer of the date and the programming device/PC time converted to UTC time to the module
- Program download to CPU
- RUN/STOP mode switchover
- CPU localization through LED flashing
- Reading out of CPU error information
- Reading the CPU diagnostic buffer
- Reset to factory settings
- Firmware update of the CPU and connected modules

You can find the SIMATIC Automation Tool on the Internet.
(https://support.industry.siemens.com/cs/ww/en/view/98161300)

PRONETA

SIEMENS PRONETA (PROFINET network analysis) is a commissioning and diagnostic tool for PROFINET networks. PRONETA Basic has two core functions:

- The "Network analysis" offers a quick overview of the PROFINET topology. It is possible to make simple parameter changes (for example, to the names and IP addresses of the devices). In addition, a quick and convenient comparison of the real configuration with a reference system is also possible.
- The "IO test" is a simple and rapid test of the wiring and the module configuration of a plant, including documentation of the test results.

You can find SIEMENS PRONETA Basic on the Internet:
(https://support.industry.siemens.com/cs/ww/en/view/67460624)
SIEMENS PRONETA Professional is a licensed product that offers you additional functions. It offers you simple asset management in PROFINET networks and supports operators of automation systems in automatic data collection/acquisition of the components used through various functions:

- The user interface (API) offers an access point to the automation cell to automate the scan functions using MQTT or a command line.
- With PROFIenergy diagnostics, you can quickly detect the current pause mode or the readiness for operation of devices that support PROFIenergy and change these as needed.
- The data record wizard supports PROFINET developers in reading and writing acyclic PROFINET data records quickly and easily without PLC and engineering.

You can find SIEMENS PRONETA Professional on the Internet.
(https://www.siemens.com/proneta-professional)

SINETPLAN

SINETPLAN, the Siemens Network Planner, supports you in planning automation systems and networks based on PROFINET. The tool facilitates professional and predictive dimensioning of your PROFINET installation as early as in the planning stage. In addition, SINETPLAN supports you during network optimization and helps you to exploit network resources optimally and to plan reserves. This helps to prevent problems in commissioning or failures during productive operation even in advance of a planned operation. This increases the availability of the production plant and helps improve operational safety.
The advantages at a glance

- Network optimization thanks to port-specific calculation of the network load
- Increased production availability thanks to online scan and verification of existing systems
- Transparency before commissioning through importing and simulation of existing STEP 7 projects
- Efficiency through securing existing investments in the long term and the optimal use of resources

You can find SINETPLAN on the Internet (https://new.siemens.com/global/en/products/automation/industrial-communication/profinet/sinetplan.html).

1.1.3 S7 Port Configuration Tool (S7-PCT)

SIMATIC S7-PCT

The Port Configuration Tool (PCT) is a PC-based software for the parameter assignment of Siemens IO-Link Master modules and IO-Link devices from any manufacturer.
You integrate IO-devices using the standardized device description "IODD", which you get from the respective device manufacturer. S7-PCT supports version 1.0 and V1.1 of the IODD.
S7-PCT is called via the hardware configuration of the IO-Link Master from STEP 7. When STEP 7 is not used or the IO-Link Master is not operated on a SIMATIC controller, "standalone"-operation is also possible.
You can find additional information on IO-Link on the Internet (https://new.siemens.com/global/en/products/automation/industrial-communication/io-link.html).

1.1.4 S7 Failsafe Configuration Tool (S7-FCT)

SIMATIC S7-FCT

Failsafe Configuration Tool (FCT) enables you to GSD configure the following devices in third-party engineering systems:

- Selected, functionally fail-safe SIMATIC I/O devices
- Functionally fail-safe SIRIUS ACT PROFINET interfaces

The engineering system must meet the following requirements for this:

- Support of the CPD system integration acc. to "PROFIsafe - Profile for Safety Technology on PROFIBUS DP and PROFINET IO"
- TCI implementation to Conformance Class C3

Additional information on S7-FCT can be found on the Internet (https://support.industry.siemens.com/cs/ww/en/view/109762827).

1.1.5 MultiFieldbus Configuration Tool (MFCT)

MultiFieldbus Configuration Tool

The MultiFieldbus Configuration Tool (MFCT) is a PC-based software for the configuration of Siemens MultiFeldbus devices.

With Siemens MF devices, you download the GSDML file of the MultiFeldbus device to the MFCT for configuring.

Four steps are required for configuring:

1. Assigning the IP address
2. Configuring and parameter assignment
3. Transfer the configuration to the MF device
4. Configuring MF devices in the Engineering System

To do this, the MFCT provides the necessary project files in EDS format (Electronic Data Sheet).

The MFCT runs under Windows and does not require installation or administrator rights.
You can find additional information on MFCT on the Internet.
(https://support.industry.siemens.com/cs/de/en/view/109773881)

1.1.6 SIMATIC Technical Documentation

Additional SIMATIC documents will complete your information. You can find these documents and their use at the following links and QR codes.

The Industry Online Support gives you the option to get information on all topics. Application examples support you in solving your automation tasks.

Overview of the SIMATIC Technical Documentation

Here you will find an overview of the SIMATIC documentation available in Siemens Industry Online Support:

Industry Online Support International
(https://support.industry.siemens.com/cs/ww/en/view/109742705)

Watch this short video to find out where you can find the overview directly in Siemens Industry Online Support and how to use Siemens Industry Online Support on your mobile device:

Quick introduction to the technical documentation of automation products per video (https://support.industry.siemens.com/cs/us/en/view/109780491)

YouTube video: Siemens Automation Products - Technical Documentation at a Glance (https://youtu.be/TwLSxxRQQsA)

mySupport

With "mySupport" you can get the most out of your Industry Online Support.

Registration	You must register once to use the full functionality of "mySupport". After registration, you can create filters, favorites and tabs in your personal workspace.
Support requests	Your data is already filled out in support requests, and you can get an overview of your current requests at any time.
Documentation	In the Documentation area you can build your personal library.
Favorites	You can use the "Add to mySupport favorites" to flag especially interesting or frequently needed content. Under "Favorites", you will find a list of your flagged entries.
Recently viewed articles	The most recently viewed pages in mySupport are available under "Recently viewed articles".
CAx data	The CAx data area gives you access to the latest product data for your CAx or CAe system. You configure your own download package with a few clicks: • Product images, 2D dimension drawings, 3D models, internal circuit diagrams, EPLAN macro files • Manuals, characteristics, operating manuals, certificates • Product master data

You can find "mySupport" on the Internet. (https://support.industry.siemens.com/My/ww/en)

Application examples

The application examples support you with various tools and examples for solving your automation tasks. Solutions are shown in interplay with multiple components in the system - separated from the focus on individual products.
You can find the application examples on the Internet.
(https://support.industry.siemens.com/cs/ww/en/ps/ae)

Safety instructions 2

2.1 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 constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit (https://www.siemens.com/industrialsecurity).

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

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed visit (https://www.siemens.com/cert).

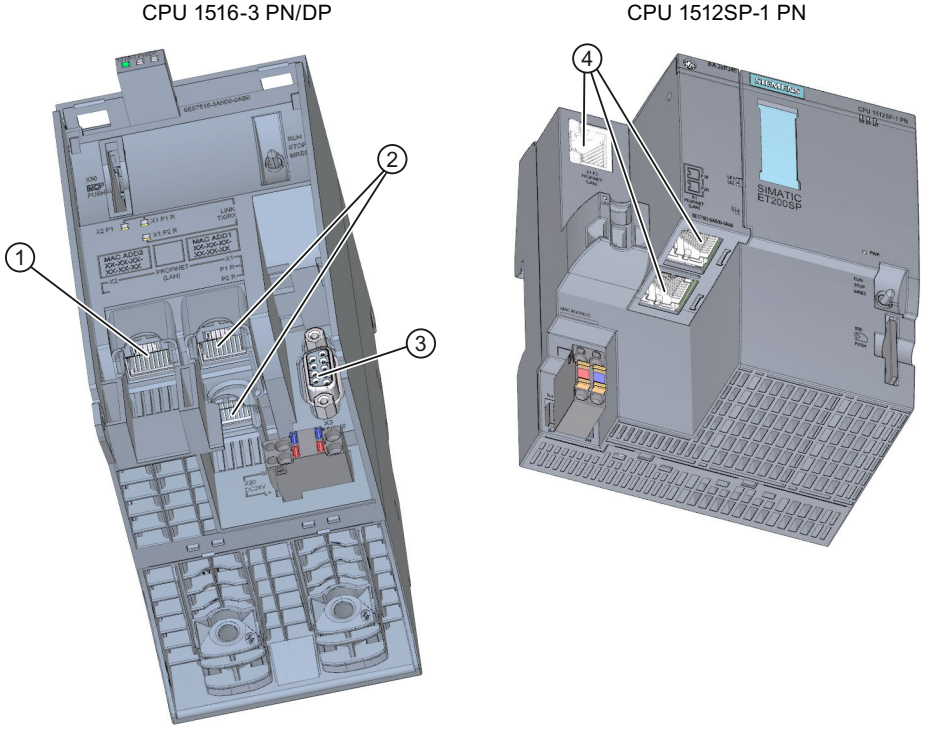
Product overview

3

CPUs, communications modules and processors, and PC systems of the S7-1500, , ET 200MPET 200SPET 200pro and ET 200AL systems provide you with interfaces for communication via PROFINET, PROFIBUS and point-to-point connections.

CPUs, communications modules and communications processors

PROFINET and PROFIBUS DP interfaces are integrated in the S7-1500 CPUs. The CPU 1516-3 PN/DP for example has 2 PROFINET interfaces and 1 PROFIBUS DP interface. Other PROFINET and PROFIBUS DP interfaces are available by using communications modules (CM) and communications processors (CP).

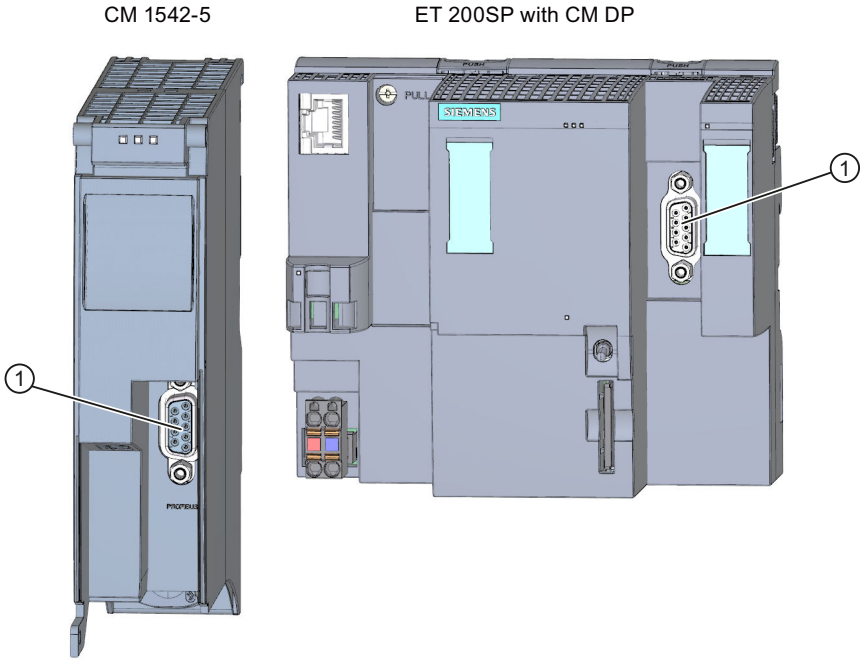


① PROFINET interface (X2) with 1 port
② PROFINET interface (X1) with 2-port switch
③ PROFIBUS DP interface (X3)
④ PROFINET interface (X1) with 3-port switch

Figure 3-1 Interfaces of the CPU 1516-3 PN/DP and CPU 1512SP-1 PN

Interfaces of communications modules

Interfaces of communications modules (CMs) extend the interfaces of CPUs (for example, the communication module CM 1542-5 adds a PROFIBUS interface to S7-1500 automation system).

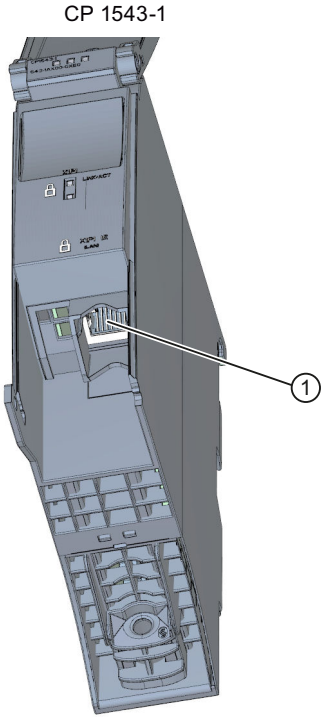


① PROFIBUS DP interface

Figure 3-2 PROFIBUS DP interface of the CM 1542-5 and CM DP (to an ET 200SP CPU)

Interfaces of communications processors

Interfaces of communication processors (CP) offer additional functionality to what is provided by the integrated interfaces of the CPUs. CPs allow special applications, for example the CP 1543-1 provides security functions for protecting Industrial Ethernet networks via its Industrial Ethernet interface.



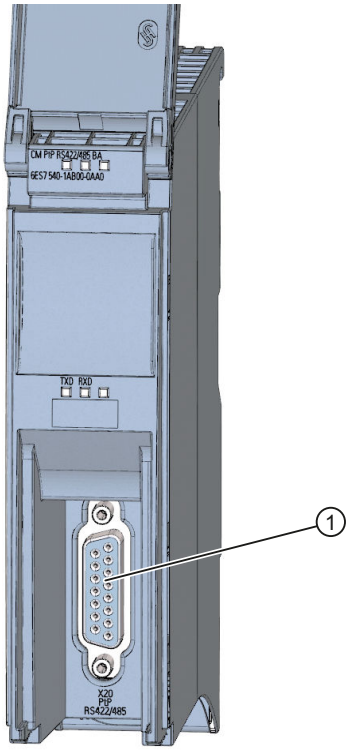
① Industrial Ethernet interface

Figure 3-3 Industrial Ethernet interface of the CP 1543-1

Interfaces of communications modules for point-to-point connections

The communication modules for point-to-point connections provide communication via their RS 232-, RS 422- and RS 485 interfaces, for example, Freeport or Modbus communication.

CM PtP RS422/485 BA

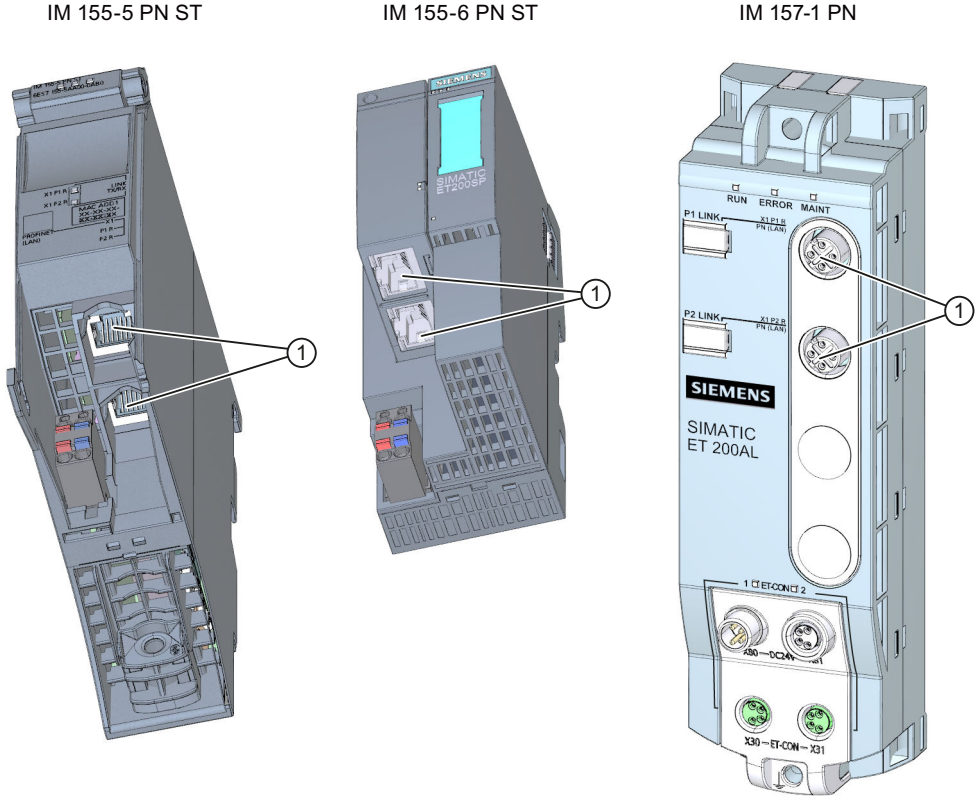


① Interface for point-to-point connections

Figure 3-4 Example of interface for point-to-point connection at the CM PtP RS422/485 BA

Interfaces of interface modules

PROFINET and PROFIBUS DP interfaces of the interface modules (IM) in ET 200MP, ET 200SP and ET 200AL are used to connect the distributed I/O ET 200MP, ET 200SP and ET 200AL to PROFINET or PROFIBUS of the higher-level IO controller or DP master.



① PROFINET interface with 2-port switch

Figure 3-5 PROFINET interfaces IM 155-5 PN ST (ET 200MP), IM 155-6 PN ST (ET 200SP), and IM 157-1 PN (ET 200AL)

Communications services

The communications services described below use the interfaces and communication mechanisms provided by the system via CPUs, communication modules and processors.

Communications services

4

4.1 Overview of communication options

Overview of communications options

The following communications options are available for your automation task.

Table 4-1 Communications options

Communications options	Functionality	Via interface:		
		PN/IE[1]	DP	serial
PG communication[2]	On commissioning, testing, diagnostics	X	X	-
HMI communication[2]	On operator control and monitoring	X	X	-
Open communication via TCP/IP[2]	Data exchange via PROFINET/Industrial Ethernet with TCP/IP Instructions: • TSEND_C/TRCV_C • TSEND/TRCV • TCON • T_DISCON	X	-	-
Open communication using ISO-on-TCP[2]	Data exchange via PROFINET/Industrial Ethernet with ISO-on-TCP Instructions: • TSEND_C/TRCV_C • TSEND/TRCV • TCON • T_DISCON	X	-	-
Open communication via UDP[2]	Data exchange via PROFINET/Industrial Ethernet with UDP Instructions: • TSEND_C/TRCV_C • TUSEND/TURCV • TCON • T_DISCON	X	-	-
Open communication via ISO (only CPs with PROFINET/Industrial Ethernet interface)	Data exchange via PROFINET/Industrial Ethernet with the ISO protocol Instructions: • TSEND_C/TRCV_C • TSEND/TRCV • TCON • T_DISCON	X	-	-

1 IE - Industrial Ethernet

2 Observe the special characteristics for S7-1500R/H

3 Only via internal PROFINET interface of the CPU and via Ethernet interface CP 1543 1 with activated "Access to PLC via communication module" function

Communications options	Functionality	Via interface:		
		PN/IE[1]	DP	serial
Open communication with FDL (only CM 1542-5 as of firmware V2.0)	Data exchange via PROFIBUS with the FDL protocol Instructions: • TSEND_C/TRCV_C • TSEND/TRCV • TUSEND/TURCV • TCON • T_DISCON	-	X	-
OPC UA server[3]	Data exchange with OPC UA clients	X	-	-
Communication via Modbus TCP	Data exchange via PROFINET with Modbus TCP protocol Instructions: • MB_CLIENT • MB_SERVER	X	-	-
Email	Sending process alarms via email Instruction: • TMAIL_C	X	-	-
FTP (only CPs with PROFINET/Industrial Ethernet interface)	File management and file access via FTP (File Transfer Protocol); CP can be FTP client and FTP server Instruction: • FTP_CMD	X	-	-
Fetch/Write (only CPs with PROFINET/Industrial Ethernet interface)	Server services via TCP/IP, ISO-on-TCP and ISO Via special instructions for Fetch/Write	X	-	-
S7 communication	Data exchange via PROFINET/PROFIBUS with the S7 protocol. Instructions: • PUT/GET • BSEND/BRCV • USEND/URCV	X	X	-
Serial point-to-point connection	Data exchange via point-to-point with Freeport, 3964(R), USS or Modbus protocol Via special instructions for PtP, USS or Modbus RTU	-	-	X
Web server	Data exchange via HTTP(S), for example for diagnostics	X	-	-
SNMP (Simple Network Management Protocol)	For monitoring and error recognition of IP networks, possibly parameterization of the IP network components via standard SNMP protocol	X	-	-
Time-of-day synchronization	Via PN/IE interface: CPU is NTP client (Network Time Protocol)	X	-	-
	Via DP interface: CPU/CM/CP is time-of-day master or time slave	-	X	-

1 IE - Industrial Ethernet

2 Observe the special characteristics for S7-1500R/H

3 Only via internal PROFINET interface of the CPU and via Ethernet interface CP 1543 1 with activated "Access to PLC via communication module" function

Information on S7-1500R/H

You can find information on the communication possibilities with the S7-1500R/H redundant system in the section Communication with the redundant system S7-1500R/H (Page 384).

More information

- An application example for the configuration of the TLS-based PG/HMI communication and protection of confidential configuration data of the CPU can be found in this Application example (https://support.industry.siemens.com/cs/ww/en/view/109798583).
- A general application example for CPU-CPU communication with SIMATIC controllers (compendium) can be found in this application example (https://support.industry.siemens.com/cs/ww/en/view/20982954).
- A TIA library "LOpcUa", which provides you with function blocks for the implementation of OPC UA PubSub for SIMATIC S7-1500, can be found in this Application example (https://support.industry.siemens.com/cs/ww/en/view/109782455).

- This FAQ (https://support.industry.siemens.com/cs/ww/en/view/102420020) describes how to configure fetch/write communication via CP1543-1 with S7-1500.
- More information about the Fetch/Write services is available in the STEP 7 online help.

- You can find more information on the PtP connection in the function manual CM PtP - Configurations for Point-to-Point Connections (https://support.industry.siemens.com/cs/us/en/view/59057093).
- You will find the description of the web server functionality in the function manual Web server (https://support.industry.siemens.com/cs/us/en/view/59193560).

- You can find general information about the standard protocol SNMP on the Service & Support pages on the Internet (https://support.industry.siemens.com/cs/us/en/view/15166742). Answers to the question of which SNMP requests support S7-1500 CPUs and S7-1200 CPUs can be found in this FAQ (https://support.industry.siemens.com/cs/at/en/view/79993228).
- You will find information about time-of-day synchronization in this FAQ (https://support.industry.siemens.com/cs/ww/en/view/86535497).

4.2 Communications protocols and port numbers used for Ethernet communication

This section provides an overview of the supported protocols and port numbers used for communication over PN/IE interfaces. For each protocol the address parameters, the respective communications layer as well as the communications role and the communications direction are specified.

This information makes it possible to match the security measures for protection of the automation system to the used protocols (for example firewall). Because the security measures are limited to Ethernet or PROFINET networks, the tables do not include PROFIBUS protocols.

NOTE

Port numbers used

The specified port numbers are the standard port numbers used by the S7-1500 CPU. Many communication protocols and implementations enable you to use other port numbers.

The following tables show the different layers and protocols used in the S7-1500 CPUs and S7 1500 communication modules.

Layers and protocols of the S7-1500 CPUs and software controllers (via PROFINET interface of the CPU)

The following table shows the protocols supported by S7-1500 CPUs, ET 200SP CPUs and the 1513/1516pro 2 PN CPUs. The S7-1500 software controllers also support the protocols listed in the following table for the Ethernet interfaces that are assigned to the software controller.

Table 4-2 Layers and protocols of the S7-1500 CPUs and software controllers (via PROFINET interface of the CPU)

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Default setting / notes
PROFINET protocols				
DCP	Not relevant	(2) Ethertype 0x8892 (PROFINET)	PROFINET Discovery and Basic Configuration Protocol. DCP determines PROFINET devices and enables the basic settings.	Default: Activated. Function can be deactivated in the CPU properties by Boundary "End of detection of accessible nodes" of the interface.
DHCP Client	68	(4) UDP	Dynamic Host Configuration Protocol. The IP address suite is obtained from a DHCP server during the startup of the PROFINET interface.	Default: Deactivated. Can be changed in the CPU properties (as of FW version 2.9).
LLDP	Not relevant	(2) Ethertype 0x88CC (LLDP)	PROFINET Link Layer Discovery Protocol. LLDP determines and manages neighborhood relations between PROFINET devices.	Default: Activated. Send function can be deactivated by Boundary "End of topology discovery" in the CPU properties; readiness to receive remains active. LLDP uses the special multicast MAC address: 01-80-C2-00-00-0E.
MRP	Not relevant	(2) Ethertype 0x88E3 (IEC 62493-2-2010)	Media Redundancy Protocol. MRP enables control of redundant transmission paths in a ring topology.	Default: "Manager (Auto)". Can be changed in the CPU properties. If you configure the CPU and connect the PN interface with a subnet, the default setting in the TIA Portal is "Not device in the ring". MRP uses standard-compliant Multicast MAC addresses.

[1] Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

[2] Do not use ports for OUC, which are already used by other protocols.

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Default setting / notes
PTCP	Not relevant	(2) Ethertype 0x8892 (PROFINET)	PROFINET Precision Transparent Clock Protocol, based on IEEE 1588. PTCP provides a time delay measurement between RJ45 ports and thus the send clock synchronization and time synchronization.	Default: Deactivated. Can be enabled by the following configurations: • IRT with a sync domain. • Port interconnection with a specified cable length. Function can be deactivated in the CPU properties by Boundary "End of sync domain" of the interface. PTCP uses standard-compliant Multicast MAC addresses.
PROFINET IO data	Not relevant	(2) Ethertype 0x8892 (PROFINET)	PROFINET Cyclic IO Data Transfer With PROFINET IO telegrams, IO data is transferred cyclically between the PROFINET IO controller and IO devices via Ethernet.	Default: Deactivated. The protocol is only activated for PROFINET IO data traffic.
PROFINET Context Manager	34964	(4) UDP	PROFINET connection without RPC. Management of application and communication relationships between IO controller and IO devices.	Default: Enabled (UDP port open). This function cannot be deactivated.
Connection-oriented communications protocols				
SMTP Client	25	(4) TCP	Simple Mail Transfer Protocol. SMTP is used for sending emails	Default: Deactivated. Can be enabled via TMAIL_C instruction in the user program.
SMTPS (SMTP over TLS) Client	465	(4) TCP	Simple Mail Transfer Protocol Secure. SMTP is used for sending emails over secure connections.	Default: Deactivated. Can be enabled via TMAIL_C instruction in the user program.
SMTP with STARTTLS Client	25 587	(4) TCP	Simple Mail Transfer Protocol with the SMTP command "STARTTLS" SMTP is used for sending emails.	Default: Deactivated. Can be enabled via TMAIL_C instruction in the user program.
HTTP Server	80	(4) TCP	Hypertext Transfer Protocol. HTTP is used for communication with the CPU-internal web server.	Default: Deactivated. Can be enabled in the CPU properties. Requirement: Web server in the CPU properties is enabled.
HTTPS Server	443	(4) TCP	Hypertext Transfer Protocol Secure. HTTPS is used for communication with the CPU-internal Web server via Secure Socket Layer (SSL).	Default: Deactivated. Can be enabled in the CPU properties. Requirement: Web server in the CPU properties is enabled.
ISO-on-TCP Server	102	(4) TCP	ISO-on-TCP protocol (according to RFC 1006). The S7 protocol uses ISO-on-TCP according to RFC 1006 for PG/HMI communication with the engineering system (TIA Portal).	Default: Activated. This function cannot be deactivated.

[1] Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

[2] Do not use ports for OUC, which are already used by other protocols.

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Default setting / notes
NTP Client	123	(4) UDP	Network Time Protocol. NTP is used for synchronization of the CPU system time with the time of an NTP server.	Default: Deactivated. Can be enabled in the CPU properties.
SNMP Agent	161 162 (trap)	(4) UDP	Simple Network Management Protocol. SNMP is used for reading and setting of network management data (SNMP managed objects) by the SNMP Manager.	Default: Activated up to FW version V2.9, deactivated as of FW version V3.0. Can be enabled via data record in the user program. Can be enabled in the CPU properties as of FW version V3.0.
MODBUS TCP Server / Client	502	(4) TCP	MODBUS Transmission Control Protocol. MODBUS/TCP is used by MB_CLIENT/MB_SERVER instructions in the user program.	Default: Deactivated. Can be activated via Modbus instructions in the user program.
OPC UA Server / Client	4840	(4) TCP	Open Platform Communications Unified Architecture (based on TCP/IP protocol). A communication standard ranging from the enterprise level to the field level.	Default: Deactivated. Server and client function can be enabled in the CPU properties. Client access can be configured in the user program.
OUC[1] Secure OUC Server / Client	1 ... 1999 can be used to limited extent[2] 2000 ... 5000 Recommended	(4) TCP (4) UDP (4) ISO-on-TCP (Port: 102)	Open User Communication (TCP/UDP). Secure Open User Communication (TLS). OUC instructions enable connection establishment, connection termination and data transfer via the user program.	Default: Deactivated. You activate the respective protocol with the corresponding Open User Communication instruction in the user program or with the configuration of connections in the network view.

[1] Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

[2] Do not use ports for OUC, which are already used by other protocols.

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Default setting / notes
OUC[1] Secure OUC Server / Client	As of FW version V3.0, the following applies to programmed and configured connections: 5001 ... 65535 can be used to limited extent[2]	(4) TCP (4) UDP (4) ISO-on-TCP (Port: 102)	Open User Communication (TCP/UDP). Secure Open User Communication (TLS). OUC instructions enable connection establishment, connection termination and data transfer via the user program.	The following applies to FW versions lower than V3.0: • Programmed connections: 5001 ... 49152 • Configured connections: 5001 ... 65535
IGMPv2	Not relevant	(3) Network layer	Internet Group Management Protocol. IGMPv2 is a network protocol for the organization of multicast groups (UDP multicast only).	IGMPv2 is a functionality of the IP stack. This system function is activated by the multicast function.
Reserved	49152 ... 65535	(4) TCP (4) UDP	If an application does not address a local port, then the CPU uses this port range for the active connection point.	-

1 Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

2 Do not use ports for OUC, which are already used by other protocols.

Layers and logs of the S7-1500 Software Controller (via Ethernet interface on the Windows side)

The following table shows the protocols that are supported by the S7-1500 software controller via the Ethernet interfaces assigned to Windows.

Table 4-3 Layers and logs of the S7-1500 Software Controller (via Ethernet interface on the Windows side)

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Notes / default setting
PROFINET protocols				
DCP	Not relevant	(2) Ethertype 0x8892 (PROFINET)	PROFINET Discovery and Basic Configuration Protocol. DCP determines PROFINET devices and enables the basic settings.	Default: Activated. Function can be disabled with Boundary "End of detection of accessible nodes" in the CPU properties.

1 Default setting for Windows assigned interfaces: 81

2 Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

3 Do not use ports for OUC, which are already used by other protocols.

4 Do not use ports for OUC, which are already used by other Windows applications.

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Notes / default setting
DHCP Client	68	(4) UDP	Dynamic Host Configuration Protocol. The IP address suite is obtained from a DHCP server during the startup of the PROFINET interface.	Default: Deactivated. Can be changed in the CPU properties (as of FW version 2.9).
Connection-oriented communications protocols				
SMTP Client	25	(4) TCP	Simple Mail Transfer Protocol. SMTP is used for sending emails.	Default: Deactivated. Can be activated by calling the block in the user program or as of version V3.0 via CPU settings.
HTTP Server	Adjustable [1]	(4) TCP	Hypertext Transfer Protocol. HTTP is used for communication with the CPU-internal web server.	Default: Deactivated. Can be changed in the CPU properties. Adapt the port number to avoid conflicts with other web servers under Windows. If you use the Web server access of the S7-1500 software controller, you must enable the assigned port in the Windows firewall.
ISO-on-TCP Server	102	(4) TCP	ISO-on-TCP protocol (according to RFC 1006). The S7 protocol uses ISO-on-TCP according to RFC 1006 for PG/HMI communication with the engineering system (TIA Portal).	Default: Deactivated.
OUC[2] and Secure OUC	1 ... 1999 can be used to limited extent[3, 4]	(4) TCP (4) UDP (4) ISO-on-TCP (Port: 102)	Open User Communication (TCP/UDP). Secure Open User Communication (TLS). OUC instructions enable connection establishment, connection termination and data transfer based on the socket layer.	Default: Deactivated. Can be enabled via data record in the user program. If you want to use OUC, you must activate the ports in the Windows Firewall.
	2000 ... 5000 recommended[4]			
	5001 ... 49151 can be used to limited extent[3, 4]			
IGMPv2	Not relevant	(3) Network layer	Internet Group Management Protocol. IGMPv2 is a network protocol for the organization of multicast groups (UDP multicast only).	IGMPv2 is a functionality of the IP stack. This system function is activated by the multicast function.

1 Default setting for Windows assigned interfaces: 81

2 Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

3 Do not use ports for OUC, which are already used by other protocols.

4 Do not use ports for OUC, which are already used by other Windows applications.

Protocol / Role	Port number	(2) Link layer (4) Transport layer	Description / function	Notes / default setting
Reserved	49152 ... 65535	(4) TCP (4) UDP	If the application does not specify the local port number, this dynamic port range is used for the active connection end point.	If you want to use this communication, you must activate the ports in the Windows Firewall.

1 Default setting for Windows assigned interfaces: 81

2 Note: OUC (open communication) provides direct access to the UDP and TCP protocols. You must take into consideration the IANA (Internet Assigned Numbers Authority) port restrictions and definitions.

3 Do not use ports for OUC, which are already used by other protocols.

4 Do not use ports for OUC, which are already used by other Windows applications.

Layers and protocols of S7-1500 communications modules

The documentation for the protocols of S7-1500 communications modules (e.g. CP 1543-1) can be found here (https://support.industry.siemens.com/cs/ww/en/view/67700710).

4.3 Overview of connection resources

Connection resources

Some communications services require connections. Connections allocate resources on the CPUs, CPs and CMs involved (for example memory areas in the CPU operating system). In most cases one resource per CPU/CP/CM is allocated for a connection. In HMI communication, up to 3 connection resources are required per HMI connection.

The connection resources available depend on the CPU being used, the CPs and CMs and must not exceed a defined high limit for the automation system.

Available connection resources in a station

The maximum number of resources of a station is determined by the CPU.

Each CPU has reserved connection resources for PG, HMI and web server communication. In addition, there are available resources for other communication services, e.g. for SNMP, e-mail connections, HMI and S7 communication as well as for open communication.

When are connection resources allocated?

The time for allocation of connection resources depends on how the connection is set up, automatic, programmed or configured (see section Setting up a connection (Page 37)).

Additional information

You will find more detailed information on the allocation of connection resources and the display of connection resources in STEP 7 in the section Connection resources (Page 370).

4.4 Setting up a connection

Automatic connection

STEP 7 sets up a connection automatically (for example PG or HMI connection) if you have connected the PG/PC interface to an interface of the CPU physically and have made the interface assignment in STEP 7 in the "Go online" dialog.

Setting up a programmed connection

You set up the programmed connection in the program editor of STEP 7 in the context of a CPU by assigning instructions for communication, for example TSEND_C.

When specifying the connection parameters (in the Inspector window, in the properties of the instruction), you are supported by the easy-to-use user interface.

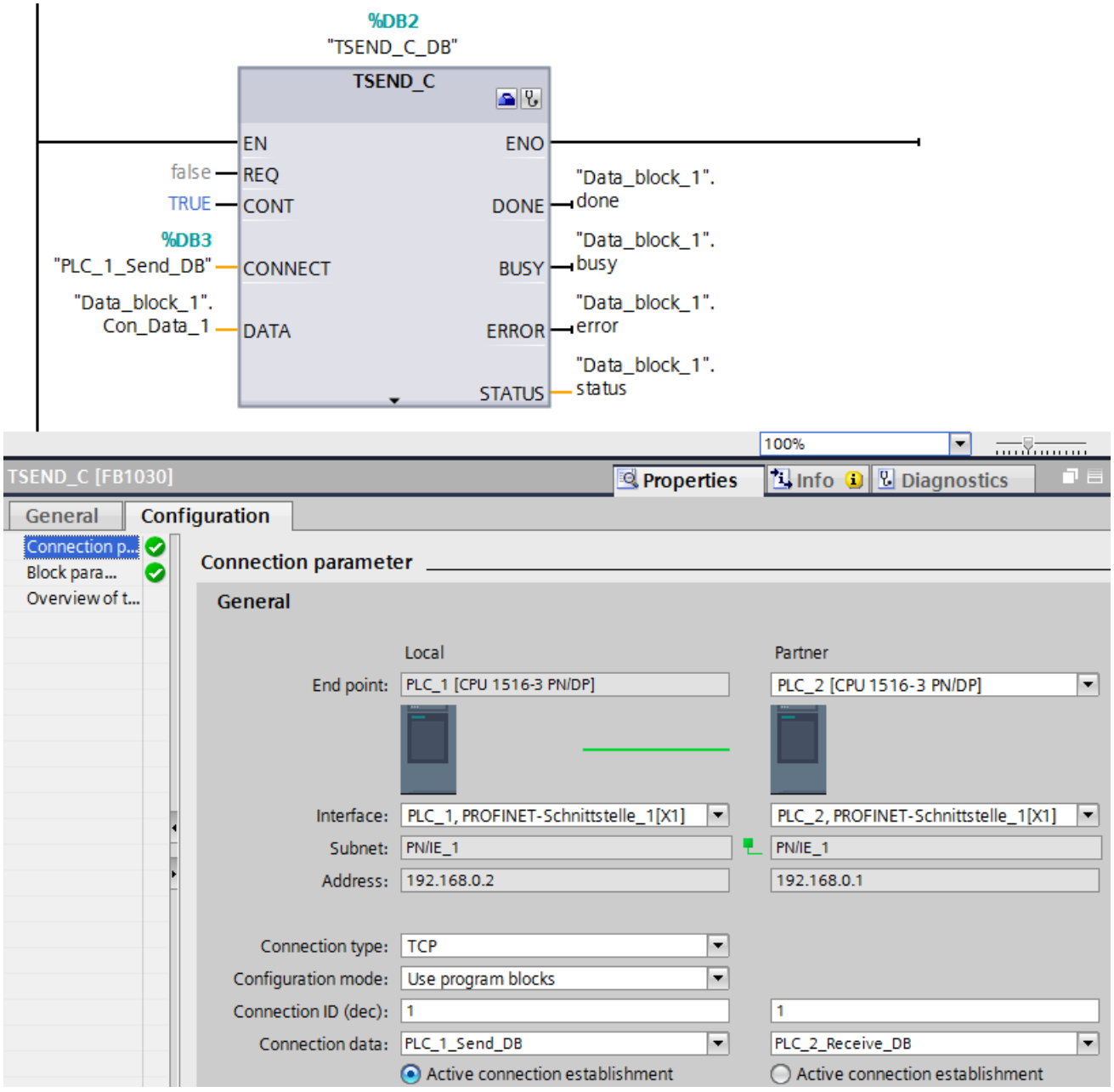


Figure 4-1 Programmed setup

Setting up a configured connection

You set up the configured connection in the network view of the Devices & networks editor of STEP 7 in the context of a CPU or a software controller.

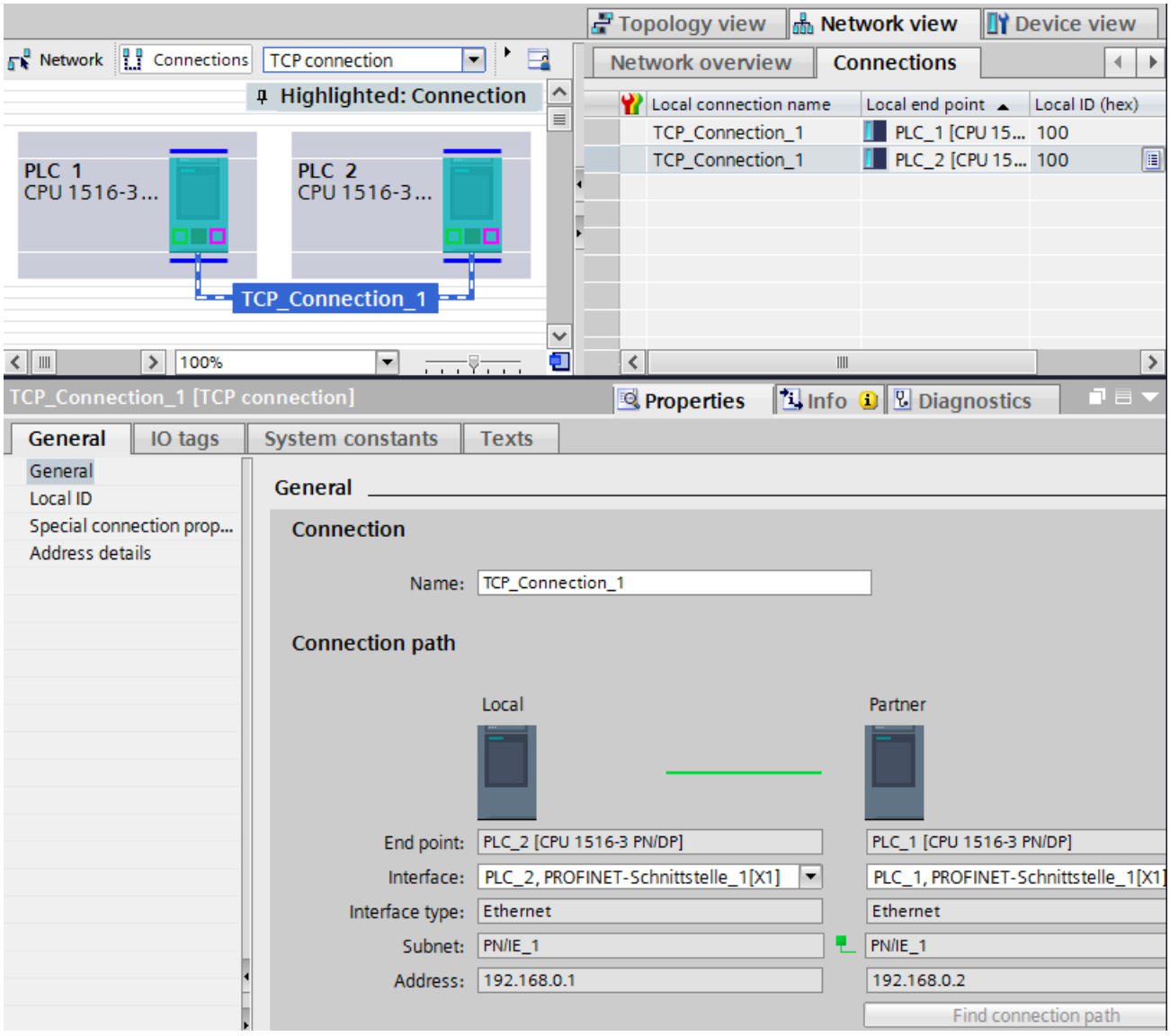


Figure 4-2 Configured setup

Effects on the connection resources of the CPU

You can often choose between a configured or a programmed connection. Programmed connection setup allows connection resources to be released following data transfer. Like routed connections, programmed connections are not guaranteed, meaning that they are only established when resources are available. With configured connection setup, the resource is available after download of the configuration until the configuration changes again. Corresponding resources are therefore reserved for connection establishment via

configured connections. The "Connection resources" table in the Inspector window of the CPU displays an overview of connection resources already used and those still available.

How do I set up a connection?

Table 4-4 Setting up the connection

Connection	Automatically	Programmed setup	Configured setup
Programming device connection	X	-	-
HMI connection	X	-	X
Web communication	X	-	-
OPC UA server communication	X	-	-
OPC UA client communication	-	X	-
Open communication via TCP/IP connection	-	X	X
Open communication via ISO-on-TCP connection	-	X	X
Open communication via UDP connection	-	X	X
Open communication via ISO connection	-	X	X
Open communication via FDL connection	-	X	X
Communication via Modbus TCP connection	-	X	-
E-mail connection	-	X	-
FTP connection	-	X	-
S7 connection*	-	-	X

* Note that for an S7-1500 CPU you must enable the use of PUT/GET communication in the properties of the CPU. You can find more information on this topic in the STEP 7 online help.

Additional information

You will find further information on the allocation of connection resources and the display of connection resources in STEP 7 in the section Connection resources (Page 370).

4.5 Data consistency

Definition

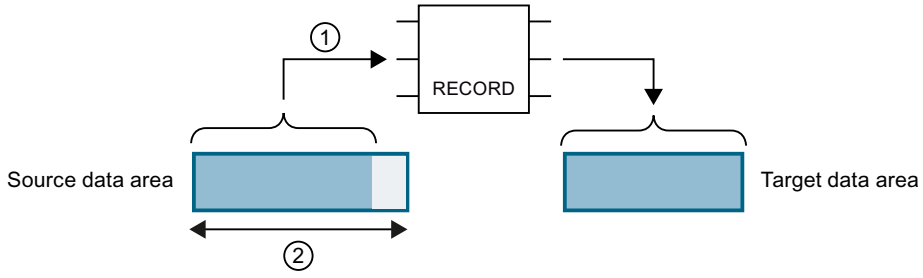
Data consistency is important for data transfer and you need to take this into account when configuring the communication task. Otherwise, malfunctions may occur.

A data area which cannot be modified by concurrent processes is called a consistent data area. This means that a data area which belongs together and which is larger than the

maximum size of the consistent data area can consist in part of new and of old data at the same time.

An inconsistency can occur when an instruction for communication is interrupted, for example by a hardware interrupt OB with higher priority. This interrupts the transfer of the data area. If the user program in this OB now changes the data that has not yet been processed by the communication instruction, the transferred data originates from different times:

The following figure shows a data area that is smaller than the maximum size of the consistent data area. In this case, when transferring the data area, it is ensured that there is no interruption by the user program during data access so that the data is not changed.

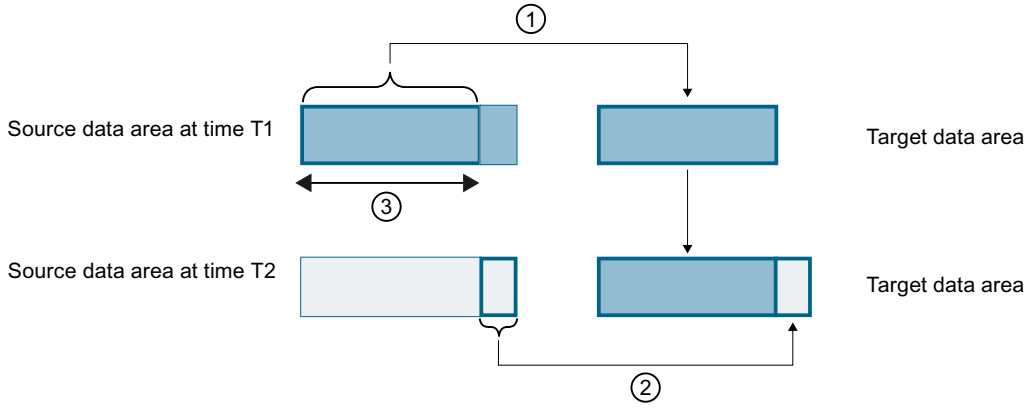


① The source data area is smaller than the maximum size of the consistent data area (②). The instruction transfers the data together to the destination data area.

② Maximum size of the consistent data area

Figure 4-3 Consistent transfer of data

The following figure shows a data area that is larger than the maximum size of the consistent data area. In this case, the data can be changed during an interruption of the data transfer. An interruption also occurs if, for example, the data area needs to be transferred in several parts. If the data is changed during the interruption, the transferred data originates from different times.



① The source data area is larger than the maximum size of the consistent data area (③). At time T1, the instruction only transfers as much data from the source data area into the destination data area as fits in the consistent data area.

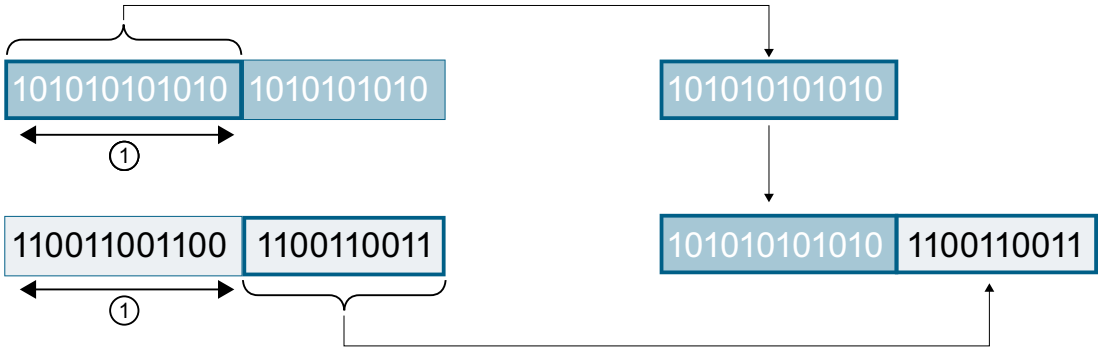
② At time T2, the instruction transfers the rest of the source data area to the destination data area. After the transfer, data from different points in time exist in the destination data area. If the data in the source data area has changed in the meantime, an inconsistency may result.

③ Maximum size of the consistent data area

Figure 4-4 Transfer of data larger than the maximum consistency area

Example of an inconsistency

The figure below shows an example of changing data during the transfer. The destination data area contains data from different points in time.



① Maximum size of the consistent data area

Figure 4-5 Example: Changing data during the transfer

System-specific maximum data consistency for S7-1500:

No inconsistency occurs if the system-specific maximum size of the consistent data is kept to. With an S7-1500, communication data is copied consistently into or out of the user memory in blocks of up to 512 bytes during the program cycle. Data consistency is not ensured for larger data areas. Where defined data consistency is required, the length of communication data in the user program of the CPU must not exceed 512 bytes. You can then access these data areas consistently, for example from an HMI device by reading/writing tags.

If more data than the system-specific maximum size needs to be transferred consistently, you yourself must ensure the data consistency with suitable measures in the user program.

Ensuring data consistency

Use of instructions for access to common data:

If the user program contains instructions for communication that access common data, for example TSEND/TRCV, you can coordinate access to this data area yourself, for example using the "DONE" parameter. The data consistency of the data areas that are transferred with an instruction for communication can therefore be ensured in the user program.

NOTE

Measures in the user program

To achieve data consistency, you can copy transferred data to a separate data area (for example, global data block). While the user program continues to work with the original data, you can transfer the data saved in the separate data area consistently with the communication instruction.

For the copying, use uninterruptible instructions such as UMOVE_BLK or UFILL_BLK. These instructions ensure data consistency up to 16 KB.

Use of PUT/GET instructions or Write/Read via HMI communication:

In S7 communication with the PUT/GET instructions or Write/Read via HMI communication, you need to take into account the size of the consistent data areas during programming or configuration. In the user program of an S7-1500 as server, there is no instruction available that can coordinate the data transfer in the user program. The data exchanged using PUT/GET instructions updates the S7-1500 while the user program is running. There is no point in time within the processing of the cyclic user program at which the data is exchanged consistently. The length of the data area to be transferred should be smaller than 512 bytes.

Additional information

- You will also find the maximum amount of consistent data in the device manuals of the communications modules in the Technical Specifications.
- You will find further information on data consistency in the description of the instructions in the STEP 7 online help.

4.6 Secure Communication

4.6.1 Basics of Secure Communication

4.6.1.1 Useful information on Secure Communication

For STEP 7 (TIA Portal) as of V14 and for S7-1500 CPUs as of firmware V2.0, the options for secure communication have been broadened considerably.
"S7-1500 CPUs" also refers to CPU versions S7-1500F, S7-1500T, S7-1500C as well as S7-1500pro CPUs and ET200SP CPUs.
In subsequent versions, additional components will support Secure Communication (e.g. secure OUC), see next section.
As of firmware version V4.4, S7-1200 CPUs also support Secure Communication.

Requirement

- CPUs that support connection description DBs with the structure of the SDT TCON_IP_V4_SEC or SDT TCON_QDN_SEC. These are the following CPUs:
  - S7-1200 as of firmware V4.4
  - S7-1500 as of firmware V2.0
- Also optional via the following CPs:
  - CP 1243-1 as of firmware V3.2
  - CP 1243-8 IRC as of firmware V3.2
  - CP 1543-1 as of firmware V2.0
  - CP 1545-1
  - CP 1543SP-1

  Secure Communication via CP 1242-7 GPRS V2 is not possible.

Public Key Infrastructure (PKI)

The attribute "secure" is used for the identification of communication mechanisms that are based on a Public Key Infrastructure (PKI) (for example RFC 5280 for Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List Profile). A Public Key Infrastructure (PKI) is a system that can issue, distribute and check digital certificates. The digital certificates issued are used in the PKI to secure computer-based communication. If a PKI uses asymmetric key cryptography, the messages in a network can be digitally signed and encrypted.

Components that you have configured in STEP 7 (TIA Portal) for secure communication use an asymmetric key encryption scheme with a public key (Public Key) and a private key (Private Key). TLS (Transport Layer Security) is used as the encryption protocol. TLS is the successor for the SSL (Secure Sockets Layer) protocol.

Objectives of secure communication

Secure communication is used to achieve the following objectives:

- Confidentiality
  i.e. the data are secret / cannot by read by eavesdroppers.
- Integrity
  i.e. the message that reaches the recipient is the same message, unchanged, that the sender sent. The message has not been altered on the way.
- End point authentication
  i.e. the end point communication partner is exactly who it claims to be and the party who is to be reached. The identity of the partner has been checked.

These objectives were in the past primarily relevant to IT and networked computers. Now, industrial machinery and control systems with sensitive data are at equally high risk, as they are also networked, and consequently pose strict security requirements for data exchange.

Protection of the automation cell by means of the cell protection concept through firewall, or via connection through VPN, for example with the security module, was common in the past and remains so.

However, it is becoming increasingly necessary to also transfer data to external computers in encrypted form via Intranet or public networks.

Common principles of secure communication

Independent of the context, secure communication is based on the concept of the Public Key Infrastructure (PKI) and contains the following components:

- An asymmetric encryption scheme that allows:
  - Encryption or decryption of messages using public or private keys.
  - The verification of signatures in messages and certificates.
    The messages/certificates are signed by the sender/certificate subject with their private key. The recipient/verifier checks the signature with the public key of the sender/certificate subject.

- Transport and storage of the public key using X.509 certificates:
  - X.509 certificates are digitally signed data that allow public key authentication in terms of the bound identity.
  - X.509 certificates can contain information that describes in more detail or restricts use of the public key. For example the date as of which a public key in a certificate is valid and when it expires.
  - X.509 certificates contain information about the issuer of the certificate in secure form.

The following paragraphs give an overview of these basic concepts, which are required for managing certificates in STEP 7 (TIA Portal), for example, and for programming communication instructions for secure Open User Communication (sOUC).

Secure communication with STEP 7

STEP 7 as of V14 provides the required PKI for the configuration and operation of secure communication.
Examples:

- The Hypertext Transfer Protokoll (HTTP) turns into Hypertext Transfer Protokoll Secure (HTTPS) with the help of the TLS (Transport Layer Security) protocol. Since HTTPS is a combination of HTTP and TLS, it is called "HTTP over TLS" in the corresponding RFC. You can see in the browser that HTTPS is being used; this is indicated by the URL "https://" instead of "http://" in the address bar of the browser. Most browsers highlight such secure connections.
- Open User Communication turns into secure Open User Communication. The underlying protocol is also TLS.
- Email providers also offer access over the "Secure SMTP over TLS" protocol to increase the security of email communication.

The figure below shows the TLS protocol in the context of communication layers.

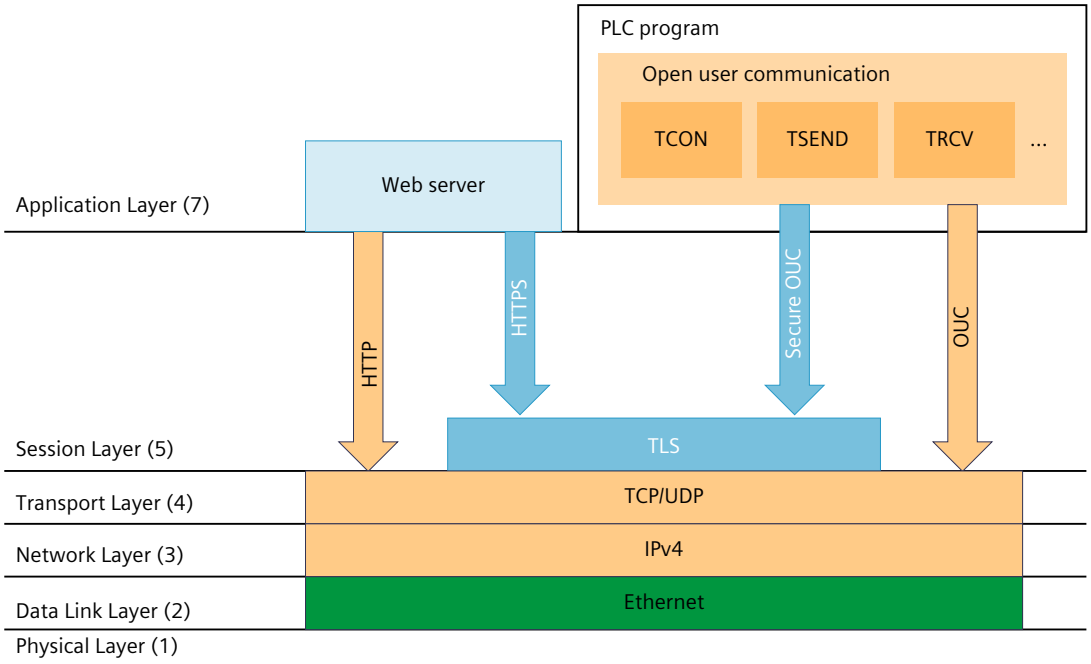


Figure 4-6 TLS protocol in the context of communication layers

Secure communication with OPC UA

An OPC UA server is implemented in S7-1500 CPUs as of firmware V2.0. OPC UA Security also covers authentication, encryption and data integrity with digital X.509 certificates and also uses a Public Key Infrastructure (PKI). Depending on the requirements placed by the application, you can select different security levels for the end point security. The description of the OPC UA server functionality is covered in a separate section.

Secure communication for PG/HMI communication

With the central components of the TIA Portal, STEP 7 and WinCC, an innovative and standardized Secure PG/PC and HMI Communication - PG/HMI communication for short - is implemented starting with version V17 together with the latest controllers and latest HMI devices.

More information

You can find more information on OPC UA in the section Using the S7-1500 as an OPC UA server (Page 196).
For more information on secure programming device/HMI communication, refer to the section Secure PG/HMI communication (Page 93).

4.6.1.2 Device-dependent security features

Transport Layer Security (TLS) is a widespread security protocol that improves the data security for communications. For the S7-1500 automation system, TLS is used for secure communication for the following certificate-based applications:

- Web server (HTTPS protocol variant)
- Secure Open User Communication (OUC) including secure email (TMAIL_C instruction)
- Secure PG/HMI communication

TLS takes care of the authenticity, confidentiality and integrity of the communication between client and server for the listed applications, for example, between the web server of CPU and web browsers, which, for example, have to display a diagnostics web page of the CPU.

The OPC UA server and OPC UA client applications do not actually directly use TLS, but the cryptographical processes used are comparable.

TLS is continuously being further developed, resulting in various TLS versions, which are distinct with regard to the cipher suites (standardized collection of cryptographic methods) supported and the performance.

The Internet Engineering Task Force (IETF) is responsible for the description of the TLS protocol. The following correlation applies:

- TLS 1.3 corresponds to RFC 8446
- TLS 1.2 corresponds to RFC 5246

In addition, not every device supports all the cryptographic methods defined in the RFCs. Therefore, after establishing the connection, the client and server negotiate a method that both support (Handshake) as well as the parameters to be used.

Supported TLS versions S7-1500

The following table shows, which TLS versions are supported in a given CPU firmware version.

CPU firmware version	Supported TLS version
V3.0	TLS 1.2, TLS 1.3
V2.9	TLS 1.2, TLS 1.3
V2.8 ... V2.0	TLS 1.2

Supported encryption methods and parameters for creating certificates

To generate the public key for a new certificate, in the TIA Portal, set the encryption method and encryption parameters. These certificate parameters are device-dependent and dependent on the application used

One possibility: In the CPU properties, go to "Protection & Security > Certificate manager" and generate a new device certificate. You can find settings for the encryption method and the encryption parameters under "Certificate Parameters" in the "Generate Certificates" dialog.

Example: RSA 2048 stands for asymmetric RSA encryption method with cryptographic key length 2048 bits.

The following table shows the supported encryption methods and encryption parameters depending on the CPU application or services.

Encryption method/parameters S7-1500 (firmware V3.0)	Web server (HTTPS) Secure PG/HMI communication Secure OUC	OPC UA
EC prime256v1	yes	no
EC secp384r1	yes	no
EC secp256k1	no	no
RSA 1024	yes	yes
RSA 2048	yes	yes
RSA 4096	yes	yes
RSA 8192	no	no

4.6.1.3 Confidentiality through encryption

Message encryption is an important element of data security. When encrypted messages are intercepted by third parties during communication, these potential eavesdroppers cannot access the information they contain.

There is a wide range of mathematical processes (algorithms) for encrypting messages.

All algorithms process a "key" parameter to encrypt and decrypt messages.

- Algorithm + key + message => encrypted message
- Encrypted message + key + algorithm => (decrypted) message

Symmetric encryption

The central aspect of symmetric encryption is that both communication partners use the same key for message encryption and decryption, as shown in the figure below. Bob uses the same key for encryption as Alice uses for decryption. In general, we also say that the two sides share the secret key with which they encrypt or decrypt a message as a secret.

① Bob encrypts his message with the symmetric key

② Alice decrypts the encrypted message with the symmetric key

Figure 4-7 Symmetric encryption

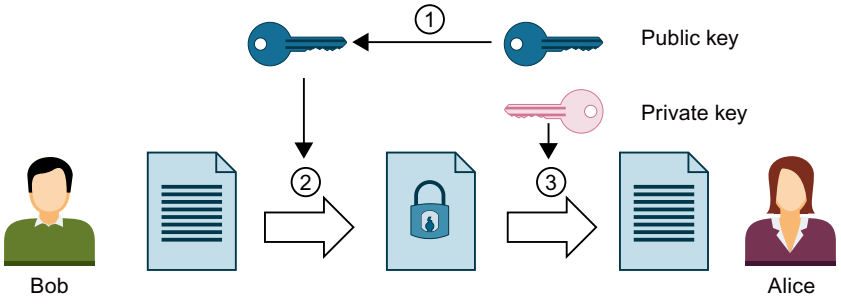
The process can be compared to a briefcase to which the sender and recipient have the same key, which both locks and opens the case.

- Advantage: Symmetric encryption algorithms (such as AES, Advanced Encryption Algorithm) are fast.
- Disadvantages: How can the key be sent to a recipient without getting into the wrong hands? This is a key distribution problem. If enough messages are intercepted, the key can also be worked out and must therefore be changed regularly.

If there are a large number of communication partners, there is also a large number of keys to distribute.

Asymmetric encryption

Asymmetric encryption works with a pair of keys consisting of one public key and one private key. Used with a PKI, it is also known as Public Key cryptography or simply PKI cryptography. A communication partner, Alice in the figure below, has a private key and a public key. The public key is provided to the public, in other words any potential communication partner. Anyone with the public key can encrypt messages for Alice. In the figure below, this is Bob. Alice's private key, which she must not disclose, is used by Alice to decrypt an encrypted message addressed to her.



① Alice provides Bob with her public key. No precautionary measures are required to this purpose: Anyone can use the public key for messages to Alice if they are sure that it is actually Alice's public key.

② Bob encrypts his message with Alice's public key.

③ Alice decrypts the encrypted message from Bob with her private key. As only Alice has the private key and never discloses it, only she can decrypt the message. With her private key, she can decrypt any message encrypted with her public key - not only messages from Bob.

Figure 4-8 Asymmetric encryption

The system can be compared to a mailbox into which anyone can put a message, but from which only the person with the key can remove messages.

- Advantages: A message encrypted with a public key can only be decrypted by the owner of the private key. As another (private) key is required for decryption, it is also much harder to work out the decryption key on the basis of large numbers of encrypted messages. This means that the public key does not have to be kept strictly confidential, unlike with symmetric keys.

  Another advantage is easier distribution of public keys. No specially secured channel is required in asymmetric cryptography to transfer the public key from the recipient to the sender encrypting the messages. Less work is thus required in managing the keys than would be the case in symmetric encryption procedures.

- Disadvantages: Complex algorithm (e.g. RSA, named after the three mathematicians Rivest, Shamir and Adleman), and therefore poorer performance than with symmetric encryption.

Encryption processes in practice

In practice, for example with a CPU Web server and Secure Open User Communication, the TLS protocol is used below the relevant application layer. Application layers are HTTP or SMTP, for example, as detailed above.

TLS (Transport Layer Security) uses a combination of asymmetric encryption and symmetric encryption (hybrid encryption) for secure data transfer, for example, over the Internet, and uses the following subprotocols:

- TLS Handshake Protocol, responsible for authentication of communication partners and negotiation of the algorithms and keys to be used for subsequent data transfer on the basis of asymmetric encryption.
- TLS Record Protocol, responsible for encryption of user data with symmetric encryption and data exchange.

Both asymmetric and symmetric encryption are considered secure encryption schemes - there is basically no difference in security between the two procedures. The degree of security depends on parameters such as the selected key length.

Abuse of encryption

You cannot tell what identity is assigned to a public key from the bit string. A fraud could provide their public key and claim to be someone else. If a third party then uses this key thinking that they are addressing their required communication partner, confidential information could end up with the fraud. The fraud then uses their private key to decrypt the message that was not intended for them, and sensitive information falls into the wrong hands.

To prevent this type of abuse, the communication partners must be confident that they are dealing with the right communication partner. This trust is established by using digital certificates in a PKI.

4.6.1.4 Authenticity and integrity through signatures

Attacks from programs that intercept communication between the server and client and act as if they themselves were client or server, are called man-in-the-middle attacks. If the false identity of these programs is not detected, they can obtain important information about the S7 program, for example, or set values in the CPU and attack a machine or plants. Digital certificates are used to avoid such attacks.

Secure communication uses digital certificates that meet the X.509 standard of the International Telecommunication Union (ITU). This allows the identity of a program, a computer or an organization to be checked (authenticated).

How certificates establish trust

The main role of X.509 certificates is to bind an identity with the data of a certificate subject (for example, e-mail address or computer name) to the public key of the identity. Identities can be people, computers or machines.

Certificates are issued by certificate authorities (Certificate Authority, CA) or by the subject of a certificate itself. PKI systems specify how users can trust the certificate authorities and the certificates that they issue.

The certificate process:

1. Anyone wishing to own a certificate submits a certificate application to a registration authority linked to the certificate authority.
2. The certificate authority assesses the application and applicant on the basis of set criteria.
3. If the identity of the applicant can be clearly established, the certificate authority confirms that identity by issuing a signed certificate. The applicant has now become the certificate subject.

The figure below is a simplified overview of the process. It does not show how Alice can check the digital signature.

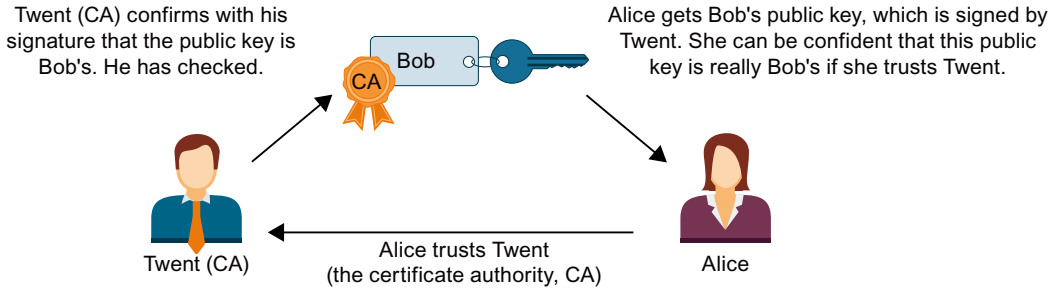


Figure 4-9 Signing of a certificate by a certificate authority

Self-signed certificates

Self-signed certificates are certificates whose signature comes from the certificate subject and not from an independent certificate authority.

Examples:

- You can create and sign a certificate yourself, for example, to encrypt messages to a communication partner. In the example above, Bob (instead of Twent) could himself sign his certificate with his private key. Using Bob's public key, Alice can check that the signature and public key from Bob match. This procedure is sufficient for simple internal plant communication that is to be encrypted.
- A root certificate is, for example, a self-signed certificate, signed by the certificate authority (CA), that contains the public key of the certificate authority.

Features of self-signed certificates

The "CN" (Common Name of Subject) for the certificate subject and "Issuer" attributes of self-signed certificates are identical: You have signed your certificate yourself. The field "CA" (Certificate Autority) must be set to "False"; the self-signed certificate should not be used to sign other certificates.

Self-signed certificates are not embedded in a PKI hierarchy.

Certificate content

A certificate to the X.509 V3 standard, the standard that is also used by STEP 7 and the S7-1500 CPUs, consists primarily of the following elements:

- Public key
- Details of the certificate subject (i.e. the holder of the key), for example, the Common Name (CN) of Subject .
- Attributes such as serial number and validity period
- Digital signature from the certificate authority (CA) confirming that the information is correct.

There are also extensions, for example:

- Specification of what the public key may be used for (Key Usage), for example, signing or key encryption.
  When you create a new certificate with STEP 7, for example in the context of Secure Open User Communication, select the correct entry from the list of possible usages, e.g. "TLS".
- Specification of a Subject Alternative Name (SAN), which is used in secure communication with Web servers (HTTP over TLS), for example, to ensure that the certificate in the address bar of the Web browser also belongs to the Web server specified in the URL.

How signatures are generated and verified

Asymmetric key usage ensures that certificates can be verified: The example of the "MyCert" certificate illustrates the "Sign" and "Verify signature" processes.

Generating a signature:

1. The issuer of the "MyCert" certificate generates a hash value from the certificate data using a specific hash function (for example SHA-1, Secure Hash Algorithm).
   The hash value is a bit string of a constant length. The advantage of the constant length of the hash value is that it always takes the same amount of time to sign.
2. Using the hash value generated in this way and the private key, the issuer of the certificate then generates a digital signature. The RSA signature scheme is often used.
3. The digital signature is saved in the certificate. The certificate is now signed.

Verifying a signature:

1. The authenticator of the "MyCert" certificate obtains the certificate of the issuer and thus the public key.
2. A new hash value is formed from the certificate data with the same hash algorithm that was used for signing (for example SHA-1).
3. This hash value is then compared with the hash value that is determined by means of the public key of the certificate issuer and the signature algorithm for checking the signature.
4. If the signature check produces a positive result, both the identity of the certificate subject as well as the integrity, meaning authenticity and genuineness, of the certificate content are proven. Anyone who has the public key, i.e. the certificate from the certificate authority, can check the signature and thus recognize that the certificate was actually signed by the certificate authority.

The figure below shows how Alice uses the public key in the certificate from Twent (who represents the certificate authority, CA) to verify the signature on Bob's public key. All that is required for verification is therefore the availability of the certificate from the certificate authority at the moment of checking. The validation itself is executed automatically in the TLS session.

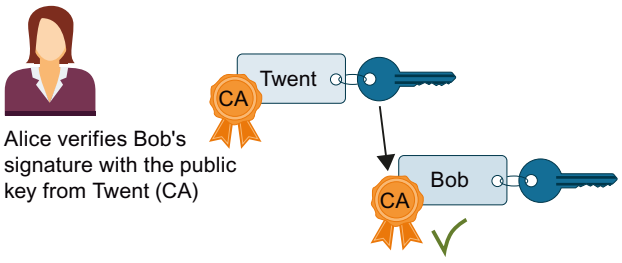


Figure 4-10 Verification of a certificate with the public key of the certificate of a certificate authority

Signing messages

The method described above for signing and verifying certificates also uses the TLS session for signing and verifying messages:

If a hash value is generated by a message and this hash value is encrypted with the private key of the sender and attached to the original message, the receiver of the message is able to check the integrity of the message. The recipient decrypts the hash value with the public key of the sender, puts together the hash value from the message received and compares the two values. If the values are not the same, the message or the encrypted hash value has been tampered with on the way.

Chain of certificates to root certificate

The certificates of a PKI are often organized hierarchically: The top of the hierarchy is formed by root certificates. Root certificates are certificates that are not signed by a higher-level certificate authority. The certificate subject and certificate issuer of root certificates are identical. Root certificates enjoy absolute trust. They form the "anchor" of trust and must therefore be known to the receiver as trusted certificates. They are stored in an area provided for trusted certificates.

Depending on the PKI, the function of root certificates is, for example, to sign certificates from lower-level certificate authorities, so-called intermediate certificates. This transfers the trust from the root certificate to the intermediate certificate. An intermediate certificate can sign a certificate just like a root certificate; both are therefore referred to as "CA certificates".

This hierarchy can be continued over multiple intermediate certificates until the end-entity certificate. The end-entity certificate is the certificate of the user who is to be identified.

The validation process runs through the hierarchy in the opposite direction: As described above, the certificate issuer is established and the signature checked with the issuer's public key, then the certificate of the higher-level certificate issuer is established along the entire chain of trust to the root certificate.

Conclusion: The chain of intermediate certificates to the root certificate, the certificate path, must be available in every device that is to validate an end-entity certificate of the communication partner, irrespective of the type of secure communication that you configure.

4.6.2 Managing certificates

4.6.2.1 What you should know about the certificate management

This section shows the available certificate management options of an S7-1500 CPU depending on the service (CPU application) used and on the versions of the TIA Portal / the CPU firmware.

Overview of certificate management options

Since TIA Portal version V14 and CPU firmware version V2.0, it has been possible to manage certificates for secure communications for different services of S7-1500 CPU in the TIA Portal and load them in the CPU.

TIA Portal as of version V17, along with the S7-1500 CPUs as of version V2.9 support another option of certificate management: With GDS push methods, you can transfer or renew certificates in the CPU at runtime without having to reload the CPU.

Using the same route, as of TIA Portal version V18, you can also transfer web server certificates for a S7-1500 CPU (as of firmware V3.0).

The following table provides an overview of the certificate management options depending on the service used and the TIA Portal firmware version or CPU firmware version.

Service	Certificate management with TIA Portal (TIA Portal version / S7-1500 CPU-FW version)	Certificate management with OPC UA GDS push methods (TIA Portal version / S7-1500 CPU-FW version)
Web server	as of V14 / as of V2.0	as of V18 / as of V3.0
Secure OUC communication	as of V14 / as of V2.0	-
OPC UA server	as of V14 / as of V2.0	as of V17 / as of V2.9
OPC UA client	as of V15.1 / as of V2.6	-
Secure PG/HMI Communication	as of V17 / as of V2.9	-

More information

Click here for a description of the certificate management with GDS-push methods: Certificate management via Global Discovery Server (GDS) (Page 180)

4.6.2.2 Certificate management with TIA Portal

STEP 7 as of version V14 together with the S7-1500 CPUs as of firmware version 2.0 support the Internet PKI (RFC 5280) in as far as an S7-1500 CPU is able to communicate with devices that also support the Internet PKI.

The usage of X.509 certificates for verifying certificates as described in the preceding sections, for example, is a result of this.

STEP 7 as of V14 uses a PKI similar to Internet PKI. Certificate Revocation Lists (CRLs), for example, are not supported.

Creating or assigning certificates in the TIA Portal

You create certificates for various applications in STEP 7 for devices with security properties, such as an S7-1500 CPU as of firmware V2.0.

The following areas in the Inspector window of the CPU allow the creation of new certificates or the selection of existing ones:

- "Web server > Security" - for the generation and assignment of Web server certificates.
- "Protection & Security > Connection mechanisms" - for the generation or assignment of PLC communication certificates (Secure PG/HMI communication, as of TIA Portal V17).
- "Protection & Security > Certificate manager" - for the generation and assignment of all types of certificates. TLS certificates for Secure Open User Communication are preset for the generation of certificates.
- "OPC UA > Server > Security" - for the generation or assignment of OPC UA server certificates.

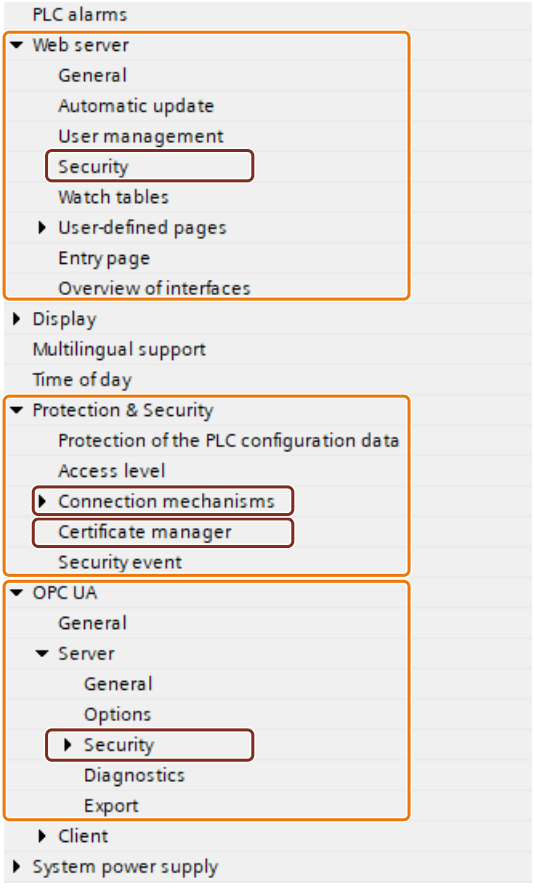


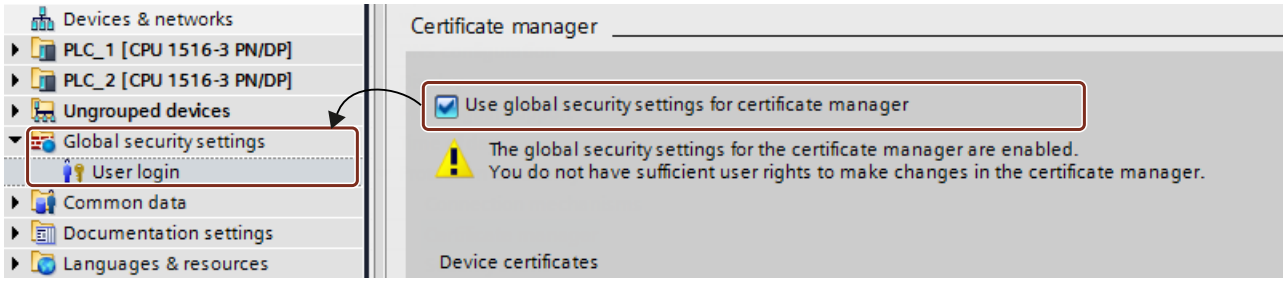
Figure 4-11 Security settings for an S7-1500 CPU in STEP 7

Special features of the section "Protection & Security > Certificate manager"

Only in this section of the Inspector window do you switch between the global, i.e. project-wide, and the local, i.e. device-specific, certificate manager (option "Use global security settings for certificate manager"). The option decides whether you have access to all the certificates in the project or not.

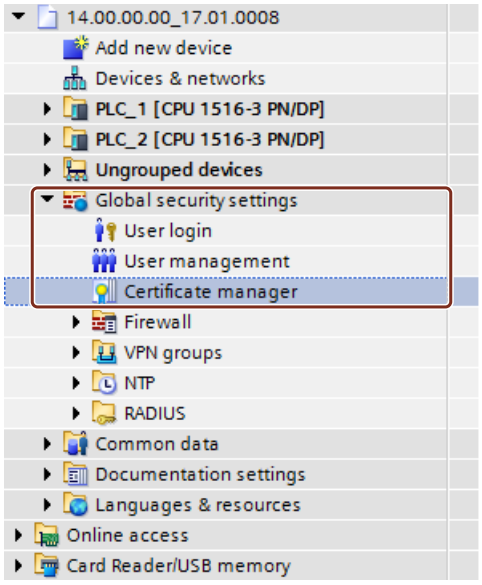
- If you do **not** use the certificate manager in the global security settings, you only have access to the local certificate memory of the CPU. You do not have access, for example, to imported certificates or root certificates. Without these certificates only a restricted functionality is available. You can, for example, only generate self-signed certificates.
- If you use the certificate manager in the global security settings and you are logged on as an administrator, you have access to the global, project-wide certificate memory. You can, for example, assign imported certificates to the CPU, or create certificates that are issued and signed by the project CA (certificate authority of the project).

The figure below shows how the "Global security settings" are shown in the project tree after the "Use global security settings for certificate manager" option has been activated in the Inspector window of the CPU.



When you double-click "User login" in the project tree below the global security settings and log in, a line called "Certificate manager" is displayed, among other data.

When you double-click the "Certificate manager" line, you obtain access to all the certificates in the project, divided into the tabs "CA" (certificate authorities), "Device certificates" and "Trusted certificates and root certificate authorities".



Private keys

STEP 7 generates private keys while generating device certificates and server certificates (end-entity certificates). The location where the private key is stored encrypted depends on the use of the global security settings for the certificate manager:

- If you use global security settings, the private key is stored encrypted in the global (project-wide) certificate memory.
- If you do not use global security settings, the private key is stored encrypted in the local (CPU-specific) certificate memory.

The existence of the private key, which is required to decrypt data, for example, is displayed in the "Private key" column of the "Device certificates" tab of the certificate manager in the global security settings.

When the hardware configuration is loaded, the device certificate, the public key as well as the private key are loaded into the CPU.

NOTICE
Enabling the "Use global security settings for certificate manager" option - Consequences The "Use global security settings for certificate manager" option influences the previously used private key: If you have already created certificates without using the certificate manager in the global security settings and then change the option for using the certificate manager, the private keys are lost and the certificate ID can change. A warning draws your attention to this fact. Therefore specify at the beginning of the project configuration which option is required for the certificate manager.

4.6.2.3 Examples for the management of certificates.

As explained in the preceding sections, certificates are required for every type of secure communication. The following section shows as an example how you handle the certificates with STEP 7 so that the requirements for Secure Open User Communication are fulfilled.

The devices which are involved at the respective communication partners are differentiated below. The respective steps for supplying the required certificates to the communications participants are described. An S7-1500 CPU or an S7-1500 Software Controller as of firmware version 2.0 is always required.

The general rule is:

While a secure connection is being established (handshake"), the communication partners as a rule only communicate their end-entity certificates (device certificates).

Therefore the CA certificates required to verify the transmitted device certificate must be located in the certificate memory of the respective communication partner.

NOTE

The current date/time must be set in the CPU.

When using secure communication (for example, HTTPS, secure OUC, OPC UA), make sure that the corresponding modules have the current time of day and the current date. Otherwise, the modules will evaluate the certificates used as invalid and secure communication will not work.

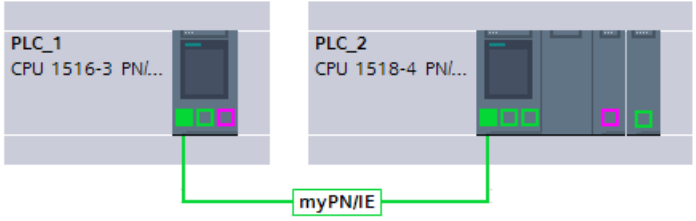
Secure Open User Communication between 2 S7-1500 CPUs

Two S7-1500 CPUs PLC_1 and PLC_2 are to exchange data with each other via Secure Open User Communication.

You generate the required device certificates with STEP 7 and assign them to the CPUs as described below.

STEP 7 project certificate authorities (CA of the project) are used to sign the device certificates.

The certificates are to be referenced by their certificate ID in the user program (TCON communication instruction in combination with the associated system data type, for example TCON_IPV4_SEC). STEP 7 assigns the certificate ID automatically during the generation or creation of certificates.



Procedure

STEP 7 automatically loads the required CA certificates together with the hardware configuration to the participating CPUs so that the requirements for certificate verification

exist for both CPUs. You therefore only have to generate the device certificates for the respective CPU; STEP 7 does the rest for you.

1. Mark PLC_1 and activate the "Use global security settings for certificate manager" option in the "Protection & Security" section.
2. Log in as a user in the project tree in the "Global security settings" section. For a new project, the "Administrator" role is planned for the first login.
3. Return to the PLC-1 in the "Protection & Security" section. Click in an empty line in the "Certificate subject" column in the "Device certificates" table to add a new certificate.
4. In the drop-down list for selecting a certificate click the "Add" button.
   The "Create Certificate" dialog opens.
5. Leave the default settings in this dialog. They are tailored to the usage of Secure Open User Communication (usage: TLS).
   Tip: Supplement the default name of the certificate subject, in this case the CPU name. In order to differentiate you better leave the default CPU name in case you have to manage a large number of device certificates.
   Example: PLC_1/TLS becomes PLC_1-SecOUC-Chassis17FactoryState.
6. Compile the configuration.
   The device certificate and the CA certificate are part of the configuration.
7. Repeat the steps described above for PLC_2.

In the next step you have to create the user programs for the data exchange and load the configurations together with the program.

Using self-signed certificates instead of CA certificates

When creating device certificates you can select the "Self-signed" option. You can create self-signed certificates without being logged in for the global security settings. This procedure is not recommended because the resulting certificates do not exist in the global certificate memory and can therefore not be assigned directly to a partner CPU.

As described above, you should select the name of the certificate subject with care so that the right certificate can be assigned to a device without any doubt.

Verification with the CA certificates of the STEP 7 project is not possible for self-signed certificates. To ensure that self-signed certificates can be verified you have to include the self-signed certificates of the communication partner into the list of trusted partner devices for each CPU. To this purpose you must have activated the "Use global security settings for certificate manager" option and be logged in as a user in the global security settings.

Proceed as follows to add the self-signed certificate of the communication partner of the CPU:

1. Mark PLC_1 and navigate to the "Certificates of partner devices" table in the "Protection & Security" section.
2. Click in an empty line in the "Certificate subject" column in the "Device certificates" table to add a new certificate.
3. Select the self-signed certificate of the communication partner from the drop-down list and confirm the selection.

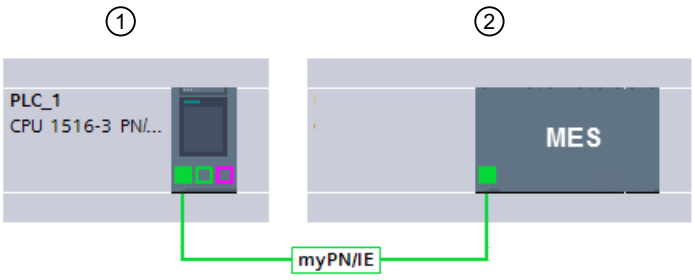
In the next step you have to create the user programs for the data exchange and load the configurations together with the program.

Secure Open User Communication between S7-1500 CPU as a TLS client and an external device as a TLS server

Two devices are to exchange data with each other via TLS connection or TLS session, for example, exchanging recipes, production data or quality data:

- An S7-1500-CPU (PLC_1) as TLS client; the CPU uses Secure Open User Communication
- An external device (for example a Manufacturing Execution System (MES)) as TLS server

The S7-1500 CPU establishes the TLS connection / session to the MES system as TLS client.



① TLS client

② TLS server

The S7-1500 CPU requires the CA certificates of the MES system to authenticate the TLS server: The root certificate and, if appropriate, the intermediate certificates for verifying the certificate path.

You have to import these certificates into the global certificate memory of the S7-1500 CPU.

Proceed as follows to import certificates of the communication partner:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the appropriate table (trusted certificates and root certificate authorities) for the certificate to be imported.
3. Right-click in the table to open the shortcut menu. Click "Import" and import the required certificate or the required CA certificates.
   Through the import the certificate has a certificate ID assigned to it and can be assigned to a module in the next step.
4. Mark PLC_1 and navigate to the "Certificates of partner devices" table in the "Protection & Security" section.
5. Click in an empty line in the "Certificate subject" column to add the imported certificates.
6. Select the required CA certificates of the communication partner from the drop-down list and confirm the selection.

Optionally the MES system can also request a device certificate of the CPU to authenticate the CPU (i.e., the TLS client). In this case, the CA certificates of the CPU must be made available to the MES system. The prerequisite for importing the certificates into the MES system is a preceding export of the CA certificates from the STEP 7 project of the CPU. Follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the matching table (CA certificate) for the certificate to be exported.
3. Right-click the selected certificate to open the shortcut menu.
4. Click "Export".
5. Select the export format of the certificate.

In the next step you have to create the user programs for the data exchange and load the configurations together with the program.

Secure Open User Communication between an S7-1500 CPU as TLS server and an external device as TLS client

If the S7-1500 CPU acts as TLS server and the external device, for example an ERP system (Enterprise Resource Planning System) establishes the TLS connection / session, you require the following certificates:

- For the S7-1500 CPU, you generate a device certificate (server certificate) with a private key and download it with the hardware configuration into the S7-1500 CPU. You use the "Signed by certificate authority" option when generating the server certificate.
  The private key is required for the key exchange as explained in the figure for the example "HTTP over TLS".
- You have to export the CA certificate of the STEP 7 project for the ERP system and import / load it into the ERP system. With the CA certificate the ERP system verifies the server certificate of the S7-1500 that was transferred from the CPU to the ERP system during the establishment of the TLS connection / session.

① TLS server

② TLS client

Figure 4-12 Secure OUC between an S7-1500 CPU and ERP system

The required steps are described in the preceding sections.

Secure Open User Communication to a mail server (SMTP over TLS)

An S7-1500 CPU can establish a secure connection to an e-mail server with the communication instruction TMAIL-C.
The system data types TMail_V4_SEC and TMail_QDN_SEC allow you to determine the partner port of the e-mail server and thus to reach the e-mail server via "SMTP over TLS".

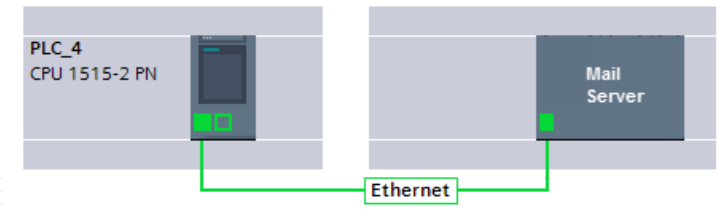


Figure 4-13 Secure OUC between a S7-1500 CPU and a mail server

Requirement for secure email connection is the importing of the root certificate and the intermediate certificates of the mail server (provider) into the global certificate memory of the S7-1500 CPU. By means of these certificates the CPU can check the server certificate that is sent by the mail server during the establishment of the TLS connection / session.

Proceed as follows to import certificates of the mail server:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the appropriate table (trusted certificates and root certificate authorities) for the certificate to be imported.
3. Right-click in the table to open the shortcut menu. Click "Import" and import the required certificate or the required CA certificates.
   As a result of the import, the certificate has a certificate ID assigned to it and can be assigned to a module in the next step.
4. Mark PLC_1 and navigate to the "Certificates of partner devices" table in the "Protection & Security" section.
5. Click in an empty line in the "Certificate subject" column to add the imported certificates.
6. Select the required CA certificates of the communication partner from the drop-down list and confirm the selection.

In the next step you have to create the user programs for the email client function of the CPU and load the configurations together with the program.

4.6.2.4 How communication with certificates works: HTTP over TLS

The following paragraphs show how the mechanisms described are used to establish a secure communication between a Web browser and the Web server of an S7-1500 CPU.
Initially the changes for the "Permit access only with HTTPS" option in STEP 7 are described. As of STEP 7 V14 you have the possibility to influence the server certificate of the Web server of an S7-1500 CPU as of firmware V2.0: The server certificate is generated as of these versions with STEP 7.
In addition it illustrates the processes that are executed when a website of the CPU Web server is called with a Web browser of a PC through an encrypted HTTPS connection.

Using Web server certificates for S7-1500 CPUs, FW V2.0 or higher

For S7-1500 CPUs with a firmware version before V2.0, you were able to set "Permit access only with HTTPS" when setting the Web server properties, without specific requirements applying.
You did not have to handle certificates for these CPUs; the CPU automatically generates the certificates required for the Web server.
For S7-1500 CPUs as of firmware V2.0, STEP 7 generates the server certificate (end-entity certificate) for the CPU. You assign a server certificate to the Web server in the properties of the CPU (Web server > Security).
Because a server certificate name is always preset, there is no change to the easy configuration of the Web server: You activate the Web server. The "Permit access only with HTTPS" option is enabled by default - STEP 7 generates a server certificate with the default name during compiling.
Irrespective of whether you use the certificate manager in the global security settings or not: STEP 7 has all the information required to generate the server certificate.

In addition, you have the possibility to determine the properties of the server certificate, for example, the name or the validity period.

NOTE

The current date/time must be set in the CPU.

When using secure communication (for example, HTTPS, secure OUC, OPC UA), make sure that the corresponding modules have the current time of day and the current date. Otherwise, the modules will evaluate the certificates used as invalid and secure communication will not work.

Loading the Web server certificate

The server certificate generated by STEP 7 is then automatically also loaded to the CPU when the hardware configuration is loaded.

- If you use the certificate manager in the global security settings, the certificate authority of the project (CA certificate) signs the server certificate of the Web server: During loading the CA certificate of the project is loaded as well automatically.
- If you do not use the certificate manager in the global security settings, STEP 7 generates the server certificate as a self-signed certificate.

When you address the Web server of the CPU over the IP address of the CPU, a new server certificate (end-entity certificate) must be generated and loaded with each change in the IP address of an Ethernet interface of the CPU. This is necessary because the identity of the CPU changes with the IP address – and the identity requires a signature in accordance with the PKI rules.

You can avoid this problem by addressing the CPU with a domain name instead of its IP address, for example "myconveyer-cpu.room13.myfactory.com". For this purpose, you have to manage the domain names of the CPU via a DNS server.

Supplying a Web browser with a CA certificate of the Web server

In the Web browser the user who accesses the websites of the CPU through HTTPS should install the CA certificate of the CPU. If no certificate is installed, a warning is output recommending that you do not use the page. To view this page, you must explicitly "Add an exception".

The user receives the valid root certificate for download from the "Intro" Web page of the CPU Web server under "Download certificate".

STEP 7 offers a different possibility: Export the CA certificate of the project with the certificate manager into the global security settings in STEP 7. Subsequently import the CA certificate into the browser.

Course of the secure communication

The figure below shows, in simplified terms, how communication is established ("handshake") focusing on the negotiation of keys used for data exchange (here with HTTP over TLS).

However, the course can be applied to all communication options that are based on the usage of TLS, i.e. also for Secure Open User Communication (see Basics for secure communication).

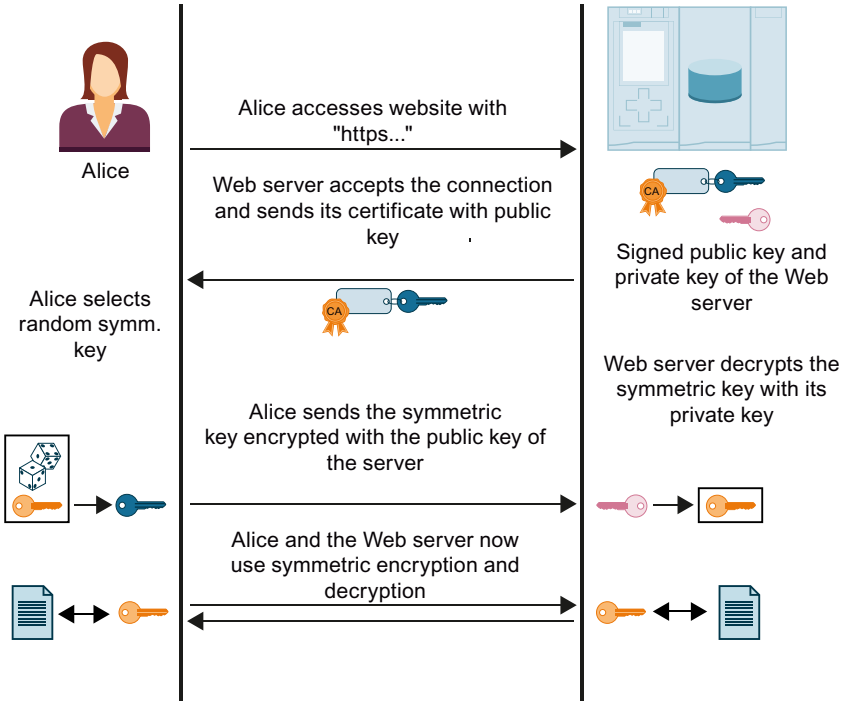


Figure 4-14 Handshake with https

The figure does not show the measures taken at Alice's end (browser) to verify the certificate sent by the Web server. Whether Alice can trust the Web server certificate received and therefore the identity of the Web server, and can accept the exchange of data, depends on positive verification.

The steps for verifying the authenticity of the Web server:

1. Alice must know the public keys of all relevant certificate authorities, which means she requires the complete certificate chain to verify the Web server certificate (i.e. the end-entity certificate of the Web server):

   Alice will generally have the required root certificate in her certificate memory. When a Web browser is installed, a range of trusted root certificates is also installed. If she does not have the root certificate, she must download it from the certificate authority and install it in the certificate authority of the browser. The certificate authority can also be the device on which the Web server is located.

   You have the following options for obtaining the intermediate certificates:

   – The server itself sends the required intermediate certificates to Alice along with its end-entity certificate – in the form of a signed message so that Alice can verify the integrity of the certificate chain.
   – The certificates often contain the URLs of the certificate issuer. Alice can load the required intermediate certificates from these URLs.

   When you work with certificates in STEP 7 it is always assumed that you have imported the intermediate certificates and the root certificate into the project and assigned them to the module.
2. Alice validates the signatures in the certificate chain with the public keys of the certificates.

3. The symmetric key must be generated and transferred to the Web server.
4. If the Web server is addressed by its domain name, Alice also verifies the identity of the Web server in accordance with the Internet PKI rules defined in RFC 2818. She is able to do this because the URL of the Web server, in this case the "Fully Qualified Domain Name" (FQDN), is saved in the end-entity certificate of the Web server. If the certificate entry in the "Subject Alternative Name" field corresponds to the entry in the address bar of the browser, everything is fine.

The process continues with the exchange of data with the symmetric key, as shown in the figure above.

4.6.3 Requirements for secure communication

4.6.3.1 Protection of confidential configuration data

As described in the basic information on secure communication, the proper functioning of certificate-based protocols requires private keys that must be protected as best as possible.
As of STEP 7 V17, you can use a password to protect these keys and other data worth protecting: The password to protect confidential PLC configuration data.
It is possible to do without the password if you have implemented measures to prevent unauthorized access to the TIA Portal project and the configuration of the CPU.
independently of whether you assign a password or not: The TIA Portal generates a key information that provides for the protection of the confidential PLC configuration data. This password has no influence on the secure communication process. However, the complexity of the password for the protection of confidential PLC configuration data determines how well the private keys, for example, are protected.
The presence of key information is a prerequisite for secure communication such as TLS-based secure PG/HMI communication: The CPU can handle certificates which are required for Secure Communication only if this key information is available.
The following figure shows the described contexts.

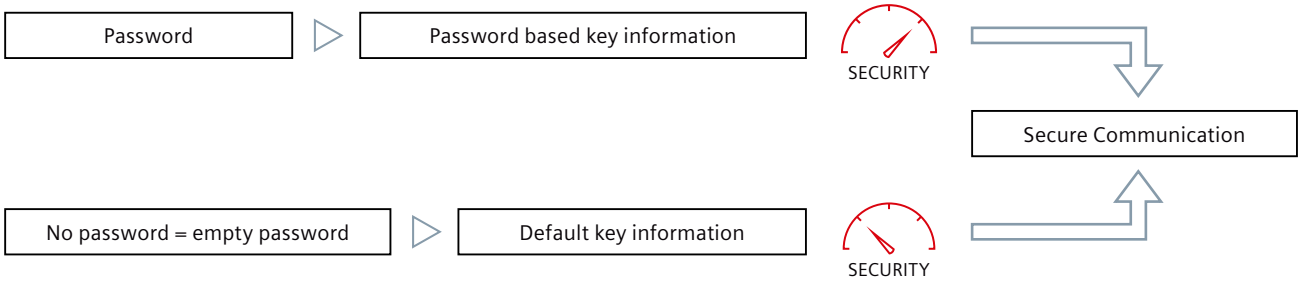


Figure 4-15 Contexts for protecting confidential configuration data

Security settings wizard

When you add a CPU to the project that supports secure PG/HMI communication in the TIA Portal from the hardware catalog, a wizard starts for the security settings of the CPU.

The wizard guides you step by step through the following CPU settings:

- Password to protect confidential PLC configuration data
- PG/PC and HMI communication mode
- Access level

Each of these settings is explained in detail in the wizard. At the end, all settings are once again summarized in an overview.

The wizard also starts, for example, when you replace a module in the network view of the TIA Portal and the new CPU, unlike the replaced CPU, supports secure PG/HMI communication.

All settings in the wizard are applied in the Inspector window (CPU properties).

You can start the wizard at any time using a Start button in the "Protection & Security" area of the CPU properties.

Requirement

- TIA Portal as of version V17
- CPU supports secure PG/HMI communication (for S7-1500 CPUs as of firmware version 2.9)
- The CPU is not yet loaded or the CPU is reset to factory settings with the option "Delete password for protection of confidential PLC configuration data"

Procedure

1. Open the CPU properties in the network view or in the device view.
2. Navigate to the area "Protection & Security > Protection of the PLC configuration data".
   Result: The "Protect confidential PLC configuration data" option is enabled first and the empty field for password entry is highlighted in red.
3. Configure the password (recommended) via the "Set" button or disable the "Protect confidential PLC configuration data" option.
4. Complete the configuration and create the user program.
5. Load the CPU.
   When loading the hardware configuration, you will be asked once to re-enter the password.
   Background: The configured password is used in the TIA Portal to generate the key information to protect confidential configuration data and thus to protect this data. For security reasons, however, neither the password nor the key information is saved in the project. In order for the key information to reach the CPU, it is re-generated when the hardware configuration is loaded, so that the password must be entered once at this point.

Certificate-based communication also between PG/HMI and CPU

Because, as of TIA Portal version V17 and CPU firmware version V2.9 (S7-1500) or V4.5 (S7-1200), the PG/HMI communication is also certificate-based, you will be prompted to accept the server certificate in the course of the commissioning.

Tips and rules for password management

- Manage your passwords in a password manager.
- To check the newly entered passwords for compliance and, for example, prevent trivial passwords, use the settings for checking the password policies in the TIA Portal:
  - In the project tree, navigate to the area "<Project name> > Security settings > Settings" area and select the "Password policies" area.
  - Specify, for example, the minimum number of characters the password must have or the minimum number of special characters.
- You do not have to assign different passwords for each CPU in a system or machine. If the requirements are met, you can also define the same password for a group of CPUs. This strategy also has advantages in the replacement parts scenario: If the group password is also assigned to the replacement CPU, the workload of replacing the CPU is reduced.
  Note here the risk that if the password of one of these CPUs is compromised, all CPUs with the same password are vulnerable.
- The definition of passwords also has an impact on the replacement part case, as the password for confidential PLC configuration data must be transferred to the new (replacement) CPU in addition to the configuration (see Rules for the replacement parts scenario (Page 75)).
- With **S7-1500R/H CPUs**, the password for confidential PLC configuration data is only loaded onto one of the two CPUs during loading. In order that the sync-up process works and that the partner CPU also works properly, the password must be transferred to the partner CPU before the sync-up, using the Online and Diagnostics editor:
  - In the Online and diagnostics view, you specify the area "Password to protect confidential PLC configuration data".
  - Enter the required password and click the "Set" button.
    If the correct password has been entered, the partner CPU can use the protected PLC configuration data and start the sync-up process.

4.6.3.2 Useful information for the protection of confidential PLC configuration data

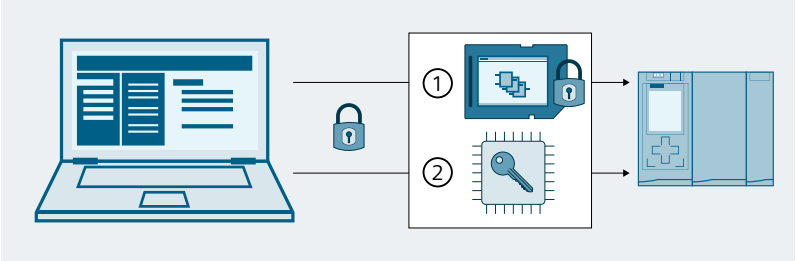
The concept for Secure Communication protected by security standards comprises the following components:

- A password-based key information that is used for protecting confidential configuration data (e.g. private keys for certificates, passwords).
- A standardized log (TLS) that ensures communication between the participants (e.g. programming device and CPU).

"Protection of confidential configuration data" principle

The following figure shows in a simplified way how confidential configuration data, for example of a standard S7-1500 CPU, is protected: The two components project and key information are placed in different memory areas when loaded for the first time. The project in the load memory (memory card), the key information in a memory area in the CPU.

For other target systems (e.g. S7-1200 CPU, software controller) with different storage concepts, the implementation is adapted to the corresponding storage concepts, but the principle is the same.



① Project with password-protected confidential configuration data (here: in load memory = memory card)

② Key information (generated from password) to use the protected confidential configuration data (here: in the memory area in the CPU)

Figure 4-16 Principle for protection of confidential configuration data

Two memory areas for more security

The charged components are related to each other like two matching puzzle pieces: The project is bound to the loaded key information, the loaded key information is bound to the password that was assigned during configuration.

Project and key information must match, otherwise the CPU will not start.

The principle of two separate memory areas also applies to the S7-1200 CPUs and S7-1500 CPU versions without a memory card, e.g. for the SW controller or PLCSim/ PLCSim Advanced. In the versions without memory card, two separate partitions are used so that the two items of information can be managed independently of one another.

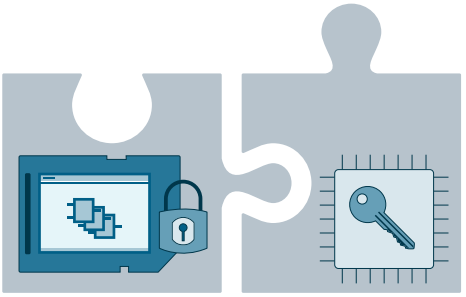


Figure 4-17 Principle of two separate memory areas

4.6.3.3 Changing your password

It makes a difference whether the CPU has already been loaded or not. If the CPU has already been loaded, it has the key information with which the password-protected PLC configuration data can be used.

Change password - configuration not yet loaded

As long as you have not yet loaded a configuration into the CPU, you can change an entered password or you can revoke the activation of the password protection.

Requirement

- The CPU is not yet loaded

Procedure

1. Open the CPU properties in the network view or in the device view.
2. Navigate to the area "Protection & Security > Protection of the PLC configuration data".
3. Click the "Change" button or deactivate the option "Protect confidential PLC configuration data".
4. Enter the previously valid password in the dialog. In case of a password change, also enter the new password and confirm the new password.

As long as you have not yet loaded a configuration into the CPU, the CPU is in a provisioning phase (see CPU behavior from loading to operational readiness (Page 97)) and you can load any valid configuration with your configured password.

Change password - configuration is already loaded

If the CPU has already been loaded with a configuration and the configuration is protected with a password for confidential PLC configuration data, you must first either reset the CPU to the factory settings and delete the password for confidential PLC configuration data in the CPU or delete it directly online and then set it.

Requirements

- You have write access to the CPU
- The CPU must be in STOP mode.

Procedure

1. Select the CPU in the network view.
2. Select the "Online & Diagnostics" command from the shortcut menu.
3. If you also change the project on the memory card, i.e. then want to reload the configuration:
   - Select the "Reset to factory settings" area in the opened online and diagnostics view.
   - Activate the option "Delete password to protect confidential PLC configuration data". To avoid a repeated start-up of the CPU, also select the "Format memory card" option.
   - Then load the project with the changed configuration and the desired password.
4. If you do not have to change the project on the memory card, i.e. only the wrong password is set:
   - In the Online and diagnostics view, you specify the area "Password for the protection of confidential PLC configuration data".
   - Click the "Delete" button. If the "Delete" button is not available, no password has been set in the CPU yet.
   - Enter the required password and click the "Set" button.

   If the correct password has been entered, the CPU can use the protected PLC configuration data.

No write access to the CPU

If you do not have write access to the load memory (read access level), remove the memory card from the CPU or delete the memory card externally, e.g. in your computer, before you reset to factory settings with the option "Delete password to protect confidential PLC configuration data".

NOTE

Reset to factory settings with mode selector of the CPU

Restoring the factory settings of the CPU via the mode selector also deletes the IP address of the CPU, but not the password for protecting confidential PLC configuration data.

More information

Information on how to proceed in case of a spare part can be found in section Rules for the replacement parts scenario (Page 75).

4.6.3.4 Resetting the password

The protection of the confidential PLC configuration data can be reset. This may be necessary, for example, if you want to change the password but no longer know the current password.

Password lost - configuration not yet loaded

Since you have to enter the password when loading the CPU via TIA Portal for the first time, the CPU configuration for this CPU can no longer be used. To change the password in the CPU properties, you must also enter the previously valid password. If you forget your password, do the following:

Requirement

- The CPU is not yet loaded

Procedure

1. Open the CPU properties in the network view or in the device view.
2. Navigate to the area "Protection & Security > Protection of the PLC configuration data".
3. Click "Reset".
   Please note that the certificates of the CPU (e.g. certificates for web server, for OPC UA server, for PG/PC communication and HMI communication) may no longer be used after the reset and may have to be created again and reassigned.
   - If you use the global security settings for the certificate manager, you must reassign the certificates from the certificate manager.
   - If you do not use the global security settings for the certificate manager, you must recreate and reassign the certificates.
4. Confirm the reset of the password.

The option for the protection of confidential PLC configuration data is still activated.

Delete password – Configuration is already loaded

If the CPU has already been loaded with a configuration and the configuration is protected with a password for confidential PLC configuration data, you can, for loading a new project, delete the password for confidential PLC configuration data online and then specify a new password.

Requirements

- You have write access to the CPU
- The CPU must be in STOP mode.

Procedure

1. Select the CPU in the network view.
2. Select the "Online & Diagnostics" command from the shortcut menu.
3. In the area "Password to protect confidential PLC configuration data", click the "delete" button.
   If the "Delete" button is not available, no password has been set in the CPU yet.

NOTICE
Deleting the password for confidential configuration data If the password is deleted and a loaded project requires a corresponding password, this project may no longer work without password.

4. If required, now enter a new password via the "Set" button.
5. Perform a restart of the CPU.

More information

Information on changing the password can be found in section Changing your password (Page 68).

4.6.3.5 Assign password via SIMATIC Memory Card

If you want to transfer the password to protect confidential PLC configuration data to a CPU without using TIA Portal, you can also use a SIMATIC memory card for this function.

The use of a SIMATIC memory card is suitable for the following purposes:

- Preparing a new CPU
  If a CPU is set up again, it should be configured with a password to protect the confidential PLC configuration data. After this configuration is completed, it is possible to use another SIMATIC memory card with the desired project.
  (S7-1200 CPU: A "transfer" card with transfer job can also be used to install the program on the CPU).
- CPU has a password to protect confidential PLC configuration data, but the password does not match the project
  If the passwords are not identical, you can set the correct password with the memory card in the CPU.
  (S7-1200 CPU: equipped either with SIMATIC "transfer" card or with SIMATIC "Program" card).
- reset the password for the protection of confidential PLC configuration data in the CPU
  As preparation for a disposal of the CPU or as preparation for a new project for the CPU.

Requirement

- TIA Portal as of version V17

Basic procedure

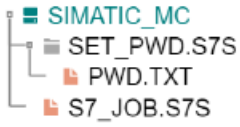
1. Creating a SIMATIC memory card with "SET PASSWORD" job
   This action creates a folder and file structure following a special pattern and writes a password for the protection of the confidential PLC configuration data as plain text to a special file on the SIMATIC memory card. See description below.
2. Insertion of a prepared SIMATIC memory card in the CPU and switch on the CPU.
   The PLC reads the password, processes it and stores the result in the internal memory. Any existing entry is overwritten.
3. Remove the SIMATIC memory card and restart the CPU.
   Results (S7-1500): While the CPU is reading the SIMATIC memory card, the LED shows the same behavior as with a firmware update. The RUN/STOP LED flashes while the CPU is setting the password. After the process has been completed successfully, the RUN/STOP LED lights up yellow and the MAINT LED flashes yellow.

The result of the operation is displayed in the diagnostics buffer as success or error message. If the password could not be set, the error LED flashes together with the other LEDs.

Creating a SIMATIC memory card with "SET PASSWORD" job

1. Create a folder in the root directory named "SET_PWD.S7S".
2. Create a text file named "PWD.TXT" with the password as text-only in the folder just created on the memory card.
3. Create a text file named "S7_JOB.S7S" in the root directory of the memory card with the content "SET_PWD".
   This file is the "Job file" for assigning a password for the protection of the confidential PLC configuration data of the PLC.

4. The file structure on the SIMATIC memory card then looks as follows:



NOTE

Safe storage of the SIMATIC memory card

Store the SIMATIC memory card in a safe location to which only authorized persons have access.

Rules and recommendations

- Setting the password must be done in a secure environment.
- The content of the text file "PWD.TXT" defines the password for the protection of confidential PLC configuration data. It must match the password that you have also assigned in the CPU configuration.
- To reset an existing password of a PLC, the text file "PWD.TXT" must be empty, i.e. the file size is 0 bytes.
- Use any text editor to create the text file. The recommended text format is "UTF-8".
- The folder and file names are not case-sensitive. However, the password itself is case-sensitive.
- Do not enter a CR/LF character at the end (PWD.TXT or S7_JOB.S7S).

4.6.3.6 Special features when backing up and restoring a CPU

You can back up a functional configuration of a CPU in the TIA Portal and access it at a later time; this means you can then restore the originally backed-up configuration. This allows you to load a modified configuration, for example, to test product enhancements, to change programs for troubleshooting in the system or you can replace components on a test basis. You can then restore the originally backed-up configuration of the CPU.

Backing up the configuration

In backing up a CPU ("Online" menu, "Load backup from online device" in TIA Portal), the password for the protection of confidential PLC configuration data is backed up as well.

Restoring the backup

When restore the backup of a CPU (menu "Online", command “Download to device” with marked backup in TIA Portal) the CPU can only communicate with a PG/PC or HMI if the following condition is fulfilled:

- After the restoration of a configuration protected with a password to protect confidential PLC configuration data, exactly this password must be present in the CPU.
  Otherwise the CPU cannot access the configuration data and therefore does not start.

Remedy

If the above error occurs, that is, the password for protecting confidential PLC configuration data does not match the backup, you must delete the password to protect confidential PLC configuration data in the CPU and then set the correct password. After a restart the CPU, the backup is functional.

4.6.3.7 Tips for error avoidance and error handling

The following description lists some use cases that may result in CPU error messages.

Diagnostic buffer provides information

The CPU detects when the password protecting confidential configuration data and the loaded configuration do not match. A message in the diagnostic buffer indicates possible causes and remedies and usually leads to a solution of the problem.

Typical "pitfalls"

You should pay attention to the following circumstances in order to avoid or correct errors:

- Configuration loaded?
  Regardless of whether you protect your confidential configuration data with a password or not: without a loaded configuration, the CPU does not leave the provisioning phase.
- You are trying to load the CPU with a configured password and the CPU has already received another password.
  Example: CPU is exchanged for another CPU from the stock. The replacement CPU was not completely reset (reset to factory settings with option "Delete password for protection of confidential PLC configuration data").
  Remedy:
  - Always prepare replacement CPUs with the appropriate option (password deleted).
  - For the configuration to be loaded, use the same password that was already used for the configuration already loaded.
  - It is also possible that the wrong project / CPU configuration was loaded. Check whether the correct CPU configuration is available.
  - Use the online function "Set password to protect confidential PLC configuration data" to delete the password or to set the same password as in the CPU configuration. Then, restart the device.

- The same error occurs if your CPU configuration does not use a password and the already loaded configuration requires a user-defined password.
  Remedy:
  – Use the online function "Set password to protect confidential PLC configuration data" to delete the password or to set the same password as in the CPU configuration. Then, restart the device.

4.6.3.8 Rules for the replacement parts scenario

The assignment of passwords to protect confidential PLC configuration data also has an impact on the replacement parts scenario.

Rules for the replacement parts scenario

Observe the following rules for the replacement parts scenario:

Configuration of the replacement CPU via TIA Portal

- A CPU as replacement for an existing CPU should not have a configuration or a configured password to protect confidential PLC configuration data.
  Advantage: You can load the project into the replacement CPU without any further preparation - regardless of whether a password is configured or not.
- If the replacement CPU has already been configured, you must reset the CPU to the factory settings with the following options set:
  – "Delete password for protection of confidential PLC configuration data"
  – "Format memory card"

Replacement CPU is supplied with configuration data via memory card

- If you have **not** assigned a password to a CPU in your project to protect confidential PLC configuration data, you can insert the memory card of the CPU to be replaced into a brand-new, unused CPU without any further action needed.
  If the replacement CPU has already been configured with a password to protect confidential PLC configuration data, you must first reset this CPU to the factory settings using the option "Delete password for protection of confidential PLC configuration data".
- If you have assigned the same password to a group of CPUs, you can also assign the group password to the replacement CPU via the TIA Portal or via an appropriately prepared memory card (see Protection of confidential configuration data (Page 65)).
  In this case you can, for example, insert a memory card with the current project into the CPU and put it into operation without any further password handling.
- If you assign different passwords to each CPU in your project, you must first set the password valid for the respective CPU with the online and diagnostics editor when using the replacement CPU (area "Set password for protection of confidential PLC configuration data", see Changing your password (Page 68)).

More information

In section Assign password via SIMATIC Memory Card (Page 71) you can read how to use the SIMATIC Memory Card to assign the password to protect confidential PLC configuration data.

4.6.4 Secure Open User Communication

4.6.4.1 Secure OUC of an S7-1500 CPU as TLS client to an external PLC (TLS server)

The following section describes how you can set up Open User Communication via TCP from an S7-1500 CPU as TLS client to a TLS server.

Setting up a secure TCP connection from an S7-1500 CPU as TLS client to a TLS server

S7-1500 CPUs as of firmware version V2.0 support secure communication with addressing via a Domain Name System (DNS).

For secure TCP communication over the domain name you need to create a data block with the TCON_QDN_SEC system data type yourself, assign parameters and call it directly at one of the instructions TSEND_C, TRCV_C or TCON.

Requirements:

- Current date and time are set in the CPU.
- Your network includes at least one DNS server.
- You have configured at least one DNS server for the S7-1500 CPU.
- TLS client and TLS server have all the required certificates.

To set up a secure TCP connection to a TLS server, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_QDN_SEC in the global data block.
   The example below shows the global data block "Data_block_1" in which the tag "DNS ConnectionSEC" of the data type TCON_QDN_SEC is defined.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	DNS Connection SEC	TCON_QDN_SEC		
3	ConnPara	TCON_QDN		parameter of the TCP connection
4	InterfaceId	HW_ANY	0	not relevant
5	ID	CONN_OUC	5	connection reference / identifier
6	ConnectionType	Byte	11	type of connection: 16#0B=11=TCP/IP, 16#13
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteQDN	String[254]	'plc_1.factory127.'	fully or partially qualified domain name of rem
9	RemotePort	UInt	4000	remote UDP / TCP port number
10	LocalPort	UInt	0	local UDP / TCP port number
11	ActivateSecureConn	Bool	true	activate the security functionality of that conn
12	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a
13	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 a
14	TLSServerCertRef	UDInt	7	for Server side: Reference to own X.509 V3 se
15	TLSClientCertRef	UDInt	0	for Client side: add id of own X.509 V3 client c

Figure 4-18 Data type TCON_QDN_SEC

3. Set the connection parameters of the TCP connection in the "Start value" column. Enter the fully qualified domain name (FQDN) of the TLS server, for example, for "RemoteQDN".

4. Set the parameters for secure communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "ExtTLSCapabilities": If you enter the value 1, the client validates the subjectAlternateName in the X.509-V3 certificate of the server to verify the identity of the server. This validation is executed in the context of the instruction.
   - "TLSServerCertRef": ID of the X.509-V3 certificate (usually a CA certificate) that is used by the TLS client to validate the TLS server authentication. If this parameter is 0, the TLS client uses all (CA) certificates currently loaded in the client certificate store to validate the server authentication.

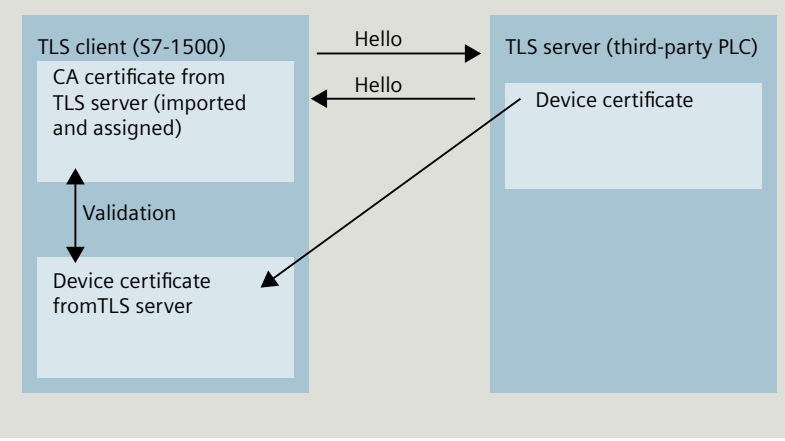


Figure 4-19 Certificate handling from the perspective of the S7-1500 as a TLS client

   - "TLSClientCertRef": ID of the own X.509-V3 certificate.
5. Create one of the instructions TSEND_C, TRCV_C or TCON in the program editor.
6. Interconnect the CONNECT parameter of one of the instructions TSEND_C, TRCV_C or TCON with the tags of the data type TCON_QDN_SEC.
   In the example below, the CONNECT parameter of the TCON instruction is interconnected with the tag "DNS connectionSEC" (data type TCON_QDN_SEC).

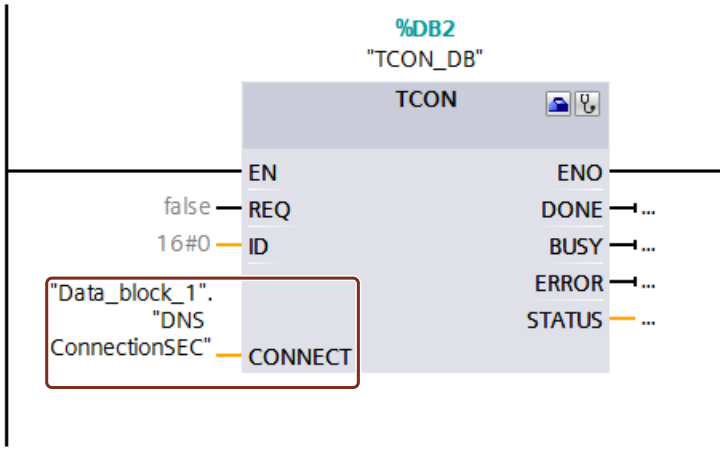


Figure 4-20 TCON instruction

Additional information

You can find more information on the TCON_QDN_SEC system data type in the STEP 7 online help.
For additional information on secure communication, refer to the section Secure Communication (Page 43).

4.6.4.2 Secure OUC of an S7-1500 CPU as TLS server to an external PLC (TLS client)

The following section describes how you can set up Open User Communication via TCP from an S7-1500 CPU as TLS server to a TLS client.

Setting up a secure TCP connection via the domain name of the communication partner

S7-1500 CPUs as of firmware version V2.0 support secure communication with addressing via a Domain Name System (DNS).
For secure TCP communication over the domain name you need to create a data block with the TCON_QDN_SEC system data type yourself, assign parameters and call it directly at one of the instructions TSEND_C, TRCV_C or TCON.
Requirements:

- Current date and time are set in the CPU.
- Your network includes at least one DNS server.
- You have configured at least one DNS server for the S7-1500 CPU.
- TLS client and TLS server have all the required certificates.

To set up a secure TCP connection to a TLS client, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_QDN_SEC in the global data block.
   The example below shows the global data block "Data_block_1" in which the tag "DNS ConnectionSEC" of the data type TCON_QDN_SEC is defined.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	DNS Connection SEC2	TCON_QDN_SEC		
3	ConnPara	TCON_QDN		parameter of the TCP connection
4	InterfaceId	HW_ANY	0	not relevant
5	ID	CONN_OUC	8	connection reference / identifier
6	ConnectionType	Byte	11	type of connection: 16#0B=11=TCP/IP, 16#13
7	ActiveEstablished	Bool	false	active/passive connection establishment
8	RemoteQDN	String[254]	''	fully or partially qualified domain name of rem
9	RemotePort	UInt	0	remote UDP / TCP port number
10	LocalPort	UInt	2010	local UDP / TCP port number
11	ActivateSecureConn	Bool	true	activate the security functionality of that conn
12	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a
13	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 a
14	TLSServerCertRef	UDInt	5	for Server side: Reference to own X.509 V3 se
15	TLSClientCertRef	UDInt	0	for Client side: add id of own X.509 V3 client c

Figure 4-21 TCON_QDN_SEC_Server

3. Set the connection parameters of the TCP connection in the "Start value" column. Enter, for example, the local ID of the TCP connection for "ID".

4. Set the parameters for secure communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerReqClientCert": Request for an X.509-V3 certificate from the TLS client.
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.

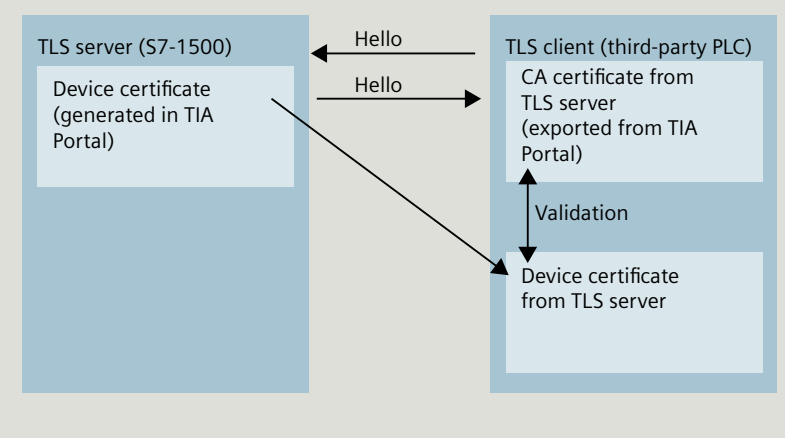


Figure 4-22 Certificate handling from the perspective of the S7-1500 as TLS server

   - "TLSClientCertRef": ID of the X.509-V3 certificate (or a group of X.509-V3 certificates) that is used by the TLS server to validate TLS client authentication. If this parameter is 0, the TLS server uses all (CA) certificates currently loaded in the server certificate store to validate the client authentication.
5. Create one of the instructions TSEND_C, TRCV_C or TCON in the program editor.
6. Interconnect the CONNECT parameter of one of the instructions TSEND_C, TRCV_C or TCON with the tags of the data type TCON_QDN_SEC.
   In the example below, the CONNECT parameter of the TCON instruction is interconnected with the tag "DNS connectionSEC" (data type TCON_QDN_SEC).

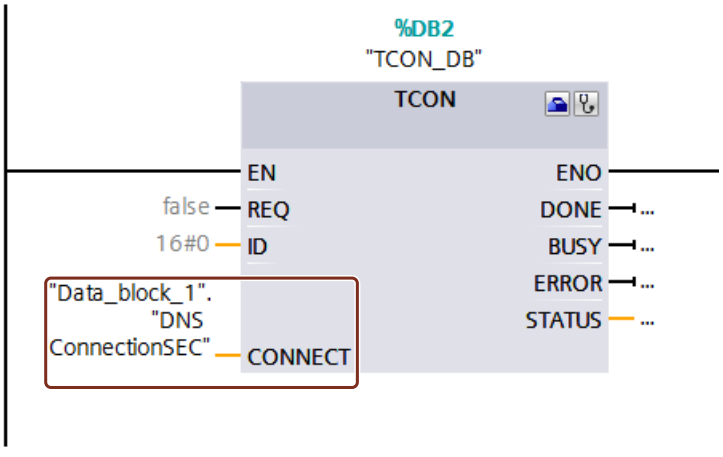


Figure 4-23 TCON instruction

More information

You can find more information about the system data types TCON_QDN_SEC in the STEP 7 online help.
For more information on secure communication, refer to the section Secure Communication (Page 43).

4.6.4.3 Secure OUC between two S7-1500 CPUs

The following section describes how you can set Secure Open User Communication via TCP between two S7-1500 CPUs. In the process one S7-1500 CPU acts as TLS client (active establishing of the connection) and the other S7-1500 CPU as TLS server (passive establishing of the connection).

Setting up a secure TCP connection between two S7-1500 CPUs

For secure TCP communication between two S7-1500 CPUs you need to create a data block with the TCON_IP_V4_SEC system data type yourself in every CPU, assign parameters and call it directly at one of the instructions TSEND_C, TRCV_C or TCON.
Requirements:

- Current date and time are set in the CPU.
- Both S7-1500 CPUs have at least firmware version V2.0
- TLS client and TLS server have all the required certificates.

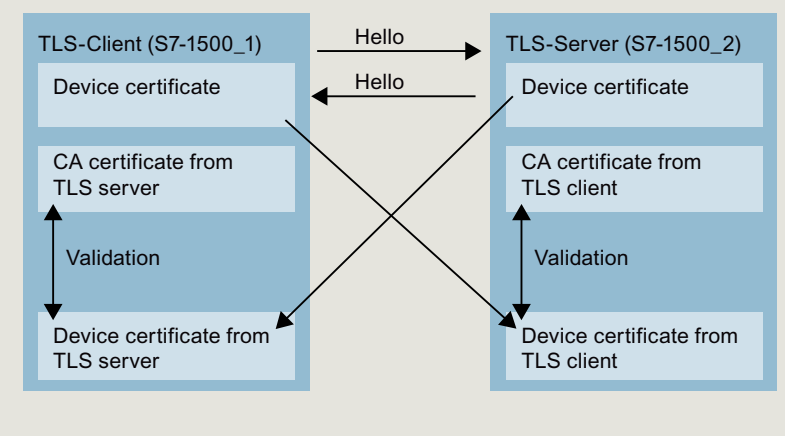


Figure 4-24 Certificate handling for Secure OUC between two S7-1500 CPUs

Settings at the TLS client

To set up a secure TCP connection in the TLS client, follow these steps:

1. Create a global data block in the project tree.

2. Define a tag of the data type TCON_IP_V4_SEC in the global data block.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS-Client" of the data type TCON_IP_V4_SEC is defined.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Client	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	72	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	100	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against th
19	TLSServerCertRef	UDInt	1	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	5	for Client side: add id of own X.509 V3 client certificate; for Ser

Figure 4-25 IP_V4_SEC_Client

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS server for "RemoteAddress".

NOTE
Connection parameter Interface ID
Note that you can enter the value "**0**" for the interface ID in the data type TCON_IP_V4_SEC. In this case, the CPU itself searches for a suitable local CPU interface.

4. Set the parameters for secure communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerCertRef": Enter the value 2 (reference to the CA certificate of the TIA Portal project (SHA256) or the value 1 (reference to the CA certificate of the TIA Portal project (SHA1)). If you use a different CA certificate, enter the corresponding ID from the certificate manager of the global security settings.
   - "TLSClientCertRef": ID of the own X.509-V3 certificate.
5. Create one of the instructions TSEND_C, TRCV_C or TCON in the program editor.
6. Interconnect the CONNECT parameter of one of the instructions TSEND_C, TRCV_C or TCON with the tags of the data type TCON_IP_V4_SEC.

Settings at the TLS server

To set up a secure TCP connection in the TLS server, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_IP_4_SEC in the global data block.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS-Server" of the data type TCON_IP_V4_SEC is defined.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Server	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	120	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	false	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	10	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	true	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against the
19	TLSServerCertRef	UDInt	6	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	1	for Client side: add id of own X.509 V3 client certificate; for Serv

Figure 4-26 IP_V4_SEC_Server

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS client for "RemoteAddress".
4. Set the parameters for secure communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerReqClientCert ": Request for an X.509-V3 certificate from the TLS client. Enter the value "true".
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.
   - "TLSClientCertRef": Enter the value 2 (reference to the CA certificate of the TIA Portal project (SHA256) or the value 1 (reference to the CA certificate of the TIA Portal project (SHA1)). If you use a different CA certificate, enter the corresponding ID from the certificate manager of the global security settings.
5. Create one of the instructions TSEND_C, TRCV_C or TCON in the program editor.

6. Interconnect the CONNECT parameter of one of the instructions TSEND_C, TRCV_C or TCON with the tags of the data type TCON_IP_V4_SEC.
   In the example below, the CONNECT parameter of the TSEND_C instruction is interconnected with the "SEC connection 1 TLS client" tags (data type TCON_IP_4_SEC).

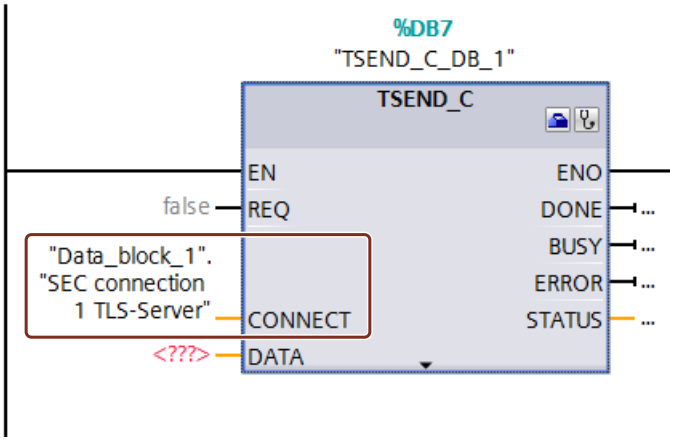


Figure 4-27 TSEND_C

Additional information

You can find more information about the system data types TCON_IP_4_SEC in the STEP 7 online help.
For additional information on secure communication, refer to the section Secure Communication (Page 43).

4.6.4.4 Secure OUC via CP interface

The following sections describes the particular points to be taken into consideration in the case of Secure Open User Communication via a CP interface. At least one station is an S7-1500 station with the following modules:

- S7-1500 CPU as of firmware version V2.0 (with the exception of S7-1500 Software Controller)
- CP 1543-1 as of firmware version V2.0 or CP 1543SP-1 as firmware version V1.0

The CP acts in an S7-1500 station as a TLS client (active connection establishment) or a TLS server (passive connection establishment).
The fundamental procedure and the concept for using secure communication via a CP interface is similar to that of secure communication via the interfaces of the S7-1500 CPUs. Essentially, you have to assign the certificates to the CPU in the role of a TLS server or TLS client and not to the CPU. Other rules and procedures therefore apply. These are described below.

Handling certificates for CPs

The following applies in general: You have to be logged on at the certificate manager in the global security settings. The generation of self-signed certificates also requires logon for the

global security settings. You have to have sufficient rights as a user (administrator or user with the "Standard" role with the right to "Configure security").

The starting point for the generation or assignment of certificates at the CP is the section "Security > Security properties". In this section, you log on for the global security settings.

Procedure:

1. In the network view of STEP 7, mark the CP and select the section "Security > Security properties" in the Inspector window.
2. Click on the "User logon" button.
3. Log on using your user name and password.
4. Enable the "Activate security functions" option.
   The security properties are initialized.
5. Click in the first line of the "Device certificates" table to generate a new certificate or select an existing device certificate.
6. If the communication partner is also an S7-1500 station, you also have to assign a device certificate to the communication partner with STEP 7 as described here or for the S7-1500 CPU.

Example: Setting up a secure TCP connection between two S7-1500 CPUs via CP interfaces

For secure TCP communication between two S7-1500 CPs you need to create a data block with the TCON_IP_V4_SEC system data type yourself in every CPU, assign parameters and call it directly at one of the instructions TSEND_C, TRCV_C or TCON.

Requirements:

- Both S7 1500 CPUs have at least firmware version V2.0. If you use the CP 1543SP-1: Firmware version as of V1.0.
- Both CPs (for example CP 1543-1) must have at least firmware version V2.0
- TLS client and TLS server have all the required certificates.
  - A device certificate (end-entity certificate) for the CP must be generated and be located in the certificate memory of the CP. If a communication partner is an external device (for example an MES or ERP system), a device certificate also has to exist for this device.
  - The root certificate (CA certificate) with which the device certificate of the communication partner is signed must also be located in the certificate memory of the CP or in the certificate memory of the external device. If you use intermediate certificates, you have to ensure that the complete certificate path exists in the validating device. A device uses these certificates to validate the device certificate of the communication partner.
- The communication partner must always be addressed via its IPv4 address, not via its domain name.

The following figure shows the different certificates in the devices for the case that both communication partners communicate via a CP 1543-1. In addition, the figure shows the transfer of the device certificates during establishment of the connection ("Hello").

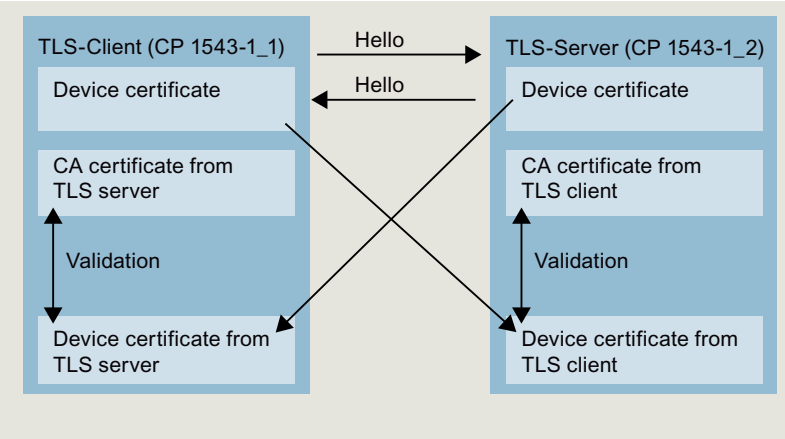


Figure 4-28 Certificate handling in secure OUC between two S7-1500 CPUs via CP interfaces.

Settings at the TLS client

To set up a secure TCP connection in the TLS client, follow these steps:

1. Create a global data block in the project tree.

2. Define a tag of the data type TCON_IP_4_SEC in the global data block. To do so, enter the string "TCON_IP_V4_SEC" in the "Data type" field.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS-Client" of the data type TCON_IP_V4_SEC is defined.
   The Interface ID has the value of the HW identifier of the IE interface of the local CP (TLS client).

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Client	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	258	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	100	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against th
19	TLSServerCertRef	UDInt	1	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	5	for Client side: add id of own X.509 V3 client certificate; for Serv

Figure 4-29 IP_V4_SEC_Client

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS server for "RemoteAddress".
4. Set the parameters for secure communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerCertRef": Enter the value 2 (reference to the CA certificate of the TIA Portal project (SHA256) or the value 1 (reference to the CA certificate of the TIA Portal project (SHA1)). If you use a different CA certificate, enter the corresponding ID from the certificate manager of the global security settings.
   - "TLSClientCertRef": ID of the own X.509-V3 certificate.
5. Create one of the instructions TSEND_C, TRCV_C or TCON in the program editor.
6. Interconnect the CONNECT parameter of one of the instructions TSEND_C, TRCV_C or TCON with the tags of the data type TCON_IP_V4_SEC.

Settings at the TLS server

To set up a secure TCP connection in the TLS server, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_IP_4_SEC in the global data block.
   The example below shows the global data block "Data_block_1" in which the tag "SEC connection 1 TLS-Server" of the data type TCON_IP_V4_SEC is defined.
   The interface ID has the value of the HW identifier of the IE interface of the local CP (TLS server).

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC connection 1 TLS-Server	TCON_IP_V4_SEC		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	260	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	10	connection reference / identifier
6	ConnectionType	Byte	11	type of connetion: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	false	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of Byte		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	1	IPv4 address
13	ADDR[4]	Byte	10	IPv4 address
14	RemotePort	UInt	4711	remote UDP/TCP port number
15	LocalPort	UInt	4711	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	true	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	16#0	Bit 0: Just for client side: validate given IPv4 address against the
19	TLSServerCertRef	UDInt	6	for Server side: Reference to own X.509 V3 server certificate; fo
20	TLSClientCertRef	UDInt	1	for Client side: add id of own X.509 V3 client certificate; for Serv

Figure 4-30 IP_V4_SEC_Server

3. Set the connection parameters of the TCP connection in the "Start value" column. For example, enter the IPv4 address of the TLS client for "RemoteAddress".
4. Set the parameters for secure communication in the "Start value" column.
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerReqClientCert ": Request for an X.509-V3 certificate from the TLS client. Enter the value "true".
   - "TLSServerCertRef": ID of the own X.509-V3 certificate.
   - "TLSClientCertRef": Enter the value 2 (reference to the CA certificate of the TIA Portal project (SHA256) or the value 1 (reference to the CA certificate of the TIA Portal project (SHA1)). If you use a different CA certificate, enter the corresponding ID from the certificate manager of the global security settings.
5. Create one of the instructions TSEND_C, TRCV_C or TCON in the program editor.
6. Interconnect the CONNECT parameter of the instruction TSEND_C, TRCV_C or TCON with the tags of the data type TCON_IP_V4_SEC.

Upload device as new station

When you upload a configuration with certificates and configured secure Open User Communication as a new station into your STEP 7 project, the certificates of the CP are not uploaded, in contrast to the certificates of the CPU. After the device has been loaded as a new station, no more certificates are contained in the corresponding tables of the CPs for the device certificates.

You have to perform configuration of certificates again after the upload. Otherwise, renewed loading of the configuration results in the certificates that originally exist in the CP being deleted so that secure communication does not function.

Secure OUC connections via CPU and CP interfaces - similarities

- Connection resources:
  No differences between OUC and secure OUC. A programmed secure OUC connection uses a connection resource just like an OUC connection, irrespective of which IE/PROFINET interface communicates with the station.
- Connection diagnostics:
  No differences between OUC and secure OUC connection diagnostics.
- Loading of projects with secure OUC connections into the CPU:
  Only possible in STOP of the CPU, if certificates are loaded as well.
  Recommendation: Load to device > Hardware and software. Reason: Ensuring the consistency between the program with secure OUC, hardware configuration and certificates.
  Certificates are loaded with the hardware configuration - therefore loading requires a stop of the CPU. The reloading of blocks that utilize further secure OUC connections is only possible in RUN if the certificates required for this purpose are already located on the module.

4.6.4.5 Secure OUC with Modbus TCP

For secure Modbus TCP connection you need to create a data block with one of the system data types TCON_IP_V4_SEC or TCON_QDN_SEC yourself, assign parameters and call it directly at the MB_Server or MB_CLIENT instruction.

Requirements:

- S7-1500 CPU CPU firmware version V2.5 or higher
- The Modbus client (TLS client) can reach the Modbus server (TLS server) over IP communication in the network.
- TLS client and TLS server have all the required certificates.

Example of setting up a secure Modbus TCP connection to a Modbus TCP server

The following section describes how you can set up a Secure Open User Communication over Modbus TCP from a Modbus TCP client to a Modbus TCP server.

To set up a secure connection from a Modus TCP client (TLS client) to a Modbus TCP server (TLS server) and set up the IPv4 address of the mail server, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_IP_V4 SEC in the global data block.

Data_block_1

	Name	Data type	Start value	Comment
1	Static			
2	SEC_ModbusTCP_1	TCON_IP_V4...		
3	ConnPara	TCON_IP_v4		parameter of the TCP connection
4	InterfaceId	HW_ANY	64	HW-identifier of IE-interface submodule
5	ID	CONN_OUC	15	connection reference / identifier
6	ConnectionType	Byte	11	type of connection: 11=TCP/IP, 19=UDP (17=TCP/IP)
7	ActiveEstablished	Bool	true	active/passive connection establishment
8	RemoteAddress	IP_V4		remote IP address (IPv4)
9	ADDR	Array[1..4] of B...		IPv4 address
10	ADDR[1]	Byte	192	IPv4 address
11	ADDR[2]	Byte	168	IPv4 address
12	ADDR[3]	Byte	10	IPv4 address
13	ADDR[4]	Byte	100	IPv4 address
14	RemotePort	UInt	502	remote UDP/TCP port number
15	LocalPort	UInt	502	local UDP/TCP port number
16	ActivateSecureConn	Bool	true	activate the security functionality of that connection in general
17	TLSServerReqClientCert	Bool	false	Just for server side: The TLS server requests a client certificate
18	ExtTLSCapabilities	Word	0	Bit 0: Just for client side: validate given IPv4 address against the subjectAlt
19	TLSServerCertRef	UDInt	2	for Server side: Reference to own X.509 V3 server certificate; for Client side:
20	TLSClientCertRef	UDInt	7	for Client side: add id of own X.509 V3 client certificate; for Server side: add

Figure 4-31 TCON_IP_V4_SEC

3. Set the connection parameters of the TCP connection in the "Start value" column. Enter the IPv4 address of the mail server, for example, for the "MailServerAddress".
4. Set the parameters for secure communication in the "Start value" column. Enter the certificate ID of the CA certificate of the communication partner, for example, for "TLSServerCertRef".
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. In this case you can set up an unsecured Modbus TCP connection.
   - "TLSServerCertRef": Reference to the X.509 V3 (CA) certificate of the Modbus TCP server, which is used by the TLS client to validate the authentication of the Modbus TCP server.
5. Create an MB_CLIENT instruction in the program editor.
6. Interconnect the CONNECT parameter of the MB_Client instruction with the tags of the data type TCON_IP_4_SEC.

4.6.4.6 Secure OUC via e-mail

Setting up a secure connection to a mail server over the CPU interface

For secure communication to a mail server you need to create a data block with one of the system data types TMAIL_V4_SEC, TMAIL_QDN_SEC yourself, assign parameters and call it directly at the TMAIL_C instruction.

Requirements:

- TMAIL_C instruction version V5.0 or higher
- STEP 7 V15 and higher
- S7-1500 CPU V2.5 and higher
- You have assigned all the CA certificates of the mail server (TLS server) to the CPU (TLS client) and have downloaded the configuration to the CPU.
- Current date and time are set in the CPU.

Process for establishing a secure connection to the mail server

You can choose between two processes for establishing the secure connection to the mail server:

- SMTPS: The client attempts to immediately establish a TLS connection to the mail server ("handshake" process). If the mail server does not support TLS, then no connection is established.
- STARTTLS: Client establishes a TCP connection to the mail server. The client sends a request to "upgrade" the existing connection to a secure TLC connection over the TCP connection. If the mail server supports TLS, the client sends the command to establish a secure connection. The mail server uses the SMTP command "STARTTLS" to do this. The client then establishes a secure connection to the mail server. Advantage: If the mail server does not support TLS, client and mail server can communicate unsecured with each other.

You use the "Remote Port" setting in the data types at the block parameter "MAIL_ADDR_PARAM" to define which process is used for the communication.

Table 4-5 Port numbers for the SMTPS and STARTTLS processes

Process	Port
SMTPS:	465[1]
STARTTLS	Any (≠465)[2]

[1] The instruction TMAIL_C uses SMTPS only for Port 465. For all other ports STARTTLS is used.

[2] According to RFC, mail servers use Ports 25 and 587 for secure connections with STARTTLS. The use of other port numbers for SMTP is not RFC-compliant, successful communication with such a mail server is not guaranteed.

Example: Setting up a secure connection to a mail server over IPv4

The following section describes how to set up a secure connection to an IPv4 mail server with the TMAIL_C communication instruction.

To set up a secure connection via the IP4 address of the mail server, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TMAIL_V4_SEC in the global data block.
   The example below shows the global data block "MailConnDB" in which the tag "MailConnectionSEC" of the data type TMAIL_V4_SEC is defined.

MailConnDB

Name	Datentyp	Startwert	Kommentar
Static			
MailConnectionSEC	TMail_V4_SEC		
InterfaceId	HW_ANY	"Local~CP_1543-1_1~Ethernet-Schnittstelle_1"	Use HW-identifier of the IE-interface to specify th
ID	CONN_OUC	100	connection reference / identifier
ConnectionType	Byte	16#20	type of connection 16#20=32=TMail_V4 or TMai
ActiveEstablished	Bool	true	active / passive connection establishment
WatchDogTime	Time	T#5000ms	watchdog time to monitor SMTP server associat
MailServerAddress	IP_V4		IPv4 address of mail server
ADDR	Array[1..4] of Byte		IPv4 address
ADDR[1]	Byte	144	IPv4 address
ADDR[2]	Byte	145	IPv4 address
ADDR[3]	Byte	2	IPv4 address
ADDR[4]	Byte	20	IPv4 address
UserName	String[254]	'myName'	user name which is necessary to login into the
PassWord	String[254]	'myPW'	user password which is necessary to login into
From	EMAIL_ADDR		source mail address
LocalPartPlusAt...	String[64]	'Mustermann@'	local part of e-mail address plus "@" sign
FullQualifiedD...	String[254]	'siemens.com'	full qualified domain name part of e-mail addre
RemotePort	UInt	587	remote TCP port number
ActivateSecureConn	Bool	TRUE	activate the security functionality of that conne
ExtTLSCapabilities	Byte	16#0	for further capability extensions of the TLS hand
TLSServerCertRef	UDInt	7	Reference to the X.509 V3 (CA-) certificate of the

Figure 4-32 Data type TMAIL_V4_SEC

3. Set the connection parameters of the TCP connection in the "Start value" column. Enter the IPv4 address of the mail server, for example, for the "MailServerAddress".

NOTE

Connection parameter Interface ID

Note that as of instruction version V5.0 of TMAIL_C instruction in the TMAIL_V4_SEC data type, you need to enter the value "**0**" for the Interface ID. In this case, the CPU itself searches for a suitable local CPU interface.

4. Set the parameters for secure communication in the "Start value" column. Enter the certificate ID of the CA certificate of the communication partner, for example, for "TLSServerCertRef".
   - "ActivateSecureConn": Activation of secure communication for this connection. If this parameter has the value FALSE, the subsequent security parameters are irrelevant. You can set up a non-secure TCP or UDP connection in this case.
   - "TLSServerCertRef": Reference to the X.509 V3 (CA) certificate of the mail server, which is used by the TLS client to validate the authentication of the mail server.

5. Create a TMAIL_C instruction in the program editor.
6. Interconnect the MAIL_ADDR_PARAM parameter of the TMAIL_C instruction with the tag of the data type TMAIL_V4_SEC.
   In the following example the MAIL_ADDR_PARAM parameter of the TMAIL_C instruction is interconnected with the tag "MailConnectionSEC" (data type TMAIL_V4_SEC).

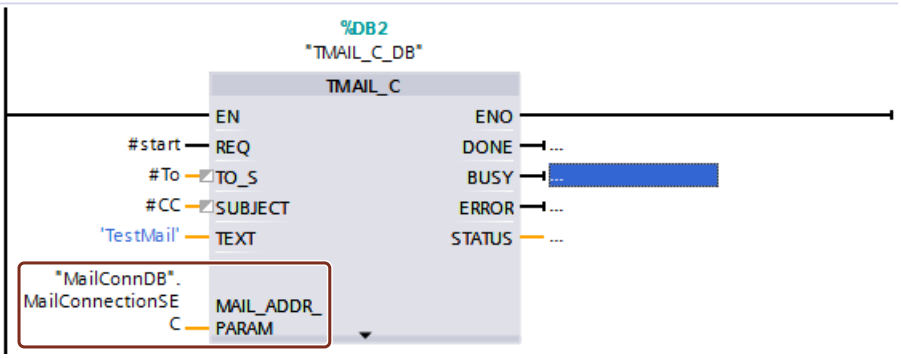


Figure 4-33 TMAIL_C instruction

Setting up a secure connection to a mail server over the interface of a communication module

For secure communication to a mail server over a communication module, you need to create a data block with one of the system data types TMAIL_V4_SEC, TMAIL_QDN_SEC or TMAIL_V6_SEC yourself, assign parameters and call it directly at the TMAIL_C instruction.
Requirements:

- TMAIL_C instruction with version **V4.0**
- S7-1500 CPU as of firmware version V2.0 with communication module CP 1543-1 as of firmware version V2.0
- ET 200SP CPU as of firmware version V2.0 with communication module CP 1542SP-1 (IRC) as of firmware version V1.0
- You have assigned all the CA certificates of the mail server (TLS server) to the CP (TLS client) and have downloaded the configuration to the CPU.
- Current date and time are set in the CPU.

The STEP 7 online help describes how to set up a secure connection to a mail server over the interface of a communication module.

Application example

This application example (https://support.industry.siemens.com/cs/ww/en/view/46817803) show how you can use the CP of an S7-1500 or S7-1200 station to set up a secure connection to an email server and send an email with the default application "TMAIL_C" from the S7 CPU.

Additional information

You can find more information about the system data types TMail_V4_SEC and TMAIL_QDN_SEC in the STEP 7 online help.

For additional information on secure communication, refer to the section Secure Communication (Page 43).

4.6.5 Secure PG/HMI communication

4.6.5.1 PG/HMI communication based on standardized security mechanisms

With the central components of the TIA Portal, STEP 7 and WinCC, an innovative and standardized Secure PG/PC and HMI Communication - PG/HMI communication for short - is implemented starting with version V17 together with the latest controllers and latest HMI devices.

The following CPU families are referred to in detail:

- S7-1500 controller family as of firmware version V2.9
- S7-1200 controller family as of firmware version V4.5
- Software controllers as of firmware version V21.9
- SIMATIC Drive controllers as of firmware version V2.9
- PLCSim and PLCSim Advanced Version V4.0

HMI components have also been updated to support Secure PG/HMI Communication:

- Panels or PCs configured with WinCC Basic, Comfort and Advanced
- PCs with WinCC RT Professional
- WinCC Unified PCs and Comfort Panels

Also updated are SINAMICS RT SW as of version V6.1 and STARTDRIVE as of version V17.

Properties of PG/HMI communication

One characteristic of PG communication and HMI communication above all is their simplicity: Establishing an online connection from a programming device with installed TIA Portal to a CPU, for example, to load a program, requires little effort. This online connection also meets criteria such as confidentiality and integrity - based on a proven SIMATIC communication standard.

In the course of integrating machines and systems into an open IT environment, however, it must be ensured that the communication between the programming device / HMI device and the CPU is not only secure in the sense of maintaining integrity and confidentiality for sensitive data but also that this security meets generally accepted standards and is thus ready for the challenges of the future.

With TIA Portal version V14, the "Open User Communication" procedure for communication based on user programs has already been extended by the "**Secure** Open User Communication" variant. Other certificate-based communication mechanisms have become established (HTTPS, Secure SMTP over TLS or OPC UA). As of TIA Portal Version V17, PG/HMI communication has also been upgraded: Here, too, the TLS (Transport Layer Security) protocol is used to secure PG/HMI communication using standardized security mechanisms.

What has changed

Additional optional password for more security

The most noticeable change in the frame of the configuring of the above devices is the ability to assign a password to protect sensitive configuration data of the respective CPU. This refers to data such as private keys that are required for the proper functioning of certificate-based protocols (Secure Communication) - as of TIA Portal V17 also for the PG/HMI communication. You can use a policy setting to check the assigned passwords as they are entered into the TIA Portal. This will ensure that your company complies with prescribed password policies.
If your machine or system does not require this protection based on the Siemens Industrial Defense-in-Depth concept, you can dispense with password assignment, for example, because another equivalent protection is present. It is possible to do without the password if you have implemented measures to prevent unauthorized access to the TIA Portal project and the configuration of the CPU.

⚠ WARNING

Without password, weak protection of private keys

Note that without a password to protect trusted configuration data, the private keys for certificates required for secure communication are only weakly protected.

Certificate-based communication between PG/HMI and CPU

Because PG/HMI communication is certificate-based, you will be asked to accept the server certificate during commissioning.
Additional parameter assignment options allow you to determine the behavior of the CPU during operation: For example, you can specify that the CPU also allows connection to devices that do not support Secure PG/HMI communication.

Maintenance / replacement parts scenario

For the problem-free exchange of the CPU in replacement parts scenario, you must observe specific rules (see Rules for the replacement parts scenario (Page 75)).

More information

An overview of how to protect confidential configuration data can be found in section Protection of confidential configuration data (Page 65).

4.6.5.2 Additional settings for the secure PG/HMI communication

In addition to the assignment of a password to protect confidential PLC configuration data, you have further setting options for the behavior of the CPU during operation.

PG/PC and HMI communication mode

You can set how the CPU can communicate with programming devices and HMI devices:

- Only via secure PG/HMI communication
- Both via Secure PG/HMI communication and via the previously used PG/HMI communication, "Legacy PG/HMI communication" for short.

Procedure

1. In the CPU properties, navigate to the area "Protection & Security > Connection mechanisms".
2. Select the option you want to use.

Select certificate or generate a new certificate

If you select the connection mechanism for PG/HMI communication, you can select a PLC communication certificate for the protection of the connection in the same context or have it generated by the TIA Portal. If you have assigned a password or if you have deactivated the option to protect confidential PLC configuration data (i.e. no password has been set), then a certificate with suitable settings and a valid default name is already preset in the "Protection & Security > Connection mechanisms" area.

Procedure

If you want to have a new certificate generated by the TIA Portal or if you want to select another existing certificate:

1. In the "PLC communication certificate" field, click the three points to expand the field.
2. Select the certificate you want or click the "Add" button.
3. When adding a certificate, a dialog appears with setting options for the certificate. The purpose is set to "TLS server", you can change other parameters (such as name, hash algorithm).

The general rules for certificate management apply. For example, if you want to generate a CA certificate, the option "Global settings for the certificate manager" must be selected. You also have the option of generating a self-signed PLC certificate.

More information

Basic information on the subject of certificate management can be found in section Certificate management with TIA Portal (Page 54).

4.6.5.3 Tip for certificate-based communication between PG and CPU

The certificate-based PG/PC communication (Secure PG/PC communication) means that the communication partner of the CPU – the programming device with installed TIA Portal – must trust the device certificate of the CPU so that a connection can be loaded.

To put it simply, from the TIA Portal perspective you have the following options to trust the certificate of a CPU:

- The PG with TIA Portal is in possession of the device certificate of the CPU because it was, for example, created or imported in the project. In this case, the certificate check runs automatically and without prompting.
- The PG with TIA Portal is not in possession of the device certificate of the CPU, because the CPU was determined via "Accessible stations", for example, and is not available in the project. In this case, the TIA Portal asks the TIA Portal user whether the certificate can be trusted. This may be possible only with great effort because the CPU is not in sight, for example, and the authenticity can therefore not be checked immediately.
- The PG with TIA Portal is in possession of the CA certificate (certification authority) and all CPUs that can be reached in the network from the TIA Portal have device certificates issued by this CA certificate.
  Advantage of this solution: TIA Portal can check device certificates automatically, even if the device certificates of the communication partners are not available in TIA Portal.

The solution with a CA certificate (certification authority) is explained in more detail below.

Requirement

You can use the certification authority of the TIA Portal to create device certificates for a CPU and use the existing CA certificates to sign the device certificates. However, you can also import another certification authority into TIA Portal and use it.
Enabling the global security policies for the certificate manager is a requirement. Only with this setting you can generate CA-signed certificates.
See also here: Certificate management with TIA Portal (Page 54)

Exporting CA certificate for programming devices

To export the corresponding CA certificate after creating and assigning a certificate, follow these steps:

1. Open the certificate manager in the global security settings in the project tree.
2. Select the table "CA certificates" for the certificate to be exported.
3. Right-click to open the shortcut menu of the selected certificate.
4. Click "Export".
5. Select the export format of the certificate and the storage location.

Store CA certificate in the TIA Portal

To make the exported certificate known to a PG with TIA Portal and thus enable automatic certificate checking, follow these steps:

1. Copy the CA certificate exported in the previous step to the following directory:
   C:\ProgramData\Siemens\Automation\Certstore\Trusted
2. Start TIA Portal.
   In the "Info" tab of the Inspector window, a message appears for each CA certificate which provides information on whether the CA certificate could be successfully transferred to the CA store of TIA Portal.
   However, no detailed causes are output in case of failure.

Adding device certificates to the TIA Portal certificate revocation list (CRL)

You have the option to add individual device certificates to a certificate revocation list (CRL), for example, because the associated key is no longer considered secure.
When the TIA Portal establishes a connection to a CPU whose device certificate is in the certificate revocation list, a dialog appears in the TIA Portal asking whether you still want to trust the certificate. If you decline, the connection will not be established.
To add a device certificate to the certificate revocation list, follow these steps:

1. Copy the device certificate to the following directory:
   C:\ProgramData\Siemens\Automation\Certstore\CRL
2. Start TIA Portal.
   In the "Info" tab of the Inspector window, a message appears for each certificate which provides information about whether the certificate could be successfully transferred to the CRL store of TIA Portal.
   However, no detailed causes are output in case of failure.

4.6.5.4 CPU behavior from loading to operational readiness

To ensure that communication between the CPU and a programming device or HMI device is secure, it must first have a certificate. However, the certificate for productive operation is only issued when the project is loaded into the CPU.
To ensure that the initial loading is also secured, the CPU first creates a self-signed certificate. The following description explains the different phases of establishing a connection.

Requirement for the initial establishment of connection to load the CPU

- No password for confidential PLC configuration data available in the CPU.
  If the CPU has already been loaded and therefore already has a password for confidential PLC configuration data, this password must then match the project that is to be loaded.
- Project with CPU configuration (including password for confidential PLC configuration data) and user program is available.
- The CPU is in STOP mode.
- Programming device and CPU are directly connected to each other and are located in a protected environment; i.e. you can identify the CPU to be loaded and control the connection between CPU and programming device.

Initial establishment of connection to the CPU - provisioning phase

The first connection establishment for loading the CPU is secured by the TLS procedure in terms of Secure PG/HMI Communication.
However, the CPU uses its manufacturer device certificate (if available) or a self-signed certificate to establish this connection. The CPU can only be used to a limited extent in this phase. In this phase, the CPU waits for the provision of the password-based key information - or more simply stated, it is expecting the password for confidential PLC configuration data. This phase is referred to below as the provisioning phase. A message in the diagnostic buffer indicates that the CPU is in the provisioning phase.

When a project is loaded into the CPU, the CPU receives the project data:

- Hardware configuration including configured certificates for secure communication (OPC UA, HTTPS, Secure OUC, Secure PG/HMI Communication)
- User program

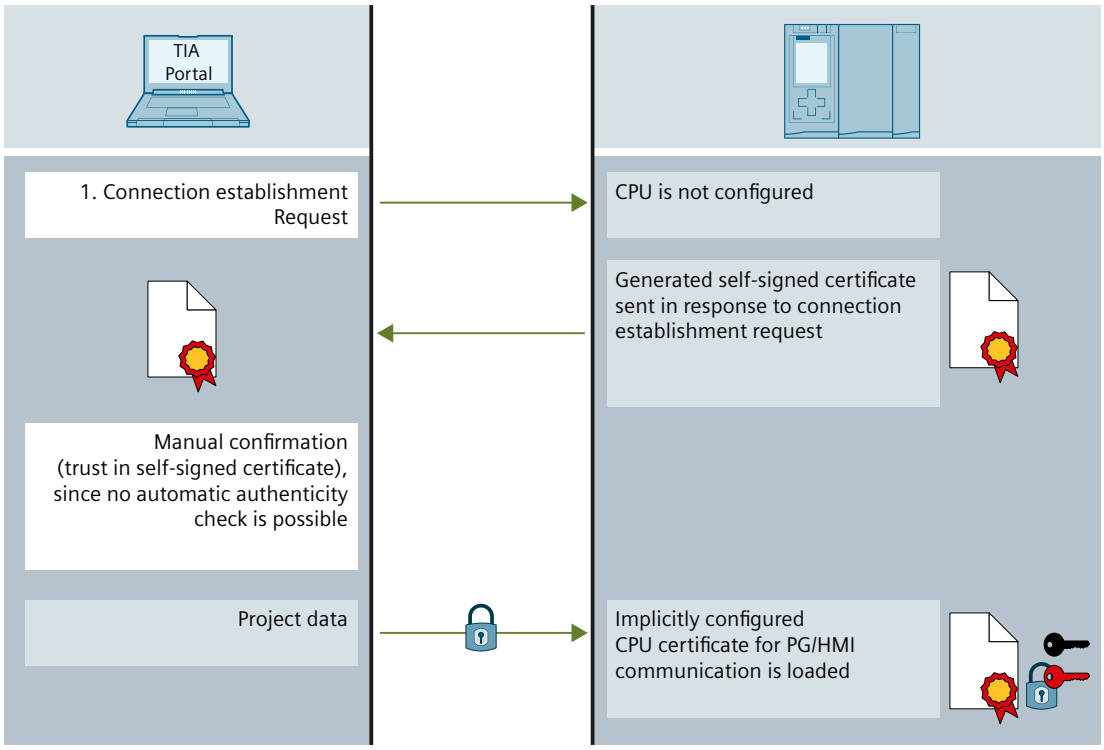


Figure 4-34 Connection establishment, provisioning phase

WARNING

Potential security risks during commissioning

During commissioning, the CPU provides a manufacturer device certificate (if available) or a self-signed certificate that you must trust in order to establish a connection. Only trust this certificate if the programming device and the CPU are in a protected network and are directly connected to each other. In an unprotected environment, these certificates can be manipulated and allow attackers to access the communication between the programming device/HMI and CPU (e.g. through man-in-the-middle attacks).

Ending the provisioning phase

TIA Portal does not store the password for confidential PLC configuration data itself or the key information generated from the password in the project.
Therefore, the password is requested in a dialog when loading the project for the first time or when loading a new project and transferred to the CPU as key information. Only after this step the CPU is able to use the protected PLC configuration data - this completes the provisioning phase and the CPU can start operating.

If you do not protect the confidential PLC configuration data with a password, there is no need to enter the password when loading the CPU for the first time. This has no influence on the flow of the PG/HMI communication but you have to consider that the confidential PLC configuration data (e.g. private keys) offer almost no protection against unauthorized access in this case.

Startup of PG/HMI communication

When the CPU is loaded and has received the CPU certificate for Secure PG/HMI Communication, the programming device connects again - this time based on the loaded CA certificate.

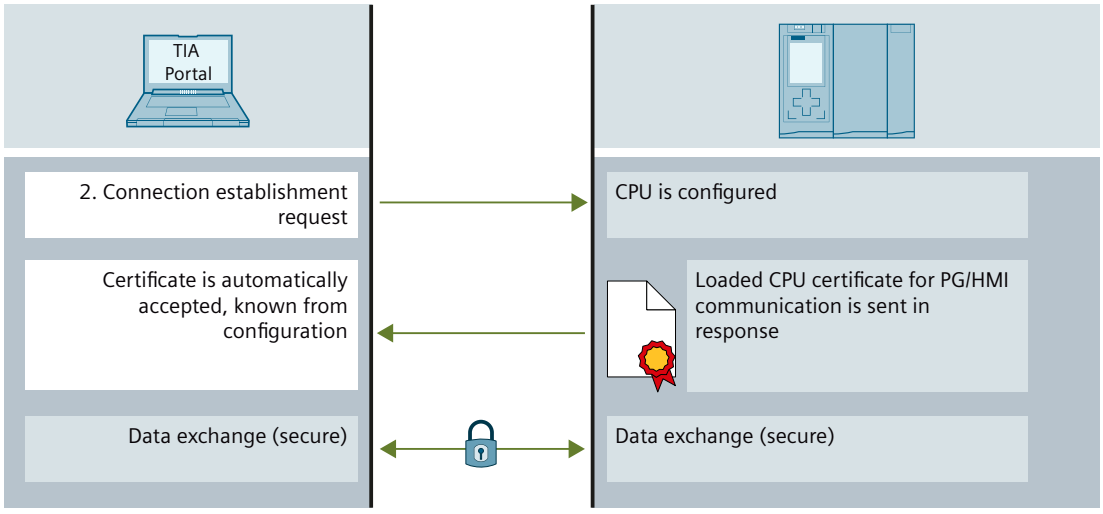


Figure 4-35 Startup of PG/HMI communication

4.6.5.5 Using secure HMI communication

As of TIA Portal version V17, the CPU and the HMI device communicate via secure HMI communication if both devices meet the requirements for this type of communication.
The basis of secure HMI communication is that the HMI device can verify the authenticity of the CPU, using the PLC communication certificate that the CPU sends when establishing communication, and considers the CPU to be "trustworthy". Secure HMI communication is only possible when this turns out to be the case.
This section describes the measures you must take for the various HMI devices to manually label the PLC communication certificate as "trustworthy".

Requirement

- CPU and HMI device support secure HMI communication.
- A current project is available on the CPU (TIA Portal V17 and higher).

Configuring secure HMI communication

1. Configure the HMI device with an alarm view.

   NOTE

   Without an alarm view you cannot identify the errors during connection establishment.

2. Configure the CPU with the required security settings. Select a PLC communication certificate to protect the HMI connection or have the TIA Portal generate a PLC communication certificate.
3. Configure the HMI connection between the CPU and the HMI device.
4. Download the project to the CPU and the HMI device. During the project transfer, the PLC communication certificate and, if necessary, a CA (Certificate Authority) certificate is transferred to the CPU and the HMI device.

Trusting the PLC communication certificate

During connection setup, the CPU transfers the PLC communication certificate to the HMI device.

- When the PLC communication certificate is already available on the HMI device with the "trustworthy" status, a secure HMI communication is automatically set up between the CPU and the HMI device.
- When the PLC communication certificate is not available in the "trustworthy" status on the HMI device, you will see a message in the alarm view of the HMI device informing you that the CPU is not trusted along with an error code.

  In this case, you must label the PLC communication certificate on the HMI device as "trustworthy".

Depending on the type of your HMI device, follow these steps.

Basic Panels 2nd Generation

1. In the Start Center, select "Settings > Internet Settings > Certificate store".
2. In the "Available certificates in Device" list, select the PLC communication certificate of the CPU.
3. Press "Trust".
4. Restart the HMI runtime software.

Unified Comfort Panels

1. Open the Control Panel.
2. Select "Security > Certificates".
3. In the "Certificate store" selection list, select the entry "Other Certificates".
4. In the "Other certificates" list, select the PLC communication certificate of the CPU.
5. Press "Trust".
6. Restart the HMI runtime software.

Comfort Panels, Mobile Panels 2nd Generation

1. Open the file manager via the Windows CE desktop icon "My Device".
2. Navigate to the directory "\flash\simatic\SystemRoot\OMS\Untrusted". There you will find the PLC communication certificate of the CPU.
3. Copy the PLC communication certificate of the CPU to the directory "\flash\simatic\SystemRoot\OMS\Trusted".
4. Restart the HMI runtime software.

When the PLC communication certificate is already available on the HMI device with the "trustworthy" status, secure HMI communication can be set up. You can find more information in the operating instructions of the HMI device.

4.6.5.6 Using Legacy PG/PC communication for TIA Portal

As of TIA Portal version V17, the TIA Portal and the S7-1200/S7-1500 CPU as of firmware version V4.5/V2.9 automatically communicate "securely" - the connection partners set their connection mechanisms automatically to the highest possible security method.
Only special circumstances (see Information about compatibility (Page 101)) cause a fallback to the old PG/PC communication, called "Legacy PG/PC communication".
There may be some cases in which the higher security is not desirable because it can impact the transmission rate of CPUs with weak communication performance.

Requirement

- There may be no online connections to the CPUs.
- For CPUs that are to be reached online, the option "Only permit secure PG/PC and HMI communication" must be disabled (CPU parameters in the area "Connection mechanisms").
- The communication partners are in a protected environment, for example, during the commissioning phase.

Setting the Legacy PG/PC communication

1. In the "Online" menu, select the command "Use only Legacy PG/PC communication".
2. Select the check box in front of the menu command.

Result: All online connections are set up as for TIA Portal versions < V17.
The setting remains active for the duration of the session. When you open a project, the option "Use only legacy PG/PC communication" is not set.

Behavior with enabled option "Use only Legacy PG/PC communication"

- A password to protect confidential PLC configuration data cannot be specified, modified or deleted online for CPUs. These functions require disabling the "Use only Legacy PG/PC communication" option.
- A CPU that is set to only permit secure PG/PC and HMI communication can no longer be reached online.

4.6.5.7 Information about compatibility

The following description provides information about the interaction between different TIA Portal versions with different CPU firmware versions and the effects on the type of PG/HMI connection.

Projects created with TIA Portal < V17

For example, if you have created a project with TIA Portal version V16 for an S7-1500 CPU (e.g. version V2.8), then the corresponding configuration with TIA Portal V17 can also be loaded into an S7-1500 CPU V2.9, e.g. in a spare part scenario - with the same behavior as a configuration on an S7-1500 CPU V2.8.

Even projects created with TIA Portal < V17 and transferred to a memory card work without problems in an S7-1500 CPU V2.9.

However, the concept of protecting confidential PLC configuration data applies as soon as you open the project with TIA Portal ≥ V17, update the firmware version of the CPU via a device replacement and thus save it as a CPU with a firmware version ≥ V2.9 (see Useful information for the protection of confidential PLC configuration data (Page 67)). The project can no longer be edited with previous versions of TIA Portal V17.

PG/HMI and CPU connected differently

As explained in the previous sections, the secure PG/HMI connection as of V17 between the PG/HMI device and CPU (current version) is characterized by the employed standardized communication procedure, TLS (Transport Layer Security).

You have the option of connecting a V2.9 CPU to a current programming device with TIA Portal V17 or higher and additionally, for example, to an HMI device with a runtime from the previous version: The devices automatically adjust their connection mechanisms accordingly. In order to be able to better differentiate between the two connection mechanisms, we call the previous procedure, which is based on a variant of S7 communication, "Legacy procedure".

In summary ("PG" here stands for a programming device with TIA Portal):

- PG/HMI and CPU come with the V17 (or subsequent version): TLS procedure is used.
- PG/HMI comes from a predecessor version (< V17): Legacy procedure is used - provided that you have deactivated the option "Only allow secure PG/PC and HMI communication" in the CPU properties.
- CPU comes with V17 (or higher), several PGs/HMIs are connected and come from both V17 (or higher) and previous versions: TLS + legacy procedures are used - provided that you have deactivated the option "Only allow secure PG/PC and HMI communication" in the CPU properties.

When the CPU state changes

The diagnostics buffer provides you with information if its status changes due to events in connection with the Secure PG/HMI Communication.

Examples:

- After successfully loading a configuration with a configured password, the diagnostics buffer reports that the CPU is changing from the provisioning phase to Secure Mode (TLS procedure).
- You have connected a PG with TIA Portal V17 to a CPU V2.9. If "Use only Legacy PG/PC communication" is deactivated in the Online menu, the Secure PG/HMI Communication (TLS procedure) is established automatically.

More information

Information on device or firmware-specific features such as the TLS version used can be found in section Device-dependent security features (Page 46).

4.7 SNMP

4.7.1 Activating and deactivating SNMP

The network management protocol SNMP (Simple Network Management Protocol) is used for performing monitoring and diagnostics of the network topology. SNMP uses the transport protocol UDP and knows two roles: the SNMP manager (client) and SNMP agent (server).

- The SNMP manager monitors the network nodes:
- The SNMP agents collect the various network-specific information in the individual network nodes and store it in a structured form in the MIB (Management Information Base). Various services and tools (as SNMP manager) can run detailed network diagnostics with the help of these data.

SNMP is also used in a PROFINET IO system for managing the network infrastructure and the IO controller/IO devices.

NOTE

If SNMP is deactivated for a device, various options for diagnostics of the network topology (for example, using the PRONETAtool) are no longer available to you.

Example: For the topology comparison Online-Offline, the TIA Portal determines which ports are actually connected and uses SNMP for this function.

Default settings depending on the firmware version

S7-1500 CPUs have an integrated SNMP agent. Depending on the firmware version, different defaults apply to SNMP (SNMP activated or deactivated).

With S7-1500 CPUs with a FW version < V3.0, the SNMP agent is **activated** by default and can only be deactivated via a data record in the user program.

Under certain conditions, it may be useful to deactivate SNMP. Examples:

- The security guidelines in your network do not allow the use of SNMP.
- You use your own SNMP solution, e.g. with your own communications instructions.

With S7-1500 CPUs as of FW version V3.0, the SNMP agent is **deactivated** by default. The default setting "disabled" is also in effect if no configuration has been loaded or if no memory card is plugged in. STEP 7 V18 offers the following options for changing the SNMP settings for S7-1500 CPUs as of FW version V3.0:

- Configure SNMP in the CPU properties of TIA Portal.
- Activate/deactivate SNMP in the use program by transferring a data record to a PROFINET interface.

NOTE

Replacement part scenario

For compatibility reasons, an S7-1500 CPU as of firmware version V3.0 with a loaded predecessor project (CPU firmware < V3.0) behaves like the CPU in the predecessor project:

SNMP is activated and "public" and "private" community strings are in effect.

Configuring SNMP

As of CPU firmware version V3.0 and TIA Portal version V18, you have the possibility to change the following settings for SNMP in the CPU properties:

- Activate SNMP (default: deactivated)
- Read-only community string (default: "public")
- Read-write community string (default: "private")

You can find the settings in the "Advanced configuration > SNMP" area.

Meaning and properties of community strings

An SNMP community string, also referred to as a "community name", is like an ID or a password, which allows access to information/statistics of a device such as a router.
In this respect, to increase the security, you should change the default community strings in the CPU properties. An SNMP manager authenticates itself with the SNMP agent by transferring the community strings when there is a request.

The community strings are transferred as plain text.

- The default community string for SNMP read-only operations (GET) is "public".
- The default community string for SNMP read-write operations (SET) is "private".

Number of characters for the community string: 1-240.

You can use the following characters for the community string:

- a-z
- A-Z
- 0-9
- -
- .

Activating/deactivating SNMP in the user program

In addition to the configuration in the CPU properties, you can also activate or deactivate SNMP in the user program. To do this, transfer a data record 0xB071 to a PROFINET interface of the CPU. In this record it is encoded whether SNMP is to be activated/deactivated. No matter to which PROFINET interface you transfer the data record, the data record applies to all interfaces of the CPU.

One way to transfer the 0xB071 data record: Define the data set structure in a data block and transfer it in the program cycle OB (e.g. OB1) with the instruction "WRREC" (write data record) to a PROFINET interface of the CPU.

To do this, follow these steps:

1. In STEP 7, create a data block that contains the structure of data record 0xB071.
   The following table shows the structure of the data record 0xB071.

Byte	Element	Code	Explanation
0-1	BlockID	0xF003	Header
2-3	BlockLength	8	The data record length is counted starting at byte 4 "Version".
4	Version	0x01	
5	Subversion	0x00	
6-7	Reserved	-	-
8-11	SNMP controller	0.1	0: Deactivate SNMP. 1: Activate SNMP.

2. Transfer the data record 0xB071, for example, in the program cycle OB (OB1) to the CPU with the "WRREC" instruction (write data record). Use an integrated PROFINET interface of the CPU as the hardware ID.

Interplay of SNMP configuration and user program

- The SNMP setting "activated/deactivated" via the user program is not stored permanently in the CPU. The configured setting is effective again, for example, after each POWER OFF/POWER ON transition, loading of a new hardware configuration, or resetting to factory settings.
- When loading the configuration from the CPU ("Upload device as new station"), the configured SNMP setting (activated/deactivated) is taken into consideration. An SNMP setting previously set by data record in the user program is ignored.
- The community strings can only be changed in the configuration; the community strings cannot be set through a data record in the user program.
  However, you can activate the configured community strings by data record.
  Example:
  You have set SNMP to deactivated in the configuration of an S7-1500 CPU. You change the default community strings in the CPU properties and then load the configuration into the CPU.
  Then, you activate SNMP via data record transfer.
  Result: The changed community strings take effect.
- For S7-1500 CPUs with a firmware version < V3.0, the preset community strings ("public" and "private") are always in effect when SNMP is activated.

4.7.2 Activating/deactivating SNMP by data record transfer: Example for a CPU 1516-3 PN/DP

Introduction

You need to activate the SNMP for a CPU 1516-3 PN/DP to manage your network infrastructure, CPUs and IO devices with SNMP. The example below shows the 0xB071 data record being transferred to a PROFINET interface for this purpose.

Requirement

- CPU 1516-3 PN/DP as of FW version V2.0
- STEP 7 version V14 or higher

Solution

Transfer the data record 0xB071 to a PROFINET interface of the CPU. As a result, SNMP is enabled in all PROFINET interfaces of the CPU.

The following example shows how you can create the data record in a global data block and transfer it in a program cycle OB (for example, OB1) to the PROFINET interface (Local~PROFINET_interface_1).

To activate SNMP for the addressed PROFINET interface of the CPU 1516-3 PN/DP, follow these steps:

1. Create a global data block.
2. Assign a name, for example, "ActivateSnmp".
3. Under "Static", create the structure of the 0xB071 data record (in the figure: "snmpRecord") and other variables for transferring the data record. The following figure shows the data block structure "ActivateSnmp".

ActivateSnmp

	Name	Data type	Start value	Re...	Ac...	Wr...	Vis...	Comment
1	Static			☐	☐	☐	☐	
2	snmpWrite	Bool	TRUE	☐	☑	☑	☑	Start writing Data record 16#B071
3	snmpRecord	Struct		☐	☑	☑	☑	Data record 16#B071
4	blockID	UInt	16#F003	☐	☑	☑	☑	Data record ID
5	blockLength	UInt	8	☐	☑	☑	☑	Length of block
6	version	USInt	1	☐	☑	☑	☑	Byte 1 of blockversion
7	subversion	USInt	0	☐	☑	☑	☑	Byte 2 of blockversion
8	reserved	UInt	0	☐	☑	☑	☑	Reserverd for future usage
9	snmpControl	UDInt	1	☐	☑	☑	☑	0: deactivate SNMP, 1: activate SNMP
10	snmpWrDone	Bool	false	☐	☑	☑	☑	writing done
11	snmpWrError	Bool	false	☐	☑	☑	☑	error while writing
12	snmpWrStatus	DWord	16#0	☐	☑	☑	☑	status of writing

Figure 4-36 Structure of the global data block "ActivateSnmp"

4. Transfer the 0xB071 data record in an OB program cycle (for example, OB1), with the "WRREC" (write data record) instruction to the CPU 1516-3 PN/DP. You can find an example program in the next section.

Programming example for the data record transfer in the OB1

The data record 0xB071 is transferred in the following program code:

```
//---------------------------------------
// Start writing SNMP settings
//---------------------------------------
IF "ActivateSnmp".snmpWrite THEN
 IF (NOT "ActivateSnmp".snmpWriteDone)
 AND (NOT "ActivateSnmp".snmpWriteError) THEN
 "instWrrec_1"(REQ := "ActivateSnmp".snmpWrite,
 ID := "Local~PROFINET-Schnittstelle_1",
 INDEX := 16#B071,
 DONE => "ActivateSnmp".snmpWriteDone,
 ERROR => "ActivateSnmp".snmpWriteError,
 STATUS => "ActivateSnmp".snmpWriteStatus,
 RECORD := "ActivateSnmp".snmpRecord);
 END_IF;
 IF "ActivateSnmp".snmpWriteError THEN
 ; // add error handling
 END_IF;
 IF "ActivateSnmp".snmpWriteDone THEN
 "ActivateSnmp".snmpWrite := FALSE;
 END_IF;
END_IF;
```

Deactivating SNMP again

You can use the program code used above, with small changes, to deactivate SNMP. Assign the value "0" to the tag "ActivateSnmp".snmpRecord.snmpControl in the user program:

```
"ActivateSnmp".snmpRecord.snmpControl := 0;
```

The next time the "WRREC" instruction is called, SNMP will be deactivated again.

4.7.3 Activating/deactivating SNMP by data record transfer with S7-1500R/H CPUs

In S7-1500R/H systems, activate/deactivate SNMP in the user program as with standard CPUs. However, the PROFINET interfaces (X1, X2,...) of both CPUs have different hardware IDs. The PROFINET interface X1 of the left CPU, for example, has a different hardware ID as the PROFINET interface X1 of the right CPU. The S7-1500R/H system does not automatically synchronize the SNMP status (activated/deactivated) for both CPUs. An SNMP status (activated/deactivated) set by the "WRREC" instruction only affects the CPU whose PROFINET interface addresses the "WRREC" instruction.

Example:

The S7-1500R/H system is in the "RUN redundant" system state. If a PROFINET interface of the left CPU is addressed with the "WRREC" instruction, for example, the SNMP status of the left CPU is changed. The SNMP status of the right CPU remains unchanged. If the left CPU fails or is replaced, the unchanged SNMP status is in place after the SYNCUP.

Solution:

Call the "WRREC" instruction 2 times. In the first call of "WRREC", address the hardware ID of a PROFINET interface of the left CPU. Call the "WRREC" instruction again. This time, address the hardware ID of a PROFINET interface of the right CPU.

Hardware IDs for PROFINET interface X1:

- The PROFINET interface X1 of the left CPU has the hardware ID 65164 (default name: Local1~PROFINET-interface_1).
- The PROFINET interface X1 of the right CPU has the hardware ID 65364 (default name: Local2~PROFINET-interface_1).

The addressing by means of the respective hardware IDs of the PROFINET interfaces X1 is also used in the following example for calling the "WRREC" instruction for both R/H CPUs.

NOTE

Transferring the data record to the backup CPU

Transfer the data record to the addressed PROFINET interface of the backup CPU only after the S7-1500R/H system has reached the "Run REDUNDANT" system state. Otherwise, the data record cannot be transferred to the addressed PROFINET interface of the backup CPU.

When the S7-1500R/H system has reached the system state "Run REDUNDANT", the CPU redundancy error OB (OB72) is started. The "Fault_ID" tag of the OB72 contains the error code "B#16#03" or "B#16#06".

Example: WRREC calls for both R/H CPUs

To activate/deactivate SNMP for the addressed PROFINET interface of both the CPUs by transferring data records, proceed as follows:

1. Create a global data block.
2. Assign a name, for example, "ActivateSnmp".
3. Under "Static", create the structure of the 0xB071 data record (in the figure: "snmpRecord") and other tags for transferring the data record. The following figure shows the structure of the data block "ActivateSnmp".

ActivateSnmp

	Name	Data type	Start value	Re...	Ac...	Wr..	Vis...	Comment
1	Static							
2	snmpWrite	Bool	TRUE					Start writing Data record 16#B071
3	snmpRecord	Struct						Data record 16#B071
4	blockID	UInt	16#F003					Data record ID
5	blockLength	UInt	8					Length of block
6	version	USInt	1					Byte 1 of blockversion
7	subversion	USInt	0					Byte 2 of blockversion
8	reserved	UInt	0					Reserverd for future usage
9	snmpControl	UDInt	1					0: deactivate SNMP, 1: activate SNMP
10	plcLeft	Struct						Writing status of left plc
11	snmpWrDone	Bool	false					writing done
12	snmpWrError	Bool	false					error while writing
13	snmpWrStatus	DWord	16#0					status of writing
14	plcRight	Struct						Writing status of right plc
15	snmpWrDone	Bool	false					writing done
16	snmpWrError	Bool	false					error while writing
17	snmpWrStatus	DWord	16#0					status of writing

Figure 4-37 Structure of the global data block "ActivateSnmp"

4. Add the organization block "CPU redundancy error" (OB72) to your user program. You can find an example program for the OB72 in the next section.

5. Open the program cycle OB (OB1).
6. In the OB1, carry out two "WRREC" instructions for transferring the data record to the respective addressed PROFINET interface of both the CPUs. You can find an example program for the OB1 in the next section.
   Result: The 0xB071 data record was transferred to the PROFINET interface of both the CPUs addressed in each case.

Programming examples for the OB72 and OB1 organization blocks

Open the OB72 that has been added. With the following program code, determine whether the R/H system has assumed the "Run REDUNDANT" state and set the starting command for the "WRREC" instructions:

```
//-----------------------------------------
// Check redundancy state and set "snmpWrite"
//-----------------------------------------
IF #Fault_ID = B#16#03 OR #Fault_ID = B#16#06 THEN
 "ActivateSnmp".snmpWrite := TRUE;
END_IF;
```

Open the program cycle OB (OB1). With the following program code, you can run 2 "WRREC" instructions for transferring the data record to the respective addressed PROFINET interface of both the CPUs:

```
//---------------------------------------
// Start writing SNMP settings
//---------------------------------------
IF "ActivateSnmp".snmpWrite THEN
 IF (NOT "ActivateSnmp".plcLeft.snmpWrDone)
 AND (NOT "ActivateSnmp".plcLeft.snmpWrError) THEN
 // write SNMP settings for the left PLC
 "instWrrec_1"(REQ := "ActivateSnmp".snmpWrite,
 ID := "Local1~PROFINET_interface_1",
 INDEX := 16#B071,
 DONE => "ActivateSnmp".plcLeft.snmpWrDone,
 ERROR => "ActivateSnmp".plcLeft.snmpWrError,
 STATUS => "ActivateSnmp".plcLeft.snmpWrStatus,
 RECORD := "ActivateSnmp".snmpRecord);
 END_IF;
 IF "ActivateSnmp".plcLeft.snmpWrError THEN
 ; // add error handling for left plc
 END_IF;
 IF (NOT "ActivateSnmp".plcRight.snmpWrDone)
 AND (NOT "ActivateSnmp".plcRight.snmpWrError) THEN
 // write SNMP settings for the right PLC
 "instWrrec_2"(REQ := "ActivateSnmp".snmpWrite,
 ID := "Local2~PROFINET_interface_1",
 INDEX := 16#B071,
 DONE => "ActivateSnmp".plcRight.snmpWrDone,
 ERROR => "ActivateSnmp".plcRight.snmpWrError,
 STATUS =>
 "ActivateSnmp".plcRight.snmpWrStatus,
 RECORD := "ActivateSnmp".snmpRecord);
 END_IF;
 IF "ActivateSnmp".plcRight.snmpWrError THEN
 ; // add error handling for right plc
 END_IF;
 IF "ActivateSnmp".plcLeft.snmpWrDone
```

```
 AND "ActivateSnmp".plcRight.snmpWrDone THEN
 "ActivateSnmp".snmpWrite := FALSE;
 END_IF;
END_IF;
```

Deactivating SNMP again

You can use the program code used above, with small changes, to deactivate SNMP. Assign the value "0" to the tag "ActivateSnmp".snmpRecord.snmpControl in the user program:

```
"ActivateSnmp".snmpRecord.snmpControl := 0;
```

The next time the "WRREC" instructions are called, SNMP will be deactivated again.

PG communication

5

Properties

Using PG communication, the CPU or another module capable of communication exchanges data with an engineering station (for example PG, PC). The data exchange is possible via PROFIBUS and PROFINET subnets. The gateway between S7 subnets is also supported.

PG communication provides functions needed to load programs and configuration data, run tests, and evaluate diagnostic information. These functions are integrated in the operating system of the module capable of communication.

NOTE

As of TIA Portal version V17, the TLS (Transport Layer Security) protocol is supported for programming device/HMI communication to secure the data exchange between programming device/PC and CPU using standardized security mechanisms.

For more information, refer to the following sections:

- Requirements for secure communication (Page 65)
- Secure PG/HMI communication (Page 93)

Requirements

- The PG/PC is physically connected to the communication-capable module.
- If the communication-capable module is to be reached via S7 routing, the hardware configuration has to be loaded in the participating stations (S7 router and end point).

Procedure for connecting online

You must establish an online connection to the CPU for the programming device communication:

1. Select the CPU in the project tree in STEP 7.
2. Select the "Online > Go online" menu command.
3. In the "Go online" dialog, make the following settings for your online connection:
   - Select interface type (e.g. PN/IE) in the "Type of PG/PC interface" drop-down list.
   - In the "PG/PC interface" drop-down list, select the PG/PC interface (e.g. Ind. Ethernet card) you want to use to establish the online connection.
   - Select the interface or the S7 subnet with which the programming device/PC is physically connected from the "Connection to interface/subnet" drop-down list.
   - If the communication-capable module can be reached via an S7 router (gateway), select the S7 router that connects the subnets in question from the "1st gateway" drop-down list.

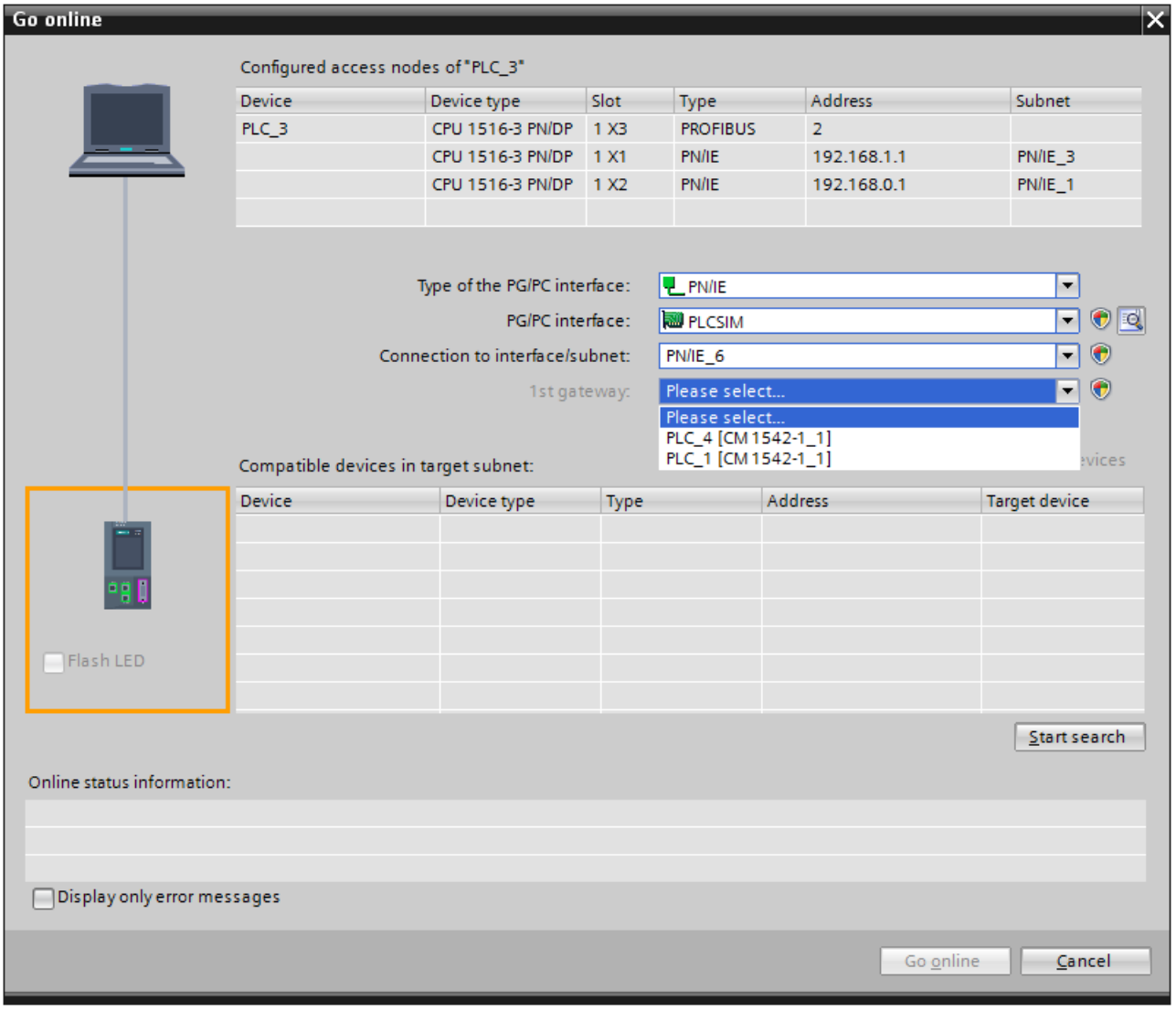


Figure 5-1 Setting up PG communication

4. Click "Start search".
   All devices that you can address with PG communication appear shortly thereafter in the table "Compatible devices in target subnet".
5. In the "Compatible devices in target subnet" table, select the relevant CPU and confirm with "Go online".

More information

You can find more information on "Go online" in the STEP 7 online help.

HMI communication

6

Properties

Using HMI communication, one or more HMI devices (for example HMI Basic/Comfort/Mobile Panel) exchanges data with a CPU for operator control and monitoring with via the PROFINET or PROFIBUS DP interface. The data exchange is via HMI connections.

If you want to set up several HMI connections to a CPU, use for example:

- The PROFINET and PROFIBUS DP interfaces of the CPU
- CPs and CMs with the relevant interfaces

NOTE

As of TIA Portal version V17, the TLS (Transport Layer Security) protocol is supported for programming device/HMI communication to secure the data exchange between programming device/PC and CPU using standardized security mechanisms.

For more information, refer to the following sections:

- Requirements for secure communication (Page 65)
- Secure PG/HMI communication (Page 93)

Procedure for setting up HMI communication

As soon as you drag-and-drop a tag, for example a tag from a global data block into an HMI screen or into the HMI tag table, STEP 7 automatically sets up an HMI connection. Alternatively, you can also set up the HMI connection yourself.

To set up an HMI connection, follow these steps.

1. Configure the HMI device in an existing configuration with a CPU in the network view of the Devices & networks editor of STEP 7.
2. Select the "Connections" button and then "HMI connection" from the drop-down list.
3. Drag-and-drop a line between the end points of the connection (HMI device and CPU). The end points are highlighted in color. If the required S7 subnet does not yet exist, it is created automatically.

4. In the "Connections" tab, select the row of the HMI connection.
   In the "General" area of the "Properties" tab, you see the properties of the HMI connection, some of which you can change.

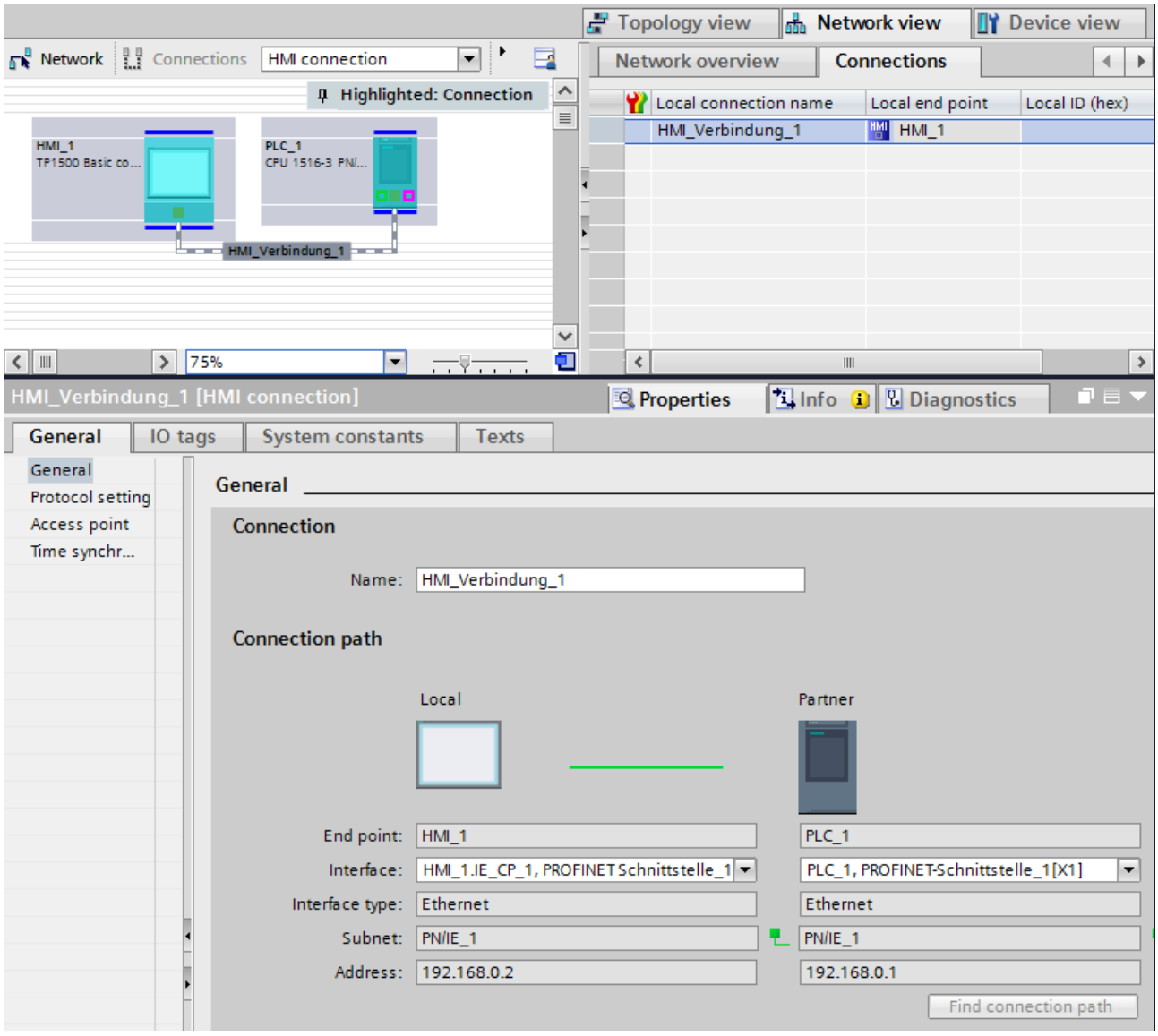


Figure 6-1 Setting up HMI communication

5. Download the hardware configuration to the CPU.
6. Download the hardware configuration to the HMI device.

More information

You can find information on S7 routing for HMI connections in the section S7 Routing (Page 352).
You can find more information on setting up HMI connections in the STEP 7 online help.

Open User Communication 7

7.1 Overview of Open User Communication

Features of Open User Communication

Through Open User Communication, also called "open communication", the CPU exchanges data with another device capable of communication. Open User Communication has the following features and characteristics:

- Open standard (communication partners can be two SIMATIC CPUs or a SIMATIC CPU and a suitable third-party device).
- Communication via various protocols (in STEP 7 known as "Connection types")
- High degree of flexibility in terms of the data structures transferred; this allows open data exchange with any communications devices as long as these support the connection types available.
- Secure Communication: To protect your automation system, you can exchange data securely over Open User Communication. With Secure Open User Communication, the data is sent signed and encrypted, see also Secure Open User Communication (Page 75).
- Open User Communication is possible in various automation systems, see technical specifications of the respective manuals.
  Examples:
  - Integrated PROFINET / Ind. Ethernet interfaces of CPUs (S7-1500, ET 200SP CPU, S7-1500 Software Controller, CPUs 1513/1516pro 2 PN)
  - PROFINET / Ind. Ethernet interfaces of communications modules (for example CP 1543-1, CM 1542-1, CP 1543SP-1)

Information on Secure Open User Communication is available in the section Secure Communication (Page 43).

Information on S7-1500R/H

You can find information on Open User Communication with the S7-1500R/H redundant system in section Communication with the redundant system S7-1500R/H (Page 384).

7.2 Protocols for Open User Communication

Protocols for Open User Communication

The following protocols are available for open communication:

Table 7-1 Transport protocols for open communication

Transport protocol	Via interface
TCP according to RFC 793	PROFINET/Industrial Ethernet
ISO-on-TCP according to RFC 1006 (Class 4)	PROFINET/Industrial Ethernet
ISO according to ISO/IEC 8073	Industrial Ethernet (only CP 1543-1)
UDP according to RFC 768	PROFINET/Industrial Ethernet
FDL	PROFIBUS

Table 7-2 Application protocols for open communication

Application protocol	Used transport protocol
Modbus TCP	**TCP** according to RFC 793
E-mail	**TCP** according to RFC 793
FTP	**TCP** according to RFC 793

TCP, ISO-on-TCP, ISO, UDP

Prior to data transfer, these protocols (except UDP) establish a transport connection to the communications partner. Connection-oriented protocols are used when potential loss of data needs to be avoided.

The following is possible with UDP:

- Unicast to one device or broadcast to all devices on PROFINET via the PROFINET interface of the CPU or the Industrial Ethernet interface of the CP 1543-1
- Multicast to all recipients of a multicast group via the PROFINET interface of the CPU* or the PROFINET / Industrial Ethernet interface of the CP 1543-1

* As of firmware version V2.0, the PROFINET interface of the CPU supports a maximum of 5 multicast groups

Maximum user data length: You can find the maximum user data lengths supported in the technical specifications of the respective device manuals.

Protocol for communication via PROFIBUS: FDL

Data transfer via an FDL connection (Fieldbus Data Link) is suitable for the transfer of related blocks of data to a communications partner on PROFIBUS that supports the sending and receiving of data according to the FDL service SDA (Send Data with Acknowledge) according to EN 50170, Vol 2. Both partners have the same rights; in other words, each partner can initiate sending and receiving event-driven.

In keeping with the FDL service SDN (Send Data with No Acknowledge) according to EN 50170, Vol 2, the following is possible with FDL:

- Broadcast to all devices on PROFIBUS via the PROFIBUS interface of the CM 1542-5
- Multicast to all recipients of a multicast group via the PROFIBUS interface of the CM 1542-5

Modbus TCP

The Modbus protocol is a communication protocol with linear topology based on a master/slave architecture. In the Modbus TCP (Transmission Control Protocol), the data is transmitted as TCP/IP packets.

Communication is controlled solely by suitable instructions in the user program.

E-mail and FTP

You can use email to send for example, data block contents (e.g. process data) as an attachment.

You can use the FTP connection (FTP = File Transfer Protocol) to transmit files to and from S7 devices.

The communication is controlled by instructions in the user program at the client end.

Application example: MQTT Publisher for the SIMATIC S7-1500 CPU

The "Message Queue Telemetry Transport" (MQTT) is a simple protocol on the TCP/IP level. It is suitable for the exchange of messages between devices with lower functionality and for the transfer via unreliable networks.

The application example provides a function block with which you can implement the MQTT protocol into the SIMATIC S7-1500.

You can find the application example on the Internet (https://support.industry.siemens.com/cs/ww/en/view/109772284).

Block library for SYSLOG messages

Syslog is a simply structured binary profile on UDP/IP level. It enables applications to send messages, warnings or error states to a Syslog server. Syslog is typically used for computer system management and security monitoring, and has established itself as a standard in the field of protocols.

The "LSyslog" library offers you a solution to implement the Syslog protocol in an S7-1500. In addition to the library, an application example is provided that shows you how to generate Syslog messages in your controller and send them to the Syslog server.

You can find the block library "LSyslog" and the associated application example on the Internet (https://support.industry.siemens.com/cs/ww/en/view/51929235).

7.3 Instructions for Open User Communication

Introduction

You set up Open User Communication via the corresponding connection (for example, TCP connection) as follows:

- By programming in the user programs of the communications partners or
- By configuring the connection in STEP 7 in the hardware and network editor

Regardless of whether you set up the connection by programming or configuring, instructions are always required in the user programs of both communications partners for sending and receiving the data.

Setting up the connection via the user program

If the connection is set up by programming, the connection establishment and termination is implemented using instructions in the user program.

In certain application areas, it is advantageous not to set up the communication connections statically via hardware configuration, but via the user program instead. You can set up the connections via a specific application program-controlled and therefore when necessary. Programmed connection setup also allows connection resources to be released following data transfer.

A data structure is necessary for each communications connection that contains the parameters for establishing the connection (for example system data type "TCON_IP_v4" for TCP).

The system data types (SDT) are provided by the system and have a predefined structure that cannot be changed.

The various protocols have their own data structures (see table below). The parameters are stored in a data block ("connection description DB") for example of the system data type TCON_IP_v4.

There are two ways in which you can specify the DB with the data structure:

- Recommendation: Have the data block created automatically in the properties in the program editor during parameter assignment of the connection for the TSEND_C, TRCV_C and TCON instructions.
- Create the data block manually, assign parameters to it and write it directly to the instruction
  Necessary for:
  - Secure OUC
  - Connection over DNS
  - E-mail
  - FTP

You can modify the connection parameters in the "connection description DB".

This FAQ (https://support.industry.siemens.com/cs/ww/en/view/58875807) describes how to program the TCON instruction to set up a connection for Open User Communication between two S7-1500 CPUs.

Protocols, system data types and employable instructions for programmed setup

The following table shows the protocols of the Open User Communication and the matching system data types and instructions.

Table 7-3 Instructions for programmed setup of the connection

Protocol	System data type	Instructions
TCP	• TCON_QDN • TCON_IP_v4	Establish connection and send/receive data via: • TSEND_C/TRCV_C or • TCON, TSEND/TRCV or • TCON, TUSEND/TURCV (connection can be terminated via TDISCON)
ISO-on-TCP	• TCON_IP_RFC	
ISO according to ISO/IEC 8073 (Class 4)	• TCON_ISOnative[1] • TCON_Configured	
UDP	• TCON_IP_v4 • TADDR_Param • TADDR_SEND_QDN • TADDR_RCV_IP	Establish connection and send/receive data via: • TSEND_C/TRCV_C • TUSEND/TURCV/TRCV (connection can be terminated via TDISCON)
FDL[1]	• TCON_FDL	Establish connection and send/receive data via: • TSEND_C/TRCV_C or • TCON, TSEND/TRCV or • TCON, TUSEND/TURCV (connection can be terminated via TDISCON)
Modbus TCP	• TCON_IP_v4 • TCON_QDN • TCON_Configured	• MB_CLIENT • MB_SERVER
Email	• TMAIL_v4 • TMAIL_v6 • TMAIL_FQDN	• TMAIL_C
FTP[2]	• FTP_CONNECT_IPV4[3] • FTP_CONNECT_IPV6[3] • FTP_CONNECT_NAME[3]	• FTP_CMD

[1] This protocol can only be used with the CM 1542-5

[2] This protocol can only be used with the CP 1543-1

[3] User-defined data type

The following table shows you the different connections of the Secure Open User Communication and the matching system data types and instructions.

Secure OUC connection	System data type	Instructions
Secure TCP connection from an S7-1500 CPU as TLS client to a third-party PLC (TLS server) Secure TCP connection from an S7-1500 CPU as TLS server to a third-party PLC (TLS client)	• TCON_QDN_SEC	• TSEND_C/TRCV_C • TCON, TSEND/TRCV
Secure TCP connection between two S7-1500 stations	• TCON_IP_V4_SEC[1]	
Secure connection to a mail server[2]	• TMAIL_V4_SEC • TMAIL_QDN_SEC	• TMAIL_C (V5.0 or higher)
Secure Modbus TCP connection	• TCON_IP_V4_SEC[1]	• MB_Client • MB_Server
	• TCON_QDN_SEC	

[1] Also possible for CP 1543-1

[2] Secure connection to a mail server also possible with CP1543-1 und TMAIL_C (V4.0)

Setting up the connection with connection configuration

When setting up through the configuration of the connection, the address parameters of the connection are specified in the hardware and network editor of STEP 7.
To send and receive the data, use the same instructions as when the connections are set up by programming:

Table 7-4 Instructions for sending/receiving with configured connections

Protocol	Send/receive with configured connections
Supported instructions:	
TCP	Send/receive data via: • TSEND_C/TRCV_C or • TSEND/TRCV or • TUSEND/TURCV
ISO-on-TCP	
ISO according to ISO/IEC 8073 (Class 4)	
UDP	Send/receive data via: • TSEND_C/TRCV_C or • TUSEND/TURCV
FDL	Send/receive data via: • TSEND_C/TRCV_C or • TSEND/TRCV or • TUSEND/TURCV
Modbus TCP	Not supported
Email	Not supported
FTP	Not supported

Additional instructions for open communication

You can use the following instructions for connections set up in the user program as well as for configured connections:

- T_RESET: Terminating and establishing a connection
- T_DIAG: Check the connection

Basic examples for Open User Communication

The Siemens Online Support offers you function blocks (FBs) that facilitate the handling of the instructions of the Open User Communication. You can find the function block with corresponding examples on the Internet (https://support.industry.siemens.com/cs/ww/en/view/109747710).

More information

The STEP 7 online help describes:

- The user and system data types
- The instructions for open communication
- The connection parameters

You will find information about the allocation and release of connection resources in the section Allocation of connection resources (Page 372).

Information about the Secure Open User Communication is available in the section Secure Open User Communication (Page 75).

7.4 Open User Communication with addressing via domain names

As of firmware version V2.0, S7-1500 CPUs, ET 200SP CPUs and the CPUs 1513/1516pro-2 PN support Open User Communication with addressing via Domain Name System (DNS). A DNS client is integrated in the CPU. In the case of communication via DNS, you use domain names as an alias for IP addresses to address communication partners. Addressing of the communication partners via domain names is possible for open communication via TCP and UDP.

At least one DNS server must be located in your network as a requirement for communication via DNS.

The S7-1500 software controller supports communication via DNS for all interfaces that are assigned to the software controller.

Setting up communication via DNS

The DNS client of the CPU must know the IPv4 address of at least one DNS server so that a CPU can establish a connection to a communication partner via its domain name. The CPU supports up to 4 different DNS servers.

To set up communication via domain names for an S7-1500 CPU, follow these steps:

1. Select the CPU in the network view of STEP 7.
2. In the Inspector window, navigate to "Properties" > "General" > "Advanced configuration" > "DNS configuration".
3. Enter the IPv4 address of a DNS server in the "DNS server addresses" column of the "Server list" table.
   You can enter up to 4 IPv4 addresses of DNS servers.

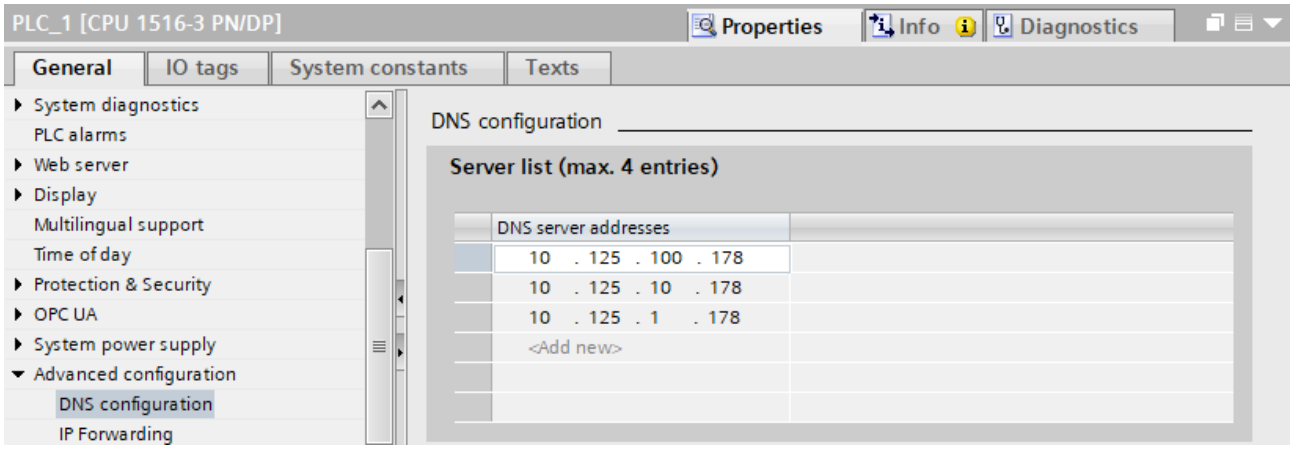


Figure 7-1 Entering DNS server addresses using a CPU 1516-3 PN/DP as an example

Setting up a TCP connection via the domain name of the communication partner

For TCP communication via the domain name you need to create a data block with the TCON_QDN system data type yourself, assign parameters and call it directly at the instruction. The TCON, TSEND_C and TRCV_C instructions support the system data type TCON_QDN:
To set up a TCP connection via the domain name of the communication partner, follow these steps:

1. Create a global data block in the project tree.
2. Define a tag of the data type TCON_QDN in the global data block.
   The example below shows the global data block "Data_block_1" in which the tag "DNS Connection1" of data type TCON_QDN is defined.

Data_block_1

	Name	Datentyp	Startwert	Kommentar
1	Static			
2	DNS Connection1	TCON_QDN		
3	InterfaceId	HW_ANY	0	not relevant
4	ID	CONN_OUC	16#0	connection reference / identifier
5	ConnectionType	Byte	16#0B	type of connection: 16#0B=11=TCP/IP, 16#13=19=UDP
6	ActiveEstablished	Bool	false	active/passive connection establishment
7	RemoteQDN	String[254]	''	fully or partially qualified domain name of remote partner
8	RemotePort	UInt	0	remote UDP / TCP port number
9	LocalPort	UInt	0	local UDP / TCP port number

Figure 7-2 Data type TCON_QDN

3. Program the parameters of the TCP connection (for example the fully qualified domain name (FQDN)) in the tag of data type TCON_QDN.

4. Create a TCON instruction in the program editor.
5. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_QDN.
   In the example below, the CONNECT parameter of the TCON instruction is interconnected with the tag "DNS connection1" (data type TCON_QDN).

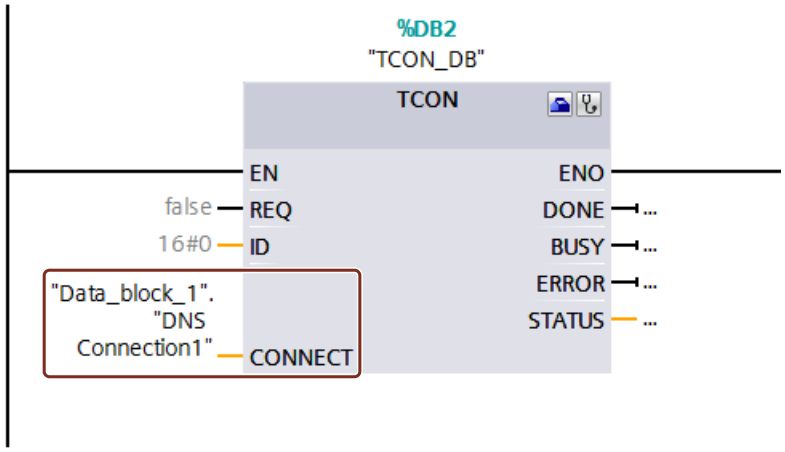


Figure 7-3 TCON instruction

Addressing a UDP connection via the domain name of the communication partner

For S7-1500 CPUs as of firmware version V2.0, you can address the recipient with its fully qualified domain name (FQDN) when sending data via UDP. With the instruction TUSEND at the parameter ADDR, you hereby reference a structure of the type TADDR_SEND_QDN.
The receiver can return an IPv4 or an IPv6 address. With the TURCV instruction at the ADDR parameter, you therefore reference a structure of the TADDR_RCV_IP type. Only this structure can include both IP address types.

NOTE

Network load

In contrast to the TCP the UDP protocol does not work connection-oriented. For every edge at the block parameter REQ, the TUSEND or TURCV command performs queries of the DNS server. This can lead to high network load or load on the DNS server.

Additional information

You can find more information about the system data types TCON_QDN, TADDR_SEND_QDN and TADDR_RCV_IP in the STEP 7 online help.
How to set up a secure TCP connection via the domain name of the communication partner is described in the section Secure Open User Communication (Page 75).

7.5 Setting up Open User Communication via TCP, ISO-on-TCP, UDP and ISO

Configuring a connection for the TSEND_C, TRCV_C or TCON instructions

Requirement: A TSEND_C, TRCV_C or TCON instruction is created in the programming editor.

1. Select a TCON, TSEND_C or TRCV_C block of Open User Communication in the program editor.
2. Open the "Properties > Configuration" tab in the inspector window.
3. Select the "Connection parameters" group. Until you select a connection partner, only the empty drop-down list for the partner end point is enabled. All other input options are disabled.

   The connection parameters already known are displayed:

   – Name of the local end point
   – Interface of the local end point

– IPv4 address of the local end point

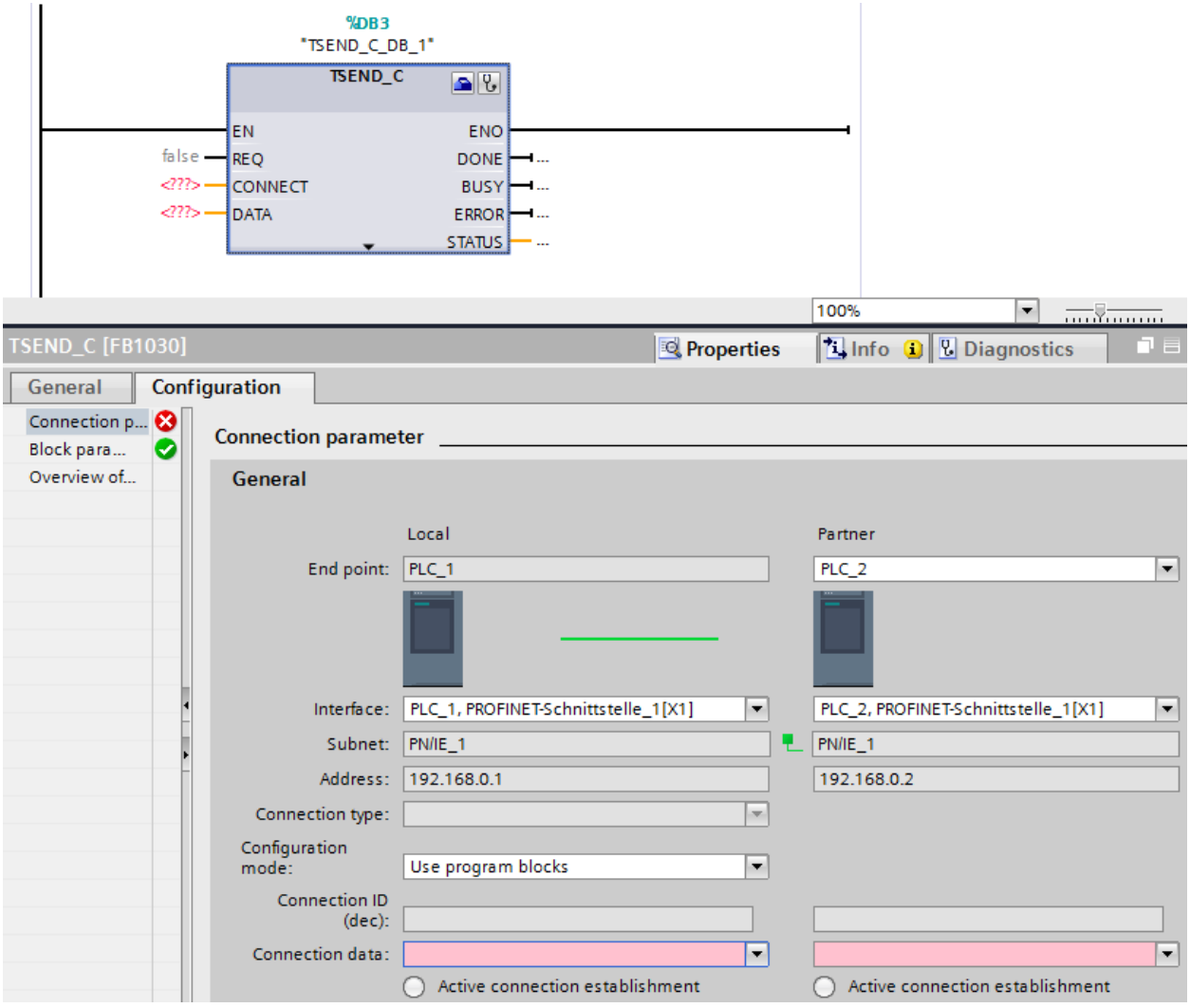


Figure 7-4 Connection parameters for TSEND_C

4. In the drop-down list box of the partner end point, select a connection partner. You can select an unspecified device or a CPU in the project as the communication partner. Certain connection parameters are then entered automatically.
   The following parameters are set:
   – Name of the partner end point
   – Interface of the partner end point
   – IPv4 address of the partner end point

   If the connection partners are networked, the name of the subnet is displayed.
5. In the "Configuration type" drop-down list, select between using program blocks or configured connections.

6. Select an existing connection description DB in the "Connection data" drop-down list or for configured connections select an existing connection under "Connection name". You can also create a new connection description DB or a new configured connection. Later, you can still select other connection description DBs or configured connections or change the names of the connection description DBs in order to create new data blocks:
   - You can also see the selected data block at the interconnection of the CONNECT input parameter of the selected TCON, TSEND_C or TRCV_C instruction.
   - If you have already specified a connection description DB for the connection partner using the CONNECT parameter of the TCON, TSEND_C or TRCV_C instruction, you can either use this DB or create a new DB.
   - If you edit the name of the displayed data block in the drop-down list, a new data block with the changed name but with the same structure and content is generated and used for the connection.
   - Changed names of a data block must be unique in the context of the communication partner.
   - A connection description DB must have the structure TCON_Param, TCON_IP_v4 or TCON_IP_RFC, depending on CPU type and connection.
   - A data block cannot be selected for an unspecified partner.

   Additional values are determined and entered after the selection or creation of the connection description DB or configured connection.
   The following is valid for specified connection partners:
   - ISO-on-TCP connection type
   - Connection ID with default of 1
   - Active connection establishment by local partner
   - TSAP ID
     for S7-1200/1500: E0.01.49.53.4F.6F.6E.54.43.50.2D.31

   The following is valid for unspecified connection partners:
   - TCP connection type
   - Partner port 2000

   The following applies for a configured connection with a specified connection partner:
   - TCP connection type
   - Connection ID with default of 257
   - Active connection establishment by local partner
   - Partner port 2000

   The following applies for a configured connection with an unspecified connection partner:
   - TCP connection type
   - Local port 2000
7. Enter a connection ID as needed for the connection partner. No connection ID can be assigned to an unspecified partner.

NOTE

You must enter a unique value for the connection ID at a known connection partner. The uniqueness of the connection ID is not checked by the connection parameter settings and there is no default value entered for the connection ID when you create a new connection.

8. Select the desired connection type in the relevant drop-down list. Default values are set for the address details depending on the connection type. You can choose between the following:
   - TCP
   - ISO-on-TCP
   - UDP
   - ISO (only with Configuration mode "Use configured connection")

   You can edit the input boxes in the address details. Depending on the selected protocol, you can edit the ports (for TCP and UDP) or the TSAPs (for ISO-on-TCP and ISO).
9. Use the "Active connection establishment" check box to set the connection establishment characteristics for TCP, ISO and ISO-on-TCP. You can decide which communication partner establishes the connection actively.

Changed values are checked immediately for input errors by the connection configuration and entered in the data block for the connection description.

NOTE

Open User Communication between two communication partners can only work when the program section for the partner end point has been downloaded to the hardware. To achieve fully functional communication, make sure that you load not only the connection description of the local CPU on the device but also that of the partner CPU as well.

Configuring connections, e.g. for TSEND/TRCV

If you want to use the instructions for TSEND/TRCV for open communication, for example, you first need to configure a connection (e.g. TCP connection).

To configure a TCP connection, follow these steps:

1. Configure the communications partners in the network view of the Devices & networks editor of STEP 7.
2. Click the "Connections" button and select the "TCP connection" connection type from the drop-down list.
3. Using drag-and-drop, connect the communication partner with each other (via an interface or local end point). If the required S7 subnet does not yet exist, it is created automatically.

   You can also set up a connection to unspecified partners.
4. Select the created connection in the network view.
5. Set the properties of the connection in the "Properties" tab in the "General" area, for example the name of the connection and the interfaces of the communications partner that will be used.

   For connections to an unspecified partner, set the address of the partner.
   You can find the local ID (reference of the connection in the user program) in the "Local ID" area.
6. In the Project tree, select the "Program blocks" folder for one of the CPUs and open OB1 in the folder by double-clicking on it. The program editor opens.
7. Select the required instruction from the "Instructions" task card, "Communication" area, "Open user communication", for example TSEND and drag it to a network of OB1.
8. At the ID parameter of the instruction, assign the local ID of the configured connection to be used for the transmission of data.

9. Interconnect the "DATA" parameter of the TSEND instruction with the user data, for example in a data block.
10. Download the hardware configuration and user program to the CPU.

Based on the procedure described above, set up the connection on the partner CPU with the instruction for receiving, TRCV, and download it to the CPU.

Point to note with ISO connections with CP 1543-1

If you use the "ISO connection" connection type, you will need to select the "Use ISO protocol" check box in the properties of the CP so that addressing using MAC addresses will work.

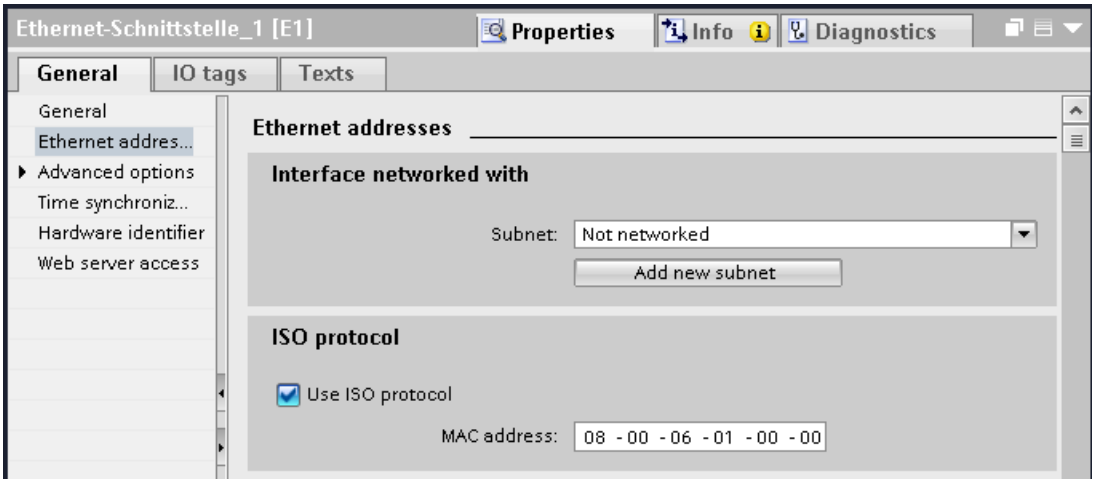


Figure 7-5 Select CP 1543-1 ISO protocol

Additional information

The STEP 7 online help describes:

- The instructions for open communication
- The connection parameters

This FAQ (https://support.industry.siemens.com/cs/ww/en/view/109479564) describes how the instructions TSEND_C and TRCV_C behave in the S7-1500.

7.6 Setting up communication over FDL

Requirements

- Configuration software: STEP 7 Professional V14
- End point of the connection: CPU S7-1500 firmware version V2.0 or higher with communication module CM 1542-5 with firmware version V2.0

Setting up a configured FDL connection

Proceed as follows to set up a configured FDL connection in STEP 7:

1. Create a TSEND_C instruction in the program editor.
2. Select the TSEND_C instruction and go to "Properties" > "General" > "Connection parameters" in the Inspector window.
3. Under End point, select the partner end point. Use one of the two partner end points below:
   - CPU S7-1500 with CM 1542-5
   - Unspecified
4. Under Configuration type, select "Use configured connection".
5. Under Connection type, select "FDL".
6. Under Interface, select the following interfaces:
   - Local: PROFIBUS interface of CM 1542-5
   - Specified partner: PROFIBUS interface of CM 1542-5
7. Under Connection data, select the setting <new>.

The figure below shows a fully configured FDL connection in STEP 7.

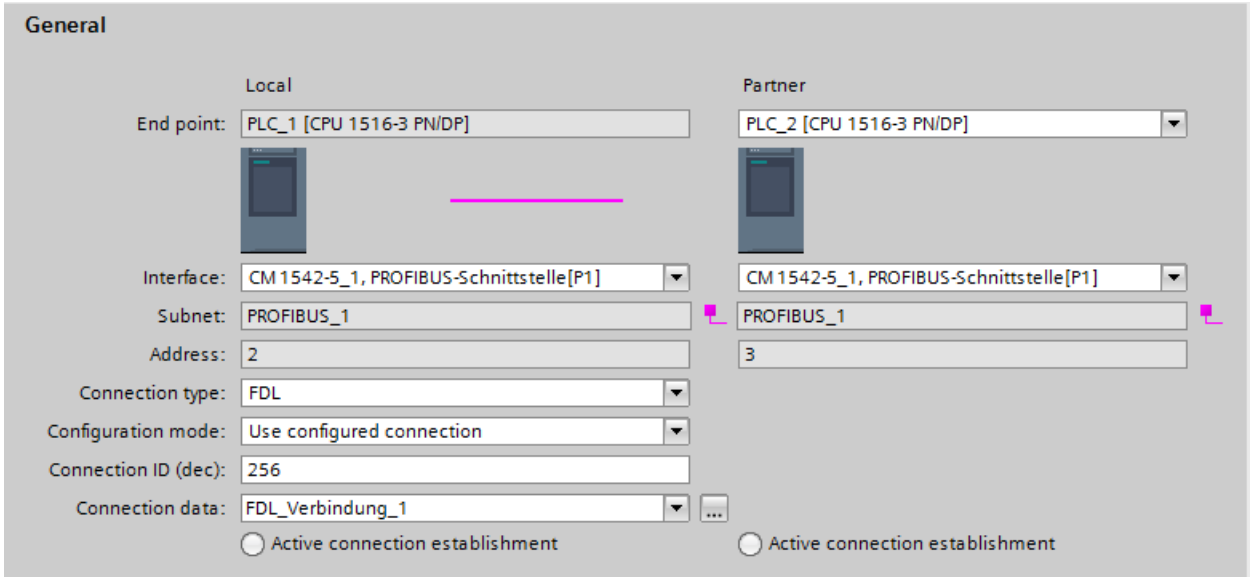


Figure 7-6 Configuring the FDL connection

Setting up an FDL connection in the user program

For communication via FDL, you need to create the data block of the TCON_FDL system data type yourself in each case, assign parameters and call it directly at the instruction. Follow these steps:

1. Create a global data block in the project tree.

2. In the global data block, define a tag of the data type TCON_FDL.
   The example below shows the global data block "FDL_connection" in which the tag "FDL_connection" of the data type TCON_FDL is defined.

FDL_connection

		Name	Data type	Start val..	R...	A...	...	V...	S...	Super...	Comment
1		Static									
2		FDL_connection	TCON_FDL								
3		InterfaceId	HW_ANY	0							HW identifier of PB interface submodule
4		ID	CONN_OUC	16#0							connection reference / identifier
5		ConnectionType	Byte	16#15							type of connection: 21= FDL connection
6		ActiveEstablished	Bool	false							active/passive connection establishment
7		ServiceId	Byte	16#0							service id: 0 – default, 1 – SDA, 2 – SDN
8		RemotePBAddress	Byte	16#0							remote ProfiBus partner address
9		LocalPBAddress	Byte	16#0							local ProfiBus partner address
10		RemoteLSAP	Byte	16#0							remote PB link-layer service access point
11		LocalLSAP	Byte	16#0							local PB link-layer service access point

Figure 7-7 Programming an FDL connection

3. Program the parameters of the FDL connection (e.g. the PROFIBUS addresses) in the tag of the data type TCON_FDL.
4. Create a TCON instruction in the program editor.
5. Interconnect the CONNECT parameter of the TCON instruction with the tag of the data type TCON_FDL.
   In the example below, the CONNECT parameter of the TCON instruction is interconnected with the tag "FDL_Connection" (data type TCON_FDL).

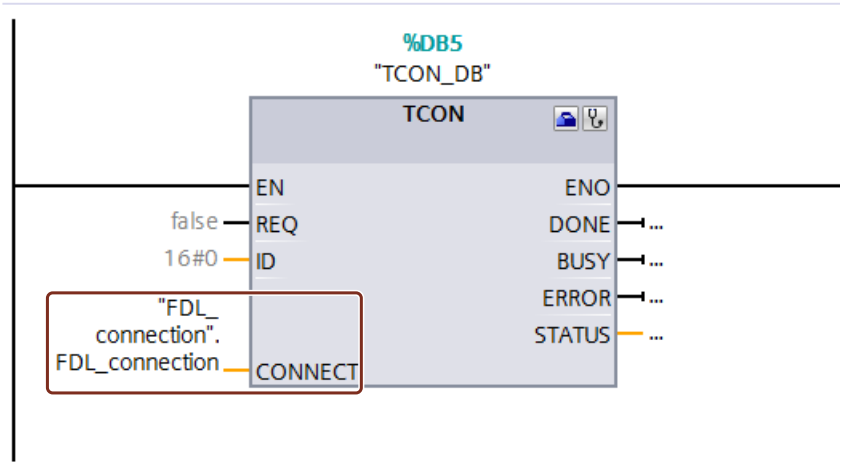


Figure 7-8 Example: TCON Instruction for FDL connection

7.7 Setting up communication with Modbus TCP

Setting up a connection for Modbus TCP via the user program

The parameter assignment takes place in the program editor at the instruction MB_CLIENT or MB_SERVER.

Procedure for setting up communication using Modbus TCP

The MB_CLIENT instruction communicates as a Modbus TCP client via the TCP connection. You establish a connection between the client and the server with the instruction, send Modbus requests to the server and receive the corresponding Modbus responses. You also control the setup of the TCP connection with this instruction.

The MB_SERVER instruction communicates as a Modbus TCP server via the TCP connection. The instruction processes connection requests of a Modbus client, receives and processes Modbus requests and sends responses. You also control the setup of the TCP connection.

Requirement: The client can reach the server via IP communication in the network.

1. Configure an S7-1500 automation system with CPU in the network view of the Devices & networks editor of STEP 7.
2. In the Project tree, select the "Program blocks" folder and open OB1 in the folder by double-clicking on it. The program editor opens.
3. Select the required instruction, for example MB_CLIENT, from the "Instructions" task card, "Communication" area, "Other", "MODBUS TCP" and drag it to a network of OB1.
4. Assign the parameters of the MB_CLIENT or MB_SERVER instruction. Observe the following rules:

   An IPv4 server address must be specified for each MB_CLIENT connection.

   Each MB_CLIENT or MB_SERVER connection must use a unique instance DB with one of the data structures TCON_IP_v4, TCON_QDN or TCON_Configured.

   Each connection requires a unique connection ID. The connection ID and instance DB belong together in pairs and must be unique for each connection.

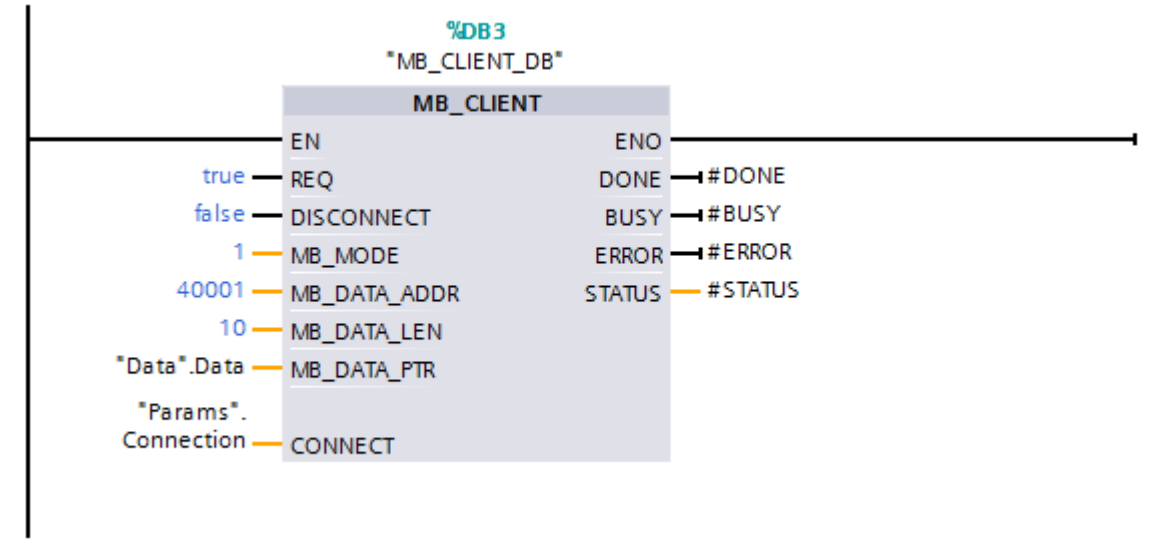


Figure 7-9 MB_CLIENT

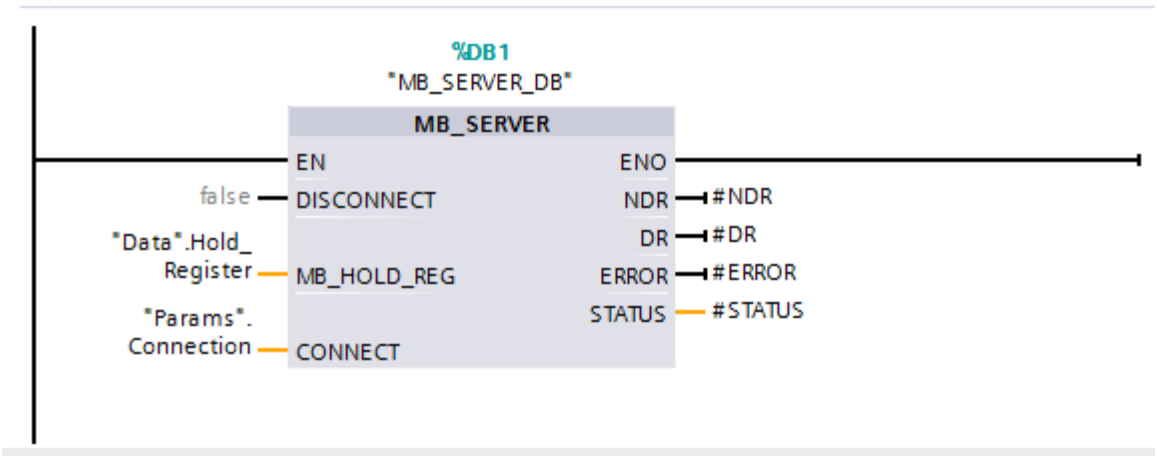


Figure 7-10 MB_SERVER

5. Download the hardware configuration and user program to the CPU.

Redundant communication over Modbus TCP

Assign parameters for redundant communication via Modbus TCP using the MB_RED_CLIENT or MB_RED_SERVER instructions:
Instruction MB_RED_CLIENT You use the instruction "MB_RED_CLIENT" to establish a redundant connection between the client and the server, send Modbus requests, receive responses and control connection termination of the Modbus TCP client.
Instruction MB_RED_SERVER The "MB_RED_SERVER" instruction processes connection requests of a Modbus TCP client, receives and processes Modbus requests and sends responses. The CPUs can:

- processing multiple server connections and
- accepting multiple connections from different clients simultaneously at one server port.

For more information on the MB_RED_CLIENT or MB_RED_SERVER instructions, refer to the STEP 7 online help.

Modbus TCP server as gateway to Modbus RTU

If you use a Modbus TCP server as a gateway to a Modbus RTU protocol, address the slave device in the serial network using the static parameter, MB_UNIT_ID. The MB_UNIT_ID parameter corresponds to the field of the slave address in the Modbus RTU protocol. The MB_UNIT_ID parameter in this case would forward the request to the correct Modbus RTU slave address.
You do not have to program the gateway function yourself.
You can find the MB_UNIT_ID parameter in the instance data block associated with MB_CLIENT instruction.
You can find more information on the MB_UNIT_ID parameter in the STEP 7 online help.

Reference

- This FAQ (https://support.industry.siemens.com/cs/ww/en/view/94766380) describes how to program and configure the Modbus TCP communication between two S7-1500 CPUs.
- This FAQ (https://support.industry.siemens.com/cs/ww/en/view/102020340) describes how to program and configure Modbus TCP communication between an S7-1500 CPU and an S7-1200 CPU.

7.8 Setting up communication via e-mail

Setting up a connection for e-mail via the user program

For communication using e-mail, you need to create the data block of the relevant system data type yourself, assign parameters and call the instruction directly. This procedure is introduced below.

Procedure for setting up communication using e-mail

A CPU can send e-mails. To send e-mails from the user program of the CPU, use the TMAIL_C instruction.

Requirement: The SMTP server can be reached via the IPv4 network.

1. Configure an S7-1500 automation system with CPU in the network view of the Devices & networks editor of STEP 7.
2. Assign parameters to the instruction TMAIL_C, for example enter the subject of the e-mail in Subject.
3. In a global data block create a variable of the type TMAIL_v4, TMAIL_v6 (only CP 1543-1) or TMAIL_FQDN (only CP 1543-1).
4. Set the connecting parameters of the TCP connection in the variable in the "Start value" column. Enter the IPv4 address of the mail server, for example, for the "MailServerAddress" (for TMAIL_v4)

   NOTE

   Connection parameter Interface ID

   Note that you can enter the value "**0**" for the interface ID with instruction version V5.0 or higher of the instruction TMAIL_C in the data type TMAIL_V4_SEC. In this case, the CPU itself searches for a suitable local CPU interface.

   Interconnect the variable to the MAIL_ADDR_PARAM parameter of the TMAIL_C instruction.
5. Download the hardware configuration and user program to the CPU.

Additional information

The STEP 7 online help describes:

- The system data types
- The instructions for open communication
- The connection parameters

7.9 Setting up communication via FTP

Setting up a connection for FTP via the user program

For communication via FTP, you need to create the data block of the relevant system data type yourself, assign parameters and call the instruction directly. This procedure is introduced below.

FTP client and server functionality

Files can be sent by a CPU to an FTP server and can be received from the FTP server. Communication with FTP is only possible for the S7-1500 using the CP 1543-1. The CP can be an FTP server, FTP client or both. FTP clients can also be third-party systems/PCs.

For the FTP server functionality, configure the CP accordingly in STEP 7.

You can use the FTP client functionality to implement, for example, the establishment and termination of an FTP connection, the transfer and deletion of files on the server. For the FTP client functionality, use the FTP_CMD instruction.

Procedure for setting up FTP server functionality

Requirement: The FTP server can be reached via the IPv4 network.

1. Configure an S7-1500 automation system with CPU and CP 1543-1 in the device view of the Devices & networks editor of STEP 7.
   At the same time, you need to select the option "Permit access with PUT/GET communication from remote partner (PLC, HMI, OPC, ...)" in the HW configuration of the S7-1500 CPU under the "Protection" area navigation in the section "Connection mechanisms".
2. Make the following settings in the properties of the CP under "FTP configuration":
   – Select the "Use FTP server for S7 CPU data" check box.
   – Assign the CPU, a data block and a file name under which the DB for FTP will be stored.

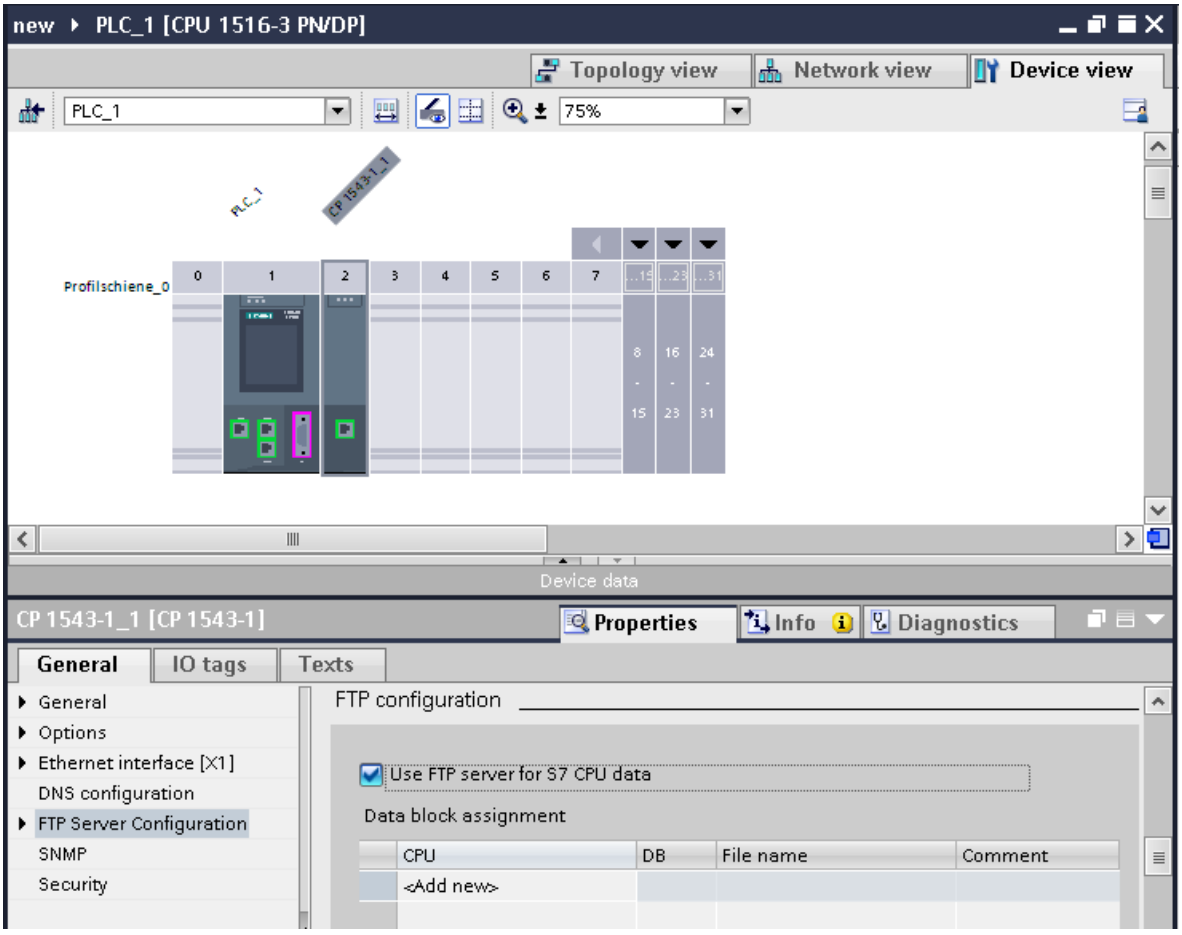


Figure 7-11 Setting up the FTP configuration

3. Download the hardware configuration to the CPU.

Procedure for setting up FTP client functionality

Requirement: The FTP server can be reached via the IPv4 network**.**

1. Configure an S7-1500 automation system with CPU and CP 1543-1 in the device view of the Devices & networks editor of STEP 7.
   At the same time, you need to select the check box "Permit access with PUT/GET communication from remote partner (PLC, HMI, OPC, ...)" in the HW configuration of the S7-1500 CPU under the "Protection" area navigation in the section "Connection mechanisms".
2. Call the FTP_CMD instruction in the user program of the CPU.
3. Set the connection parameters for the FTP server in the FTP_CMD instruction.
4. Create a global DB and within this DB a tag of the type FTP_CONNECT_IPV4, FTP_CONNECT_IPV6 or FTP_CONNECT_NAME.
5. Interconnect the tag within the data block with the FTP_CMD instruction.
6. For the connection to the FTP server, specify the following in the DB:
   – The user name, the password and the IP address for the FTP access in the relevant data type (FTP_CONNECT_IPV4, FTP_CONNECT_IPV6 or FTP_CONNECT_NAME)
7. Download the hardware configuration and user program to the CPU.

Application examples

- Application example: FTP communication with S7-1500 and CP 1543-1
  You can find the application example on the Internet (https://support.industry.siemens.com/cs/ww/en/view/103550797).
- Application example: FTP client communication with S7-1200/1500
  You can find the application example on the Internet (https://support.industry.siemens.com/cs/ww/en/view/81367009).

Additional information

The STEP 7 online help describes:

- The system data types
- The instructions for open communication
- The connection parameters

7.10 Establishment and termination of communications relations

Establishment and termination of communications

The table below shows the establishment and termination of communications as part of open communication.

Table 7-5 Establishment and termination of communications

Setting up the connection	Establishing communication	Terminating communication
With the user program	After downloading the user program to the CPUs: The passive communications partner sets up the local connection access by calling TSEND_C/TRCV_C or TCON. Calling TSEND_C/TRCV_C or TCON on the active partner starts connection establishment. If the connection could be established, there is positive feedback to the instructions in the user program. After you have terminated a connection using the instruction T_RESET, the connection is reestablished. If the connection aborts, the active partner attempts to re-establish the connection. This applies only if the connection was successfully established beforehand with TCON.	• Using the TSEND_C/TRCV_C, TDISCON and T_RESET instructions • When the CPU changes from RUN to STOP mode • With POWER OFF/POWER ON on a CPU
By configuring a connection	After downloading the connection configuration and the user program to the CPUs.	By deleting the connection configuration in STEP 7 and downloading the changed configuration to the CPU.

S7 communication

8

Characteristics of S7 communication

S7 communication as homogeneous SIMATIC communication is characterized by vendor-specific communication between SIMATIC CPUs (not an open standard). S7 communication is used for migration and for connecting to existing systems (S7-300, S7-400).

For data transfer between two S7-1500 automation systems, we recommend that you use open communication (see section Open User Communication (Page 115)).

Properties of S7 communication

Using S7 communication, the CPU exchanges data with another CPU. Once the user has received the data at the receiver end, the reception data is automatically acknowledged to the sending CPU.

The data is exchanged via configured S7 connections. S7 connections can be configured at one end or at both ends.

S7 communication is possible via:

- Integrated PROFINET or PROFIBUS DP interface of a CPU
- Interface of a CP/CM

S7 connections configured at one end

For an S7 connection configured at one end, the configuration for this connection takes place in only one communication partner and is only downloaded to it.

A one-sided S7 connection can be configured to a CPU that is only a server of an S7 connection (e.g. CPU 315-2 DP). The CPU is configured and the address parameters and interfaces are thus known.

In addition, a one-sided S7 connection can be configured to a partner who is not in the project and whose address parameters and interface and therefore are not known. You need to enter the address; it is not checked by STEP 7. The partner is initially unspecified (no partner address is registered when you create the S7 connection). Once you enter the address, it is "unknown" (i.e. it is named, but the project is unknown).

This makes it possible to use S7 connections beyond the boundaries of a project. The communication partner is unknown to the local project (unspecified) and is configured in another STEP 7 or third-party project.

S7 connections configured at both ends

When an S7 connection is configured at both ends, the configuration and download of the configured S7 connection parameters takes place in both communication partners.

Instructions for S7 communication

For S7 communication with S7-1500, the following instructions can be used:

- PUT/GET
  You write data to a remote CPU with the PUT instruction. You can use the GET instruction to read data from a remote CPU. The PUT and GET instructions and are one-sided instructions, i.e. you need only an instruction in one communication partner. You can can easily set up the PUT and GET instructions via the connection configuration.

  NOTE

  Data blocks for PUT/GET instructions

  When using the PUT/GET instructions, you can only use data blocks with absolute addressing. Symbolic addressing of data blocks is not possible.

  You must also enable this service for protection in the CPU configuration in the "Protection" area.

  This FAQ (https://support.industry.siemens.com/cs/ww/en/view/82212115) provides information about how to configure and program an S7 instruction and the GET and PUT communication instructions for data exchange between two S7-1500 CPUs.
- BSEND/BRCV
  The BSEND instruction sends data to a remote partner instruction of the type BRCV. The BRCV instruction receives data from a remote partner instruction of the type BSEND. You use the S7 communication via the BSEND/BRCV instruction pair for secure transmission of data.
- USEND/URCV
  The USEND instruction sends data to a remote partner instruction of the type URCV. The URCV instruction receives data from a remote partner instruction of the type USEND. You use the S7 communication via the USEND/URCV instruction pair for fast, non-secure transmission of data regardless of the timing of the processing by the communications partner; for example for operating and maintenance messages.

S7 communication via PROFIBUS DP interface in slave mode

You can find the "Test, commissioning, routing" check box in STEP 7 in the properties of the PROFIBUS DP interface of communications modules (e.g. CM 1542-5). Using this check box, you decide whether the PROFIBUS DP interface of the DP slave is an active or passive device on PROFIBUS.

- Check box selected: The slave is an active device on PROFIBUS.
- Check box cleared: The DP slave is a passive device on PROFIBUS. You can only set up S7 connections configured at one end for this DP slave.

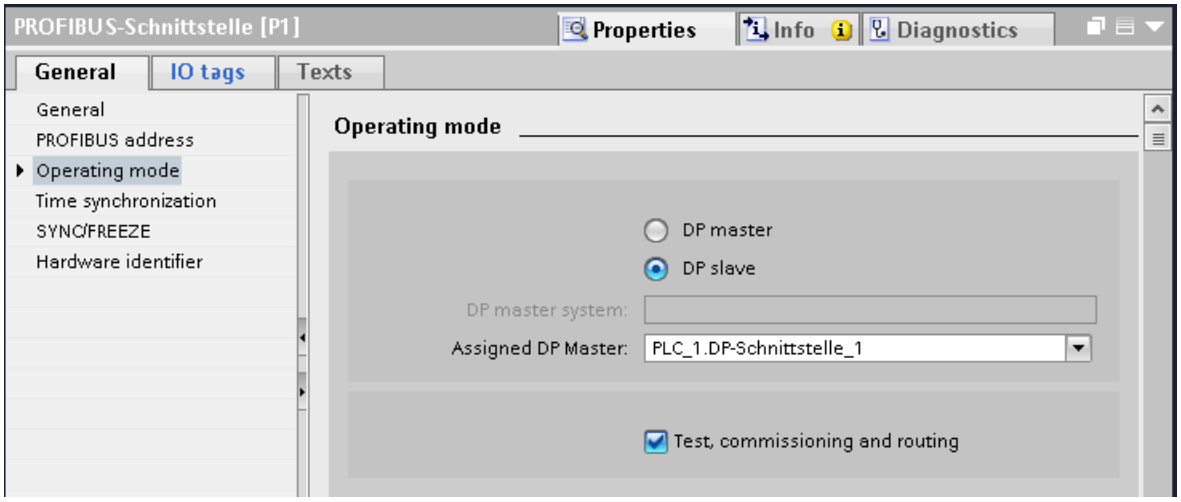


Figure 8-1 "Test, commissioning, routing" check box

Configuring S7 connections for PUT/GET instructions

You can create S7 connections and assign the parameters for these in the connection parameter assignment of the PUT/GET instructions. Changed values are checked immediately by the connection parameter assignment for input errors.

Requirement: A PUT or GET instruction is created in the programming editor.

To configure an S7 connection using PUT/GET instructions, follow these steps:

1. In the program editor, select the call of the PUT or GET instruction.
2. Open the "Properties > Configuration" tab in the inspector window.
3. Select the "Connection parameters" group. Until you select a connection partner, only the empty drop-down list for the partner end point is enabled. All other input options are disabled.

   The connection parameters already known are displayed:

   – Name of the local end point
   – Interface of the local end point

– IPv4 address of the local end point

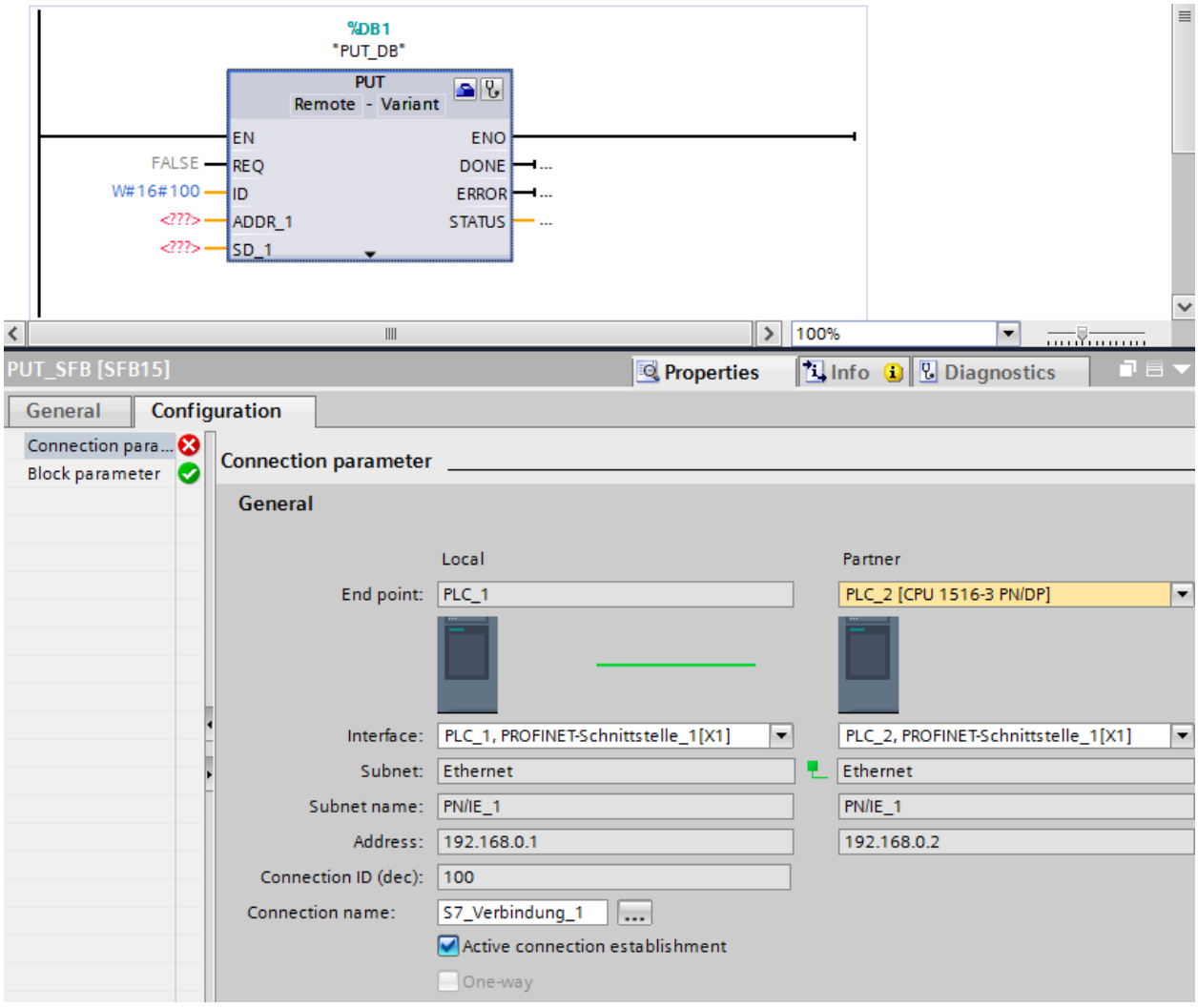


Figure 8-2 Connection configuration for PUT instruction

4. In the drop-down list box of the partner end point, select a connection partner. You can select an unspecified device or a CPU in the project as the communication partner.
   The following parameters are automatically entered as soon as you have selected the connection partner:
   – Name of the partner end point
   – Interface of the partner end point. If several interfaces are available, you can change the interface as required.
   – Interface type of the partner end point
   – Subnet name of both end points
   – IPv4 address of the partner end point
   – Name of the connection which is used for the communication.

5. If required, change the connection name in the "Connection name" input box. If you want to create a new connection or edit an existing connection, click on the "Select connection" button on the right side next to the input box for the connection name.

NOTE

The PUT and GET instructions between two communication partners can only run if both the hardware configuration and the program part for the partner end point have been loaded into the hardware. To achieve fully functional communication, make sure that you load not only the connection description of the local CPU on the device but also that of the partner CPU as well.

Configuring S7 connections for e.g. BSEND/BRCV

If you want to use the instructions for BSEND/BRCV for S7 communication, for example, you first need to configure an S7 connection.

To configure a S7 connection, follow these steps:

1. Configure the communications partners in the network view of the Devices & networks editor of STEP 7.
2. Select the "Connections" button and the "S7 connection" entry from the drop-down list.
3. Using drag-and-drop, connect the communication partner with each other (via an interface or local end point). If the required S7 subnet does not yet exist, it is created automatically.

   You can also set up a connection to unspecified partners.
4. In the "Connections" tab, select the row of the S7 connection.
5. Set the properties of the S7 connection in the "Properties" tab in the "General" area, for example the name of the connection and the interfaces of the communications partner that will be used.

   For S7 connections to an unspecified partner, set the address of the partner.
   You can find the local ID (reference of the S7 connection in the user program) in the "Local ID" area.
6. In the Project tree, select the "Program blocks" folder for one of the CPUs and open OB1 in the folder by double-clicking on it. The program editor opens.
7. In the program editor, call the relevant instructions for S7 communication in the user program of the communication partner (configured at one end) or in the user programs of the communication partners (configured at both ends). Select the BSEND and BRCV instructions from the "Communication" area of the "Instructions" task card, for example, and drag them to a network of OB1.
8. At the ID parameter of the instruction, assign the local ID of the configured connection to be used for the transmission of data.
9. Assign the parameters for the instructions indicating which data will be written to where and which data will be read from where.
10. Download the hardware configuration and user program to the CPU(s).

S7 communication via CP 1543-1

If you set up S7 communication via the Industrial Ethernet interface of the CP 1543-1, you can select the transport protocol for data transfer in the properties of the S7 connection under "General":

- "TCP/IP" check box selected (default): ISO-on-TCP (RFC 1006): for S7 communication between S7-1500 CPUs
- "TCP/IP" check box cleared: ISO protocol (ISO/IEC 8073): Addressing using MAC addresses

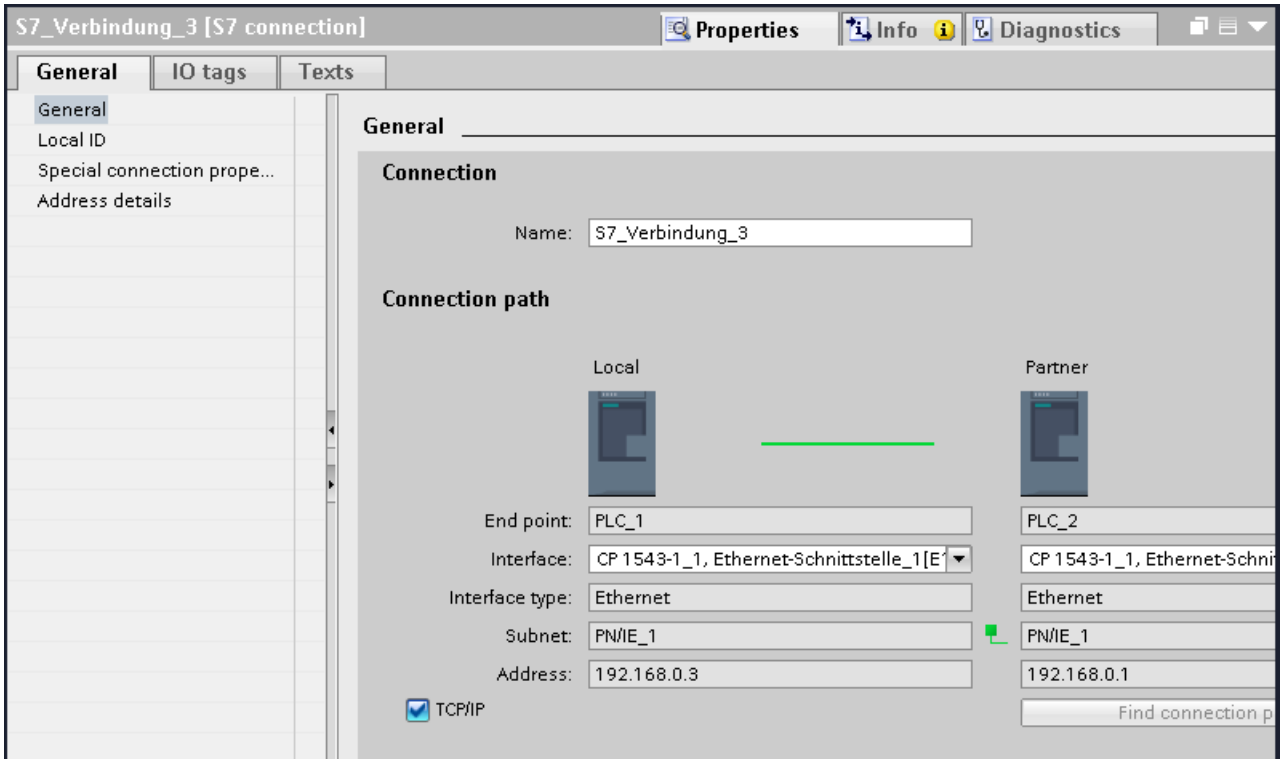


Figure 8-3 Selecting the CP 1543-1 transport protocol

Procedure for setting up an S7 connection via different S7 subnets

You have the option of using an S7 connection over multiple S7 subnets (PROFIBUS, PROFINET/Industrial Ethernet) (S7 routing (Page 352)).

1. Configure the communications partners in the network view of the Devices & networks editor of STEP 7.
2. Select the "Network" button.
3. Connect the relevant interfaces with the S7 subnets (PROFIBUS, PROFINET/Industrial Ethernet) using drag-and-drop.
4. Select the "Connections" button and the "S7 connection" entry from the drop-down list.

5. Using drag-and-drop in our example, connect PLC_1 in the left S7 subnet (PROFIBUS) to PLC_3 in the right S7 subnet (PROFINET).
   The S7 connection between CPU 1 and CPU 3 is configured.

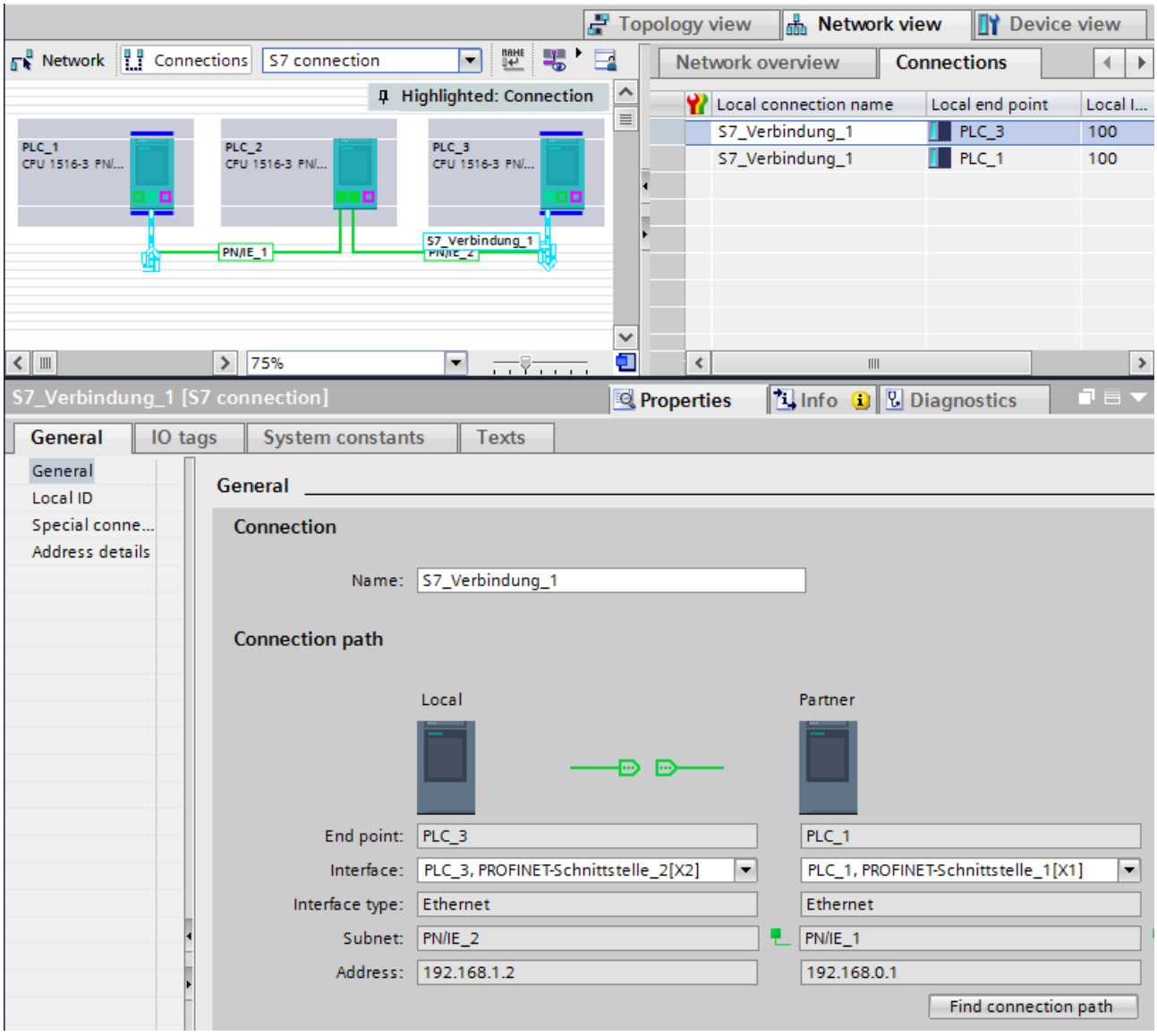


Figure 8-4 S7 connections via different subnets

ET 200SP Open Controller as router for S7 connections

If you assign the "PROFINET onboard [X2]" interface to the CPU 1515SP PC (F) of the SIMATIC PC station, the CPU 1515SP PC (F) can be used as a router for S7 connections. If you use the CP interface for "None, or a different Windows setting", you cannot use the Open Controller as a router for routed S7 connections.

An existing S7 connection routed by the CPU 1515SP PC (F) becomes invalid if the assignment of the interface of the CPU 1515SP PC (F) is changed from "SIMATIC PC station" to "None, or a different Windows setting". Since the PLC now no longer handles routing

functions for this connection, when the CPU 1515SP PC (F) is compiled, no message relating to the invalid connection is displayed. The invalid routed S7 connection is displayed only when the end points of the connection are compiled.

The interfaces required for routed S7 connections must remain explicitly assigned on the CPU 1515SP PC (F) . You can edit the assignment of the interface of the CPU 1515SP PC (F) in the properties under "PROFINET onboard [X2] > Interface assignment".

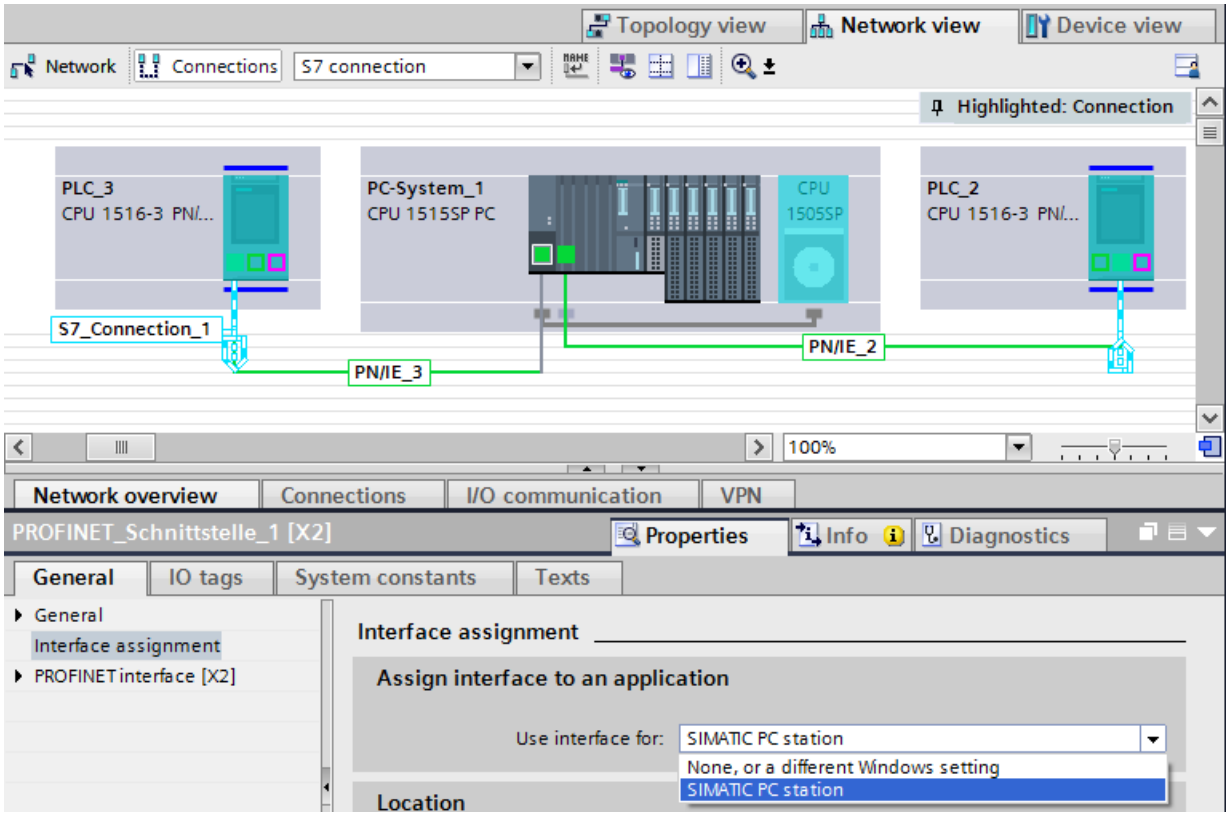


Figure 8-5 S7 routing PC station

Additional information

You can find detailed information on configuring S7 connections and how to use the instructions for S7 communication in the user program in the STEP 7 online help.

Point-to-point link

9

Functionality

A point-to-point connection for S7-1500, ET 200MP and ET 200SP is established via communications modules (CMs) with serial interfaces (RS232, RS422 or RS485):

- S7-1500/ET 200MP:
  - CM PtP RS232 BA
  - CM PtP RS422/485 BA
  - CM PtP RS232 HF
  - CM PtP RS422/485 HF
- ET 200SP:
  - CM PtP

The bidirectional data exchange via a point-to-point connection works between communications modules or third-party systems or devices capable of communication. At least 2 communication partners are required for communication ("point-to-point"). With RS422 and RS485, more than two communications partners are possible.

Protocols for communication via a point-to-point connection

- Freeport protocol (also called ASCII protocol)
- Procedure 3964(R)
- Modbus protocol in RTU format (RTU: Remote Terminal Unit)
- USS protocol (universal serial interface protocol)

The protocols use different layers according to the ISO/OSI reference model:

- Freeport: Uses layer 1 (physical layer)
- 3964 (R), USS and Modbus: Use layer 1 and 2 (physical layer and data link layer; therefore greater transmission reliability than with Freeport). USS and Modbus use additionally layer 4.

Properties of the Freeport protocol

- The recipient recognizes the end of the data transfer by means of a selectable end criterion (e.g. character delay time elapsed, receipt of end character, receipt of a fixed amount of data).
- The sender cannot recognize whether the sent data arrived free of errors at the recipient.

Properties of procedure 3964 (R)

- When the data is sent, control characters are added (start, end and block check characters). Make sure that these control characters are not included as data in the frame.
- Connection establishment and termination makes use of control characters.
- If transfer errors occur, data transfer is automatically repeated.

Data exchange using Freeport or 3964 (R) communication

The data to be sent is stored in the user program of the corresponding CPU in data blocks (send buffer). A receive buffer is available on the communications module for the received data. Check the properties of the receive buffer and adapt them if necessary. You must create a data block for receiving in the CPU.

In the user program of the CPU, the "Send_P2P" and "Receive_P2P" instructions handle the data transfer between the CPU and CM.

Procedure for setting up Freeport or 3964 (R) communication

1. Configure an S7-1500 configuration with CPU and CM in the device view of the hardware and network editor of STEP 7.
2. Select the interface of the CM in the device view of STEP 7.
3. Assign the parameters of the interface (for example connection communication, configuration of message sending) in the Inspector window of STEP 7 under "Properties" > "General".
4. Select the "Send_P2P" or "Receive_P2P" instruction in the "Instructions" task card under "Communication" > "Communications processor" and drag-and-drop the instruction into the user program (for example into a FB).
5. Assign the parameters for the instructions according to your configuration.
6. Download the hardware configuration and user program to the CPU.

Otherwise: Dynamic parameter assignment of the communications module

In certain types of application it is an advantage to set up communication dynamically; in other words, program-controlled by a specific application.

Typical applications for this, could be, for example manufacturers of serial machines. To make the user interfaces as convenient as possible for their customers, these manufacturers adapt the communications services to the particular operator entries.

Instructions for Freeport communication

There are 3 instructions available for the dynamic configuration in the user program for Freeport communication. The following applies to all 3 instructions: the previously valid configuration data is overwritten but not stored permanently in the target system.

- The "Port_Config" instruction is used for the program-controlled configuration of the relevant port of the communications module.
- The "Send_Config" instruction is used for the dynamic configuration, for example of time intervals and breaks in transmission (serial transmission parameters) for the relevant port.
- The "Receive_Config" instruction is used for dynamic configuration, for example of conditions for the start and end of a message to be transferred (serial receive parameters) for the relevant port.

Instructions for 3964 (R) communication

There are 2 instructions available for dynamic configuration in the user program for 3964 (R) communication. The following applies to the instructions: the previously valid configuration data is overwritten but not stored permanently in the target system.

- The "Port_Config" instruction is used for the program-controlled configuration of the relevant port of the communications module.
- The "P3964_Config" instruction is used for the dynamic configuration of protocol parameters.

Properties of the USS protocol

- Simple, serial data transfer protocol with cyclic message frame traffic in half duplex mode that is tailored to the requirements of drive technology.
- Data transfer works according to the master-slave principle.
  - The master has access to the functions of the drive and can, among other things, control the drive, read status values and read and write the drive parameters.

Data exchange using USS communication

The communications module is the master. The master continuously sends frames (job frames) to the up to 16 drives and expects a response frame from each addressed drive.

A drive sends a response frame under the following conditions:

- When a frame is received without errors
- When the drive is addressed in this frame

A drive must not send if these conditions are not met or the drive was addressed in the broadcast.

The connection to the relevant drives exists for the master once it receives a response frame from the drive after a specified processing time (response delay time).

Procedure for setting up USS communication

1. Configure an S7-1500 configuration with CPU and CM in the device view of the hardware and network editor of STEP 7.
2. In the Project tree, select the "Program blocks" folder and open OB1 in the folder by double-clicking on it. The program editor opens.
3. Select the instructions for USS communication according to your task in the "Communication" area, "Communications processor" folder of the "Instructions" task card and drag them to a network of OB1:
   - The "USS_Port_Scan" instruction allows you to communicate via the USS network.
   - The "USS_Drive_Control" instruction prepares send data for the drive and evaluates the response data of the drive.
   - The "USS_Read_Param" instruction is used to read out parameters from the drive.
   - The "USS_Write_Param" instruction is used to change parameters on the drive.
4. Assign the parameters for the instructions according to your configuration.
5. Download the hardware configuration and user program to the CPU.

Properties of the Modbus protocol (RTU)

- Communication takes the form of serial, asynchronous transfer with a transmission speed of up to 115.2 kbps, half duplex.
- Data transfer works according to the master-slave principle.
- The Modbus master can send jobs for reading and writing operands to the Modbus slave:
  - Reading inputs, timers, counters, outputs, memory bits, data blocks
  - Writing outputs, memory bits, data blocks
- Broadcast to all slaves is possible.

Data exchange using Modbus communication (RTU)

The communications module can be a Modbus master or Modbus slave. A Modbus master can communicate with one or more Modbus slaves (the number depends on the physical interface). Only the Modbus slave explicitly addressed by the Modbus master is permitted to return data to the Modbus master. The slave detects the end of the data transfer and acknowledges it. If an error occurs, it provides an error code to the master.

Procedure for setting up Modbus communication (RTU)

1. Configure an S7-1500 configuration with CPU and CM in the device view of the hardware and network editor of STEP 7.
2. In the Project tree, select the "Program blocks" folder and open OB1 in the folder by double-clicking on it. The program editor opens.

3. Select the instructions for Modbus communication according to your task in the "Communication" area, "Communications processor" folder of the "Instructions" task card and drag them to a network of OB1:
   – The "Modbus_Comm_Load" instruction configures the port of the CM for Modbus communication.
   – The "Modbus_Master" instruction is used for Modbus master functionality.
   – The "Modbus_Slave" instruction is used for Modbus slave functionality.
4. Assign the parameters for the instructions according to your configuration.
5. Download the hardware configuration and user program to the CPU.

Additional information

- You can find more detailed information on communication via point-to-point connections and basics of serial data transmission in the function manual CM PtP communication module - Configurations for point-to-point connections (https://support.industry.siemens.com/cs/us/en/view/59057093).
- You can find a description of how to use the instructions for point-to-point connections in the user program in the STEP 7 online help.
- You can find information about the communications modules with a serial interface in the manual of the particular communications module.

OPC UA communication 10

10.1 What you need to know about OPC UA

10.1.1 OPC UA and Industrie 4.0

Uniform standard for information and data exchange

Industry 4.0 stands for the intensive utilization, evaluation and analysis of the large volumes of data from production in IT systems at the enterprise level. With Industry 4.0, data exchange between the production and enterprise levels is rapidly increasing. However, a prerequisite for success is a uniform standard for the information and data exchange.

Classic OPC only runs on Windows operating systems. To get around this restriction, the OPC Foundation developed the OPC UA (OPC Unified Architecture) standard.

The OPC UA standard is particularly suitable for data exchange across different levels thanks to its independence from specific operating systems, its secure transfer procedures and the semantic description of data. Machine data (controlled variables, measured values or parameters) can also be transferred in this way.

An important component of this concept is that OPC UA communication can take place in parallel with real-time communication for time-critical, machine-level data transfer.

OPC UA is highly scalable so that a consistent information exchange between sensors, controllers and MES or ERP systems is possible.

OPC UA makes available not only data but also information about the data (data types), at the same time making possible machine-interpretable access to the data.

OPC UA topic page

For an overview of the most important articles and links on the subject of OPC UA, refer to the SIEMENS Industry Online Support.
OPC UA topic page (https://support.industry.siemens.com/cs/ww/en/view/109770435)

10.1.2 General features of OPC UA

OPC UA and PROFINET

OPC UA and PROFINET can be used together. The two protocols use the same network infrastructure.

Independence from the operating system

The OPC UA standard is platform-independent and uses an optimized TCP-based binary protocol for high-performance applications.
OPC UA can be used, for example, under Windows, Linux, Mac OS X, a realtime operating system or a mobile operating system (Android or iOS).

Independence of a specific transport layer

OPC UA currently supports the following transport mechanisms and protocols:

- The transfer of messages as a binary stream directly via TCP/IP
- The transfer of messages with XML via TCP/IP and HTTP. This transport mechanism allows only a slow transfer and is therefore almost never used. S7-1500 CPUs do not support this transport mechanism.

Binary data exchange is supported by all OPC UA applications (required in OPC UA specification).

Simple client-server principle

An OPC UA server provides a great deal of information within a network, e.g. relating to the CPU, the OPC UA server itself, the data and the data types. An OPC UA client accesses this information.

Implementation in different programming languages

The OPC Foundation has implemented the OPC UA standard in several programming languages: Stacks for .NET, ANSI C and Java are available, although maintenance has been discontinued for the stacks for ANSI C and Java.
The OPC Foundation offers the .NET stack as well as example programs as open source software. See Github (https://github.com/opcfoundation).
A number of companies offer Software Development Kits (SDK). These development packages contain the stacks of the OPC Foundation and other functionalities that facilitate the development of solutions.
Advantages of using SDKs:

- Support from the supplier
- Tested software
- Detailed documentation
- Clear license conditions (important for selling of solutions)

Scalability

OPC UA can be used for devices of different performance classes:

- Sensors
- Embedded systems
- Controllers
- PC systems
- Smartphones
- Servers running MES or ERP applications.

The performance class of the devices is differentiated by profiles. Different OPC UA profiles offer the possibility to scale OPC UA for very small and simple devices as well as for very high-performance devices.

An OPC UA profile describes functions and services that must be supported by the server and client. In addition, other functionalities/services that are not required by the profile can be optionally provided.

OPC UA profiles differ from PROFINET profiles; the latter define additional cross-vendor properties and behavior for installations and systems in the sense of a vendor-neutral software interface.

Nano Embedded Device 2017 Server Profile

For the smallest devices with severely limited functionality, there is the "Nano Embedded Device 2017 Server Profile" of the OPC Foundation. This profile is functionally equivalent to the core server facet and defines the OPC UA TCP binary protocol as the required transport profile. The profile allows for connections without UA Security and does not allow subscriptions or method calls. Support for diagnostic objects and variables is optional for this profile.

Additional profiles build on the "Nano Embedded Device 2017 Server Profile", requiring more resources and offering more functionality.

Micro Embedded Device 2017 Server Profile

This profile provides limited functionality; it requires at least two parallel connections. Additionally, it allows subscriptions/data monitoring, but no UA Security and no method calls.

- S7-1200 Basic Controllers support the "Micro Embedded Device 2017 Server Profile". The S7-1200 additionally supports UA Security.

Embedded 2017 UA Server Profile

This profile has been developed for devices with more than 50 MB RAM and a more high-performance processor. It is based on the Micro Embedded Device Server profile. In addition, it requires UA security and method calls.

In addition, the servers must make their used type model (data types, reference types, variable types, etc.) available.

- S7-1500 Advanced Controllers support the "Embedded 2017 UA Server Profile".

Standard and global discovery profiles

The "OPC UA Specification Part 7" defines additional profiles:

- The "Standard 2017 UA Server Profile", which is suitable for PC-based OPC UA servers
- 2 global profiles, "Global Discovery Server 2017 Profile" and "Global Discovery and Certificate Management 2017 Server Profile", that cover the required service and information models of a Global Discovery Server

Type-instance concept

OPC UA offers a fully networked (full-meshed network), object-oriented information model for namespaces, including metadata for the object description. Any object structures can be generated via referencing of the instances among each other and their types. Because servers disclose their instance and type systems, clients can navigate through this network and obtain all the information they need. Both instances and their type definitions are available in runtime.

Procedures or concepts on how to handle references to types are optimized over time. These optimizations lead to new versions of the OPC UA Specification (e.g. V1.03 => V1.04).

PLC tag mapping

The information of the OPC UA server (for example the PLC tags) is modeled as nodes connected to one another via references. The semantics are displayed by the server in the address space and can be acquired by clients (while navigating). This makes it possible to browse from node to node with an OPC UA client and find out what content can be read, monitored or written.

Integrated security mechanisms

OPC UA uses security mechanisms at various levels:

- A secure connection can only be established between an OPC UA server and an OPC UA client if the client and server can register with X.509-v3 certificates and accept each other's certificates (security at the application level). Various security policies are possible, including an unsecured connection between server and client (Security Policy: "No security").
- A server can always request the following information from the user for authorized access (authentication):
  - A user certificate (not configurable in STEP 7)
  - User name and password
  - No user authorization

The security mechanisms are optional and configurable.

More information

You can find more information on the website of the OPC Foundation (https://opcfoundation.org).

10.1.3 OPC UA for S7-1200/S7-1500 CPUs

In OPC UA, one system operates as a server and provides data the existing information to other systems (clients).

OPC UA clients, for example, have read and write access to data on an OPC UA server. OPC UA clients call methods on the OPC UA server.

You can access this data online with a client, including e.g. information on performance and diagnostics. In OPC UA terminology, this function is called "Browsen". The "Subscription"

function eliminates regular reading of a tag; with this function, the server informs a client about value changes.
A system can be both a client and a server at the same time.

OPC UA server of the S7-1500 CPU

As of firmware version 2.0, an S7-1500 CPU is equipped with an OPC UA server.
The following sections describe how you configure the OPC UA server of the S7-1500 CPU to make data and methods available for OPC UA clients so that clients have read or write access to PLC tags on the CPU and can call server methods.
The following sections also set out how to integrate companion specifications into the address space of the OPC UA server.

OPC UA server of the S7-1200 CPU

As of firmware V4.4, an S7-1200 CPU is equipped with an OPC UA server.
The OPC UA server is generally configured as it is for an S7-1500 CPU; the scope of functions and the quantity limits are limited according to the supported "Micro Embedded Device 2017 Server Profile". Unlike for an S7-1500 CPU, the functions "Registered Read" and "Registered Write" are **not** available.
As of firmware V4.5, S7-1200 CPUs support server methods as well as structured data types (structures and arrays).
You can find additional information in the STEP 7 online help.

OPC UA client of the S7-1500 CPU

As of firmware version V2.6, an S7-1500 CPU is additionally equipped with an OPC UA client.
The following sections show how to use standardized instructions (PLCopen function blocks) to create a user program that, as an OPC UA client, provides the following functions:

- Reading data from an OPC UA server
- Writing data to an OPC UA server
- Calling methods of an OPC UA server

STEP 7 (TIA Portal) assists you in creating user programs by providing an editor for client interfaces and a parameter assignment for OPC UA connections.
The OPC UA instructions for an S7-1500 CPU as client are described in detail in the help to the instructions (Instructions > Communication > OPC UA).

OPC UA client for test purposes

The following description uses various different OPC UA clients to illustrate the use of OPC UA clients:

- "UaExpert" of Unified Automation. An extensive client that can be used free of charge:
  Link for downloading UaExpert (https://www.unified-automation.com/downloads/opc-ua-clients.html)
- "UA Sample Client" of the OPC Foundation. This client is available free of charge for users who are registered with the OPC Foundation :
  Link for downloading the example client of the OPC Foundation
  (https://opcfoundation.org)

Application example in Industry online support

Siemens Industry Online Support provides a free application example with a client API for various applications. You use the functions of this interface to create your own OPC UA clients that match your application. To simplify handling the API, we offer a higher-level .NET helper class.

The client API is based on the .NET OPC UA stack of the OPC Foundation.

The application example shows how to establish connections between servers and clients, for example. It also demonstrates the reading and writing of PLC tags.

Link to download: OPC UA .NET client for the SIMATIC S7-1500 OPC UA Server (https://support.industry.siemens.com/cs/us/en/view/109737901)

10.1.4 Access to OPC UA applications

The access possibilities that an S7-1500 CPU with an OPC UA application (client or server) has via a CP in the same station are described below. In addition, an approach for combining these access possibilities with the "IP Forwarding" function to allow access to devices of another IP subnet via an S7-1500 station is presented.

All the settings for this can be found in the CPU properties, "Advanced configuration" area in the Inspector window.

The possibility of accessing the OPC UA application in the CPU via CP interface is subject to the following requirements:

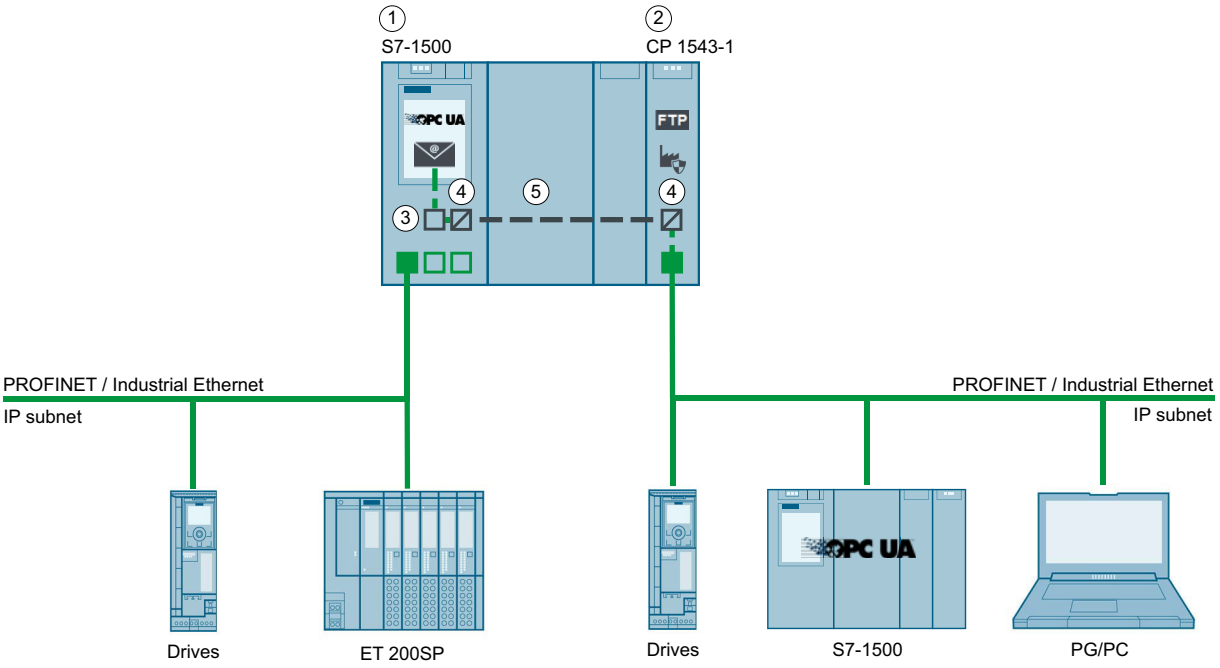
- S7-1500 CPU (except S7-1500 R/H) as of firmware version V2.8
- CP 1543-1 firmware version V2.2 or higher

Recommendation: Use a CP 1543-1 with firmware V3.0 or higher. As of this version, the security functions (firewall) are also available for the virtual interface and no additional firewall needs to be installed between the station and an insecure network.

Principle: Interface for access via communication module

For a CPU application, such as OPC UA, to be accessed via CP interface, you must configure a virtual interface (W1). IP-based applications can then be accessed via the IP address parameters of this virtual interface.

The schematic is shown in the following figure.



① CPU S7-1500 FW V2.8 or higher (e.g. CPU 1515-2 PN)

② CP 1543-1 (FW V2.2 or higher)

③ Virtual interface (W1)

④ Protocol conversion PROFINET / Industrial Ethernet on backplane bus or backplane bus on PROFINET / Industrial Ethernet

⑤ Backplane bus

Figure 10-1 Principle: Interface for access via communication module

Example: Access of OPC UA clients to the OPC UA server of the CPU

For access of an OPC UA client to the OPC UA server of the CPU, the following interfaces of the S7-1500 station are available:

- The local PROFINET interfaces of the S7-1500 CPU
- The Ethernet interface of a CP 1543-1 (firmware version V2.2 and higher)

The following figure shows an example of a possible configuration: The CPU could also have the role OPC UA client and the device on the subnet of the CP could have the role OPC UA server.

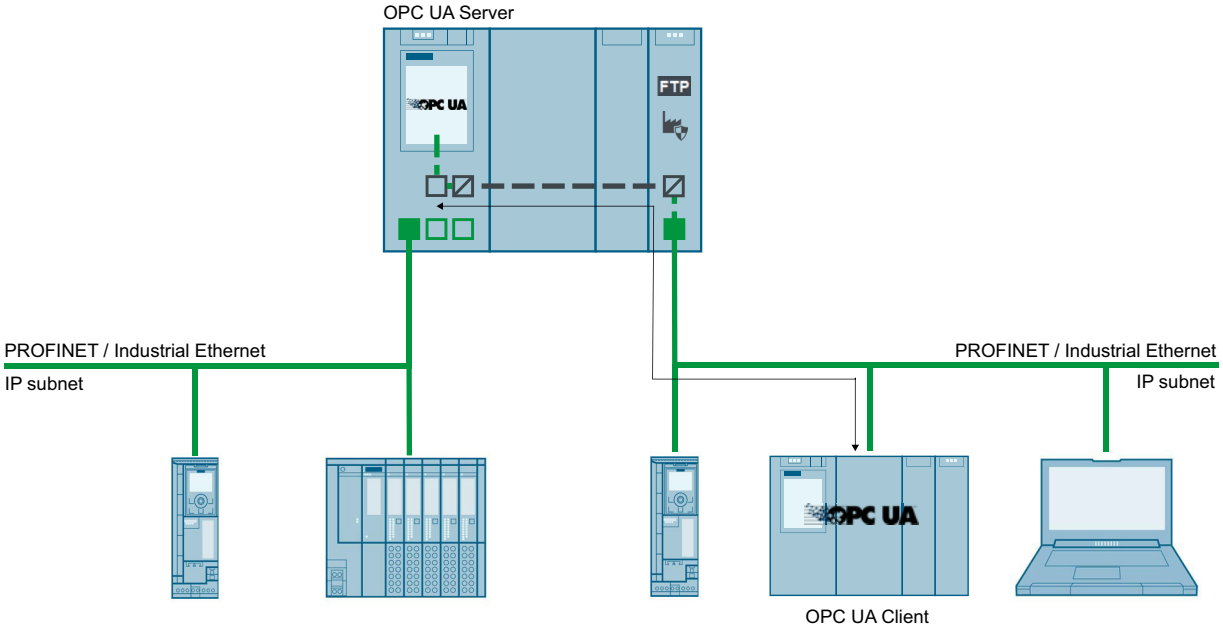


Figure 10-2 Example: Access of OPC UA clients to the OPC UA server of the CPU

Example: Access of OPC UA clients to OPC UA servers via S7-1500 CPU with activated IP Forwarding

OPC UA client and OPC UA server can also be connected to one another via an S7-1500 CPU, in which case the S7-1500 CPU operates as an IP Forwarder. This configuration option allows for flexible expansion of existing systems.

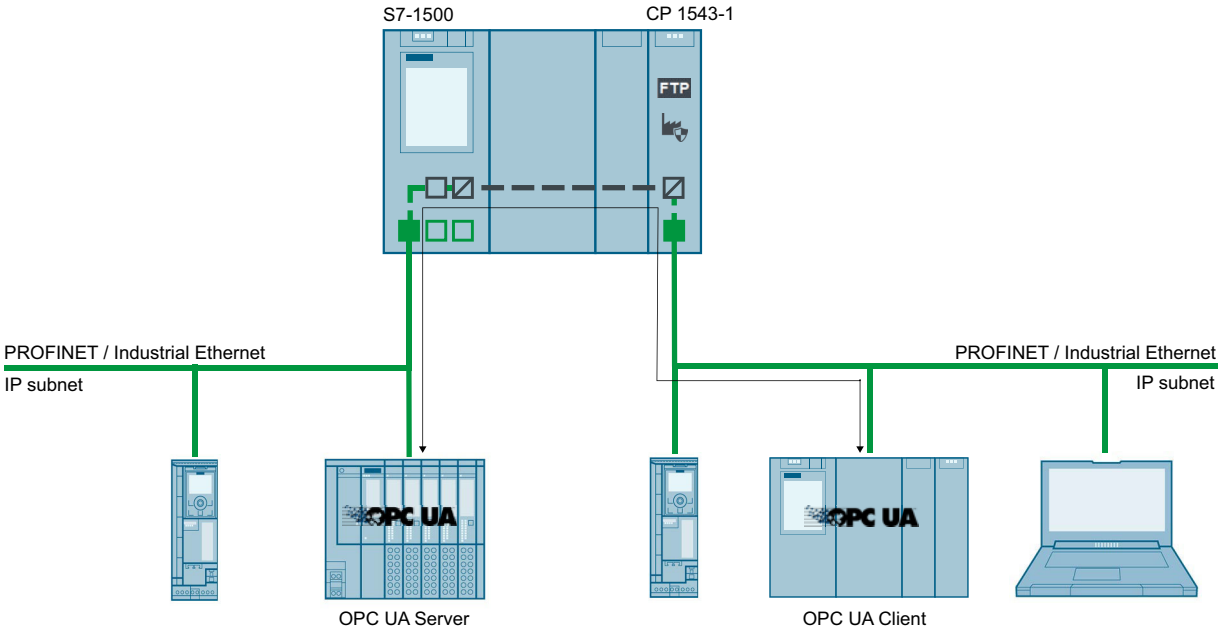


Figure 10-3 Example: Access of OPC UA clients to OPC UA servers via S7-1500 CPU with activated IP Forwarding

More information

More information on access options via the virtual interface and via IP forwarding can be found in the following sections:

- IP forwarding (Page 356)
- Virtual interface for IP-based applications (Page 365)

10.1.5 Addressing nodes

Nodes are the basic elements of OPC UA, they are comparable with objects from object-oriented programming. Nodes are used, for example, for user data (tags) or other metadata. Nodes are used to model an OPC UA address space that also contains a type model with type definitions.

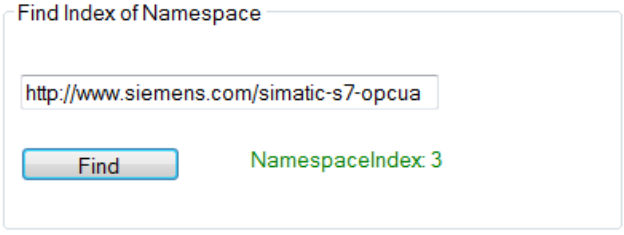
Node ID (NodeId)

Nodes in the OPC UA address space are uniquely identified by a NodeId (Node Identifier).

The NodeId consists of an identifier, identifier type and a namespace index. Namespaces are used to avoid naming conflicts.
The OPC Foundation has defined a wide range of nodes that provide information about the given OPC UA server. These nodes can be found in the namespace of the OPC Foundation and have the index 0.
The OPC Foundation also defines data types and tag types.

Namespace (Namespace)

In addition to the above-described namespace of the OPC Foundation, the namespace for accessing CPU data is of interest: All the tags or methods of an S7-1500 OPC UA server are contained in the namespace (Namespace) of the standard server interface "http://www.siemens.com/simatic-s7-opcua".
By default this namespace has the Index 3. The index may change later if additional namespaces are inserted into the server or if existing ones are deleted. It is therefore necessary for an OPC UA client to request the current index of the namespace (e.g. "http://www.siemens.com/simatic-s7-opcua") from the server before reading or writing its values.
The following figure shows an example of the result of such a request.

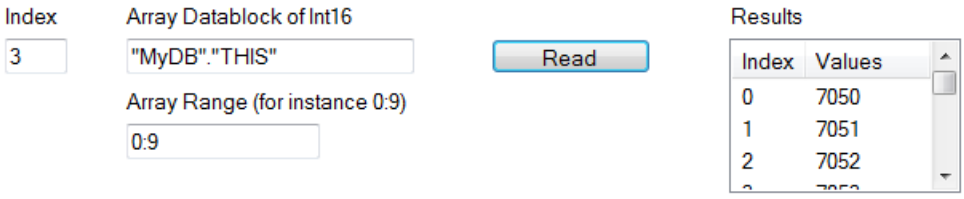


Identifier

The Identifier corresponds to the name of the PLC tag in quotation marks. The quotation mark is the only sign that is not permitted as part of a name in STEP 7. Quotation marks avoid naming conflicts.
The following example reads the value of the "StartTimer" tag:

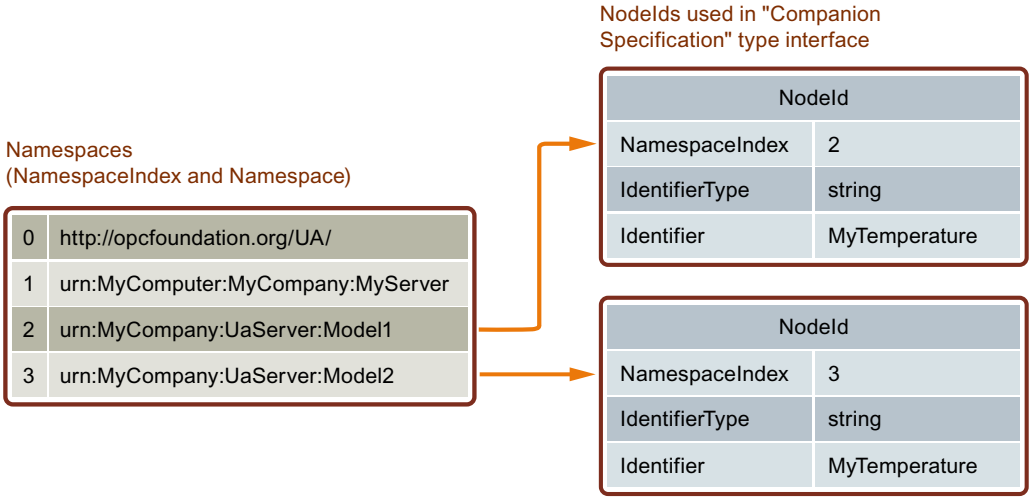
The Identifier can consist of several components. The individual components are then separated by a dot.

The following example reads the "MyDB" array data block completely. This data block contains an array with ten integer values. All ten values should be read in one pass. Therefore, "0:9" is entered at the array range.



Example of NodeIds, identifiers and namespaces

The following figure illustrates the relation between NodeIds, identifiers and namespaces: It is no problem if two nodes have the same identifiers but belong to different namespaces. STEP 7 (TIA Portal) allows you to easily import namespaces via a server interface.



PLC tags in the address space of the OPC UA server

The figure below shows where the PLC tags in the example are located in the address space of the OPC UA server (excerpt from UA client):

The "MyDB" data block is a global data block. The data block is therefore located below the node "DataBlocksGlobal". "StartTimer" is a memory tag and is therefore stored below the "Memory" node.

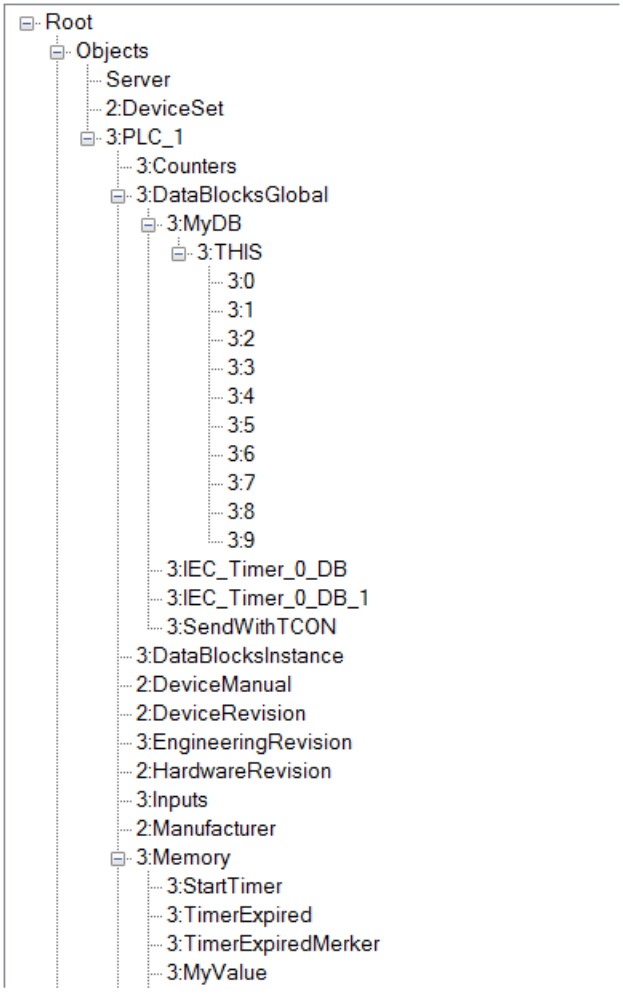


Figure 10-4 PLC tags in the address space of the OPC UA server

Methods in the address space of the OPC UA server

If you implement a method via your user program, it takes the following form in the address space of the OPC UA Server

(see Providing methods on the OPC UA server (Page 267)):

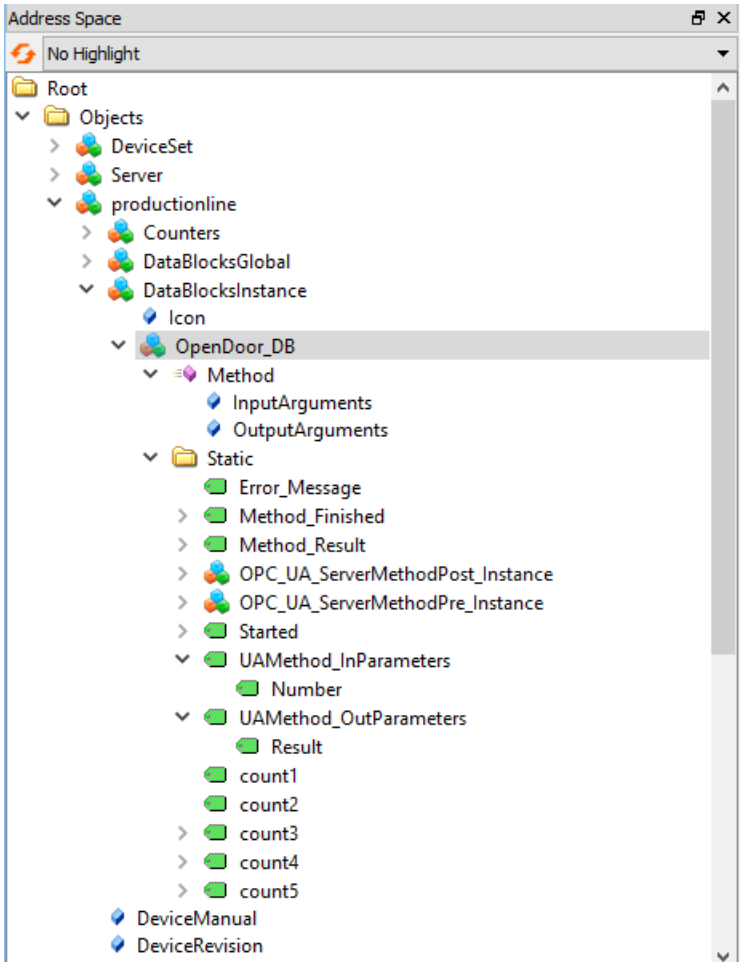


Figure 10-5 Methods in the address space of the OPC UA server

10.1.6 What you need to know about OPC UA clients

Basics of OPC UA clients

OPC UA clients are programs that do the following:

- Access the information from an OPC UA server (for example an S7-1500 CPU): read/browse access, write access, subscriptions
- Execute methods through the OPC UA server

However, OPC US clients can only access data that is enabled for this purpose (see "Managing write and read rights (Page 207)").

You need the endpoint of the server to establish a connection to an OPC UA server (see "Endpoints of the OPC UA servers (Page 198)").

Reading out information from the OPC UA server

When a connection to an end point of the server exists, you can use the navigation function of the client: You navigate starting from a defined starting point (from the "root" node) through the address space of the server.

The following information is provided in the process:

- Enabled PLC tags, data blocks and data block components
- Namespace index and identifiers of these PLC tags, data blocks and DB components
- Data types of the PLC tags and DB components
- Number of components in arrays (required for reading and writing arrays)

In addition, you receive information about the OPC UA server itself as well as information about the S7-1500 based on the "OPC UA for Devices" standard of the OPC Foundation (for example, serial number, firmware version).

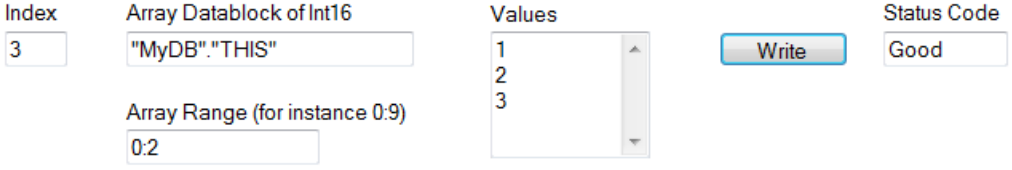
Reading data from the server and writing to the server

You now know the namespace, identifier and data type of PLC tags. This means that you can now specifically read individual PLC tags and DB components as well as complete arrays and structures.

You can find examples of the reading of Boolean tags and array data blocks in Addressing nodes (Page 159).

Rules for access to structures are available here (Page 322).

With the information that you obtain while navigating through the address space of the server (index, identifier and data type), you can also transfer values to the S7-1500 with the OPC UA client. The following example overwrites the first three values in the array data block "MyDB".



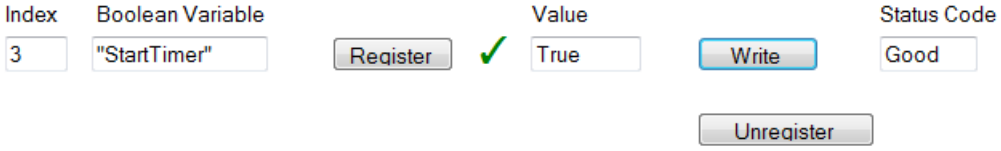
For "Array Range" you specify which components of the array you want to overwrite. The "Good" status code indicates that the values were transferred successfully. However, you can only write the values to the S7-1500 but not the time stamps of these values. The time stamps can only be read.

Faster access through registration

Registered Read/Write lends itself to repeated, optimized access to data – with maximum performance. When tag nodes are registered, the OPC UA server creates a numerical Identifier (numerical NodeId) that directly references the registered node. For read or write jobs of the client to this numerical Identifier, the server does not have to resolve any strings as Identifier and can access the requested tag in an optimized manner.

This Identifier applies solely to the current session and has to be queried again when the session connection is terminated/lost.

In the following example, the "StartTimer" tag is first registered on the server. Afterwards, the rapid function "RegisteredWrite" is used for setting the value.



In accordance with the same scheme, the "RegisteredRead" function can also be used, which is particularly useful for recurring data readouts. Take into account, however, that depending on the application it may be advisable to use a Subscription instead.

Recommendation: It is best to place registrations in the startup program of the OPC UA client, since the registration takes up time.

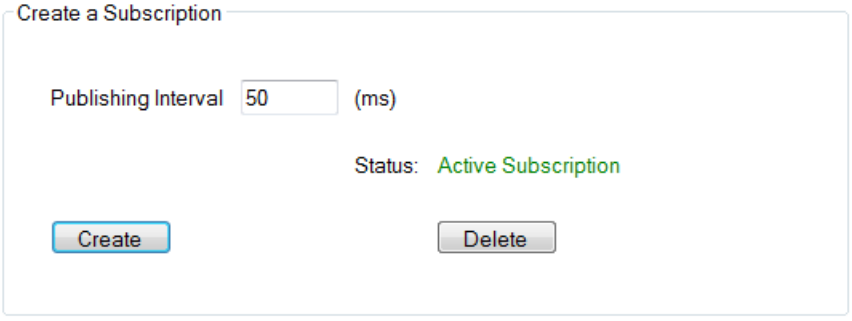
Please note that you can set the maximum number of registered nodes in the properties of the S7-1500 CPU and that the Clients have to respect this number, see General settings of the OPC UA server (Page 220).

Subscription

The term "Subscription" is used for a function in which only those tags for which an OPC UA client has registered at the OPC UA server are transferred. The OPC UA server only sends a message to the OPC UA client for these registered tags (monitored Items) when a value has changed. The monitoring of these tags makes constant sampling by the OPC UA client (Polling)superfluous, which reduces the network load.

You have to create a Subscription to use this function. For this purpose, you specify the "Publishing Interval" at the UA client and click the "Create" button. The publishing interval is the time interval in which the server sends new values to the client in a notification (data change notification).

In the following example a subscription has been created: The client receives a message with the new values (publishing interval 50 ms) every 50 milliseconds here.



Preventing server overload

You can set the OPC UA server of the S7-1500 CPU by means of the "Minimum publishing interval" so that it does not serve extremely short send intervals requested by the client, see Settings of the server for subscriptions (Page 222).

Example: A client wants to be operated at a publishing interval of 50 ms as detailed above. Such a short publishing interval would, however, result in a high network and server load. You should therefore set 1000 ms as the "Minimum publishing interval" for the server. Clients

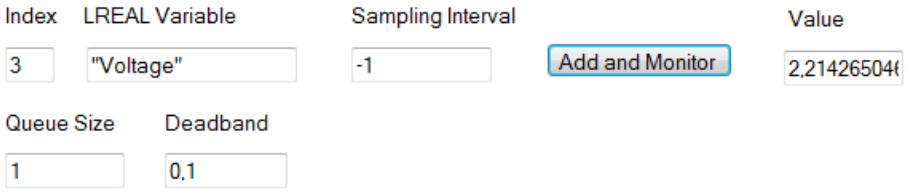
whose subscription requires shorter publishing intervals are "slowed" to 1000 ms and the server is protected from overload.

Sampling and transmission (Sampling & Publishing) within the scope of a subscription are communication processes which, like other communication processes (TCP/UDP/Web server communication...), are processed by the CPU with priority 15. OBs with higher priority interrupt the communication. If you set the sampling and transmission intervals too short, this setting causes a high communication load. Therefore, select intervals as large as possible, which are still sufficient for the application.

For information about the consistency of tags, refer to Consistency of CPU tags (Page 212).

Monitoring of PLC tags

When the Subscription has been created, you inform the server which tags are to be monitored with it. In the following example, the "Voltage" tag was added to the subscription.



The "Voltage" tag contains the value of a voltage that is detected by an S7-1500 CPU.

The sampling interval ("Sampling Interval") contains a negative value (-1). This determines that the default setting of the OPC UA server is used for the sampling interval. The default setting is defined by the transmission interval ("Publishing Interval") of the subscription. If you want to set the smallest possible sampling interval, select the value "0".

In this example, the length of the queue is set to "1": Only one value is read from the CPU at an interval of 50 milliseconds and subsequently sent to the OPC UA client when the value has changed.

The "Deadband" parameter in this example is "0.1": Changes in value have to amount to 0.1 Volt; only then does the sender send the new value to the client. The server does not send smaller changes in value. You can use this parameter, for example, to disable signal noise: Slight changes in a process variable which do not have a real meaning.

10.1.7 Mapping of data types

SIMATIC and OPC UA data types

SIMATIC data types do not always correspond with OPC UA data types.

S7-1500 CPUs provide SIMATIC tags (with SIMATIC data types) to their own OPC UA server as OPC UA data types. OPC UA clients can then access these tags with OPC UA data types via the server interface.

A client can read the attribute "DataType" from such a tag and reconstruct the original data type in SIMATIC.

Example

A tag has the SIMATIC data type "COUNTER". You read COUNTER → UInt16 in the table. You now know that you do not need to convert; the COUNTER value is sent over the line as a UInt16 data type.

The client detects from the attribute "DataType" that the tag is actually the SIMATIC data type "COUNTER". With this knowledge, the client reconstructs the data type.

Table 10-1 SIMATIC and OPC UA data types

SIMATIC data type	OPC UA data type
BOOL	Boolean
BYTE	BYTE → Byte
WORD	WORD → UInt16
DWORD	DWORD → UInt32
LWORD	LWORD → UInt64
SINT	SByte
INT	Int16
DINT	Int32
LINT	Int64
USINT	Byte
UINT	UInt16
UDINT	UInt32
ULINT	UInt64
REAL	Float
LREAL	Double
S5TIME	S5TIME → UInt16
TIME	TIME → Int32
LTIME	LTIME → Int64
DATE	DATE → UInt16
TIME_OF_DAY (TOD)	TOD → UInt32
LTIME_OF_DAY (LTOD)	LTOD → UInt64
DATE_AND_TIME (DT)	DT → Byte[8]
LDT	DateTime

SIMATIC data type	OPC UA data type
DTL Special note: You can only describe the structure completely with an OPC UA client. You have read-only access individual elements of this structure (e.g. "YEAR")	mapped as structure
CHAR	CHAR → Byte
WCHAR	WCHAR → UInt16
STRING (Code page 1252 or Windows-1252)	STRING → String
WSTRING (UCS-2; Universal Coded Character Set)	String
TIMER	TIMER → UInt16
COUNTER	COUNTER → UInt16

Arrays

A read or write job with OPC UA is always an array access, which means that it always has an index and length. A single tag is a special case of an array (index 0 and length 1). The data type is simply sent repeatedly on the line. For the tags, the "DataType" attribute indicates the basic data type. The attributes "ValueRank" and "ArrayDimensions" show whether or not you are dealing with an array and how large the array is.

Data types based on arrays

There are SIMATIC data types for which an OPC UA value is mapped to an array of bytes. An array of these data types is then mapped to a two-dimensional array.

Example: The SIMATIC data type DATE_AND_TIME (DT) is mapped on the OPC UA side to an 8-byte array (Byte[8]), see table above. When you define an array of the SIMATIC data type DATE_AND_TIME (DT), then it is considered as two-dimensional array.

This fact affects the use of system data types such as OPC_UA_NodeAdditionalInfo and OPC_UA_NodeAdditionalInfoExt, for example:

For the data types described above, you must use the system data type OPC_UA_NodeAdditionalInfoExt for multidimensional arrays instead of OPC_UA_NodeAdditionalInfo.

Structures

Structures are transferred as ExtensionObject. The S7-1500 server uses binary representation for transmission of the ExtensionObjects over the line; the individual structure elements come one after the other. At the front is the NodeId of the data type; this is used by the client to establish the structure.

For OPC UA Specification <= V1.03, a client has to read, decode and interpret the complete DataTypeDictionary for this (unless it has already learned this library offline through an XML import).
Starting in OPC UA V1.04, the DataTypeDescription attribute is also available for this, which can be read and interpreted more quickly and easily. A client only determines the setup of the structure once, before or during the first access, and then uses this information for the duration of the session.

Special SIMATIC data types

SIMATIC data types that are not in the table above and cannot be defined as elements of a structure or PLC data type are not supported by the OPC UA client.
These are, for example, "ANY" or "POINTER" pointers, function block "Block_FB", function "Block_FC" or hardware data type "REMOTE".
The selection of an unsupported data type leads to an error message.

Additional information

More details on mapping of basic data types, arrays and structures can be found in the OPC UA Specification Part 6, "Mappings" (see OPC UA BINARY there).
What must be considered with arrays and data types DTL and LDT in the OPC UA server of a SIMATIC S7-1500? FAQ (https://support.industry.siemens.com/cs/ww/en/view/109766726)

10.2 Security at OPC UA

10.2.1 Security settings

Addressing risks

OPC UA allows the exchange of data between different systems, both within the process and production levels and to systems at the control and enterprise level.
This possibility also entails security risks. That is why OPC UA provides a range of security mechanisms:

- Verification of the identity of OPC UA server and clients.
- Checking of the identity of the users.
- Signed/encrypted data exchange between OPC UA server and clients.

These security policies should only be bypassed in cases where it is absolutely necessary:

- During commissioning
- In stand-alone projects without external Ethernet connection

If you have selected the endpoint "None" for "UA Sample Client" of the OPC Foundation, for example, the program issues a clear warning:

When STEP 7 compiles your project it also checks whether you have considered the setting options for the protection and warns you of possible risks. This also includes an OPC UA security policy with the setting "no security", which corresponds to the end point "None".

NOTE

Disabling security policies you do not want

If you have enabled all security policies in the secure channel settings of the S7-1500 OPC UA server – thus, also the end point "None" (no security) – unsecured data traffic (neither signed nor encrypted) between the server and client is also possible. The OPC UA server of the S7-1500 CPU also sends its public certificate to the client at "None" (No security). And some clients check this certificate. However, the client is not forced to send a certificate to the server. The identity of the client may possibly remain unknown. Each OPC UA client can then connect to the server irrespective of any subsequent security settings.

When configuring the OPC UA server, make sure that only security policies that are compatible with the security concept of your machine or plant are selected. All other security policies should be disabled.

Recommendation: Use the setting "Basic256Sha256 - Sign and Encrypt", which means that the server only accepts Sha256 certificates. The security policies "Basic128Rsa15" and "Basic256" are deactivated by default and should not be used as an end point. Select end points with a higher security policy.

Additional security rules

- Only use the end point "None" in exceptional cases.
- Only use the "guest authentication" of the user in exceptional cases.
- Only allow access to PLC tags and DB components via OPC UA if it is genuinely necessary.
- Use the list of trusted clients in the settings of the S7-1500 OPC UA client to allow access to certain clients only.

10.2.2 Certificates pursuant to ITU X.509

Security mechanisms are integrated in several layers in OPC UA. Digital certificates have an important role here. An OPC UA client can only establish a secure connection to an OPC UA server when the server accepts the digital certificate of the client and classifies it as trusted.

See section Handling client and server certificates (Page 224).

The client must also check and trust the certificate of the server. The server and client must show their identities and prove that they are what they claim to be: They must prove their identity. Mutual authentication of client and server, for example, prevents man-in-the-middle attacks.

Man-in-the-middle attacks

A "man-in-the-middle" could have positioned itself between server and client. A man-in-the-middle is a program that intercepts communication between server and client and claims to be a client or server, and is thus able to obtain information about the S7 program or to set values in the CPU and attack a machine or plant.

OPC UA uses digital certificates that meet standard X.509 of the International Telecommunication Union (ITU).

This allows the identity of a program, a computer or an organization to be proven (authenticated).

X.509 certificates

An X.509 certificate includes the following information:

- Version number of the certificate
- Serial number of the certificate
- Information on the algorithm used by the certificate authority to sign the certificate.
- Name of the certificate authority
- Start and end of the validity period of the certificate
- Name of the program, person or organization for which/whom the certificate has been signed by the certificate authority.
- The public key of the program, person or organization.

An X509 certificate thus links an identity (name of a program, person or an organization) to the public key of the program, person or organization.

Check during connection establishment

When a connection is being established between the client and server, the devices check all information from the certificate that is required to determine its integrity, such as signature, period of validity, application name (URN) and, in case of firmware version V2.5 only, also the IP address of the client in the client certificate.

NOTE

The validity period stored in the certificate is also checked. The CPU clock must therefore be set and date/time must be within the validity period, otherwise no communication takes place.

Signing and encryption

To allow you to check whether a certificate has been manipulated, certificates are signed.

There are various possible procedures here:

- Within the TIA Portal you have the possibility to generate and sign certificates. If you have protected your project and are logged in as a user with the function right to make security settings, you can use the global security settings. The global security settings allow access to the certificate manager and therefore to the certificate authority (CA) of the TIA Portal.
- Additional options are available for creating and signing certificates. In the TIA Portal, you can import certificates into the global certificate manager.
  - You contact a certificate authority (CA) and have your certificate signed.
    In this case, the certificate authority checks your identity and signs your certificate with the private key of the certificate authority. For this purpose you send a CSR (Certificate Signing Request) to the certificate authority.
  - You yourself create a certificate and sign it.
    To this purpose you use, for example, the "Opc.Ua.CertificateGenerator" program of the OPC Foundation. Alternatively, you use OpenSSL.
    You can find more information in Generating PKI key pairs and certificates yourself (Page 175).

Useful information: Certificate types

- Self-signed certificate:
  Each device generates and signs its own certificate. Application examples: Static configuration with limited number of communication nodes.
  No new certificates can be derived from a self-signed certificate. However, you need to load all self-signed certificates from partner devices to the CPU (STOP required).
- CA certificate:
  All certificates are generated and signed by a certificate authority. Application examples: Dynamically growing plants.
  You only need to download the certificate from the certificate authority to the CPU. The certificate authority can generate new certificates (partner devices can be added without CPU STOP).

Signing

The signature makes it possible to prove the integrity and source of a message as detailed below.
Signing starts with the sender creating a hash value from the plain text (plain text message). The sender then encrypts the hash value with its private key and subsequently transfers the plain text together with the encrypted hash value to the recipient. To verify the signature, the recipient needs the public key of the sender (this is contained in the X509 certificate of the sender). The recipient uses the sender's public key to decrypt the hash value received. The recipient then forms the hash value themselves from the plain text received (the hash process is contained in the sender's certificate). The recipient compares the two hash values:

- If the two hash values are identical, the plain text message has reached the receiver unchanged and has not been manipulated.
- If the two hash values do not match, the plain text message has not reached the receiver unchanged. The plain text message has been manipulated or has been distorted during transfer.

Encryption

Encrypting data prevents unauthorized parties from reading the content. X509 certificates are not encrypted; they are public and can be viewed by anyone.
Encryption involves the sender encrypting the plain text message with the public key of the recipient. To do so, the sender requires the recipient's X509 certificate, as it contains the public key of the recipient. The recipient decrypts the message with their private key. Only the recipient can decrypt the message: They alone hold the private key. The private key must therefore never be disclosed.

Secure channel

OPC UA uses the private and public key of client and server to establish a secure connection, the secure channel. Once the secure connection has been established, the client and server generate an internal key only known to them which they both use for signing and encrypting messages. This symmetric process (a shared key) is much faster than asymmetric processes (private and public key).

More information

An application example for the use of certificates with the TIA Portal can be found here: Using certificates with TIA Portal (https://support.industry.siemens.com/cs/ww/en/view/109769068).

10.2.3 Certificates with OPC UA

Usage of X509 certificates with OPC UA

OPC UA uses various types of X.509 certificates for establishing a connection from client to server:

- OPC UA application certificates
  Such X.509 certificates identify the software instance, the installation of client or server software. For the "Organization name" attribute, you enter the name of the company that uses the software.

  NOTE

  The OPC UA server of the S7-1500 uses application certificates even for the security setting "None" (no security). This ensures compatibility to OPC UA V1.1 and earlier versions.

- OPC UA software certificates
  This X-509 certificate identifies a specific version of the client or server software. These certificates contain attributes that describe which tests this version of the software has passed during certification by the OPC Foundation (or recognized test laboratories). For the "Organization name" attribute, you enter the name of the company that has developed or markets the software.

  NOTE

  Software certificates are not supported in STEP 7.

- OPC UA user certificates
  This X.509 certificate identifies the specific user who, for example, retrieves process data from the OPC UA server. This certificate is not required if the user can authenticate itself with a password, or if anonymous access is configured.

  NOTE

  User certificates are not supported in STEP 7.

The described certificates are end-entity certificates: They identify, for example, a person, an organization, a company or an instance (installation) of a software.

10.2.4 Creating self-signed certificates

Using the client's certificate generator

Many OPC UA client applications or SDKs are integrated in a sample application that allows you to generate certificates for the client from this application.
The description for certificate generation can generally be found in the context for describing the OPC UA client application.

Example client from the online support

The OPC UA .NET client for the SIMATIC S7-1500 OPC UA server (https://support.industry.siemens.com/cs/ww/en/view/109737901) creates a self-signed software certificate of the client application in the Windows Certificate Store during the first program start. The documentation for this example describes the procedure for handling these certificates.

Using the certificate generator of the TIA Portal

If you use an OPC UA client that does not generate a client certificate, you can create self-signed certificates with STEP 7.
To do this, follow these steps:

1. In the properties of the CPU, double-click "<Add new>" under "Protection & Security > Certificate manager > Device certificates".
2. Click "Add".
3. In the "Create a new certificate" dialog, select the "OPC UA client" option for "Usage".
4. Click "OK".

In the field "Subject Alternative Name" STEP 7 automatically enters the URI for the generated certificate. In the program-specific certificate generation by means of the .NET stack of the OPC Foundation, the field is called, for example, "ApplicationUri" - it can have a different name in other tools for certificate generation.

More information

For more information on handling client certificates, refer to the section Handling of the client certificates of the S7-1500 CPU (Page 327).

10.2.5 Generating PKI key pairs and certificates yourself

This section is only relevant if you want to use an OPC UA client that cannot itself create a PKI key pair and a client certificate. In this case, you generate a private and a public key using OpenSSL, generate an X.509 certificate, and sign the certificate yourself.

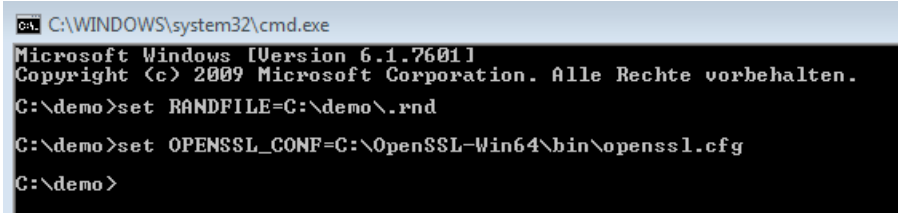
Using OpenSSL

OpenSSL is a tool for Transport Layer Security that you can use to create certificates. You can also use other tools, for example XCA, a type of key management software with a graphical user interface for an improved overview of certificates issued.

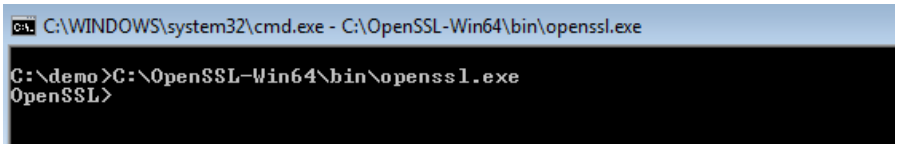
To work with OpenSSL under Windows, follow these steps:

1. Install OpenSSL under Windows. If you are using a 64-bit version of the operating system, install OpenSSL in the "C:\OpenSSL-Win64" directory, for example. You can obtain OpenSSL-Win64 as a download from various providers for open source software.
2. Create a directory, for example "C:\demo".
3. Open the command prompt. To do so, click "Start" and enter "cmd" or "command prompt" in the search field. Right-click "cmd.exe" in the results list and run the program as an administrator. Windows opens the command prompt.
4. Change to the "C:\demo" directory. To do this, enter the following command: "cd C:\demo".
5. Set the following network variables:
   - set RANDFILE=c:\demo\.rnd
   - set OPENSSL_CONF=C:\OpenSSL-Win64\bin\openssl.cfg

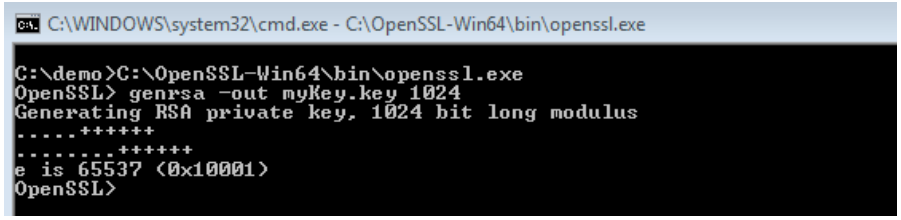
   The figure below shows the command line with the following commands:



6. Now start OpenSSL. If OpenSSL has been installed in the C:\OpenSSL-Win64 directory, enter the following: C:\OpenSSL-Win64\bin\openssl.exe The figure below shows the command line with the following command:



7. Generate a private key. Save the key to the "myKey.key" file. The key in this example is 1024 bits long; for greater RSA security, use 2048 bits in practice. Enter the following command: "genrsa -out myKey.key 2048" ("genrsa -out myKey.key 1024" in the example). The figure below shows the command line with the command and the output of OpenSSL:



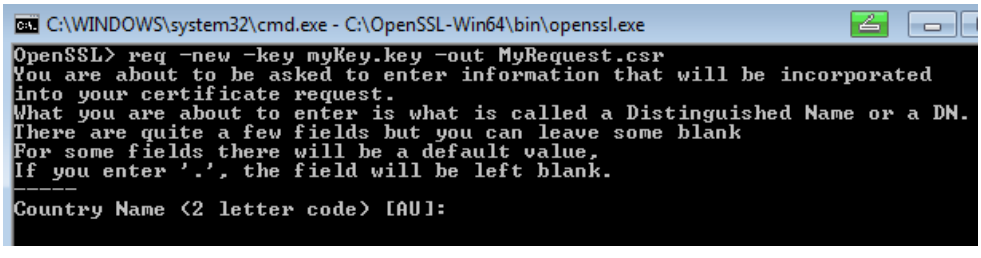
8. Generate a CSR (Certificate Signing Request). To do this, enter the following command: "req -new -key myKey.key -out myRequest.csr". During execution of this command, OpenSSL queries information about your certificate:
   - Country name: for example "DE" for Germany, "FR" for France
   - State or province name: for example "Bavaria".
   - Location Name: for example "Augsburg".
   - Organization Name: Enter the name of your company.
   - Organizational Unit Name: for example "IT"
   - Common Name: for example "OPC UA client of machine A"
   - Email Address:

NOTE

Note for S7-1500 CPU as server with firmware version V2.5

The IP address of the client program has to be stored in the "Subject Alternative Name" field of the created certificate for S7-1500 CPUs version V2.5 (only for this version); otherwise, the CPU will not accept the certificate.

The information you enter is added to the certificate. The figure below shows the command line with the command and the output of OpenSSL:



The command creates a file in the C:\demo directory containing the Certificate Signing Request (CSR); in the example, this is "myRequest.csr".

Using the CSR

There are two ways to use a CSR:

- You send the CSR to a certificate authority (CA): Read the information of the respective certification authority. The certificate authority (CA) checks your information and identity (authentication) and signs the certificate with the private key of the certificate authority. You receive the signed X.509 certificate and use this certificate for OPC UA, HTTPS or Secure OUC (secure open user communication), for example. Your communication partners use the public key of the certificate authority to check whether your certificate was really issued and signed by that CA (i.e. that the certificate authority has confirmed your information).
- You sign the CSR yourself: Using your private key. This option is shown in the next step.

Signing the certificate yourself

Enter the following command so that you can generate and sign your certificate (self-signed certificate) yourself: "x509 -req -days 365 -in myRequest.csr -signkey myKey.key -out myCertificate.crt".

The figure below shows the command line with the command and OpenSSL:

The command generates an X.509 certificate with the attributes that you transfer with the CSR (in the example "myRequest.csr"), for example with a validity of one year (-days 365). The command also signs the certificate with your private key ("myKey.key" in the example). Your communication partners can use your public key (contained in your certificate) to check that you are in possession of the private key that belongs to this public key. This also prevents your public key from being misused by an attacker.

With self-signed certificates, you yourself confirm that the information in your certificate is correct. There is no independent body that checks your information.

More information

You will find information about the handling of client certificates of the S7-1500 CPU in the section Handling of the client certificates of the S7-1500 CPU (Page 327).

10.2.6 Secure transfer of messages

Establishing secure connections with OPC UA

OPC UA uses secure connections between client and server. OPC UA checks the identity of the communication partners. OPC UA uses certificates in accordance with X.509-V3 from the ITU

(International Telecommunication Union) for client and server authentication. Exception: A secure connection is not established with the "No security" security policy.

Message security mode

OPC UA uses the following security policies to protect messages:

- No security
  All messages are unsecured. In order to use this security policy, establish a connection to a None end point of a server.
- Signing
  All message are signed. This allows the integrity of the messages received to be checked. Manipulations are detected. In order to use this security policy, establish a connection to a Sign end point of a server.
- Sign & Encrypt
  All messages are signed and encrypted. This allows the integrity of the messages received to be checked. Manipulations are detected. What is more, no attacker can read the content of the message (protection of confidentiality). In order to use this security policy, establish a connection to a "SignAndEncrypt" end point of a server.

The security policies are also named according to the algorithms used. Example: "Basic256Sha256 - Sign & Encrypt" means: Secure endpoint, supports a series of algorithms for 256-bit hashing and 256-bit encryption.

Layers required

The figure below shows the three layers that are always required for establishing a connection: the transport layer, the secure channel and the session.

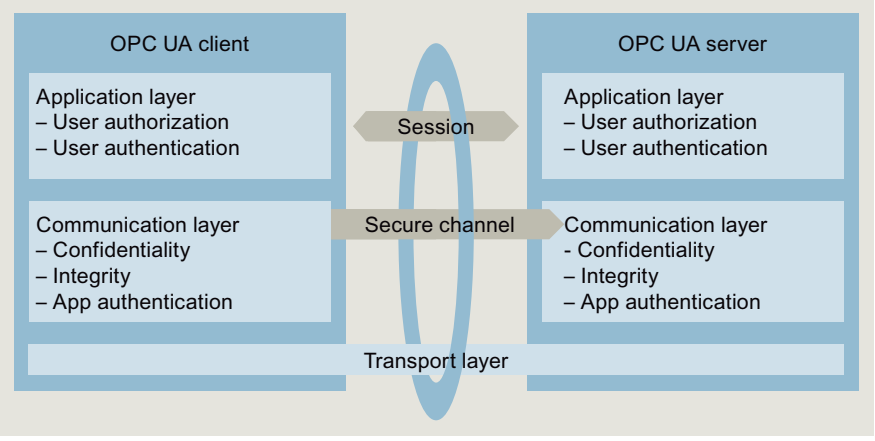


Figure 10-6 Necessary layers: transport layer, secure channel and session

- Transport layer:
  This layer sends and receives messages. OPC UA uses an optimized TCP-based binary protocol here. The transport layer is the basis for the subsequent secure channel.

- Secure channel
  The secure channel receives the data received from the transport layer, and forwards that data to the session. The secure channel forwards data of the session that is to be sent to the transport layer.
  In "Sign" security mode, the secure channel signs the data (messages) that is sent. When a message is received, the secure channel checks the signature to detect any manipulations.
  With a "SignAndEncrypt" security policy, the secure channel signs and encrypts the send data. Data received is decrypted by the secure channel, and the secure channel then checks the signature.
  With the "No security" security policy, the message packages pass the secure channel unchanged (the messages are received and sent in plain text).
- Session
  The session forwards the messages from the secure channel to the application, or receives from the application the messages that are to be sent. The application uses the process values or provides the values.

Establishing the secure channel

The secure channel is established as follows:

1. The server starts establishing the secure channel when it receives a request to this effect from the client. This request is signed or signed and encrypted, or the message is sent in plain text (security mode of the selected server end point). With "Sign" and "Sign & Encrypt", the client sends a "secret" (random number) with the request.
2. The server validates the client certificate (contained in the request, unencrypted) and checks the identity of the client. If the server trusts the client certificate,
   – it decrypts the message and checks the signature ("Sign & Encrypt"),
   – checks the signature only ("Sign"),
   – or leaves the message unchanged ("No security")
3. The server then sends a response to the client (same level of security as the request). The server secret is contained in the response. The client and server calculate a symmetric key from the client and server secret. The secure channel is now established.

The symmetric key (instead of the private and public key of client and server) is now used for signing and encrypting messages.

Establishment of the session

The session is executed as follows:

1. The client starts establishing the session by sending a CreateSessionRequest to the server. This message contains a Nonce, a random number that is only used once. The server must sign this random number (Nonce) to prove that it is the owner of the private key. The private key belongs to the certificate that the server uses to establish the secure channel. This message (and all subsequent messages) is secured in line with the security policies of the selected server endpoint (selected security policies).
2. The server responds with the CreateSession Response. This message contains the public key of the server and the signed Nonce. The client checks the signed Nonce.

3. If the server passes the test, the client sends a SessionActivateRequest to the server. This message contains the information that is required for user authentication:
   - User name and password, or
   - X.509 certificate of the user (not supported in STEP 7), or
   - No data (if anonymous access is configured).
4. If the user has the necessary rights, the server returns a message to the client (ActivateSessionResponse). This activates the session.

The secure connection between the OPC UA client and server has been established.

Establishing a connection to PLCopen function block

The PLCopen specification defines a range of IEC 61131 function blocks for OPC UA clients. The instruction UA_Connect initiates both a secure channel and a session following the pattern described above.

10.2.7 Certificate management via Global Discovery Server (GDS)

10.2.7.1 Automated certificate management with GDS

As of TIA Portal V17 and S7-1500 CPU firmware version V2.9, you can use the certificate management services of the OPC UA server to transfer OPC UA server certificates during runtime.

OPC UA certificates, trust lists and certificate revocation lists (CRLs) for the OPC UA server of the S7-1500 CPU can be updated automatically using GDS push management functions. The automation of the certificate management eliminates any manual work required for reconfiguring the CPU, for example, after the period of validity of a certificate has expired, and a fresh download of the CPU. You can also use the GDS push management functions to transfer updated certificates and lists in the STOP and RUN operating states of the CPU.

The certificate management information model is specified in OPC UA Part 12 (OPC 10000-12: OPC Unified Architecture, Part 12: Discovery and Global Services).

As of TIA Portal version V18 and S7-1500 CPU firmware version V3.0, you can also use the GDS push management function for web server certificates. The sequence of, for example, certificate updates via GDS push-management functions is identical here in principle to the certificate update of OPC UA server certificates. Instead of OPC UA server certificates, you transfer web server certificates to the CPU at runtime or during operation. The differences or restrictions are explained in the following description at the corresponding places.

The following sections provide a general overview of Global Discovery Services and the function of an automated certificate update supported as of TIA Portal V17 / CPU firmware version V2.9.

Discovery server

To connect to an OPC UA server, an OPC UA client requires information about its endpoint such as the endpoint URL and the security policy. When a large number of possible servers are available in the network, a discovery server can take over the search and management of this server information.

- OPC UA servers register with the discovery server.
- OPC UA clients request a list of accessible servers from the discovery server and then connect to the desired OPC UA server.

Global Discovery Server (GDS)

The OPC UA GDS concept allows the configuration of cross-subnet discovery services on the one hand and provides interfaces for central certificate management on the other hand.

A Global Discovery Server (GDS) makes mechanisms available for the central management of the following components:

- CA-signed certificates and self-signed certificates
- Trusted Lists and Certificate Revocation Lists (CRL)

A GDS thus provides an access point to central certificate management and takes over the task of a security server within an OPC UA network.

The main application of GDS is the management of CA-signed certificates with the corresponding CRLs:

- Initial creation of an OPC UA application certificate
- Regular update of the trust list and the CRLs
- Renovation of an application certificate

Certificate management

Certificate management has the task of automating the administration and distribution of certificates and trust lists for different services or UA applications.

In this context, a distinction is made between the following roles:

- Certificate manager - an OPC UA application that provides certificate management functions
- Certificate recipient – an OPC UA application that receives certificates, trust lists and CRLs from the certificate manager.

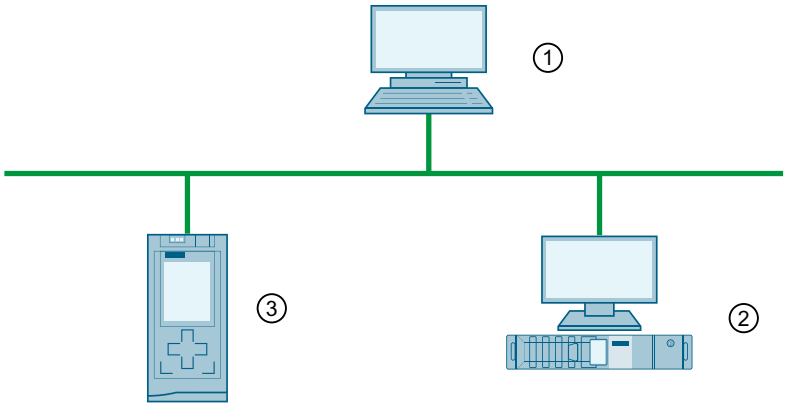
There are two models for certificate management: Pull and push management.

- With pull management, the OPC UA application acts as a client of the GDS server and uses certificate management methods to request certificate updates and trust list updates.
- With push management, the OPC UA application acts as a server and provides methods for an OPC UA GDS as OPC UA client. The GDS in the role of certificate manager uses these methods to transfer ("push") certificates and trusted list updates, see explanation of the concept for automated certificate update below.

As of firmware version V2.9, the S7-1500 CPU currently only supports push management for the OPC UA server of the CPU.

System configuration with GDS

The figure below shows an example of the tasks of the devices involved in combination with a GDS that provides certificate management functions.



① Root CA - device that issues certificates for the system (these certificates can also be transmitted in other ways, for example, by email)

② OPC UA GDS with certificate manager creates or signs device certificates, manages trust lists and certificate revocation lists (CRLs), and writes certificates and lists to the devices (push function). This device requires OPC UA client functionality for the push function.

③ Device with OPC UA application receives "pushed" certificates and lists

Concept for automated certificate update for STEP 7 version V17 and higher

GDS and certificate manager are usually combined into one application; however, in the figure below, they are two separate components.

Devices such as "normal" OPC UA clients are also suitable as certificate managers, but they need to support the Bytestring data type that is required to transfer certificates, for example, an S7-1500 CPU firmware V2.9 and higher as OPC UA client or the UA Expert tool (Unified Automation) with GDS plugin.

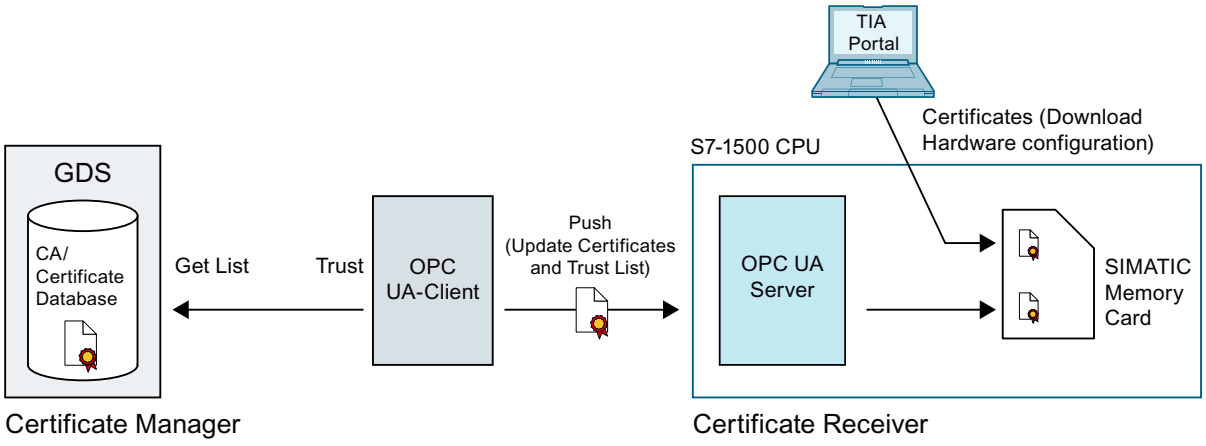
The OPC UA server of the S7-1500 CPU as certificate receiver provides the standardized methods and attributes that the OPC UA client certificates need to read and write trust lists and CRLs.

The focus in the context of the OPC UA server of the S7-1500 CPU is the description of the push function in contrast to the usual manner in which certificates are provided to the CPU: By loading the hardware configuration.

The figure below shows how to transfer certificates and lists for OPC UA in an S7-1500 CPU FW V2.9 or higher:

- Either by loading the hardware configuration in STOP of the CPU; the certificates are part of the hardware configuration.
- Or via GDS push methods in RUN or in STOP mode of the CPU.

It is not possible to use both transmission paths in parallel. If, for example, you have opted for transfer of OPC UA server certificates with GDS push functions at runtime, you must also transmit all the other certificate types to the CPU via this route.



More information

You can find more information on the certificates for OPC UA in the section Certificates with OPC UA (Page 173).

10.2.7.2 Configuration limits for Push function

Number of certificates for Push function

For the OPC UA Push function, an S7-1500 CPU, regardless of the type, has a configuration limit of 62 trust list entries as of firmware version V2.9.

- Each activated certificate-based service (CPU application) "consumes" one entry each for a certificate and an entry for the private key.
- A Certificate Revocation List entry (CRL) counts just as an entry in the list of trusted certificates.
- A certificate that is used by different services (CPU applications) counts as a single trust list entry.

Size of elements for Push function (e.g. certificates)

Max. 4096 bytes

Example

You want to grant access to the OPC UA server for up to 62 OPC UA clients and fill the trusted list accordingly.

When you add a Certificate Revocation List entry in the trusted list, you can only trust up to 61 client certificates.

Additional OPC UA certificates can **not** be transferred by loading the hardware configuration to the CPU.

Tip

To keep the number of required certificates low, we recommend having the OPC UA client certificates signed by the same CA.
In this case, the CPU as OPC UA server only needs the corresponding CA certificate and CRLs. With these elements, the OPC UA server can then verify all client certificates signed by the CA. This means you do not have to add the individual client certificates to the trusted list.

10.2.7.3 Setting and loading GDS parameters

The following describes the settings required for the certificate update.

Requirement

- Depending on the application certificate, the corresponding STEP 7/TIA Portal version and S7-1500 CPU firmware version is required.
  See also here: What you should know about the certificate management (Page 54)
  - For OPC UA server certificates, for example, TIA Portal from V17 onwards, CPU firmware version V2.9
  - For web server certificates, for example, from TIA Portal V18, CPU firmware version V3.0
- Timet/date of the CPU is set (generally applies to certificate-based communication)
- The OPC UA server is enabled.
- The service that the GDS push management uses must be enabled. For example, the web server must be enabled for the transfer of web server certificates.
- At least one endpoint with the "Sign & Encrypt" security policy must be configured. The partner must use this endpoint.
- An authenticated user with sufficient function rights is configured
  The user must have a role that has the function right "Manage certificates".
  This function right, in turn, has the following requirements:
  - **Project protection** must be enabled in the project tree: Project tree: "Security settings > Settings > Project protection".
  - In the "OPC UA > General" area of the CPU settings, the following general user management setting must be enabled: "Enable additional user management via project security settings"

The Users and roles with OPC UA function rights (Page 232) section describes how to set the function rights.

Activating GDS

When the requirements listed above are met, you must still enable the GDS:

1. In the Inspector window (CPU parameters), go to the "OPC UA > Server > General" area.
2. Enable the "Enable Global Discovery Services (Push)" option.

Determining the certificate store used

Certificates that are managed using GDS, are in a different memory area than the certificates that are downloaded via the TIA Portal (STEP 7).

When GDS push certificate management is enabled, the services (applications) of the CPU also use certificates from the certificate store whose certificates are managed at runtime.

1. In the CPU settings, navigate to the area "Protection & Security > Certificate management".
2. Select the "Use certificates provided by the certificate management at runtime" option.
   The other option (use certificates configured and downloaded using TIA Portal) uses the certificates that are downloaded to the CPU from the TIA Portal with the configuration in CPU STOP. Certificates or trust lists cannot be updated in this certificate store during runtime.

Enabling the diagnostics for the lapsing of certificates

If you wish to be informed in advance about the lapsing of a certificate, select the "Enable system diagnostics event for the certificate lapsing" option in the area "Protection & Security > Certificate management".

In the input field "Show event at remaining certificate validity period of:" enter a percent value.

Effect of these settings:

- At the instant when this value is reached by a certificate, a corresponding system diagnostics message appears till the certificate lapses or is refreshed.
- If the end of the validity period of a certificate has been reached, the CPU generates a corresponding system diagnostics message as well as an entry in the diagnostics buffer and the maintenance LED lights up.

Example:
The certificate transferred via GDS on June 01, 2022 has a validity from June 01, 2022 to June 30, 2022 (30 days). You have input a percent value of 10 for the diagnostics event. On June 27, 2022, when 90% of the period of validity lapses, a corresponding message will appear stating that the certificate that had been transferred will lapse on June 30, 2022. Regardless of the configured percentage value, upon lapsing of the period of validity of a certificate, a corresponding message is displayed in any case, an entry is made in the diagnostics buffer, and the maintenance LED lights up.

Download to CPU

When downloading the configuration to the CPU, you can delete the certificates that are managed via GDS before the download. When you confirm the deletion, the download is followed by a provisioning phase (see section on commissioning).

When you download the memory card outside of the CPU (card reader), this certificate store is always deleted.

When Global Discovery Services (Push) is activated and no pushed certificates are available, then no separate certificate, trust list or CRL is available for the OPC UA server.

10.2.7.4 GDS commissioning

Part 12 of the OPC UA specification distinguishes between a provisioning phase and a run time phase during certificate management.
In the provisioning phase, a GDS or OPC UA client provides initial trust lists and CRLs for clients of the OPC UA server. In this phase, the OPC UA server of the CPU accepts all client certificates and lists it is offered – similar to the "Trusted clients" setting for the OPC UA server that all client certificates are accepted during runtime. This is the only way in which a connection to clients not known to the server is possible. For example, clients that the server cannot authenticate using existing certificates or trust lists until it has received the corresponding client certificate or the corresponding trust list.
The provisioning phase is characterized by lower security; therefore, the provisioning phase is indicated by a lit Maintenance LED and a corresponding diagnostics buffer entry (Maintenance demanded).
During the runtime phase, the existing CRLs are updated, for example, and the certificates and trust lists are renewed. Communication is secure in this phase.

Requirement

Only authorized users with sufficient function rights can set up a connection in the provisioning phase. The users must have a role with the function right "Manage certificates".
See also Setting and loading GDS parameters (Page 184).

Rules for the provisioning phase

In the provisioning phase, the OPC UA server of the CPU cannot authenticate the OPC UA clients that initiate connection establishment. Therefore, the following rules must be observed:

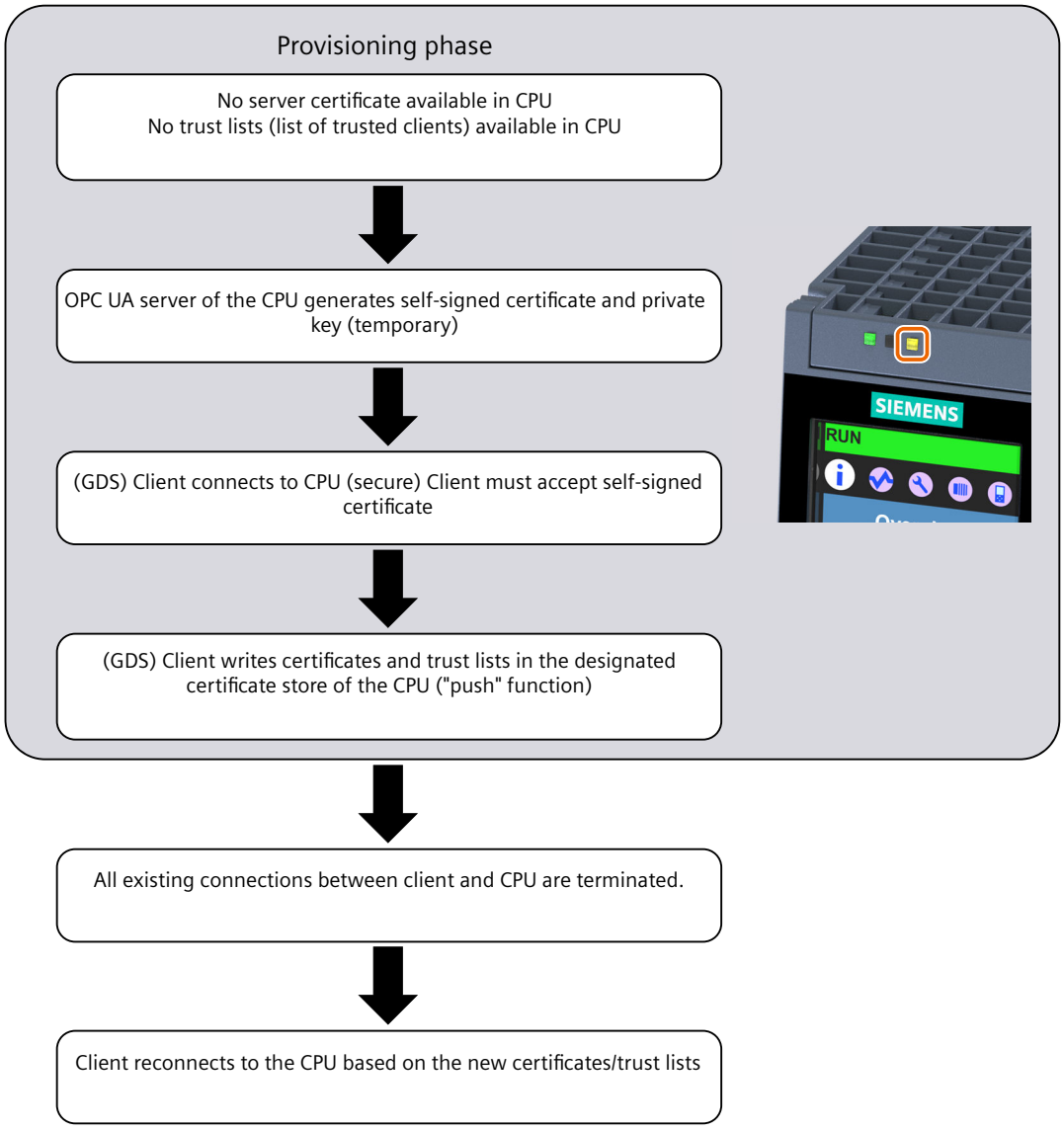
- Provide a secure environment, for example, access to the CPU is limited to commissioning personnel. Check that the right devices are communicating with one another.
- Limit the time for this phase.

The CPU signals that it is in the provisioning phase by the lit Maintenance LED as well as a corresponding diagnostics buffer entry (Maintenance demanded).

Sequence of the provisioning phase

The following provides an outline of the process of the provisioning phase for OPC UA server certificates and
trust lists.

The process of the provisioning phase for web server certificates is comparable. In contrast to OPC UA, the GDS client only pushes web server certificates but not trust lists into the corresponding certificate store.



Entering the provisioning phase

After startup of the OPC UA server, the CPU automatically enters the provisioning phase when one of the following conditions is met:

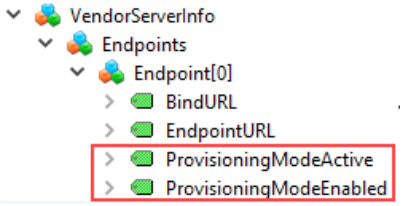
- The OPC UA server certificate is the initial self-signed certificate generated by the CPU and has not yet been replaced by a valid server certificate.
- The trust list (list of trustworthy clients) is empty.

The OPC UA server certificate generated by the CPU contains the most important parameters of the OPC UA server and is re-generated, including the private key, after each startup of the

OPC UA server following POWER ON - until the valid server certificate is present. For this reason, the OPC UA server may take longer to start up after a POWER ON.
After the hardware configuration is downloaded, the certificate store for certificates that can be updated during runtime is deleted on download or the certificates are retained, depending on the setting. This means that if GDS is activated and the certificate store has been deleted, the CPU is in the provisioning stage after the hardware configuration has be loaded.

Provisioning phase diagnostics

In addition to the lit Maintenance LED, the GDS address model has two nodes that provide information on whether the OPC UA server of the CPU is in the provisioning phase:



You can only use the two nodes as marked in the figure for diagnostics if the requirements for GDS are met (endpoint security signed & encrypted plus administrator function rights available).
ProvisioningModeEnabled: Indicates that a provisioning phase is supported
ProvisioningModeActive: Indicates that the OPC UA server of the CPU is in the provisioning phase.

End of the provisioning phase

The CPU ends the provisioning phase automatically when the following conditions are met:

- The certificate generated and self-signed by the CPU for the provisioning phase has been overwritten by a valid server certificate. This valid server certificate can be a self-signed certificate or a CA-signed certificate.
- The trust list in the CPU is not empty, i.e. client certificates of the OPC UA clients to be trusted or CA certificates for checking the client certificates are available.

If the OPC UA client transfers a CA-signed certificate and also adds the CA certificate to the trust list, the OPC UA server of the CPU can automatically accept all other certificates from OPC UA clients that were signed by the same CA.

Request of a valid server certificate

As of TIA Portal version V18 / S7-1500 CPU version V3.0, in addition to OPC UA server certificates, certificates for other services can also be transferred to the CPU, for example, for the web server.

The corresponding service, for example, the OPC UA server of the CPU, receives a valid certificate in the following steps:

1. A GDS client (OPC UA client) calls the method "CreateSigningRequest" to request a server certificate: with a Certificate Signing Request (CSR).
2. This CSR must be signed by a Certificate Authority (CA).
3. The signed CSR must then be transferred back to the OPC UA server of the CPU as a certificate.

The OPC UA server of the CPU makes this method available if the client has the required function right "Manage certificates".

The "CreateSigningRequest" method allows for the following variants:

- Certificate update without creating a new key pair (internal CPU keys that are already available are used)
- Certificate update with creation of a new key pair (CPU-internal)

There is also the possibility to generate certificates with externally created key pairs.

NOTICE
Recommended procedure to generate certificates Transport of private keys should be avoided; a private key should not leave a device. We, therefore, recommend the generation of a certificate without creating a new key pair or with the creation of a key pair inside the CPU.

Create certificate without key pair

- The "CreateSigningRequest" method returns a Certificate Signing Request (CSR), that is, a file (*.csr) with specific information on the server or on the service, for example, application name and URL.
- Outside of the CPU, this CSR must be validated and signed by a Certificate Authority (CA) and the certificate must be returned.
- The certificate must then be transferred to the CPU ("pushed") using the "UpdateCertificate" method.

The key does not leave the CPU in this scenario.

Create certificate with internally created key pair

The procedure is similar to the method explained in the previous section; the only difference is that a key pair is generated in addition to the CSR. You specify in the parameter of the "CreateSigningRequest" method that it is to generate a key pair.

The private key does not leave the CPU in this procedure either.

The generation of a new key pair creates a very heavy load on the CPU. The CPU processes this request over a longer period of time with lower priority in the reserved area for the communication load. The duration of this time period depends on the performance of the CPU.

Because the share of the set communication load is fully utilized during key generation over a longer period of time, set the "Cycle load due to communication" share so that the maximum cycle time is not exceeded and sufficient reserves are available. For this, use the web server page "Diagnostics > Runtime information" of the CPU. This page shows information about the current program/communication load and cycle time of your user program. Via a controller, you can get help on the effects of a changed communication load on the cycle time.

Create certificate with externally created key pair

The certificate is generated with the help of tools, for example, that can generate additional keys.

Certificate and keys are transferred to the CPU using the "UpdateCertificate" method.

Due to low security, this procedure is not recommended.

NOTICE
Different keys for different target systems Always use newly generated keys for a production system. If you simulate and test your project, e.g. with PLCSIM Advanced on your PC, do not under any circumstances use the keys used for the simulation also for a productive system. Restrict the access to PC-based controllers by setting up appropriate permissions.

10.2.7.5 Address model for the push certificate management

The OPC UA specification Part 12 (OPC 10000-12: Discovery, Global Services) defines methods and attributes for OPC UA servers, for example, that enable GDS or OPC UA clients to update certificates and trust lists on the server ("Push certificate management"). These methods and attributes are also included in the address model of the OPC UA server.

The relevant section in the address model of the OPC UA server of the S7-1500 CPU is explained below.

Requirement

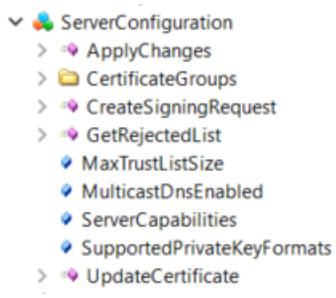
The following requirements must be met for the relevant methods and attributes to become visible for the GDS push functionality:

- GDS is activated.
- The set security policy supports the integrity and confidentiality of the data through signature and encryption (Sign & Encrypt).
- Access with runtime function right "Manage certificates"

Address model for the GDS push functionality

The address model for the GDS push functionality corresponds to the "Information Model for Push Certificate Management" of the OPC UA specification OPC 10000-12: Discovery, Global Services.

You will find the following structure below the "ServerConfiguration" node:



Methods and attributes for access to the address model

The methods and attributes are briefly described below with special features and restrictions of the specific address model of the S7-1500 CPU. The OPC UA specification listed above contains the general description.
You can find a detailed description of the individual methods below this overview table.

Method / Attribute (Variable)	Description
CreateSigningRequest	Method for generating a PKCS#10-encoded certificate request signed with the private key of the service (for example, OPC UA server).
UpdateCertificate	Method for updating the server certificate, for example, for the OPC UA server.
ApplyChanges	Method for applying a security-relevant change if the "ApplyChangesRequired" attribute was set when executing a previously executed method. **Note** If, as a result of "ApplyChanges", a certificate is changed, the CPU interrupts the connections/sessions that are secured via this certificate. Background: The basis for the secured connections - the certificate - is no longer valid.
GetRejectedList	Method that returns a list of certificates that were rejected by the OPC UA server. Rejected certificates are currently not stored by the OPC UA server of the S7-1500 CPUs. The method returns an empty array (RejectedList).
ServerCapabilities	Variable is not supported by the OPC UA server of the S7-1500 CPU.
SupportedPrivateKeyFormats	Variable that specifies permitted formats of the private key. For S7-1500 CPUs only "PEM" (String Array)
MaxTrustListSize	Variable that specifies the maximum size of the trust list.
MulticastDnsEnabled	Variable that specifies whether multicast DNS is supported. For S7-1500 CPUs, the value is "False".

Method / Attribute (Variable)	Description
CertificateGroups	Object (folder) that organizes all certificate groups supported by the OPC UA server. The certificate groups contain the objects that can be updated dynamically during runtime: For example, one trust list each and one or multiple certificates that are assigned to a service (for example, OPC UA application). Details on the structure of the CertificateGroups object and the methods and attributes that are available in the object are described in the next section.

CreateSigningRequest

The method has the following parameters:

Parameter	Data type	Description
[in] certificateGroupId	NodeId	NodeId of the CertificateGroup object.
[in] certificateTypeId	NodeId	Requested certificate type. List of permitted certificate types is specified by the "CertificateTypes" variable of the certificate group. For the OPC UA server for example, certificate type "RsaSha256ApplicationCertificateType", for the web server the certificate type "HttpsCertificateType"
[in] subjectName	String	Subject Name that is requested in the Certificate Request. If not specified, the current Subject Name of the certificate is used.
[in] regeneratePrivateKey	Boolean	True: Server generates a new private key. This key is saves until the UpdateCertificate methods with the matching signed certificate is called. False: Server uses the available private key.
[in] nonce	ByteString	Additional nonce for generating the new private key (see regeneratePrivateKey). Must be at least 32 bytes long.
[out] certificateRequest	ByteString	PKCS #10 - DER coded Certificate Request.

Method Result Codes

Result Code	Description
Bad_InvalidArgument	certificateTypeId, certificateGroupId or subjectName is invalid.
Bad_UserAccessDenied	The current user does not have the required function rights.

UpdateCertificate

Applications:

- Generation of certificate with CreateSigningRequest. No private key is available.
- New private key and new certificate were generated outside of the server. Both are updated with UpdateCertificate.
- Certificate generated and signed with the private key of the existing certificate. No private key is available.

Parameter	Data type	Description
[in] certificateGroupId	NodeId	NodeId of the CertificateGroup object.
[in] certificateTypeId	NodeId	Requested certificate type. List of permitted certificate types is specified by the "CertificateTypes" variable of the certificate group.
[in] certificate	ByteString	DER-coded certificate that replaces the existing certificate.
[in] issuerCertificates	ByteString	Issuer certificates
[in] privateKeyFormat	String	Format of the private key. Currently only PEM is supported. If the privateKey is not specified: zero or empty string.
[in] privateKey	ByteString	Privater key coded as specified in privateKey-Format.
[out] applyChangesRequired	Boolean	Indicates that the "ApplyChanges" method must be called before you use the new certificate.

Method Result Codes

Result Code	Description
Bad_InvalidArgument	certificateTypeId or certificateGroupId is invalid.
Bad_CertificateInvalid	The certificate is invalid or the format is not supported.
Bad_NotSupported	The private key is invalid or the format is not supported.
Bad_UserAccessDenied	The current user does not have the required function rights.
Bad_SecurityChecksFailed	Error occurred when verifying the integrity of the certificate.

Apply Changes

The method has no parameters.

Method Result Codes

Result Code	Description
Bad_UserAccessDenied	The current user does not have the required function rights.

GetRejectedList

The method has the following parameters:

Parameter	Data type	Description
[out] certificates	ByteStrings	DER-coded list of rejected certificates. The method currently returns an empty list (empty array), because rejected certificates are not stored.

Method Result Codes

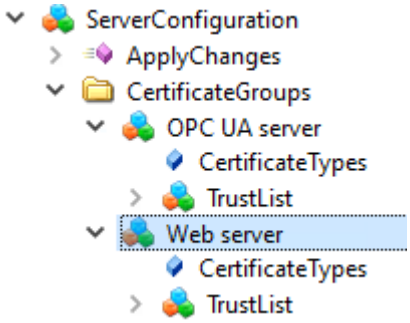
Result Code	Description
Bad_UserAccessDenied	The current user does not have the required function rights.

10.2.7.6 CertificateGroups in the address model

Certificates and trust lists for services or applications of the CPU (for example, OPC UA servers) that can be updated during runtime are located in the address model in the "CertificateGroups" object - there is one certificate group each for the various services of the S7-1500 CPU. For the OPC UA server certificate, the certificate group has the name "OPC UA server".

CertificateGroup in the address model

The following figure shows the structure of the "CertificateGroups" object below the "ServerConfiguration" node.



You can change the Display Name of the CertificateGroups (e.g. of the "OPC UA server") in STEP 7 (TIA Portal):

1. In the Inspector window (CPU properties), navigate to the area "Protection & Security > Certificate management".
2. Enable the option "Use certificates provided by certificate management during runtime" option.
3. Change the group name (DisplayName) of the certificate group in the table below. 1-64 characters in 7-bit ASCII format are permitted.
   The first column contains the activated service for which certificates can be transferred at runtime and the "ID" column contains a fixed numeric identifier that is used CPU-internally for referencing the certificate.

Here is an example for the display in the area "Certificate management":

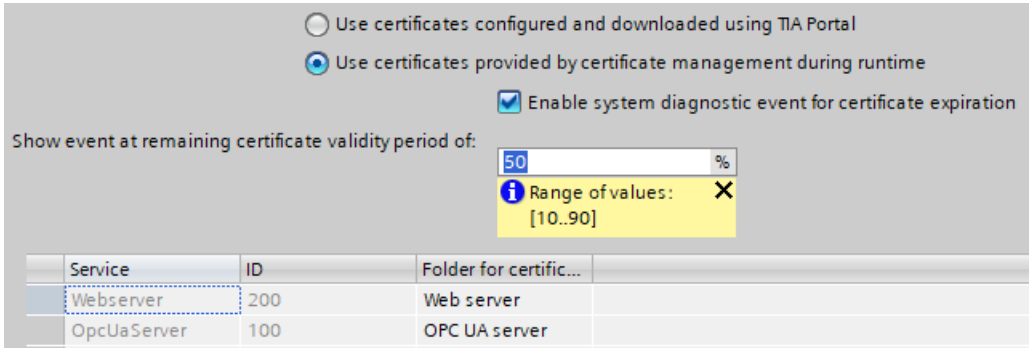


Figure 10-7 Certificate management settings

"CertificateTypes" node

The "CertificateTypes" variable specifies the NodeIds of the certificate types that are assigned to the server application.

For the OPC UA server service, for example, the "RsaSha256ApplicationCertificateType" CertificateType is supported, for the web server, it is the "HttpsCertificateType CertificateType".

"TrustList" node

The node for the trust list object (TrustList file) defines an OPC UA file type (Binary encoded stream) that contains information on the certificates and CRLs that can be read and updated in the "pki store\trusted\issuer" directory of the Memory Card. This node provides methods and attributes that make reading and updating possible.

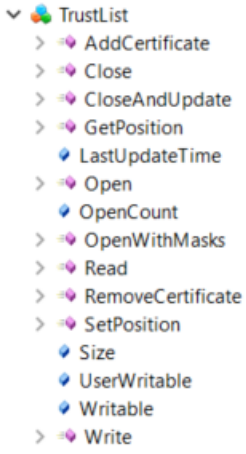
The node is an instance of the OPC UA data type "TrustListDataType" with the following structure:

Parameter	Data type	Description
specifiedLists	TrustListsMasks	Bit mask that shows which lists contain information.
trustedCertificates	ByteStrings	List of the application certificates and CA certificates that can be trusted.

Parameter	Data type	Description
trustedCrls	ByteStrings	CRLs for the certificates in the "trustedCertificates" list.
issuerCertificates	ByteStrings	List of the CA certificates that are required for validating the CA signed certificates.
issuerCrls	ByteStrings	CRLs of the CA certificates in the "issuerCertificates" list.

Structure of the "TrustList" node

The "TrustList" node has the following structure:



Methods and attributes for the "TrustList" node

Below is a description of the nodes under "TrustList" that supplement the methods of the Object Type "FileType". The TrustList Type is derived from FileType (see OPC 10000-5: OPC Unified Architecture, Part 5: Information Model).

Method / Attribute (Variable)	Description
LastUpdateTime	Variable that shows the time of the last update.
OpenWithMasks	Method that permits a client to only read a part of the TrustList.
CloseAndUpdate	Method for closing the TrustList file and applying the changes.
AddCertificate	Method for adding a single certificate to the TrustList.
RemoveCertificate	Method for removing a single certificate from the TrustList.

Description of the methods

The description of the methods with their result codes, attributes and types of the TrustList object is available in the OPC UA specification Part 12, Discovery and Global Services.

10.3 Using the S7-1500 as an OPC UA server

10.3.1 Interesting information about the OPC UA server of the S7-1500 CPUs

10.3.1.1 The OPC UA server of the S7-1500 CPUs

The S7-1500 CPUs as of firmware V2.0 are equipped with an OPC UA server. Apart from the Standard-S7-1500 CPUs this applies to the variants S7-1500F, S7-1500T, S7-1500C, S7-1500pro CPUs, ET 200SP CPUs, SIMATIC S7-1500 SW controllers and PLCSIM Advanced.
Convention: "S7-1500 CPUs" also includes the above-mentioned CPU variants.

S7-1500 CPU OPC UA server basics

Access to the OPC UA server of the CPU is possible via all integrated Ethernet interfaces of the S7-1500 CPU.
Direct access to the OPC UA server of the CPU over the backplane bus of the automation system is not possible via CPs under the following conditions:

- Configuration with TIA Portal Version V16 or higher
- S7-1500 CPU firmware version 2.8 or higher and CP 1543-1 firmware version V2.2 or higher

For configuration, see Access to OPC UA applications (Page 156).
Direct access to the OPC UA server of the CPU over the backplane bus of the automation system is not possible via CMs.
For access by clients, the server saves the enabled PLC tags and other information in the form of nodes (see Accessing OPC UA server data (Page 203)). These nodes are interconnected and form a network. OPC UA defines access points to this network (well-known nodes) that enable navigation to subordinate nodes.
With an OPC UA client you can read, observe or write values of tags of the PLC program as well as call methods that are available to the server. As of firmware version 2.5 you can implement methods, see Useful information about server methods (Page 268).

Node classes

OPC UA servers provide information in the form of nodes. A node can be, for example, an object, a tag, a method or a property.

The example below shows the address space of the OPC UA server of an S7-1500 CPU (extract from the OPC UA client "UaExpert" from Unified Automation).

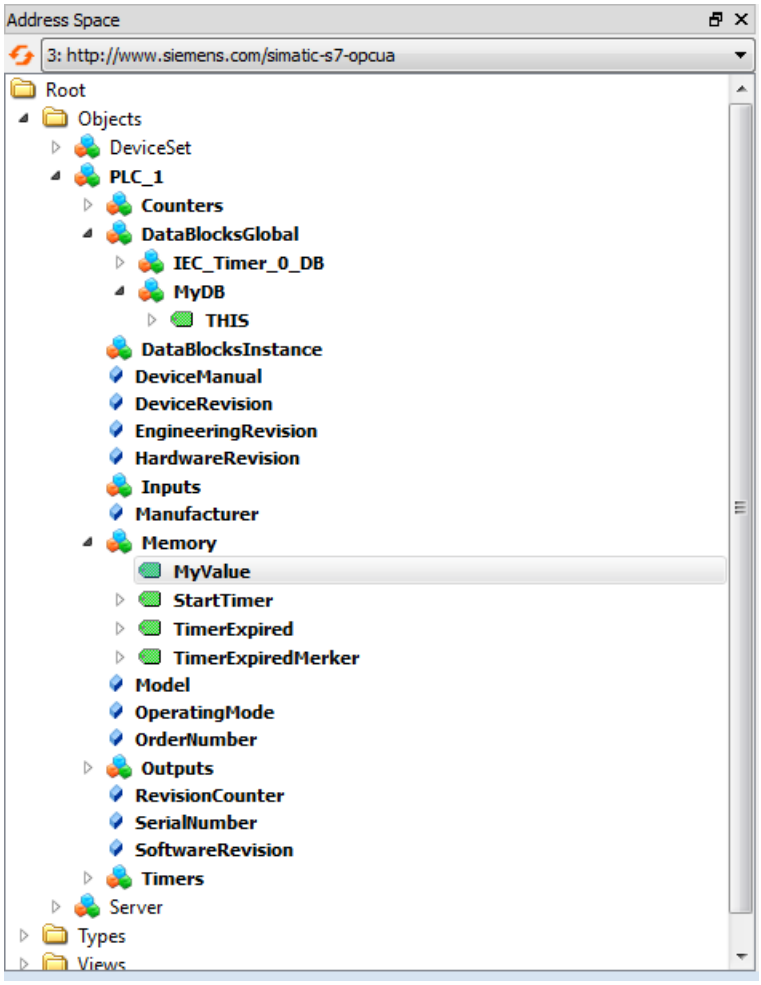


Figure 10-8 Example of the address space of the OPC UA server of an S7-1500 CPU

In the figure above, the "MyValue" tag is selected (highlighted in gray).
This tag is located below the "Memory" node, which has the node class "Object".
"Memory" is below the "PLC_1" node (also an Object).

Address space

The nodes are linked over references, for example, the reference "HasComponent, which represents a hierarchical relationship between a node and its subordinate nodes. With their references, the nodes form a network that can, for example, take the form of a tree.
A network of nodes is also called an address space. Starting from the root, all nodes can be reached in the address space.

10.3.1.2 End points of the OPC UA server

The end points of the OPC UA server define the security level for a connection. Depending on the purpose of use or desired security level, you have to carry out the corresponding settings for the connection at the end point.

Different security settings

Before establishing a secure connection, OPC UA clients ask the server with which security settings connections are possible. The server returns a list with all the security settings (endpoints) that the server offers.

Structure of end points

End points consist of the following components:

- Identifier for OPC: "opc.tcp"
- IP address: 192.168.178.151 (in the example)
- Port number for OPC UA: 4840 (standard port)
  The port number can be configured.
- Security setting for messages (Message Security Mode): None, Sign, SignAndEncrypt.
- Encryption and hash procedures (Security Policy): None, Basic128Rsa15, Basic256, Basic256Sha256 (in the example).

The following figure shows the "UA Sample Client" of the OPC Foundation.

The client has established a secure connection to the OPC UA server of an S7-1500 CPU to the end point "opc.tcp://192.168.178.151:4840 - [SignAndEncrypt: Basic256Sha256:Binary]". The security settings "SignAndEncrypt:Basic256Sha256" are contained in the end point.

NOTE

Select an endpoint with as strict as possible a security policy

Select an application-appropriate security policy for the end point and disable the less strict security policy at the OPC UA server.

A Sha256 certificate is required for the most secure end points (Basic256Sha256) of the S7-1500 CPU OPC UA server.

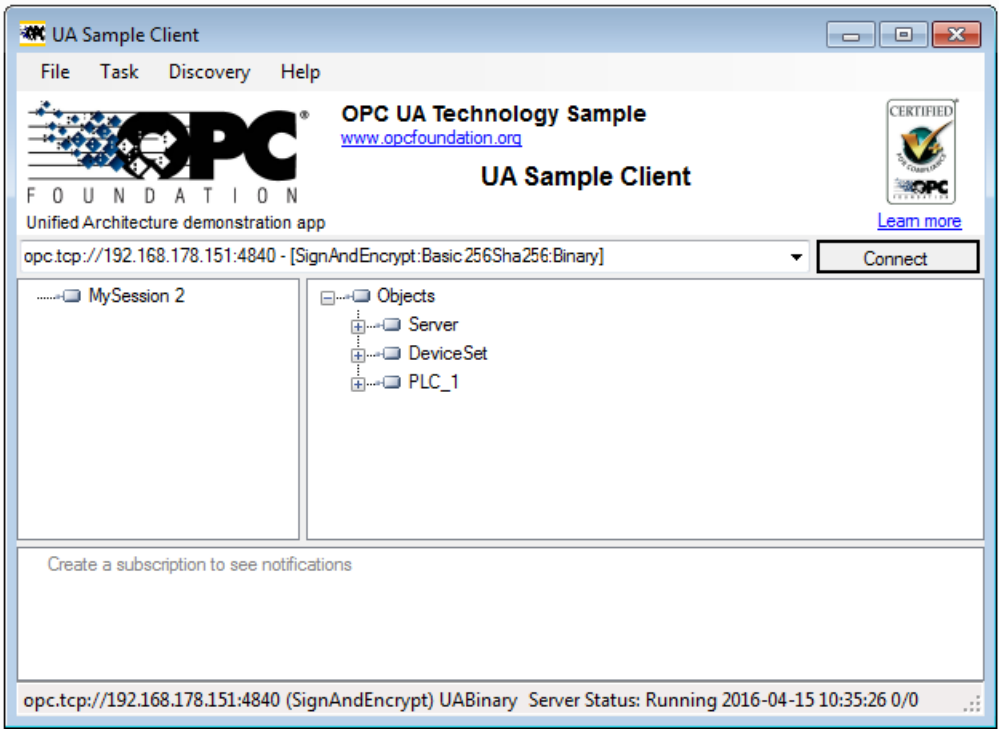


Figure 10-9 "UA Sample Client" program of the OPC Foundation

A connection to a server end point is only established if the OPC UA client complies with the security policies of that end point.

Through the information provided by the OPC UA server

OPC UA servers provide a wide range of information:

- The values of PLC tabs and DB components which clients may access.
- The data types of these PLC tags and DB components.
- Information on the OPC UA server itself and on the CPU.

This gives clients an overview and allows them to read out specific information. Previous knowledge of the PLC program and the CPU data is not required. You do not need to ask the developer of the PLC program when PLC tags are to be read. All necessary information is stored on the server itself (for example, the data types of the PLC tags).

Display of the information of the OPC UA server

You have the following options:

- Online: You have all the available information displayed during the runtime of the OPC UA server. To do so, navigate (browse) the address space of the server.
- Offline: You export an XML file that is based on the XML schemes of the OPC Foundation.
  Server methods created by the user (FB instance that can be called by an OPC UA client) are not exported as of STEP 7 V15.1), see Providing methods on the OPC UA server (Page 267).
- Offline with the Openness API: In your program, you use the API (Application Programming Interface) of the TIA Portal to access the function for exporting all PLC tags that can be read by OPC UA. This requires .NET Framework 4.0; see TIA Portal Openness, Automating SIMATIC projects with scripts (https://support.industry.siemens.com/cs/ww/en/view/109477163).
- If you already know the syntax and the PLC program, you can access the OPC UA server without first researching the information.

10.3.1.3 Runtime behavior of the OPC UA server

OPC UA server in operation

The OPC UA server of the S7-1500 CPU starts when you activate the server and download the project to the CPU.
How to activate the OPC UA server is described here.

Behavior in the operating state STOP of the CPU

An activated OPC UA server remains in operation even if the CPU switches to "STOP". The OPC UA server continues to respond to requests from OPC UA clients.
Server response in detail:

- If you request the values of PLC tags, you will get what were the latest values before the CPU switched to or was set to "STOP".
- If you write values to the OPC UA server, the OPC UA server will accept those values.
  However, the CPU will not process the values because the user program is not executed in "STOP" mode.
  An OPC UA client can nonetheless read the values written at STOP from the OPC UA server of the CPU.
  During restart, the CPU overwrites the values written at STOP with the start of the PLC tags.
- If you call a server method, the error message 16#00AF_0000 (BadInvalidState) is output because the server method (user program) is not running.
- Connections to the OPC UA server remain active during an operating mode transition (STOP > RUN or RUN > STOP). Exception: OPC UA-relevant data is loaded, see next section.

Download to the CPU may affect OPC UA server

If you load a CPU with running OPC UA server, you may need to stop and restart the server depending on the loaded objects. In this case, active connections are interrupted and must be re-established once the server restarts.

The duration of the restart depends mainly on the following parameters:

- The scope of the data structure
- The number of variables visible in the OPC UA address space
- The setting for downward compatible data type definition according to OPC UA specification to V1.03 (TypeDictionary enabled)
- Settings for the communication load and minimum cycle time, you can find additional information here (Page 336)

With CPU FW versions older than V2.8, the OPC UA server was stopped at each download to the CPU and then restarted.

As of FW version V2.8, the behavior of the OPC UA server has been optimized as follows:

- When objects are downloaded in STOP operating state of the CPU, the OPC UA server still always stops and then restarts. STEP 7 does not show a warning in this case.
- When objects are downloaded in RUN operating state of the CPU, the OPC UA server only stops if the downloaded objects are, or could be, OPC UA-relevant. The OPC UA server restarts after re-initialization due to the modified OPC UA data.

  Before OPC-UA-relevant objects are loaded into the CPU and stop the OPC UA server, STEP 7 displays a warning in the preview dialog for loading. You can then decide whether a server restart is compatible for the running process or whether you want to cancel the download. These warnings are only displayed when the OPC UA server is running. If the OPC UA server is not enabled, modified OPC UA data have no influence on the download process.

Examples

- You only want to add another code module to the program.
  Neither data blocks nor inputs, outputs, flags, times or counters are affected.
  Reaction during loading: A running OPC UA server is not interrupted.
- You want to load a new data module and you have flagged the data module as non-OPC-UA-relevant:
  Reaction during loading: A running OPC UA server is not interrupted.
- You want to overwrite a data module.
  Reaction during loading: A warning appears that the server will be restarted.
  Background: STEP 7 cannot determine whether the changes refer to OPC-UA-relevant data or not.

Reading CPU operating mode over OPC UA server

The OPC UA server allows you to read out the CPU mode, see figure below:

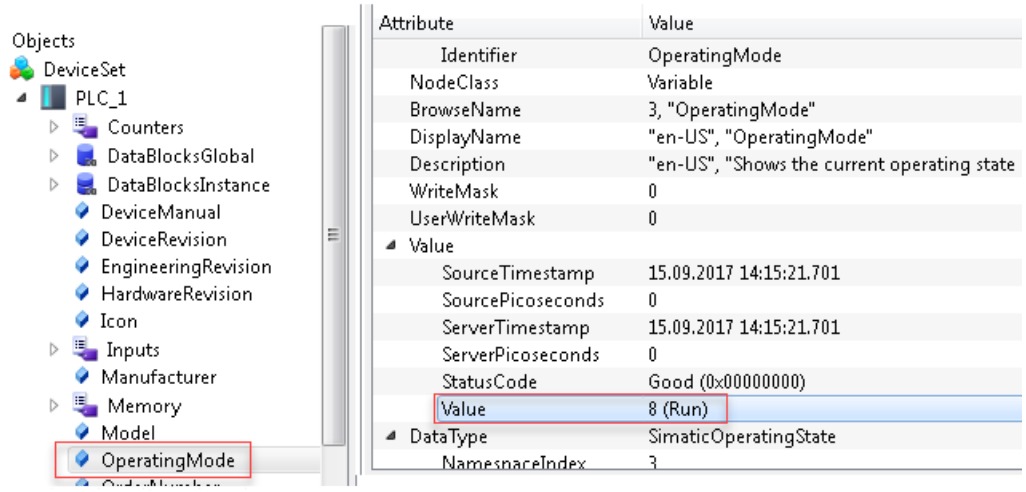


Figure 10-10 Reading CPU operating mode over OPC UA server

In addition to the operating mode of the CPU you can, for example, read out information in the manual (DeviceManual) or firmware version (HardwareRevision).

10.3.2 Accessing OPC UA server data

10.3.2.1 Client accesses and local accesses to the OPC UA server

An OPC UA server provides a lot of information for OPC UA clients within a network. The following section describes the options for making CPU variables (PLC variables and DB elements) available in the address space of your own OPC UA server.

Provide CPU variables via server interfaces in the OPC UA address space

The easiest way to transfer CPU variables automatically into the address space of the OPC UA server:

- Activate the standard SIMATIC server interface in the OPC UA properties of the CPU.
  All CPU variables released for OPC UA are then automatically also available in the OPC UA address space under the name of the CPU.

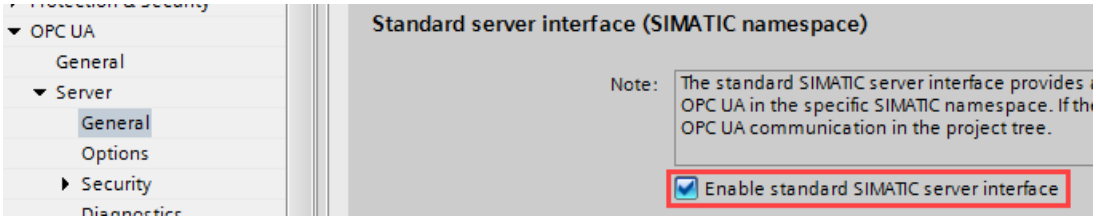


Figure 10-11 Standard SIMATIC server interface of the OPC UA server

The use of OPC UA server interfaces is more flexible and clearer; you configure the server interfaces in the project tree (below the CPU, "OPC UA Communication folder"). User-defined OPC UA server interfaces allow you to easily map OPC UA tags and CPU variables (local data).

OPC UA server interface

Browse name	Node type	Access level	Local data
myServerInterface	Interface	---	
myOPC_UA_Variable	BOOL	RD	"myDataBlock"."myVarBool"
myOPC_UA_Variable2	Word	RD/WR	"myDataBlock"."myVarWord"

Figure 10-12 Creating a user-defined server interface with mapped CPU tags

The data exchange between OPC UA client and OPC UA server is clearly illustrated in the following example of two S7-1500 CPUs.
Here, an S7-1500 CPU as client writes values to an OPC UA tag of the OPC UA server. The mapping between CPU variable and OPC UA tag makes it look as though the OPC UA client writes a value directly into the CPU variable. For an S7-1500 client CPU, use the "OPC_UA_WriteList" instruction in connection with additionally required instructions for the data exchange.

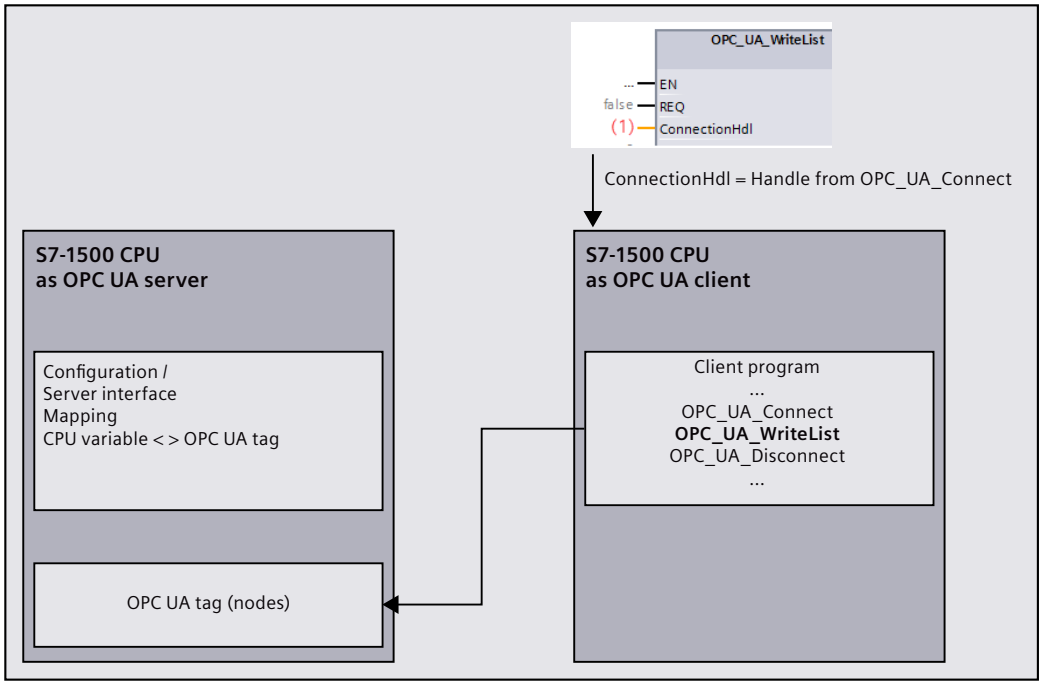


Figure 10-13 Client access to the server's OPC UA tag

Write CPU variable values directly into OPC UA tag (set OPC UA DataValue)

As of firmware version V3.0, S7-1500 CPUs offer, in addition to mapping tags, the possibility to write values directly into local OPC UA tag nodes of the server via the "OPC_UA_WriteList" instruction. Normally, the "OPC_UA_WriteList" instruction in the client program of the CPU is used to write values into OPC UA tags of a remote OPC UA server.

Advantage of using "OPC_UA_WriteList" in the server: In addition to the value, you can provide the OPC UA tag node with the following additional information:

- SourceTimestamp
- StatusCode

OPC UA provides a built-in "DataValue" data type. DataValue is a structure that holds the value (Value) as well as SourceTimestamp and StatusCode as more information to the value. The DataValue structure is only used by OPC UA services and you cannot write directly in the program of the CPU to the elements of this structure. Write access is only possible via a special use of the "OPC_UA_WriteList" instruction.

Application options

CPU variables cannot record a time stamp indicating when a value was last written to the CPU variable. If you map CPU tags and OPC UA tags via server interfaces, the OPC UA server does not therefore set the SourceTimestamp to the time when the CPU tag changed, but to the time when the value was "collected" in the server; e.g. by a read service or by sampling in the context of a subscription.

If you write DataValue directly with "OPC_UA_WriteList" into an OPC UA tag node, for example, you can provide a time stamp determined in the program as SourceTimestamp for the value.

How the OPC_UA_WriteList instruction works in principle for setting DataValues

The DataValue structure is, for example, modeled as UDT and a tag of this data type is transferred to the "OPC_UA_WriteList" instruction. The instruction then consistently transfers the elements of the tag to the OPC UA tag node.

The value of the "ConnectionHdl" instruction parameter defines the how the "OPC_UA_WriteList" works: "Normal" client instruction or instruction to write to local OPC UA tag nodes. OPC UA clients can read the value with more information in the latter case and evaluate it accordingly.

The principle is shown in the following figures, once with any client and once with an S7-1500 CPU as OPC UA client. In the case of the S7-1500 CPU client, the assignment of the DataValue elements to the corresponding instruction parameters of the OPC_UA_ReadList instruction is shown. You have full access to all elements of the DataValue structure.

The value of "ConnectionHdl" (-42) of the "OPC_UA_WriteList" instruction causes the server to write to local OPC UA tag nodes.

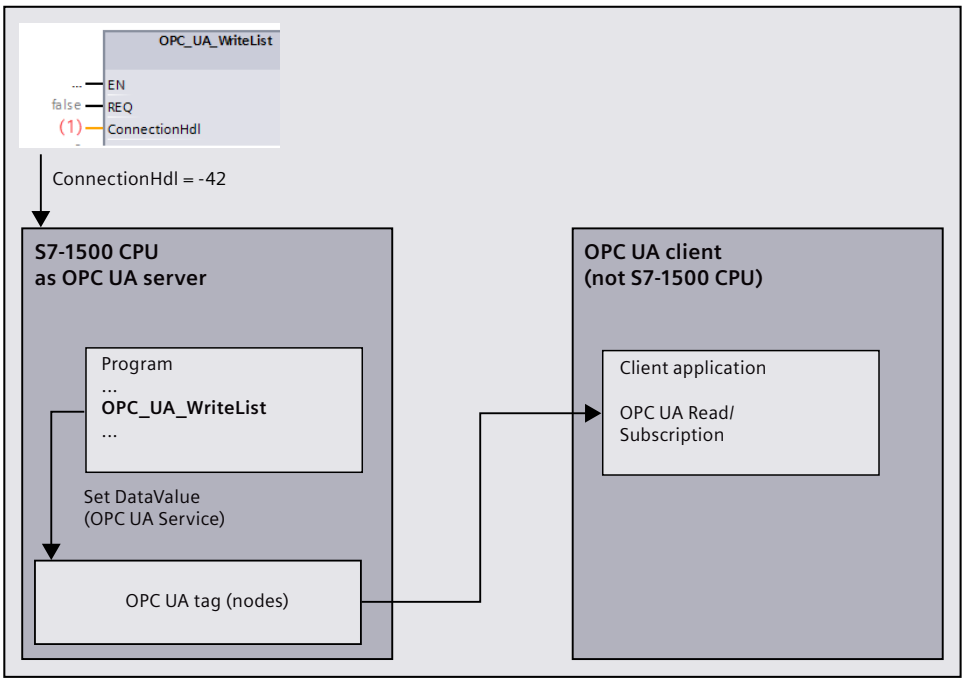


Figure 10-14 Set data value on local OPC UA tag of the server

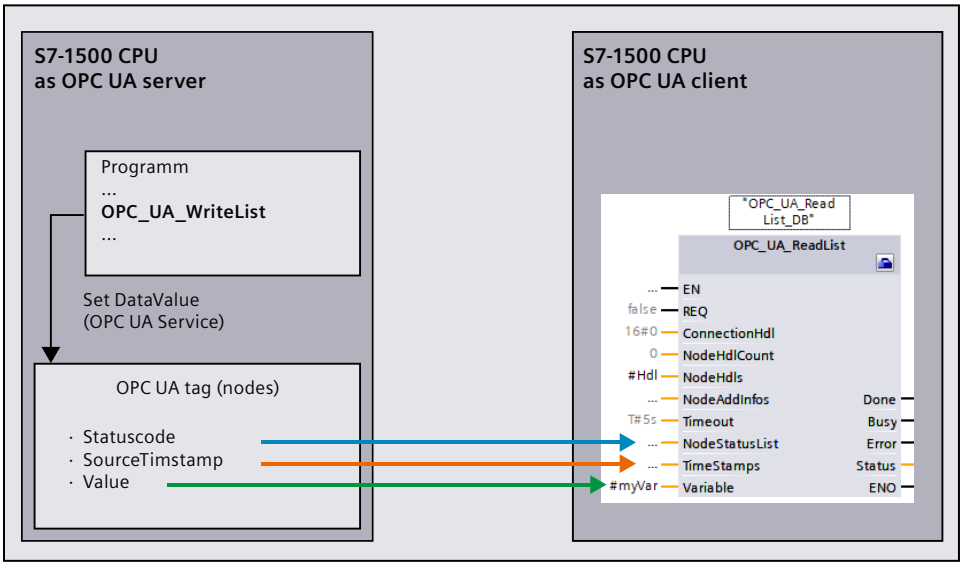


Figure 10-15 Client reads data value (OPC UA tag of the server of the S7-1500 CPU)

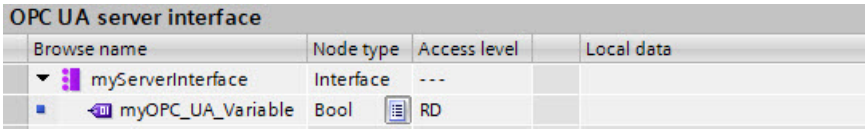
Other application options

If OPC UA clients register with an S7-1500 CPU for value changes (monitored items) in the context of a subscription and you supply the corresponding DataValue with both the value and the more information mentioned above, then changes to the more information can also trigger a notification.

Example: A binary value changes so fast that it falls back to its original value in the sampling interval (fast change TRUE > FALSE > TRUE). A change of the value is not detected. But the change of the time stamp is detected. Similarly, a notification can be triggered when the StatusCode changes - even without the value changing.

Constraints

- OPC UA clients are only allowed to read the OPC UA tag; the "AccessLevel" attribute for read/write permissions must be set accordingly for the OPC UA tag.
- Only OPC UA tags of the user-defined server interfaces can be set locally.
- In the user-defined server interface, there must be no mapping to CPU variables for the directly written OPC UA tags.



OPC UA server interface

Browse name	Node type	Access level	Local data
myServerInterface	Interface	---	
myOPC_UA_Variable	Bool	RD	

Figure 10-16 User-defined server interface

Details about the usage of the "OPC_UA_WriteList" instruction in the "Set OPC UA-DataValue" context can be found in the corresponding section of the communication instruction help.

10.3.2.2 Managing write and read rights

Enabling PLC tags and DB tags for OPC UA

OPC UA clients can have read and write access to PLC tags and DB tags if the tags are enabled for OPC UA (default setting). For an enabled tag the check box "Accessible from HMI/OPC UA" is activated.

You can change the default setting in the settings of the TIA Portal: Command "Settings > PLC programming > General" in "Options" menu. You will find the corresponding options in the "Block interface/data block elements" area.

The following example shows an array data block:

MyDB

Name	Data type	Accessible from HMI/OPC UA	Writable from HMI/OPC UA	Visible in HMI engineering
MyDB	Array[0..9] o...	☑	☑	☑
MyDB[0]	Int	☑	☑	☑
MyDB[1]	Int	☑	☑	☑
MyDB[2]	Int	☑	☑	☑
MyDB[3]	Int	☑	☑	☑
MyDB[4]	Int	☑	☑	☑
MyDB[5]	Int	☑	☑	☑
MyDB[6]	Int	☑	☑	☑
MyDB[7]	Int	☑	☑	☑
MyDB[8]	Int	☑	☑	☑
MyDB[9]	Int	☑	☑	☑

Figure 10-17 Enabling PLC tags and DB tags for OPC UA tags

This array can be read completely in one pass by OPC UA clients (see Addressing nodes (Page 159)). The check boxes at "Accessible from HMI/OPC UA" and "Writable from HMI/OPC UA" are activated for all the components of the array.

Result: OPC UA clients can both read and write these components.

Removing write rights

If you want to write-protect a tag, deselect the "Writable from HMI/OPC UA" option for that tag. This removes the write right for the OPC UA clients and HMI devices.
Result: Only read access by OPC UA clients and HMI devices is possible. OPC UA clients cannot assign values to this tag and therefore cannot influence execution of the S7 program.

Removing write and read rights

To write-protect and read-protect a tag, disable the "Accessible from HMI/OPC UA" option for that tag (checkbox not selected). This makes the OPC UA server remove the tag from its address space. OPC UA clients can no longer see that CPU tag.
Result: OPC UA clients and HMI devices can neither read nor write the tag.

Write and read rights of structures

If you remove the read or write right for the component of a structure, the structure or the data block cannot be written or read as a whole.
If you remove read and write rights for individual components of a PLC data type (UDT), the rights will also be removed from any data block based on that data type!

Visible in HMI engineering

The option "Visible in HMI Engineering" applies to Siemens engineering tools. If you disable the option "Visible in HMI Engineering" (check mark not set), you can no longer configure the tag in WinCC (TIA Portal).
The option does not have any effect on OPC UA.

Rules

- Only allow read access to PLC tags and tags of data blocks in STEP 7 if this is necessary for communication with other systems (controllers, embedded systems or MES).
  You should not enable other PLC tags.
- Only allow write access over OPC UA if write rights are genuinely necessary for specific PLC tags and tags of data blocks.
- If you have reset the "Accessible from HMI/OPC UA" option for all elements of a data block, the data block for an OPC UA client is no longer visible in the address space of the OPC UA server of the S7-1500 CPU.
- You can also prevent access to an entire data block centrally (see Managing write and read rights for a complete DB (Page 209)). This setting "overrules" the settings for the components in the DB editor.

More information

For information on how to coordinate write and read rights for CPU tags, refer to the section Coordinating write and read rights for CPU tags (Page 210).

10.3.2.3 Managing write and read rights for a complete DB

Hiding DBs or DB contents for OPC UA clients

You can easily prevent access to a complete data block by an OPC UA client.

With this option, the data of the corresponding DB, including instance DBs of function blocks, remains hidden for OPC UA clients.

In the default setting, data blocks can be read and written from OPC UA clients. You can change this default setting in the settings of the TIA Portal: Command "Settings > PLC programming > General" in "Options" menu. You will find the corresponding option in the "Default settings for new blocks" area.

Procedure

Proceed as follows to completely hide a data block for OPC UA clients or to protect a data block from write access from OPC UA clients:

1. Select the data block to be protected in the project tree.
2. Select the "Properties" shortcut menu.
3. Select the "Attributes" area.
4. Select/clear the "DB accessible from OPC UA" checkbox as required.

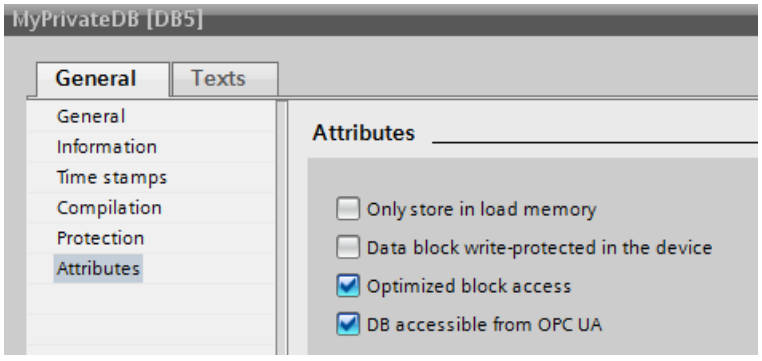


Figure 10-18 Hiding DBs or DB contents for OPC UA clients

NOTE

Effect on settings in the DB editor

If you hide a DB using the DB attribute described here, the settings for the components in the DB editor are no longer relevant; individual components can no longer be accessed or written.

Tip: Using the overview of all program blocks

If you are using multiple data blocks, it is appropriate to use the detailed overview of the "Program blocks" folder for selective activation or deactivation of the OPC UA accessibility.
Follow these steps:

1. Select the "Program blocks" folder in the project tree.
2. Select the "Overview" command in the "View" menu.
3. Select the "Details" tab.
   An overview of the blocks with their attributes is displayed.
4. Ensure that the "Data block accessible via OPC UA" column is selected.
5. Select only the data blocks that are to be accessible via OPC UA.

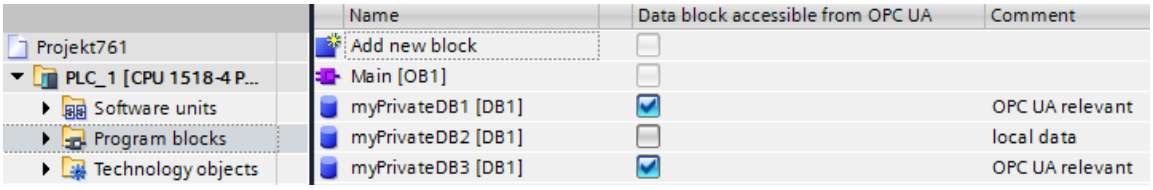


Figure 10-19 Overview of the program blocks

10.3.2.4 Coordinating write and read rights for CPU tags

Definition of write and read rights in the information model (OPC UA XML)

In the OPC UA information model, the attribute "AccessLevel" regulates access to tags.
AccessLevel is defined bit by bit:
Bit 0 = CurrentRead and Bit 1 = CurrentWrite. The meaning of the bit combinations is as follows:

- AccessLevel = 0: no access
- AccessLevel = 1: read only
- AccessLevel = 2: write only
- AccessLevel = 3: read+write

Example of the assignment of write and read rights (read+write)

```
<UAVariable NodeId="ns=3;s=&quot;Data_block_2&quot;.&quot;Static_1&quot;"
BrowseName="3:Static_1"
ParentNodeId="ns=3;s=&quot;Data_block_2&quot;"
DataType="INT"
AccessLevel="3">
    <DisplayName>Static_1</DisplayName>
```

Definition of write and read rights in STEP 7

When you define tags, you specify the access rights using the properties "Accessible from HMI/OPC UA" and "Writable from HMI/OPC UA".

Example of the assignment of write and read rights

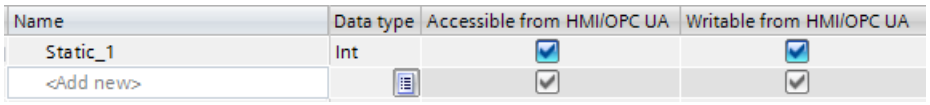


Figure 10-20 Example of the assignment of write and read rights

Interaction between write and read rights

If you have imported an OPC UA server interface and AccessLevel attributes are set in this OPC UA XML file, the write and read rights are defined by the following rule: The least extensive access rights for each setting apply.

Example

- AccessLevel = 1 (read only) in the OPC UA server interface
- Both "Accessible from HMI/OPC UA" and "Writable from HMI/OPC UA" are selected in the PLC tag table

Result: This tag can only be read.

Rules

If write rights are required:

- AccessLevel = 2 oder 3
- "Writable from HMI/OPC UA" enabled

If read rights are required:

- AccessLevel = 1 (AccessLevel 3 is also possible, but misleading. The settings suggests that an OPC UA client has write and read rights)
- "Accessible from HMI/OPC UA" enabled, "Writable from HMI/OPC UA" disabled

If neither read nor write rights are to be granted (no access):

- AccessLevel = 0
- "Accessible from HMI/OPC UA" disabled

  Only one of the two conditions needs to be met to block all access. In this case, review whether the tag in the OPC UA server interface is actually necessary at all.

Access table

"Accessible from HMI/OPC UA" must be set if access over OPC UA is to be possible at all. "Writable from HMI/OPC UA" must be set to allow an OPC UA client to write a tag / DB element.

Please see the table for the resulting access right.

Table 10-2 Access table

OPC UA XML	**STEP 7 (TIA Portal), for example tag table**		
AccessLevel	**Accessible from HMI/OPC UA**	**Writable from HMI/OPC UA**	**Resulting access right**
0	x	x	No access
x	0	x	No access

OPC UA XML	STEP 7 (TIA Portal), for example tag table		
AccessLevel	**Accessible from HMI/OPC UA**	**Writable from HMI/OPC UA**	**Resulting access right**
1	Enabled	x	Read only
2	Enabled	Disabled	No access
3	Enabled	Disabled	Read only
2	Enabled	Enabled	Write only
3	Enabled	Enabled	Read+write

(x = don't care)

10.3.2.5 Consistency of CPU tags

"AccessLevelEx" attribute extends access properties

As of firmware version V2.6, the OPC UA server of the S7-1500 CPU supports not only the attribute "AccessLevel" (see Coordinating write and read rights for CPU tags (Page 210)) but also the attribute "AccessLevelEx" which, in addition to the already explained bits for read access and write access, provides information on the consistency of a OPC UA tag. The new attribute was introduced with version V1.04 of the OPC UA specification (Part 3, Address Space Model).

Reading consistency properties

In the OPC UA information model of the OPC UA server, the attribute "AccessLevel" defines the access.

AccessLevelEx is defined bit by bit; in this case the relevant bits are:

- Bit 0 = CurrentRead
- Bit 1 = CurrentWrite
- Bits 2 to 7 are not relevant for the OPC UA server of an S7-1500 CPU

The meaning of the bit combinations is explained in the section on read and write rights.

The following bits for consistency are also added:

- Bit 8 = NonatomicRead; the bit is set if the tag cannot be read consistently. For read consistency of tags, bit 8=0.
- Bit 9 = NonatomicWrite; is set if the tag cannot be written consistently. For write consistency of tags, or if no write access is granted, bit 9=0.

Examples

An OPC UA tag (structure) is readable and writable; but inconsistent for reading and writing access.

Consequently: Bits 0, 1, 8 and 9 are set: AccessLevelEx = "771" (1+2+256+512).

Another structure is read-only.

Consequently: Bits 0 and 8 are set, bit 1 and bit 9 are not set: AccessLevelEx = "257" (1+0+256+0).

Handling of the attribute in the server

The "AccessLevelEx" attribute is only available in the OPC UA server. The attribute is not present in a node set file (XML export file).
However, the attribute "AccessLevel", which is exported, includes the information from "AccessLevelEx", see next section.

Export

During XML export of the standard SIMATIC server interface, the server sets the "AccessLevel" attribute, which was expanded to 32 bits in V1.04 compared to V1.03, to the value of the "AccessLevelEx" attribute.

Import

When importing a node set file (e.g. from an export of a server interface), the S7-1500 CPU sets the attribute "AccessLevelEx" according to its own estimate of the consistency of the imported data type, see next section. The imported value is ignored.

Consistency of data types at the server interface

The consistency of tags ("atomicity" in the language usage of OPC UA) within a program cycle of an S7-1500 CPU is ensured at the nodes of the server interface for the following data types:

- BOOL, BYTE, WORD, DWORD, LWORD
- SINT, INT, LINT, DINT, USINT, UINT, ULINT, UDINT
- REAL, LREAL
- DATE, LDT, TIME, LTIME, TIME_OF_DAY, LTIME_OF_DAY, S5TIME
- CHAR, WCHAR
- System data types and hardware data types that are based on the above-mentioned data types are also consistent.
  Example: HW_ANY, derived from UINT (UInt16).

Tip: If you browse in the address space of the S7-1500 CPU (e.g. with the OPC UA Client UaExpert), you can find the consistent data types under Types > BaseDataType > Enumeration/Number/String.

Tags of the following data types are **not** consistent (in the language usage of OPC UA: "nonatomic"):

- SIMATIC structures are generally not consistent. This means that all tags which, for example, have unknown structures or a UDT data type are not consistent.
- System data types such as DTL, IEC_Counter, IEC_TIMER, etc. are data types that are derived from structures.
- Strings (Array of Char) are not consistent.

Tip: If you browse in the address space of the S7-1500 CPU (e.g. with the OPC UA Client UaExpert), you can find data types based on structures under Types > BaseDataType > Structure.

10.3.2.6 Write accesses to OPC UA variables from S7-1500 Motion Control.

In addition to the consistency of the data types, the CPU examines the variables of the technology objects for plausibility and validity.

If an OPC UA client writes an invalid or implausible value to a variable, the original value is retained in the variable of the technology object.
Despite an unsuccessful write access, the status "Good" is output.

Example 1

Interpolation type of the cyclic cam
The variable "Cam_1".InterpolationSettings.InterpolationMode is of type INT, but may only assume the values 1...2.
If you want to change the variable using OPC UA to an invalid value, for example, 3, then the status code "Good" is output, but the variable is not changed.

Example 2

Position the software limit switch at a positioning axis
The position of the positive HW limit switch must be more positive than the position of the negative SW limit switch.
"PosAxis_1".PositionLimits_SW.MaxPosition > "PosAxis_1".PositionLimits_SW.MinPosition
If you want to change the variable using OPC UA to a value that does not meet this condition, then the status code "Good" is output, but the variable is not changed.

Which values are valid for variables of the technology objects can be looked up in the documentation of the technology objects (https://support.industry.siemens.com/cs/ww/en/view/109751049).

10.3.2.7 Accessing OPC UA server data

High performance in line with application

OPC UA is designed for the transfer of a high volume of data within a short period of time. You can increase the performance significantly if you do not access individual PLC tags, but rather read and write arrays and structures as a whole.
It is fastest to access arrays. Therefore, you should combine the data for OPC UA clients into arrays.

Recommendations for access to the OPC UA server by the OPC UA client

- For one-off or infrequent data access, use standard read/write access.
- For cyclic access to small amounts of data (up to ca. every 5 seconds), use subscriptions. Optimize the settings for the smallest publishing interval and the smallest sampling interval at the OPC UA server.
- If you access specific tags regularly (recurring access), you should use the functions "RegisteredRead" and "RegisteredWrite".

Allow a greater communication load for the PLC by increasing the value for "Cycle load due to communication". Make sure that your application still works properly with the changed settings.

Procedure for creating an array DB

You can create arrays for example in global data blocks, in the instance data block of a function block or as an array DB . The following sections describe how to create an Array-DB.

To create a data block with an array (array data block), follow these steps:

1. Select the CPU with the OPC UA server in the project tree.
2. Double-click "Program blocks".
3. Double-click "Add new block".
4. Click "Data block".
5. Select a unique name for the data block and accept the name that is already entered.
6. Select the "Array DB" entry from the drop-down list for "Type".
7. Select the data type for the individual components of the array from the drop-down list for "Array data type".
8. Enter the high limit of the array for "Array limit".
9. Click "OK".

10.3.2.8 MinimumSamplingInterval attribute

MinimumSamplingInterval attribute of tags

In addition to "Value", "DataType" and "AccessLevel", you can also set the "MinimumSamplingInterval" attribute for a tag in the XML file that represents the server address space.

The attribute specifies how fast the server can sample the tag value.

The OPC UA server of the S7-1500 CPU handles the values for MinimumSamplingInterval as follows:

- Negative values and values greater than 4294967 are set to -1; this means: The minimum sampling rate is indeterminate. The server does not specify how fast the tag value can be sampled.
- Decimal numbers are rounded to three decimal places.

10.3.2.9 Export OPC UA XML file

Generating an OPC UA export file

The OPC Foundation has specified a standard XML-based format for describing information models. It allows the information model of an OPC UA server to be provided to a client in advance, or information models can be downloaded to an OPC UA server. A file in this format is called a node set file because it describes an information model as a set of nodes.

With STEP 7 (TIA Portal), you can easily export the standard SIMATIC information model of the S7-1500 CPU as a server to an OPC UA XML file (node set file); including the following elements that you have enabled for OPC UA:

- CPU variables (PLC tags and DB elements)
- Function blocks with their inputs/outputs

Elements that exist in the CPU but are not used in the program do not appear in the OPC UA XML file after the export. Examples of such unused elements are:

- UDTs that are not mapped to a data block
- Function blocks with inputs/outputs without assigning inputs/outputs to CPU tags

You use the OPC UA XML file for the offline configuration of an OPC UA client; it is structured according to the OPC UA specification and acts as a standard SIMATIC server interface.
To create and export the OPC UA XML file, follow these steps:

1. Select the CPU. Click on the CPU symbol (for example in the network view).
2. Click "General > OPC UA > Server > Export" in the properties of the CPU.
3. Click "Export OPC UA XML file".
4. Select the directory in which you want to save the export file.
5. Select a new name for the file or keep the name that is already entered.
6. Click "Save".

NOTE

As of STEP 7 (TIA Portal) V15.1, server methods are contained in the OPC UA export file (node set) together with their input and output parameters.

Exporting all array elements separately

If the "Export all array elements as separate nodes" option is selected in the CPU properties under "OPC UA > Server > Export", the OPC UA XML file contains all elements of arrays as individual XML elements. In addition, the arrays themselves are each described in an XML element in the XML file.
If an array contains many array elements, the XML file can be very large.

Tip

The following FAQ contains a converter with which you can convert the export file into CSV format. You then obtain a list of the tags of the CPU that can be accessed by OPC UA.
You can find the FAQ on the Internet (https://support.industry.siemens.com/cs/ww/en/view/109742903).

10.3.3 Configuring the OPC UA server

10.3.3.1 Enabling the OPC UA server

Requirement

- If you use **certificates** for secured communication, e.g. HTTPS, Secure OUC, OPC UA, make sure that the modules involved have the **current time of day and the current date**. Otherwise, the modules evaluate the used certificates as invalid and secure communication does not work.
- You have acquired a runtime license for the operation of the OPC UA functions, see License for OPC UA (Page 235).

Commissioning an OPC UA server

By default, the OPC UA server of the CPU is not enabled for reasons of security: OPC UA clients have neither write nor read access to the S7-1500 CPU.

Follow these steps to activate the OPC UA server of the CPU:

1. Select the CPU. Click on the CPU symbol (for example in the network view).
2. Click "OPC UA > Server" in the properties of the CPU.
3. Activate the OPC UA server of the CPU.
4. Confirm the security notes.
5. Go to the CPU properties, select "Runtime licenses" and set the runtime license acquired for the OPC UA server.
6. Compile the project.
7. Download the project to the CPU.
   The OPC UA server of CPU now starts.

Settings remain stored

If you have already enabled the server and made settings, those settings are not lost if the server is disabled. The settings are saved as before and are available when you enable the server again.

Application name

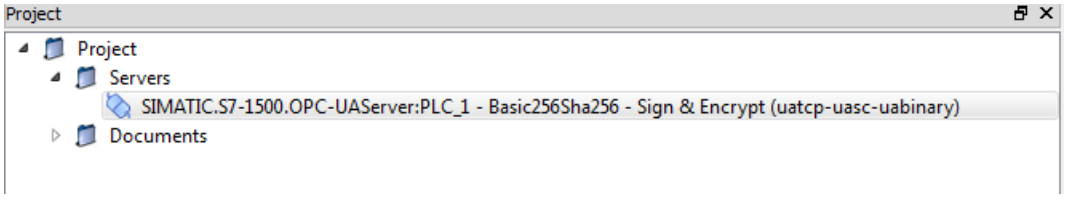
The application name is the name of the OPC UA application and applies to the server and the client. The name is displayed under "OPC UA > General":

- The default for the application name is: "SIMATIC.S7-1500.OPC-UA.Application:PLC_1".
- The default consists of "SIMATIC.S7-1500.OPC-UA.Application:" and the name of the CPU selected under "General > Product information > Name", in this case "PLC_1".
- The OPC UA server uses this application name to identify itself to a communication partner (OPC UA client), for example, when an OPC UA client uses the discovery service to detect accessible servers.
- The displayed application name uses the OPC UA client of the CPU when connecting to an OPC UA Server. This means that the CPU enters this application name automatically as "ApplicationName" for the instruction "OPC_UA_Connect" (tag of type "OPC_UA_SessionConnectInfo" at the parameter "SessionConnectInfo" of the instruction "OPC_UA_Connect").

  When you program the instruction "OPC_UA_Connect" you must therefore assign an empty string to the "ApplicationName". You can use the application name, for example, to identify the client and its sessions (SessionNames) for diagnostic purposes.

If you have activated the server, you can also use a different name that is meaningful in your project and that fulfills the requirements of your project, e.g. for worldwide uniqueness.

The example below originates from UaExpert:



Changing the application name

To change the application name, follow these steps:

1. Select the CPU. Click on the CPU symbol (for example in the network view).
2. Click "OPC UA > General" in the properties of the CPU.
3. Enter a meaningful name.

Please note that the application name is also entered on the certificate (Subject Alternative Name) and you may have to generate an existing certificate again after changing the application name.

10.3.3.2 Access to the OPC UA server

Server addresses

The OPC UA server of the S7-1500 CPU can be reached over all integrated PROFINET interfaces of the CPU (firmware V2.0 and higher).

Direct access to the OPC UA server of the CPU over the backplane bus of the automation system is not possible via CPs under the following conditions:

- Configuration with TIA Portal Version V16 or higher, S7-1500 CPU firmware version 2.8 or higher and CP 1543-1 firmware version V2.2 or higher.
  For configuration, see Access to OPC UA applications (Page 156).

Direct access to the OPC UA server of the CPU over the backplane bus of the automation system is not possible via CMs.

With SIMATIC S7 1500 SW controllers, access to the OPC UA server is possible via PROFINET interfaces that are assigned to the software PLC.

Additional access options of SW controllers are described in the following application example: Internal and external OPC UA connection via the virtual Ethernet interface of the software controller V2.5 or higher
(https://support.industry.siemens.com/cs/ww/en/view/109760541).

Example for URLs (Uniform Resource Locator) that can be used to set up connections to the OPC UA server of the CPU:

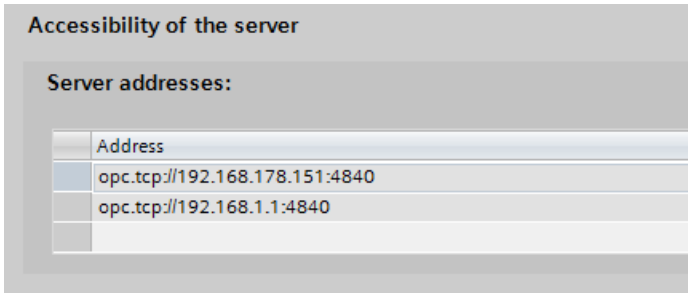


Figure 10-21 Display of the server addresses

The URLs are structured as follows:

- Protocol identifier "opc.tcp://"
- IP address
  - 192.168.178.151
    The IP address at which the OPC UA server can be accessed from the Ethernet subnet 192.168.178.
  - 192.168.1.1
    The IP address at which the OPC UA server can be accessed from the Ethernet subnet 192.168.1.
- TCP Port number
  - Default: 4840 (standard port)
    The port number can be changed under "OPC > UA > Server > Port".

Dynamic IP addresses

In the example below, the IP address for the PROFINET interface [X2] has not yet been specified.

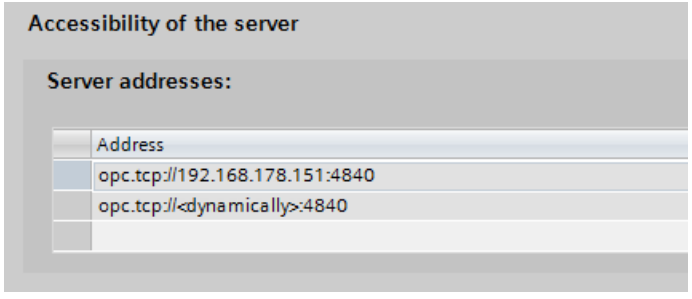


Figure 10-22 Display of the server addresses with dynamic IP address

The placeholder "<dynamically>" appears in the table.
The IP address of this PROFINET interface is set later on the device, e.g. via the display of the CPU.

Activating the standard SIMATIC Server interface

If the "Enable standard SIMATIC server interface" option is selected, the OPC UA server of the CPU provides the enabled PLC tags and server methods to the clients, as was specified by SIEMENS in the self-defined namespace.

This option is selected in the default setting.

Leave the option selected so that OPC UA clients can automatically connect to the OPC UA server of the CPU and exchange data.

If you do not select this option, you must add the server interface by entering the "OPC UA communication" entry in the project tree. This interface is then used as OPC UA server interface, see OPC UA server interface configuration (Page 236).

NOTE

General device information is readable even with deactivated standard SIMATIC server interface

Even if you disable the standard SIMATIC server interface, OPC UA clients can read general device information about the OPC UA server of the CPU.

Examples of such device information: DeviceManual, DeviceRevision, OrderNumber. In this case, however, all objects of the application program remain invisible to clients.

If you want to prevent that this device information is not visible, you have to disable the OPC UA server of the CPU.

10.3.3.3 General settings of the OPC UA server

TCP port for OPC UA

By default, OPC UA uses TCP port 4840. You can, however, select a different port. Entries from 1024 to 49151 are possible. You must, however, make sure that there are no conflicts with other applications. OPC UA clients must use the selected port when establishing a connection.

In the example below, port 48400 is selected:

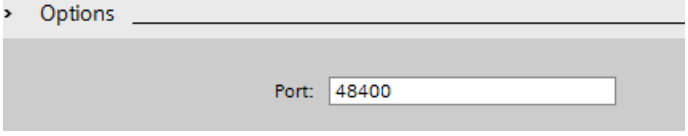


Figure 10-23 TCP port for OPC UA

An overview of the supported protocols and the port numbers used by the S7-1500 CPUs can be found in section Communications protocols and port numbers used for Ethernet communication (Page 30).

Settings for sessions

- Maximum timeout for sessions
  In this field, you specify the maximum time period before the OPC UA server closes a session without data exchange.
  Possible values between 1 and 600000 seconds.
- Maximum number of OPC UA sessions
  In this field, you specify the maximum number of sessions the OPC UA server starts and simultaneously operates.
  The maximum number of sessions is dependent on the performance capability of the CPU. Each session ties up resources.

Maximum number of registered nodes

In this field, you specify the maximum number of nodes the OPC UA server registers.
The maximum number of registered nodes depends on the capacity of the CPU and is displayed when you configure the field content (place cursor in field). Each registration ties up resources.

NOTE

No error message following attempt to register more than the configured maximum number of registrable nodes

If a client tries to register more nodes during runtime than the configured maximum number, the server of the S7-1500 CPU only registers the configured maximum number. Starting from the configured maximum number of registrable nodes, the server returns the regular string node IDs unchanged to the client so that the speed advantage gained by registration for these nodes is lost. The client does not receive an error message.

When configuring, make sure you have a sufficient reserve by taking into account the maximum number of nodes that can be registered (for example, using the technical data of the CPU).

Additional information

Details on which ports are used by the various services for data transfer via TCP and UDP, and what are the points to note when using routers and firewalls can be found in the FAQ (https://support.industry.siemens.com/cs/ww/en/view/8970169).

Backward compatible data type definitions according to OPC UA specification ≤ V1.03

The OPC UA specification (<= V1.03) defines mechanisms in order to read out data type definitions, for example for user-defined structures (UDTs), from a server by means of the TypeDictionaries.
In the OPC UA server properties of the CPU, you can set whether the CPU generates these backward compatible data type definitions according to the OPC UA specification ≤ V1.03 for the standard SIMATIC server interface or not.
Because TypeDictionaries are complex and result in large OPC UA XML files (server interfaces) which the client has to interpret, a simpler solution was introduced with OPC UA Specification

V1.04 (attribute "DataTypeDefinition" at the data type node). If your client supports the OPC UA specification V1.04 or higher, then disable the option.

Advantages of the data type definitions according to OPC UA specification as of V1.04:

- The server starts faster
- The memory is used more efficiently
- The "Browse" function is faster

10.3.3.4 Settings of the server for subscriptions

Subscription instead of cyclic queries

An alternative to cyclic queries for a PLC tag (polling) is to monitor this value. Use a Subscription: The server informs the client if the value of PLC tags changes. See "The OPC UA client".

One server usually monitors a large number of PLC values. For this reason, the server sends notifications to the client at regular intervals containing the new values of the PLC tags.

Advantages of subscriptions:

- The server starts faster
- The memory is used efficiently

How frequently does the server send notifications?

When a Subscription is set up, the OPC UA client specifies the intervals at which it wants to be sent the new values in the event of changes. To limit the communication load through OPC UA, set a minimum interval for the messages. For this purpose, use the parameters for the minimum publishing interval and the minimum sampling interval.

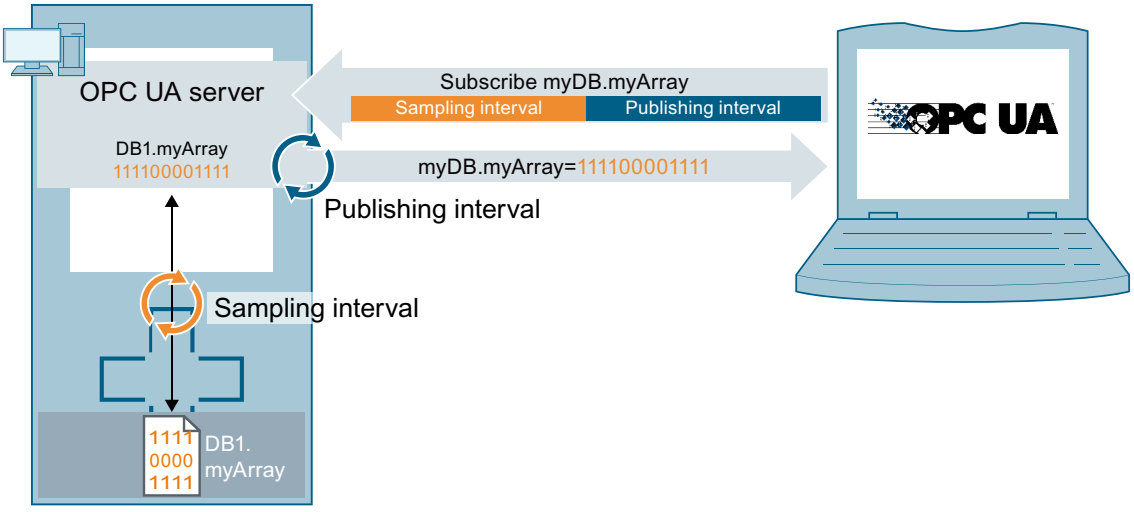


Figure 10-24 Principle of a subscription

Minimum publishing interval

With "Minimum publishing interval", you set the time intervals at which the server sends a message to the client with the new values in the event of changes.

250 ms is used as the "Minimum sampling interval" in the figure below. The value 200 ms is entered as the "Minimum publishing interval".

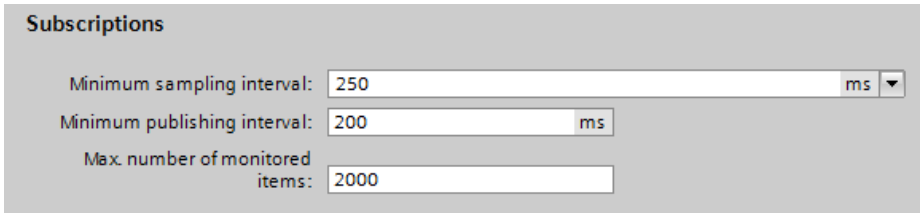


Figure 10-25 Subscription settings

In the example, following a value change the OPC UA server will send a new message every 200 ms if the OPC UA client requests an update.
If the OPC UA client requests an update every 1000 ms, the OPC UA server will only send a message with the new values once every 1000 ms (one second).
If the OPC UA client requests an update every 100 ms, the server will still only send a message every 200 ms (minimum publishing interval).

Minimum sampling interval

With "Minimum sampling interval", you set the time intervals at which the OPC UA server records the value of a CPU tag and compares it with the previous value to detect any changes.
If the sampling interval is selected smaller than the publishing interval and an OPC UA client requests a high sampling rate for certain PLC tags, two or more values may be measured during each publishing interval.
In this case, the OPC UA server writes the value changes into the queue and sends all value changes to the client after the completion of the publishing interval. If more value changes occur in the publishing interval than fit in the queue, the OPC UA server overwrites the oldest values (depending on the set "Discard Policy" of the client subscribing to the data, the option "Discard Oldest" has to be activated in this case). The most recent values are sent to the client.

Maximum number of monitored elements (monitored items)

In this field, you specify the maximum number of elements that the OPC UA server of the CPU simultaneously monitors for a value change.
The monitoring ties up resources. The maximum number of monitored elements is dependent on the utilized CPU.

More information

Information about the system limits of the OPC UA server of the S7-1500 CPUs (firmware V2.0 and V2.1) regarding subscriptions, sampling intervals and publishing intervals can be found in the following FAQ (https://support.industry.siemens.com/cs/ww/en/view/109755846).
When using subscriptions, certain status codes of errors provide information on the error that occurred. For information on causes and remedies for status codes of OPC UA client that

appear, see the list of error codes in the online help of STEP 7 (TIA Portal) or in the following FAQ (https://support.industry.siemens.com/cs/ww/en/view/109755860).
The rules for subscriptions are available in section Rules for subscriptions (Page 336).
You will find information about the diagnostics of subscriptions in the section Subscription diagnostics (Page 299).

10.3.3.5 Handling client and server certificates

A secure connection between the OPC UA server and an OPC UA client is only established when the server can prove its identity to the client. This is done with the server certificate.

Certificate of the OPC UA server

When you have activated the OPC UA server and have confirmed the security prompts, STEP 7 automatically generates the certificate for the server and saves it in the local certificate directory of the CPU. You can view and manage this directory with the local certificate manager of the CPU (exporting or deleting certificates).
The figure below shows the local certificate manager of the CPU with the automatically generated certificate for the OPC UA server:

Figure 10-26 Local certificate manager of the CPU

Alternatively, you can also generate a server certificate yourself.
The certificate of the server is transferred from the server to the client during establishment of a connection. The client checks the certificate.

The client user decides whether the server certificate is to be trusted.

The user at the client side now has to decide whether the server certificate is to be trusted. If the user trusts the server certificate, the client stores the server certificate in its directory containing the trusted server certificates.

The following example shows a dialog of the client "UA Sample Client". When the user clicks the "Yes" button, the client trusts the server certificate:

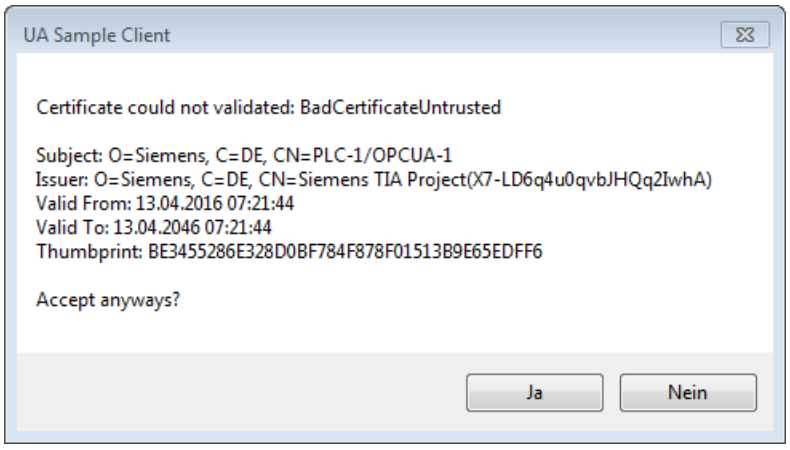


Figure 10-27 Dialog of the client "UA Sample Client"

Where does a client certificate come from?

Client of the S7-1500

If you are using the OPC UA client of an S7-1500 CPU (OPC UA client enabled), you can create certificates for these clients with STEP 7 V15 and higher.

1. In the project tree, select the CPU you want to use as a client.
2. Double-click "Device configuration".
3. In the properties of the CPU, click "Protection & Security > Certificate manager".
4. Double-click "<Add new>" in the "Device certificates" table.
   STEP 7 opens a dialog.
5. Click the "Add" button.
6. Select the "OPC UA client" entry from the "Usage" list.
   Note:
   The IP addresses under which the CPU can be accessed in your system must be entered under "Subject Alternative Name (SAN)".
   You must therefore configure the IP interfaces of the CPU before you generate a client certificate.
7. Click "OK".
   STEP 7 now shows the client certificate in the "Device certificates" table.
8. Right-click this line and select the "Export certificate" entry from the shortcut menu.
9. Select a directory where you will store the client certificate.

Clients of other manufacturers

When you use UA clients from manufacturers or the OPC Foundation, a client certificate is generated automatically during installation or upon the first program call. You have to import these certificates via the global certificate manager in STEP 7 and use them for the corresponding CPU (as shown above).

When you program an OPC UA client yourself, you can have the certificates generated by the program; see the section "Instance certificate for the client". Alternatively, you can generate

certificates with tools, for example with OpenSSL or the certificate generator of the OPC Foundation:

- The procedure for OpenSSL is described here: "Generating PKI key pairs and certificates yourself".
- Working with the certificate generator of the OPC Foundation is described here: "Creating self-signed certificates".

Announcing client certificates to the server

You need to send client certificates to the server to allow a secure connection to be established.

To do this, follow these steps:

1. Select the "Use global security settings for certificate manager" option in the local certificate manager of the server. This makes the global certificate manager available.
   You will find this option under "Protection & Security > Certificate manager" in the properties of the CPU that is acting as server.
   If the project is not yet protected, select "Security settings > Settings" in the STEP 7 project tree, click the "Protect this project" button and log on.
   The "Global security settings" item is now displayed under "Security settings" in the STEP 7 project tree.
2. Double click "Global security settings".
3. Double click "Certificate manager".
   STEP 7 opens the global certificate manager.
4. Click on the "Trusted certificates" tab.
5. Right-click in the tab on a free area (not on a certificate).
6. Select the "Import" command from the shortcut menu.
   The dialog for importing certificates is displayed.
7. Select the client certificate that the server is to trust.
8. Click "Open" to import the certificate.
   The certificate of the client is now contained in the global certificate manager.
   Note the ID of the client certificate just imported.
9. Click the "General" tab in the properties of the CPU that is acting as server.
10. Click "OPC UA > Server > Security > Secure Channel".
11. Scroll down in the "Secure Channel" dialog to the section "Trusted clients".
12. Double-click in the table on the empty row with "<add new>". A browse button is displayed in the row.
13. Click this button.
14. Select the client certificate that you have imported.
15. Click the button with the green check mark.
16. Compile the project.
17. Load the configuration onto the S7-1500 CPU.

Result:
The server now trusts the client. If the server certificate is also considered trusted, the server and client can establish a secure connection.

Accepting client certificates automatically

When you select the option "Automatically accept all client certificates during runtime" (below the "Trusted clients" list), the server automatically accepts all client certificates.

NOTICE
Setting after commissioning In order to avoid security risks, deactivate the "Automatically accept client certificates during runtime" option again after commissioning.

Configuring security settings of the server

The figure below shows the available server security settings for signing and encrypting messages.

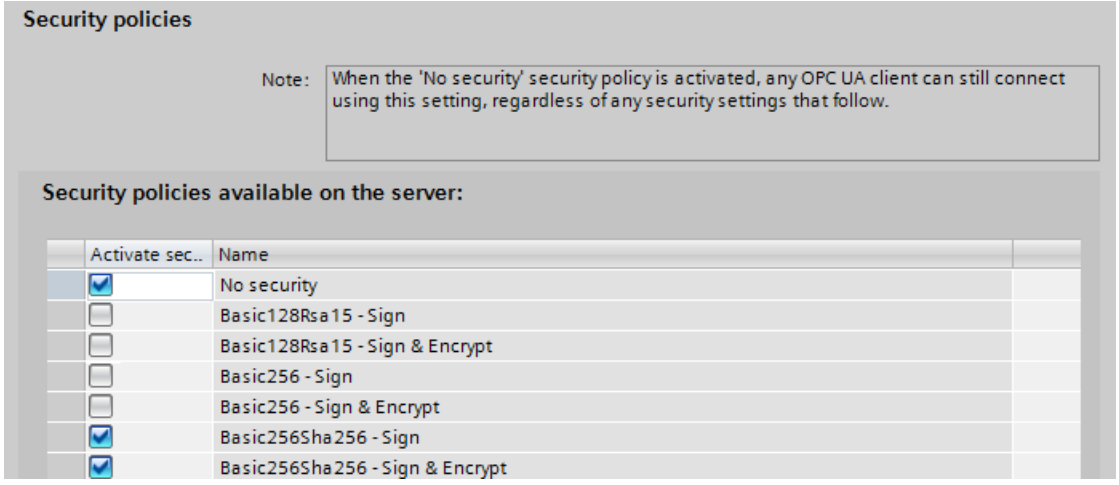


Figure 10-28 Configuring security settings of the server

By default, a server certificate is created that uses SHA256 signing. The following security policies are enabled:

- None
  Unsecured end point

NOTE

Disabling security policies you do not want

If you have enabled all security policies in the secure channel settings of the S7-1500 OPC UA server (default setting) – thus, also the end point "None" (no security) – unsecured data traffic (neither signed nor encrypted) between the server and client is also possible. The identity of the client remains unknown with "No security". Each OPC UA client can then connect to the server irrespective of any subsequent security settings.

When configuring the OPC UA server, make sure that only security policies that are compatible with the security concept of your machine or plant are selected. All other security policies should be disabled.

Recommendation: If possible, use the setting "Basic256Sha256".

- Basic128Rsa15 -Sign
  Insecure end point, supports a series of algorithms that use the hash algorithm RSA15 and 128-bit encryption.
  This endpoint protects the integrity of the data through signing.
- Basic128Rsa15 -Sign & Encrypt
  Secure endpoint, supports a series of algorithms that use the hash algorithm RSA15 and 128-bit encryption.
  This endpoint protects the integrity and confidentiality of the data through signing and encrypting.
- Basic256Rsa15 -Sign
  Secure endpoint, supports a series of algorithms that use the hash algorithm RSA15 and 256-bit encryption.
  This endpoint protects the integrity of the data through signing.
- Basic256Rsa15 -Sign & Encrypt
  Secure endpoint, supports a series of algorithms that use the hash algorithm RSA15 and 256-bit encryption.
  This end point protects the integrity and confidentiality of the data through signing and encrypting.
- Basic256Sha256 - Sign
  Secure endpoint, supports a series of algorithms for 256-bit hashing and 256-bit encryption.
  This endpoint protects the integrity of the data through signing.
- Basic256Sha256 - Sign & Encrypt
  Secure endpoint, supports a series of algorithms for 256-bit hashing and 256-bit encryption.
  This endpoint protects the integrity and confidentiality of the data through signing and encryption.

- Aes256_Sha256_RsaPss - Sign
  Secure endpoint, supports a range of algorithms for 256-bit encryption and 256-bit hashing. All certificates must use at least Sha256 signatures. This endpoint protects the integrity of the data by signing it.
  For high security requirements. PKI infrastructure required.
- Aes256_Sha256_RsaPss - Sign & Encrypt
  Secure endpoint, supports a range of algorithms for 256-bit encryption and 256-bit hashing. All certificates must use at least Sha256 signatures. This endpoint protects the integrity and confidentiality of data by signing and encrypting it.
  For high security requirements. PKI infrastructure required.

To enable the security setting, click the check box in the relevant line.

NOTE

If you use the settings "Basic256Sha256 -Sign" and "Basic256Sha256 -Sign & Encrypt", the OPC UA server and OPC UA clients must use "SHA256"-signed certificates.

For the settings "Basic256Sha256 -Sign" and "Basic256Sha256 -Sign & Encrypt", the certificate authority of STEP 7 automatically signs the certificates with "SHA256".

"No Security" security policy and authentication via user name and password

You can set the following combination:
Security policy = "No Security" and authentication via user name and password.

- The OPC UA server of the S7-1500 supports this combination. OPC UA clients can connect and encrypt the authentication data or not.
- OPC UA client of the S7-1500 CPU also supports this combination: However, in runtime it only connects if it can send the authentication data encrypted via cable!

10.3.3.6 Generating server certificates with STEP 7

The description below shows the procedure for generating new certificates with STEP 7 and applies in principle to various uses of the certificates. STEP 7 sets the appropriate purpose - in this case "OPC UA Client & Server" - depending on which area of the CPU properties is used to start the following dialog.
Recommendation: To use the full functionality for the security of the OPC UA server, use the global security settings.
The global security settings are enabled in the CPU properties under "Protection & Security > Certificate manager".

Customizing server certificates

STEP 7 automatically generates a certificate for the OPC UA server of the S7-1500 when you activate the server (see "Activating the OPC UA server (Page 216)"). In the process STEP 7 uses the default values for the parameters of the certificate. If you want to change the parameters, follow these steps:

1. Click the Browse button under "General > OPC UA > Server > Security > Secure channel > Server certificate" in the properties of the CPU. A dialog is displayed that shows the certificates available locally.
2. Click the "Add" button.
3. The dialog for generating new certificates is displayed (figure below). The values for an example are already entered:

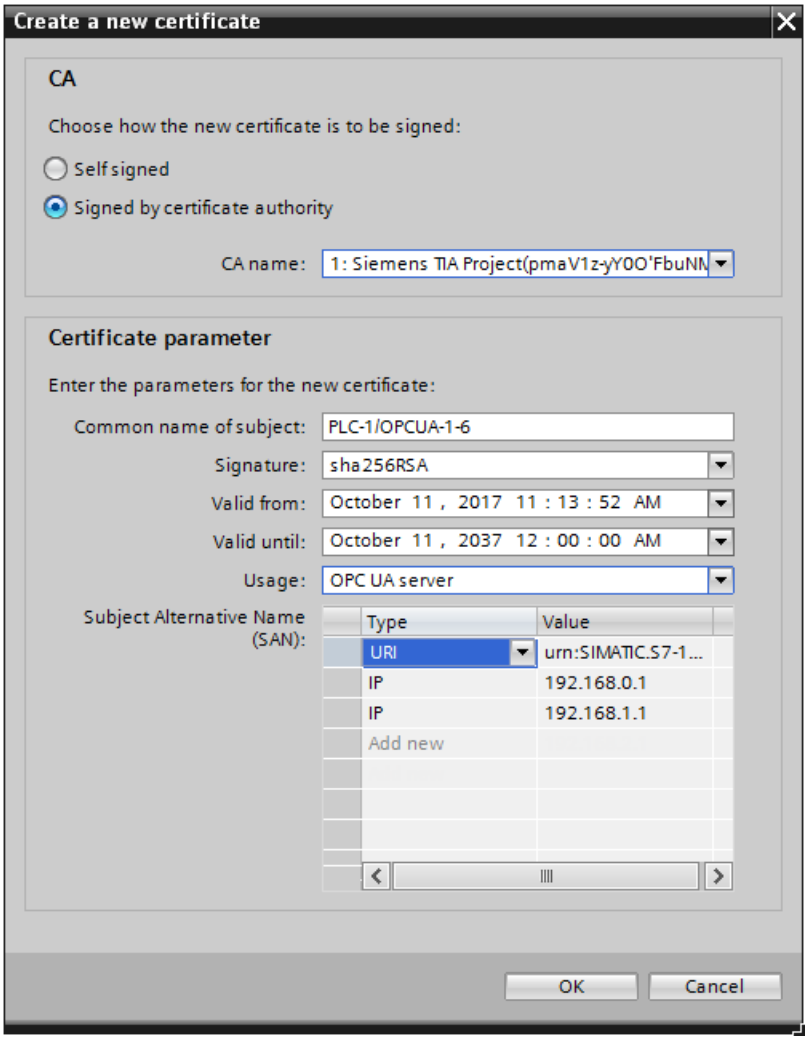


Figure 10-29 Customizing server certificates

4. Use other parameters if this is necessary in accordance with the security specifications in your company or your customer.

Explanation of fields for certificate generation

- CA
  Select whether the certificate is to be self-signed or signed by one of the CA certificates of the TIA Portal. The certificates are described under "Certificates with OPC UA". If you want to generate a certificate that is to be signed by one of the CA certificates of the TIA-Portal, the project must be protected and you must be logged in as a user with all the required function rights. Further information can be found under "Basics of user administration in the TIA Portal".
- Certificate holder
  The default setting always consists of the name of the project and "\OPCUA-1". In the example, the project name is "PLC1". In the properties of the CPU set the project name under "General > Project information" > Name". Keep the default or enter a different name that is more meaningful for the OPC-UA server under "Certificate holder".
- Signature
  Here you select the hash and encryption process that is to be used when signing the server certificate. The following entries are available:
  – "sha1RSA",
  – "sha256RSA".
- Valid from
  Here you enter the date and time for the beginning of the validity of the server certificate.
- Valid until
  Here you enter the date and time for the end of the validity of the server certificate. Ensure that the certificate is valid not only for one year or a few years. In the example the certificate is valid for 30 years. However, for reasons of security you should renew the certificate at much shorter intervals. The long period of validity gives you the opportunity to decide when a suitable moment would be, for example, when the system is being serviced.
- Usage
  The default is "OPC UA client & server". Keep this default for the OPC UA server. The "Create a new certificate" dialog can be called from several points in STEP 7. If, for example, you call this dialog for the Web server of the CPU, "Web server" is entered under "Usage". The following entries are available in the Usage drop-down list:
  – "OPC UA client"
  – "OPC UA client & server"
  – "OPC UA server"
  – "TLS"
  – "Web server"
- Subject Alternative Name (SAN)
  The following is entered in the example above: "URI:urn:SIMATIC.S7-1500.OPC-UAServer:PLC1,IP:192.168.178.151,IP:192.168.1.1". This URI must be correctly entered because it is checked against the communicated application description.
  The following entry would also be valid: "IP: 192.168.178.151, IP: 192.168.1.1". The important thing here is that the IP addresses via which the OPC UA server of the CPU can be accessed are entered here.
  See "Access to the OPC UA server (Page 218)".
  This allows OPC UA clients to verify whether a connection to the OPC UA server of the S7-1500 is really to be established or whether in fact an attacker is trying to send manipulated values from another PC to the OPC UA client.

10.3.3.7 User authentication

Types of user authentication

For the OPC UA server of the S7-1500, you can set what authentication is required for a user of the OPC UA client wishing to access the server.

You have the following options:

- **Guest authentication**
  The user does not have to prove their authorization (anonymous access). The OPC UA server does not check the authorization of the client user
  If you want to use this type of user authentication, select the "Enable guest authentication" option under "OPC UA > Server > Security > User authentication".

  NOTE

  To increase security, you should only allow access to the OPC UA server with user authentication.

- **User name and password authentication**
  The user has to prove their authorization (no anonymous access). The OPC UA server checks whether the client user is authorized to access the server. Authorization is given by the user name and the correct password.
  If you want to use this type of user authentication, select the "Enable user name and password authentication" option under "OPC UA > Server > Security > User authentication".
  Deactivate the guest authentication.
  Enter the user in the "User management" table.
  To do so, click the "<Add new user>" entry. A new user is created with an automatically assigned name. You can edit the user name and enter the password for the user name. You can add a maximum of 21 users.
- **Additional user administration via the security settings of the project**
  The "Enable additional user administration via the security settings of the project" option can be found under the general OPC UA settings (CPU properties: OPC UA > General). If you select this option, the user management for the open project will also be used for user authentication for the OPC UA server: The same user names and passwords are then valid in OPC UA as in the current project.
  Proceed as follows to activate user management for the project:
  – Click "Security settings > Settings" in the project tree.
  – Click the "Protect this project" button.
  – Enter your user name and your password.
  – Enter additional users under "Security settings > Users and roles".

  If you configure an additional OPC UA server in your project, also select the option "Enable additional user administration via the security settings of the project". Repeated input of user names and passwords is then unnecessary.

10.3.3.8 Users and roles with OPC UA function rights

The following options for user authentication use central project settings for project users:

- For the server:
  For configuration of CPU properties (OPC UA > Server > Security > User authentication). Option: "Enable additional user administration via the security settings of the project"
- For the client:
  For configuration of client interface ("Configuration" tab, "Security"). Option: "User (TIA Portal - security settings)"

Requirement

Before you can edit the security settings, the project must be protected and you must be logged on with sufficient rights, for example as administrator.

Settings in the project tree > "Security settings"

You access the central user settings and roles in the protected project in the project tree under "Security settings". This is where you centrally define users with user name, password and function rights. You can simply use these settings elsewhere.

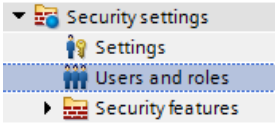


Figure 10-30 Setting user and roles

Reusing central security settings

Examples for reusing elsewhere:

- User selection for user authentication for OPC UA server
  With this setting, you tell the server which client (user) with which user name and which password is allowed to access the server.
- User selection for OPC UA client authentication
  With this setting, you tell the client the user name and password that it is to use for client authentication for the server.

The settings for the client and server must correspond: The user name and password used by the client to log on must have been set up on the server and assigned the required authorizations.

Function rights for server and client

The corresponding function rights for the client or the server must also be enabled for users of the client function and users of the server function on an S7-1500 CPU. It is not enough simply to save the user name and password centrally.

Here is an example to illustrate this type of rights use.

1. Under "Security settings > Users and roles", you define a new role in the "Roles" tab with the name "PLC-opcua-role-all-inclusive", for example.
   Tip: The tab may be covered by an information window ("The current status has not yet been checked..."). In this case, first close the information window.
2. In the "Function rights categories" section, you navigate to the runtime rights and then to the CPU function rights, and select the CPU whose function rights you want to set.
3. You will find the following function rights in the "Function rights" section:
   - **OPC UA server access**
     This function right applies on the OPC UA server of the S7-1500 CPU. Only when this option is selected, can the user with the role "PLC-opcua-role-all-inclusive" transfer certificates, CRLs or trusted lists to the CPU at runtime (push function). This function right is required for automated certificate handling, for example, in the context of GDS (Global Discovery Service).
   - **Managing certificates**
     This function right applies on the OPC UA server of the S7-1500 CPU. Only when this option is enabled, can the user with the role "PLC-opcua-role-all-inclusive" transfer certificates, CRLs or trusted lists to the CPU at runtime (push function). This function right is required for automated certificate handling, for example, in the context of GDS (Global Discovery Service).
   - **User authentication of the OPC UA client**
     This function right applies on the OPC UA client of the S7-1500 CPU (with client instructions). Only when this option is selected, can the user with the role "PLC-opcua-role-all-inclusive" use the user name and password for authentication to establish a session with a server.

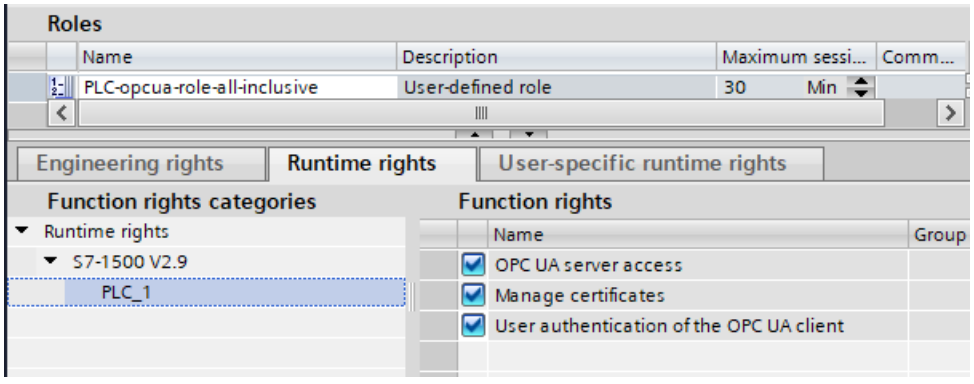


Figure 10-31 Setting function rights

4. The role "PLC-opcua-role-all-inclusive" still needs to be assigned to the relevant users ("Users" tab under "Security settings" in the project tree).

NOTE

"Runtime timeout" for users with OPC UA function rights

The value in the column "Runtime timeout" (max. session duration) in the table for user configuration does not evaluate the CPU for OPC UA runtime rights.

Therefore, a user is not automatically logged out after a certain period of time. For this purpose use OPC UA specific mechanisms such as the parameter "Max. session timeout" (area OPC UA > Server > Settings).

10.3.3.9 Diagnostic settings of the server

Diagnostics

You can specify the scope of the diagnostics of the OPC UA server in the CPU settings.
To change the diagnostics scope, navigate to the "OPC UA > Server > Diagnostics" area.

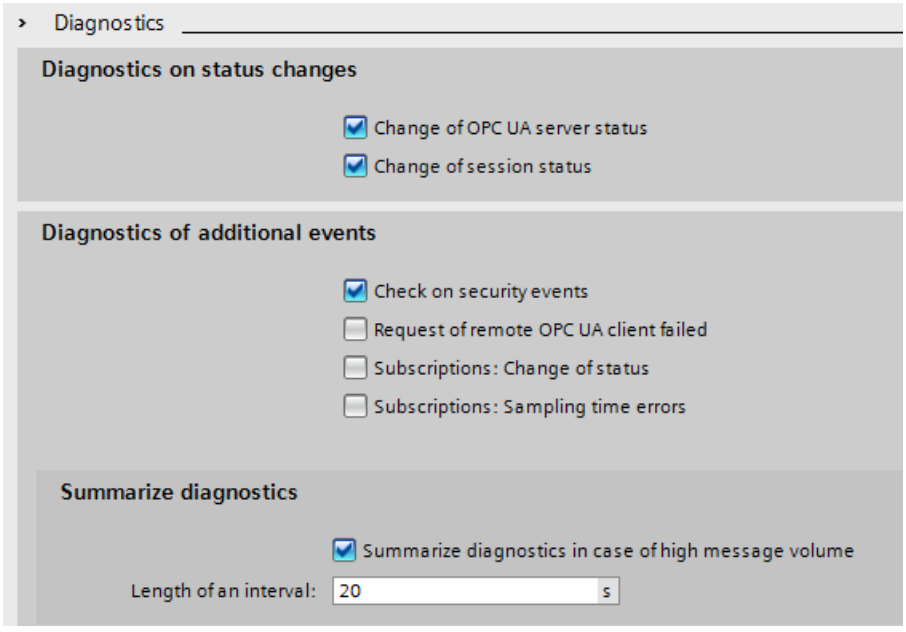


Figure 10-32 Diagnostic settings of OPC UA server

Default setting

The default setting is a diagnostics behavior that supports the most important diagnostics without appreciably increasing the communication load.
You enable diagnostics for subscriptions when the OPC UA server also uses subscriptions, i.e. if necessary during the commissioning phase only.
Reason: A large volume of diagnostic activity generates a high communication load in the CPU and may suppress other important messages. Or, the high volume of diagnostics may result in important messages disappearing in the mass of messages and being ignored.

Additional information

You will find additional information on the meaning and effect of the settings shown above here (Page 292).

10.3.3.10 License for OPC UA

Runtime licenses

A license is required to run the OPC UA server of the S7-1500 CPU. The type of license required depends on the performance of the respective CPU. The following license types are differentiated:

- SIMATIC OPC UA S7-1500 small (required for CPU 1511, CPU 1512, CPU 1513, ET 200SP CPUs, CPU 1515SP PC)
- SIMATIC OPC UA S7-1500 medium (required for CPU 1515, CPU 1516, Software Controller CPU 1507, CPU 1516pro-2PN)
- SIMATIC OPC UA S7-1500 large (required for CPU 1517, CPU 1518)

The required license type is displayed under "Properties > General > Runtime licenses > OPC-UA > Type of required license":

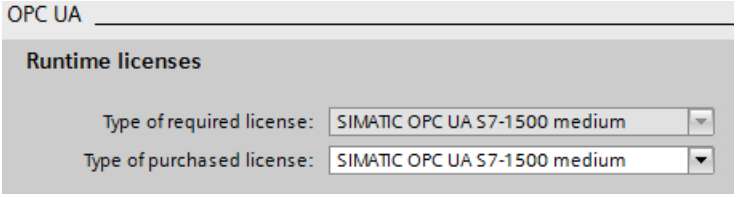


Figure 10-33 OPC UA server Runtime licenses

To confirm purchase of the required license, follow these steps:

1. Click "Runtime licenses > OPC UA" in the properties of the CPU.
2. Select the required license from the "Type of purchased license" drop-down list.

10.3.4 OPC UA server interface configuration

10.3.4.1 What is a server interface?

Definition

A server interface combines nodes of an OPC UA address space of a CPU into a unit, so that a specific view on this CPU is provided for OPC UA clients.
Each server interface defines one or more namespaces in the OPC UA server of the CPU.
STEP 7 (TIA Portal) differentiates between the following types of server interfaces:

- Companion specification
  For this type of server interface, you use a Companion Specification created by a workgroup, for example.
  The workgroup is typically composed of members of the OPC Foundation and another industry organization who have jointly specified an OPC UA information model for a specific purpose (for example, for data exchange with RFID devices or with injection molding machines).
  This information model is realized in the form of OPC UA nodes in the address space of an OPC UA server. OPC UA clients can access these OPC UA nodes.

You can also use the server interface type "Companion specification", for example, to download company-internal information models, e.g. in SiOME.

If you implement a certain companion specification in your project, you apply the specifications of this companion specification into your project as server interface.

For "Companion specification"-type server interfaces, you can import multiple namespaces which the Companion specification uses.

Additional information on companion specifications is available here (Page 238).

Additional information on SiOME is available here (https://support.industry.siemens.com/cs/ww/en/view/109755133).

- When companion specifications refer to type definitions in dependent specifications, use the reference namespaces for this. You import reference namespaces as you would the actual companion specifications.

  See Creating a server interface for companion specification (Page 245).
- If you want to make instance data from FBs or UDTs of the CPU accessible to OPC UA clients, you can have these instance data assignments automatically made as of TIA Portal version V17. You only need to map the FB types or the UDTs to suitable OPC UA data types of an imported reference namespace. For this mapping to be possible, enable the option "Generate OPC UA nodes based on the local data mapping" in the dialog for creating an OPC UA server interface of the type companion specification/reference namespace.

  See Generating OPC UA nodes based on local data mappings of FB types and UDTs (Page 263)

- User-defined server interface:

  For this type of server interface you combine OPC UA nodes of an OPC UA server into a unit.

  To do this, use the specifications for your project or the requirements for your machine or your plant as a basis.

  Additional information on the user-defined server interface is available here (Page 249).

Injection molding machine as an example for companion specification

In this example, a server interface contains the following elements:

- OPC UA nodes which you can write with an OPC UA client to receive information about this injection molding machine (in readable PLC tags)
- OPC UA nodes which you can write with an OPC UA client to transfer values to the injection molding machine (in writable PLC tags)
- OPC UA nodes which you can call with an OPC UA client to start functions of the injection molding machine (via server methods)

This server interface enables a default view of a CPU, which can be used to control an injection molding machine.

For injection molding machines, the companion specifications "OPC UA specifications for plastics and rubber machines" (previously "Euromap") define a whole series of OPC UA nodes which you can apply as a server interface.

Other OPC UA nodes of the CPU are not included in this server interface. This provides a better overview.

Example of user-defined server interface

A CPU should control the production of workpieces. Production begins when a production job arrives from the higher-level control system.

The production jobs are transferred via a server method: A control system transmits information on a workpiece by calling the server method in the CPU. This server method also starts production.

The control system, i.e. the connected OPC UA client, should only see this one server method. Therefore, you create a user-defined server interface in the CPU and assign the server method to this server interface. You enable only this server interface for OPC UA clients and thus limit the view of the CPU to this one function.

10.3.4.2 Using OPC UA companion specifications

Introduction

OPC UA is universally applicable: The standard itself does not, for example, specify how PLC tags are to be named. It is also up to the individual user (application developer) to program and name server methods that can be called over OPC UA.

Information modeling and standardization for devices and sectors

For applications of the same kind, it is worth standardizing your device or machine interface with the "OPC UA toolkit".

Many different bodies and working groups have driven forward standardization and developed a range of companion specifications.

These specifications define:

- The objects, methods and tags with which a typical device or machine is to be described.
- The namespace intended for the specified objects.

Machines are typically structured in functional or technological units, and these units are then standardized.

Companion specifications offer machine and plant operators the benefits of a standardized interface. For example, all RFID readers that comply with the AutoID specifications can be integrated in the same way. This means that all RFID readers that comply with the AutoID specifications can be addressed by OPC UA clients in the same way irrespective of manufacturer.

Another example of companion specifications is the Euromap 77 Companion Specifications from the injection molding machinery sector.

The following section uses the example of Euromap 77 to detail how to apply companion specifications in STEP 7 (TIA Portal) and create the necessary PLC tags.

NOTE

EUROMAP and the OPC Foundation have established the Joint Working Group "OPC UA Plastics and Rubber Machinery".

The existing EUROMAP recommendations EUROMAP 77 (data exchange between injection moulding machines and MES), 82.1 (temperature control devices) and 83 (general definitions) were published under the neutral umbrella of the OPC Foundation as OPC 40077, 40082-1 and 40083.

A major change it the change of the namespace, for example, for EUROMAP 77: Currently "http://opcfoundation.org/UA/PlasticsRubber/IMM2MES/".

The examples listed below use the previously valid designations and references.

Example Euromap 77 (currently: OPC 40077)

Euromap 77 or the successor standard OPC 40077 standardizes the exchange of data between injection molding machines and the higher-level MES (Manufacturing Execution System). This allows the MES to connect all lower-level injection molding machines in the same way.

The standardized data interface facilitates the integration of injection molding machines into a plant.

Using companion specifications: Overview

Euromap 77 is described in the OPC UA XML file "Opc_Ua.EUROMAP77.NodeSet2.xml".

NOTE

Euromap 77, Euromap 83 and OPC UA for Devices (DI)

With Release Candidate 2, some of the Euromap definitions have been transferred from Euromap 77 to Euromap 83 (currently OPC 40083). You will therefore also need to import the OPC UA server interface of Euromap 83.

"OPC UA for Devices" is a generally applicable information model for the configuration of hardware and software components. The information model also serves as the basis for other companion standards and is therefore also imported.

The OPC UA XML files are available here:
Euromap77 (https://www.euromap.org/euromap77)
Euromap83 (https://www.euromap.org/euromap83)
OPC UA for Devices (https://opcfoundation.org/UA/schemas/DI/)

These XML files define an OPC UA interface of an injection molding machine that complies with Euromap 77.

Using Euromap 77: Overview

To use Euromap 77, proceed as follows:

1. Generate an XML file by creating an instance of the type "IMM_MES_InterfaceType" using the SiOME program.
   How to proceed is described below in "Step 1: Create instances in SiOME".
2. In STEP 7 (TIA Portal), create PLC tags and server methods that correspond to the instance of the type "IMM_MES_InterfaceType" (created in Step 1).
   See below for information on how to proceed in "Step 2: Create PLC tags in STEP 7".
   An example of OPC UA nodes and the corresponding PLC tags can be found in section "Creating a server interface for companion specification (Page 245)".
3. In STEP 7 (TIA Portal), add a new server interface of companion specification type and import the XML file you created in step 1.
   The "Creating a server interface for companion specification (Page 245)" section describes how to proceed.
4. Assign the OPC UA nodes of the new server interface to the corresponding PLC tags, which you created in step 2.
   The "Creating a server interface for companion specification (Page 245)" section describes how to proceed.

Step 1: Create instances in SiOME

The following section describes how to use the free program "SiOME", the "Siemens OPC UA Modeling Editor".
With SiOME, you can create an OPC UA XML file, which describes the server interface (an information model).
Download link and explanations about SiOME are available here (https://support.industry.siemens.com/cs/ww/en/view/109755133).

Procedure in STEP 7

To use the new server interface, import the server interface into the STEP 7 project, see section "Creating a server interface for companion specification (Page 245)".
When the project is loaded into the CPU, the new server interface is available for OPC UA clients.

Procedure in SiOME 1.7.3

NOTE

The following description shows the work steps in SiOME 1.7.3.

Follow-up versions of SiOME make it easier for you, for example, to create corresponding DBs, structures, variables or methods in the user program. Using a drag-and-drop operation, you can transfer data, for example, from SiOME to the TIA Portal (user program). In this case, the variables, etc. are already mapped correctly or, for methods, the corresponding FB elements are also generated correctly in the user program.

Download the current SiOME version using the download link listed above, and follow the instructions in the documentation included in the download.

The following description shows the work steps in SiOME 1.7.3.
To use Euromap 77, create an XML file with an instance of "IMM_MES_InterfaceType".

The object type must be instantiated in order for the information model of the specific machine to appear in the address space of the OPC UA server.
The object type "IMM_MES_InterfaceType" is the root object type of Euromap 77. "IMM" stands for "Injection Moulding Machine".
Follow these steps:

1. Download the files "Opc_Ua.EUROMAP77.NodeSet2.xml" and "Opc_Ua_EUROMAP83_NodeSet2.xml" from the Euromap website (see above).
2. Download the file "Opc.Ua.Di.NodeSet2.xml" from the OPC Foundation website.
   The "Opc.Ua.Di.NodeSet2.xml" file contains type definitions which Euromap 77 uses.
3. Start SiOME.
4. First, import the namespace "http://opcfoundation.org/UA/DI/".
   To do so, click the "Import XML" button in the "Information model" area.

   Figure 10-34 "Import XML" button in SiOME

   SiOME displays the dialog for the opening files.
5. To import the file, select the "Opc.Ua.Di.NodeSet2.xml" file and click "Open".
   Result: SiOME imports the XML file and shows the namespace "http://opcfoundation.org/UA/DI/" in the "Namespaces" area.
   The standard namespace "http://opfoundation.org/UA/" is always available in SiOME and does not have to be imported.
6. Now import the namespace "http://www.euromap.org/euromap83/"
   To do so, click the "Import XML" button again in the "Information model" area.
   Select the file "Opc_Ua.EUROMAP83.NodeSet2.xml".
   Result: SiOME imports the XML file and shows the namespace "http://www.euromap.org/euromap83/" in the "Namespaces" area.
7. Now import the namespace "http://www.euromap.org/euromap77/"
   To do so, click the "Import XML" button again in the "Information model" area.
   Select the file "Opc_Ua.EUROMAP77.NodeSet2.xml".
8. Create your own namespace for your project.
   To do this, right-click in the "Namespaces" area on "OPC UA Modelling Editor Project" or on "Namespaces" and select "Add Namespace".
   SiOME opens the "Add Namespace" dialog.

9. Enter the name of a new namespace.
   The "YourCompany.org" namespace is used in the example.
   SiOME now also displays the new namespace:

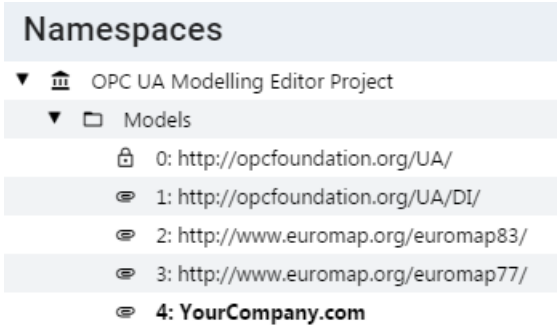


Figure 10-35 Display of the namespace in SiOME

10. Create an instance from the root object type IMM_MES_InterfaceType of the Companion specification Euromap 77.
    To do so, in the "Information model" area, right-click the "DeviceSet" directory and select "Add Instance".
    SiOME displays the "Add Instance" dialog.
11. For "Name", enter a meaningful name for your instance.
    In the example, enter "IMM_Manufacturer_01234".
    For "TypeDefinition", select "IMM_MES_InterfaceType".
    This object type is the root object type of Euromap 77: If you generate an instance of this object type, then use the Euromap 77 once in the address space of your OPC UA server.

12. Click "OK".
    SiOME shows the new instance "IMM_Manufacturer_01234" in the "Information model" area under "DeviceSet":

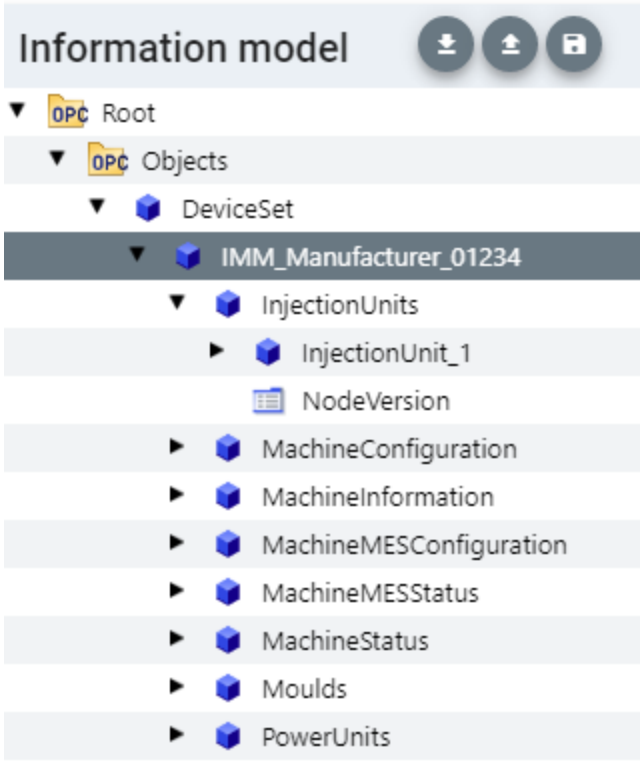


Figure 10-36 Display information model

13. Create an instance of the data type "InjectionUnitType".
    To do this, right-click on the "InjectionUnits" directory in the "Information model" area and select "Add Instance".
    SiOME displays the "Add Instance" dialog.
    For "Name", enter a meaningful name for the instance.
    In the example, enter "InjectionUnit_1".
    For "TypeDefinition", select "InjectionUnitType".
    Click "OK".
14. Create a new "Mould_1" instance of the "MouldType" object type in the "Moulds" directory.
15. Create a new instance "PowerUnit_1" of the "PowerUnitType" object type in the "PowerUnits" directory.

16. Save the XML file.
    To do so, click the "Quick save" button in the "Information model" area:

Figure 10-37 "Quick save" button in SiOME

17. Export the XML file.
    To do so, click the "Export XML" button in the "Information model" area.

Figure 10-38 "Export XML" button in SiOME

    SiOME shows the "Export XML" dialog.
18. Leave all namespaces activated and click "OK".
    SiOME displays the "Save as" dialog.
19. Select a meaningful name and save the exported file.
    In the example, name the XML file "IMM_Manufacturer_01234".

Result:
You have now created an XML file which uses the companion specification "Euromap 77" once (with one instance).

Step 2: Creating PLC tags for the Euromap 77 instance in STEP 7.

For Euromap 77, you must provide PLC tags and server methods in your user program and assign the instance of the "IMM_MES_InterfaceType" type.
To create PLC tags for the instance of the "IMM_MES_InterfaceType" type, proceed as follows.

1. Create a user-defined data type (UDT).
   The figure below shows the beginning of the user-defined data type "InjectionUnit" as example.
   This data type has the same structure as "InjectionUnit" in the type "IMM_MES_InterfaceType".
   Make sure that you use SIMATIC data types that are compatible with the OPC UA data types (see "Mapping of data types" below).

InjectionUnit

	Name	Data type	Default value	Accessible from HMI/OPC UA	Writable from HMI/OPC UA
	BarrelId	String	''	☑	☑
	Index	UDInt	0	☑	☑
	InProduction	Bool	false	☑	☑
	IsPresent	Bool	false	☑	☑

Figure 10-39 Creating a UDT in STEP 7

2. Add a new global data block to your STEP 7 project.
   In the example, name the data block "IMM_Manufacturer_01234", so that there is a reference to the injection molding machine of the respective manufacturer and the serial number.

3. Create a new element in this data block.
   In the example, name this element "InjectionUnit_1"
4. Assign the new user-defined data type "InjectionUnit" to this element.

Result

In your STEP 7 project, you have created a tag for the Euromap 77 in the "IMM_Manufacturer_01234" data block.

10.3.4.3 Creating a server interface for companion specification

For basic information on companion specifications, refer to the section "Using OPC UA companion specifications (Page 238)". The benefits of the Euromap 77 companion scpecification, which provides a model for injection molding machines, is also discussed in detail there.

Using this companion standard, the S7-1500 CPU can control an injection molding machine, for example, and provide an OPC UA client, such as a higher-level MES system, with an interface for accessing the functions and tags of injection molding machine.

An OPC UA server interface of the type "Companion Standard" limits the access of clients to exactly those functions and tags that are required, for example, for higher-level systems (MES systems).

The following description shows how to create a server interface in STEP 7 (TIA Portal) which contains only the Euromap 77 companion specification.

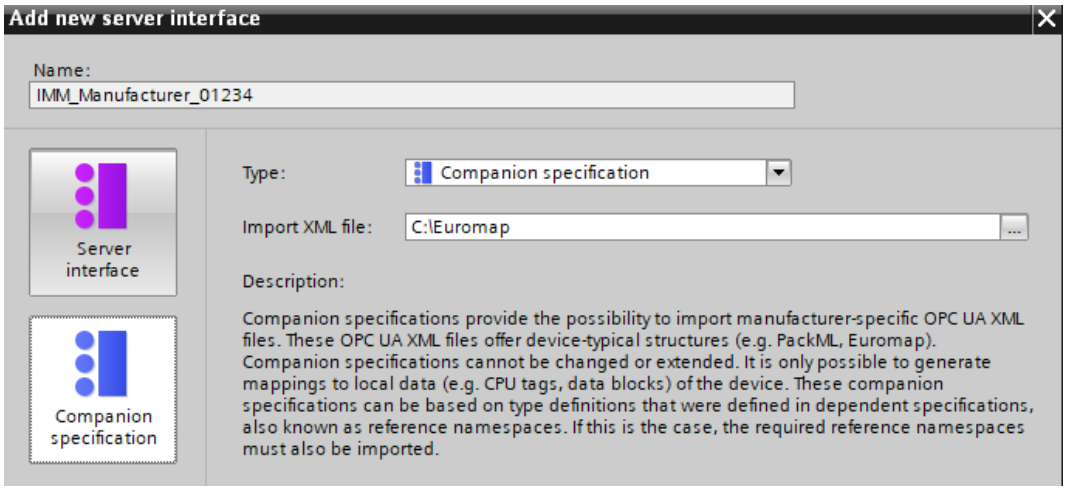
If you want to make OPC UA clients accessible to other tags or methods than those required for the management of an injection molding machine, simply create another OPC UA server interface. In this way, you can clearly arrange the functionality of the CPU as OPC UA server.

Creating a server interface for a companion specification

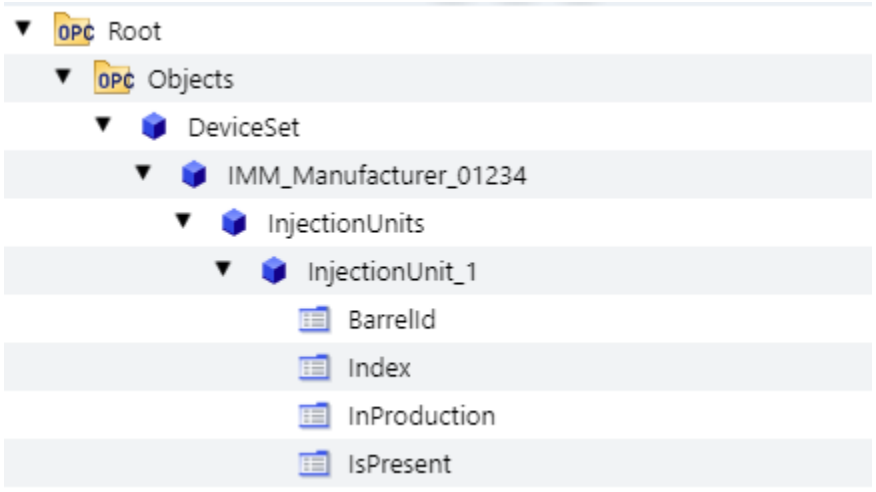
To create a server interface for a companion specification with STEP 7 (TIA Portal), proceed as follows:

1. Select the CPU that you want to use as an OPC UA server.
2. In the project tree, click "OPC UA communication > Server interfaces".
3. Double-click "Add new server interface".
4. To select this type of server interface, click "Companion specification".
   A general name for the new server interface is entered in the dialog, for example "Server_Interface_1".

5. Change the name of the new server interface so that it is descriptive in your project.
   The name should have the following structure according to Euromap 77:
   "IMM_<Manufacturer>_<Serial number>".
   The example uses the name "IMM_Manufacturer_01234".



6. In the "Import XML file" field, select an XML file that describes an information model.
   The "Using OPC UA companion specifications (Page 238)" section describes how to create such an XML file with the SiOME tool.
   The figure below shows a section from the information model:
   "IMM_MANUFACTURER_0123456" an instance (use) of the type "IMM_MES_InterfaceType" which was defined by Euromap 77 . "InjectionUnit_1" is an instance of the "InjectionUnitType" type of Euromap 77.



7. Click "OK".
   STEP 7 (TIA Portal) imports the information model described in the selected XML file.
   An error occurs when type definitions are used in the imported XML file that are not yet present in STEP 7 (TIA Portal) and that are also not contained in the imported XML file.
   In the example, an XML file is imported that uses type definitions defined in the following namespaces (Namespaces):
   – http://opcfoundation.org/UA/DI/
   – http://www.euromap.org/euromap83/
   – http://www.euromap.org/euromap77/
   Tip: STEP 7 displays missing namespaces in the lower area of the OPC UA interface editor ("Properties" tab).
   To do this, select the server interface in the project tree (here: IMM_Manufacturer_01234) and select the "Namespaces" area in the inspector window. Missing namespaces are selected.
   If one or more namespaces are missing in your STEP 7 project, create a new server interface of the "Reference namespace" type for each namespace.
   The "Creating a server interface for reference namespace (Page 261)" section describes the procedure.
   If all reference namespaces are available, STEP 7 displays the table without errors:

DeviceSet	Object
IMM_Manufacturer_01234	Object
InjectionUnits	Object
InjectionUnit_1	Object
TemperatureZones	Object
BarrelId	String
Index	UInt32
InProduction	Boolean
IsPresent	Boolean

8. Drag the OPC UA elements from the right area of the table (OPC UA elements) to the left part of the table (OPC UA server interface) so that the respective OPC UA elements (the local PLC tags) are assigned to the respective OPC UA nodes of Euromap 77.
   The figure below shows a section from the assignment of the local data (PLC tags) to the OPC UA nodes of the Euromap 77:

OPC UA-Server-Schnittstelle

Name	Node type	Local data	Data type
DeviceSet	Object		
IMM_Manufacturer_01234	Object		
InjectionUnits	Object		
InjectionUnit_1	Object		
TemperatureZones	Object		
NodeVersion	String		
BarrelId	String	"IMM_Manufacturer_01234"."InjectionUnit_1"."BarrelId"	String
Index	UInt32	"IMM_Manufacturer_01234"."InjectionUnit_1"."Index"	UDInt
InProduction	Boolean	"IMM_Manufacturer_01234"."InjectionUnit_1"."InProduction"	Bool
IsPresent	Boolean	"IMM_Manufacturer_01234"."InjectionUnit_1"."IsPresent"	Bool

NOTICE
Checking the mapping of CPU local data on nodes of the OPC UA server interface When invalid assignments (mappings) exist in the server interface, they can result in incorrect read and write operations. Check the assignments and run a consistency check.

Information on the server interface

The editor for configuring the OPC UA server interface is structured as a table and provides the following information:

- **Name**
  The top node (root node) is named "IMM_Manufacturer_01234" in the example. If a client browses in the address space of the server, this node is the container for all lower-level nodes. BrowseName and the DisplayName of this node depend on the name you have assigned for the server interface.
  In this case, for example, this name stands for the injection molding machine as a whole. It is the name of the instance of the Euromap 77 companion specification that is used here. According to the companion specification, the instance name should begin with "IMM", followed by the name of the manufacturer of the injection molding machine; the serial number of the machine is added to the end. This allows a unique identification of the machine.
  The names of all other (lower-level) nodes are defined by the specification (in the example above by Euromap 77). These node names must not be changed. This ensures a uniform view of all injection molding machines, which complies with the specification.
- **Node type**
  Type of the OPC UA node. The type is specified by the companion specification that is used.
  In the following cases, STEP 7 marks a node type in the table in color:
  – No definition is included for this in the imported XML file or
  – The namespace in which the type was defined is not available in STEP 7.
  In this case, create a server interface of the "Reference namespace" type for the missing namespace or for each of the missing namespaces.
  The missing namespaces can be found under "Namespaces" in the properties of the server interface.
- **Local data**
  STEP 7 displays the data block which is assigned to the OPC UA node: The CPU reads the value of the OPC UA node from this data block.
  If a data block is highlighted in color (e.g. after a consistency check), the specified data block is not available in the CPU.
  In this case, you have to create the missing data block in the CPU (of user program) and supply it with values.
- **Data type**
  The SIMATIC data type of the PLC tag (e.g. element of a data block) in the CPU, from which the value of an OPC UA node (UAVariable type) is read, or to which a value is assigned.

Generate local data

You have the option of generating local data for all or for selected nodes of the server interface that are not already assigned ("mapped") to local data of the CPU. The newly created local data are mapped automatically.
You trigger the automatic generation of local data by clicking the "Generate local data" button (all nodes that are not already mapped) or by selecting individual nodes and then clicking the "Generate local data" shortcut menu.
"Generate local data" button:

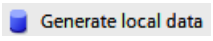


Figure 10-40 "Generate local data" button:

You can only generate nodes that can be mapped to local data, which means no objects, no folders, no methods or input/output arguments of methods.
After you have clicked the button or selected the shortcut menu, you must select in the follow-up dialog box whether the local data should be created in a new DB or in an existing DB.

Consistency check

You have the option to check the server interface.
STEP 7 (TIA Portal) checks whether the OPC UA node of the server interface PLC tags (data blocks) has been assigned compatible SIMATIC data types.
To check the consistency of the server interface, click on the following icon in the toolbar of the OPC UA server interface editor:

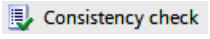


Figure 10-41 "Consistency check" button

Export interface

You have the option of exporting the OPC UA server interface as an XML file. This XML file contains all data type definitions referenced by the server interface.
To export the OPC UA server interface, click on the following icon in the toolbar of the OPC UA server interface editor:

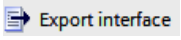


Figure 10-42 "Export interface" button

10.3.4.4 Creating a user-defined server interface

Introduction

The description is based on the following example:
A protective fence surrounds the production cell "Cell_1". The fence is equipped with the gate "Gate_1".

An S7-1500 CPU controls the entire production cell and also controls access through Gate_1.
A robot packs drugs into boxes in the production cell and then stacks the boxes on pallets.
Self-driving vehicles for automated material transport move the pallets to the central warehouse, thereby passing through Gate_1.
The CPU publishes a server interface via which the driverless transport systems arrange for Gate_1 to open.
The server interface contains the server method "smOpenGate" for opening the gate and the tag "Gate_1_State" which indicates the status of the gate (open or closed).

Creating a user-defined server interface

To create an Server interface, follow these steps:

1. Select the CPU that you have used and configured as OPC UA server.
2. Click "OPC UA communication > Server interfaces".
3. Double-click "Add new server interface".
   STEP 7 displays the following dialog.

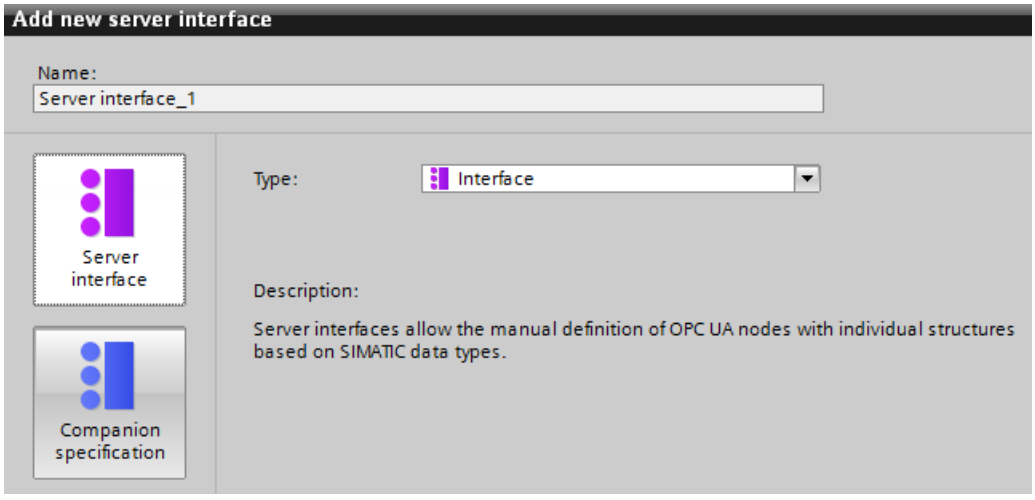


Figure 10-43 Adding the server interface

4. Change the name of the new server interface so that it is descriptive in your project.
   In the example, change the name "Server-interface_1" suggested by STEP 7 to "Cell_1".
5. Click "Server interface" and then "OK".

6. Click on the triangle in front of "Program blocks" in the area "OPC UA elements" to open the "Program blocks" folder.
   STEP 7 displays the following table for editing:

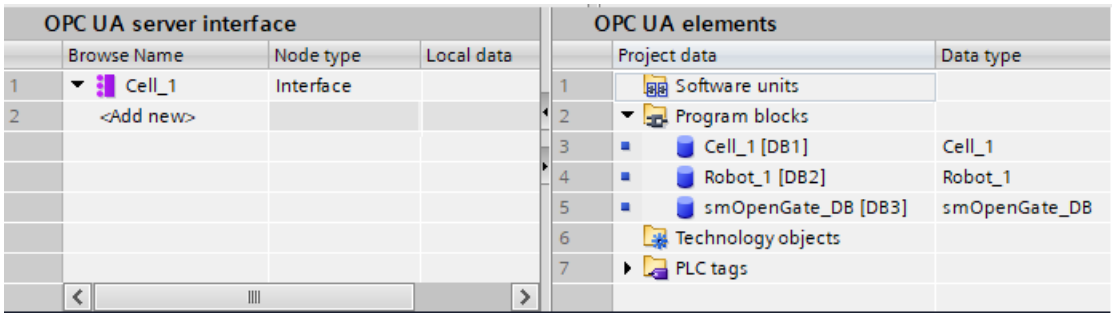


Figure 10-44 Editing the server interface

The editor is divided into two areas.

- **OPC UA server interface**
  On the left is the root node of the server interface "Cell_1".
  This interface is currently still empty: No OPC UA elements have been added to the server interface yet.
- **OPC UA elements**
  On the right are the OPC UA elements.
  OPC UA elements are objects that have been created so far in the STEP 7 project and have the property "Accessible from HMI/OPC UA".
  You can add the OPC UA elements to the new server interface "Cell_1".

7. Drag the OPC UA elements into the "<Add new>" line of the new server interface.

NOTE

The following applies in general: If you store data blocks or technology objects in the left area of the table, STEP 7 (TIA Portal) creates an object in the server interface. The elements of the data blocks are arranged as separate nodes below this.

If you store structures in the left area of the table, STEP 7 creates a node for the structure as a whole and nodes for each element of the structure.

The same applies to arrays: Again, STEP 7 creates a node for the array as a whole and nodes for each element of the array.

When you place a method in the left area of the table, STEP 7 creates a single node; the arguments of the inserted method are displayed for information purposes.

In the example, you drag the "Gate_1_State" tag from the right area to the left area to "<Add new>".
Then, drag the server method into the left area.
This server method is located within the "smOpenGate_DB [DB3]" data block in the right area.

STEP 7 (TIA Portal) displays the dialog as follows:

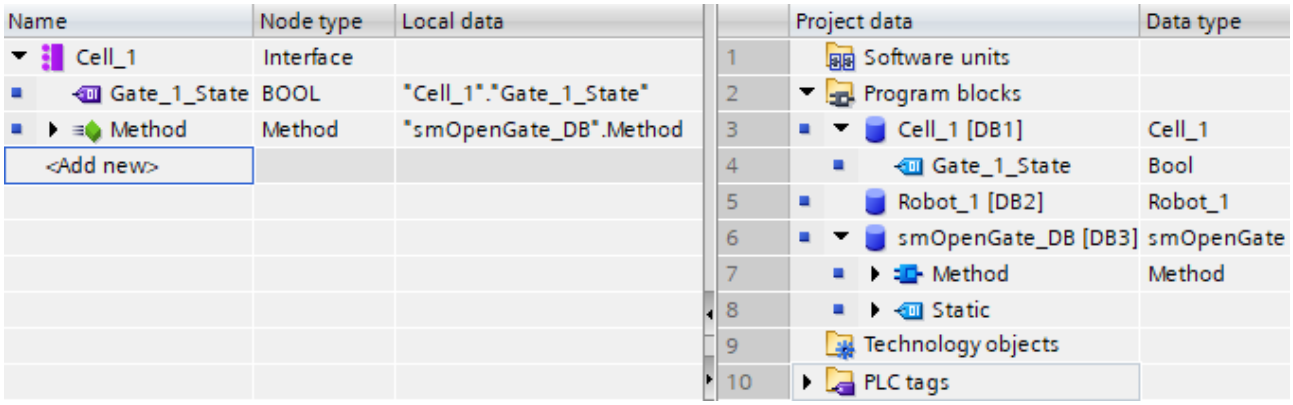


Figure 10-45 Adding OPC UA elements to the server interface

NOTICE
Checking the mapping of CPU local data on nodes of the OPC UA server interface When invalid assignments (mappings) exist in the server interface, they can result in incorrect read and write operations. Check the assignments and run a consistency check.

Limiting the view to OPC UA servers

By selecting the OPC UA elements, you limit the view to the OPC UA server and the options of the OPC UA clients.

In the server interface of the example, the "Robot_1" data block is missing because industrial trucks do not need access to the server methods and tags of the robot.

In this case, it is best to disable the standard server interface (SIMATIC namespace) in the OPC UA properties of the S7-1500 CPU so that the filtered nodes cannot be accessed any other way.

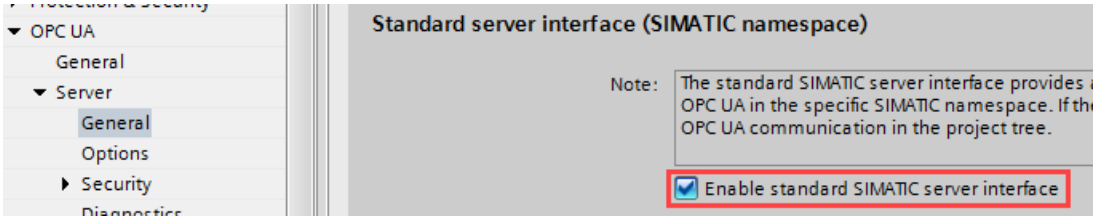


Figure 10-46 Disabling the standard server interface

You can also disable the visibility of each configured OPC UA server interface in the properties of the server interface and thus prevent that this server interface can be used by clients during operation.

- To do this, select the server interface and right-click on the "Properties" command.

This option lets you define multiple server interfaces centrally, for example, and enable and download only the required server interface.

Once a server interface has been defined, you can drag it to another CPU in the project tree.

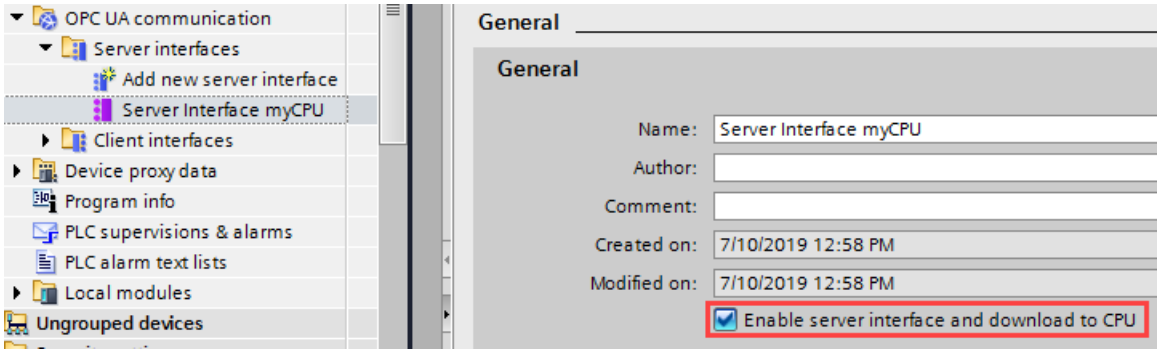


Figure 10-47 Disabling the visibility of the server interface

Information on the server interface

The "OPC UA Server Interface" editor is structured as a table and provides the following information:

Note that not all columns are displayed initially. You determine what is displayed by right-clicking on the header line of the table.

When a row is selected, you can display the OPC UA attributes of the node in the Inspector window ("OPC UA attributes" area), such as node ID, node class, node type, and description.

- **BrowseName**
  The language neutral name of the user-defined server interface is at the top (BrowseName). This name can be freely selected.
  The names (BrowseNames) of the individual OPC UA nodes that have been added to the server interface are under the name of the interface.
  You cannot change the name of an OPC UA node in this dialog. The names come from the STEP 7 project.
  You can delete an OPC UA node from the table. This means that it no longer belongs to the server interface and is no longer visible to OPC UA clients.
- **DisplayName**
  Similar to BrowseName. However, the name can be translated and is displayed, if available, in the corresponding language.
- **Node ID**
  NodeId of the OPC UA node, e.g. http://Server-Node_1; i=1
- **Node type**
  Type of the OPC UA node, for example BOOL, BYTE, INT.
  These node types were defined by Siemens, not by the OPC Foundation. For example, the OPC Foundation uses the Boolean node type for BOOL. BOOL is directly derived from Boolean.
  The specified node type cannot be changed in this dialog: If you want to use a different node type, you must change the type of the respective PLC tags in the STEP 7 project.
- **Data type**
  The SIMATIC data type used in the STEP 7 project is specified, for example, Bool, Byte, Int. etc.

- **Access level**
  - If an OPC UA node is a tag (UAVariable type), the node can only be readable (RD) or readable and writable (RD/WR).
  - If an OPC UA node is a method (UAMethod type), this node can always be called.
- **Local data**
  The SIMATIC data type of the data block in the CPU, from which the value of an OPC UA node (UAVariable type) is read, or to which a value is written.

Generate local data

You have the option of generating local data for all or for selected nodes of the server interface that are not already assigned ("mapped") to local data of the CPU. The newly created local data are mapped automatically.

You trigger the automatic generation of local data by clicking the "Generate local data" button (all nodes that are not already mapped) or by selecting individual nodes and then clicking the "Generate local data" shortcut menu.

"Generate local data" button:

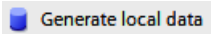


Figure 10-48 "Generate local data" button:

You can only generate nodes that can be mapped to local data, which means no objects, no folders, no methods or input/output arguments of methods.

After you have clicked the button or selected the shortcut menu, you must select in the follow-up dialog box whether the local data should be created in a new DB or in an existing DB.

Consistency check

You have the option to check the consistency of the server interface.

During the consistency check, STEP 7 checks whether the OPC UA nodes of the server interface are each assigned to a suitable OPC UA element (identical data type) or whether the used element still exists in the CPU.

To check the consistency of the server interface, click on the following icon in the toolbar of the OPC UA server interface editor:

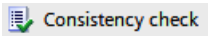


Figure 10-49 "Consistency check" button

Export interface

You have the option of exporting the OPC UA server interface as an XML file. This XML file contains all data type definitions referenced by the server interface.

To export the OPC UA server interface, click on the following icon in the toolbar of the OPC UA server interface editor:

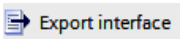


Figure 10-50 "Export interface" button

More information

You can find information on master copies for the OPC UA communication in the section Master copies for OPC UA communication (Page 338).

10.3.4.5 Data types for companion specifications

Mapping of data types

The table below shows the compatible SIMATIC data type for each OPC UA data type.
Assign the data types as shown below (SIMATIC data type - OPC UA data type). Other assignments are not permitted. STEP 7 does not check the observance of this rule and does not prevent an incorrect assignment. You are responsible for the rule-compliant selection and assignment of the data types.
You can also use the listed data types, for example, as elements of structures/UDTs for input and output parameters of self-created server methods (UAMethod_InParameters and UAMethod_OutParameters).

Table 10-3 Mapping of data types

SIMATIC data type	OPC UA data type
BOOL	Boolean
SINT	SByte
INT	Int16
DINT	Int32
LINT	Int64
USINT	Byte
UINT	UInt16
UDINT	UInt32
ULINT	UInt64
REAL	Float
LREAL	Double
LDT	DateTime
WSTRING	String
DINT	Enumeration (Encoding Int32) and all derived data types

SIMATIC data type	OPC UA data type
User-defined data type required (UDT, user-defined data type) The user-defined data type must be created with the prefix "Union_", for example "Union_MyDatatype". See example below the table. The first element (Selector) in this UDT must have the data type "UDINT".	UNION and all derived data types
See LocalizedText and ByteString support for the OPC UA server (Page 256)	LocalizedText ByteString

User-defined data type for UNION required

The figure below shows the tag "MyVariable", which has the "Union_MyDatatype" data type.

This SIMATIC data type corresponds to an OPC UA tag with the data type UNION.

The figure shows an example of the declaration: When Selector = 1, Union takes a ByteArray; when Selector = 2, Union takes a WString.

Name	Data type
Static	
Selector	UDInt
ByteArray	Array[0..1] of Byte
WString	WString[42]

10.3.4.6 LocalizedText and ByteString support for the OPC UA server

As of TIA Portal version V17 and S7-1500 CPU firmware version V2.9, the two OPC UA Built-in data types "LocalizedText" and "ByteString" are available for mapping to corresponding SIMATIC data structures. For the definition of these OPC UA data types, see also OPC 10000-3 DataType definitions.

These data types are used in companion specifications, for example, and can be easily handled by the user program with the OPC UA interface editor.

LocalizedText

A structure containing a string with locale identifier (e.g. 'en-US').

The structure has three elements with a defined order and the following structure in SIMATIC:

- **Encoding** (data type OPC_UA_LocalizedTextEncodingMask): Indicates in bit 0 whether the "Locale" field has a content and in bit 1 whether the "Text" field has a content. Both fields should have a content. We therefore recommend setting the "Encoding" value for SIMATIC to 2#00000011.
- **Local** (WString data type): Locale, for example, 'en-US'.
- **Text** (WString data type): Text box, for example, 'Text'.

ByteString

A sequence of octets.

The structure is built up as follows:

- **ActualLength** (data type "OPC_UA_ByteStringActualLength"): Length of the ByteString array that is filled
- **ByteString** ("Array of Byte" data type): Byte array

Requirement

An OPC UA server interface has been created.

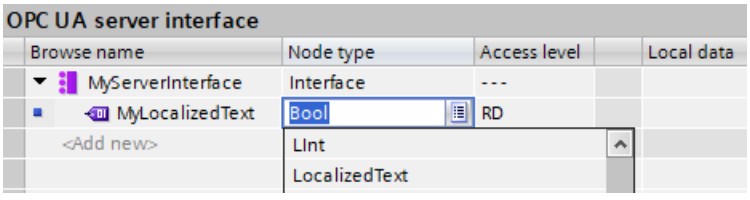
Application

You can import a companion specification or a reference namespace that contains definitions of the "LocalizedText" or "ByteString" type.
Likewise, you can create a server interface and define an address model yourself with the data types "LocalizedText" or "ByteString". The procedure is described in the next section.

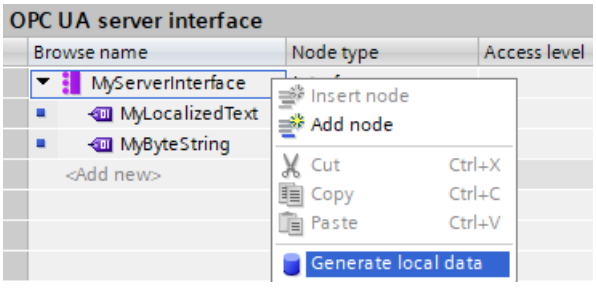
Procedure

You will learn how to create a node of the type "LocalizedText" or "ByteString" with the interface editor and then have a SIMATIC data structure created automatically for this node in the paragraphs below.
To define OPC UA nodes of the type "Localized Text"/"ByteString" in a server interface, follow these steps:

1. Create nodes of the type "LocalizedText" or "ByteString" in the "OPC UA server interface" area. These node types are included in the list of selectable node types.

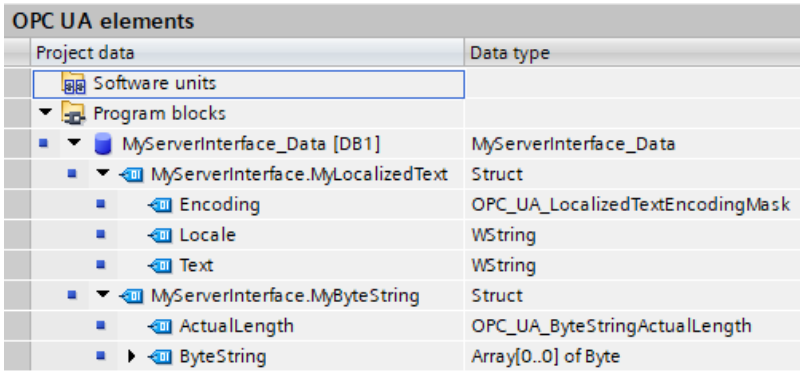


2. Select the "Generate local data" command from the shortcut menu. To generate the local data, select a data block, for example, a new DB with the name "MyServerInterface_Data".



Result: STEP 7 generates the corresponding structure for the mapping in which you still need to adjust the required text length (Text) and the required locale (Locale) for "LocalizedText".
The same is true for "ByteString"; in this case, you must adjust the length and the array.

The consistency check generates a warning to indicate the required adjustments.



Rules

- You can also create UDTs with the structure as shown above for the node types "LocalizedText" or "ByteString" and use them for DB elements.
- You can use the node types "LocalizedText" or "ByteString" in other structures (nests).
- The SIMATIC structures for "LocalizedText" or "ByteString" may only be used completely; an isolated data type, such as "OPC_UA_LocalizedTextEncodingMask" for other purposes, is not provided.
- Input and output parameters of methods can also be of the data type/node type "LocalizedText" or "ByteString".

10.3.4.7 Using additional OPC UA basic data types for companion specifications

Apart from the OPC UA data types listed in the section "Mapping of data types" and their correspondences on the SIMATIC side, there are the following OPC UA basic data types which you can also use:

- OpcUa_NodeId
- OpcUa_QualifiedName
- OpcUa_Guid
- OpcUa_XmlElement
- OpcUa_ByteString
- OpcUa_LocalizedText

Requirement for the use of the basic data types listed above as variables in the application program: The basic data types have to exist as complex data types that are structured exactly like the corresponding OPC UA basic data types.

- OpcUa_NodeId and OpcUa_QualifiedName exist as system data types; that's why you can use these data types not only for single variables but also as elements of a structure.
- For the basic data types or built-in data types GUID and XmlElement you have to create a PLC data type in accordance with the OPC UA specification and subsequently use it as an element in a structure so that the data types of the elements can be resolved. What each PLC data type must look like is described below for every single basic data type.

- For OpcUa_ByteString and OpcUa_LocalizedText, the requirements have been created in TIA Portal V17 to simply use these data types in the server interface of the "Companion Specification" type:
  - You create the corresponding node type in the server interface (for example, OpcUa_LocalizedText)
  - You click on "Generate local data"

  STEP 7 then automatically generates the appropriate data structures in a DB.

System data type "OPC_UA_NodeId"

For the OPC UA basic data type "OpcUa_NodeId", please refer to the following table for the meaning of the parameters. Use OPC_UA_NodeId for the identification of a node in the OPC UA server.

Parameter	S7 data type	Meaning
NamespaceIndex	UINT	Namespace index of the node in the OPC UA server. A node can, for example, be a tag.
Identifier	WSTRING[254]	The designation of the node (object or tag) depends on the identifier type: • Numeric identifier: The node is labeled with a number, for example "12345678". • String identifier: The node is labeled with a name, for example "MyTag". No distinction is made between upper and lower case.
IdentifierType	UDINT	Type of identifier • 0: Numeric identifier • 1: String identifier • 2: GUID • 3: Opaque

System data type "OPC_UA_QualifiedName"

See the following table for the structure of the system data type "OPC_UA_QualifiedName":

Name	S7 data type	Meaning
NamespaceIndex	UINT	The namespace index of the name.
Name	WSTRING[64]	Name of the node or tag.

UDTs for basic data types GUID and XmlElement

UDT "Guid"

For the basic data type "Guid", create the following PLC data type. The default values used as examples can also be set differently.

Guid

	Name	Data type	Default value
1	Data1	UDInt	16#11223344
2	Data2	UInt	16#5566
3	Data3	UInt	16#7788
4	Data4	ULInt	16#99AABBCCDDEEFF11

UDT "XmlElement"

An XmlElement is a serialized XML fragment (UTF-8 string).

For the basic data type "XmlElement", create the following PLC data type:

XmlElement

	Name	Data type	Default value
1	XmlElement	WString	WSTRING#''

10.3.4.8 Rules for OPC UA XML files

Importing exported OPC UA XML files to an S7-1500 CPU

Please note the following information when importing server interfaces that come from the OPC UA XML export of an S7-1500.

NOTE

Import blocked for namespace "http://www.siemens.com/simatic-s7-opcua"

You cannot import server interfaces with the namespace "http://www.siemens.com/simatic-s7-opcua" to an S7-1500 CPU because this namespace is reserved for S7-1500 CPUs (standard SIMATIC server interface) and is not available for imports.

If you want to import a server interface with the namespace "http://www.siemens.com/simatic-s7-opcua", open the server interface to be imported (OPC UA XML file) and change the namespace in the relevant places. The file thus changed can then be imported.

Integrity of the OPC UA XML files

OPC UA XML files represent the server address space. These files are, for example, imported by you in the context of OPC UA Companion specifications as a server interface after adaptation to the application, loaded into the S7-1500 CPU and tested.

⚠ WARNING

No checking of imported OPC UA XML files

Protect these OPC UA XML files against unauthorized manipulation since STEP 7 does not check the integrity of these files.

Recommendation

To minimize risks in the case of an extension or adaptation of the server address space, follow these steps:

1. Protect the project (project navigation: Security settings > Settings).
2. Export the corresponding server interface before the extension or adaptation.
3. Revise this OPC UA XML file.
4. Import the file again as a server interface.

10.3.4.9 Creating a server interface for reference namespace

Companion specifications and referenced namespaces

A series of OPC UA object types (as well as additional definitions) are defined in a companion specification. These object types are each defined in namespaces so that the names of the object types (type definitions) are unique.

To use a companion specification in your project, create instances of object types of this companion specification.

To do this the object definitions must be available in your STEP 7 project. If this is not the case, you must import the object definitions. To import all definitions of a namespace, create a server interface of type "Reference namespace" for each namespace in STEP 7.

NOTE

EUROMAP and the OPC Foundation have established the Joint Working Group "OPC UA Plastics and Rubber Machinery".

The existing EUROMAP recommendations EUROMAP 77 (data exchange between injection moulding machines and MES), 82.1 (temperature control devices) und 83 (general definitions) were published under the neutral umbrella of the OPC Foundation as OPC 40077, 40082-1 and 40083. However, the examples listed below use the previously valid designations and references.

Example Euromap 77 (currently OPC 40077)

You have added a server interface for the companion specification Euromap 77 (currently OPC 40077).

The server interface uses object types defined in OPC UA DI as well as in Euromap 83 and Euromap 77 in their corresponding namespaces.

Therefore, in addition to the server interface Euromap 77 of the "Companion Specification" type, create additional server interfaces of "Reference namespace" type in STEP 7, in each case for the following namespaces:

- http://opcfoundation.org/UA/DI/
- http://www.euromap.org/euromap83/
- http://www.euromap.org/euromap77/

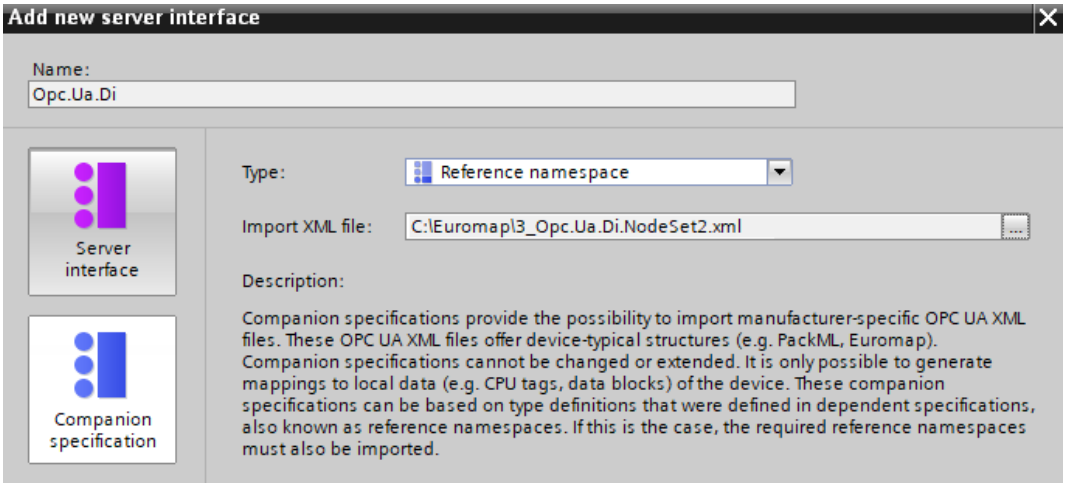
The following description shows you how to proceed.

Creating a server interface for a reference namespace

To create a server interface for a namespace, proceed as follows:

1. Select the CPU that you want to use as an OPC UA server.
2. Click "OPC UA communication > Server interfaces".
3. Double-click "Add new server interface".
   STEP 7 (TIA) now displays the dialog "Add new server interface".
   A general name for the new server interface is entered in the dialog, for example "Server_Interface_1".
4. Assign a descriptive name for the new server interface.
   In the example, select the name "OPC.Ua.Di" or a similar name that clearly references the namespace "http://opcfoundation.org/UA/DI/".
   This namespace must be imported first. It contains basic definitions (for example, the UAObjectType "DeviceType").

5. For "Import XML file", select an XML file that contains the definitions of the namespace "http://opcfoundation.org/UA/DI/".
   Select the file "Opc.Ua.Di.NodeSet2.xml" in the example. You can download this file here: Opc.Ua.Di.NodeSet2.xml (https://opcfoundation.org/UA/schemas/DI/)
   The figure below shows the dialog with the entries:



6. Click "OK".
   STEP 7 (TIA) now generates the new server interface.

You can find the server interface in the project tree of STEP 7 (TIA Portal), under "OPC UA Communication > Server interfaces > Namespace references".

If a companion specification uses additional namespaces, add a new server interface for each namespace.

Add additional server interfaces for Euromap77

For Euromap 77, you still need the following namespaces:

- http://www.euromap.org/euromap83/
- http://www.euromap.org/euromap77/

First, add a server interface for the namespace "http://www.euromap.org/euromap83/".
This namespace contains basic definitions for Euromap 77, therefore it is required here first. All definitions of this namespace are included in the XML file "Opc_Ua.EUROMAP83NodeSet2.xml", which you can download from the Euromap website (https://www.euromap.org/en/euromap83).
Then add a server interface for the namespace "http://www.euromap.org/euromap77". All definitions of this namespace are included in the XML file "Opc_Ua.EUROMAP77.NodeSet2.xml", which you can also download from the Euromap website (https://www.euromap.org/en/euromap77).

10.3.4.10 Generating OPC UA nodes based on local data mappings of FB types and UDTs

If you want to make instance data from FBs or UDTs of the CPU accessible to OPC UA clients you can, as of TIA Portal version V17, have these instance data assignments automatically made.
You only have to map the FB types or the UDTs to suitable OPC UA data types of imported reference namespaces. Based on these mappings created in STEP 7 (TIA Portal), generate the

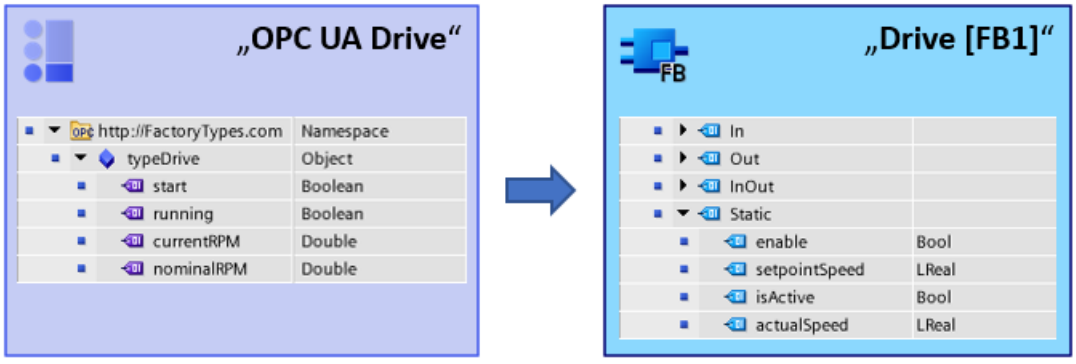
required nodes in the server interface for each FB instance or for each UDT usage during the compile.

If you extend your program and add more FB instances or UDT usages, or if you add existing instances delete, you do not need to worry about adapting the server interface: STEP 7 automatically adjusts the server interface when compiling the program.

Example

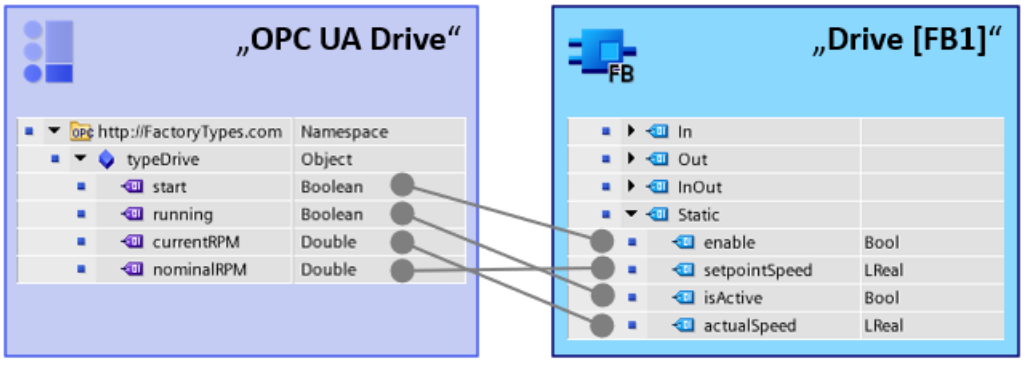
- You create a function block (FB) in the user program of the CPU and define in the "Static" area of the interface of the FB the parameters that form the "memory" of the FB. The instances (values) of this parameter are to be accessible for OPC UA clients.
- You create an OPC UA data type (e.g. with SiOME) with elements that correspond to the data type the parameters in the static area of interface of the FB. The order of the elements does not matter. Then import the reference node set file (reference namespace) as a reference name space.

The following figure shows the assignment of elements as comparison of the reference namespace view (server interface) and the OPC UA elements view (program).



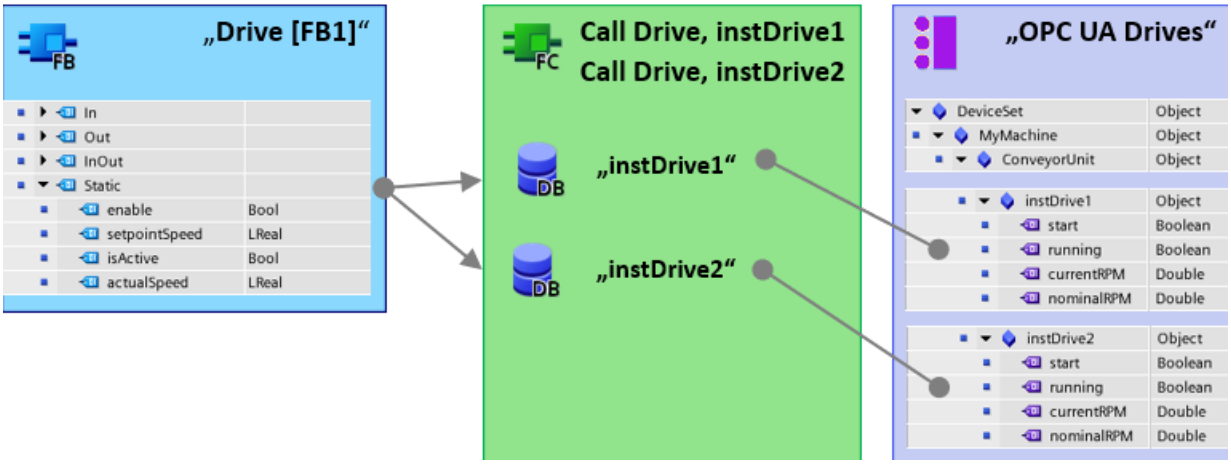
Mapping of data types (FB interface - OPC UA interface): Principle

The following figure shows the assignment of the elements from the user program of the CPU to the elements of the OPC UA server interface. The order of the elements does not have to match.

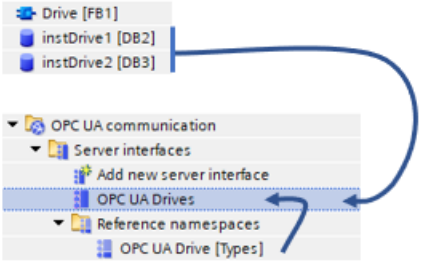


Automatic generation of the OPC UA server instances in the server interface: Principle

The figure below shows the compilation of the project. The instances of the user program are also generated in the server interface.



By mapping between FB type information / UDT type information and OPC UA type information, STEP 7 is able to create all instances present in the program as nodes in the server interface.



Rules

- Only the FB elements in the "Static" area of an FB interface can be mapped to OPC UA type descriptions.
- When mapping the data types, OPC UA elements from the same FB interface or from the same UDT must always be selected for an object. Mapping from different FBs or UDTs to an object is not permitted.

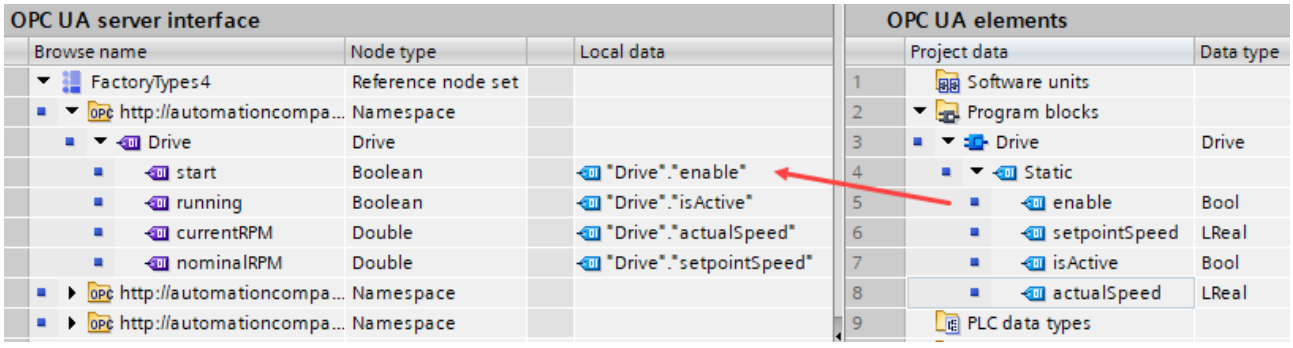
Requirement

- The FB types used, defined in the "Static" area of an FB, must be configured as "Accessible for OPC UA".
- The UDTs used must be configured as "Accessible for OPC UA".
- A nodeset file (XML file) is available with OPC UA data type definitions that match the FB types or UDTs defined in user program (can be mapped).
  Use the "SiOME" tool to create your node set file (Siemens Industry Online Support).
- The user program with the FB instances and UDT usages is available.

Procedure

To map a data type from a reference namespace to an FB type or UDT data type, follow these steps:

1. Select the CPU that you want to use as an OPC UA server.
2. Import the prepared node set file (XML file) with the type definitions as a reference namespace.
   – In the "Add new server interface" dialog, enable the option "Generate OPC UA nodes based on the local data mapping".
   Only when this option is enabled can you map FB types or UDTs by dragging them to the OPC UA type descriptions.
3. Double-click the icon for the server interface of the "Reference namespace" type that you just generated.
   The editor for mapping between OPC UA server interface and OPC UA elements opens. In the properties area of the editor, in the "Mapping of local data" area, the option "Generate OPC UA nodes based on the local data mapping" is enabled. If not, enable the option now.
   In the "Interface name" field, adapt the name of the server interface to be created. A new server interface of the "Companion specification" type with this name is created during the compile.
4. Assign the existing FB types or UDTs to the nodes of the server interface (reference namespace) by dragging the OPC UA element (right side of the editor) to the corresponding node of the server interface (reference namespace, "Local data" column).



5. Compile the project.
   After the compile, the newly generated nodes of the instances are located in the newly generated server interface. STEP 7 creates an object for each instance DB. The generated elements are located under each of these objects.
   Similarly, STEP 7 also creates an object for each global DB that is created when a UDT is instantiated.

Create user program with FB types or UDT

How to create FBs and UDTs is not explained in detail here; for this purpose use the description for creating a user program, e.g. declare block interface and declare PLC data types (UDT).

Consistency check

The consistency check ("Consistency check" button of the editor) also checks the mapping of the data types and updates the display of the data types in the corresponding column of the editor.

10.3.4.11 Notes on configuration limits when using server interfaces

When you use OPC UA server interfaces, you must comply with limits for the following objects in line with the S7-1500 CPU performance class:

- Number of server interfaces
- Number of OPC UA nodes
- Load object data volume
- If you have implemented methods: Number of server methods or server method instances

Configuration limits for OPC UA server interfaces and methods

The table below sets out the configuration limits for S7-1500 CPUs; these must also be taken into account when you compile and load a configuration (up-to-date technical specifications of the CPUs can be found on the Internet (https://support.industry.siemens.com/cs/ww/en/ps/td)).
A violation of configuration limits results in an error message.

Table 10-4 Configuration limits for OPC UA server interfaces

Technical specification value	**CPU 1510SP (F) CPU 1511 (C/F/T/TF) CPU 1512C CPU 1512SP (F) CPU 1513 (F)**	**CPU 1505 (S/SP/SP F/SP T/SP TF) CPU 1515 (F/T/TF) CPU 1515 SP PC (F/T/TF) CPU 1516 (F/T/TF)**	**CPU 1507S (F) CPU 1517 (F/T/TF) CPU 1518 (F)**
Use of imported companion specifications (information models)			
Maximum number of OPC UA server interfaces: • "Companion specification" type • "Reference namespace" type • "Server interface" type	 10 20 10	 10 20 10	 10 20 10
• Maximum number of OPC UA nodes in user-defined server interfaces	1000	5000	30000
• Maximum size of loadable OPC UA server interfaces	1024 KB	5120 KB	8192 KB
Provision of methods			
Maximum number of usable server methods or max. number of server method instances (instructions OPC_UA_ServerMethodPre, OPC_UA_ServerMethodPost)	20	50	100

10.3.5 Providing methods on the OPC UA server

10.3.5.1 Useful information about server methods

Providing user program for server methods

On the OPC UA server of an S7-1500 CPU (as of firmware V2.5), you have the option of providing methods via your user program. These methods can be used by OPC UA clients, for example to start a manufacturing job using the method call of the S7-1500 CPU.

OPC UA methods, an implementation of "Remote Procedure Calls", provide an efficient mechanism for interactions between different communication nodes. The mechanism provides both job confirmation and feedback values so you no longer have to program handshaking mechanisms.

Using OPC UA methods, you can transfer data consistently without trigger bits/handshaking, for example, or trigger specific actions on the controller.

How does an OPC UA method work?

An OPC UA method in principle operates like a know-how protected function block that is called by an external OPC UA client in runtime.

The OPC UA client only "sees" the defined inputs and outputs. The content of the function block, the method or algorithm, remains hidden to the external OPC UA client. The OPC UA client receives feedback on successful execution and values returned by the function block (method), or an error message if execution has not been successful.

As the programmer, you have full control over and responsibility for the program context in which the OPC UA method runs.

Rules for programming a method and runtime behavior

- Make sure that the values returned by the OPC UA method are consistent with the input values provided by the OPC UA client.
- Follow the rules on assigning name and the structure of parameters, and the permitted data types (see description of the OPC UA server instructions).
- Behavior during runtime: The OPC UA server accepts **one** call per instance. The method instance is not available for other OPC UA clients until the call has been processed by the user program or has timed out.

The basic procedure for implementing a user program as a server method is set out below.

Implementing a server method

A program (function block) for implementing a server method is structured as follows:

1. **Querying the server method call with OPC_UA_ServerMethodPre**
   You first call the "OPC_UA_ServerMethodPre" instruction in your user program (i.e. in your server method).
   This instruction has the following tasks:
   – With this instruction you ask the OPC UA server of the CPU whether your server method was called from an OPC UA client.
   – If the method was called and the server method has input parameters, your server method now receives the input parameters.
     The input parameters of the server method come from the calling OPC UA client.

2. **Editing the server method**
   In this section of the server method, you provide the actual user program.
   You have the same options as in any other user program (for example, access to other function blocks or global data blocks).
   If the server method uses input parameters, these parameters are available to you.
   This section of the server method should only be executed if an OPC UA client has called the server method.
   After successful execution of the method, you set the output parameters of the server method if the method has output parameters.
3. **Responding to server method with OPC_UA_ServerMethodPost**
   To complete the server method, call the "OPC_UA_ServerMethodPost" instruction.
   Use the parameters to notify the "OPC_UA_ServerMethodPost" instruction whether or not the user program has been processed.
   If the user program has been successfully executed, the OPC UA server is notified via the relevant parameters. The OPC UA server then sends the output parameters of the server method to the OPC UA client.

Always call the instructions "OPC_UA_ServerMethodPre" and "OPC_UA_ServerMethodPost" as a pair irrespective of whether the user program is processed by both instructions or continued in the next cycle.
You will find an example of a server method implementation in the STEP 7 online help.

Integrating the server method

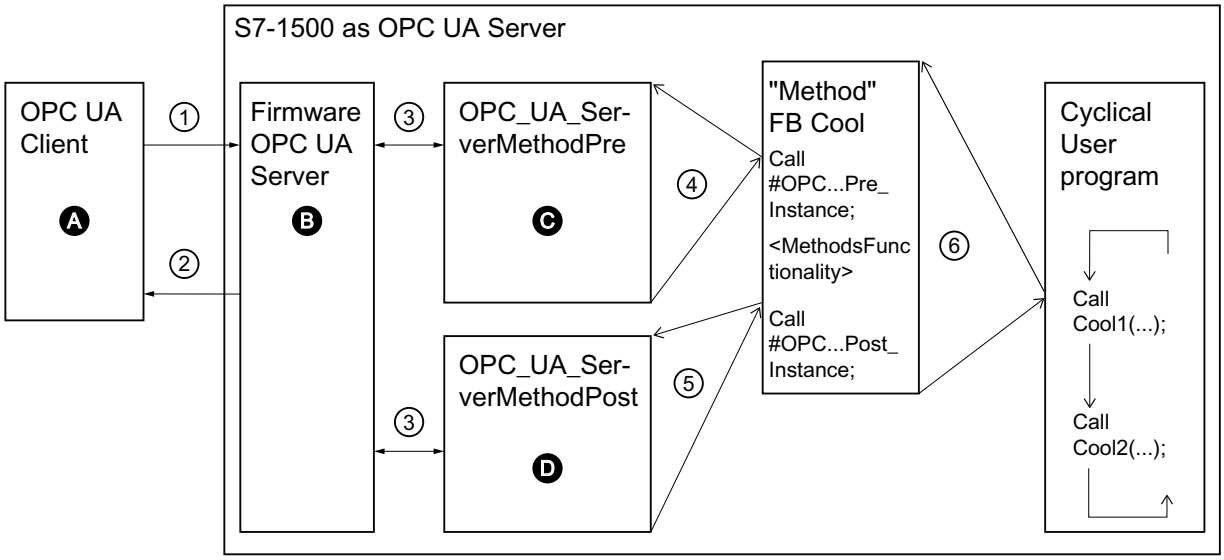
The diagram below shows how an OPC UA client (A) calls the server method "Cool":
The CPU executes the instance "Cool1" of the server method "Cool" in the cyclic user program ⑥.

The CPU first uses the instruction "OPC_UA_ServerMethodPre" to query ④ whether an OPC UA client has called the server method "Cool" ①.

- If the server method has not been called, program execution returns directly to the cyclic user program over ④ and ⑥. The CPU resumes the cyclic user program after "Cool1".
- If the server method has been called, this information is returned to the server method "Cool" over ④. The actual functionality is now executed in the Cool server method, see"<Method Functionality>" in the graphic.

  The server method then uses the instruction "OPC_UA_ServerMethodPost" ⑤ to notify the firmware (B) that the instruction has been executed ③.

  The firmware returns this information over ② to the calling OPC UA client (A).
  The CPU resumes the cyclic user program after "Cool1".



A Call of the server method and management of the "Done" information (method complete)

① Asynchronous call of the server method

② Asynchronous "Done" information for the method called (method complete)

B Wait for OPC UA client calls, management of calls in the queue, forwarding of "Done" information from the cyclic user program to the OPC UA client

③ Data transfer from the OPC UA server to the method instances of the user program and vice versa

C Check whether method has been called.
If it has, forwarding of input data from the OPC UA server to the method instance of the user program and feedback to the method instance that the method has been called ("called")

④ Synchronous call of the instruction OPC_UA_ServerMethodPre as a multi-instance stating the storage area for the input data from the OPC UA server.
The return value indicates whether or not the method has been called by the OPC UA client.

⑤ Check whether the method has been completed or is still active ("busy").

D Check whether the method has been completed.
If it has, the output data of the method instance is forwarded to the OPC UA server and the method instance is notified that the method has been completed. The OPC UA server is notified.

⑥ Call of the method FB (in this case: FB Cool) with the required instance and the process parameters

Figure 10-51 Example: Calling the "Cool" server method

Information about server instructions

The "OPC_UA_ServerMethodPre" and "OPC_UA_ServerMethodPost" are described in detail in the help to the Instructions > Communication > OPC UA > OPC UA server.

10.3.5.2 Boundary conditions for using server methods

Permitted data types

If you provide server methods, observe the following rule:

- Assign the data types as shown below (SIMATIC data type - OPC UA data type). Other assignments are not permitted.

STEP 7 does not check the observance of this rule and does not prevent an incorrect assignment. You are responsible for the rule-compliant selection and assignment of the data types.

You can also use the listed data types, for example, as elements of structures/arrays/UDTs for input and output parameters of self-created server methods (UAMethod_InParameters and UAMethod_OutParameters).

SIMATIC data type	OPC UA data type
BOOL	Boolean
SINT	SByte
INT	Int16
DINT	Int32
LINT	Int64
USINT	Byte
UINT	UInt16
UDINT	UInt32
ULINT	UInt64
REAL	Float
LREAL	Double
LDT	DateTime
WSTRING	String
DINT	Enumeration (Encoding Int32) and all derived data types
User-defined data type required (UDT, user-defined data type) The user-defined data type must be created with the prefix "Union_", for example "Union_MyDatatype". The first element (Selector) in this UDT must have the data type "UDINT".	UNION and all derived data types

Number of implementable server methods and number of arguments

If you implement server methods via your user program, the number of usable methods is limited depending on the CPU type, see the following table (up-to-date technical data of the CPUs can be found in the Internet (https://support.industry.siemens.com/cs/ww/en/ps/td)).

Technical specification value	CPU 1510SP (F) CPU 1511 (C/F/T/TF) CPU 1512C CPU 1512SP (F) CPU 1513 (F)	CPU 1505 (S/SP/SP F/SP T/SP TF) CPU 1515 (F/T/TF) CPU 1515 SP PC (F/T/TF) CPU 1516 (F/T/TF)	CPU 1507S (F) CPU 1517 (F/T/TF) CPU 1518 (F)
Maximum number of usable server methods or max. number of server method instances (OPC_UA_ServerMethodPre, OPC_UA_ServerMethodPost instructions)	20	50	100
Maximum number of arguments per method (More than the specified number of arguments can be configured and loaded into the CPU, but an OPC UA client cannot call the method).	20	20	20

Error message when exceeded

If the maximum number of server methods is exceeded, the OPC_UA_ServerMethodPre or OPC_UA_ServerMethodPost instructions report the error code 0xB080_B000 (TooManyMethods).

Supply of structured data types with nested arrays

If a structured data type (Struct/UDT) contains an array, the OPC UA server does not provide information about the length of this array.

If you use such a structure as the input or output parameter of a server method, for example, you must ensure that the nested array is supplied with the correct length when the method is called.

If you do not adhere to this rule, the method fails with the error code "BadInvalidArgument".

10.3.6 Providing alarms on the OPC UA server

10.3.6.1 Useful information on alarms

Alarms allow you to detect errors in process control in the automation system quickly, to localize them precisely, and to eliminate them. This leads to a significant reduction in downtimes in a plant. The OPC UA information model "Alarms & Conditions" provides a standardized and platform-independent way of message processing.

As of firmware version V2.9, the OPC UA server of an S7-1500 CPU supports the OPC UA information model "Alarms and Conditions". In this way, the OPC UA server provides access to controller alarms.

The following sections describe which alarm types available in SIMATIC are supported at the OPC UA interface of the OPC UA server.

The following sections also describe how you configure the OPC UA server of the S7-1500 CPU for Alarms & Conditions, the main points of how the Alarms & Conditions model is structured with OPC UA, and which special points must be taken into consideration when using alarms from the address space of the OPC UA server as compared to the SIMATIC controller alarms of the CPU alarm system.

Basis for the converting alarms to OPC UA Alarms & Conditions

The Alarms and Conditions information model is specified in the "OPC 10000-9: OPC Unified Architecture Part 9: Alarms & Conditions" specification.

Controller alarms in SIMATIC

The OPC UA server of the S7-1500 CPUs supports the controller alarms listed below, which are available to S7-1500 CPUs. You configure or program these alarms in the usual way, without the need to consider additional rules for use of these alarms by OPC UA clients.

The added benefit of OPC UA Alarms and Conditions is that these alarm types can be displayed not only by HMI devices, web browsers, the CPU display or the TIA Portal, but also by all OPC UA clients that support OPC UA Alarms and Conditions.

- PLC supervision alarms with ProDiag
  You can integrate simple supervisions in your program with just a few configuring steps and without having to change the program code. The configuration of the supervisions is independent of the programming languages of the TIA Portal because only individual operands are supervised, and you do not need any additional programming sections.
- System diagnostics alarms
  Configuration-dependent module events are known by the hardware configuration of the CPU and can be evaluated by connected display devices. They can only be viewed in the alarm editor and cannot be edited.
- Program alarms (Program_Alarm instruction)
  For reporting program synchronous events, program alarms are assigned to one block at a time. They are created in the program editor and edited in the alarm editor (TIA Portal).
- GRAPH alarms
  For GRAPH function blocks, you can also enable alarms; e.g. for interlocks, supervisions and GRAPH warnings (step-time supervisions).

Important information on the alarm types

The following characteristics are significant to the differences in the behavior of alarms:

- Do alarms have a state (e.g. are they incoming, outgoing - with the corresponding time stamps)?
- Do alarms require acknowledgment?

If none of these characteristics apply, which means the alarms have no state and do not require acknowledgment, alarms simply provide information about an event that has occurred ("Fire and Forget"). It is up to the device receiving the alarm to buffer the alarm for further use or only for display.

Alarm class determines acknowledgment behavior

This section covers the setting options for program alarms. You can also set the behavior of the alarms for system diagnostics alarms and PLC supervision alarms (e.g. ProDiag supervision settings) - refer to the linked more information for details.

You can find the setting options for program alarms in the alarm editor (double-click on "PLC supervisions and alarms" in the project tree, select the "Alarms" tab).

For the S7-1500 CPU, you set here via the alarm class whether an alarm needs to be acknowledged or not. In addition to the acknowledgment behavior, when creating a new alarm class, you define the default priority of alarms of this alarm class.

You set whether an alarm has a state or not by means of the "Information only" option, for example, at the alarm type; this results in "Fire-and-Forget" behavior of the alarm.

Here is an example with settings in the alarm editor with different alarm classes ("PLC supervisions and alarms" in the project tree):

- First line "Program_Alarm": Does not require acknowledgment, information only ("Fire and Forget").
- Second line "Program_Alarm_1": Requires acknowledgment and has a state, which means the alarm includes information on whether it is incoming or outgoing.
- Third line "Program_Alarm_2": Does not require acknowledgment and has a state, which means the alarm includes information on whether it is incoming or outgoing.

Alarm types

Name	Type	ID	Location	Alarm text	Info text	Alarm class	Acknowle...	Information only
Program_Alarm	PLC alarm		AlarmType	myText	myINFO	No Acknowledgement	☐	☑
Program_Alarm_1	PLC alarm		AlarmType			Acknowledgement	☑	☐
Program_Alarm_2	PLC alarm		AlarmType			No Acknowledgement	☐	☐

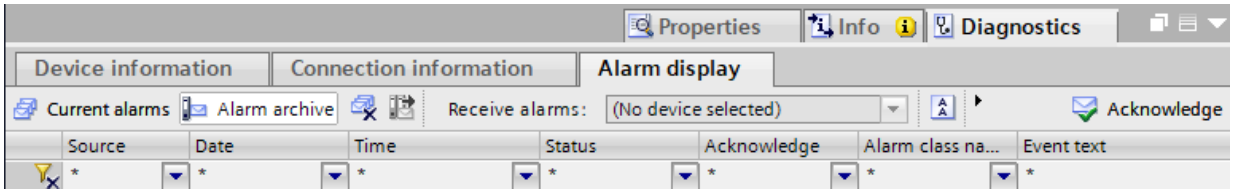
Display of alarms in the TIA Portal

During runtime, you have the option of viewing the alarms in the TIA Portal: The alarm display is located directly under the alarm editor ("Diagnostics" tab > "Alarm display" tab).

The following applies to the state and acknowledgment behavior:

- When you click on the "Current alarms" button, alarms that have been recently incoming, outgoing or acknowledged are displayed. Only alarms with a state and alarms requiring acknowledgment are displayed here. You can also acknowledge an alarm requiring acknowledgment (blue font) in this view via the shortcut menu or using the "Acknowledge" button.
- If you want to observe the chronological development (e.g. alarm came in, was acknowledged and went out), you need to click the "Alarm archive" button. The three events that belong to this alarm are listed one after another only in this view. You can only find the current state in the "Current alarms" view.
- Info reports (alarms with the "Information only" property) are only displayed in the "Alarm archive" view. Because these alarms are only triggered once and not buffered, they do not appear in the "Current alarms" view.
- PLC supervisions are also shown in the alarm display.
- System alarms usually belong to the "No Acknowledgement" alarm class with the "Information only" option. These alarms are logged in the diagnostics buffer in the CPU and thus allow the sequence of system alarms to be analyzed over a limited period. In contrast, operating state changes that are also logged in the diagnostics buffer have a state, which means it is indicated that or when a CPU went into STOP state and if or when

it exited this state again, for example, entered RUN state. This information is displayed with the states "incoming/outgoing".



Provision of controller alarms by the OPC UA server

When an OPC UA client wants to receive alarms of the S7-1500 CPU, it needs to subscribe to OPC UA events (MonitoredEventItems).

For this purpose, the address space of the OPC UA server of the S7-1500 CPU contains corresponding nodes that are "Event-Notifiers" and create a subscription for the OPC UA clients to receive the alarms.

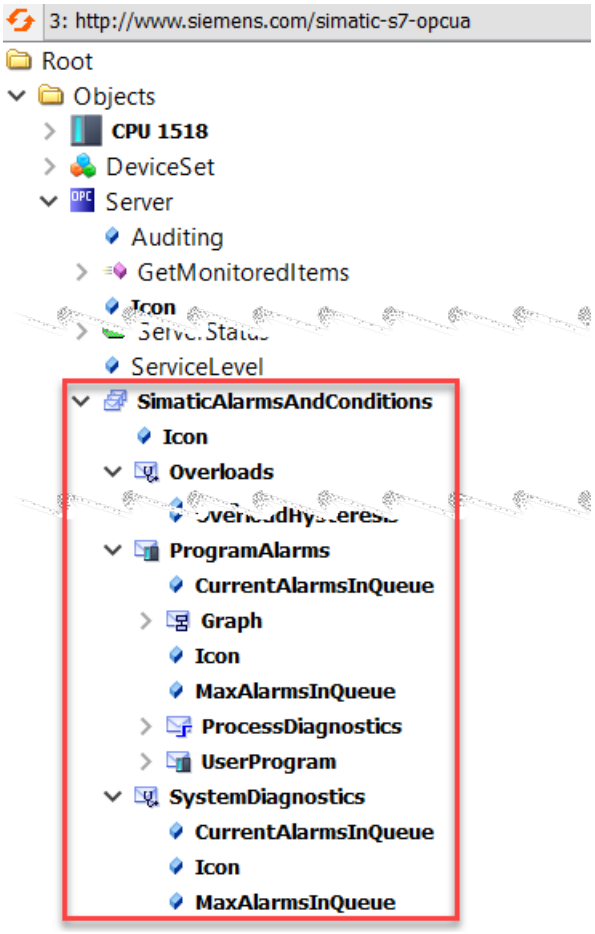
For completeness, we also mention that further type definitions are contained in the server address space for this purpose as nodes under "Types". Type definitions under "BaseEventType" and "ConditionType" ensure that the fields used by SIMATIC alarms are also provided in the OPC UA server.

After the activation of OPC UA Alarms and Conditions (CPU properties in the hardware configuration), the OPC UA address space of the S7-1500 CPU thus reflects the various alarm types (controller alarms) as described above:

- ProcessDiagnostics
  Corresponds to the PLC supervision alarms with ProDiag
- SystemDiagnostics
  Corresponds to the system diagnostics alarms
- UserProgram
  Corresponds to the program alarms
- Graph
  Corresponds to GRAPH alarms

By selecting the node for a subscription, you determine which alarm types are received by the OPC UA client. For example, the "Server" node enables receipt of all alarms, the "UserProgram" node only the receipt of program alarms.

Details about the OPC UA model for Alarms and Conditions are given in the next selection, especially for the "Overloads" node, you can find more information here: Handling memory limits for OPC UA Alarms and Conditions (Page 289).



More information on the alarm types

The concepts and configuration options for controller alarms are not described further here. For information on alarm configuration, alarm display and the associated instructions, such as "Progam_Alarm", refer to the STEP 7 online help.

10.3.6.2 OPC UA Events

The basic concepts for alarm processing in OPC UA are expanded on here - the basic concept of "Events" is covered here. The terms used in the various parts of the OPC UA specification have been retained here.

Properties of events

In the address model of the OPC UA server, as of CPU firmware version V2.9, you not only have the option to access PLC tags (read, write) via nodes and to use methods - you can also receive events or alarms via nodes. In OPC UA terminology, these are called "events".

An event includes an event text (Message), time stamp (Time) and event source (SourceNode).

The information supplied with an event from the server depends on the event type. OPC UA defines a BaseEventType in part 5 of the specification (Information Model).

Other event types that provide different alarm behavior are derived from the BaseEventType. This type information of the different event types is visible in the address space of an OPC UA server ("Types" folder). This also applies, for example, to the event types of "Conditions" and "Alarms", which are discussed in the next section.

The OPC UA specification defines for the BaseEventType and for derived EventTypes which properties (fields) of an event are mandatory and which are optional.

The following figure shows the hierarchical structure of BaseEventType.

The following sections show how specialized EventTypes are derived from the root of the derivation hierarchy, the BaseEventType. The SIMATIC-specific derivations ensure that the information supplied in SIMATIC with an alarm and displayed on an HMI device, for example, can also be subscribed to by an OPC UA client in the address space of the OPC UA server.

An event itself is not available as a node in the address space. Events are only triggered by nodes or objects that have the "Event-Notifier" property. These nodes are often also referred to as event signaling objects. Only nodes with this property can be specified as EventMonitoredItem in a subscription to receive corresponding events in the client.

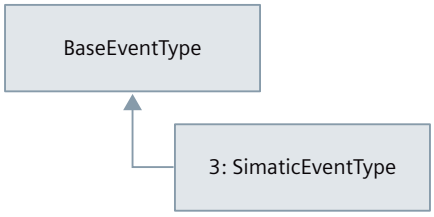
Nodes that can trigger events with an S7-1500 CPU are objects such as "Server", the "SimaticAlarmsAndConditions" object below it, and the three objects below that, ProcessDiagnostics, SystemDiagnostics and UserProgram. The "EventNotifier" attribute is set for these objects in the address space of the OPC UA server of the CPU.

Definition of SimaticEventType

The figure below shows that the type "SimaticEventType" is derived directly from BaseEventType.

BaseEventType is the basic type definition for events with OPC UA.

All event types for OPC UA can be defined, directly or indirectly, based on BaseEventType.



The "SimaticEventType" type is defined in the SIMATIC namespace (http://www.siemens.com/simatic-s7-opcua).

SimaticEventType has all properties of BaseEventType as well as the special properties which are an image of the field structure of SIMATIC alarms.

Description of the event fields for SimaticEventType

The following table contains information on the fields of SimaticEventType for alarms of the type "Information only". Fields that are optional according to OPC UA and are not used by the OPC UA server of the CPU are omitted. You can also find a general description of the fields in the specification OPC 10000-5: OPC Unified Architecture, Part 5: Information Model, Release 1.04.

BrowsePath	DataType	Explanations
EventId	ByteString	Unique Event ID of the event
EventType	NodeId	Node ID of the event type
Time	UtcTime	Time stamp of the event (Event occurrence)
ReceiveTime	UtcTime	Current time stamp at which the OPC UA event was generated.
Message	LocalizedText	Event text
Severity	UInt16	Priority of the alarm from SIMATIC (0..16), spread into a range from 1..1000 with OPC UA, see following table. The priority indicates the urgency with which the event needs a response.
3:AdditionalText_01	LocalizedText	Optional additional text 1
...	...	...
3:AdditionalText_09	LocalizedText	Optional additional text 9
3:AssociatedValue_01	3:SimaticAssociatedAlarmValue	Optional associated value 1 (not for system diagnostics)
...	...	...
3:AssociatedValue_10	3:SimaticAssociatedAlarmValue	Optional associated value 10 (not for system diagnostics)
3:InfoText	LocalizedText	Info text
3:ID	UInt16	Alarm number - A number (ID) assigned by the system that is unique within the CPU and identifies an alarm.
3:DisplayClass	UInt16	Display class (used by HMI devices. Determines the events displayed on specific HMI devices.
3:GroupID	UInt8	Acknowledgment group for alarms that are acknowledged together.

Assignment of Priority (SIMATIC) - Severity (OPC UA)

The following table shows how the 17 priorities that you can assign to alarms in the SIMATIC environment are mapped to the 1000-level Severity with the OPC UA server of the S7-1500 CPU.

This assignment depends on the manufacturer. Different assignments may apply to other devices.

OPC Range	Priority 0..16 (SIMATIC)	Severity 1..1000 (OPC UA)
HIGH (667 – 1 000)	16	1000
	15	930
	14	860
	13	790
	12	720
MEDIUM (334 – 666)	11	650
	10	600
	9	550
	8	500
	7	450
	6	400
	5	350
LOW (1 – 333)	4	300
	3	225
	2	150
	1	75
	0	1

10.3.6.3 OPC UA conditions and OPC UA alarms

The following goes further in depth about the basic concepts for OPC UA Conditions and OPC UA Alarms based on the explanations of events in the previous sections. Again, the terms used in the various parts of the OPC UA specification have been retained here.

Properties of Conditions

A prerequisite for understanding is the concept of "Events" in OPC UA.

In OPC UA, if an event alarm object provides status information in addition to its ability to fire Events, we speak of Conditions. Conditions represent a state of a system or one of its components. Basic states are "enabled" and "disabled", other state definitions are also possible.

In turn, interested OPC UA clients are notified of state changes by means of events (Condition Events).

An example of a Condition is state information, for example, that a device requires maintenance.

Properties of Alarms

However, the properties of ConditionType are not sufficient to completely map the characteristics of SIMATIC alarms in the OPC UA server.

From the ConditionType, which is derived from the BaseEventType, OPC UA defines further derived event types such as AcknowledgeableConditionType and AlarmConditionType.

AcknowledgeableConditionType supplements the properties of ConditionType with the "Acknowledgeable" characteristic (AckedState).

AlarmConditionType thus adds the "ActiveState" characteristic to the properties of ConditionType and the AcknowledgeableConditionType. In SIMATIC terminology, this is an incoming alarm. The ActiveState signals that the situation, which the Condition represents, is currently present or has occurred.

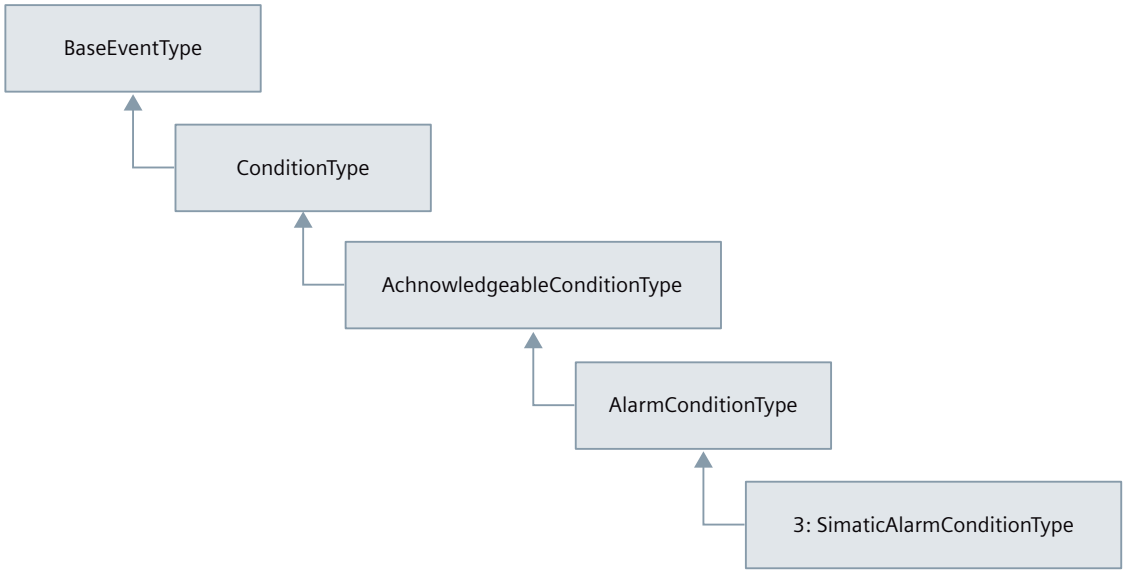
Example: A temperature has exceeded a limit. If "ActiveState" is not set, the situation that represents the condition no longer exists - this is usually referred to as a "normal state". In SIMATIC terminology, this corresponds to an outgoing alarm.

In OPC UA, other statuses such as SilenceState and ShelvingState are defined, but these are not relevant to mapping to the SIMATIC alarm system and will therefore not be described further here.

The SimaticAlarmConditionType is derived from the AlarmConditionType and contains all event fields to map the state and acknowledgment information of SIMATIC messages.

Definition of SimaticAlarmConditionType

The following figure shows how events of the type "SimaticAlarmConditionType" are defined by a series of expansions to the OPC UA "BaseEventType".



Description of the event fields for SimaticAlarmConditionType

The following table contains information about the fields of SimaticAlarmConditionType for stateful and acknowledgeable alarms, which are added to the event fields such as SimaticEventType. Fields that are optional according to OPC UA and are not used by the OPC UA server of the CPU are omitted. You can also find a description of the fields in the

specification OPC 10000-9: OPC Unified Architecture, Part 9: Alarms & Conditions, Release 1.04.

BrowsePath	DataType	Explanations
ConditionClassId	NodeId	Node IDs for SystemConditionClassType, ProcessConditionClassType or BaseConditionClassType are possible
ConditionClassName	LocalizedText	Display name of the ConditionClassId
Retain	Boolean	Indicates that the alarm is still relevant for OPC UA clients (set if alarm is still pending and not acknowledged).
Comment	LocalizedText	• Most recent comment entered using the "AddComment" or "Acknowledge" method. • ZERO after a server restart and if no comment was entered.
Comment.SourceTimestamp	UtcTime	Time stamp for the last change of the comment field
AckedState	LocalizedText	"Acknowledged" or "Unacknowledged"
AckedState.Id	Boolean	Set, if acknowledged
AckedState.TransitionTime	UtcTime	Time at which the alarm was acknowledged. ZERO if not acknowledged or not acknowledgeable.
ActiveState	LocalizedText	"Active" or "Inactive"
ActiveState.Id	Boolean	Set when "active"

10.3.6.4 Activating Alarms and Conditions

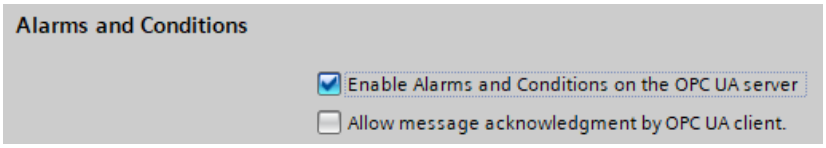
Requirements

- S7-1500 CPU firmware version V2.9 or higher
- Runtime license for OPC UA were purchased according to the license specifications and set in the CPU properties

Procedure

To activate alarms through OPC UA Alarms and Conditions, proceed as follows:

1. In the CPU properties, go to the "OPC UA > Server > General" area.
2. Select the "Enable Alarms and Conditions on the OPC UA server" option.
   The corresponding types and objects that can trigger events only become visible in the address space when the option is activated.
3. If required, also activate the option "Allow message acknowledgment by OPC UA client".
   In this case, every connected OPC UA client can acknowledge an alarm requiring acknowledgment with the "Acknowledge" method.



Recommendation: Activate diagnostics "Requests of a remote OPC UA client failed"

When the OPC UA server cannot allocate sufficient memory, it is not possible to generate OPC UA alarms in this state; a message loss for OPC UA clients is possible.

Therefore, you should activate the diagnostics "Requests of a remote OPC UA client failed" to diagnose this state (Properties of the CPU > OPC UA > Server > Diagnostics).

In addition, you should also activate the option "Summarize diagnostics in case of high message volume".

As soon as sufficient memory is available again, OPC UA clients should call the ConditionRefresh method to receive the current state of the alarm system.

More information

You can find information on methods for OPC UA Alarms and Conditions in the section Methods for OPC UA Alarms and Conditions (Page 286).

You can find information on failed requests of a remote client in the section Request of a remote client failed (Page 297).

10.3.6.5 Subscribing to events of an OPC UA server

Subscribing to all events via the "Server" node

OPC UA servers provide events via the "Server" node and lower-level nodes. When OPC UA clients subscribe to the "Server" node, they receive all events and alarms of the OPC UA server.

The "Server" node is located in the "Objects" folder below "Root".

OPC UA servers inform OPC UA clients which event types they use (under "Root > Types > EventTypes" in the address space).

Filter options for events

OPC UA clients can choose and only subscribe to specific nodes under the "Server" node and thus to specific event types, for example, only the "UserProgram" node. This reduces the number of events from the OPC UA server to program alarms.

Another way of filtering is to select the event fields, known as "Select Clause" in OPC UA terminology.

This means that in the subscription, the OPC UA client makes a selection of the event fields in addition to the event alarm object (e.g. the "UserProgram" node). You select the event fields via the browse name of the corresponding field.

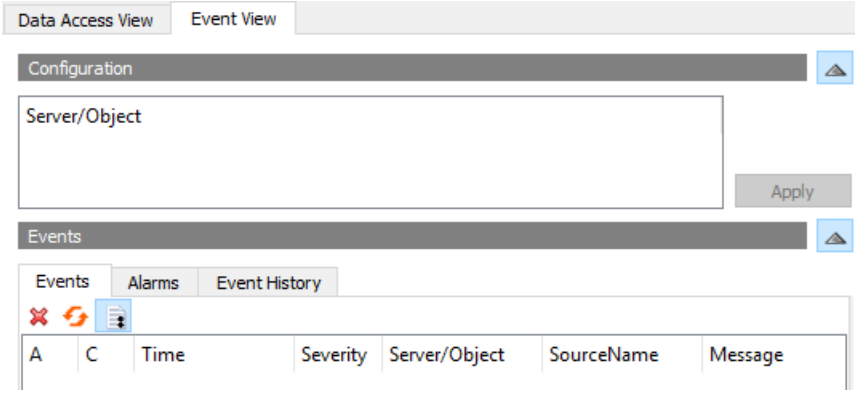
OPC UA also defines so-called "Where Clauses". A Where Clause in the event filter is used to further limit the number of events that are supplied by the OPC UA server for the selected object, for example, by filtering to a severity range.

Example client UaExpert

The example of the UaExpert OPC UA client shows how events of an OPC UA server can be received via a subscription. Here is the most important information on the displayed events/alarms:

- Event View is a separate view of events in addition to the Data Access View.
- The "Configuration" area contains the selected event signaling object with the fields for the Select Clause. A configuration of Where clauses is currently not possible in UaExpert.
- In the "Events" area, "Events" tab: Corresponds to the TIA alarm view with activated "Alarm archive" button; events of the "Information only" category and outgoing alarms are also visible there, because UaExpert buffers them in the background for display. These events are not visible in the "Alarms" tab.
- In the "Events" field, "Alarms" tab: Corresponds to the TIA alarm display with activated "Current alarms" button; alarms are visible here with their status, e.g. "active" (corresponds to "incoming") and these alarms can also be acknowledged using the shortcut menu. Outgoing alarms are no longer visible in this view.

In the columns of the Event area, a selection of event fields is offered, for example, the event text (Message) and whether the alarm was acknowledged (A=Acknowledged).



Special features of the display of alarms via the OPC UA server of the CPU

The following once again summarizes the special features of the alarm display via OPC UA Alarms and Conditions for the current status.

Topic	Explanation
Comment	Via OPC UA, you can add a comment to a alarm using the "AddComment" method or the "Acknowledge" method. This comment is no longer available after a server restart.

Topic	Explanation
Pending alarms are not lost after a server restart	The OPC UA server supports the "ConditionRefresh" method with which it makes the current state of the system available to the OPC UA client, for example, after download of a new data block (requires server restart and re-establishment of the connection).

10.3.6.6 Processing associated values of alarms

You can specify placeholders for SIMATIC alarms. With placeholders you can integrate up to 10 associated values (SD_1 to SD_10) into the alarm text. Placeholders can also be specific text list entries.

When you are using alarms with placeholder, you must observe the following rules:

- Placeholders that represent values in the alarm are only inserted automatically for system diagnostic alarms or security event alarms. For other categories of alarms (e.g. program alarms), the placeholders for values are not resolved. The OPC UA clients must resolve these alarms.
- Placeholders that reference text lists are resolved by the CPU (format, e.g.: %t#<name of the text list>).

Example of how values and placeholders are assigned through UaExpert

1. Make sure that you have selected all fields that you need in the UaExpert configuration. Note that each field that is not needed causes a communication load. Therefore, you should avoid a complete selection as shown in the example below.

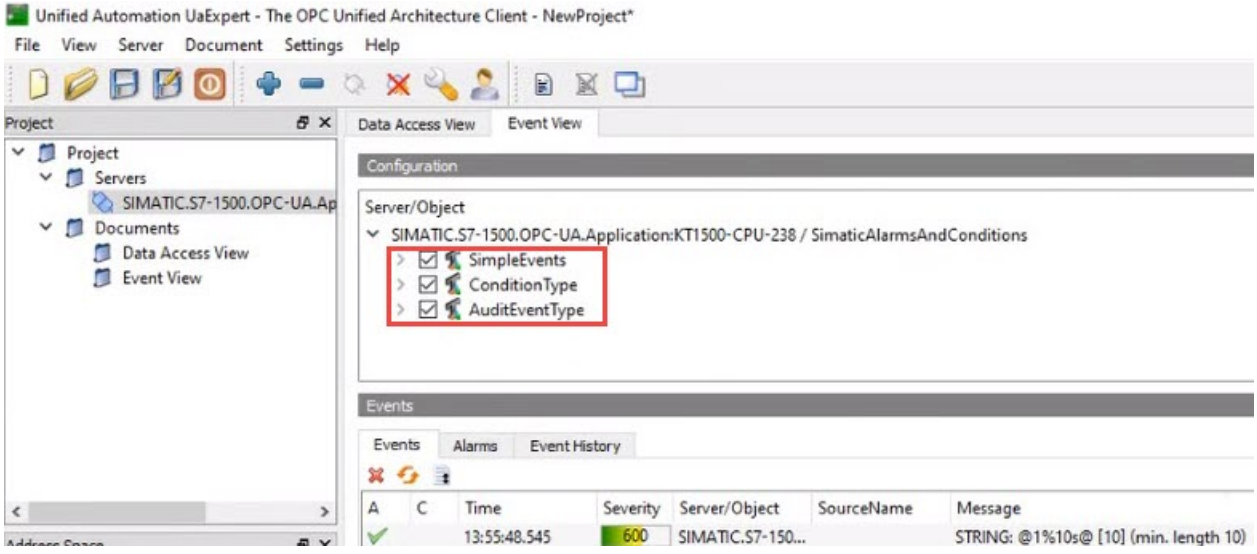


Figure 10-52 ServerObject-Configuration_neutral120

2. In the "Events" tab of UaExpert, you select the alarm with the integrated associated values. In the "Details" area of the alarm, you will find the value that is to be integrated into the alarm.
   Example: "AssociatedValue_01" is assigned to SD_1 (format @1% ...@).
   You will find explanations on the formats for associated values in the TIA Portal information system (e.g. by searching for "Example of associated values").

Support of simple data types as associated values

The field type from "AssociatedValue_01 to ..._10 is of the Union type and is limited to some simple types. The OPC UA data type is SimaticAssociatedAlarmValue. The following simple data types are supported:

3:AssociatedValue_01	SimaticAssociatedAlarmValue
Switch Field	2
Boolean	True
3:AssociatedValue_02	SimaticAssociatedAlarmValue
Switch Field	6
SByte	123
3:AssociatedValue_03	SimaticAssociatedAlarmValue
Switch Field	3
Int16	1234
3:AssociatedValue_04	SimaticAssociatedAlarmValue
Switch Field	4
Int32	12345
3:AssociatedValue_05	SimaticAssociatedAlarmValue
Switch Field	7
Byte	123
3:AssociatedValue_06	SimaticAssociatedAlarmValue
Switch Field	8
UInt16	1234
3:AssociatedValue_07	SimaticAssociatedAlarmValue
Switch Field	11
Float	1
3:AssociatedValue_08	SimaticAssociatedAlarmValue
Switch Field	12
Double	2
3:AssociatedValue_09	SimaticAssociatedAlarmValue
Switch Field	9
UInt32	12345
3:AssociatedValue_10	SimaticAssociatedAlarmValue
Switch Field	13
String	HelloWorld

Figure 10-53 UnionDataTypes_neutral120

Mapping to SIMATIC data types

The following assignment SIMATIC data type => OPC UA data type applies:

Supported data types for SD_1 to SD_10	Mapping to OPC UA
BOOL	Boolean
BBOOL	Boolean
BYTE	Byte

Supported data types for SD_1 to SD_10	Mapping to OPC UA
CHAR	Byte
SINT	SByte
USINT	Byte
WORD	UInt16
WChar	UInt16
INT	Int16
UINT	UInt16
DWORD	Uint32
DINT	Int32
UDINT	Uint32
REAL	Float
LREAL	Double
String	String
WString	String

10.3.6.7 Methods for OPC UA Alarms and Conditions

The OPC UA specification Part 9 (OPC 10000-9: Alarms & Conditions) defines methods for OPC UA servers to enable OPC UA clients to react to state changes, for example.
In the following, the methods are described that are supported by the OPC UA server of the S7-1500 CPU with their special features.

Requirement

Using the relevant methods for the Alarms and Conditions functionality requires the following:

- Alarms and Conditions is activated
- For the "Acknowledge" method, the acknowledgment of alarms by OPC UA clients must be allowed on the server side.

Methods for OPC UA Alarms and Conditions

In the following, the methods are briefly explained with the special features and restrictions caused by the implementation for the OPC UA server of the S7-1500 CPU.
The methods are visible in the type space.
The OPC UA specification listed above contains the general description.
You can find a detailed description of the individual methods below this overview table.

Method	Description
Acknowledge	Method for acknowledging an alarm object that is uniquely identified by a EventId.

Method	Description
ConditionRefresh	Method for requesting an update of all alarm objects (in SIMATIC language: updating of all pending alarms). All monitored items of the subscription are updated. Synchronization of pending alarm objects from the OPC UA server of the CPU is indicated e.g. in the following situations: • Connecting for the first time or resuming the connection (after interrupting the communication) • Screen change on an operator screen of an HMI device
AddComment	Method for adding comments to alarm objects.

Calling the "Acknowledge" and "AddComment" methods

Method calls in OPC UA use MethodId and ObjectId.

In the case of an alarm object, ObjectId is the node ID for the instance of the alarm object.

Since the address model of Simatic Alarms and Conditions does not provide instances for alarm objects, the OPC UA specification provides in this case that the OPC UA client uses the ConditionId as ObjectId.

You can find information on how to determine the ConditionId using a SimpleAttributeOperands in the SelectClause of the event filter, see also the OPC UA specification Part 9 (OPC 10000-9: Alarms & Conditions):

Name	Type	Description
SimpleAttributeOperand		
typeId	NodeId	*NodeId* of the *ConditionType Node*
browsePath[]	QualifiedName	empty
attributeId	IntegerId	Id of the *NodeId Attribute*

Acknowledge

The Acknowledge method (MethodId: i=9111) has the following parameters:

Parameter	Data type	Description
[in] EventId	ByteString	EventId identifies a specific event notification. Only events whose AckedState.Id field has the value "False", can be acknowledged with the "Acknowledge" method.
[in] comment	LocalizedText	Text for commenting the acknowledgment, e.g. by an operator. See also supplementary description of the "AddComment" method.

Method Result Codes

Result Code	Description
Good	Method was successfully executed.

Result Code	Description
BadNotSupported	Method cannot be called because the option for acknowledging for Alarms and Conditions by OPC UA clients is deactivated in the CPU properties for OPC UA.
BadConditionBranchAlreadyAcked	Acknowledgment has already been made.
BadNodeIdUnknown	Method was called with the wrong ConditionId (see notes on ObjectId).
BadEventIdUnknown	Method was called with the wrong EventId.

ConditionRefresh

The ConditionRefresh method (MethodId: i=3875) has the following parameters:

Parameter	Data type	Description
[in] SubscriptionId	Uint32	SubscriptionId of the subscription to be updated.

Method Result Codes

Result Code	Description
Bad_SubscriptionIdInvalid	The SubscriptinId is not valid.
Bad_RefreshInProgress	The "ConditionRefresh" is currently running.
Bad_UserAccessDenied	The method "ConditionRefresh" runs in the context of a wrong session. This means that the subscription belongs to a different session.

NOTE

ConditionRefresh2 method

The OPC UA server of the S7-1500 CPU does not support the ConditionRefresh2 method which can specifically synchronize a monitored item (MonitoredItem) in a subscription. In this case, the OPC UA Server returns the result code "Bad_MethodInvalid". Use the method "ConditionRefresh" instead.

AddComment

You have the possibility to add comments to Alarms- objects of the SimaticAlarmConditionType type because the support of comments is mandatory for OPC UA Alarms and Conditions .

A comment was save in the "Comment" event field.

The following time stamp event fields belong to the comment:

- "Comment.SourceTimestamp" for the time when the comment is transferred to the CPU
- "Time" for the time when the Alarms object was modified

When the "AddComment" method is called, "Time" and "Comment.SourceTimestamp" are identical.

Special features of Alarms and Conditions comments for the OPC UA server of the CPU

The AddComment method (MethodId: i=9029) has the following parameters:

Parameter	Data type	Description
[in] EventId	ByteString	EventId identifies the event notification for which a state was reported.
[in] comment	LocalizedText	Text for commenting the specified Alarms object.

Method Result Codes

Result Code	Description
Good	Method was successfully executed.
BadNodeIdUnknown	Method was called with the wrong ConditionId (see notes on calling the methods "Acknowlege" and "AddComment").
BadEventIdUnknown	Method was called with the wrong EventId.

Special features of Alarms and Conditions comments for the OPC UA server of the CPU

You have the possibility to add comments to alarm objects of the "SimaticAlarmConditionType" type with the AddComment method. A comment is also set when the Acknowledge method is called. The "AddComment" method can be called several times.

- A comment was save in the "Comment" event field. The "Comment.SourceTimestamp" indicates the last time at which a **comment** was set.
- The "Time" time stamp marks the **last modification time of the alarm object**.

When the "AddComment" method is called, "Time" and "Comment.SourceTimestamp" are identical.

When the "Acknowledge" method is called, the two time stamps may differ, since the acknowledgment is asynchronous.

The support of comments for OPC UA Alarms and Conditions is mandatory. The SIMATIC alarm system does not know corresponding comments for alarms. Therefore, some special features have to be considered:

- There is only one comment:
  There is only one comment for an alarm object, so that an existing comment is always overwritten when several method calls are made in succession.
- Lifetime and time stamp:
  Comments are only stored at the current alarm object. If the alarm object no longer exists, e.g. after a server restart, the comment no longer exists either. The corresponding "Comment" and "Comment.SourceTimestamp" event fields are then reset (zero).
  The "Time" event field is then set as if the method call "AddComment" did not exist.
  Example: If you comment on an unacknowledged Alarms object, the "Time" event field receives the time of this comment change. After a server restart, the "Time" event field does not show the time when the comment was set, but the time when the corresponding Event arrived.

10.3.6.8 Handling memory limits for OPC UA Alarms and Conditions

The OPC UA server of the S7-1500 CPU has product-specific limited memory capacity for the "Alarms and Conditions" function (see CPU specifications).

Two memory pools for different categories of alarms are available:

- Pool only for ProgramAlarms (corresponds to program-related alarm originators (producers) such as program alarms via Program_Alarm, ProDiag, Graph)
- Pool only for SystemDiagnostics (corresponds to system diagnostic alarms)

Under unfavorable conditions (e.g. alarm burst) the CPU cannot make all pending alarms (ProgramAlarms or SystemDiagnostics) from the SIMATIC alarm area available to the OPC UA Alarms and Conditions system. However, alarms are not lost in this case.

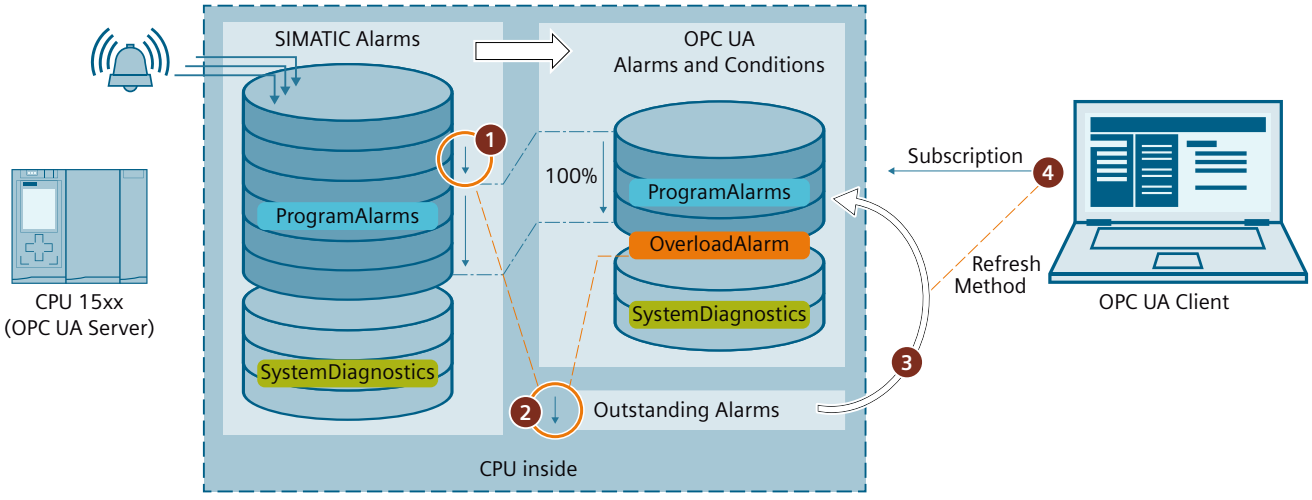
You have the possibility to react to this overload event in the user program. According to your application you can use the "ConditionRefresh" method to make alarms that "did not make it into the OPC UA Alarms and Conditions system" available to the OPC UA Alarms and Conditions system again.

Requirement

- Alarms and Conditions is activated
- Event subscriptions are set up in the OPC UA client

Principle

The following figure shows a simplified process for temporarily storing ProgramAlarms to make them available again at another time for the OPC UA Alarms and Condition System. The nodes mentioned in the caption are visible in the following image of the address model.



① Number of active alarms is too high to make all alarms accessible via OPC UA Alarms and Conditions

② Overloads alarm (overload alarm) is triggered. This overload alarm is active until the following situation occurs:

- No more alarms are pending for the OPC UA Alarms and Conditions system (OutstandingProgramAlarms = 0) and
- Number of alarms in the OPC UA Alarms and Conditions system < Hysteresis-cleaned maximum value for OPC UA alarms (= MaxAlarmsInQueue - OverloadHysteresis)

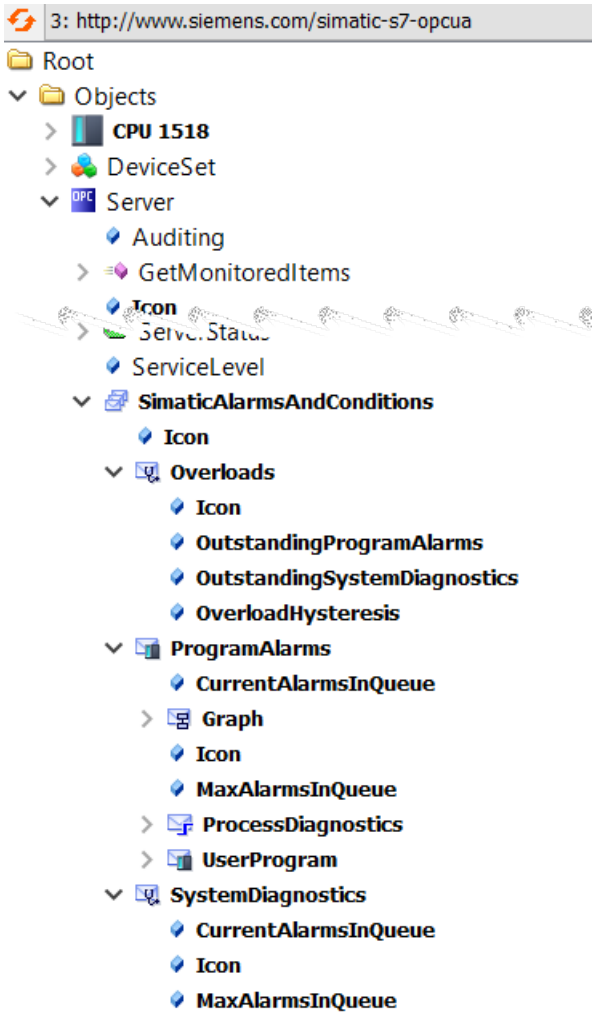
Alarms that are not available in the OPC UA Alarms and Conditions system due to the overload situation are buffered by the CPU as "OutstandingAlarms".

③ When an OPC UA client executes the ConditionRefresh method, not only are all alarm objects of the relevant subscription synchronized, but also the alarms outstanding for OPC UA Alarms and Conditions (OutstandingAlarms) are transferred to the Alarms and Conditions memory area - until the maximum number of alarms is reached. "Oldest" alarms are transferred first. After that every subscription to these alarms receives the transferred alarms - not only the OPC UA client that called the ConditionRefresh method.

④ The OPC UA client controls the handling of pending alarms via the information of the Overloads nodes.

Address model for Alarms and Conditions

The following figure shows the nodes of the OPC UA Alarms and Conditions address model.



Special features

- When pending alarms go out or are acknowledged, they no longer enter the OCP UA Alarms and Conditions system area via the ConditionRefresh method. They are then "invisible" to OPC UA Alarms and Conditions and thus to the connected OPC UA clients. This fact influences e.g. statistical evaluations of alarm progressions.
- To avoid a high alarm frequency for the Overloads Alarm if the number of alarms oscillates around the maximum value, the limit for triggering the alarm is higher than the limit for canceling this alarm: The value for this difference is displayed in the "OverloadHysteresis" node.
  Example: Maximum number of alarms: 200, OverloadHysteresis: 3.
  Overloads alarm is triggered starting with 200 alarms and is only canceled if there are fewer than 197 alarms. If the number of alarms increases again, it will be triggered again when 200 alarms are exceeded.

10.3.7 Using diagnostics options

10.3.7.1 Diagnostics of the OPC UA server

Online diagnostics of the OPC UA server

The S7-1500 CPU OPC UA server can be diagnosed online with standard OPC UA clients, such as UaExpert.
The diagnostic information is subdivided into the following areas:

- Server Diagnostics
- Sessions Diagnostics
- Subscriptions Diagnostics

In the address space of the server, for example, the following nodes are available with diagnostic information:

- **ServerDiagnosticsSummary**: Server diagnostics summary
  - CurrentSessionCount: Number of active sessions
  - SecurityRejectedSessionCount: Number of sessions rejected due to mismatching end point security settings between client and server
- **SessionsDiagnosticsSummary**: Session diagnostics summary
  - ActualSessionTimeout: Set time that a session lasts, e.g. in the event of disconnection
- **SubscriptionsDiagnosticsArray**: ARRAY with one element per subscription for each session

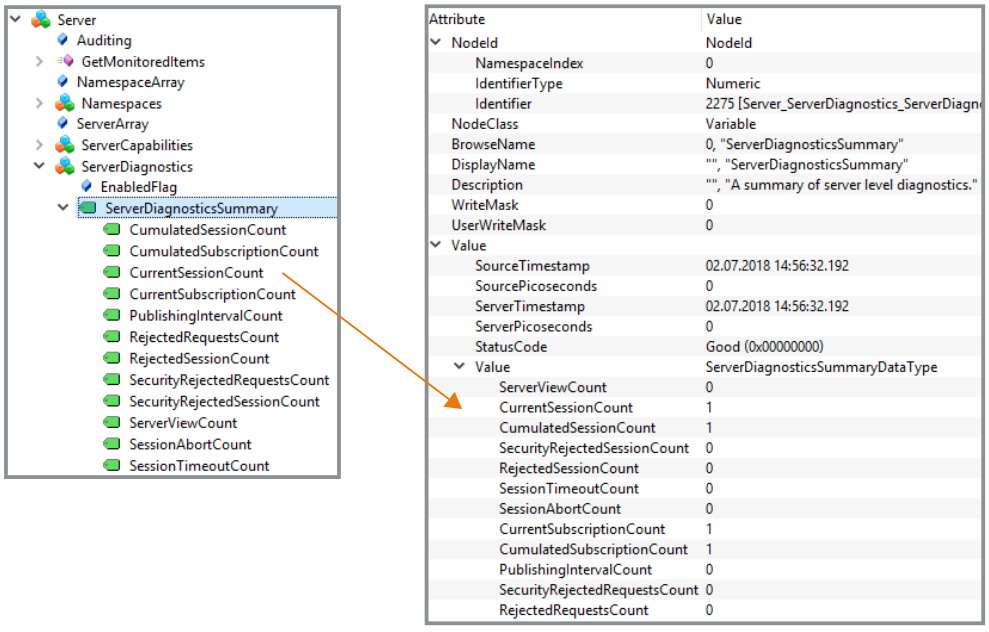


Figure 10-54 Server diagnostics

The SessionsDiagnosticsSummary node also shows the properties of the client application accessing the server within the session.

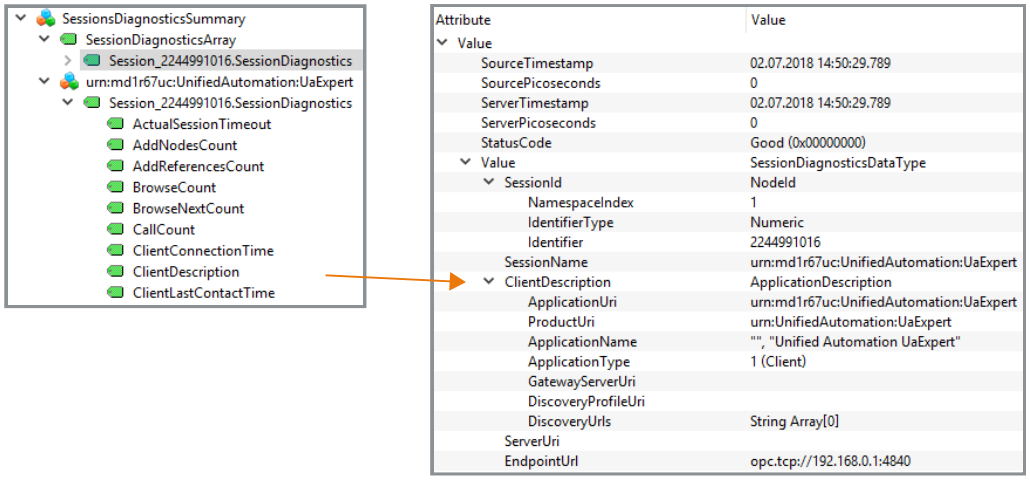


Figure 10-55 Sessions diagnostics with the properties of the client application

Diagnostics of the connection between client and server

To diagnose the status of the connection during program runtime in the client, use the following instruction:

OPC_UA_ConnectionGetStatus: Read connection status.

10.3.7.2 Running diagnostics for OPC UA servers in the program

From STEP 7 (TIA Portal) V18 onwards, you can access nodes in the OPC UA address space of an S7-1500 CPU (firmware version V3.0 onwards) to evaluate the contents for diagnostic purposes in the program.

Functional principle

In the local address space of the CPU, there are numerous nodes where the OPC UA server of the CPU stores data and states. The "OPC_UA_ReadList" instruction enables you to access this information and evaluate it in the user program.

Example: "ServerState" is a node in the address space of the CPU that contains values for the server state or for state transitions (Running, Shutdown, Failed, etc.).

You do not use the instruction as a client instruction, but instead as an instruction for reading nodes of the own local OPC UA address space. In this regard, special rules and requirements apply to this application case.

More information

You can find more information on calling the "OPC_UA_ReadList" instruction for diagnostic purposes in the TIA Portal help, topic "Diagnosing OPC UA servers with OPC_UA_ReadList".

10.3.7.3 Server state transition diagnostics

Information on the server state

S7-1500 CPUs as of firmware version V2.8 are able to create an entry in the diagnostic buffer upon state changes of the OPC UA server.
The diagnostic buffer displays the new state.
The cause of the state change is also displayed, such as download to the CPU, POWER OFF - POWER ON transition, user program instruction or service request from a partner (client).

Requirement

The "Change of OPC UA server status" option is selected (OPC UA > Server > Diagnostics) in the OPC UA properties of the CPU.

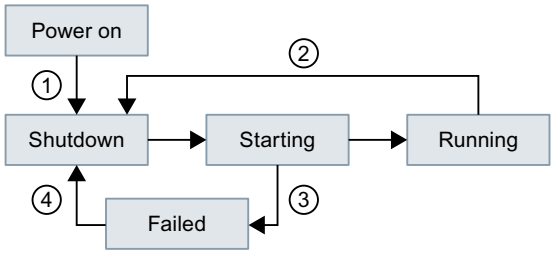
NOTE

If this option is selected, the CPU also automatically enters the lowest set security policy into the diagnostic buffer after startup.

Examples

If the OPC UA server of the CPU shuts down due to a download process and then starts with a valid new configuration, the diagnostic buffer shows new server state, e.g. Shutdown => Starting => Running.
If the OPC UA server shuts down due to a download process and the server cannot start because the type dictionary is too large, the diagnostic buffer finally shows the state "Failed" (Shutdown => Starting => Failed).

Server states and state transitions



①, ④ POWER ON or Load in RUN, if OPC UA relevant data could be affected.

② Loading the hardware configuration with deactivated OPC UA server. The server remains shut down.
Loading the hardware configuration with activated OPC UA server and faulty OPC UA data (for example, too many structures with the result that the type dictionary becomes too large). In this case, the server cannot start (see ③).

③ OPC UA server cannot start due, for example, to faulty configuration.

Figure 10-56 Server states and state transitions

Description of the server states

The individual states that the OPC UA server can assume are explained below.

Server states	Explanation
Shutdown	Initial status • After POWER ON • After loading the hardware configuration with activated or deactivated OPC UA server. • After loading OPC UA relevant data
Starting	OPC UA address space in server is initialized.
Running	OPC UA server running (normal productive state for OPC UA server).
Failed	Error state. OPC UA server cannot start due, for example, to faulty configuration.

10.3.7.4 Session state transition diagnostics

Information on the session state

S7-1500 CPUs as of firmware version V2.8 are able to create an entry in the diagnostic buffer for state changes of an OPC UA session.
The diagnostic buffer displays the new state. The corresponding session ID is also displayed.

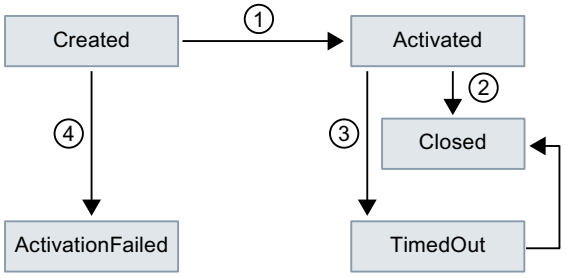
Requirement

The "Change of session states" option (OPC UA > Server > Diagnostics) is selected in the OPC UA properties of the CPU.

Example

A client transmits incorrect authentication data (for example, incorrect password) when a connection is established. The new state of the "ActivationFailed" session is entered with the corresponding session ID in the diagnostic buffer.

Subscription states and state transitions



① Client connects to server, login with correct authentication data (correct credentials).
② Client closes connection correctly.
③ Client no longer sends messages; session ends with timeout.
④ Client connects to server, login with incorrect authentication data.

Figure 10-57 Session states and state transitions

10.3.7.5 Check for security events

If the CPU diagnostics detects a security event during the OPC UA communication, it can enter it in the diagnostic buffer.

Requirements

- S7-1500 CPUs as of firmware version 2.8
- The "Check for security events" option is activated (properties of the CPU > OPC UA > Server > Diagnostics).

Security events detected in diagnostics

S7-1500 CPUs perform diagnostics on the following OPC UA relevant security events:

- Client-certificate is invalid (for example, syntactically or semantically incorrect, incorrect signature, current date is not in the validity period)
- User name/password login failed (deactivated or incorrect data)
- Client wants to use a specific security policy or a specific message security mode; the server does not support the security policy or the requested security mode.
- Client does not establish connection according to specification (OPC UA Spec) (for example, unexpected SecureChannelID/SessionID/client Nonce)

Example

If an attempt is made to compromise communications (for example, by session hijacking, man-in-the-middle attacks etc.), the server detects this via analysis.

10.3.7.6 Request of a remote client failed

S7-1500 CPUs as of firmware version V2.8 are able to create an entry in the diagnostic buffer for the following events:

- Bad client requests (incorrect use)
- Service error occurred
- CPU-specific high limits of the OPC UA server were violated

Example of a faulty client request

For example, there is an incorrect request when a client addresses a node (tag) that does not exist or if a resource is requested that does not exist.
In this case, the corresponding service that caused the fault is entered in the diagnostic buffer and the corresponding session ID is also entered.

Service fault

If a service itself fails, the server returns a ServiceFault. In this case, the status code (Bad...) and the according session ID are entered in the diagnostics buffer.

Example of limit violations

If a service request exceeds a CPU-specific limit, for example, number of sessions, number of monitored items, number of subscriptions, etc., this diagnostics is entered in the diagnostics buffer. Together with the message, it is indicated which limit has been violated.
Exception: If you summarize diagnostics and the message occurs frequently, the limit causing the error is not entered. You receive general information that the supported configuration limit has been violated.

Possible entries for the service that is causing the error

Depending on the client application used, requests to the server can be triggered differently from the user's viewpoint, for example, by an online tool with a graphical user interface or by instructions in a client's program.
With its service-oriented architecture, OPC UA follows a request-response paradigm, therefore the respective client application converts the requests into the service requests defined in OPC UA.
The names of these services are defined and grouped according to their use, see also opcfoundation.org.
In the case of an incorrect use, you can find precisely these names of the services, together with the corresponding session ID, in the diagnostic buffer as the service that caused the error.
The services available with OPC UA are listed below.

Discovery Service Set
FindServers
GetEndpoints

Session Service Set
CreateSession
ActivateSession
CloseSession
Cancel

View Service Set
Browse
BrowseNext
TranslateBrowsePathsToNodeIds
RegisterNodes
UnregisterNodes

Attribute Service Set
Write
Read

Method Service Set
Call

Monitored Item Service Set
CreateMonitoredItems
ModifyMonitoredItems
DeleteMonitoredItems
SetMonitoringMode
SetTriggering

Subscription Service Set
CreateSubscription
ModifySubscription
DeleteSubscriptions
Publish
Republish
SetPublishingMode

10.3.7.7 Subscription diagnostics

Information about a subscription

S7-1500 CPUs as of firmware version V2.8 are able to create an entry in the diagnostic buffer at state changes of a subscription.
The diagnostic buffer displays the new state; exception: "KeepAlive".

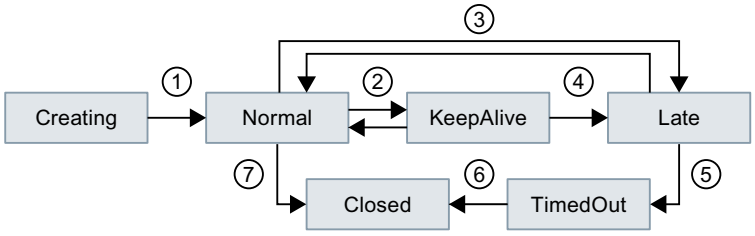
Requirement

In the OPC UA properties of the CPU, the option "Subscriptions: Change of status" (OPC UA > Server > Diagnostics) is selected.

Example

An OPC UA client is connected to an S7-1500 CPU as OPC UA server and generates a subscription in the server.
The diagnostic options for subscriptions are selected in the OPC UA properties of the CPU.
The "Creating" and "Normal" states are entered one after the other with the corresponding subscription ID in the diagnostic buffer.

Subscription states and state transitions



① Subscription is generated and is then active.

② Status change is not entered in the diagnostic buffer because too many entries may be made in the diagnostic buffer depending on the amount of data.

③ See explanation in table for "Late"; for example, no requests to send from client.

④ Maximum KeepAlive value reached.

⑤ See explanation in table for "TimedOut".

⑥ Maximum lifetime of subscription reached.

⑦ Client has deleted subscription.

Figure 10-58 Subscription states and state transitions

Description of the subscription states

A subscription in the OPC UA server can have the following states:

Status	Meaning
Creating	Client has requested a subscription in the server; the server creates the subscription.
Normal	Subscription is created in the server and active.
Closed	Client has deleted the subscription.
KeepAlive	Status if the monitored items do not change over a long period of time. These state transitions are not entered in the diagnostic buffer.
Late	Client has generated a subscription with minimal sampling and publishing intervals. The amount of monitored items is not transmitted to the client during this time. Client no longer transmits requests to send (for example, due to failure).

Status	Meaning
TimedOut	The client has requested a subscription. The server can only honor the subscription (send Publish Response) when there is a sufficient number of send requests (Publish Requests) from the client. When the client stops sending subscription requests, the subscription enters the "TimedOut" state after a certain time.

Subscription: Error in the sampling times

As of firmware V2.5 of the SIMATIC S7-1500 CPU, the OPC UA server can transmit the status code "GoodOverload" when using subscriptions, if an overload of the CPU occurs when sampling the items.

As of firmware V2.8 of the SIMATIC S7-1500 CPU, the OPC UA server can also enter this event into the diagnostic buffer.

Requirement

In the OPC UA properties of the CPU, the option "Subscriptions: Sampling time errors" (OPC UA > Server > Diagnostics) is selected.

Error-free subscription

In the case of an OPC UA subscription to various elements (such as tags), the OPC UA server of the SIMATIC S7-1500 must check the elements for value changes at specified intervals (sampling interval). This check, referred to as "sampling", requires some time, which depends on the number and the data type of the items. After the sampling is completed and a publishing request has been received, the server sends the elements to the client.

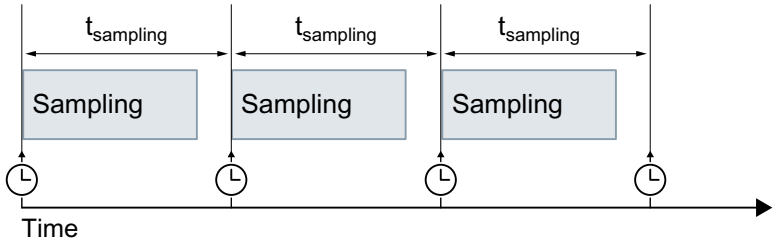
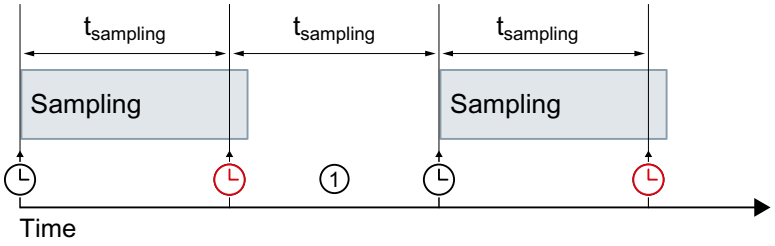


Figure 10-59 Error-free subscription

Subscription with error

If there are too many elements in the queue, there may be an overload of the communication stack. The CPU cannot check all elements in the given sampling interval and must therefore skip the next sampling job.

In this case, the CPU sends the status code "GoodOverload" (0x002F0000) per element, even though the elements were not checked. The meaning of the status code according to IEC 61131-3 is as follows: "Sampling has slowed down due to resource limitations".



① Sampling job is skipped

Figure 10-60 Subscription with error

See also FAQ 109763090 (https://support.industry.siemens.com/cs/ww/en/view/109763090).

More information

You can find information about the server settings for subscriptions in the section Settings of the server for subscriptions (Page 222).

10.3.7.8 Summarizing diagnostics

To prevent the diagnostics buffer being "swamped" by large numbers of identical OPC UA diagnostics, as of STEP 7 V16 service pack 1, you can set parameters so that these diagnostics are entered in the diagnostics buffer as group alarm. Per interval (monitoring time), the CPU then only generates one group alarm per OPC UA diagnostics.
The following sections describe which diagnostics the CPU groups together and how the process runs with a high message volume.

Requirement

The "Summarize diagnostics in case of high message volume" option is activated in the OPC UA properties of the CPU (OPC UA > Server > Diagnostics, "Summarize diagnostics" area).

Example

An OPC UA client repeatedly "overloads" an S7-1500 CPU as OPC UA server with a sampling rate that the server cannot handle (overload).
The "Summarize diagnostics in case of high message volume" setting is activated.
A message appears in the diagnostics buffer for this diagnostic option. It states that the sampling rate cannot be reached; followed by the number of these events within the configured interval.

OPC UA diagnostics that can be summarized

The diagnostics listed below each form their own groups (type). Diagnostic events from the same group are combined using the setting "Summarize diagnostics in case of high message volume":

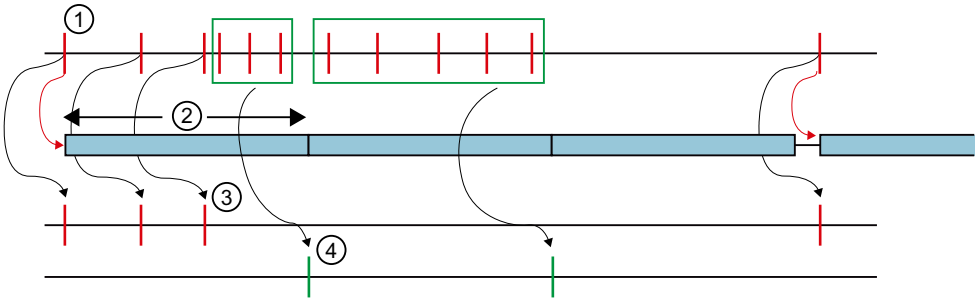
- Incorrect use of an OPC UA service
- OPC UA Service error
- Subscription status has changed
- Sampling rate could not be achieved (subscriptions, overload)
- OPC UA security check failed
- Configuration limit of the OPC UA server violated

Principle of operation

The CPU enters the first three events of an event type in the diagnostics buffer. It then ignores all subsequent diagnostics of this group.

At the end of the monitoring time (interval), the CPU generates a group alarm in which it enters the diagnostics and the frequency of this diagnostics during the elapsed interval. If these diagnostics also occur in the intervals that follow, the CPU only generates one group alarm per subsequent interval.

A diagnostic surge leaves the following pattern in the diagnostics buffer: Three individual messages followed by a series of group alarms. This series can consist of two, three or more group alarms depending on the selected monitoring time and duration of the diagnostic surge.



① Diagnostic results of a group (of a type), for example "Sampling rate could not be reached".

② Interval (monitoring time): When a diagnostic event occurs the first time (or reoccurs), the monitoring time is started (or restarted).

③ Single alarms: The first three diagnostic events from the same group are entered in the diagnostics buffer immediately. Starting with the fourth diagnostic event, the CPU generates only group alarms. If a diagnostic event of this group occurs after a pause of at least one interval, the CPU enters a single alarm in the diagnostics buffer and restarts the monitoring time.

④ Group alarms: After three diagnostic events, the CPU only generates a group alarm as a summary of all additional diagnostic events in this interval. If these diagnostic events also occur in the intervals that follow, the CPU only generates one group alarm per subsequent interval.

Figure 10-61 Summary of diagnostics

10.4 Using the S7-1500 CPU as an OPC UA client

10.4.1 Overview and requirements

With STEP 7 (TIA Portal) Version V15.1 and higher, you can assign parameters and program an OPC UA client that can read PLC tags in an OPC UA server. Furthermore it is possible to transfer new values for PLC tags to an OPC UA server. In addition, you can call methods that an OPC UA server provides in your user program. You use the instructions for OPC UA clients in your user program for this.

The instructions of the OPC UA client are based on the standard "PLCopen OPC UA Client for IEC61131-3".

PLCopen specification

With these standardized instructions, you can develop an OPC UA client functions in your user program that can be executed in an S7-1500 CPU.

In addition, it is possible with just a few adaptations to run this user program in controllers of other manufacturers if these manufacturers have also implemented the OPC UA Specification "PLCopen OPC UA client for IEC61131-3".

Convenient editors in STEP 7

For the parameter assignment of the instructions for OPC UA clients, a convenient editor is available in the TIA Portal – the connection parameter assignment (Page 216).

As of Version 15.1, STEP 7 also features an editor for client interfaces (Page 309).

This section describes how you work with these editors.

First, you will be shown how to create and configure a new interface with the interface editor, because you need this type of interface for the subsequent connection parameter assignment.

The description uses an example for better comprehensibility, see Description of the example (Page 308).

Requirements

- You have the required runtime license for OPC UA and have configured the license in STEP 7 (CPU Properties > Runtime Licenses).
- The client of the S7-1500 CPU is activated.

To use the client of the S7-1500 CPU, you must enable it:

1. Select the area "OPC UA > Client" in the properties of the CPU.
2. Select the "Enable OPC UA client" option.

If you do not enable the client, the connection is not established. You receive a corresponding error message at the instructions, for example "OPC_UA_Connect".

For information about the application name, which also applies to the server and the client, see here (Page 216).

Overview

To use the editor and the connection parameter assignment, follow these steps:

1. First, specify a client interface. Add to this the PLC tags and PLC methods interface that you want to access ("First step (Page 309)").
2. Next, configure the connection to the OPC UA server (Second step (Page 323)).
3. Finally, use the configured connection for the OPC UA client instructions (Third step (Page 330)).

10.4.2 Useful information about the client instructions

With the standardized OPC UA client instructions you are able to control communication for the following tasks with the S7-1500 CPU as an OPC UA client:

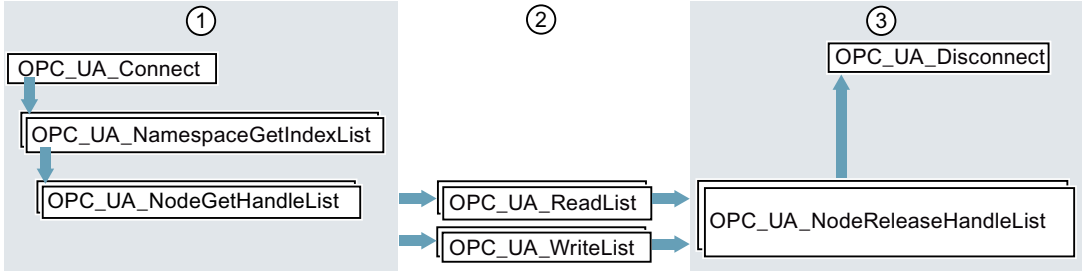
- Read/write tags of the OPC UA server
- Call methods in the OPC UA server

Optional instructions can be used to determine the following information:

- The status of the connection between the OPC UA client and OPC UA server
- Node IDs of nodes with known hierarchy of the address space

Standardized sequence of OPC UA communication

The sequence of the communication, and thus the order of the instructions, follows a pattern that is illustrated in the following.

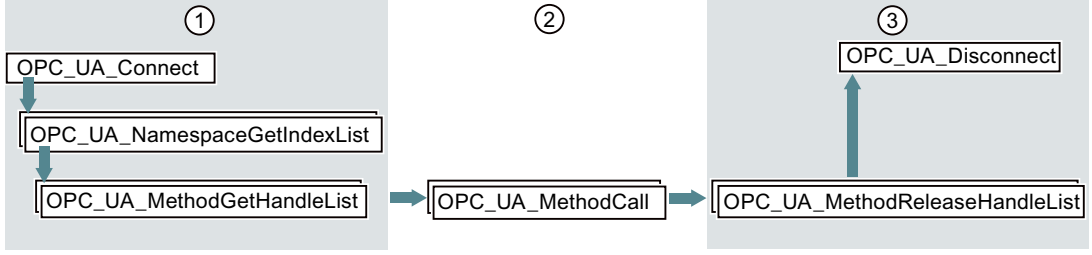


① Instructions for preparation of read and write operations

② Read and write instructions

③ Instructions for "clean-up" after a completed read or write operation

Figure 10-62 Run sequence for a read or write operation

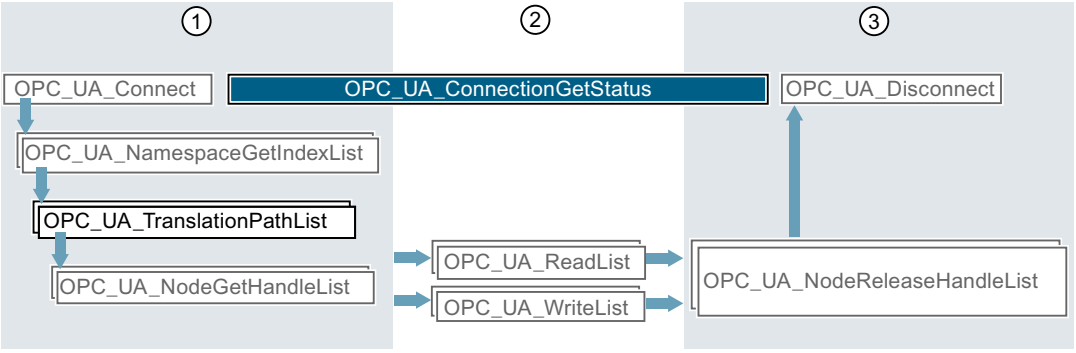


① Instructions for preparation of method calls
② Method calls
③ Instructions for "clean-up" after completed method calls

Figure 10-63 Run sequence for a method call in the OPC UA server

Optional instructions (reading out the status of a connection / reading out node IDs of nodes with known hierarchy of the address space)

- OPC_UA_ConnectionGetStatus
- OPC_UA_TranslatePathList



① Instructions for preparation of read and write operations with inserted instruction for requesting, for example, the NodeIDs of nodes of the OPC UA server.
② You can determine the connection status between the establishment and termination of the connection in parallel with other instructions.
③ Instructions for "clean-up"

Figure 10-64 Run sequence of optional instructions

Convenient editors in STEP 7

The OPC UA client instructions are described in detail in the reference part (STEP 7 information system). For parameter assignment of the instructions, a convenient editor is available in the TIA Portal – the connection parameter assignment (Page 323).
We recommend starting with the connection parameter assignment for the first program draft and using additional instructions and manually optimizing the program as required.

Information about the client instructions

The client instructions are described in detail in the help to the Instructions > Communication > OPC UA > OPC UA client.

Application example in Online Support

This application example (https://support.industry.siemens.com/cs/ww/en/view/109762770) provides you with an S7 user block "OpcUaClient" that summarizes the most important functions of the OPC UA instructions, accelerates the implementation for you and simplifies

the programming. The OPC UA server in the example is an S7-1500 controller with a simple simulation program for process values.

The S7 user block performs the following:

- Establishment and termination of the connection to the server
- Diagnostics of the connection and automatic reconnection after connection terminations
- Registered Read
- Registered Write
- Registered Method Call

10.4.3 Number of client instructions that can be used simultaneously

SIMATIC error codes for OPC UA client instructions

The following limits apply to the simultaneous use of OPC UA client instructions (up-to-date technical specifications of the CPUs can be found on the Internet (https://support.industry.siemens.com/cs/ww/en/ps/td)):

Table 10-5 Quantity structures for OPC UA client instructions

OPC UA instruction	Maximum number for CPU 1510SP (F) CPU 1511 (C/F/T/TF) CPU 1512C CPU 1512SP (F) CPU 1513 (F)	Maximum number for CPU 1505 (S/SP/SP F/SP T/SP TF) CPU 1515 (F/T/TF) CPU 1515 SP PC (F/T/TF) CPU 1516 (F/T/TF)	Maximum number for CPU 1507S (F) CPU 1517 (F/T/TF) CPU 1518 (F)
OPC_UA_Connect	4	10	40
OPC_UA_NamespaceGetIndexList	4*	10*	40*
OPC_UA_NodeGetHandleList	4*	10*	40*
OPC_UA_MethodGetHandleList	4*	10*	40*
OPC_UA_TranslatePathList	4*	10*	40*
OPC_UA_ReadList	20 in total (max. 5 per connection, see OPC_UA_Connect)	50 in total (max. 5 per connection, see OPC_UA_Connect)	200 in total (max. 5 per connection, see OPC_UA_Connect)
OPC_UA_WriteList	20 in total (max. 5 per connection, see OPC_UA_Connect)	50 in total (max. 5 per connection, see OPC_UA_Connect)	200 in total (max. 5 per connection, see OPC_UA_Connect)
OPC_UA_MethodCall	20 in total (max. 5 per connection, see OPC_UA_Connect)	50 in total (max. 5 per connection, see OPC_UA_Connect)	200 in total (max. 5 per connection, see OPC_UA_Connect)
OPC_UA_NodeReleaseHandleList	4*	10*	40*
OPC_UA_MethodReleaseHandleList	4*	10*	40*
OPC_UA_Disconnect	4*	10*	40*
OPC_UA_ConnectionGetStatus	4*	10*	40*

* maximum 1 per connection

Maximum number of usable OPC UA client interfaces

If you create OPC UA client interfaces using the connection parameter assignment, the number of client interfaces is limited to 40.
Create the OPC UA client interfaces by double-clicking the "Add new client interface" symbol in the project tree of the "OPC UA communication" area.
The maximum number of OPC UA client interfaces is independent of whether you also use the CPU as OPC UA server.

10.4.4 Example configuration for OPC UA

The following sections describe how you can use the client interfaces editor and the connection parameter assignment.
The description is based on a specific example: Two S7-1500 CPUs operate in the system: One CPU serves as the OPC UA client and the other as the OPC UA server.
You can, of course, also use controllers, sensors and IT systems of other manufacturers as OPC UA clients or servers. In particular, the data exchange between different systems (interoperability) is a major advantage of OPC UA.

Connection parameter assignment using an example:

The plant produces blanks in a production line.
The following controllers are used:

1. An S7-1511 CPU serves as the controller of the production line.
   The controller is named "**Productionline**" in the example.
   The OPC UA server of the controller is enabled.
   The CPU has the IP address 192.168.1.1 in the example.
   This CPU publishes the values of following tags via the OPC UA server:
   - **NewProduct**
     The tag has the data type "BOOL".
     When this PLC tag has the value TRUE, the production line has processed a blank.
     The blank is ready for pick-up.
   - **ProductNumber**
     This tag contains the identification number of the blank.
     The tag has the data type "Int".
   - **Temperature**
     This tag contains temperature values recorded during the production of the blank.
     The tag is an array with elements of the "Real" data type.

   In addition, this CPU provides the following writable tag:
   - **ProductionEnabled**
     The tag is set by the OPC UA client.
     The tag has the data type "BOOL".
     If the value is set to TRUE, the production line is released and may produce blanks.

   In addition, this CPU provides the following method via the OPC UA server:
   - **OpenDoor**
     OPC UA clients can hereby arrange for an access door to be opened to the production line.

2. An S7-1516 CPU controls the interaction with other production lines.
   This CPU is named "**Supervisor**" in the example.
   The OPC UA client of this CPU is enabled.
   Using OPC UA, this CPU can read the NewProduct and ProductNumber tags, set the ProductionEnabled tag and call the OpenDoor method.
   The CPU has the IP address 192.168.1.2 in the example.

The following figure shows the example in the network view of the TIA Portal:

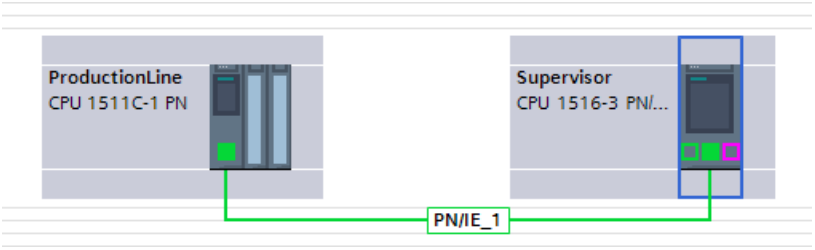


Figure 10-65 Example of assigning connection parameters in the network view

10.4.5 Creating client interfaces

As of Version 15.1, the TIA Portal has an editor for client interfaces.
You group all PLC tags that you want to read or write from an OPC UA server in a client interface.
In addition, the client interface contains all methods that the OPC UA server provides and that you want to call with your user program (that acts as an OPC UA client).
If you create a client interface, STEP 7 also creates data blocks for the parameter assignment of the connection to the OPC UA server from which you want to read data or to which you want to write data.

Maximum number of client interfaces
You can create a maximum of 40 client interfaces.

Editor for client interfaces

To create a client interface, follow these steps:

1. Select the project view in the TIA Portal.
2. In the "Devices" area, select the CPU you want to use as an OPC UA client.
3. Click "OPC UA communication > Client interfaces".

4. Double-click "Add new client interface".
   STEP 7 creates a new client interface and display in the editor.

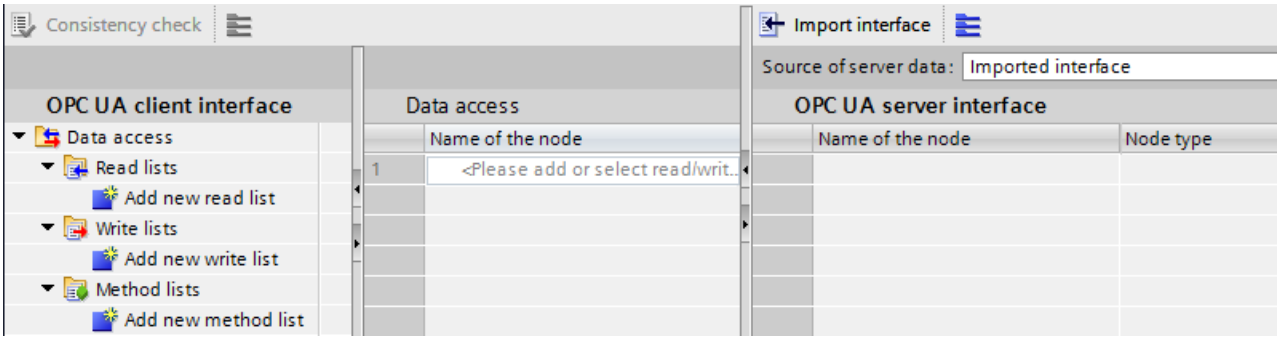


Figure 10-66 Adding OPC UA client interface

   STEP 7 names the new interface "Client interface_1". If a "Client interface_1" already exists, the new interface receives the designation "Client interface_2" etc.
   In addition, STEP 7 creates the following data blocks:
   - Client_Interface_1_Configuration
     The data block already contains all system data types that are needed for the instructions of the OPC UA client.
     This data block is filled when you configure the connection to the OPC UA server.
     You configure a connection in the properties of the client interface, see: Example configuration for OPC UA (Page 308).
   - Client_Interface_1_Data
     A data block for the PLC tags that you want to read or write from an OPC UA server as well as for methods that you want to call in the OPC UA server.
     You use this data block in your user program.
     This data block is currently still empty.
5. Select a descriptive name for the new client interface.
   Select "Productionline" in the example.
   This also changes the names of the associated data blocks to:
   - Productionline_Data
   - Productionline_Configuration
6. To import an OPC UA server interface, click the "Import interface" button in the top right of the editor.
   This allows you to import an XML file which describes the server interface of an OPC UA server.
   Alternative: To determine online the server interface of a connected OPC UA server, see: Determine server interface online (Page 316).
7. STEP 7 displays a dialog with which you can select an XML file.
   This XML file describes a address space of an OPC UA server.
   The address space of an OPC UA server contains all PLC tags and server methods published by an OPC UA server.
   OPC UA clients can access this address space:
   - Read PLC tags
   - Write PLC tags
   - Calling Server Methods

   The address space of an OPC UA server can be divided into one or more server interfaces.

For creating server interfaces, see: Creating a server interface for companion specification (Page 245).

8. Create a **read list** in this client interface.
   To do this, follow these steps:
   – Click "Add new read list" in the left section of the editor.
     STEP 7 adds a new list named "ReadList_1".
     For the example, change the name to "ReadListProduct"
   – Now add the new read list of the PLC tags that you want to read from the OPC UA server.
     In the example the "NewProduct" and "ProductNumber" tags are added to the "ReadListProduct" read list.
     Select the "NewProduct" tag in the right-hand field of the editor ("OPC UA Server interface"). Drag the "NewProduct" tag to the "ReadProduct" read list in the middle field of the editor. Follow the same procedure with the "ProductNumber" tag.

   The figure below shows the right field of the editor.

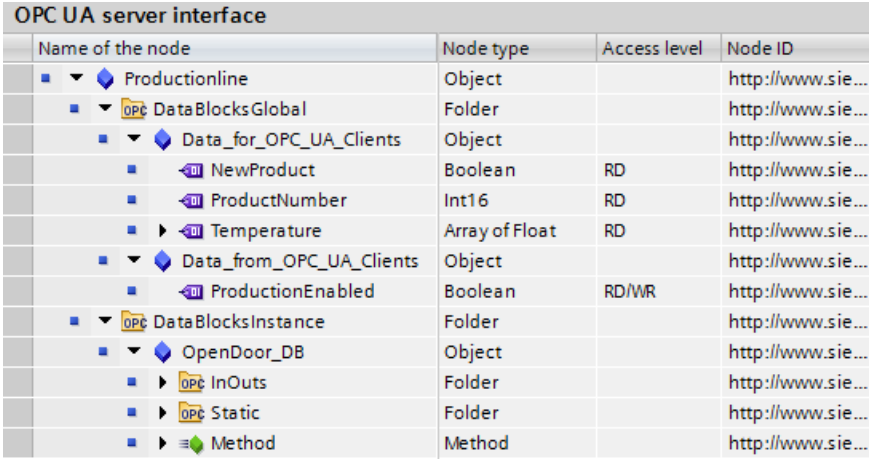


Figure 10-67 Read list in the OPC UA server interface

Alternative:

You can also select a new read list by dragging the right field of the editor ("OPC UA Server interface") to a node of the type Object or Folder and then dragging it to "Add new read list" in the left field of the editor. The new read list then contains all PLC tags of the node that has been moved.

In the example, select the object "Data_for_OPC_UA_Clients", which contains the tags "NewProduct" and "ProductNumber". STEP 7 generates the new read list "Data_for_OPC_UA_Clients". In addition, the object contains the tag "Temperature". Delete the "Temperature" tag from the read list. Since they should not be read in the example.

Change the name of the read list in "ReadListProduct".

The following figure shows the content of the read list:

ReadListProduct

Name of the node	Node type	Access level	Node ID
NewProduct	BOOL	RD	http://www.siemen...
ProductNumber	INT	RD	http://www.siemen...

Figure 10-68 Read list

NOTE

Read and write lists do not support all node types.

The OPC UA client of the S7-1500 CPU does not support all OPC UA data types (node types) that can be made available via an OPC UA server interface. If you place an unsupported node type, for example, in a read list or write list a corresponding error signal appears. In this case, you cannot include the corresponding node in the read or write list.

Which types are supported is described here: Mapping of data types (Page 166)

9. If you want assign new values to PLC tags, create a **write list** in this client interface.
   To do this, follow these steps:
   – Click "Add new write list" in the left section of the editor.
     STEP 7 adds a new list with the name "ReadList_1".
     For the example, change the name to "WriteListStatus".
   – Now add the new write list of all OPC UA server tags to which you want to assign new values.
     In the example, add the "WriteListStatus" tag to the write list "ProductionEnabled".
     Select the Tag of right field of the editor ("OPC UA Server interface"). Drag the tag to the write list in the middle field of the editor.

   Alternative:

   You can also create a new write list by selecting a node of the type Object or Folder in the right field of the editor ("OPC UA server interface") and then dragging to "Add new write list" in the left field of the editor.

   The new write list then contains all tags of the relevant node.

   In the example, select the object "Data_from_OPC_UA_Clients", which contains the tag "ProductionEnabled". STEP 7 generates the new write list "Data_from_OPC_UA_Clients". Change the name in "WriteListStatus".

   The following figure shows the content of the write list:

WriteListStatus

Name of the node	Node type	Access level	Node ID
ProductionEnabled	BOOL	RD/WR	http://www.siemen...

Figure 10-69 Write list

10. If you want to call a method of this OPC UA server, generate a new method list.
    To do this, follow these steps:
    - In the left section of the editor, click "Add new method list".
      STEP 7 adds a new list with the name "Method list_1".
      For the example, change the name to "MethodListOpenDoor".
    - Now add a method of the OPC UA server to the new method list.
      In this example, add the method "OpenDoor" to the method list "MethodListOpenDoor".
      Select the method of right field of the editor ("OPC UA Server interface"). Drag the method to the method list in the middle field of the editor.

    Alternative:
    You can also generate a new method list by selecting a method (node of the type Object) in the right field of the editor (OPC UA Server interface) and then dragging it to "Add new method list" in the left field of the editor. The new method list then contains the method of the relevant node.
    The following figure shows the content of the method list:

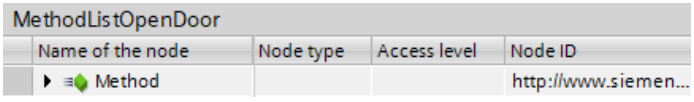


Figure 10-70 Methods list

    If you want to call another method of the OPC UA server, you must create a new method list. Each method list contains only one method.
    See also Useful information about server methods (Page 268).
11. Compile the project.
    To do so, select the project and click the following button in the toolbar:

STEP 7 compiles the project and updates the data blocks that belong to the "Productionline" client interface.

NOTE

During compilation, STEP 7 overwrites all data in the data blocks belonging to the client interface. For this reason, you should neither add to nor correct these data blocks manually.

NOTE

Renaming nod names (DisplayNames)

In read lists, write lists and method lists you can rename the name of a node by means of the shortcut menu. This is the "DisplayName" in the OPC UA language usage.

If you rename the name of a method list node and the node is already used in a programmed block for the method call "OPC_UA_MethodCall", the compilation of the project leads to consistency errors: During the compilation the UDTs of the method are generated with the changed name. The references to the method used in the program are then no longer correct.

To correct the consistency errors, you can either undo the name change of the method in the client interface or navigate to the method call and assign the relevant parameters again there under "Properties > Block parameters" ("Configuration" tab).

Data blocks of client interface

The following data blocks belong to the "Productionline" client interface:

- **Productionline_Configuration**
  A data block for the configuration.
  In the example, this data block is called "Productionline_Configuration".
  The data block already contains all system data types that are needed for the instructions of the OPC UA client.
  In addition, the data block contains general default values for parameter assignment of the connection to an OPC UA server.
  If you are working with connection parameter assignment, this data block will be filled.

- **ProductionLine_Data**
  A data block for the PLC tags that you have entered in the client interface editor.
  In the example, this data block is called "Productionline_Data".
  The figure below shows the data block.

Static	
ReadListProduct	Struct
Variable	"Productionline.ReadListProduct"
NewProduct	Bool
ProductNumber	Int
NodeStatusList	Array[0..1] of DWord
NodeStatusList[0]	DWord
NodeStatusList[1]	DWord
TimeStamps	Array[0..1] of LDT
TimeStamps[0]	LDT
TimeStamps[1]	LDT
WriteListStatus	Struct
Variable	"Productionline.WriteListStatus"
ProductionEnabled	Bool
NodeStatusList	Array[0..0] of DWord
NodeStatusList[0]	DWord
MethodListOpenDoor	Struct
MethodStatusList	Array[0..0] of DWord
MethodStatusList[0]	DWord
MethodResultList	Array[0..0] of DWord
MethodResultList[0]	DWord
Method	Struct
Inputs	"Productionline.MethodListOpendoor.Method.Inputs"
Number	Int
Outputs	"Productionline.MethodListOpendoor.Method.Outputs"
Result	Int

Figure 10-71 "Productionline_Data" data block

Use the "Productionline_Data" data block in your user program and access the read values of the "NewProduct" and "ProductNumber" PLC tags. This is explained in the following section using an example.

Reading and writing PLC tags of the client interface

Example: Reading the "ProductNumber" value

For example, you write in an SCL program:

```
#MyLocalVariable :=
"Productionline_Data".ReadListProduct.Variable.ProductNumber;
```

You use this, for example, to assign the number of the blank that was just produced in the production line to the local tag "#MyLocalVariable".

Requirements:

- A connection exists to the OPC UA server of the CPU, which controls the production line.
- The OPC UA client has read the current values.

For this reason you check whether a read value is valid:

- Check whether the value in "Productionline_Data".ReadListProduct.NodeStatusList[1] is equal to 0.
- Optional: Check when this value was sent from the OPC UA server. This value is in "Productionline_Data".Product.TimeStamps[1]. If no time stamp is requested, the communication load is reduced.

Example: Writing the "ProductEnabled" value

Transfer the new values for PLC tags, in the example for the "ProductionEnabled" tag, to the OPC UA server using the data block.

With the following assignment, you enable the production line in the example plant:

```
"Productionline_Data".WriteListStatus.Variable.ProductionEnabled := TRUE;
```

This is only successful, however, if the following requirements are met:

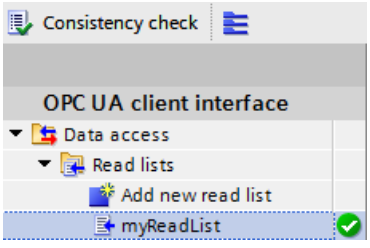
- A connection exists to the OPC UA server of the CPU, which controls the production line.
- Current values are being written via the OPC UA client

Consistency check

Finally, check the consistency of the read/write list or method list.

1. Select the list that you want to check.
2. Click the "Consistency check" button above the "OPC UA client interface" area.

A green check mark indicates an error-free assignment of the tags or methods to the corresponding elements of the server interface.



You can assume that the data exchange between client and server and method calls operate without problem in runtime.

In the event of an error a list appears in the Inspector window. From this list you can jump to the respective error.

During the consistency check, STEP 7 checks:

- Whether all elements that you use in the respective list are also present in the server.
- Do the data types used match?
- For methods: Do the number, name, order, and data types of method arguments match?

10.4.6 Determine server interface online

With STEP 7 (TIA Portal) you can determine the interface of an OPC UA server online. This provides information on which tags of a connected OPC UA server you can read or set (write) with OPC UA clients. It also provides information on which server methods of the OPC UA server are available for OPC UA clients.

If you are work offline you can create the interface of the OPC UA server by means of an OPC UA XML file. The address space of the server is described in the OPC UA XML file, see: Export OPC UA XML file (Page 215).

Determine online server interfaces

To determine a server interface online, follow these steps:

1. In the STEP 7 project tree, select the CPU which is configured as OPC UA client (Supervisor in the example).
2. Select the client interface (in the example, OPC UA Communication > Client interfaces > Productionline).
   If no client interface has been created, double-click "Add new client interface".
3. Double-click the selected client interface.
   The editor for client interfaces is displayed.

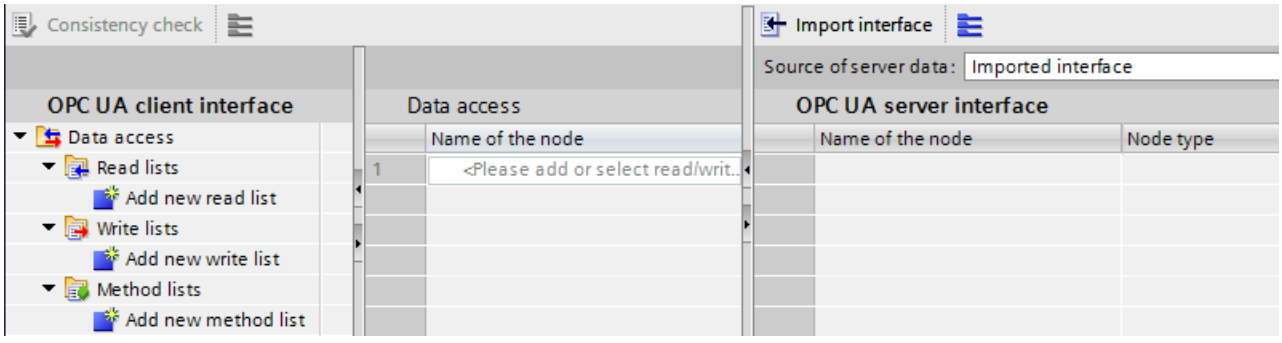


Figure 10-72 Editor for client interface

4. In the left section of the editor, click "Add new read list", "Add new write list", or "Add new method list".
5. In the right field of the editor, select "Online[]" as data source for "Source of server data":

6. Click the "Online Access" button.
   STEP 7 displays the "Connect to OPC UA server" dialog.

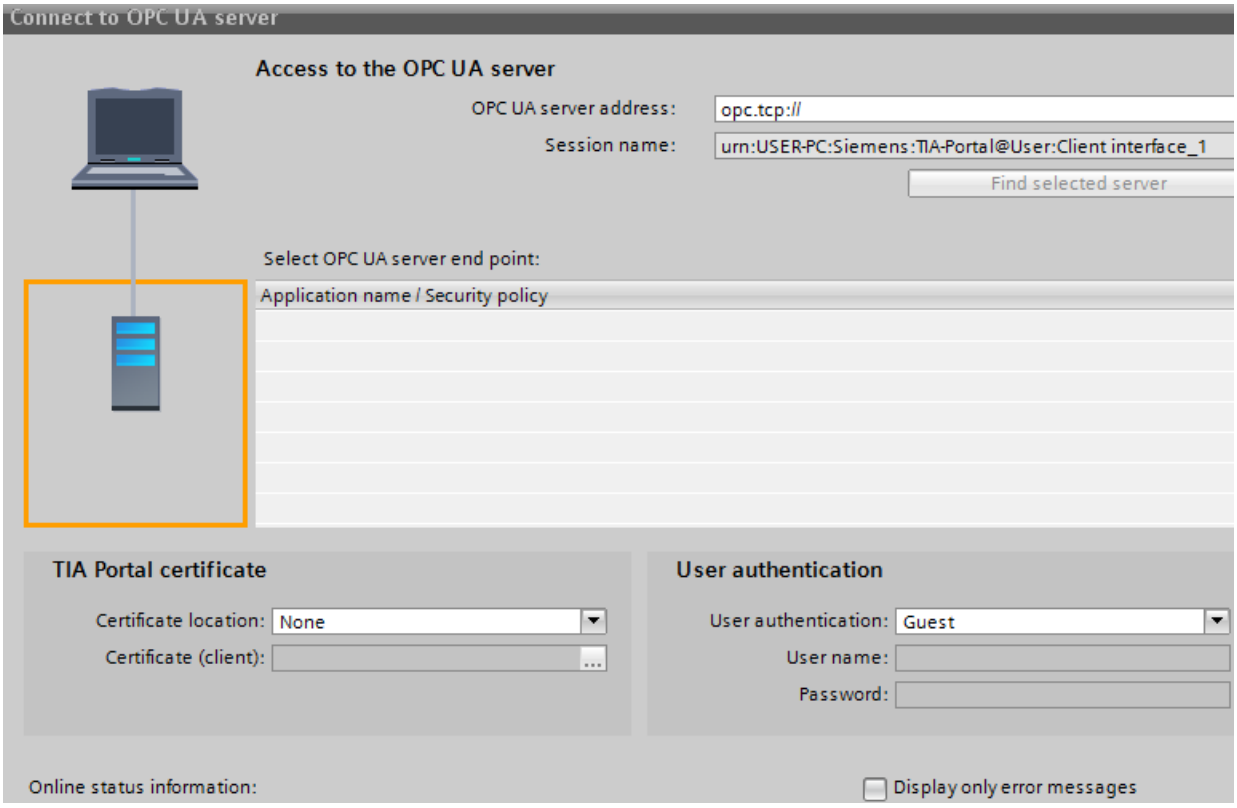


Figure 10-73 "Connect to OPC UA server" dialog

Tip: When establishing an online connection to an OPC UA server for the first time, use the "Online access" button. When reconnecting after a disconnection, select the "Connect to Online Server" button next to the "Online" selection field.
In the top right, enter the IP address of the OPC UA server whose server interface you want to determine online.

7. Click "Find selected server".
   STEP 7 establishes a connection to the OPC UA server and determines all security settings (server endpoints) that the server holds in readiness.
   STEP 7 displays the end points as list:

Figure 10-74 Found OPC UA server with all server endpoints

8. Click on the end point you want to use for a connection of STEP 7 to the OPC UA server.

9. Do you want to use a secure connection?
   - If you have selected a secure end point, then select the entry "TIA Portal" for the "Certificate location".
     And under "Certificate (Client)", select a client certificate for your PC on which STEP 7 (TIA Portal) is currently running.
     If a client certificate does not yet exist for your PC, you can generate a client certificate here in the TIA Portal.
     Proceed as follows to generate a certificate for your PC:
     - Click on the button in the "Certificate (Client)" input field.
     - Click "Add".
     - For "Certificate owners" enter "STEP 7 (TIA Portal)".
     - Select the "OPC UA client" entry at "Usage".
     - For "Subject Alternative Name (SAN)", enter the IP address of your PC, on which you are currently running STEP 7 (TIA Portal), under "Value". Overwrite the already entered IP address.
     - If your PC uses a second IP address, enter this address as well. If your PC does not use a second IP address, delete the second IP address already entered.
     - Click "OK".
   - If you have not selected a secure end point, then keep the default ("None").
10. How do you want log on?
    - If you want to log onto the OPC UA server as guest, then apply the default with "User authentication".
    - If you want to log on with user name and password, select "User name and password".
      Use the user name and password which was stored during the configuration of the OPC UA server in the properties of the CPU under "General > OPC UA > Server > Security > User authentication > User management".
11. Click on the "Go online" button.
    When a secure connection is established, a message appears that you must accept the server certificate for the secure connection to be established. In the message window, you can display further details about the server certificate via a link.
    This standard Windows window only provides information about the server certificate. If you click on the button to install the server certificate, the server certificate is not saved to the certificate memory of the TIA Portal, i.e. at the next connection attempt you will be prompted again to accept the server certificate.
    STEP 7 then establishes a connection to the OPC UA server and again displays the editor for client interfaces.

In the right field of the editor, STEP 7 displays the uppermost level of the address space of the OPC UA server:

12. Click on the small black triangle next to "Objects".
    STEP 7 now also displays the level below Objects.
13. Click on the small black triangle next to "Productionline".
    STEP 7 now also displays the level below Productionline.
14. Now open additional lower-level folders:

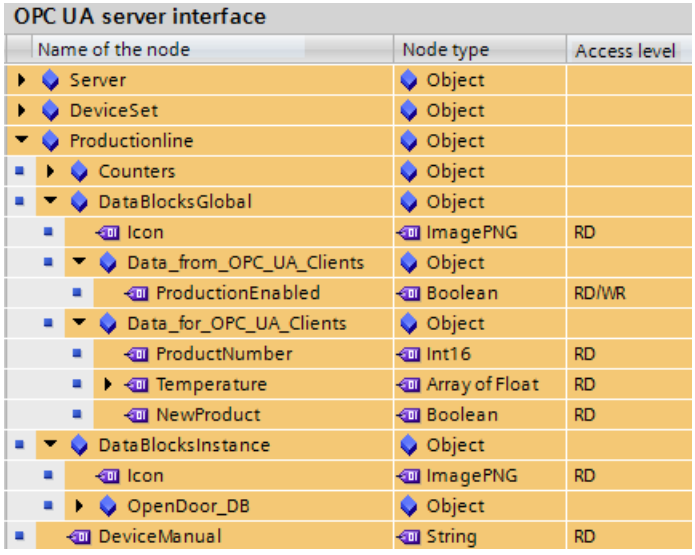


Figure 10-75 Online view of OPC UA server interface

More information

You can find information about mapping of data types in the section Mapping of data types (Page 166).
For information on how to create a client interface, refer to the section Creating client interfaces (Page 309).

10.4.7 Using multilingual texts

In the client interface editor, you are also importing texts that can be displayed in different languages with the OPC UA XML files (information models). Multilingualism is optional, and each node can be defined differently regarding the languages it offers.
In the XML file, these are the following fields that can be prepared for different languages:

- Display name
- Description

Example for multilingual texts in an OPC UA XML file

In the XML file below, the display name and the description, for example, are entered with a "default" text and multiple localizable texts.

- Default text is the first entry without localization information.
- Localized text is the text after "Locale=" followed by a language code, e.g. "it-IT" for Italian

```
<UAVariable NodeId="ns=3;i=6070" BrowseName="3:EngineeringRevision" ParentNodeId="ns=3;i=1002"
DataType="String">
    <DisplayName>EngineeringRevision</DisplayName>
    <DisplayName Locale="en-US">EngineeringRevision</DisplayName>
    <DisplayName Locale="de-DE">Revisionsstand</DisplayName>
    <Description>Revision Level of the engineering environment.</Description>
    <Description Locale="en-US">Revision Level of the engineering environment.</Description>
    <Description Locale="de-DE">Revisionsstand der Engineeringumgebung.</Description>
    <Description Locale="fr-FR">Niveau de révision de l'environnement d'ingénierie.</Description>
    <Description Locale="it-IT">Livello di revisione dell'ambiente di ingegneria.</Description>
<References>
    <Reference ReferenceType="HasTypeDefinition">i=68</Reference>
    <Reference ReferenceType="HasModellingRule">i=78</Reference>
</References>
</UAVariable>
```

Figure 10-76 Example of multilingual texts in an OPC UA XML file

Display of multilingual texts

When importing a server interface, the available multilingual texts are saved internally and downloaded to a CPU together with the project.

The client editor displays the text from the OPC UA XML file in the columns "Name of the node" (corresponds to "DisplayName") and "Description" (corresponds to "Description").

The following cascading rules determine which language is shown for a node:

- When the node contains text in the currently used editing language, the text is also displayed in the editing language.
  (Setting the editing language: In the project tree, select the area "Languages & resources > Project languages")
- When the node does not contain text in the editing language but a default text is defined there (without language code), the default text is displayed.
- "Name of the node" column: If no default text is defined either but a text in any other language exists, the DisplayName text is displayed in the first available language. This rule does not apply to description texts.
- If none of the conditions listed above is met, no text is displayed.

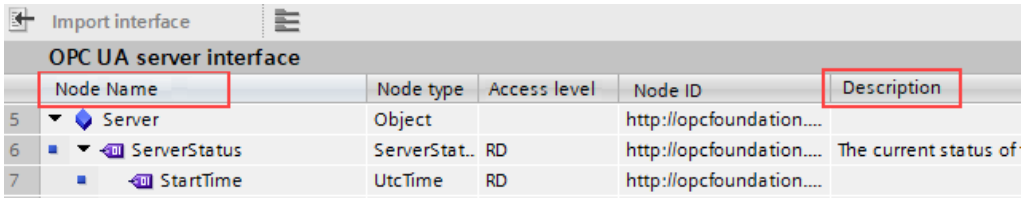


Figure 10-77 Display for multilingual texts

When you change the editing language, the multilingual text in the imported interface will also change according to the rules explained above.

You can then apply the nodes in the corresponding lists (read list, write list, method list) with drag and drop.

You cannot change the language in the lists (read list, write list, method list).

Applying the displayed description texts as comment in PLC data types

When you compile the program, STEP 7 automatically creates PLC data types (UDTs) for each read list, for each write list and for inputs or outputs of each method. These UDTs each have one element for each node.

The UDTs apply the description text as comment according to the rules stated above. STEP 7 creates the comment in only one language, just like the texts in the OPC UA server interface can only be displayed in one language.

10.4.8 Rules for the access to structures

The rules for the access to structures are explained below. Note these rules when reading and writing values of complete structures provided by an OPC UA server.

How the client of the S7-1500 CPU accesses structures

The OPC UA client of the S7-1500 CPU uses neither TypeDictionaries nor DataTypeDefinition attributes, which a server offers for the resolution of these structures.

These options of the OPC UA client for checking structural elements in runtime are limited in the client.

Rules for the access to structures

If you use the client interfaces to configure the read and write lists (connection parameterization) and assign the PLC data types to the imported or online determined address model of the server, the read and write accesses to structures operate trouble-free in runtime.

The configuration by means of client interface automatically ensures that the sequence and the data type of the structural elements are coordinated on client and server side.

Recommendation: Update an S7-1500 CPU (as server) to the current firmware version (e.g. V2.0 > V2.5.2 or higher).

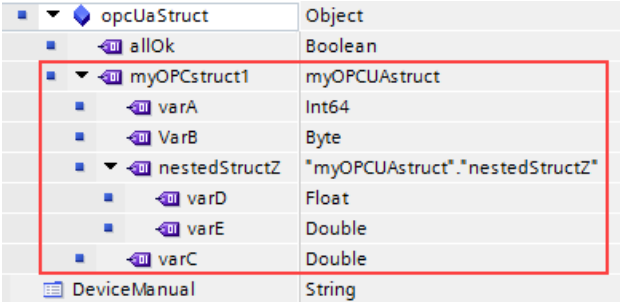
In runtime the OPC UA client only checks the total length of the transmitted value; more detailed checks are not possible.

Strings (WSTRING, STRING and OPC UA ByteString) are also permitted in structures. Strings do have a variable length, but OPC UA surmounts this variability: At the time of transfer, a length field in which the length of the string is encoded is prefixed to each string. Therefore, a S7-1500 CPU as a OPC UA client can check the length of a string and determine whether the string "fits into" the assigned CPU tag. In this manner, the CPU can also check the total length of the structure.

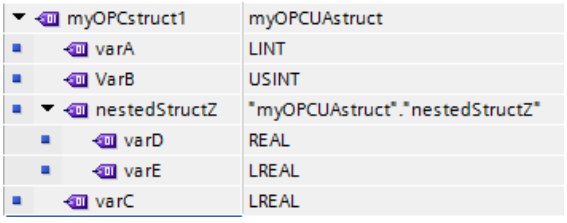
Mapping rules apply to the assignment of OPC UA structures to PLC tags or DB tags (see Mapping of data types (Page 166)).

Example of an error-free assignment of the structure elements

In the imported node set file (XML export), the structure is defined as follows:



The structure mapped in the read list matches, both in the order and in the assigned data types, the corresponding nodes of the node set file.



If the structure now changes on the server, for example tagA and tagB are swapped, and the read list remains the same in the client, the assignment is no longer correct:

- The total length of the data remains the same (only the order has changed)
- The configuration of the structure is different for client and server!

⚠ WARNING

No error message in the case of different structure configuration between client and server

If the structures of client and server do not match, this rule violation will possibly not generate any error during compilation and also not in runtime.

Make sure not to change the configured assignments for structures in runtime. If required, reconfigure the assignment in the read and write lists!

10.4.9 Using connection parameter assignment

10.4.9.1 Creating and configuring connections

With the instructions for OPC UA clients, you create a user program that exchanges data with an OPC UA server. A series of system data types are required for this.

To simplify your work with these system data types, a connection parameter assignment for OPC UA clients is available starting in STEP 7 (TIA Portal) Version 15.1.

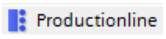
Use of the connection parameter assignment is optional and not mandatory. You can also manually create the required system data types.

We use an example to make the description easier to follow, see description of the example (Page 308).

Opening the connection parameter assignment

To configure the connection to an OPC UA server, follow these steps:

1. In the "OPC UA communication" area, double-click the client interface whose parameters you want to assign in the project tree.
   For the example configuration: Double-click the "ProductionLine" client interface.

   The section "Create client interface (Page 309)" describes how to create a client interface.
2. Click the "Properties" tab (Inspector window) if the tab is not already displayed.
   STEP 7 now displays the connection parameter assignment for the instructions of the OPC UA client.
   The "General" tab is open.
3. Click on the "Configuration" tab and set the connection to the OPC UA server.

Setting the connection parameters

1. Select a descriptive name for the session. For the example, select the name "OPC UA Connection to ProductionLine".
2. In the "Address" field, enter the IP address of the OPC UA server to which your user program (that operates as an OPC UA client) is to establish a connection. In the example configuration, the CPU that controls the production line has the IP address "192.168.1.1". A connection to the OPC UA server of this CPU is to be established. For this reason, you enter this IP address in the "Address" field. In this case, the OPC UA server uses the default port 4840.
   Alternatively, you can enter a valid DNS name in the "Address" field. The length of the DNS name is limited to 242 characters.
   If the address is not valid, the error message is shown: "Enter a valid address".
   If the string of the "Address", "Port" and "Path" fields contains more than 254 characters, an error message is also displayed.
3. Enter a path within the OPC UA server to restrict access to this path. The information is optional. However, some servers only establish a connection if a server path is specified.
   When you specify a path, it is automatically entered at the "ServerEndpointUrl" entry in the configuration DB for the client interface. The entry then consists of the components "OPC Schematic Prefix", "IP address", "Port number" and "Server path", for example: "opc.tcp://192.168.0.10:4840/example/path".

The following figure shows the entry of the IP address for the OPC UA server:

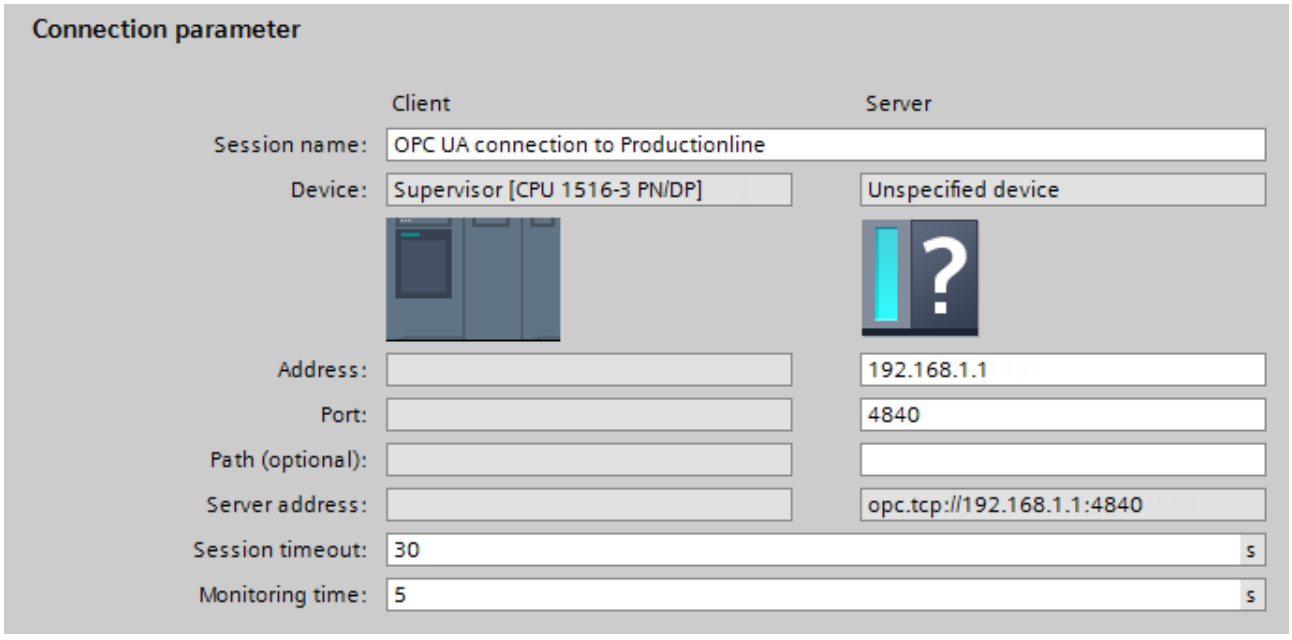


Figure 10-78 Connection parameters

4. If the OPC UA server is not using the standard port 4840, you must insert the port number here.
   For example, enter the number 65535 in the field, if the OPC UA server to which you want to establish a connection uses this port number.
5. In addition, you accept the default settings for session timeout (30 seconds) and monitoring time (5 seconds).

Setting the security parameters

1. Click the "Security" area in the "Configuration" tab.
   This area contains all security settings for the connection to the OPC UA server.

The following settings are possible, for example:

"General" area

Security mode:

Select the security mode that the connection to the OPC UA server must meet from the drop-down list.

If the server does not meet the selected mode, a session is not established.

The following settings are available:

- No security: No secure connection!
- Sign: OPC UA server and OPC UA client sign the data transmission (all messages): Manipulations can thus be detected.
- Sign & Encrypt: OPC UA server and OPC UA client sign and encrypt the data transmission (all messages):

Security policy:
Set the encryption techniques for the signing and encryption of messages.
The following settings are possible:

- No security
- Basic128Rsa15
- Basic256
- Basic256Sha256

To configure a secure connection, you must observe the following items:

- A certificate is required for the client for a secure connection.
- You have to make the client certificate known to the server.

To find out how to proceed, see the section "Handling client and server certificates (Page 224)" under "Certificate of the OPC UA client".

"Certificates" area

Client certificate:
The certificate confirms the authenticity of the OPC UA client.
To select a certificate, click the following symbol:

STEP 7 displays a list of certificates.
Select the certificate that you have made known to the server.
Click the symbol with the green check mark:

Or, create a new certificate. To do so, click the "Add" symbol.
If you create a new certificate, you must make this certificate known to the server.

"User authentication" area
The following settings are possible for user authentication:

- Guest
- User name and password
- Users (TIA Portal - Security Settings)

For more information, see Users and roles with OPC UA function rights (Page 232).

Setting languages

UA tags of the String type can be localized with OPC UA, that is, texts (values for the UA tag) can be available in different languages for the server. For example, localized texts can be available for DisplayName (Name of the node) and Description (Description).
In the "Languages" area of the "Configuration" tab you can, for example, influence the language of the texts returned by the server as follows:
In the "Languages" area, enter a number of languages that the server transfers to the client during connection setup.

The language or the local ID ("language code") associated with it that you enter in the first line is the language preferred by the client.

- If the server can provide the UA tag in the requested language, it is transferred to the client.
- If the server cannot provide the UA tag in the requested language, it checks whether it can provide the UA tag in the language you have entered in the second line (first substitute language).
- The server works its way down the list, and when it can provide neither the requested language nor a substitute language, it will provide the default language.

More information

What causes the connection to an OPC UA server to fail? FAQ.
(https://support.industry.siemens.com/cs/ww/en/view/109766709)

10.4.9.2 Handling of the client certificates of the S7-1500 CPU

Where does the client certificate come from?

If you are using the OPC UA client of an S7-1500 CPU (OPC UA client enabled), you can create certificates for these clients with STEP 7 V15.1 and higher as described in the following sections.

When you use UA clients from manufacturers or the OPC Foundation, a client certificate is generated automatically during installation or upon the first program call. You have to import these certificates with the global certificate manager in STEP 7 and use them for the respective CPU.

If you program an OPC UA client yourself, you can generate certificates through the program. Alternatively, you can generate certificates with tools, for example with OpenSSL or the certificate generator of the OPC Foundation:

- The procedure for OpenSSL is described here: "Generating PKI key pairs and certificates yourself (Page 175)".
- Working with the certificate generator of the OPC Foundation is described here: "Creating self-signed certificates (Page 174)".

Certificate of the OPC UA client of the S7-1500 CPU

A secure connection between the OPC UA server and an OPC UA client is only established if the server classifies the certificate of the client as trusted.

Therefore you have to make the client certificate known to the server.

The following sections describe how you can initially generate a certificate for the OPC UA client of the S7-1500 CPU and then make it available to the server.

1. Generate and export a certificate for the client

For a secure connection you have to generate a client certificate and - if the server and client are located in different projects - export the certificate.
If client and server are in the same project, exporting the client certificate and subsequent import are not necessary.

Requirements

The IP interface of the CPU is configured, an IP address is available.
Background: The IP address under which the CPU can be accessed in your system is entered under "Subject Alternative Name (SAN)".

Creating an OPC UA Client certificate

The easiest way to generate a client certificate for an S7-1500 CPU is to configure a client interface.
The configuration of the client interface provides for the selection or generation of a client certificate, see Creating and configuring connections (Page 323).
Alternatively, you can generate the client certificate as follows:

1. In the project tree, select the CPU you want to use as a client.
2. Double-click "Device configuration".
3. In the properties of the CPU, click "Protection & Security > Certificate manager".
4. Double-click "<Add new>" in the "Device certificates" table.
   STEP 7 opens a dialog.
5. Click the "Add" button.
6. Select the "OPC UA client" entry from the "Usage" list.
7. Click "OK".
   STEP 7 now shows the client certificate in the "Device certificates" table.
8. If the server is in another project: Right-click this line and select "Export certificate" from the shortcut menu.
9. Select a directory where you will store the client certificate.

2. Announcing the client certificate to the server

You have to make the client certificate available to the server to allow a secure connection to be established.
To do this, follow these steps:

1. If the client was configured in another project and you created and exported the client certificate there:
   - Select the "Use global security settings for certificate manager" option in the local certificate manager of the server. This makes the global certificate manager available.
     You will find this option under "Protection & Security > Certificate manager" in the properties of the CPU that is acting as server.
   - If the project is not yet protected, select "Security settings > Settings" in the STEP 7 project tree, click the "Protect this project" button and log on.
     The "Global security settings" item is now displayed under "Security settings" in the STEP 7 project tree.
   - Double click "Global security settings".

- Double click "Certificate manager".
  STEP 7 opens the global certificate manager.
- Click the "Device certificates" tab.
- Right-click in the tab on a free area (not on a certificate).
- Select the "Import" shortcut menu.
  The dialog for importing certificates is displayed.
- Select the client certificate that the server is to trust.
- Click "Open" to import the certificate.
  The certificate of the client is now contained in the global certificate manager. Note the ID of the client certificate just imported.

2. Click the "General" tab in the properties of the CPU that is acting as server.
3. Click "OPC UA > Server > Security > Secure Channel".
4. Scroll down in the "Secure Channel" dialog to the section "Trusted clients".
5. Double-click in the table on the empty row with "<add new>". A browse button is displayed in the row.
6. Click this button.
7. Select the prepared client certificate.
8. Click the button with the green check mark.
9. Compile the project.
10. Load the configuration onto the S7-1500 CPU (server).

Result

The server now trusts the client. If the server certificate is also considered trusted, the server and client can establish a secure connection.

10.4.9.3 User authentication

In the OPC UA client interface of the S7-1500, you can set what authentication is required for a user of the OPC UA client wishing to access the server. To do so, you must select the corresponding client interface in the project tree of the requested S7-1500 CPU under "OPC UA communication > Client interfaces" and select the type of user authentication in the Inspector window under "Properties > Configuration > Security".

Types of user authentication

The following options are available for user authentication:

- **Guest**
  The user does not need to verify authorization (anonymous access). The CPU creates an anonymous session for the user, and the OPC UA server does not check the authorization of the client user.

- **User name and password**
  The user must prove authorization (no anonymous access). The OPC UA server checks whether the client user is authorized to access the server. Authorization is given by the user name and the correct password. These inputs cannot be checked by the client interface, which means all values are accepted as being valid.

  NOTE

  STEP 7 stores user name and password unencrypted in the data block/instance data block. Recommendation: Use the user authentication for the "User (TIA Portal - Security Settings)" TIA project.

- **Users (TIA Portal - Security Settings)**
  You can enter a user name from the list of users entered in the project for authentication. The names of the registered users for the current project are available in the user administration in the project tree under "Security Settings > Users and roles". There you can also enter additional users.
  You can also enter a name that is not listed in the user administration of the project or leave the field blank. This is necessary when the corresponding user name is only provided by a different source during runtime, for example, via HMI or from a different OPC UA client.

"No Security" security policy and authentication via user name and password

You can set the following combination:
Security policy = "No Security" and authentication via user name and password.

- The OPC UA server of the S7-1500 supports this combination. OPC UA clients can connect and encrypt the authentication data or not.
- OPC UA client of the S7-1500 CPU also supports this combination: However, in runtime it only connects if it can send the authentication data encrypted via cable!

Result: With the following configuration, not connection can be established in runtime.

- S7-1500 as OPC UA client
- OPC UA server which supports no encryption of authentication data when "No Security" (="none") is set as security policy.

More information

You can find information about the users and roles with OPC UA function rights in the section Users and roles with OPC UA function rights (Page 232).

10.4.9.4 Using a configured connection

Introduction

This section shows you how to use a configured connection for OPC UA instructions (third step).

Requirements

- You have created a client interface and added PLC tags and PLC methods to this interface, see ("First step (Page 309)").
- You have configured a connection to an OPC UA server (Second step (Page 323)).

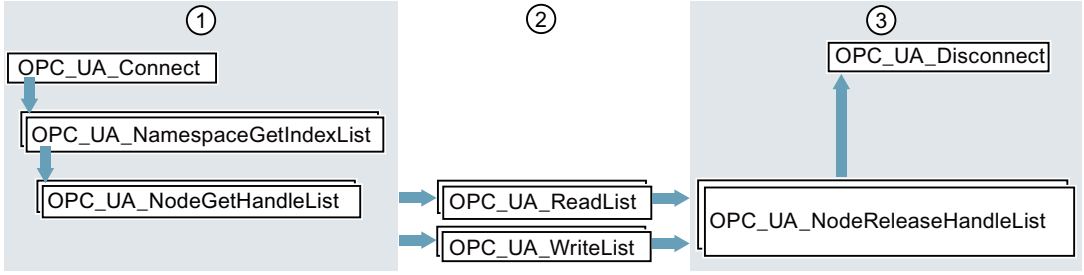
Overview

To read data from an OPC UA server or write data to an OPC UA server, use the following instructions:

- OPC_UA_Connect
- OPC_UA_NamespaceGetIndexList
- OPC_UA_NodeGetHandleList
- OPC_UA_ReadList or OPC_UA_WriteList
- OPC_UA_NodeReleaseHandleList
- OPC_UA_Disconnect

Order of the OPC UA instructions

The following figure shows the order in which the OPC UA instructions are called in a user program in order to use these instructions to read or write PLC tags:



① Instructions for preparation of read and write operations

② Read and write instructions

③ Instructions for "clean-up" after a completed read or write operation
The "OPC_UA_NodeReleaseHandleList" instruction can be omitted if "OPC_UA_Disconnect" is called immediately afterwards.

Figure 10-79 Call sequence for write and read operations

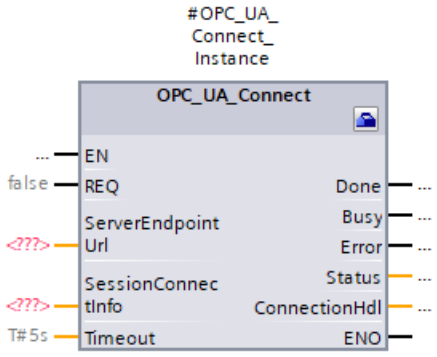
STEP 7 (TIA Portal) automatically supplies the parameters of these instructions if you are using a client interface and a configured connection to an OPC UA server.
The procedure is shown in the following section.

Using a client interface and configured connection

To use a configured OPC UA connection, follow these steps:

1. Open your user program in the TIA Portal.
2. Using drag-and-drop, move the "**OPC_UA_Connect**" instruction into the program editor.
   You will find the instruction under "Instructions > Communication > OPC UA" in the TIA Portal.

3. Select a call option for the instruction
   The example uses a multi-instance.
   STEP 7 displays the instruction in the program editor.
   The editor for the Function Block Diagram (FBD) programming language uses the following display:



   The editor for the Ladder Logic (LAD) programming language displays the instruction similarly.
4. Click the toolbox symbol in the editor for FBD or LAD.
   The symbol is located in the heading of the instruction:

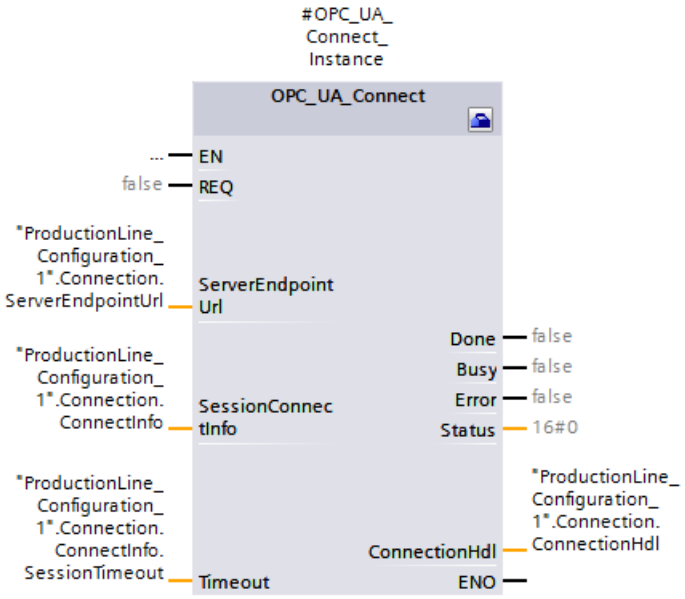
   

   If you are using the editor for STL or SCL: Click the small green rectangle below the first character of the instance name:

   ```
   #OPC_UA_Connect_Instance
   ```

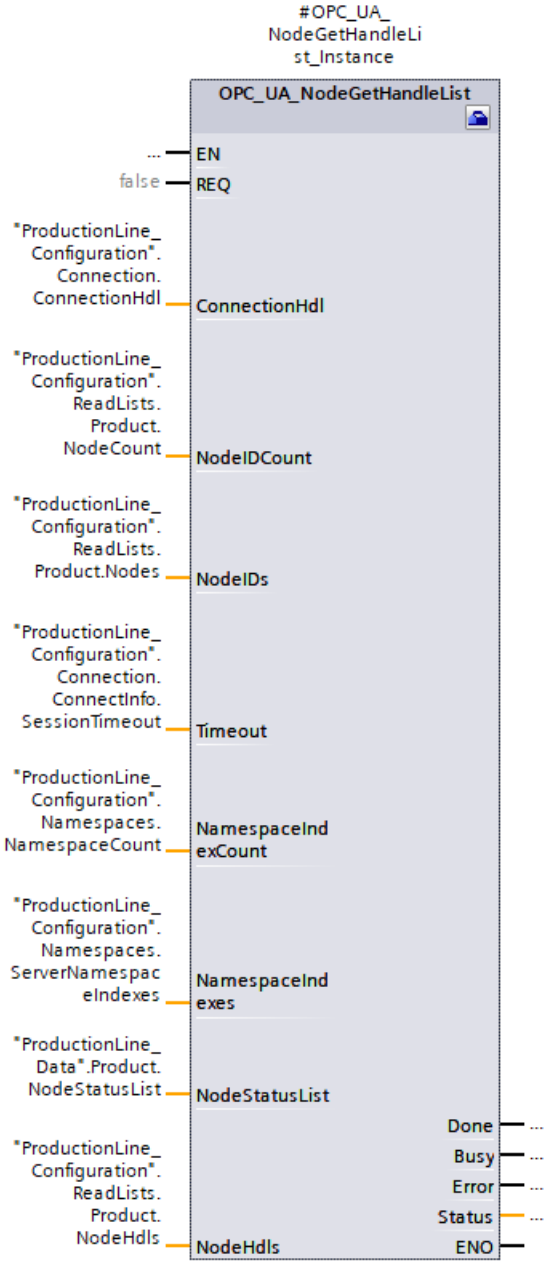
   The example (Page 308) uses "#OPC_UA_Connect_Instance" as the instance name.
   STEP 7 displays the properties of the instruction in a separate dialog.

5. For "Client interface" select the client interface that you want to use for the instruction.
   In the example, we select the "ProductionLine" client interface.
   STEP 7 now interconnects the "ProductionLine" client interface with the parameters of the OPC_UA_Connect instruction:



   "ProductionLine" is the interface that the OPC UA client of the example (Page 308) uses for data exchange with the OPC UA server "ProductionLine".
6. Using drag-and-drop, move the "**OPC_UA_NamespaceGetIndexList**" instruction into the program editor.
   You will find the instruction under "Instructions > Communication > OPC UA" in the TIA Portal.
   Select the "Multi-instance" call option.
   Click the toolbox symbol (LAD and FBD) or the small green box below the instance name (STL and SCL) if the editor is not already open.
   Select the client interface that you want to use (in the example "ProductionLine").
   STEP 7 now automatically interconnects all parameters of the "OPC_UA_NamespaceGetIndexList" instruction:
7. Using drag-and-drop, move the "**OPC_UA_NodeGetHandleList**" instruction into the program editor.
   Select the "Multi-instance" call option.
   Click the toolbox symbol (LAD and FBD) or the small green box below the instance name (STL and SCL) if the editor is not already open.
   Select the client interface that you want to use. The example uses the "ProductionLine" client interface.
   Under "Data access > Read/Write list" select the read or write list that you want to use (in the example the read list "Product").

STEP 7 now automatically interconnects all parameters of the "OPC_UA_NodeGetHandleList" instruction:



If you want to write data to an OPC UA server, select the write list you want to use under "Data access > Write list" (the "ProductionStatus" write list in the example).

8. Using drag-and-drop, move the "**OPC_UA_ReadList**" instruction into the program editor.
   Select the "Multi-instance" call option.
   Click the toolbox symbol (LAD and FBD) or the small green box below the instance name (STL and SCL) if the editor is not already open.
   Select the client interface that you want to use. The example uses the "ProductionLine" client interface.

Under "Data access > Read list" select the read list that you want to use (in the example the "Product" read list).

STEP 7 now automatically interconnects all parameters of the "OPC_UA_ReadList" instruction.

If you want to write data to an OPC UA server, use the "**OPC_UA_WriteList**" instruction and select the list of tags you want to send to the server under "Data access > Write list" ("ProductionStatus" write list in the example).

9. If you use different read lists or write lists as program-controlled lists in your user program, move the "**OPC_UA_NodeReleaseHandleList**" instruction to the program editor using drag-and-drop operation.

   Select the client interface that you want to use.

   Now select a read list or write list that you want to release: Only release read or write lists that you rarely use, since re-registering is time-consuming.

   Then, repeat the steps starting with step 7 with the "OPC_UA_NodeGetHandleList" instruction.

10. Using drag-and-drop, move the "**OPC_UA_Disconnect**" instruction into the program editor.

    Select the "Multi-instance" call option.

    Click the toolbox symbol (LAD and FBD) or the small green box below the instance name (STL and SCL) if the editor is not already open.

    Select the client interface that you want to use. The example uses the "ProductionLine" client interface.

    STEP 7 now automatically interconnects all parameters of the "OPC_UA_Disconnect" instruction.

Supported instructions

For the following instructions, STEP 7 automatically supplies the parameters if you are using a client interface and a configured connection to an OPC UA server:

- OPC_UA_Connect
- OPC_UA_NamespaceGetIndexList
- OPC_UA_NodeGetHandleList
- OPC_UA_MethodGetHandleList
- OPC_UA_MethodReleaseHandleList
- OPC_UA_ReadList
- OPC_UA_WriteList
- OPC_UA_MethodCall
- OPC_UA_NodeReleaseHandleList
- OPC_UA_Disconnect

Compact instructions

As of TIA Portal V17, compact instructions are available for OPC UA which summarize the write job/read job/method call and the connection establishment:

- OPC_UA_ReadList_C for generating a connection and reading tags
- OPC_UA_WriteList_C for generating a connection and writing tags
- OPC_UA_MethodCall_C for generating a connection and calling methods

You will find information about the compact instructions services in the TIA Portal Help.

10.5 Tips and recommendations

10.5.1 Rules for subscriptions

The following rules apply to subscriptions:

- Group subscriptions in the client according to different sampling and publishing intervals and distribute the monitored elements (variables) to these groups.
  Example: Create a subscription for longer publishing intervals (e.g. 5 seconds) and a subscription for shorter publishing intervals (e.g. 0.1 second).
- Disable unneeded subscriptions.
  Reason: The "Deactivated" subscription mode reduces resource consumption.
- Consider the maximum number of monitored items of subscriptions for the corresponding S7-1500 CPU.
  The information can be found in the technical specification of the respective CPU. The information is based on a sampling / publishing interval of 1 second.
  You can find more information in the FAQ 109755846 (https://support.industry.siemens.com/cs/us/en/view/109755846).
- Select the same sampling and publishing intervals for the OPC UA client and for the OPC UA server.
- Avoid arrays and structures as elements of subscriptions – if the process allows.
  Reason: If even one value of an array/structure changes, the entire structure is transferred, creating an unnecessary communication load.
- Occasional non-compliance with the required sampling rate is acknowledged by the OPC UA server of the S7-1500 CPU according to OPC UA specification with a "GoodOverload" error code, see also TIA Portal Help. Different OPC UA clients handle "Good" error codes unequal to "0" differently. Consider this behavior and, if needed, reduce the communication load according to the measures described above.

More information

For information on how to set the server for subscriptions, refer to the section Settings of the server for subscriptions (Page 222).

10.5.2 Rules for the user program

User programs for OPC UA

The following rules apply to user programs:

- If your application allows it and the communication load is high, you should set a minimum time for cycle OBs.
  Advantages:
  - The cycle time remains constant for the most part
  - The CPU has more time for communication tasks throughout

  Tip: Use the instruction "Runtime_Info"; mode 21 or mode 25 (see TIA Portal Help) to analyze the CPU utilization (e.g. communication).
- Reduce the number of variables or data blocks that can be reached from OPC UA/HMI. By default, all variables from OPC UA/HMI are accessible when creating variables/DBs/IDBs. This measure leads to improved performance when loading in RUN.
  Tip: Using the detailed object display in the TIA Portal, you can easily mark the non-OPC-UA-relevant data blocks as "not accessible from OPC UA".
- Consistent transfer of data beyond the limits of simple data types is only possible with OPC UA methods. If you use other OPC UA functions (Subscriptions, Read/Write), you must ensure data consistency in the application.
- OPC UA offers the "RegisterNodes" service for repeated read and write accesses to the same variables. Servers can use this service to prepare for optimized access to variables. The instruction "OPC_UA_NodeGetHandleList" of the S7-1500 as OPC UA client implicitly calls this service to prepare the server for optimized accesses (in OPC UA usage "Registered Read/Write").

Calling detailed object display in the TIA Portal

To call up the detailed object display, proceed as follows:

1. Switch to the "PLC Programming" portal in the portal view.
2. Select "Show all objects".
3. Switch to the "Details" tab in the selection window.

4. In the "DB accessible from OPC UA" column, disable the accessibility from OPC UA for individual objects.

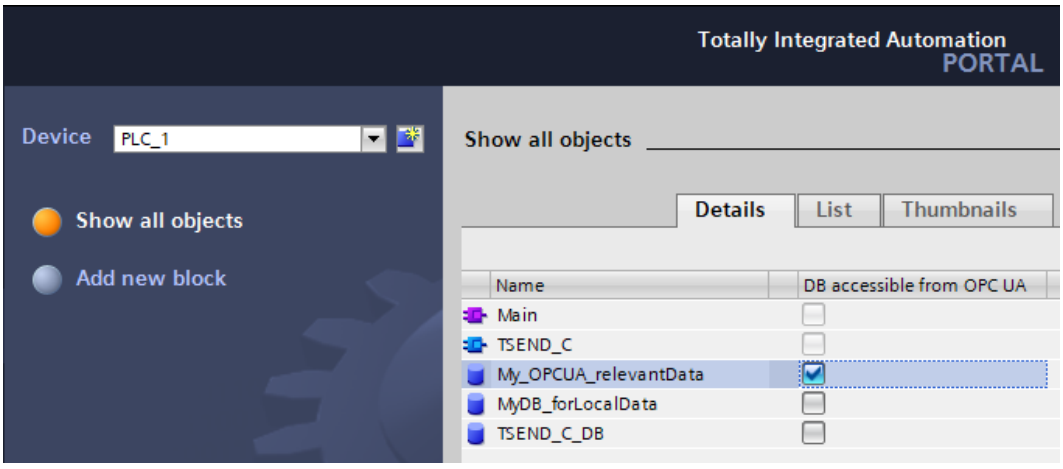


Figure 10-80 Calling detailed object display in the TIA Portal

10.5.3 Master copies for OPC UA communication

Master copies for the OPC UA interfaces

Interfaces of OPC UA servers and OPC UA clients that you want to use multiple times can be stored either in the project library or in a global library. Master copies in the project library can only be used within the project. When you create the master copy in a global library, it can be used in different projects.

The OPC UA capable CPUs differentiate between 3 interface types of the OPC UA server:

- Standard OPC UA server interface
- Companion specification interface
- Namespace reference

When adding the OPC UA interface in the project tree under "OPC UA Communication" each interface type gets its own symbol. The same symbol is used by the master copy.

Create either single master copies or a master copy with multiple interfaces.

Creating multiple master copies from selection

You select one or more elements and create individual master copies from them

1. Open the library in the "Libraries" task card.
2. Select the desired elements.
3. Using a drag-and-drop operation, move the elements to the "Master copies" folder or any subfolder of "Master copies".

Creating a master copy from selection

You select multiple elements and create a single master copy from them that contains all selected elements.

1. Copy to the clipboard the elements that you want to create as master copies.
2. Right-click on the "Master copies" folder or any of its subfolders in the library.
3. In the shortcut menu, select "Paste as a single master copy" command.

If multiple interfaces are added to a master copy from the OPC UA server or OPC UA client, the label and the symbol in the library are changed accordingly.
A symbol with "+" is displayed instead of the simple symbol.

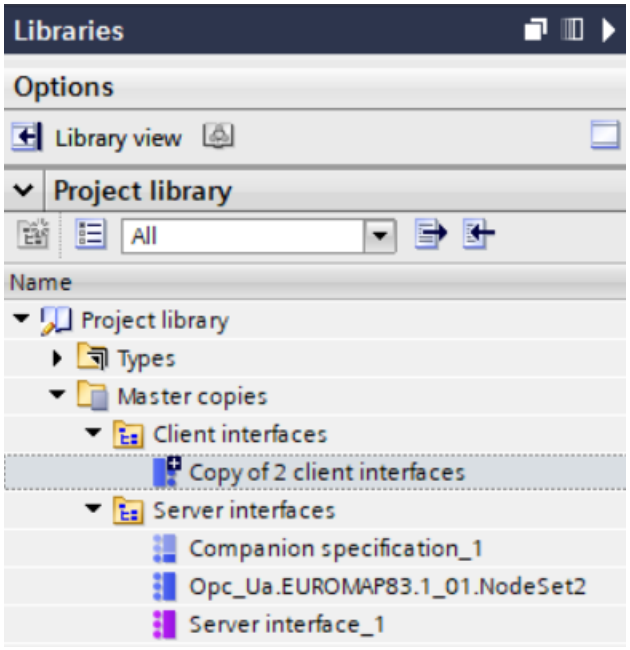


Figure 10-81 Create copy template in STEP 7

More information

For information on how to create a user-defined server interface, refer to the section Creating a user-defined server interface (Page 249).

Addressing via DHCP

11

In order to provide future-proof, efficient and flexible automation, more and more components from the production area support IT standards. Worldwide Ethernet standards, integrated communication and versatility make IT-supported automation an economical solution for your requirements. Functional expansions of the communication options of the S7-1500 CPUs in this direction give you more freedom for the possible uses of your system or machine. You use IT technology to automate efficiently. With the introduction of DHCP and the expansions of DNS for S7-1500 CPU as of firmware version V2.9, you gain more flexibility for the design of your automation solution.

For the interfaces of an S7-1500 CPU you can set that address parameters such as the IP address with subnet mask, can be got from a DHCPv4 server (hereinafter DHCP server).

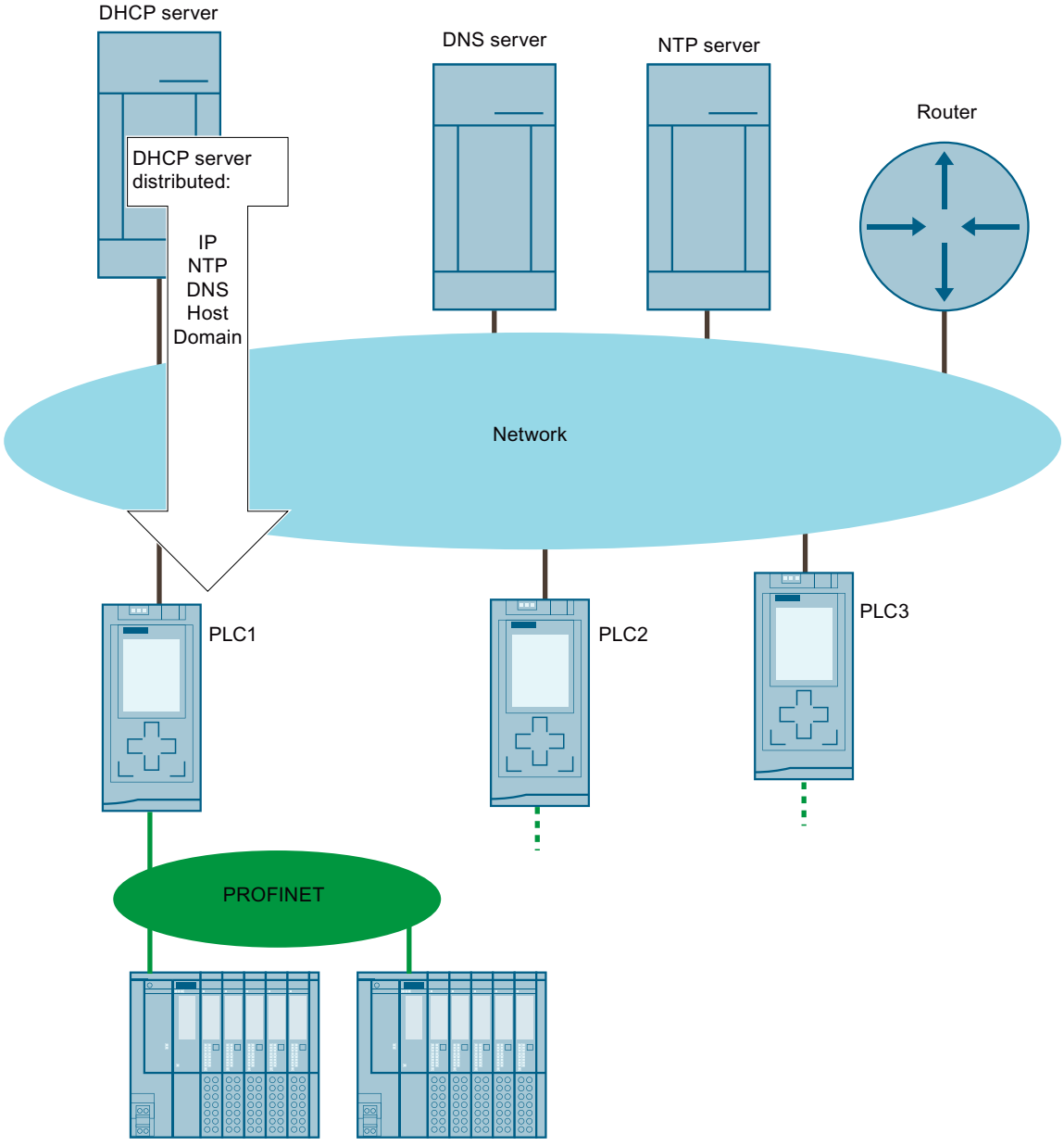


Figure 11-1 Overview of DHCP

Areas of application

- Use of the S7-1500 CPU in a managed IT environment
- Adding new devices in a modular manufacturing structure

11.1 Principle of address assignment via DHCP

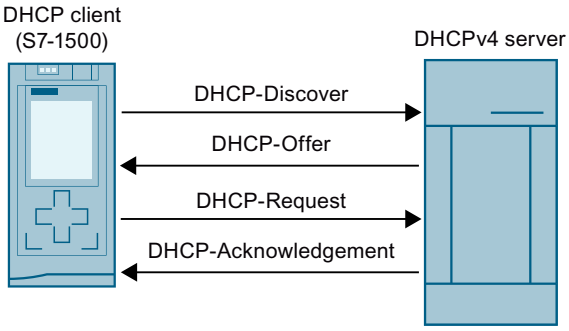
Requirement configuration

The following requirements must be met so that a PROFINET interface of the S7-1500 CPU can obtain IP address parameters via a DHCP server:

- The address assignment via a DHCP server is configured.
  Activate DHCP (Page 347)
- No PROFINET IO communication may be configured for the interface.

Principle of DHCP address assignment

The process begins as soon as the project is loaded into the CPU or the CPU is switched on with configured DHCP address assignment and started up, the process of DHCP assignment begins:



DHCP Discover	The DHCP client searches for a suitable DHCP server via broadcast. The DHCP client identifies itself to the DHCP server with the configured client ID or with its MAC address.
DHCP Offer	The DHCP server offers the DHCP-client IP address parameters (IPv4 address, subnet mask, optional default router) and if necessary further data (options).
DHCP Request	The DHCPclient requests the IP address parameters and options offered in the DHCP offer. The DHCP client of the S7-1500 CPU always accepts the first DHCP offer of a DHCP server that meets the requirements (IP address with subnet mask).
DHCP Acknow-ledgment	The DHCP server confirms and transmits the IP address parameters and options offered in the DHCP offer. The DHCP server also notifies the DHCP client how long the DHCP client can use the address parameters (lease time).

Figure 11-2 Principle of address assignment with DHCP

The IP address parameters and options are stored in the load memory of the CPU. After a general reset or restart of the CPU, the IP address parameters and options are obtained again via DHCP.

DHCP address assignment options

For the S7-1500 CPU you can configure that the following options are obtained via a DHCP server:

- Addresses of up to four DNS servers
  Get addresses of the DNS servers via DHCP (Page 349)
- Address of up to four NTP servers
  Get addresses of the NTP servers via DHCP (Page 349)
- Host and domain name
  Obtain host and domain name via DHCP (Page 350)

If necessary, the DHCP server also supplies the address of a router (default gateway) as option.

How long can the S7-1500 CPU use the DHCP address parameters?

In addition to the address parameters, the DHCP server also notifies the S7-1500 CPU (DHCP client)of the lease time. The lease time defines how long the CPU can use the address parameters.

When the lease time has fully expired, the CPU returns the assigned address parameters. The CPU has an internal time monitoring for the lease time.

At certain times when the lease time expires, the CPU has the option of extending the lease time:

- Renewal: Half of the lease time has expired: The CPU contacts the original DHCP server and asks for an extension of the lease time. The original DHCP server can either confirm the existing lease time or assign a new lease time. With a new lease time, the time monitoring in the CPU is reset.
- Rebinding: 7/8 of the lease term has expired: The CPU contacts all available DHCP servers via broadcast and asks for an extension of the lease time. A DHCP server can either confirm the existing lease time or assign a new lease time. With a new lease time, the time monitoring in the CPU is reset.
  In the event of a negative response from DHCP server during rebinding or if no DHCP server replies, the CPU returns the address parameters after 8/8 of the lease time.

If the CPU has returned the address parameters after the lease time has expired, the CPU starts a new cycle for DHCP addressing with a new DHCP discover.

More information

For information on how to configure client ID, refer to the section Configuring the client ID (Page 348).

11.2 DHCP with DNS

As of STEP 7 V17, the S7-1500 CPU supports the host name and domain address parameters used in name-based communication (DNS).

For specific communication services, name-based addressing via the complete name consisting of host name and domain, is useful:

- The CPU is addressable under the complete name, for example at OPC UA, Open User Communication. In the case of dynamic IP address assignment by the DHCP server, unique addressing is always possible through the DNS name.
- Certificates of the S7-1500 CPU may contain the complete name, for example, for OPC UA communication, web server, secure communication.
  - Only if you configure host name and domain for the S7-1500 CPU in STEP 7, then the complete name is entered in the device certificates in the project as subject alternate name (SAN).
  - As soon as you obtain the host name and/or domain via DHCP or assign it via the user program, the complete name is not stored in the device certificates in the project.

How you set the DNS configuration for your CPU depends on how you assign the host name and domain in your network .

- Central assignment of host name and domain

  You assign the host name and domain centrally in your network, for example via a configured DNS server. In STEP 7 you configure that the CPU obtains the host name and domain via DHCP.

  In the following configuration, only the client ID is configured in the S7-1500 CPU. At the DHCP address assignment the DHCP-server returns the host name and domain options to the CPU.

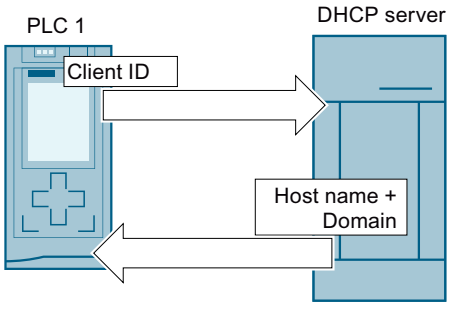


Figure 11-3 Obtain host and domain name via DHCP

  For this configuration, you must first activate the host name and domain configuration in STEP 7. You then configure that the host and domain name are obtained via DHCP.

  Obtain host and domain name via DHCP (Page 350)

- Local assignment of host name and domain

  You can configure the host name and domain in STEP 7 or assign them in the user program.

NOTE

Validity of the data obtained from DHCP

If you change the host name and/or domain in the user program, then all data obtained via DHCP (IP suite, host name, domain, NTP server, DNS server) becomes invalid and is retrieved again from DHCP server. Therefore, you should only change the Hostname and/or Domain in urgent cases and not during operation.

All connections can be dropped if the IP address of the interface changes.

In the following configuration, the host name and its domain are configured in the S7-1500 CPU in addition to its client ID. When assigning DHCP addresses, the CPU supplies

the client ID as well as the host name and the domain to the DHCP server.. The DHCP server receives the information to update, for example a DNS server with the address data of the CPU.

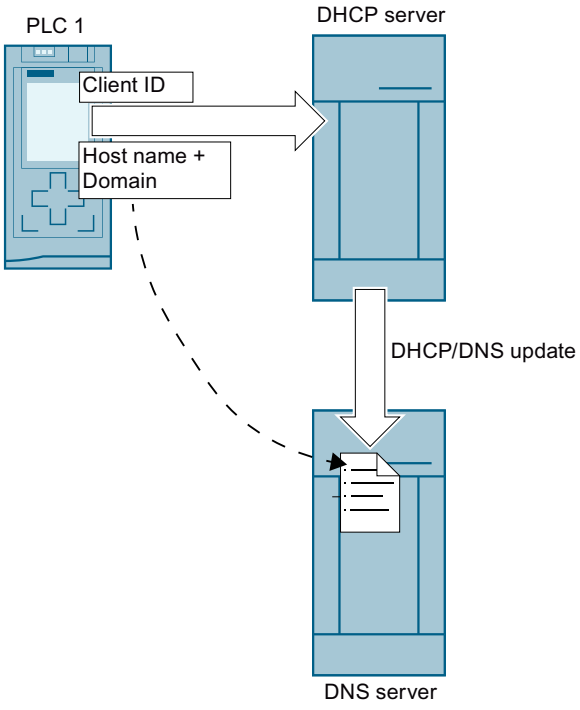


Figure 11-4 Configure host and domain name

For this configuration, you must first activate the host name and domain configuration in STEP 7. You then configure the host name and domain in STEP 7.

- Central assignment of the domain and local assignment of host name.
  - You configure in STEP 7 that the CPU gets the domain via DHCP.
  - You can configure the host name in STEP 7 or assign it via the user program.

  In the following configuration, the host name is also configured in the S7-1500 CPU in addition to its client ID. When assigning DHCP addresses, the CPU supplies the client ID as well as the host name to the DHCPv4 server. The DHCP server supplies the domain option to the CPU.

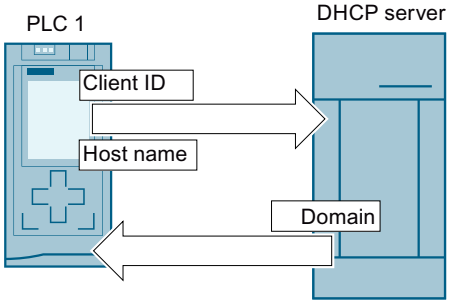


Figure 11-5 Configure host name, obtain domain name via DHCP

For this configuration, you must first activate the host name and domain configuration in STEP 7. Then configure the host name in STEP 7 and configure that the domain is obtained via DHCP.

Requirements

- You have activated the address assignment via DHCP for at least one interface of the S7-1500 CPU.

Configuration of host and domain name

To activate the host name and domain configuration in STEP 7, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name" > "Host and domain name configuration".
3. Select the "Enable host name and domain" check box.

Configuring the host name in STEP 7

To configure the host name in STEP 7, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name" > "Host and domain name configuration"> "Host name".
3. For "Host name configuration:" select "Set host name in the project" from the drop-down list.
4. Enter the host name for "Host name:".
   - Enter your desired host name.
   - If you select the "Host name identical to device name" check box, STEP 7 automatically assigns the device name as the host name.

Only if you have configured the host and domain name in STEP 7, the full name is displayed for "Full name:".

Assign host name in the user program

To assign the host name in the user program, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name" > "Host and domain name configuration" > "Host name".
3. Under "Host name configuration:" select "Set hostname directly on the device (e.g. PLC program, display)" in the drop-down list.
4. Call the instruction "CommConfig" in the user program. The DATA parameter must point to a UDT "Conf_Hostname" where the host name is defined.

You can find more information on the "CommConfig" instruction and on the UDT "Conf-Hostname" in the STEP 7 online help.

Configuring a domain in STEP 7

To configure the domain in STEP 7, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name" > "Host and domain name configuration" > "Domain".
3. For "Domain configuration:" select "Set domain in the project" from the drop-down list.
4. Enter your desired domain under "Domain:".

Only if you have configured the host and domain name in STEP 7, the full name is displayed for "Full name:".

Assigning a domain in the user program

To assign the domain in the user program, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name" > "Host and domain name configuration" > "Domain".
3. Under "Hostname configuration:" select "Set domain directly on the device (for example PLC program, display)" in the drop-down list.
4. Call the instruction "CommConfig" in the user program. The DATA parameter must point to a UDT "Conf_Domainname" where the domain name is specified.

You can find more information on the "CommConfig" instruction and on the UDT "Conf-Domainname" in the STEP 7 online help.

Rules for maximum lengths of host name, domain and client ID

Note the following maximum lengths in bytes. One byte corresponds to one character:

- Host name: Maximum 63 bytes
- Domain: Maximum 252 bytes
- Client ID: Maximum 254 bytes
- Host + Domain name: Maximum 254 bytes
- Host + Domain name + Client ID: Maximum 260 bytes
  Applies only when host and domain name must be sent to the DHCP server.

11.3 Activate DHCP

Requirements

- S7-1500 CPU firmware V2.9 or higher

Procedure

To activate DHCP for the PROFINET interface of an S7-1500 CPU, follow these steps:

1. Select the PROFINET interface of the S7-1500 CPU in STEP 7.
2. In the properties of the interface, navigate to "Ethernet addresses" > "Internet Protocol Version 4 (IPv4)".
3. Select the option "IP address of DHCP server".

Result

You have set the interface so that it obtains your IP address via a DHCP server.
"Use MAC address as client ID" is set as the operating mode for DHCP on the S7-1500 CPU.
How to adjust the client ID is described under Configuring the client ID (Page 348).

11.4 Configuring the client ID

The client ID

The S7-1500 CPU always identifies itself to a DHCP server with the client ID (DHCP option 61). The client ID is interface specific.
The S7-1500 CPU supports the following two operating modes with regard to the client ID:

- Use the MAC address as the client ID: The MAC address of the CPU is used as the client ID for the DHCP client. Note, if you execute a device exchange of the CPU in this operating mode, the MAC address and therefore also the client ID changes.
- User-defined client ID: With this option you specify the client ID in the configuration in STEP 7. You also have the option of adapting the client ID during runtime, for example, in the user program using the "CommConfig" instruction.
  If you perform a device exchange of the CPU in this operating mode, the new CPU is assigned the configured client ID.

Requirement

- You have activated address assignment via DHCP for the interface.

Configuring the client ID

To configure the client ID in STEP 7, follow these steps:

1. Select the PROFINET interface of the S7-1500 CPU in STEP 7.
2. In the properties of the interface, navigate to "Ethernet addresses" > "Internet Protocol Version 4 (IPv4)"> "IP address of DHCP server".
3. For "Operating mode:" select the required operating mode from the drop-down list:
   – Use MAC address as client ID (default setting)
   – User-defined client ID

   If you have selected the option "Use MAC address as client ID", then you are already finished here. For "User-defined client ID", continue with step 4.

4. Enter a valid client ID for "Client ID".
   - A character string with 7-bit ASCII characters is allowed in the area from 0x21 to 0x7e.
   - Some DHCP servers expect a leading "0" (e.g. some SCALANCE devices). In this case, enter "\ 0" followed by your client ID.
   - You can also leave the field blank. In this case, you must select the "Client ID can be changed at runtime" check box.
5. In order to make the user-defined client ID adaptable during runtime, select the "Client ID can be changed during runtime" check box.

Adapting the client ID during runtime

You can use the "CommConfig" instruction to adapt the client ID via the user program. Call the instruction. The DATA parameter must point to a UDT "Conf_ClientId" or a UDT "Conf_ClientId_Opaque". The client ID must be specified in the UDT.
If you have left the user-defined client ID free in the configuration in STEP 7, then the CPU uses the MAC address as the client ID until the client ID is adapted for the first time.

NOTE
Validity of data obtained via DHCP
If you change ClientId with "CommConfig", all data obtained via DHCP will be invalid: IP Suite, domain name, NTP server, DNS server. Therefore, you should only change ClientId in urgent cases and not during operation.

You can find more information on the instruction "CommConfig" and the UDTs "Conf_ClientId" and "Conf_ClientId_Opaque" in the STEP 7 online help.

11.5 Get addresses of the DNS servers via DHCP

Requirements

- You have activated the address assignment via DHCP for at least one interface of the S7-1500 CPU.

Get addresses from DNS servers via DHCP

To obtain the addresses of up to 4 DNS servers via DHCP, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "DNS configuration" > "Server list".
3. For "Name resolution via DNS:" select "Set DNS server remotely (e.g. DHCP)" in the drop-down list.

Result: If the DHCP server supplies addresses from DNS servers as option, the CPU uses up to 4 addresses.

11.6 Get addresses of the NTP servers via DHCP

Requirements

- You have activated the address assignment via DHCP for at least one interface of the S7-1500 CPU.

Obtaining addresses from NTP servers via DHCP

To obtain the addresses of up to four NTP servers via DHCP, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Time of day" > "Time synchronization"> "NTP mode".
3. For "Time synchronization:" select "Set NTP server remotely (e.g. DHCP)" in the drop-down list.

Result: If the DHCP server supplies addresses from NTP servers as option , the CPU uses up to 4 addresses.

11.7 Obtain host and domain name via DHCP

Requirement

- You have activated the address assignment via DHCP for at least one interface of the S7-1500 CPU.
- You have activated the host name and domain configuration in STEP 7.

Obtain host name via DHCP

To obtain the host name via DHCP, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name"> "Host and domain name configuration"> "Host name".
3. For "Host name configuration:" select "Set host name remotely (e.g. DHCP)" in the drop-down list.

Result: If the DHCP server supplies a host name as option, the CPU uses this host name.

Obtaining a domain via DHCP

To obtain the domain via DHCP, follow these steps:

1. Select the S7-1500 CPU in STEP 7.
2. In the properties of the CPU, navigate to "Advanced configuration" > "Host and domain name" > "Host and domain name configuration" > "Domain".
3. For "Domain configuration:" select "Set domain remotely (e.g. DHCP)" in the drop-down list.

Result: If the DHCP server delivers a domain as option, the CPU uses this domain.

Routing

12

12.1 Overview of the routing mechanisms of S7-1500 CPUs

The following table gives an overview of the routing mechanisms of the S7-1500 CPU.

Routing mechanism	Description	Applications	Section
S7 routing	S7 routing is the transfer of data beyond S7 subnet boundaries. You can send information from a transmitter to a receiver across several s7 subnets.	Download user programs Load hardware configuration Execute testing and diagnostics functions	S7 routing (Page 352)
IP forwarding	IP forwarding is a function of devices to forward IP packets between two connected IP subnets.	Simple access from the control level to the field level for configuration and parameter assignment of devices, e.g. via PDM or Web browser. Simplified integration of devices for remote access, e.g. for diagnostics during remote maintenance or firmware update.	IP forwarding (Page 356)
Data record routing	Data can be sent over PROFINET from an engineering station to field devices via multiple networks. Since the engineering station addresses the field devices using standardized records and these records are routed via S7 devices, the term "data record routing" is used to refer to this type of routing.	Data record routing is used, for example, when field devices of different manufacturers are used. The field devices are addressed using standardized data records (PROFINET) for configuration and diagnostics.	Data record routing (Page 364)

12.2 S7 routing

Definition of S7 routing

S7 routing is the transfer of data beyond S7 subnet boundaries. You can send information from a transmitter to a receiver across several s7 subnets. The gateway from one S7 subnet to one or more other subnets is provided by the S7 router The S7 router is a device which has interfaces to the respective S7 subnets. S7 routing is possible via various S7 subnets (PROFINET/Industrial Ethernet and/or PROFIBUS).

Requirements for S7 routing

- All devices that can be reached in a network have been configured in a project in STEP 7 and downloaded.
- All devices involved in the S7 routing must receive routing information about the S7 subnets that can be reached through specific S7 routers. The devices obtain the routing information by downloading the hardware configuration to the CPUs, since the CPUs play the role of an S7 router.
  In a topology with several consecutive S7 subnets, the following order must be kept to when downloading: First download the hardware configuration to the CPU(s) directly connected to the same S7 subnet as the PG/PC, then download one by one the CPUs of the S7 subnets beyond this starting with the nearest S7 subnet through to the S7 subnet furthest away.
- The PG/PC you want to use to establish a connection via a S7 router must be assigned to the S7 subnet it is physically connected to. You can assign the PG/PC to a PG/PC in STEP 7 under Online & Diagnostics > Online accesses > Connection to interface/subnet.
- For S7 subnets of the type PROFIBUS: Either the CPU must be configured as DP master or, if it is configured as a DP slave, the "Test, commissioning, routing" check box must be selected in the properties of the DP interface of the DP slave.
- S7 routing for HMI connections is possible as of STEP 7 V13 SP1.

NOTE

Firewall and S7 routing

A firewall does not recognize the IP address of the sender during S7 routing when the sender is located outside the S7 subnet adjacent to the firewall.

An overview of the devices that support the "S7 routing" function is provided in this FAQ (https://support.industry.siemens.com/cs/ww/en/view/584459).

S7 routing for online connections

With the PG/PC, you can reach devices beyond S7 subnets, for example to do the following:

- Download user programs
- Download a hardware configuration
- Execute test and diagnostics functions

In the following figure, CPU 1 is the S7 router between S7 subnet 1 and S7 subnet 2.

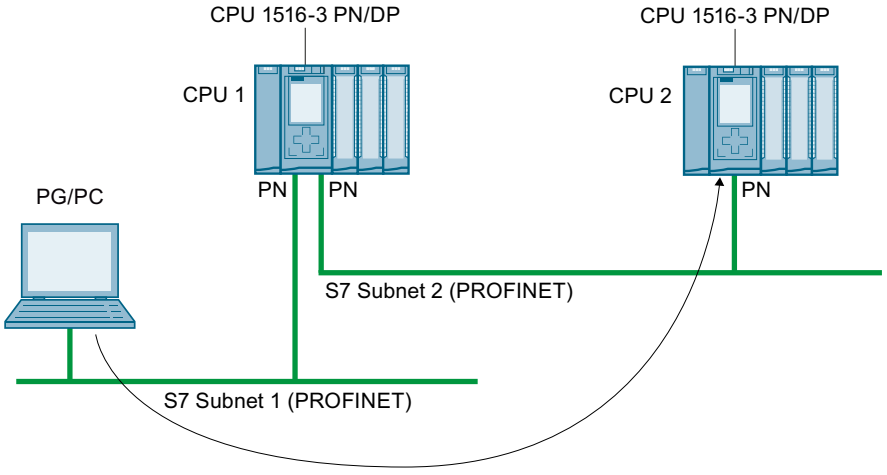


Figure 12-1 S7 routing: PROFINET - PROFINET

The following figure shows the access from a PG via PROFINET to PROFIBUS. CPU 1 is the S7 router between S7 subnet 1 and S7 subnet 2; CPU 2 is the S7 router between S7 subnet 2 and S7 subnet 3.

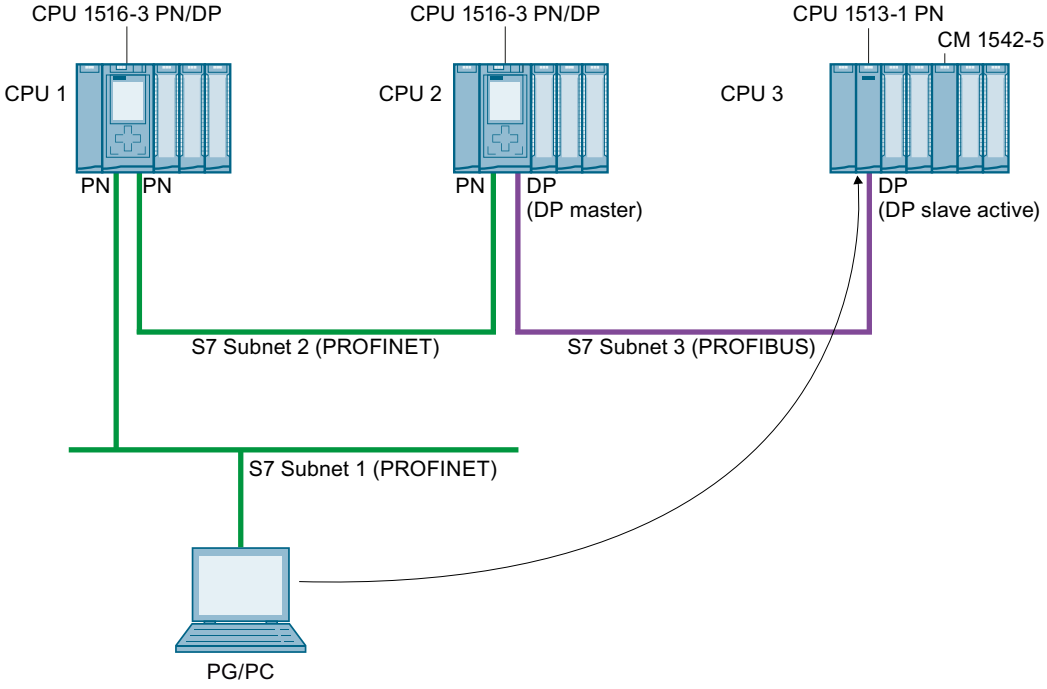


Figure 12-2 S7 routing: PROFINET - PROFIBUS

S7 routing for HMI connections

You have the option of setting up an S7 connection from an HMI to a CPU via different subnets (PROFIBUS and PROFINET or Industrial Ethernet). In the following figure, CPU 1 is the S7 router between S7 subnet 1 and S7 subnet 2.

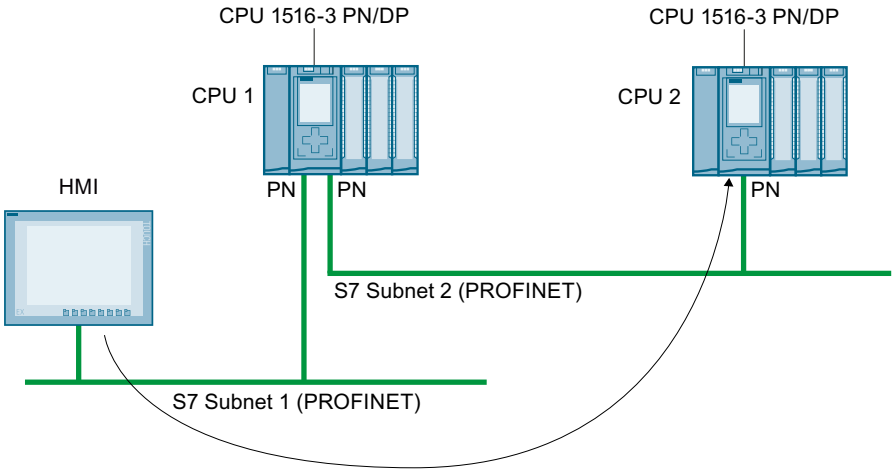


Figure 12-3 S7 routing via HMI connections

S7 routing for CPU-CPU communication

You have the option of setting up an S7 connection from a CPU to another CPU via different subnets (PROFIBUS and PROFINET or Industrial Ethernet). The procedure is described based on examples in the section S7 communication (Page 138).

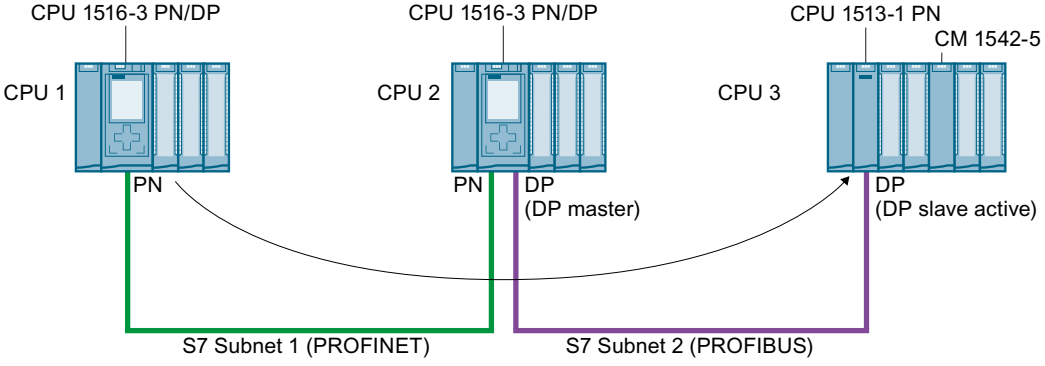


Figure 12-4 S7 routing via CPU-CPU communication

Using S7 routing

For the CPU, select the PG/PC interface and the S7 subnet in the "Go online" dialog of STEP 7. S7 routing is performed automatically.

Number of connections for S7 routing

The number of connections available for S7 routing in the S7 routers (CPUs, CMs or CPs) can be found in the technical specifications in the manuals of the relevant CPU/CM/CP.

S7 routing: Example of an application

The figure below shows the example of an application for remote maintenance of a system using a PG. The connection is made here beyond two S7 subnets via a modem connection.

You configure a remote connection via TeleService in STEP 7 using "Online access" or "Go online".

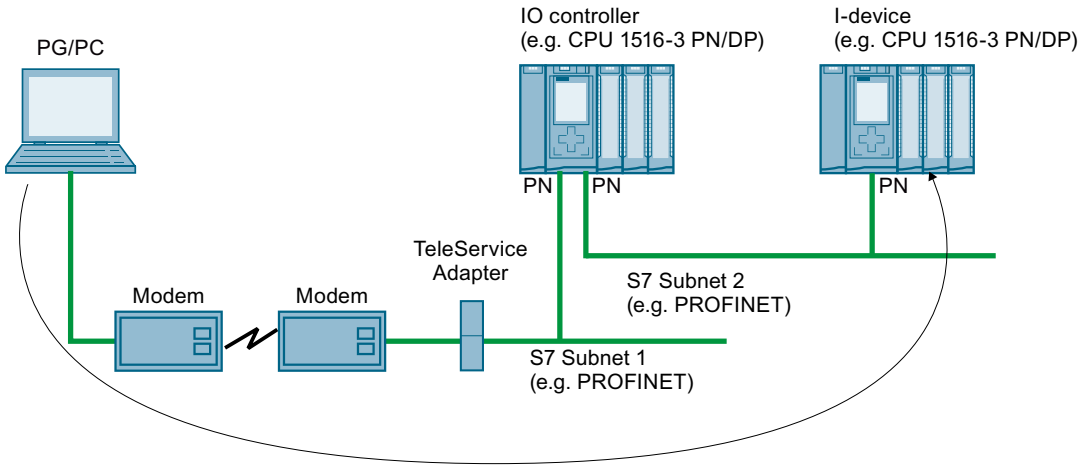


Figure 12-5 Remote maintenance of a plant using TeleService

More information

- The allocation of connection resources with S7 routing is described in the section Allocation of connection resources (Page 372).
- You can find more information on setting up TeleService in the STEP 7 online help.
- For information on HMI communication, refer to the section HMI communication (Page 113).
- You can find more information on S7 routing and TeleService adapters when you search the Internet using the following links:
  - Device manual Industrial Software Engineering Tools TS Adapter IE Basic (https://support.industry.siemens.com/cs/us/en/view/51311100)
  - Downloads for the TS Adapter (https://support.industry.siemens.com/cs/us/en/ps/16006/dl)

12.3 IP forwarding

Forwarding of IP packets with IP forwarding

IP forwarding is a function of devices to forward IP packets between two connected IP subnets.

Enable/disable the IP forwarding function in STEP 7. When IP forwarding is enabled, the S7-1500 CPU forwards received IP packets not addressed to the CPU to locally connected IP subnets or to a configured router.

The following figure shows how a programming device accesses data of an HMI device. Programming device and HMI device are located in different IP subnets. The IP subnets are connected to the two interfaces X1 and X2 of the CPU.

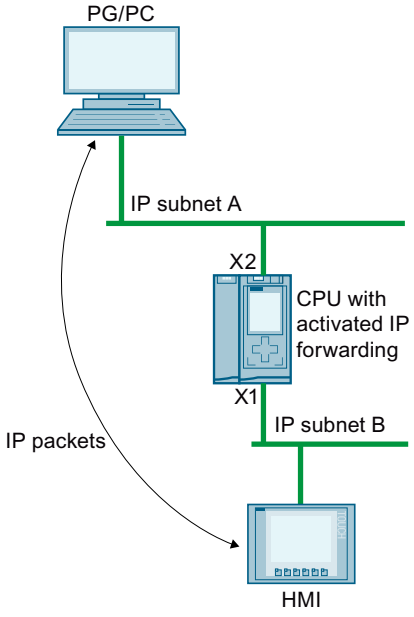


Figure 12-6 Access of a programming device to an HMI via IP forwarding

Areas of application

- Easy access from the control level to the field level for configuration and parameter assignment of field devices, e.g. via PDM or web browser
- Simplified integration of devices for remote access, e.g. for diagnostics during remote maintenance or firmware update

Requirements for using IP forwarding

- S7-1500 CPU as of firmware version V2.8
- Number of Ethernet interfaces:
  - The CPU has at least two Ethernet interfaces.
  - Or the CPU has one Ethernet interface, and a CP 1543-1 as of firmware version V2.2 provides the other Ethernet interface. In this case, the "Access to PLC via communication module" function must be enabled for the CP in the CPU.
- IP forwarding is enabled.
- Suitable standard gateways/routes are configured in each participating device along the outgoing and return paths of the IP packets.

IP route table

When IP forwarding is enabled, the CPU forwards received IP packets that are not addressed to itself. How the CPU forwards the IP packets is defined in its internal IP route table.
The CPU automatically creates the IP route table from the following information of the loaded hardware configuration:

- IP configuration of the Ethernet interfaces
- Configured router

Example of a configuration with IP forwarding

The following figure shows a sample configuration along with the required IP address settings and router settings.

- A PC on the IP subnet 192.168.4.0 communicates with an HMI device on the IP subnet 192.168.2.0.
- The IP address of a router ("Standard Gateway") is configured at the CPU, Ethernet interface X3; in the figure below it is the device that is designated as "IP Router".
  In STEP 7, you configure a router in the interface properties under "Ethernet Addresses" > "IP Protocol".

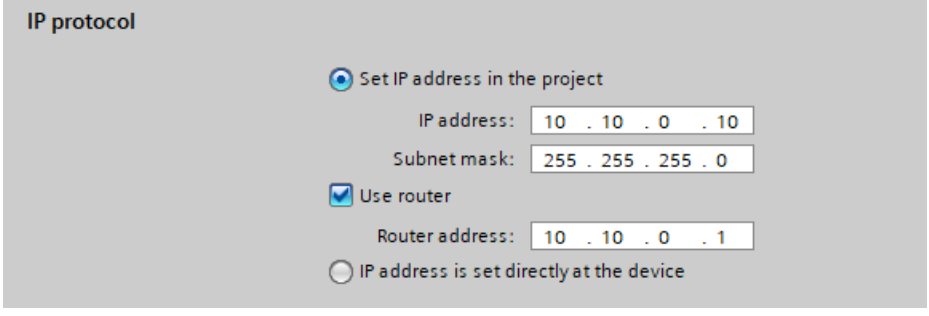


Figure 12-7 Configuring the router

- For the PC, the IP router, the IO device and the HMI device, the IP addresses of a standard gateway or the corresponding routes are also entered.

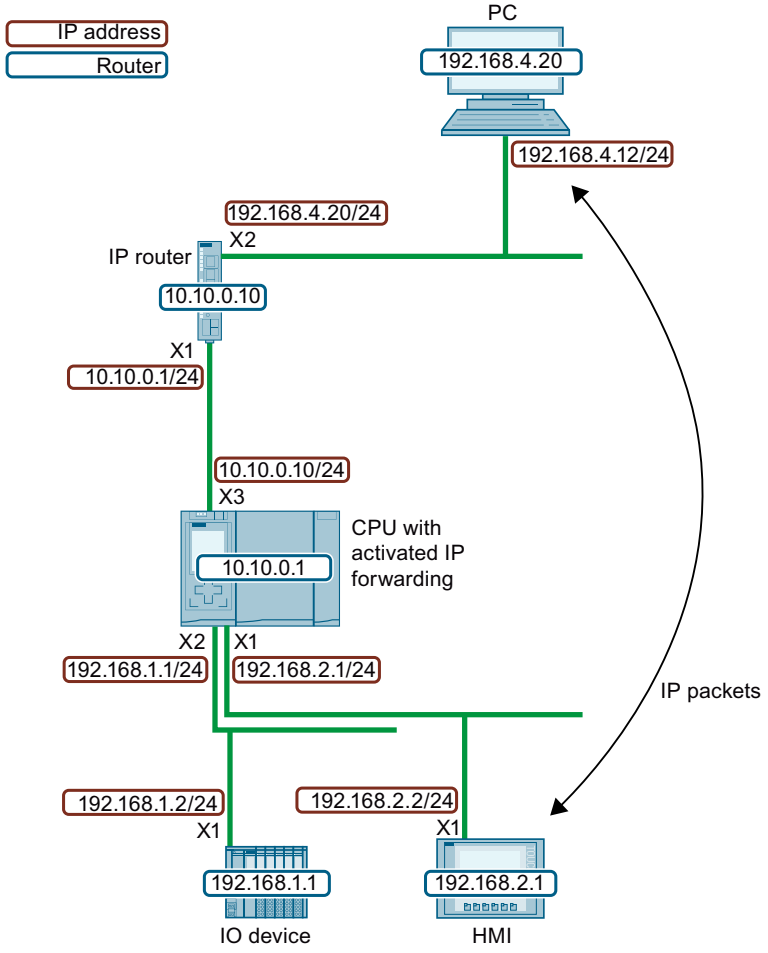


Figure 12-8 Sample configuration

This example configuration results in the following IP routing table for the CPU.

Table 12-1 IP route table of the CPU

Network destination	Interface	Gateway
0.0.0.0/0	10.10.0.10	10.10.0.1
192.168.1.0/24	192.168.1.1	-
192.168.2.0/24	192.168.2.1	-
10.10.0.0/24	10.10.0.10	-

For IP communication between the PG/PC and the HMI device, you need to set up additional IP routes to the IP subnet of the HMI device both in the PC and in the IP router. In the HMI device, you configure the IP address of the CPU interface X1 as the standard gateway.
In a Windows computer, for example, you set up an additional IP route from the command prompt using the command "route add <destination IP subnet> mask <subnet mask> <gateway>". However, you need certain access rights for this. For this example, enter the following prompt:

- "route add 192.168.2.0 mask 255.255.255.0 192.168.4.20"

In an IP router, you set up additional routes, e.g. via a web interface. Set up the following route for this example:

- Destination IP subnet: 192.168.2.0
- Subnet mask: 255.255.255.0
- Gateway: 10.10.0.10

Restrictions

You cannot configure any additional IP routes other than the router ("Standard Gateway") for an S7-1500 CPU. The network destination is either a connected IP subnet, or the network destination can be reached via exactly one configurable router. Because the S7-1500 CPU does not support additional IP routes, you cannot build bi-directional IP router cascades.

In the following configuration, you can configure either "Router 1" or "Router 2" in the CPU. "Router 1" is configured as an example. In this case, you cannot configure "Router 2". IP communication between the PC and the HMI device is not possible because the route is not continuous in both directions.

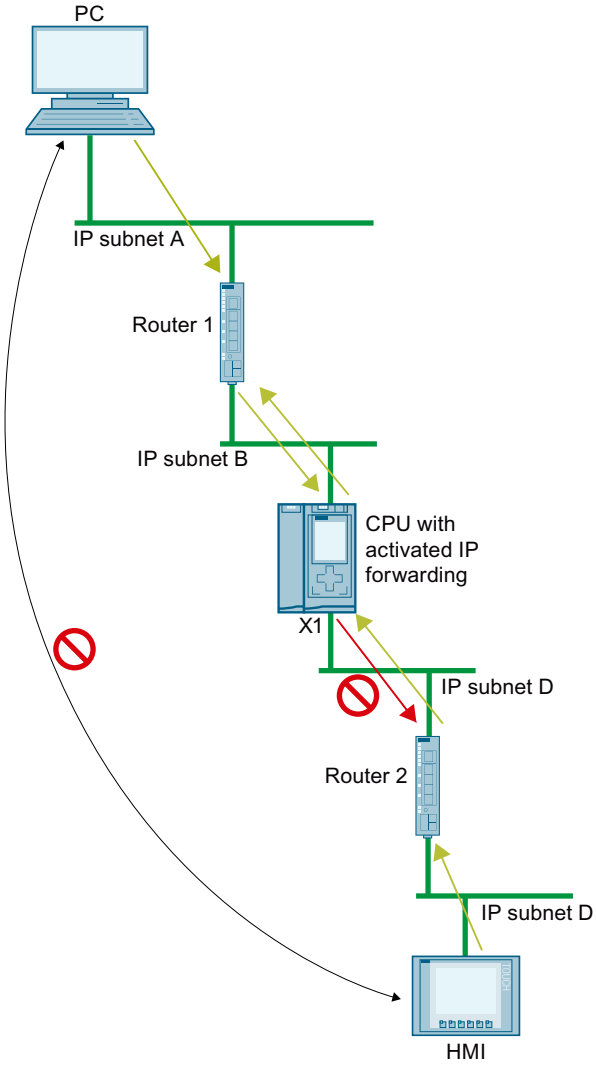


Figure 12-9 Unsupported IP router cascade

IP forwarding via the interface of a CP

IP forwarding also works via the interface of a CP. For this you have to activate the "Access to PLC via communication module" function for this CP in the CPU.

How you enable the "Access to PLC via communication module" function is described in the online help of STEP 7.

Reaching C/C++ Runtime of the CPU 1518 4 PN/DP MFP via interfaces X1 or X2

If you activate PN/DP MFP IP forwarding for the CPU 1518 4 PN/DP, you will not only reach devices in the IP subnet of interface X3 via interfaces X1 and X2, but also C/C++ Runtime. From the C/C++ Runtime of the CPU 1518 4 PN/DP MFP, you reach all devices in the IP subnets of the interfaces X1, X2 and X3.

Conditions:

- IP forwarding is enabled for the CPU 1518 4 PN/DP MFP.
- The IP address of C/C++ Runtime and the IP address of interface X3 are located in the same IP subnet.
- The routes to the IP subnets at X1 and X2 are entered in C/C++ Runtime.
  In C/C++ Runtime, enter a route with the following command: "Route add-net <destination IP subnet> mask <subnet mask> gw <gateway>

The following figure shows a configuration in which a PC accesses the C/C++ Runtime of CPU 1518-4 PN/DP MFP via interface X2.

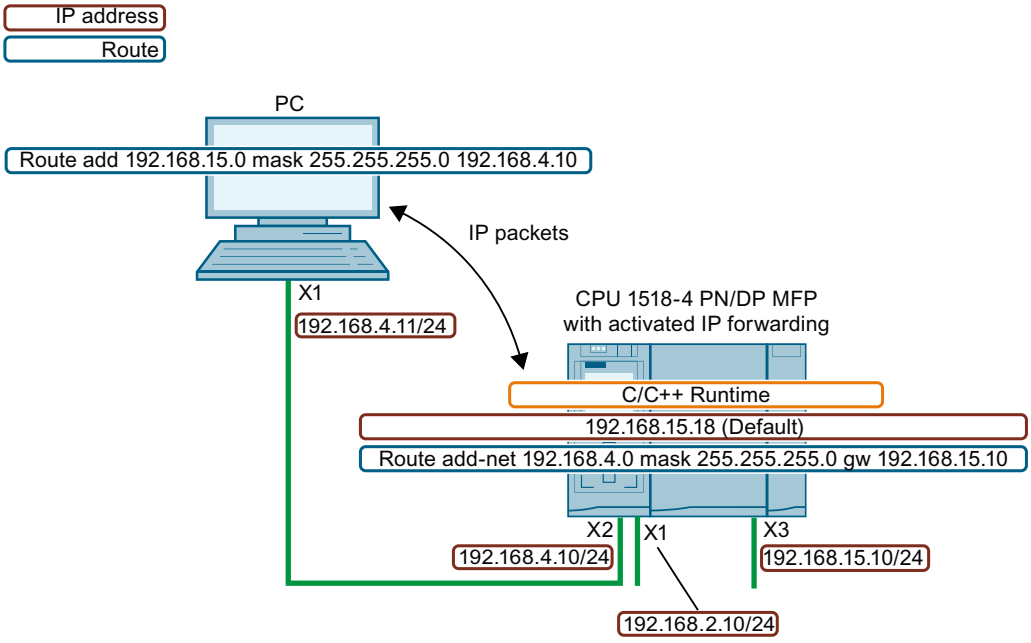


Figure 12-10 Access to C/C++ Runtime via interface X2

Take network security into account for IP forwarding

If you activate IP forwarding for a CPU, you enable "external" access to devices that are actually only accessible and controlled by the CPU. These devices are therefore usually not protected against attacks.

The following figure shows how to protect your automation system against unauthorized access.

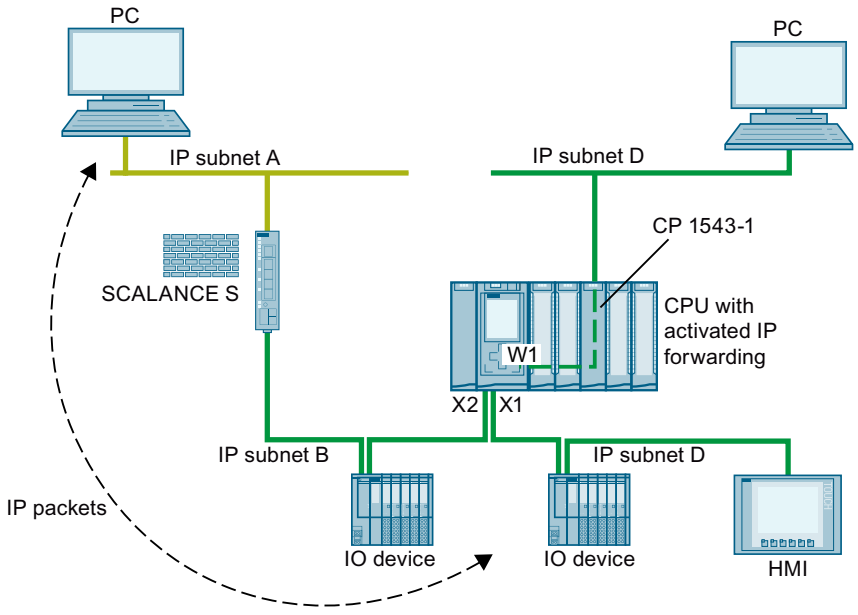


Figure 12-11 Network security for IP forwarding

- The CPU accesses all devices within the dark green IP subnets B and C close to the CPU via the interfaces X1 and X2.
- A SCALANCE S router is configured in the CPU. The CPU accesses the devices in the remote, light green IP subnet A via the router.
- The "Access to PLC via communication module" function is enabled for the CP 1543 in the CPU. The CPU reaches all devices within the IP subnet D via W1 interface.

If IP forwarding is activated in the CPU, a device from IP subnet A can access any device within IP subnets B, C and D close to the CPU.

Protect your automation system and connected devices against unauthorized access from outside.

Separate the CPU-related IP subnets from the remote IP subnets with a firewall. For example, use the SCALANCE S security modules with integrated firewall.
This application example (https://support.industry.siemens.com/cs/ww/en/view/22376747) describes how to protect an automation cell with a firewall using the SCALANCE S602 V3 and SCALANCE S623 security modules.

Enabling/disablng IP forwarding

To enable IP forwarding, proceed as follows:

1. Select the CPU in the network view of STEP 7 (TIA Portal).
2. In the properties of the CPU of the Inspector window, navigate to "General" > "Advanced Configuration" > "IP forwarding".
3. In the "Configuration IPv4 Forwarding" area, select the check box "Activate IPv4 for interfaces of this PLC".

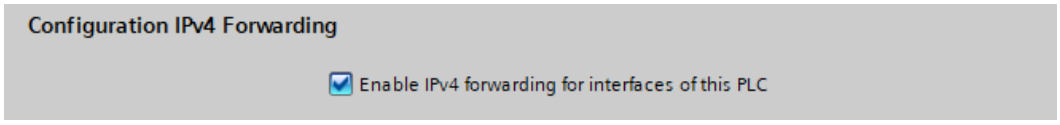


Figure 12-12 Enabling IP forwarding

Result: IP forwarding is enabled for all interfaces of the S7-1500 CPU.

You disable IP forwarding by clearing the check box "Enable IPv4 forwarding for interfaces of this PLC".

12.4 Data record routing

Definition of data record routing

Data can be sent over PROFINET from an engineering station to field devices via multiple networks. Since the engineering station addresses the field devices using standardized records and these records are routed via S7 devices, the term "data record routing" is used to refer to this type of routing.

The data sent using data record routing include the parameter assignments for the participating field devices (slaves) and device-specific information (e.g. setpoint values, limit values).

Data record routing is used, for example, when field devices of different manufacturers are used. The field devices are addressed using standardized data records (PROFINET) for configuration and diagnostics.

Data record routing with STEP 7

You can perform data routing with STEP 7 by calling a device tool (for example, PCT) via the TCI interface (Tool Calling Interface) and passing call parameters. The device tool uses the communication paths that STEP 7 also uses for communication with the field device.

No configuration is required for this type of routing except the installation of the TCI tools on the STEP 7 computer.

Example: Data record routing with the Port Configuration Tool (PCT)

You can use the Port Configuration Tool (PCT) to configure the IO link master of the ET200 and assign parameters to connected IO link devices. The subnets are connected via data record routers. Data record routers are, for example, CPUs, CPs, IMs, IO link master.

You can learn about the constellations of data record routers supported by the PCT in this FAQ (https://support.industry.siemens.com/cs/us/en/view/87611392).

The figure below shows an example configuration with the data record routing with PCT.

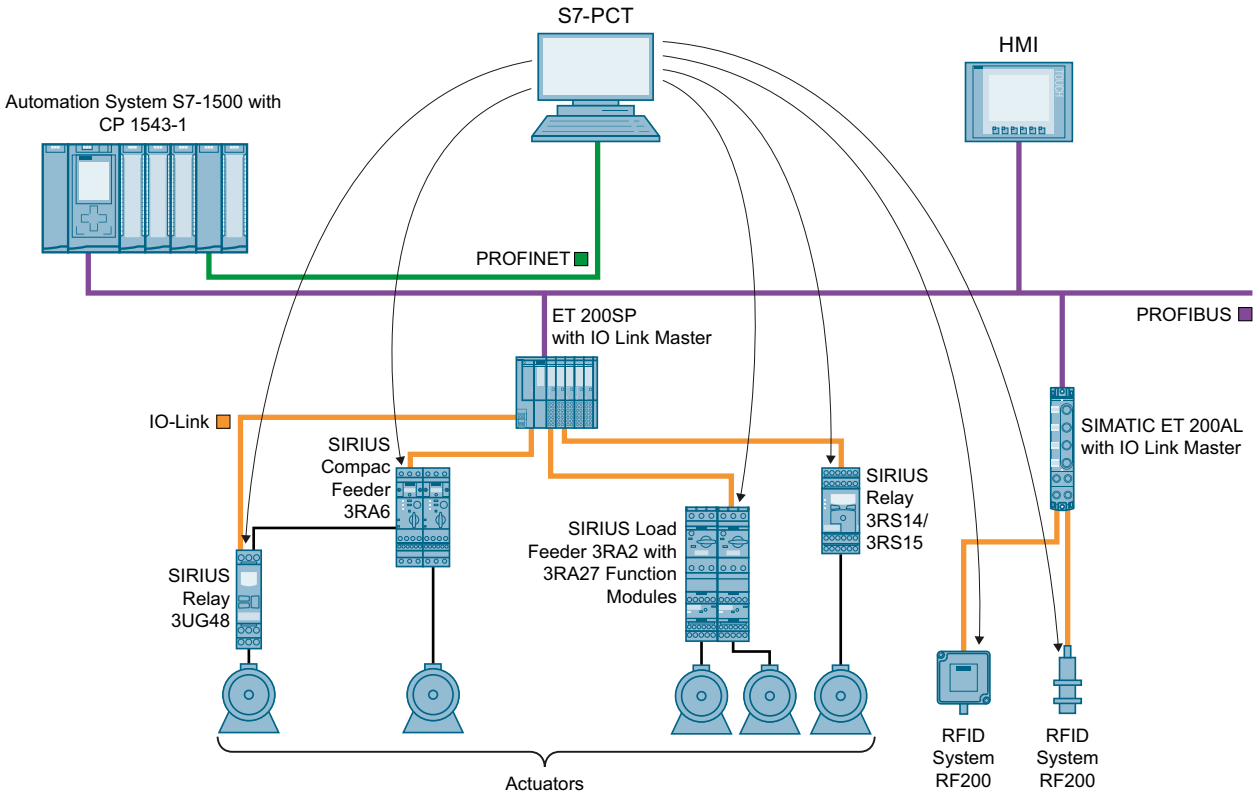


Figure 12-13 Example configuration for data record routing with PCT

Additional information

- The differences that exist between "normal" routing and data record routing are described in this FAQ (https://support.industry.siemens.com/cs/ww/en/view/7000978).
- Whether or not the CPU, CP or CM you are using supports data record routing can be found in the relevant manuals.
- The allocation of connection resources with data record routing is described in the section Allocation of connection resources (Page 372).
- You can find additional information on configuration with STEP 7 in the STEP 7 online help.

12.5 Virtual interface for IP-based applications

As of firmware version 2.8, the S7-1500 CPU offers the option of reaching its IP-based application, such as OPC UA, not only via its local (PN) interfaces, but also via the interfaces of communications processors in the same station. The supported communications processors can be found in the requirements. A communication partner reaches these IP-based applications via a virtual interface that can be configured in the TIA Portal as of version V16. The virtual interface is called W1 (according to IEC 81346-2).

Features of the virtual interface

The virtual interface is not a fully diagnosable interface with the familiar properties of conventional interfaces. The virtual interface is not displayed in the graphical views, because the internal connection via the backplane bus does not represent an S7 subnet and does not have any ports. A physical connection by means of a network cable therefore cannot be established.

The IP address of the virtual interface is displayed (in the TIA Portal, in the CPU display) and can be configured.

The following communication options can be used via the virtual interface W1:

- OPC UA (client and server)
- Programmed OUC connections
- Programming device/HMI communication
- Partner accesses from S7 CPUs via PUT/GET instructions

The activated interface can be used in dialogs where IP-based connections are configured.

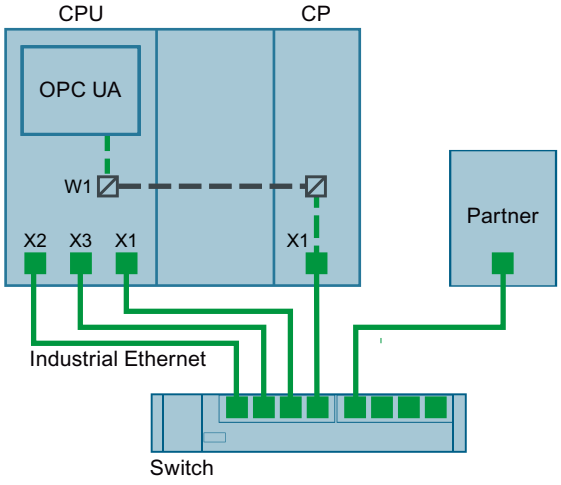


Figure 12-14 Principle of the virtual interface

Compared to conventional interfaces, the virtual interface has the following restrictions:

- No access to the web server over the virtual interface.
- Online backup is not possible via a connected programming device with the TIA Portal.
- If the CPU and communication partners are connected via the virtual interface, they cannot exchange data via LLDP (Link Layer Discovery Protocol).
- The S7 routing service does not use the virtual interface W1.

Requirement

For a CPU service to be accessible via the Ethernet interface of a CP, the following requirements must be fulfilled:

- S7-1500 CPU firmware V2.8 or higher
- CP 1543-1 firmware V2.2 or higher

Recommendation: Use a CP 1543-1 with firmware V3.0 or higher. As of this version, the security functions (firewall) are also available for the virtual interface and no additional firewall needs to be installed between the station and an insecure network.

R/H CPUs do not support this function because R/H CPUs do not support CPs.

Configuration of the virtual interface W1

In the properties of an S7-1500 CPU as of firmware V2.8, you can assign a plugged communication module to the virtual interface W1 under "Advanced Configuration > Access to PLC via communications module". You can then use this for external access to the CPU. If no CPs are plugged in or the plugged CPs do not support access to the CPU, the selection is empty.

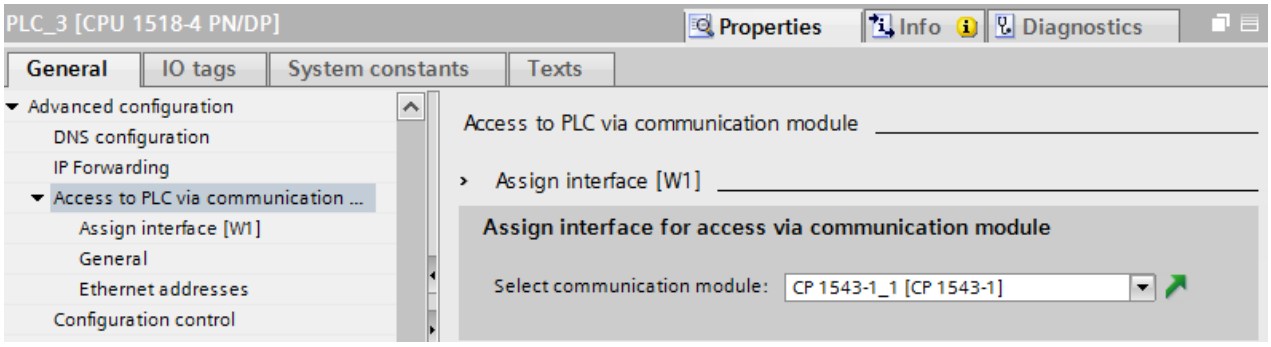


Figure 12-15 Selecting the CP in the CPU properties

After selecting the CP, the specifications and parameters for the virtual interface are displayed. Here, you can edit the settings for the IP protocol and the PROFINET parameters.

- The IP subnet is freely selectable, just like with the CP. The IP subnet is entered via the subnet mask and IP address of the virtual interface.
- When entering the IP subnet for the virtual interface, note that you are not using the same IP subnet as for the local interfaces of the CPU.

Once the IP address is entered, it is shown in the properties dialog of the OPC UA server in the list of server addresses. These settings provide the CPU with the new virtual interface W1, via which the CPU services described above, such as the OPC UA server, can be accessed via a communications module. The corresponding connections and S7 communication (e.g. HMI and BSEND, BRCV) are made via this interface. The OPC UA server does not allow selection of a specific interface (selection via an IP address), either all or none are possible.

NOTE

The IP address of the virtual interface is not listed as W1 in the device display under the currently displayed local interfaces (Xn) but is available under "Addresses" in the "Settings" section. The virtual interface is also visible when no CP is plugged or when the virtual interface is not activated. If no IP suite is available, the IP address and the subnet mask are 0.0.0.0.

If you change the configured and loaded IP address parameters of the virtual interface via display, T_CONFIG instruction or online, the loaded configuration is active again after the CPU restarts.

Configuration changes on the CP

A change of the assigned communication module may have an effect on the configuration of the virtual interface:

- In the properties of the CPU:
  - Assignment of a different CP: The configuration is used for the new CP.
  - Deselect the assigned CP: The virtual interface W1 is deactivated and the configuration is lost.
    When a CP is assigned again, the configuration must be performed again.
- On the device:
  - Moving the CP: If the CP is only moved to another slot of the device, the configuration continues to be valid.
  - Removing the CP: If the CP is deleted or moved to another device, the configuration is retained. In the drop-down list of the CPU, the CP is displayed as missing and compiling the configuration indicates an error. The missing CP can be deselected or assigned to another CP.

Display in the diagnostics and the system constants

The virtual interface W1 is displayed in the diagnostics view under "Online & Diagnostics". The hardware ID of the virtual interface is displayed in the system constants of the CPU properties.

Settings in the communications module (CP 1543-1 as of firmware V3.0)

As of firmware V3.0, you can use the CP's internal firewall to secure data traffic via the virtual interface. To activate the firewall in the communications module, proceed as follows in the protected project:

1. Select the communications module in the STEP 7 work area.
2. In the Inspector window, go to "Properties > Security".
3. Activate the "Enable security functions" option.
   The configurable security functions now appear in the Inspector window.
4. Activate the options "Enable firewall".
   In the Inspector window you can now allow the use of the virtual interface W1.

You can only secure data traffic via the virtual interface W1 of the CPU for the OPC UA service.

NOTE

Checking the manual configuration

If the firewall is enabled, you need to manually check whether the desired services are allowed by the firewall. Only activate those services for the IP and MAC filters that you also want to access over the CP interface. See the notes on security settings and firewall rules of the S7-1500 CPs in the info system of the TIA Portal.

Settings in the communications module (CP 1543-1, firmware V2.2, < V3.0)

The security functions of the CP 1543-1 with a firmware version lower than V3.0 cannot secure data traffic via the virtual interface. Although you can activate the security functions in the TIA Portal, such a configuration cannot be compiled.

NOTICE
Connecting to non-secure networks If you connect the CP to a non-secure network, it is absolutely necessary to connect an additional firewall between the CP and the non-secure network. For example, use the security modules SCALANCE S602 V3 and SCALANCE S623 with integrated firewall.

Connection resources 13

13.1 Connection resources of a station

Introduction

Some communications services require connections. Connections occupy resources in the automation system (station). The connection resources are made available to the station by the CPUs, communications processors (CPs) and communications modules (CMs).

Connection resources of a station

The connection resources available depend on the CPUs, CPs and CMs being used and must not exceed a maximum number per station.

The maximum number of resources of a station is determined by the CPU.

Reserved connection resources

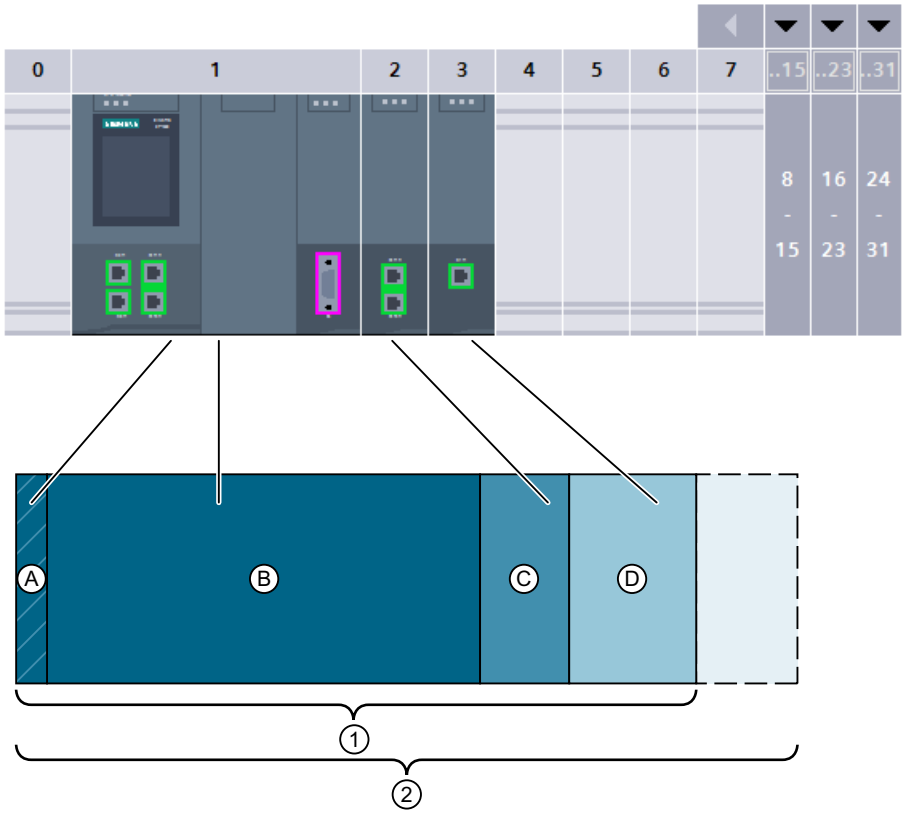
Each CPU has reserved connection resources for PG, HMI and Web server communication. This ensures, for example, that a PG can always establish at least one online connection with the CPU regardless of how many other communications services are already using connection resources.

Dynamic connection resources

In addition, dynamic resources exist. The difference between the maximum number of connection resources and the number of reserved connection resources is the maximum number of dynamic connection resources.

The communication services PG Communication, HMI Communication, S7 Communication, Open User Communication, Web Communication, OPC UA Client/Server Communication and other communication from the pool of dynamic connection resources are used.

The figure below shows an example of how individual components make connection resources available to an S7-1500 station.



① Available connection resources of the station, of which

- A: Reserved connection resources of the station
- A + B: Connection resources of CPU 1518
- C: Connection resources of communications module CM 1542-1
- D: Connection resources of communications processor CP 1543-1

② Maximum communications resources of the station using the example of a configuration from CPU 1518, CM 1542-1 and CP 1543-1

Figure 13-1 Connection resources of a station

Number of connection resources of a station

Table 13-1 Maximum number of connection resources supported for some CPU types

Connection resources of a station	1511 1511C	1512C 1513	1515	1516	1517	1518
Maximum connection resources of the station	96	128	192	256	320	384
of which reserved	10					
of which dynamic	86	118	182	246	310	374
Connection resources of the CPU	64	88	108	128	288	320

Connection resources of a station	1511 1511C	1512C 1513	1515	1516	1517	1518
Max. additionally usable connection resources by plugging in CMs/CPs	32	40	84	128	32	64
Additional connection resources CM 1542-1	64					
Additional connection resources CP 1543-1	118					
Additional connection resources CM 1542-5	40					
Additional connection resources CP 1542-5	16					

The number of connection resources that a CPU or a communication module supports is specified in the device manuals in the Technical Specifications.

Example

You have configured a CPU 1516-3 PN/DP with a CM 1542-1 communication module and a CP 1542-5 communication processor.

- Maximum connection resources of the station: **256**
- Available connection resources:
  - CPU 1516-3 PN/DP: 128
  - CM 1542-1: 64
  - CP 1542-5: 16
  - Total: **208**

The setup provides 208 connection resources. By adding further communication modules, the station can support a maximum of 48 additional connection resources.

Reserved connection resources

10 connection resources are reserved for stations with S7-1500 CPU, ET 200SP CPU and ET 200pro CPU based on S7-1500:

- 4 for PG communication required by STEP 7, for example, for test and diagnostics functions or downloading to the CPU
- 4 for HMI communication which are occupied by the first HMI connections configured in STEP 7
- 2 for communication with the Web server

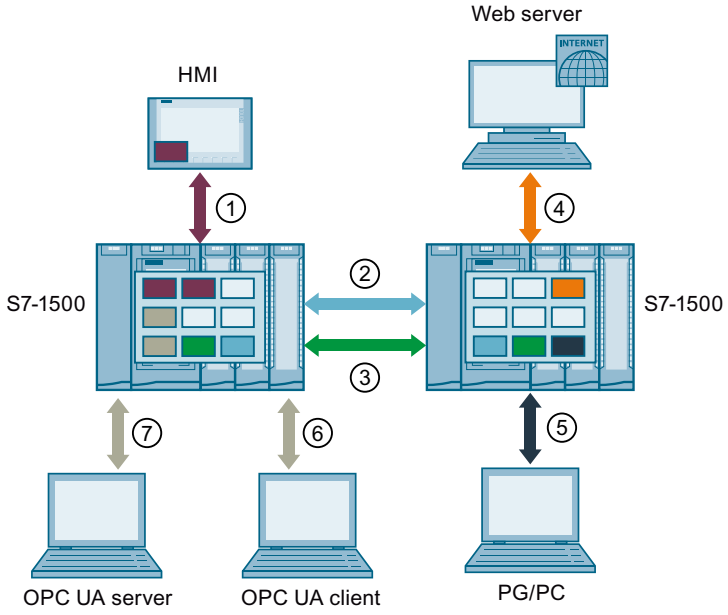
More information

Information on the connection resources of the S7-1500R/H redundant system is available in the section Connection resources of the redundant system S7-1500R/H (Page 391).

13.2 Allocation of connection resources

Overview - occupation of connection resources

The following figure shows how different connections occupy the resources of the S7-1500.



① HMI communication: See below.

② Open User Communication: Connections of Open User Communication occupy a connection resource in every end point.

③ S7 communication: Connections of S7 communication occupy a connection resource in every end point.

④ Web communication: The Web server connection occupies at least one connection resource in the station. The number of occupied connections depends on the browser.

⑤ PG communication: The PG connection occupies one connection resource in the station.

⑥ OPC UA client/server communication: Connection resource allocation for the server, see below

⑦ OPC UA client/server communication: Connection resource allocation for the client, see below

Connection resource for HMI communication

Connection resource for Open User Communication

Connection resource for S7 communication

Connection resource for Web communication

Connection resource for PG communication

Connection resources for OPC UA server communication

Figure 13-2 Allocation of connection resources

Connection resources for HMI communication

With HMI communication, the occupation of connection resources in the station depends on the HMI device being used.

Table 13-2 Maximum occupied connection resources for different HMI devices

HMI device	Maximum occupied connection resources of the station per HMI connection
Basic Panel	1
Comfort Panel	2[1]
RT Advanced	2[1]
RT Professional	3

[1] If you do not use system diagnostics or alarm configuration, the station occupies only one connection resource per HMI connection.

Example: You have configured the following HMI connections for a CPU 1516-3 PN/DP:

- Two HMI connections to an HMI TP700 Comfort. (2 connection resources each)
- One HMI connection to an HMI KTP1000 Basic. (1 connection resource)

In total 5 connection resources are occupied for HMI communication in the CPU.

Connection resources for OPC UA client communication

Each connection that the OPC UA client of the CPU has established to an OPC UA server occupies a connection resource in the station.

When establishing and closing an OPC UA connection, the OPC UA client temporarily occupies an additional connection resource. According to RFC 793, this connection resource is released again after a wait of approx. 60 seconds.

NOTE

Lack of resources due to temporary connection resources

A lack of connection resources occurs in the following situation:

- The OPC UA client of the CPU establishes or closes several connections simultaneously.
- The number of available connection resources of the station is insufficient for permanent and temporary connection resources of the OPC UA client communication.

Ensure that there are always enough available connection resources in the station to establish and end OPC UA connections.

Measures:

- Plan enough reserve for the OPC UA client connections.
- If necessary, establish or close the OPC UA connections one after the other.

Connection resources for routing

To transfer data beyond S7 subnets ("S7 routing"), an S7 connection is established between two CPUs. The S7 subnets are connected via gateways known as S7 routers. CPUs, CMs and CPs in S7-1500 are S7 routers.

The following applies for a routed S7 connection:

- A routed connection occupies one connection resource each in both end points. STEP 7 shows these connection resources in the "Connection resources" table.
- On the S7 router, two special connection resources are occupied for S7 routing. STEP 7 does not show the special connection resources for S7 routing in the "Connection resources" table. The number of resources for S7 routing depends on the CPU. You will find the resources for S7 routing in the technical specifications of the CPU in "Number of S7 routing connections".

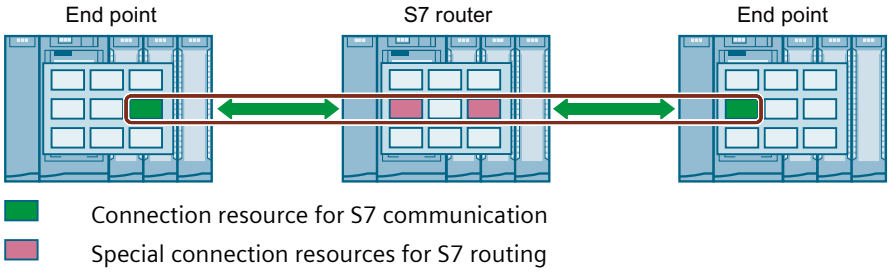


Figure 13-3 Connection resources with S7 routing

Data record routing also enables transfer of data beyond S7 subnets from an engineering station connected to PROFINET to various field devices via PROFIBUS.
With data record routing, as with S7 routing, two of the special connection resources for S7 routing are also occupied on every data record router.

NOTE

Connection resources with data record routing

With data record routing, on the data record router, two special connection resources for S7 routing are occupied. Neither the data record connection nor the allocated connection resources are displayed in the table of connection resources.

When are connection resources occupied?

The time for the occupation of connection resources depends on how the connection is set up (see section Setting up a connection (Page 37)).

- **Programmed setup of a connection**:
  As soon as an instruction to establish a connection is called in the user program (TSEND_C/TRCV_C or TCON), a connection resource is occupied.
  With suitable parameter assignment of the CONT parameter of the TSEND_C/TRCV_C instructions or by calling the TDISCON instruction, the connection can be terminated following data transfer and the connection resource is available again. When the connection is terminated, the connection resources on the CPU/CP/CM are available again.
- **Configured connections** (e.g. S7 connection):
  If you have configured a connection in STEP 7, the connection resource is occupied as soon as the hardware configuration is downloaded to the CPU.
  After using a configured connection for data transfer, the connection is not terminated. The connection resource is permanently occupied. To release the connection resource again, you need to delete the configured connection in STEP 7 and download the modified configuration to the CPU.

- **PG connection**:
  As soon as you have connected the PG to a CPU online in STEP 7, connection resources are occupied.
- **Web server**:
  As long as you have opened the Web server of the CPU in a browser, connection resources are occupied in the CPU.
- **OPC UA server**
  Each connection to the OPC UA server of the CPU occupies a connection resource in the station. This connection resource is released immediately when the connection is terminated.
- **OPC UA client**
  Each connection that the OPC UA client of the CPU has established to an OPC UA server occupies a connection resource in the station. When an OPC UA connection is established, the OPC UA client temporarily occupies an additional connection resource. When an OPC UA connection is terminated, the connection resource is not released again until after a wait time of approx. 60 seconds in accordance with RFC 793.

Monitoring the maximum possible number of connection resources

Offline

During configuration of connections, STEP 7 monitors the occupation of the connection resources. If the maximum possible number of connection resources is exceeded, STEP 7 signals this with a suitable warning.

Online

The CPU monitors the use of connection resources in the automation system. If you establish more connections in the user program than those provided by the automation system, the CPU acknowledges the instruction to establish the connection with an error.

S7-1500 and S7-300 comparison

You will find a comparison of how the communication resources of the S7-1500 and S7-300 are managed in this FAQ (https://support.industry.siemens.com/cs/ww/en/view/109747092).

13.3 Display of the connection resources

Display of the connection resources in STEP 7 (offline view)

You can display the connection resources of an automation system in the hardware configuration. You will find the connection resources in the Inspector window in the properties of the CPU.

Connection resources

	① Station resources			② Module resour...	Module resour...	Module resour...
	Reserved		Dynamic ⚠	PLC_1 [CPU 15...	CM 1542-1_1 [..	CP 1543-1_1 [...
Maximum number of resources:		10	246	128	64	118
	Maximum	Configured	Configured	Configured	Configured	Configured
PG communication:	4	-	-	-	-	-
HMI communication:	4	4	6	6	0	4
S7 communication:	0	-	7	4	3	0
Open user communication:	0	-	13	8	5	0
Web communication:	2	-	-	-	-	-
OPC UA client/server communicatio...	0	-	-	-	-	-
Other communication:	-	-	0	0	0	0
Total resources used:		4	26	18	8	4
Available resources:		6	220	110	56	114

Figure 13-4 Example: Reserved and available connection resources (offline view)

① Station-specific connection resources

The columns of the station-specific connection resources provide information about the used and available connection resources of the station.

In the example, a maximum of 256 station-specific connection resources are available for the automation system.

- 10 reserved connection resources, of which 4 are already in use and a further 6 available.
  The used resources are divided up as follows:
  - 4 Resources for HMI communication
- 246 dynamic connection resources, of which 26 are already in use and a further 220 available.
  The used resources are divided up as follows:
  - 6 resources for HMI communication
  - 7 resources for S7 communication
  - 13 resources for Open User Communication

The warning triangle in the column of the dynamic station resources is displayed because the sum of the maximum available connection resources of CPU, CP and CM (= 310 connection resources) exceeds the station limit of 256.

NOTE

Available connection resources exceeded

STEP 7 signals the exceeding of the station-specific connection resources with a warning. To make full use of the connection resources from the CPU, CP and CM, either use a CPU with a higher maximum number of available station-specific connection resources or reduce the number of communications connections.

② Module-specific connection resources

The columns of the module-specific connection resources provide information about the use of resources on the CPUs, CPs and CMs of an automation system:

The display is per module and not per interface.

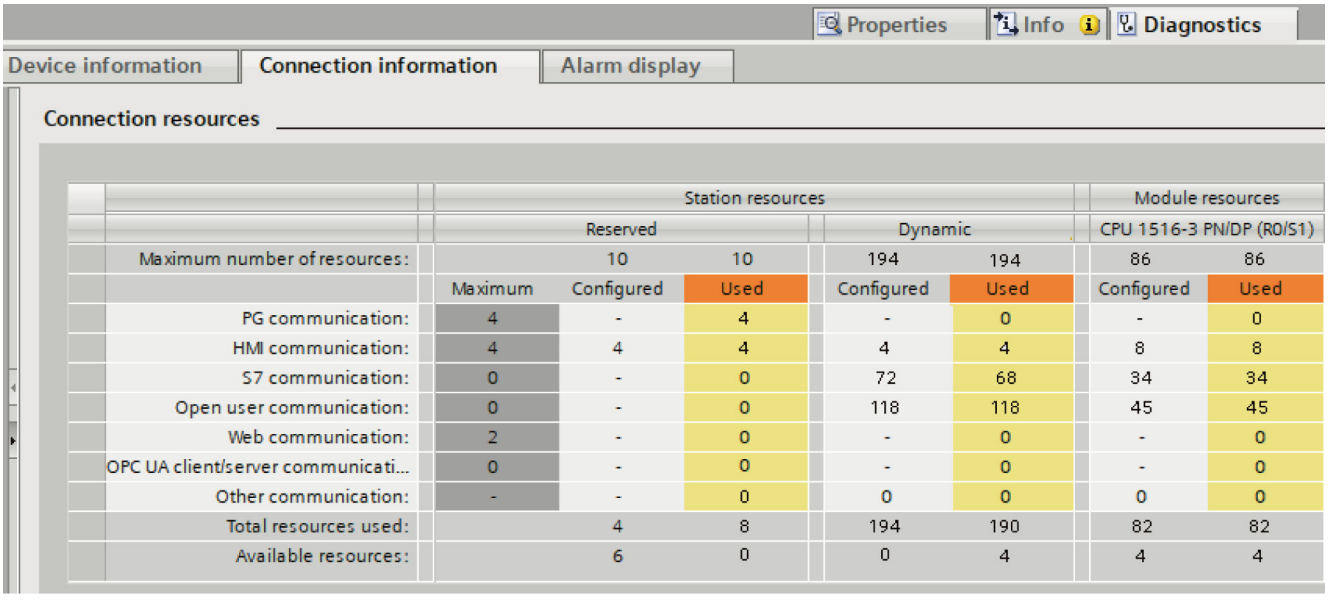
In the example, the CPU makes a maximum of 128 connection resources available, of which 18 are already in use and 110 still available.

The used resources are divided up as follows:

- 6 resources for HMI communication
- 4 resources for S7 communication
- 8 resources for Open User Communication

Display of the connection resources in STEP 7 (online view)

If you are connected to the CPU online, you can also see how many resources are currently being used under "Connection information".



	Station resources					Module resources	
	Reserved			Dynamic		CPU 1516-3 PN/DP (R0/S1)	
Maximum number of resources:		10	10	194	194	86	86
	Maximum	Configured	Used	Configured	Used	Configured	Used
PG communication:	4	-	4	-	0	-	0
HMI communication:	4	4	4	4	4	8	8
S7 communication:	0	-	0	72	68	34	34
Open user communication:	0	-	0	118	118	45	45
Web communication:	2	-	0	-	0	-	0
OPC UA client/server communicati...	0	-	0	-	0	-	0
Other communication:	-	-	0	0	0	0	0
Total resources used:		4	8	194	190	82	82
Available resources:		6	0	0	4	4	4

Figure 13-5 Connection resources - online

The online view of the "Connection resources" table in addition to the offline view also contains columns with the connection resources currently being used. Thus, the online view

displays **all** used connection resources in the automation system, regardless of how the connection was set up.
The "Other communication" row displays connection resources assigned for communication with external devices. The table is updated automatically.

NOTE

If a routed S7 connection goes through a CPU, the required connection resources of the CPU do not appear in the table of connection resources.

Display of the connection resources for HMI

For information regarding the availability and assignment of connection resources for HMI connections, refer to the "Connection resources" properties in the Inspector window of the offline view (in the context of the HMI device).

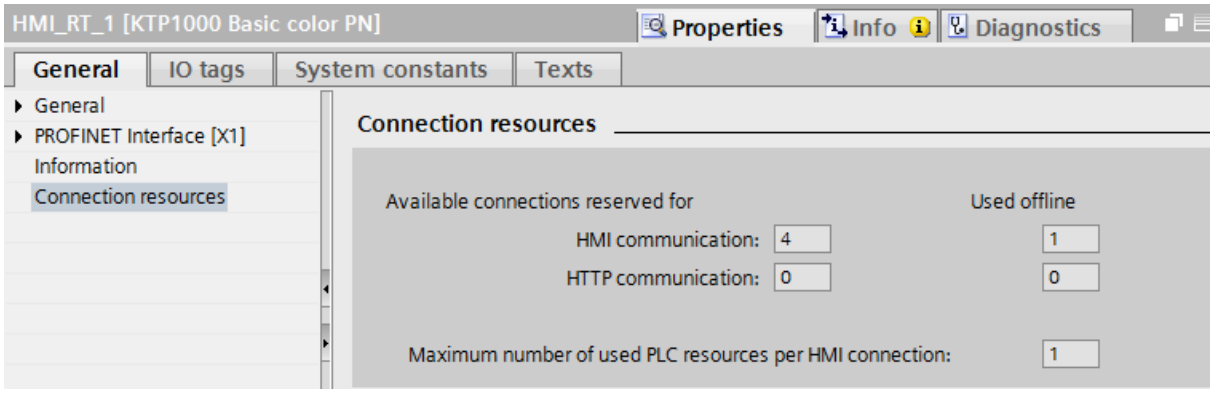


Figure 13-6 Connection resources - HMI communication

The following is displayed in the connection resources area:

- Number of available connections on the HMI reserved for HMI communication and HTTP communication
- Number of connection resources for HMI communication and HTTP communications used offline in the HMI
  If the maximum number of available connection resources for an HMI device is exceeded, a corresponding message is output by STEP 7.
- "Maximum number of used PLC resources per HMI connection". This parameter is a factor that is to be multiplied by the number of HMI connections used offline. The product is the number of HMI resources occupied on the CPU.

Displaying the connection resources in the Web server

You can display the connection resources not only in STEP 7, but also with a browser that displays the relevant page of the Web server.
You will find information on displaying connection resources in the Web server in the Web Server (https://support.industry.siemens.com/cs/us/en/view/59193560)function manual.

Diagnostics and fault correction 14

14.1 Connection diagnostics

Connections table in the online view

After selecting a CPU in the Devices & networks editor of STEP 7, you will see the status of your connections displayed in the online view of the connections table.

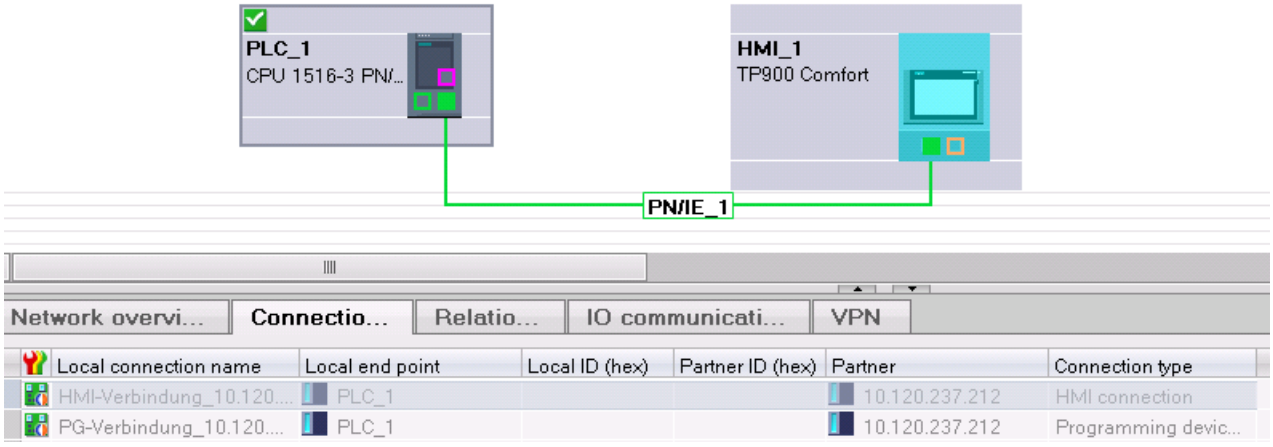


Figure 14-1 Online view of the connections table

After selecting the connection in the connections table, you obtain detailed diagnostic information in the "Connection information" tab.

"Connection information" tab: Connection details

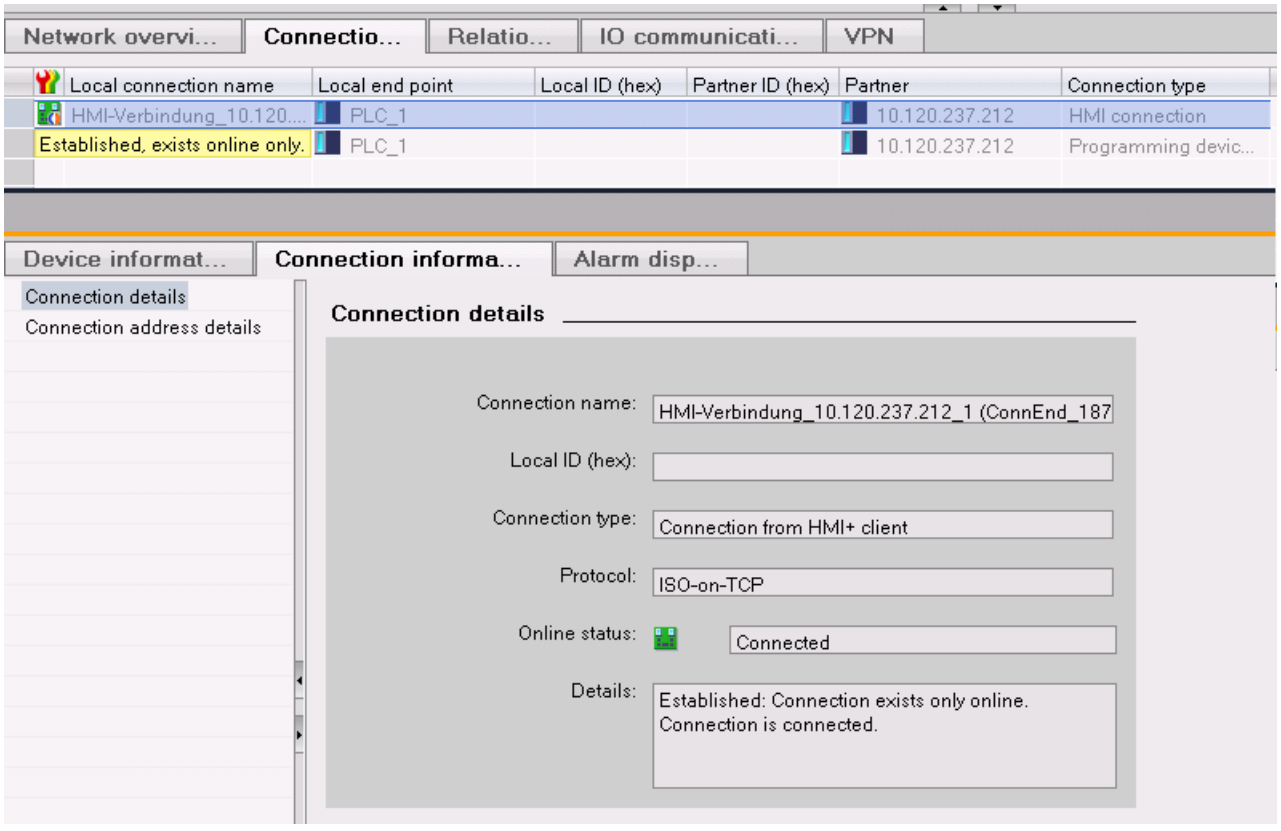


Figure 14-2 Diagnostics of connections - connection details

"Connection information" tab: Address details

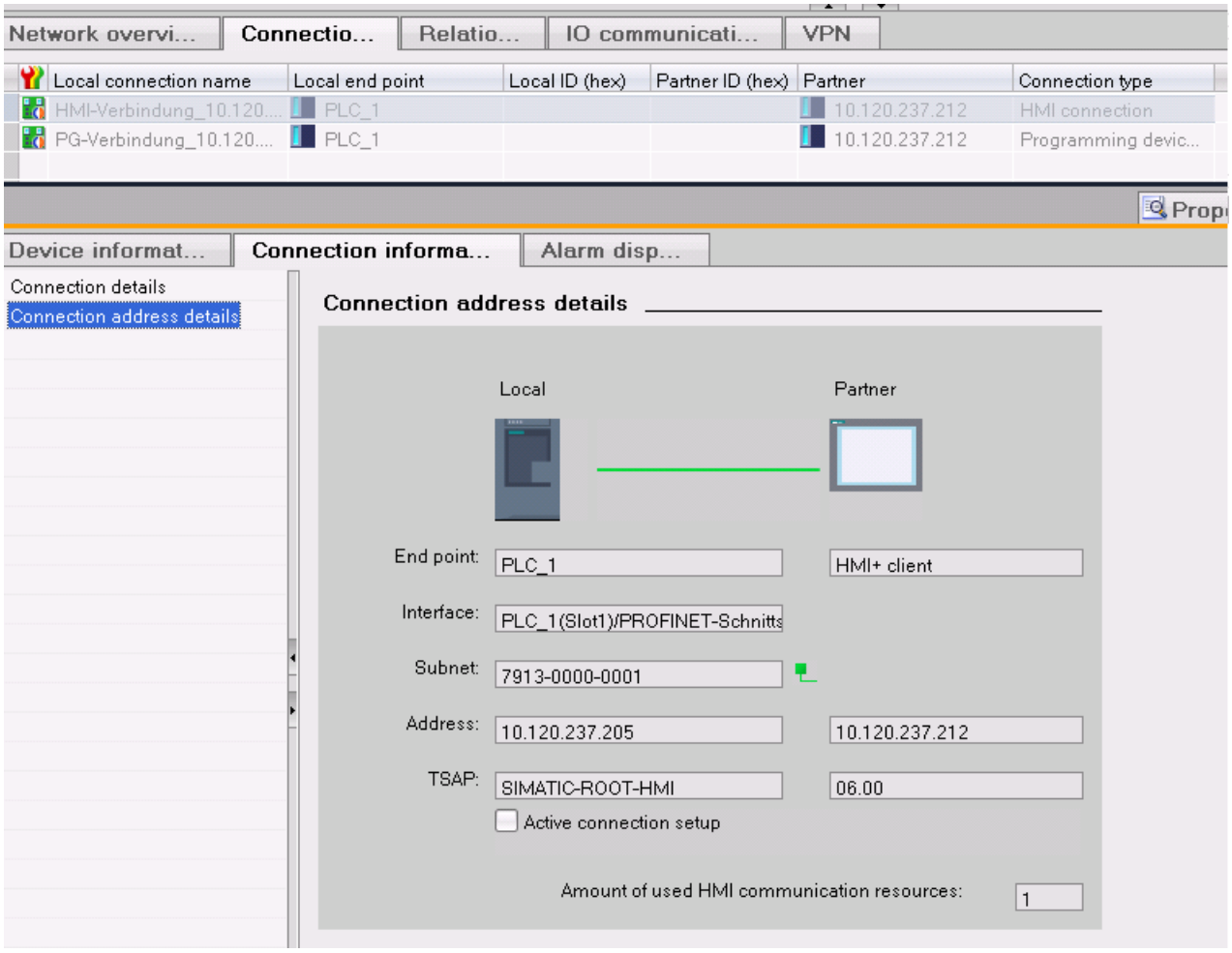


Figure 14-3 Diagnostics of connections - address details

Diagnostics via web server

You can evaluate diagnostic information from the CPU using a web browser via the integrated web server of a CPU.

On the "Communication" Web page, you will find the following information about communication via PROFINET in various tabs:

- Information on the PROFINET interfaces of the CPU (for example addresses, subnets, physical properties).
- Information on the quality of the data transfer (for example number of data packets sent/received error-free).
- Information about the allocation/availability of connection resources.
- The "Connection status" page is similar to the online view in STEP 7 and also provides an overview of all connections with detail view.

Diagnostics with the user program

When you program the T_DIAG instruction, you can evaluate diagnostic information about the configured and programmed connections of the CPU using the user program.

Additional information

You will find the description of the web server functionality in the function manual Web server (https://support.industry.siemens.com/cs/us/en/view/59193560).

14.2 Emergency address

If you cannot reach the CPU via the IP address, you can set a temporary emergency address (emergency IP) for the CPU. Via this emergency address, you can re-establish the connection with a CPU in order to load a device configuration with a valid IP address.

You can set an emergency address regardless of the protection level of the CPU

When do you need an emergency address?

Your CPU cannot be reached in the following cases:

- The IP address of your PROFINET interface is assigned twice.
- The subnet mask is set incorrectly.

Requirements

- You have selected "Set IP address in the project" for the IP protocol in the device configuration in STEP 7.
- The CPU is in STOP mode.

Restoring a valid device configuration with an emergency address

1. Set the emergency address for the interface of the CPU with a DCP tool. For example, the SIMATIC Automation Tool has a DCP command "Define IP address".
   The maintenance LED of the CPU lights up. The diagnostic buffer also shows that an emergency address was activated for an Ethernet interface.
2. Load a STEP 7 project with a valid IP address into the CPU.
3. Switch the CPU off and on again.
   The emergency address is reset.

Result

The CPU starts up with the valid IP address.

Communication with the redundant system S7-1500R/H

15

Introduction

Communication with the S7-1500R/H redundant system basically functions as with the S7-1500 standard system.

This chapter describes the special features and restrictions for communication with the S7-1500R/H redundant system.

Communication options for the S7-1500R/H redundant system

- Open User Communication via TCP/IP, UDP, ISO on-TCP and Modbus/TCP
- S7 communication as server
- HMI communication, PG communication
- Secure PG/HMI communication (see also Secure PG/HMI communication (Page 93))
- SNMP
- Time-of-day synchronization via NTP

Restrictions for communication with the S7-1500R/H redundant system

- Open User Communication:
  - no configured connections
  - No support for Secure Open User Communication
  - Email: The S7-1500R/H CPUs support only the versions < V5.0 of the "TMAIL_C" instruction. Versions from V5.0 are not supported.
  - No support of connection descriptions according to "TCON_Param"
- no OPC UA
- no S7 communication as client
- no web server
- PG communication: It is not possible to access two CPUs online at the same time. You can either access the primary CPU or the backup CPU.
- The CPUs of the S7-1500R/H do not support centrally plugged communication modules.

15.1 System IP addresses

The system IP address of the S7-1500R/H redundant system

In addition to the device IP addresses of the CPUs, the S7-1500R/H redundant system supports system IP addresses:

- System IP address for the PROFINET interfaces X1 of the two CPUs (system IP address X1)
- System IP address for the PROFINET interfaces X2 of the two CPUs (system IP address X2)
- System IP address for the PROFINET interfaces X3 of the two CPUs (system IP address X3)

You use the system IP addresses for communication with other devices (for example, HMI devices, CPUs, PCs). The devices always communicate via the system IP address with the primary CPU of the redundant system. This ensures, for example, that the communication partner can communicate with the new primary CPU (previously backup CPU) in the RUN-Solo system state after failure of the original primary CPU in redundant operation.

There is a virtual MAC address for each system IP address.

You enable the system IP addresses in STEP 7.

Advantages of the system IP addresses compared to device IP addresses

- The communication partner communicates specifically with the primary CPU.
- Communication of the S7-1500R/H redundant system via a system IP address still also works in the event of the failure of the primary CPU.

Applications

You use the system IP addresses for the following applications:

- HMI communication with the S7-1500R/H redundant system: You can use an HMI device to control or monitor the process on the redundant S7 1500R/H system.
- Open User Communication with the S7-1500R/H redundant system:
  - Another CPU or an application on a PC accesses data of the S7-1500R/H redundant system.
  - The S7-1500R/H redundant system accesses a different device.

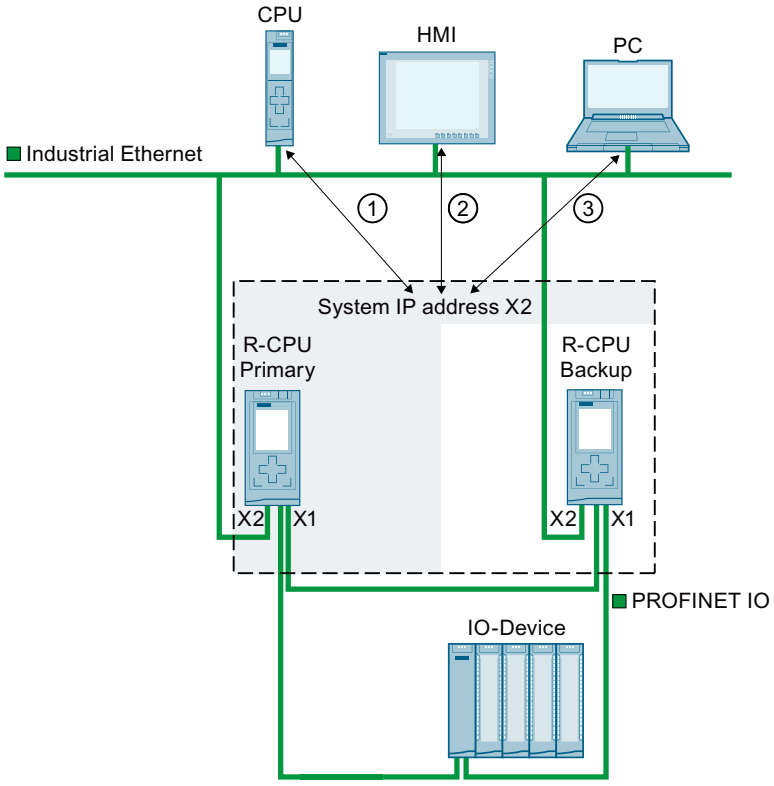
  TCP, UDP and ISO-on-TCP-connections are possible.
- IP forwarding: If you use the system IP addresses as the gateway/default route for IP routes through the S7-1500R/H redundant system, IP packets are forwarded even if one CPU fails.

Requirements

- The interface of the communication partner is connected to both CPUs, each via the same interface (e.g. X2).
- The system IP address for the interfaces of the S7-1500R/H system is enabled.

Communication via the system IP addresses X2 and X3

If the CPUs of the redundant S7-1500R/H system have two or three PROFINET interfaces, preferably use the PROFINET interface X2 or X3 for communication with other devices.
The following figure shows a configuration in which the communication partners are connected via the respective PROFINET interfaces X2 with the CPUs of the redundant system S7-1500R/H.

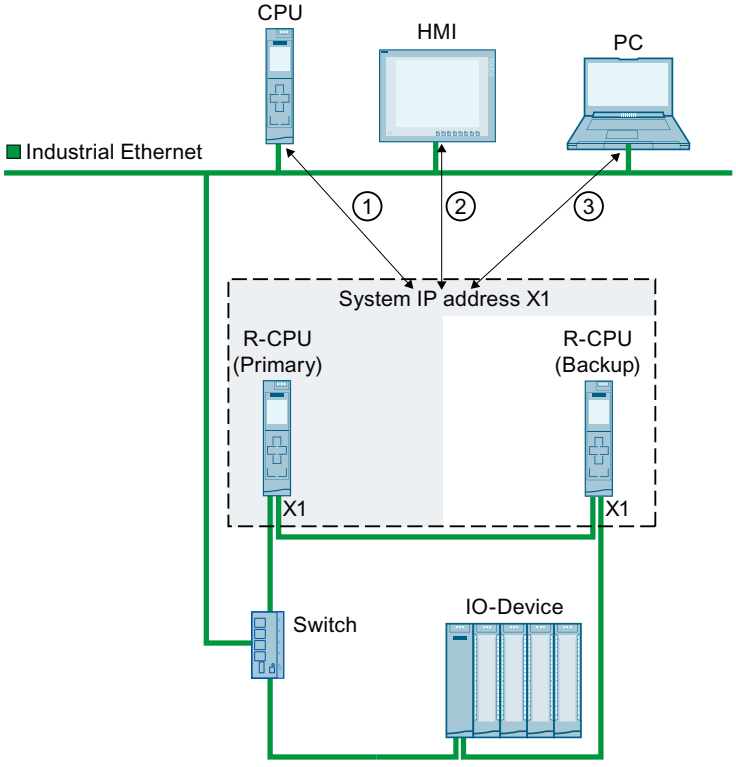


① Open User Communication between a different CPU and the S7-1500R/H redundant system
② HMI communication with the S7-1500R/H redundant system
③ Open User Communication between the S7-1500R/H redundant system and a PC

Figure 15-1 Example: Communication of the S7-1515R redundant system via the system IP address X2

Communication via the system IP address X1

The following diagram shows a configuration where the communication partners are connected with a switch to the PROFINET ring of the S7-1500R/H redundant system. The PROFINET ring connects the communication partners with the respective PROFINET interfaces X1 of the two CPUs.
As the CPU 1513R only has one PROFINET interface, connection via the PROFINET ring is the only possibility of communicating via the system IP address X1.



① Open User Communication between the S7-1500R/H redundant system and a different CPU

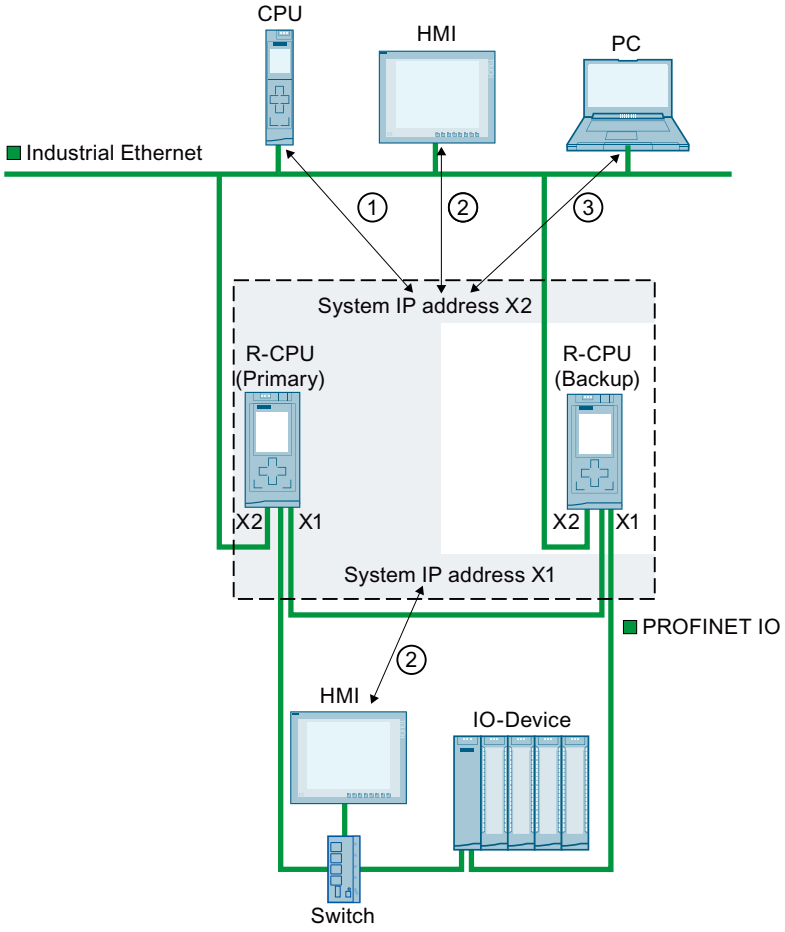
② HMI communication with the S7-1500R/H redundant system

③ Open User Communication between the S7-1500R/H redundant system and a PC

Figure 15-2 Example: Communication of the S7-1513R redundant system via the system IP address X1

Communication via the system IP addresses X1, X2 and X3

You can use one system IP address for each PROFINET interface of the redundant S7-1500R/H system. PROFINET devices which are connected to the interfaces X1 of the CPUs communicate via the system IP address X1. PROFINET devices which are connected to the interfaces X2 of the CPUs communicate via the system IP address X2. PROFINET devices which are connected to the interfaces X3 of the CPUs communicate via the system IP address X3.



① Open User Communication between the S7-1500R/H redundant system and a different CPU.

② HMI communication with the S7-1500R/H redundant system

③ Open User Communication between the S7-1500R/H redundant system and a PC

Figure 15-3 Example: Communication of the S7-1515R redundant system via the system IP addresses X1 and X2

With an S7-1500H(F) system, you also have the option of dividing your system into several PROFINET rings.

In this case you must connect the required S1/S2 devices in a separate PROFINET ring behind a Y-switch.

Recommendation: For an increased availability of the S1/S2 devices you need two Y-switches with DNA redundancy (SCALANCE XF204-2BA DNA). A Y-switch takes on the roles of MRP manager and DNA manager. Another Y-switch takes on the roles MRP client and DNA client. DNA redundancy is only possible with a connected PROFINET ring.

You can find more information on configuration scenarios with Y-switch in the S7-1500R/H Redundant System (https://support.industry.siemens.com/cs/ww/en/view/109754833) system manual.

IP forwarding via the system IP address

If you use the system IP addresses as the gateway/default route for IP routes through the S7-1500R/H redundant system, IP packets are forwarded even if one CPU fails.
In the following figure, the PC is connected to the two X2 interfaces of the S7-1500R CPUs. Enter the system IP address X2 as gateway in the PC for the route to the HMI device. The HMI device is connected to the PROFINET ring of the redundant system S7-1500 via a switch. The system IP address X1 is configured as router in the HMI device.

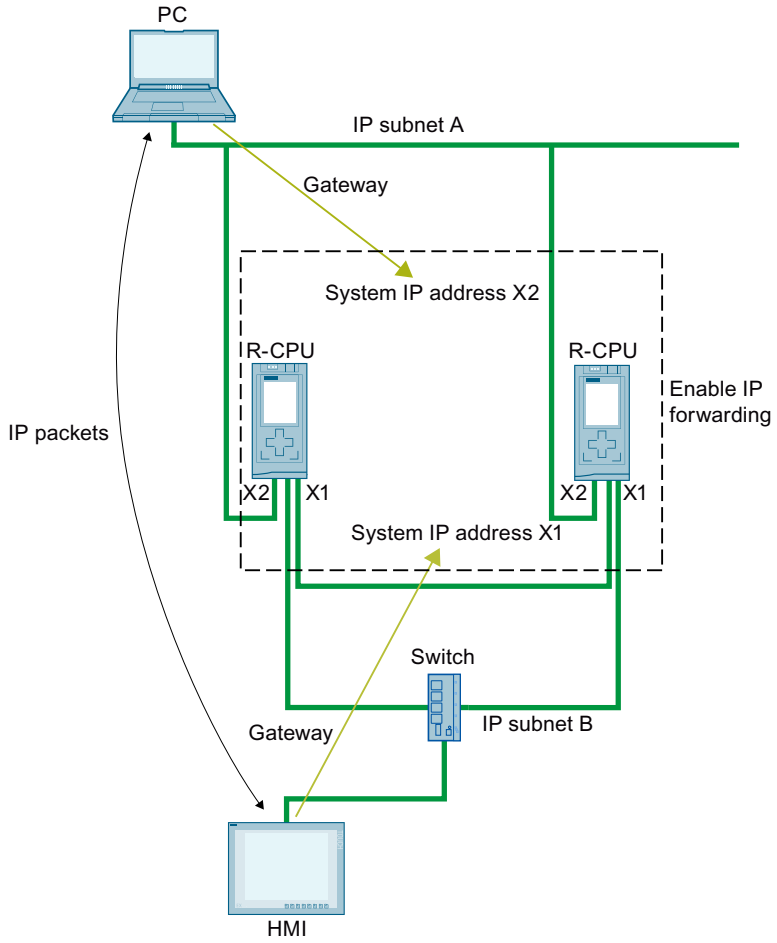


Figure 15-4 Example: IP forwarding via the system IP address

Enable system IP addresses

Requirements:

- STEP 7 V15.1 or higher
- S7-1500R/H redundant system with two CPUs, e.g. two CPUs 1513R-1PN

If the CPUs of the S7-1500R/H redundant system have two PROFINET interfaces (X1 and X2), then you can use a system IP address for both PROFINET interfaces. The following section describes how to enable the system IP address for the interface X1.

Proceed as follows to enable the system IP address for your S7-1500R/H redundant system:

1. In the network view of STEP 7, select the interface X1 of one of the two CPUs.
2. In the Inspector window, go to "Properties" > "General" > "Ethernet addresses" in the area "System IP address for switched communication".
3. Select the check box "Enable the system IP address for switched communication".
   STEP 7 automatically creates a system IP address.

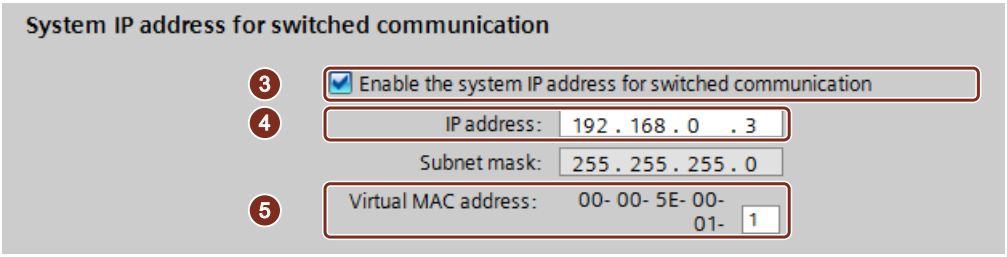


Figure 15-5 Configure IP address

4. Change the system IP address if necessary.
5. If required, change the virtual MAC address. To do this, in "Virtual MAC address", assign a project-wide unique value (value range 01_H to FF_H) for the last byte.

NOTE

Uniqueness of the virtual MAC address

The redundant system S7-1500R/H uses a MAC address from the address range 00-00-5E-00-01-00 to 00-00-5E-00-01-00 for each system IP address. This address range is also used for VRRP (Virtual Redundancy Protocol).

If you use devices with VRRP, e.g. switches, ensure the uniqueness of the MAC addresses within an Ethernet broadcast domain.

Result: The system IP address X1 for the PROFINET interface X1 of the two CPUs is enabled.

15.2 Response to Snycup

Response of communication connections via the system IP address in the system state SYNCUP

- HMI, PG- and S7-connections are temporarily closed. For a short time during the SYNCUP it is not possible to establish connections to the S7-1500R/H redundant system.
- All existing connections of Open User Communication are interrupted:
  - Connections set up by the CPUs of the redundant system as an active connection partner are set up again after the SYNCUP.
  - The S7-1500R/H redundant system sets up connection endpoints again for the passive connection establishment after the SYNCUP.
- The processing of running instances of the instructions TSEND and TRCV is stopped. The block parameter STATUS returns $80C4_H$ (temporary communication error).

15.3 Response to primary-backup switchover

Response of communication connections via the system IP address during a primary-backup switchover

- Running instances of the instructions TSEND and TRCV are stopped and return the status $80C4_H$ (temporary communication error).
- Connections successfully established by the S7-1500R/H redundant system are established again by the new primary CPU.
- The new primary CPU sets up connection endpoints again for the passive connection establishment.

NOTE

Increased duration of connection interruption

If the remote system does not transmit actively after the primary-backup switchover, the connection monitoring (e.g. TCP-Keep-Alive or application) may have to be performed by the remote system until the connection can be re-established.

15.4 Connection resources of the redundant system S7-1500R/H

Maximum number of connection resources of the S7-1500R/H redundant system

The S7-1500R/H redundant system supports a maximum number of connection resources.

The CPU used determines the maximum number of resources for the redundant system:

- CPU 1513R: max. 88 connection resources
- CPU 1515R: max. 128 connection resources (as of V3.0), max. 108 connection resources (up to V2.9.x)
- CPU 1517H: max. 288 connection resources
- CPU 1518HF: max. 320 connection resources

Allocation of connection resources

Communication connections occupy communication resources in the S7-1500R/H redundant system.

Each communication connection to the redundant system S7 1500R/H occupies connection resources in the S7 1500R/H station. The S7-1500R/H station comprises the hardware setup of both CPUs of the redundant S7-1500R/H system.

Depending on the IP address used, a communication connection also uses connection resources in one or both CPUs of the redundant S7-1500R/H system. The S7-1500R/H station can also be used to establish a communication connection.

The following table shows in which CPU a communication connection occupies connection resources depending on the IP address used.

Connect via...	Connection resources of the station	Connection resources CPU with redundancy ID 1	Connection resources CPU with redundancy ID 2
a system IP address	X	X	X
a device IP address of the CPU with redundancy ID 1	X	X	-
a device IP address of the CPU with redundancy ID 2	X	-	X

Display of the occupied connection resources in STEP 7

Requirement: Online connection to the redundant system S7-1500R/H

You will find the online display of the connection resources in the inspector window under "Diagnostics" > "Connection information". STEP 7 always displays the connection resources of the selected CPU and the S7-1500R/H-station.

Connection resources

	Station resources					Module resources	
	Reserved			Dynamic		CPU 1517H-3 PN (R0/S1)	
Maximum number of resources:		10	10	150	150	160	160
	Maximum	Configured	Used	Configured	Used	Configured	Used
PG communication:	4	-	2	-	0	-	2
HMI communication:	4	0	0	0	0	0	0
S7 communication:	0	-	0	0	0	0	0
Open user communication:	0	-	0	0	0	0	0
Web communication:	2	-	0	-	0	-	0
Other communication:	-	-	0	0	0	0	0
Total resources used:		0	2	0	0	0	2
Available resources:		10	8	150	150	160	158

Figure 15-6 Display of the connection resources of the S7-1500R/H redundant system in STEP 7

15.5 HMI communication with the redundant system S7-1500R/H

15.5.1 HMI connection via the system IP address

Requirements

- A S7-1500R/H redundant system, e.g. CPU 1513R-1PN
- System IP address is enabled
- HMI device with PROFINETI interface

Procedure

To set up a HMI connection to an S7-1500R/H redundant system, follow these steps:

1. In the network view of STEP 7, select a PROFINET interface of the HMI device.
2. Using a drag&drop operation, draw a line between the PROFINET interface of the HMI device and a PROFINET interface of the S7-1500R/H redundant system.
   The HMI device and the S7-1500R/H redundant system are networked together.

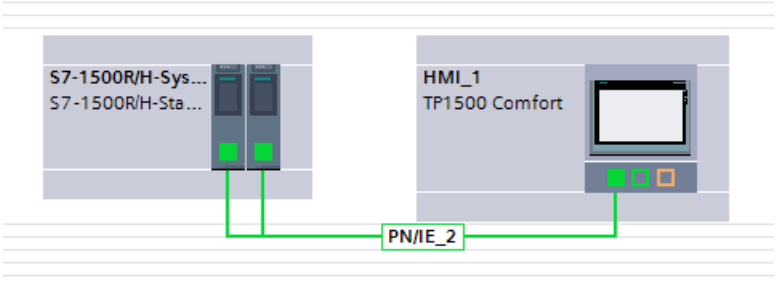


Figure 15-7 Networking an HMI device with the S7-1500R/H redundant system

3. In the list of functions, click the "Connections" icon. This activates connection mode.
4. Using a drag-and-drop operation, draw a line between the HMI device and a CPU of the S7-1500R/H redundant system.
   The list "Connection partners" opens.

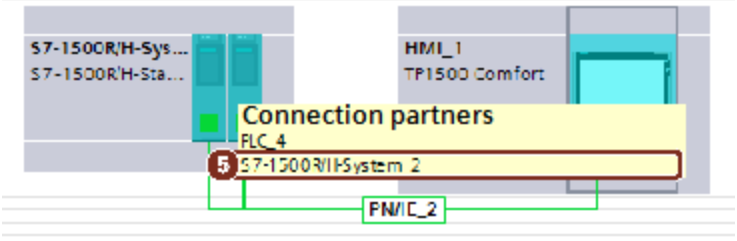


Figure 15-8 Setting up an HMI connection to the S7-1500R/H redundant system

5. Select the S7-1500R/H redundant system in the list "Connection partners".

Result: You have set up a HMI connection between the HMI device and the S7-1500R/H redundant system. The HMI connection uses the system IP address. The HMI device always connects to the primary CPU.

Changing the HMI connection over to the device IP address

To permanently change the HMI connection over to the selected CPU, clear the check box "Use the system IP address for switched communication" in the properties of the HMI connection. The HMI connection then uses the device IP address of the PROFINET interface. In the event of the failure of this CPU, then the HMI connection to this CPU permanently fails.

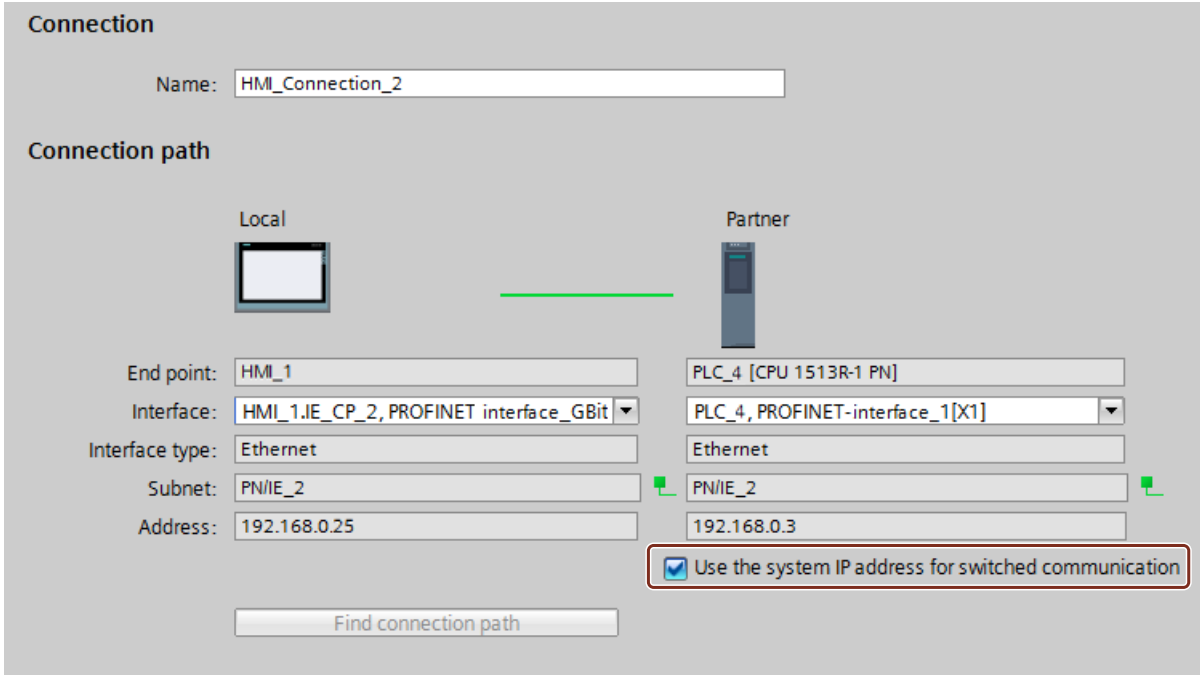


Figure 15-9 Properties of the HMI connection

NOTE

Automatic setup of HMI connection

When you drag-and-drop a tag from the S7-1500R/H redundant system into an HMI screen or into the HMI tag table, STEP 7 automatically sets up an HMI connection. This HMI connection exists by default between the PROFINET interface of the HMI device and the PROFINET interface X1 of the CPU with redundancy ID 1. The connection uses the device IP address of the PROFINET interface X1.

You can change the HMI connection to a system IP address in the properties of the HMI connection.

More information

You can set up an HMI connection to the S7-1500R/H redundant system via the devices IP address. With help of scripts in the HMI configuration, the connection of the failed CPU are switched automatically to the still running CPU. A description to this procedure can be found in the following FAQ (https://support.industry.siemens.com/cs/us/en/view/109781687).

15.6 Open User Communication with the redundant system S7-1500R/H

The following table shows which protocols of the Open User Communication you can use for the S7-1500R/H redundant system and the matching system data types and instructions.

Table 15-1 Protocols, system data types and usable instructions for Open User Communication with the redundant system S7-1500R/H

Protocol	System data type	Instructions
TCP	• TCON_QDN • TCON_IP_v4	Establish connection and send/receive data via: • TSEND_C/TRCV_C or • TCON, TSEND/TRCV or • TCON, TUSEND/TURCV (connection can be terminated via TDISCON)
ISO-on-TCP	• TCON_IP_RFC	
UDP	• TCON_IP_v4 • TADDR_Param • TADDR_SEND_QDN • TADDR_RCV_IP	Establish connection and send/receive data via: • TSEND_C/TRCV_C • TUSEND/TURCV/TRCV (connection can be terminated via TDISCON)
Modbus TCP	• TCON_IP_v4 • TCON_QDN	• MB_CLIENT • MB_RED_CLIENT • MB_SERVER • MB_RED_SERVER

15.6.1 Setting up the connection of the Open User Communication with the redundant S7-1500R/H system

Introduction

The S7-1500R/H redundant system can communicate with other devices via Open User Communication.

You set up the connections in the user program, e.g. via the "TSEND_C" instruction. The S7-1500R/H redundant system does not support configured connections.

You can either set up the connections either via the device IP addresses or via the system IP addresses of the PROFINET interfaces.

Open User Communication via a system IP address of the redundant system S7 1500R/H

If you set up the connection via a system IP address, then communication always takes place via the primary CPU.

Recommendation: Always use a system IP address for Open User Communication.

Open User Communication via a device IP address of the redundant system S7 1500R/H

In redundant mode, the redundant system can establish or terminate connections and send or receive data via every device IP address.

If you set up the connection via a device IP address, then communication takes place via the associated CPU. In the event of the failure of the CPU, then the entire communication via the device IP addresses of this CPU fails.

Setting up a connection via a system IP address

The following describes how to establish a connection to another CPU via a system IP address of a PROFINET interface of the redundant S7 1500R/H system.

You set up the connection in the user program of the redundant system S7-1500R/H with a TSEND_C instruction. You create a corresponding TRCV_C instruction in the user program of the other CPU.

The procedure is described using the example of a TCP connection between the S7-1500R/H redundant system and a CPU 1516-3PN/DP.

Requirements

- An S7-1500R/H redundant system, e.g. 2 CPUs 1513 1PN
- System IP address of the PROFINET interface X1 is enabled.
- CPU 1516-3PN/DP
- The PROFINET interfaces X1 of the CPUs 1513R and the PROFINET interface X2 of the CPU 1516-3PN/DP are located in the same subnet.

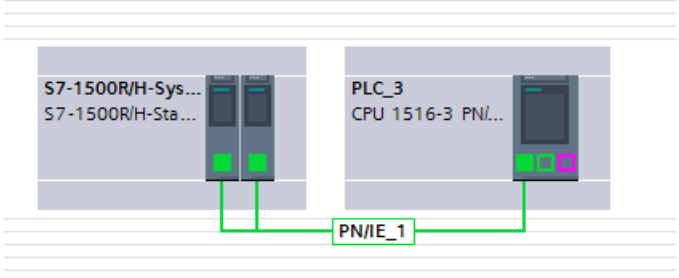


Figure 15-10 Example configuration for TCP-connection

TSEND_C instruction in the user program of the S7-1500R/H redundant system

To set up a TCP-connection to a different CPU, follow these steps:

1. Create a "TSEND_C" instruction in the user program.

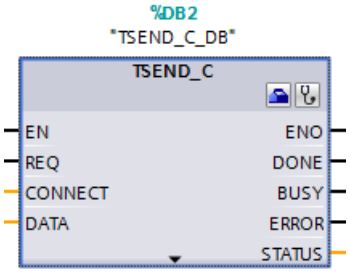


Figure 15-11 S7-1500R/H: "TSEND_C" instruction

2. Select the "TSEND_C" instruction.
3. In the Inspector window, go to "Properties" > "Configuration" > "Connection parameters". On the left-hand side you can see the S7-1500R/H redundant system as a local end point of the connection:
   - "Interface:": X1 is the preset interface.
   - "Subnet:": If the interface X1 is assigned to an S7-subnet, then STEP 7 displays the name of the S7-subnet.
   - The check box "Use address of the H-system" is selected. The system IP address of the S7-1500R/H redundant system is in "Address".

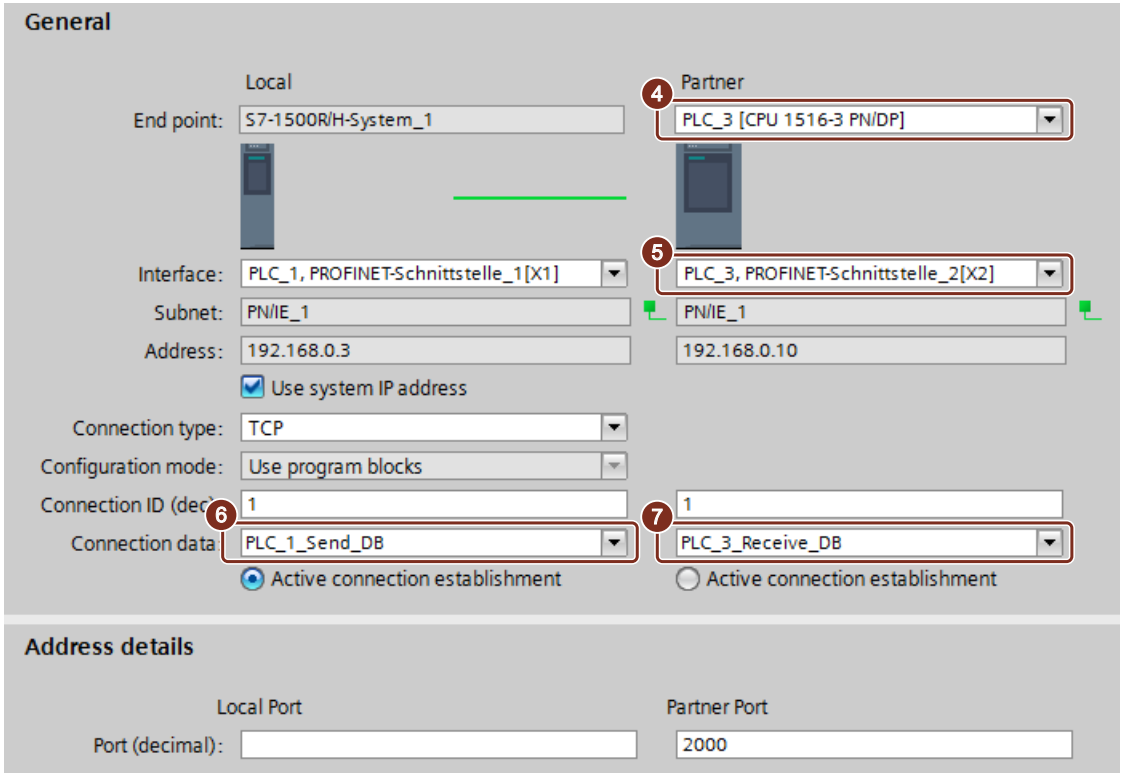


Figure 15-12 S7-1500R/H: Assigning parameters to the TSEND_C instruction in STEP 7

4. In "Partners" under "End point:" select the CPU 1516-3PN/DP as the communication partner.
5. In "Partners" under "Interface:" select the PROFINET interface X2 of the CPU 1516-3PN/DP.
6. In "Local" under "Connection data" select the setting "<new>".
   STEP 7 creates a data block for the connection data in the user program of the S7-1500R/H redundant system, for example "PLC_1_Send_DB".
   "TCP" is set by default as the connection type.
7. In "Partners" under "Connection type" select the setting "NEW".
   STEP 7 creates a data block for the connection data in the user program of the other CPU, for example "PLC_3_Receive_DB".

TRCV_C instruction in the user program of the CPU 1516 3PN/DP

Create a TRCV_C instruction in the user program of the CPU 1516-3PN/DP and assign parameters as below:

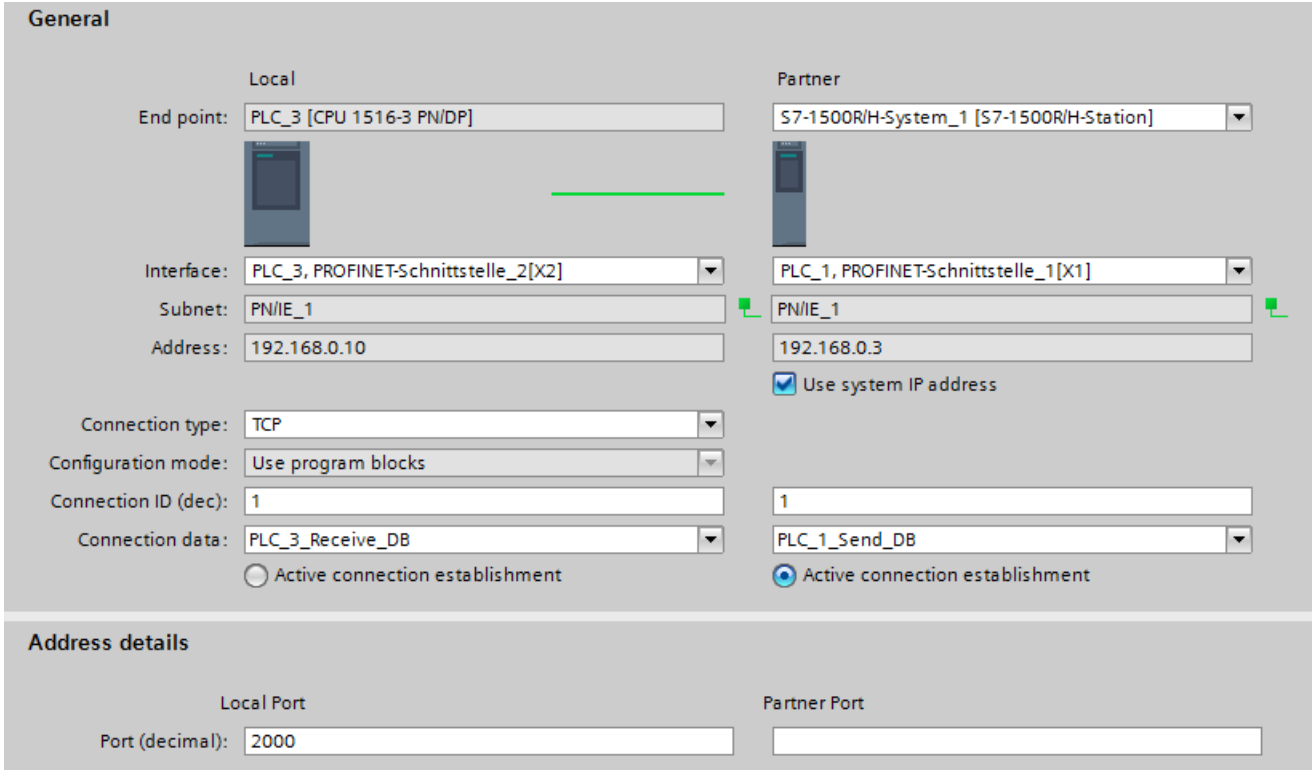


Figure 15-13 S7-1500-3PN/DP: Assigning parameters to the TRCV_C instruction in STEP 7

Setting up a connection via a device IP address

To set up an OUC-connection via a device IP address of one of the two CPUs:

- Select a suitable PROFINET interface of the S7-1500R/H redundant system.
- Deselect the "Use address of H-system" check box.

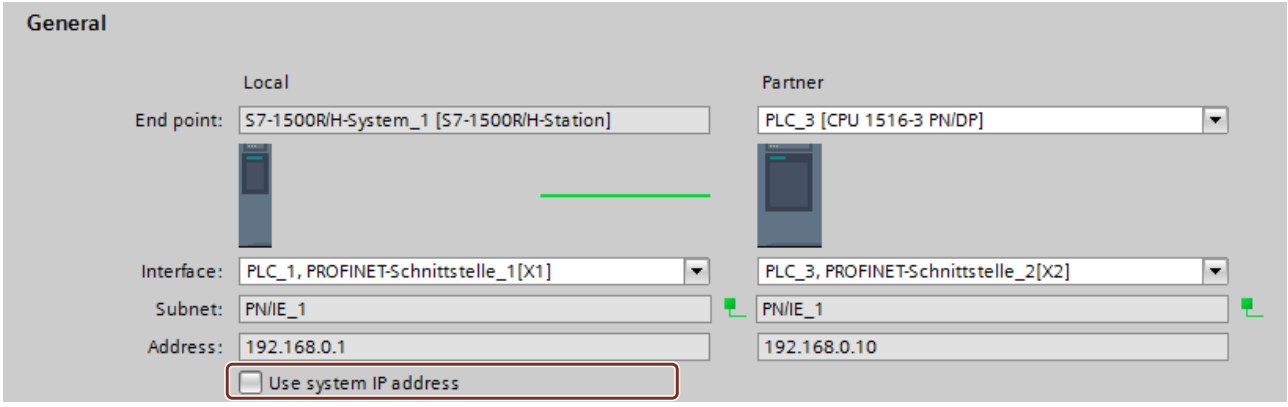


Figure 15-14 OUC-connection via a device IP address

More information

You can find more information on system states in the S7-1500R/H (https://support.industry.siemens.com/cs/ww/en/view/109754833) system manual.
You can find more information on the configuration and parameter assignment of your PROFINET IO system in the PROFINET function manual (https://support.industry.siemens.com/cs/ww/en/view/49948856).

Industrial Ethernet Security with CP 1543-1 16

All-round protection - the task of Industrial Ethernet Security

With Industrial Ethernet Security, individual devices, automation cells or network segments of an Ethernet network can be protected. Data transfer can also be protected by a combination of different security measures:

- Data espionage
- Data manipulation
- Unauthorized access

Security measures

- Firewall
  - IP firewall with stateful packet inspection (layer 3 and 4)
  - Firewall also for Ethernet "non-IP" frames according to IEEE 802.3 (layer 2)
  - Bandwidth limitation
  - Global firewall rules

  All network nodes located in the internal network segment of a CP 1543-1 are protected by its firewall. Exception: If you access the CPU via the interface of the CP with the "Access to PLC via communication module" function, the firewall does not protect this connection.
- Logging
  To allow monitoring, events can be stored in log files that can be read out using the configuration tool or can be sent automatically to a Syslog server.
- HTTPS
  For encrypted transfer of websites, for example during process control.
- FTPS (explicit mode)
  For encrypted transfer of files.
- Secure NTP
  For secure time-of-day synchronization and transmission.
- SNMPv3
  For secure transmission of network analysis information safe from eavesdropping.
- VPN groups
  You can combine the CP 1543-1 with other security modules into VPN groups through configuration. IPsec tunnels are established between all the security modules of a VPN group (VPN). All internal nodes of these security modules can communicate securely with each other through this tunnel.
- Protection for devices and network segments
  The firewall and VPN groups protective functions can be applied to the operation of single devices, multiple devices, or entire network segments.

Additional information

An overview with links to the most important contributions on Industrial Security is available in this FAQ (https://support.industry.siemens.com/cs/ww/en/view/92651441).

16.1 Firewall

Tasks of the firewall

The purpose of the firewall functionality is to protect networks and stations from outside influences and disturbances. This means that only certain previously specified communications relations are permitted.

To filter the data traffic, IPv4 addresses, IPv4 subnets, port numbers or MAC addresses among other things can be used.

The firewall functionality can be configured for the following protocol levels:

- IP firewall with stateful packet inspection (layer 3 and 4)
- Firewall also for Ethernet "non-IP" frames according to IEEE 802.3 (layer 2)

Firewall rules

Firewall rules describe which packets are permitted or forbidden in which direction.

16.2 Logging

Functionality

For test and monitoring purposes, the security module has diagnostics and logging functions.

- Diagnostics functions
  These include various system and status functions that you can use in online mode.
- Logging functions
  This involves the recording of system and security events. Depending on the event type, the recording is made in volatile or non-volatile local buffer areas of the CP 1543-1. As an alternative, it is also possible to record on a network server.
  The parameter assignment and evaluation of these functions is only possible with a network connection.

Recording events with logging functions

You specify which events should be recorded with the log settings. Here you can configure the following recording variants:

- Local logging
  With this variant, you record the events in local buffers of the CP 1543-1. In the online dialog of the Security Configuration Tool, you can then access these recordings, visualize them and archive them on the service station.
- Network Syslog
  With the network Syslog, you use a Syslog server in the network. This records the events according to the configuration in the log settings.

16.3 NTP client

Functionality

To check the time validity of a certificate and the time stamp of log entries, the date and time are maintained on the CP 1543-1 as on the CPU. This time can be synchronized with NTP. The CP 1543-1 forwards the synchronized time to the CPU via the backplane bus of the automation system. This way the CPU also receives a synchronized time for the time events in program execution.

The automatic setting and periodic synchronization of the time takes place either via a secure or non-secure NTP server. You can assign a maximum of 4 NTP servers to the CP 1543-1. A mixed configuration of non-secure and secure NTP servers is not possible.

16.4 SNMP

Functionality

Like the CPU, the CP 1543-1 supports the transfer of management information using the Simple Network Management Protocol (SNMP). To achieve this, an "SNMP agent" is installed on the CP/CPU that receives and responds to the SNMP queries. Information about the properties of devices capable of SNMP is contained in so-called MIB files (Management Information Base) for which the user needs to have the appropriate rights.

With SNMPv1, the "community string" is also sent. The "community string" is like a password that is sent along with the SNMP query. The requested information is sent when the "community string" is correct. The request is discarded when the string is incorrect.

With SNMPv3, data can be transferred encrypted. To do this, select either an authentication method (e.g. SHA) or an authentication and encryption method (e.g AES).

You can activate and deactivate the use of SNMP for the CP/CPU. Deactivate SNMP if the security guidelines in your network do not permit SNMP or if you use your own SNMP solution.

To find out how to activate and deactivate SNMP for the CPU, refer to section SNMP (Page 103).

16.5 VPN

Functionality

For security modules that protect the internal network, VPN (Virtual Private Network) tunnels provide a secure data connection through the non-secure external network.

The module uses the IPsec protocol (tunnel mode of IPsec) for tunneling.

In STEP 7 you can assign VPN groups to security modules. VPN tunnels are automatically established between all modules of a VPN group. A module in one project can belong to several different VPN groups at the same time in the process.

Glossary

Automation system

Programmable logic controller for the open-loop and closed-loop control of process chains of the process engineering industry and manufacturing technology. The automation system consists of different components and integrated system functions according to the automation task.

Backup CPU

If the R/H system is in RUN-Redundant system state, the primary CPU controls the process. The backup CPU processes the user program synchronously and can take over process control if the primary CPU fails.

Bus

Transmission medium that connects several devices together. Data transmission can be performed electrically or via optical fibers, either in series or in parallel.

Client

Device in a network that requests a service from another device in the network (server).

CM

→ Communications module

Communications module

Module for communications tasks used in an automation system as an interface expansion of the CPU (for example PROFIBUS) and providing additional communications options (PtP).

Communications processor

Module for expanded communications tasks covering special applications, for example in the area of security.

Consistent data

Data that belongs together in terms of content and must not be separated when transferred.

CP

→ Communications processor

CPU

Central **P**rocessing **U**nit - Central module of the S7 automation system with a control and arithmetic unit, memory, operating system and interface for programming device.

Device

Generic term for:

- Automation systems (PLC, PC, for example)
- Distributed I/O systems
- Field devices (for example, PLC, PC, hydraulic devices, pneumatic devices) and
- Active network components (for example, switches, routers)
- Gateways to PROFIBUS, AS interface or other fieldbus systems

Device certificates

Such certificates are signed by a certificate authority (CA).
The signature of an end-entity certificate is checked with the public key of the certificate authority certificate.
The "Subject" attribute must not be identical to the "Issuer" attribute.
The "Subject", for example, contains the name of a program as with the OPC UA application certificate.
"Issuer" is the certificate authority that signed the certificate.
The "CA" field must be set to "False".

DP master

Within PROFIBUS DP, a master in the distributed I/O that behaves according to the EN 50170 standard, Part 3.
→ *See also DP slave*

DP slave

Slave in the distributed I/O that is operated on PROFIBUS with the PROFIBUS DP protocol and behaves according to the EN 50170 standard, Part 3.
→ *See also DP master*

Duplex

Data transmission system; a distinction is made between full and half duplex.
Half duplex: One channel is available for alternate data exchange (sending or receiving alternately but not at the same time).
Full duplex: Two channels are available for simultaneous data exchange in both directions (simultaneous sending and receiving in both directions).

End-entity certificate

→ *See also device certificate*

Ethernet

International standard technology for local area networks (LAN) based on frames. It defines types of cables and signaling for the physical layer and packet formats and protocols for media access control.

Ethernet network adapter

Electronic circuitry for connecting a computer to an Ethernet network. It allows the exchange of data / communication within the network.

FETCH/WRITE

Server services using TCP/IP, ISO-on-TCP and ISO for access to system memory areas of S7 CPUs. Access (client function) is possible from a SIMATIC S5 or a third-party device/PC. FETCH: Read data directly; WRITE: Write data directly.

Field device

→ Device

Freeport

Freely programmable ASCII protocol; here for data transfer via a point-to-point connection.

FTP

File **T**ransfer **P**rotocol; a network protocol for transferring files via IP networks. FTP is used to download files from the server to the client or to upload files from the client to the server. FTP directories can also be created and read out and directories and files can be renamed or deleted.

HMI

Human **M**achine **I**nterface, device for visualization and control of automation processes.

IE

→ Industrial Ethernet

IM

→ Interface module

Industrial Ethernet

Guideline for setting up an Ethernet network in an industrial environment. The essential difference compared with standard Ethernet is the mechanical ruggedness and immunity to noise of the individual components.

Instruction

The smallest self-contained unit of a user program characterized by its structure, function or purpose as a separate part of the user program. An instruction represents an operation procedure for the processor.

Interface module

Module in the distributed I/O system. The interface module connects the distributed I/O system via a fieldbus to the CPU (IO controller/DP master) and prepares the data for the I/O modules.

Intermediate CA certificate

This is a certificate authority certificate that is signed with the private key of a root certificate authority.
An intermediate certificate authority signs end-entity certificates with its private key.
The signature of these end-entity certificates is verified with the public key of the intermediate certificate authority.
The "Subject" and "Issuer" attributes of the intermediate CA certificate must not be identical. This certificate authority has after all not signed its certificate itself.
The "CA" field must be set to "True".

IO controller, PROFINET IO controller

Central device in a PROFINET system, usually a classic programmable logic controller or PC. The IO controller sets up connections to the IO devices, exchanges data with them, thus controls and monitors the system.

IO device, PROFINET IO device

Device in the distributed I/O of a PROFINET system that is monitored and controlled by an IO controller (for example distributed inputs/outputs, valve islands, frequency converters, switches).

IP address

Binary number that is used as a unique address in computer networks in conjunction with the Internet Protocol (IP). It makes these devices uniquely addressable and individually accessible. An IPv4 address can be evaluated using a binary subnet mask that results in a network part or a host part as a structure. The textual representation of an IPv4 address consists, for example, of 4 decimal numbers with the value range 0 to 255. The decimal numbers are separated by periods.

IPv4 subnet mask

Binary mask, with which an IPv4 address (as a binary number) is divided into a "network part" and a "host part".

ISO protocol

Communications protocol for message or packet-oriented transfer of data in an Ethernet network. This protocol is hardware-oriented, very fast and allows dynamic data lengths. The ISO protocol is suitable for medium to large volumes of data.

ISO-on-TCP protocol

Communications protocol capable of S7 routing for packet-oriented transfer of data in an Ethernet network; provides network addressing. The ISO-on-TCP protocol is suitable for medium and large volumes of data and allows dynamic data lengths.

Linear bus topology

Network topology characterized by the arrangement of the devices in a line (bus).

MAC address

Worldwide unique device identification for all Ethernet devices. The MAC address is assigned by the manufacturer and has a 3-byte vendor ID and 3-byte device ID as a consecutive number.

Master

Higher-level, active participant in the communication/on a PROFIBUS subnet. The master has rights to access the bus (token) and can request and send data.
→ *See also DP master*

Modbus RTU

Remote **T**erminal **U**nit; Open communications protocol for serial interfaces based on a master/slave architecture.

Modbus TCP

Transmission **C**ontrol **P**rotocol; Open communications protocol for Ethernet based on a master/slave architecture. The data are transmitted as TCP/IP packets.

Network

A network consists of one or more interconnected subnets with any number of devices. Several networks can exist alongside each other.

NTP

The **N**etwork **T**ime **P**rotocol (NTP) is a standard for synchronizing clocks in automation systems via Industrial Ethernet. NTP uses the connectionless UDP transport protocol for the Internet.

OPC UA

OPC Unified **Automation** is a protocol for communication between machines, developed by the OPC Foundation.

Operating states

Operating states describe the behavior of a single CPU at a specific time.
The CPUs of the SIMATIC standard systems have the STOP, STARTUP and RUN operating states.
The primary CPU of the redundant system S7-1500R/H has the operating states STOP, STARTUP, RUN, RUN-Syncup and RUN-Redundant. The backup CPU has the operating states STOP, SYNCUP and RUN-Redundant.

Operating system

Software that allows the use and operation of a computer. The operating system manages resources such as memory, input and output devices and controls the execution of programs.

PG

→ Programming device

PNO

→ PROFIBUS user organization

Point-to-point connection

Bidirectional data exchange via communications modules with a serial interface between two communications partners (and two only).

Port

Physical connector to connect devices to PROFINET. PROFINET interfaces have one or more ports.

Primary CPU

If the R/H system is in RUN-Redundant system state, the primary CPU controls the process. The backup CPU processes the user program synchronously and can take over process control if the primary CPU fails.

Process image (I/O)

The CPU transfers the values from the input and output modules to this memory area. At the start of the cyclic program, the CPU transfers the process image output as a signal state to the output modules. The CPU then reads the signal states of the input modules into the process image input. The CPU then executes the user program.

PROFIBUS

Process **Fi**eld **Bus** - European Fieldbus standard.

PROFIBUS address

Unique identifier of a device connected to PROFIBUS. The PROFIBUS address is sent in the frame to address a device.

PROFIBUS device

Device with at least one PROFIBUS interface either electrical (for example RS-485) or optical (for example Polymer Optical Fiber).

PROFIBUS user organization

Technical committee dedicated to the definition and development of the PROFIBUS and PROFINET standard.

PROFIBUS DP

A PROFIBUS with DP protocol that complies with EN 50170. DP stands for distributed I/O = fast, real-time capable, cyclic data exchange. From the perspective of the user program, the distributed I/O is addressed in exactly the same way as the centralized IO.

PROFINET

Open component-based industrial communications system based on Ethernet for distributed automation systems. Communications technology promoted by the PROFIBUS user organization.

PROFINET device

Device that always has a PROFINET interface (electrical, optical, wireless).

PROFINET interface

Interface of a module capable of communication (for example CPU, CP) with one or more ports. A MAC address is assigned to the interface in the factory. Along with the IP address and the device name (from the individual configuration), this interface address ensures that the PROFINET device is identified uniquely in the network. The interface can be electrical, optical or wireless.

PROFINET IO

IO stands for input/output; distributed I/O (fast, cyclic data exchange with real-time capability). From the perspective of the user program, the distributed I/O is addressed in exactly the same way as the centralized IO.

PROFINET IO as the Ethernet-based automation standard of PROFIBUS & PROFINET International defines a cross-vendor communication, automation, and engineering model.

With PROFINET IO, a switching technology is used that allows all devices to access the network at any time. In this way, the network can be used much more efficiently through the simultaneous data transfer of several devices. Simultaneous sending and receiving is enabled via the full-duplex operation of Switched Ethernet.

PROFINET IO is based on switched Ethernet with full-duplex operation and a bandwidth of 100 Mbps.

Programming device

Programming devices are essentially compact and portable PCs which are suitable for industrial applications. They are identified by a special hardware and software configuration for programmable logic controllers.

Protocol

Agreement on the rules by which the communication between two or more communication partners transpires.

PtP

Point-**t**o-**P**oint, interface and/or transmission protocol for bidirectional data exchange between two (and only two) communications partners.

Redundant systems

Redundant systems have multiple (redundant) instances of key automation components. Process control is maintained if a redundant component fails.

Ring topology

All devices of a network are connected together in a ring.

Root CA certificates

→ *See also root certificate*

Root certificate

This is the certificate of a certificate authority: It signs end-entity certificates and intermediate CA certificates with its private key.
The "Subject" attribute and the "Issuer" of this certificate must be identical. This certificate authority has signed its certificate itself.
The "CA" field must be set to "True".
TIA Portal V14 has such a root CA certificate:
If you configure the OPC UA server of an S7-1500 in the TIA Portal, the TIA Portal generates an end-entity certificate for the OPC UA server and signs that certificate with its own private key.
The signature of this end-entity certificate can be verified with the public key of the TIA Portal. This key can be found in the root CA certificate of the TIA Portal.

Router

Network node with a unique identifier (name and address) that connects subnets together and allows transportation of data to uniquely identified communications nodes in the network.

RS232, RS422 and RS485

Standard for serial interfaces.

RTU

Modbus RTU (RTU: **R**emote **T**erminal **U**nit, transfers the data in binary form; allows a good data throughput. The data must be converted to a readable format before it can be evaluated.

S7 routing

Communication between S7 automation systems, S7 applications or PC stations in different S7 subnets via one or more network nodes functioning as S7 routers.

SDA service

Send Data with Acknowledge. SDA is an elementary service with which an initiator (for example DP master) can send a message to other devices and then receives acknowledgment of receipt immediately afterwards.

SDN service

Send Data with No Acknowledge. This service is used primarily to send data to multiple stations and the service therefore remains unacknowledged. Suitable for synchronization tasks and status messages.

Security

Generic term for all the measures taken to protect against

- Loss of confidentiality due to unauthorized access to data
- Loss of integrity due to manipulation of data
- Loss of availability due to the destruction of data

Self-signed certificates

These are certificates that you sign with your private key and use as end-entity certificates.
The signature of these end-entity certificates is verified with your public key.
The "Subject" and "Issuer" attributes of self-signed certificates must be identical: You have signed your certificate yourself.
The "CA" field must be set to "False".
You can, for example, use self-signed certificates as application certificates for an OPC UA client.
The procedure required to generate a self-signed certificate with the certificate generator of the OPC Foundation is described here (Page %getreference).

Server

A device or more generally an object that can provide certain services; the service is performed at the request of a client.

Slave

Distributed device in a fieldbus system that can only exchange data with a master after the master has requested this.
→ *See also DP slave*

SNMP

Simple **N**etwork **M**anagement **P**rotocol, uses the wireless UDP transport protocol. SNMP works in much the same way as the client/server model. The SNMP manager monitors the network nodes. The SNMP agents collect the various network-specific information in the individual network nodes and makes this information available in a structured form in the MIB (**M**anagement **I**nformation **B**ase). This information allows a network management system to run detailed network diagnostics.

Subnet

Part of a network whose parameters must be matched up on the devices (for example in PROFINET). A subnet includes the bus components and all connected stations. Subnets can be linked together, for example using gateways or routers to form one network.

Switch

Network components used to connect several terminal devices or network segments in a local network (LAN).

Switched communication

In addition to the device IP addresses of the CPUs, the redundant system S7-1500R/H supports system IP addresses:

- System IP address for the X1 PROFINET interfaces of the two CPUs (system IP address X1)
- System IP address for the X2 PROFINET interfaces of the two CPUs (system IP address X2)

You use the system IP addresses for communication with other devices (for example, HMI devices, CPUs, PG/PC). The devices always communicate over the system IP address with the primary CPU of the redundant system. This ensures that the communication partner can communicate with the new primary CPU (previously backup CPU) in the RUN-Solo system state after failure of the original primary CPU in redundant operation.

System states

The system states of the redundant system S7-1500R/H result from the operating states of the primary and backup CPUs. The term system state is used as a simplified expression that refers to the operating states that occur simultaneously on both CPUs. The redundant system S7-1500R/H has the system states STOP, STARTUP, RUN-Solo, SYNCUP and RUN-Redundant.

TCP/IP

Transmission **C**ontrol **P**rotocol / **I**nternet **P**rotocol, connection-oriented network protocol, generally recognized standard for data exchange in heterogeneous networks.

Time-of-day synchronization

Capability of transferring a standard system time from a single source to all devices in the system so that their clocks can be set according to the standard time.

Tree topology

Network topology characterized by a branched structure: Two or more bus nodes are connected to each bus node.

Twisted-pair

Fast Ethernet via twisted-pair cables is based on the IEEE 802.3u standard (100 Base-TX). The transmission medium is a shielded 2x2 twisted-pair cable with an impedance of 100 Ohms (22 AWG). The transmission characteristics of this cable must meet the requirements of category 5.

The maximum length of the connection between the terminal and the network component must not exceed 100 m. The connectors are designed according to the 100Base-TX standard with the RJ-45 connector system.

UDP

User **D**atagram **P**rotocol; communications protocol for fast and uncomplicated data transfer, without acknowledgment. There are no error checking mechanisms as found in TCP/IP.

User program

In SIMATIC, a distinction is made between the CPU operating system and user programs. The user program contains all instructions, declarations and data by which a system or process can be controlled. The user program is assigned to a programmable module (for example, CPU, FM) and can be structured in smaller units.

USS

Universal Serial Interface protocol (**U**niverselles **S**erielles **S**chnittstellen-Protokoll); defines an access method according to the master-slave principle for communication via a serial bus.

Web server

Software/communications service for data exchange via the Internet. The web server transfers the documents using standardized transmission protocols (HTTP, HTTPS) to a Web browser. Documents can be static or put together dynamically from different sources by the web server on request from the Web browser.

Index

G

H

I

L

M

N

O

P

R

S

T

U

W

X