SIEMENS

Ingenuity for life

Integration of S7-1500 Package Units with SIMATIC PCS 7/OPEN OS

SIMATIC PCS 7 V9.0

https://support.industry.siemens.com/cs/ww/en/view/49740087

Siemens Industry Online Support

Legal information

Table of contents

1 Task

Introduction

System landscapes that have grown over time often consist of heterogeneous automation technologies. With the introduction of the TIA portal, there exists the requirement that new Package Units containing S7-1500 CPUs can also be operated and monitored using a PCS 7 operator system.

SIMATIC PCS 7/OPEN OS is a PCS 7 option with which controllers that does not belong to the range of SIMATIC PCS 7 system components can be integrated into the PCS 7 process control system.

PCS 7/OPEN OS V9.0 enables the data exchange between the PCS 7 operator station and various automation systems via the existing WinCC channels or via the OPC channel. For third-party systems that can only be integrated via the OPC channel, only the appropriate OPC server for the particular controller type is necessary. PCS 7/OPEN OS supports the data exchange with the controllers via Classic OPC DA, Classic OPC A&E and OPC UA DA.

The core of PCS 7/OPEN OS is based on the database automation software (DBA) familiar from other OS options. This software is mainly composed of the following components:

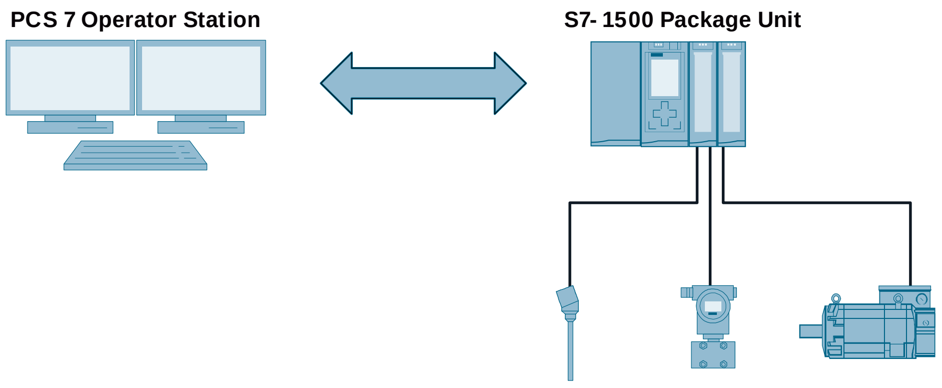
- SIMATIC PCS 7 OS engineering and runtime software
- PCS 7/OPEN OS DBA data base automation software
- PCS 7/OPEN OS runtime software option

The supplied base functionality of PCS 7/OPEN OS allows configuration engineers to effectively integrate the S7-1500 controllers into the process control using the PCS 7 operator system. This gives operators the option of operating and monitoring the entire system from a single operator station, even if it contains package units with S7-1500 controllers.

Overview of the automation task

An S7-1500 package unit is to be operated and monitored using the SIMATIC PCS 7 operator station. The integration of the system into the PCS 7 landscape will take place using SIMATIC PCS 7/OPEN OS via S7-1500 OMS+ Channel.

Figure 1–1



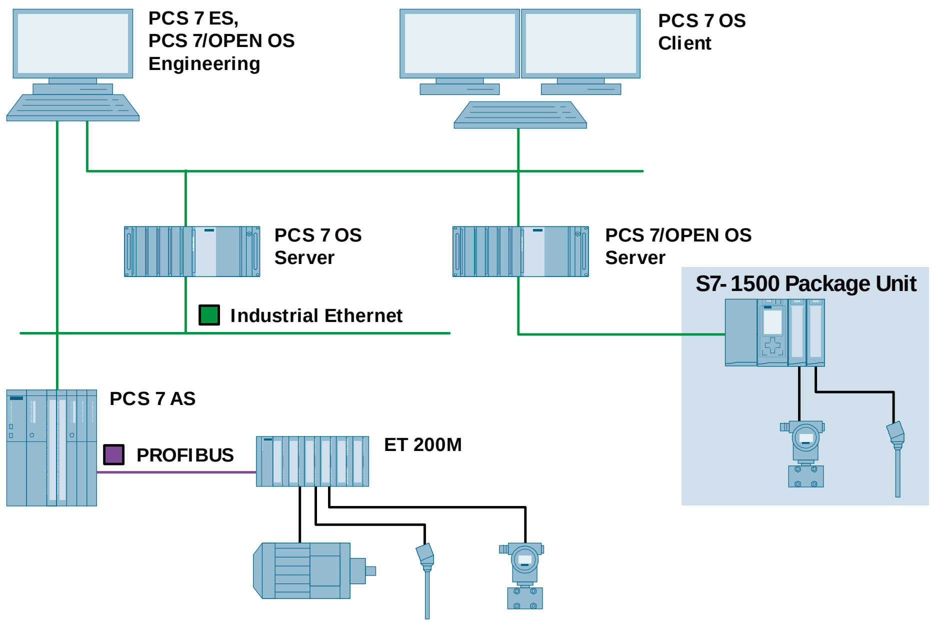
2 Solution

Using the PCS 7 SIMATIC PCS 7/OPEN OS option, this application example will show you how you can effectively integrate S7-1500 controllers that do not belong to the PCS 7 range into the PCS 7 system landscape.

2.1 Overview

The example contained in this document shows you how to integrate an S7-1500 package unit into an existing PCS 7 landscape. The S7-1500 controller is integrated with the aid of the SIMATIC S7-1200 or S7-1500 channel of the PCS 7 operator station.

Figure 2–1



Advantages

The solution presented in this document offers you the following advantages:

- Complete integration of controllers that do not conform to PCS 7 into the PCS 7 operator system
- Step by step instruction for configuration with PCS 7/OPEN OS
- Shared alarm and tag logging management of PCS 7 and third-party systems on one operator system

Required knowledge

The following basic knowledge is required:

- Systems configuration with PCS 7 AS engineering
- Generation of a visualization with PCS 7 OS engineering

2.2 Description of the core functionality

A core component for the PCS 7/OPEN OS engineering is the PCS 7/OPEN OS DBA database automation software.

The DBA generates OS data such as:

- Plant hierarchy
- Tags and archive tags
- Connections
- Alarms and messages

The DBA can use the channels available in the PCS 7 OS to connect the systems that are not PCS 7-conform. For example, drivers are integrated in WinCC for the following third-party systems:

- SIMATIC S7-1200, S7-1500
- SIMATIC 505
- Allen Bradley
- Mitsubishi
- …

Third-party systems for which no special connections exist can use the OPC or Modbus TCP open standards.

3 Basic principles

3.1 Configuration guidelines

You will find here an overview of the steps necessary for integrating a third-party system into the PCS 7 operator station using PCS 7/OPEN OS.

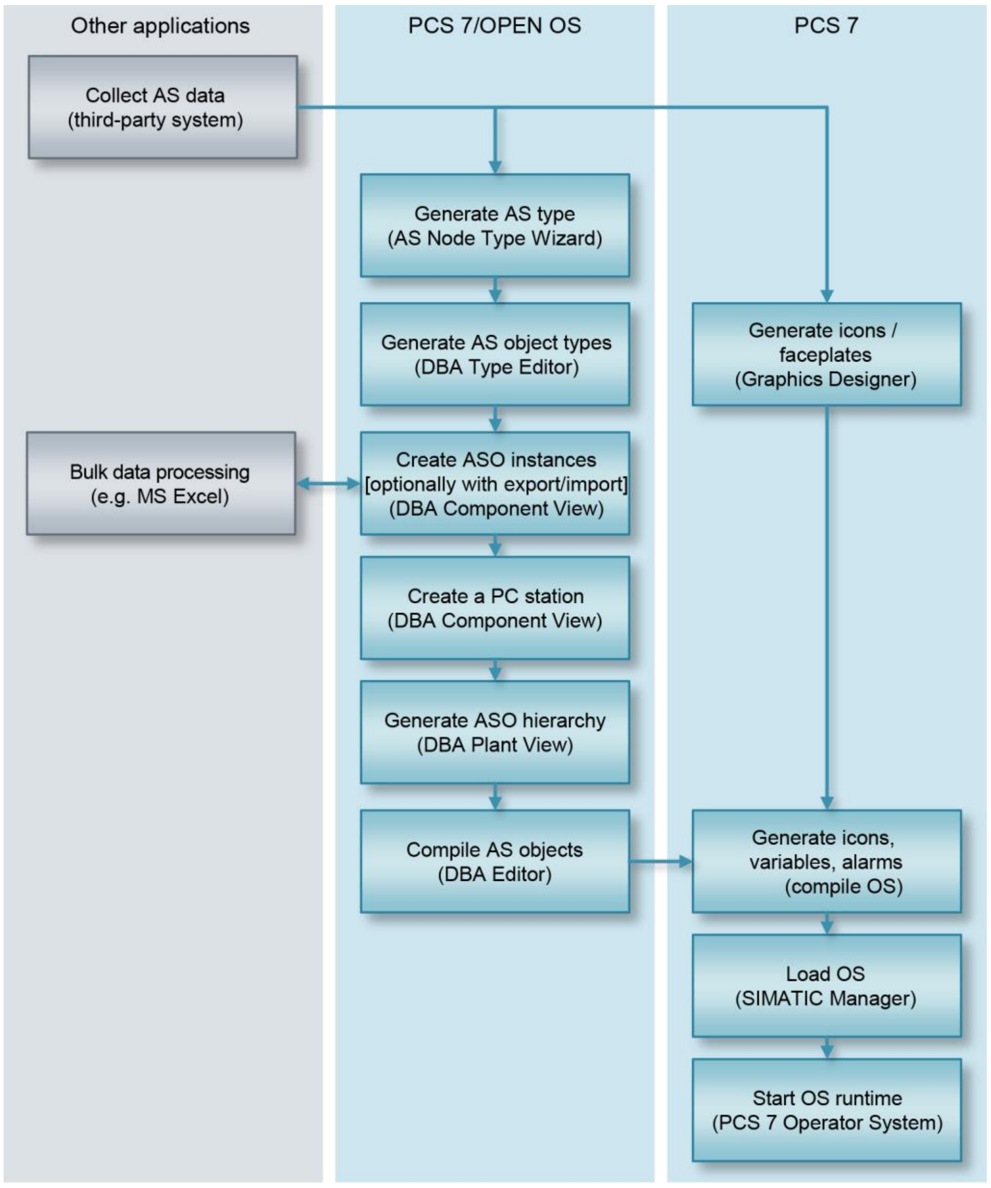
Table 3–1

No.	Description
1.	**Collecting data** Before you start configuring, you will need the following information in order to operate and monitor the package unit: • Access type (S7 connection, OPC DA, OPC A&E, ...) • Network addresses (OPC server, automation systems) • Syntax of tag addresses • Tags, interfaces, alarms • Project file path of the target operator system Plant hierarchy
2.	**Generating OS block icons and faceplates** Using the available knowledge of the structure of the objects (type of objects and related tags) in the AS, you can generate the block icons and faceplates. In conformance with PCS 7, the icons are stored in a Typicals file and the individual views of the faceplates are generated. Further information on the generation of icons and faceplates can be found in the "SIMATIC PCS 7 Process Control System APL Style Guide" configuration instructions in the following article: https://support.industry.siemens.com/cs/ww/en/view/65601446 **Note** It is imperative that this work be completed before the initial compilation of the plant hierarchy.
3.	**Generating the AS node type** You will generate the AS node type using the AS Node Type Wizard. The Wizard already contains information about the channel used and the corresponding connections as well as the syntax of the tag addresses.
4.	**Creating the AS node** In the AS View of the DBA, you will generate an instance of the previously generated AS node type. Depending on the setting, you may still be able to adjust the connection parameters and assign an instance name in this view.
5.	**Generating AS object types** The AS objects (ASO) represent the objects present in the automation system, e.g. motors, valves, closed-loop controllers, analog and digital values, and are connected to the AS node type. The ASO types can contain tag addresses, the tag format, tag attributes, and tag alarms and messages.

No.	Description
6.	**Creating ASO instances** Create the ASO instances. These instances are always assigned to an AS node.
7.	**Creating a PC station** In the PC Station View of the DBA, you can define the OS project to which the data will be compiled. Here, you will set the project path of the target system and also specify a data log if one exists.
8.	**Specifying the plant hierarchy** An existing plant hierarchy can be read out from the SIMATIC project and synchronized with the DBA project. In the Plant View of the DBA, you can then create additional hierarchy folders and subsequently synchronize them with the SIMATIC project again.
9.	**Assigning ASO instances to the plant hierarchy** Drag the ASO instances on to the corresponding hierarchy folder in the plant view to establish the relationship of the AS objects to the operator system. Depending on the mode of configuration, you may still be able to assign or adjust tag addresses and attributes.
10.	**Compiling the DBA project** This process roughly corresponds to that of the OS compilation from the SIMATIC Manager. Here, the necessary connections, tags, messages and icons are created in the OS project.
11.	**Loading the OS and starting the runtime** If all sequences have been executed properly, the OS project can be loaded and the OS runtime can be started.

The following flow diagram shows the procedure for configuring third-party systems using SIMATIC PCS 7/OPEN OS in abbreviated form.

Figure 3–1



Other applications
PCS 7/OPEN OS
PCS 7
Collect AS data (third-party system)
Generate AS type (AS Node Type Wizard)
Generate AS object types (DBA Type Editor)
Generate icons / faceplates (Graphics Designer)
Bulk data processing (e.g. MS Excel)
Create ASO instances [optionally with export/import] (DBA Component View)
Create a PC station (DBA Component View)
Generate ASO hierarchy (DBA Plant View)
Compile AS objects (DBA Editor)
Generate icons, variables, alarms (compile OS)
Load OS (SIMATIC Manager)
Start OS runtime (PCS 7 Operator System)

3.2 Editors

PCS 7/OPEN OS includes the following editors, which are briefly described below:

- PCS 7 DBA AS Node Type Wizard
- PCS 7 DBA
- PCS 7 DBA Type Editor

3.2.1 PCS 7 DBA AS Node Type Wizard

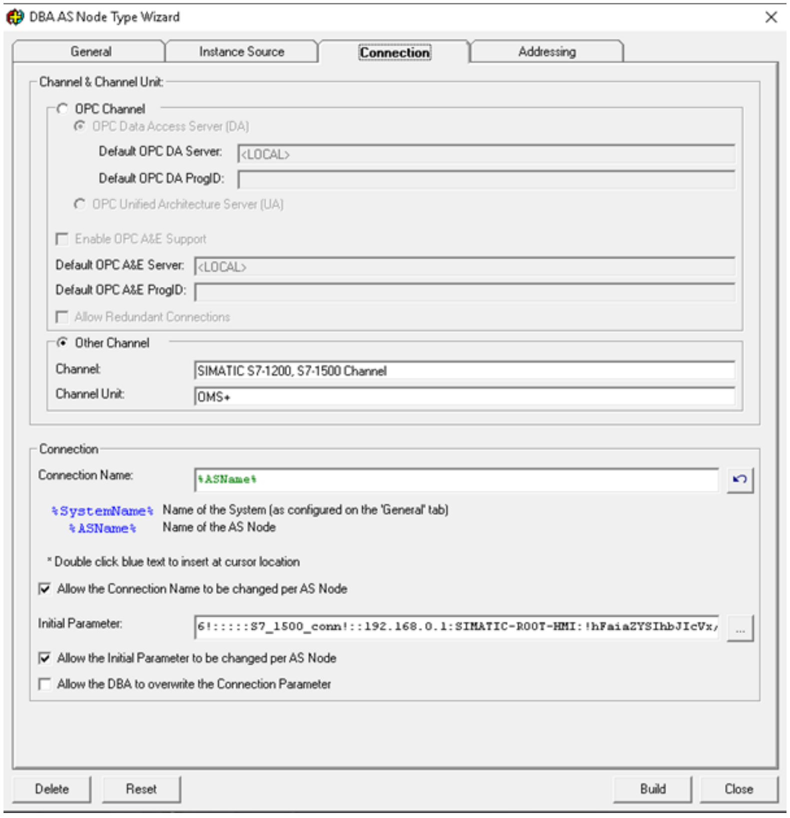
You will generate the AS node types using the AS Node Type Wizard.
Here, the following information will be stored:

- AS name
- Typical display
- Instance source (DBA or XML)
- Connection parameters
- Address syntax

You can start the AS Node Type Wizard at "Start > Siemens Automation > AS Node Wizard".

You will find detailed information in the "PCS 7 Open OS Engineering Workflow Guide" manual. The manuals are copied to your system during the OPEN OS installation.

Figure 3–2



Channel & Channel Unit:

OPC Channel

OPC Data Access Server (DA)

Default OPC DA Server: <LOCAL>

Default OPC DA ProgID:

OPC Unified Architecture Server (UA)

☐ Enable OPC A&E Support

Default OPC A&E Server: <LOCAL>

Default OPC A&E ProgID:

☐ Allow Redundant Connections

● Other Channel

Channel: SIMATIC S7-1200, S7-1500 Channel

Channel Unit: OMS+

Connection

Connection Name: %ASName%

%SystemName%	Name of the System (as configured on the 'General' tab)
%ASName%	Name of the AS Node

* Double click blue text to insert at cursor location

✓ Allow the Connection Name to be changed per AS Node

Initial Parameter: 6!:::::S7_1500_conn!::192.168.0.1:SIMATIC-ROOT-HMI:!hFaiaZYSIhbJIcVx/

✓ Allow the Initial Parameter to be changed per AS Node

☐ Allow the DBA to overwrite the Connection Parameter

3.2.2 PCS 7 DBA

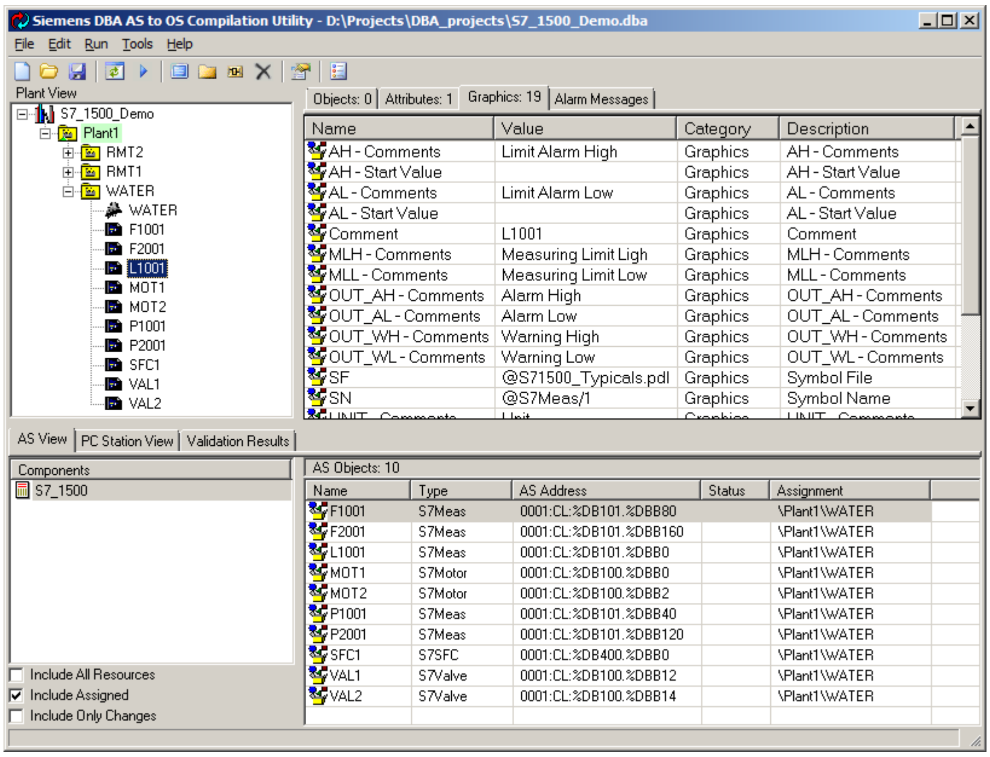
The DBA Editor is the heart of PCS 7/OPEN OS. This is where you configure the AS structures and connect them to the PCS 7 operator system. Here, you must carry out the following tasks:

- Create the AS node and AS object instances (ASO's)
- Generate the plant hierarchy and assign ASO instances
- Compile the DBA project

You can start the DBA at "Start > Siemens Automation > PCS 7 DBA".

You will find detailed information in the "PCS 7 Open OS DBA" manual. The manuals are copied to your system during the OPEN OS installation.

Figure 3–3



- S7_1500_Demo
  - Plant1
    - RMT2
    - RMT1
    - WATER
      - WATER
      - F1001
      - F2001
      - L1001
      - MOT1
      - MOT2
      - P1001
      - P2001
      - SFC1
      - VAL1
      - VAL2

Name	Value	Category	Description
AH - Comments	Limit Alarm High	Graphics	AH - Comments
AH - Start Value		Graphics	AH - Start Value
AL - Comments	Limit Alarm Low	Graphics	AL - Comments
AL - Start Value		Graphics	AL - Start Value
Comment	L1001	Graphics	Comment
MLH - Comments	Measuring Limit Ligh	Graphics	MLH - Comments
MLL - Comments	Measuring Limit Low	Graphics	MLL - Comments
OUT_AH - Comments	Alarm High	Graphics	OUT_AH - Comments
OUT_AL - Comments	Alarm Low	Graphics	OUT_AL - Comments
OUT_WH - Comments	Warning High	Graphics	OUT_WH - Comments
OUT_WL - Comments	Warning Low	Graphics	OUT_WL - Comments
SF	@S71500_Typicals.pdl	Graphics	Symbol File
SN	@S7Meas/1	Graphics	Symbol Name

Name	Type	AS Address	Status	Assignment
F1001	S7Meas	0001:CL:%DB101.%DBB80		\Plant1\WATER
F2001	S7Meas	0001:CL:%DB101.%DBB160		\Plant1\WATER
L1001	S7Meas	0001:CL:%DB101.%DBB0		\Plant1\WATER
MOT1	S7Motor	0001:CL:%DB100.%DBB0		\Plant1\WATER
MOT2	S7Motor	0001:CL:%DB100.%DBB2		\Plant1\WATER
P1001	S7Meas	0001:CL:%DB101.%DBB40		\Plant1\WATER
P2001	S7Meas	0001:CL:%DB101.%DBB120		\Plant1\WATER
SFC1	S7SFC	0001:CL:%DB400.%DBB0		\Plant1\WATER
VAL1	S7Valve	0001:CL:%DB100.%DBB12		\Plant1\WATER
VAL2	S7Valve	0001:CL:%DB100.%DBB14		\Plant1\WATER

3.2.3 PCS 7 DBA Type Editor

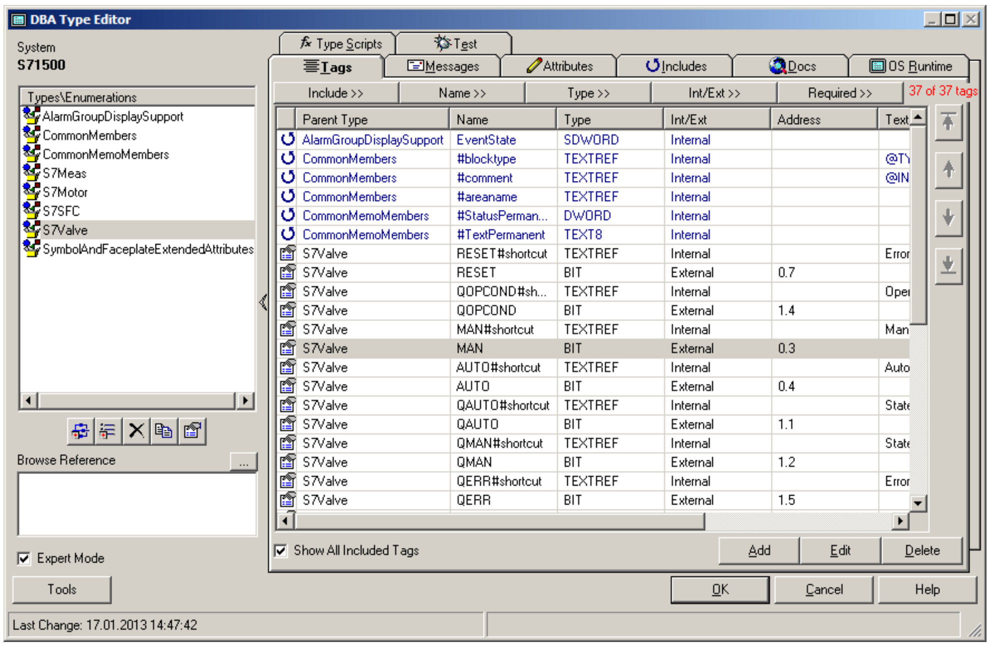
Using the DBA Type Editor, you can generate the AS objects, which map the structure of the blocks in the third-party system. They contain:

- Tags, possibly with addresses
- Enumerations
- Alarms and messages
- Attributes
- Runtime scripts

You can start the Editor via the shortcut menu of an AS node in the DBA by selecting the "Edit AS Object Types..." item.

You will find detailed information in the "PCS 7 Open OS DBA Type Editor" manual. The manuals are copied to your system during the OPEN OS installation.

Figure 3–4

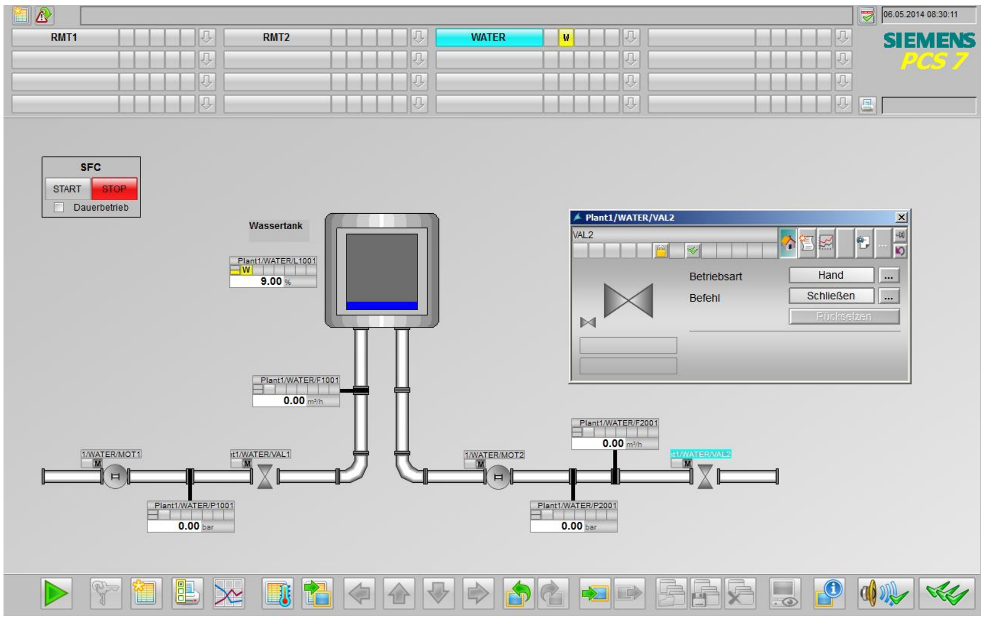


Parent Type	Name	Type	Int/Ext	Address	Text
AlarmGroupDisplaySupport	EventState	SDWORD	Internal		
CommonMembers	#blocktype	TEXTREF	Internal		@TY
CommonMembers	#comment	TEXTREF	Internal		@IN
CommonMembers	#areaname	TEXTREF	Internal		
CommonMemoMembers	#StatusPerman...	DWORD	Internal		
CommonMemoMembers	#TextPermanent	TEXT8	Internal		
S7Valve	RESET#shortcut	TEXTREF	Internal		Error
S7Valve	RESET	BIT	External	0.7	
S7Valve	QOPCOND#sh...	TEXTREF	Internal		Oper
S7Valve	QOPCOND	BIT	External	1.4	
S7Valve	MAN#shortcut	TEXTREF	Internal		Man
S7Valve	MAN	BIT	External	0.3	
S7Valve	AUTO#shortcut	TEXTREF	Internal		Auto
S7Valve	AUTO	BIT	External	0.4	
S7Valve	QAUTO#shortcut	TEXTREF	Internal		State
S7Valve	QAUTO	BIT	External	1.1	
S7Valve	QMAN#shortcut	TEXTREF	Internal		State
S7Valve	QMAN	BIT	External	1.2	
S7Valve	QERR#shortcut	TEXTREF	Internal		Error
S7Valve	QERR	BIT	External	1.5	

4 Configuring an S7-1500 package unit

The following chapters describe how to integrate an S7-1500 PU into PCS 7 using OPEN OS. The SIMATIC project used in the example is provided solely for purposes of demonstration. However, the basic procedure for real package units with S7-1500 controllers is largely identical.

Figure 4–1



In order to purge a pipe system, a tank is filled with water and the water is then fed into the pipe system. A maximum level and a minimum level of the tank can be predetermined for the purging sequence.

The sequence places the valves and motors in automatic mode and controls them until the purging process is complete. The actuators are then switched into manual mode again.

4.1 Acquiring data

This example project contains the following components, which are relevant for operation and monitoring at the OS:

Table 4–1

Type	Name	AS address	Description
Motor	MOT1	DB100, DBB0	Pump drive for filling the tank
Valve	VAL1	DB100, DBB12	Tank intake valve
Analog	P1001	DB101, DBB40	Tank intake pressure transducer
Analog	F1001	DB101, DBB80	Tank intake flow sensor
Motor	MOT2	DB100, DBB2	Pump drive for emptying the tank
Valve	VAL2	DB100, DBB14	Tank drain valve
Analog	P2001	DB101, DBB120	Tank drain pressure transducer
Analog	F2001	DB101, DBB160	Tank drain flow sensor
Analog	L1001	DB101, DBB0	Tank level
Sequenc e	SFC1	DB400, DBB0	Sequence for purging the pipelines

The object types motor, valve and analog value are always based on the same function block. This makes it possible to derive a structure from the instance DB of a block.

Motor block

The following motor block variables are required for configuring the OS:

Table 4–2

Tag	Type	DB Offset	Type	Description
MAN_ON	BOOL	0.0	IN	Start motor manually
MAN_OFF	BOOL	0.1	IN	Stop motor manually
AUTO_ON	BOOL	0.2	IN	Start motor in automatic mode
MAN	BOOL	0.3	IN	Manual mode
AUTO	BOOL	0.4	IN	Automatic mode
FB	BOOL	0.5	IN	Motor feedback signal
RESET	BOOL	0.6	IN	Reset of a feedback error
QSTART	BOOL	0.7	OUT	Signal for starting the motor
QAUTO	BOOL	1.0	OUT	Automatic mode active
QMAN	BOOL	1.1	OUT	Manual mode active
QMOTRUN	BOOL	1.2	OUT	Motor running
QOPCOND	BOOL	1.3	OUT	Operating condition
QERR	BOOL	1.4	OUT	Motor feedback error

The time limit for the block to report a feedback error is permanently set to 5 seconds.

Valve block

The following valve block variables are required for configuring the OS:

Table 4–3

Tag	Type	DB Offset	Type	Description
MAN_OPEN	BOOL	0.0	IN	Open valve manually
MAN_CLOSE	BOOL	0.1	IN	Close valve manually
AUTO_OPEN	BOOL	0.2	IN	Open valve in automatic mode
MAN	BOOL	0.3	IN	Manual mode
AUTO	BOOL	0.4	IN	Automatic mode
FB_OPEN	BOOL	0.5	IN	Open valve feedback signal
FB_CLOSE	BOOL	0.6	IN	Close valve feedback signal
RESET	BOOL	0.7	IN	Reset of a feedback error
QOPEN	BOOL	1.0	OUT	Signal for opening the valve
QAUTO	BOOL	1.1	OUT	Automatic mode active
QMAN	BOOL	1.2	OUT	Manual mode active
QVALOPEN	BOOL	1.3	OUT	Valve open
QOPCOND	BOOL	1.4	OUT	Operating condition
QERR	BOOL	1.5	OUT	Valve feedback error

The time limit for the block to report a feedback error is permanently set to 5 seconds.

Analog value monitoring

The following analog block variables are required for configuring the OS:

Table 4–4

Tag	Type	DB Offset	Type	Description
VALUE	REAL	0.0	IN	Process value
AL	REAL	4.0	IN	Lower alarm limit
WL	REAL	8.0	IN	Lower warning limit
WH	REAL	12.0	IN	Upper warning limit
AH	REAL	16.0	IN	Upper alarm limit
OUT_AL	BOOL	20.0	OUT	Lower alarm limit reached
OUT_WL	BOOL	20.1	OUT	Lower warning limit reached
OUT_WH	BOOL	20.2	OUT	Upper warning limit reached
OUT_AH	BOOL	20.3	OUT	Upper alarm limit reached
MLH	REAL	22.0	IN	Maximum value
MLL	REAL	26.0	IN	Minimum value
UNIT	STRING [8]	30.0	IN	Process value unit

Block sequence

The following block sequence variables are required for configuring the OS:

Table 4–5

Tag	Type	DB address	Type	Description
START	BOOL	0.0	IN	The sequence is started.
STOP	BOOL	0.1	IN	The sequence is stopped.
CYCLE	BOOL	0.3	IN	The sequence runs in a continuous loop

4.2 Generating OS block icons and faceplates

You can generate OS block icons and faceplates with the aid of the WinCC Graphics Designer. The process conforms to the PCS 7 standard. For this reason, the topic is not discussed in greater detail in this documentation. Detailed information on the generation of user-defined icons and faceplates can be found in the "SIMATIC Process Control System PCS 7 APL Style Guide" manual in the following article:

https://support.industry.siemens.com/cs/ww/en/view/65601446

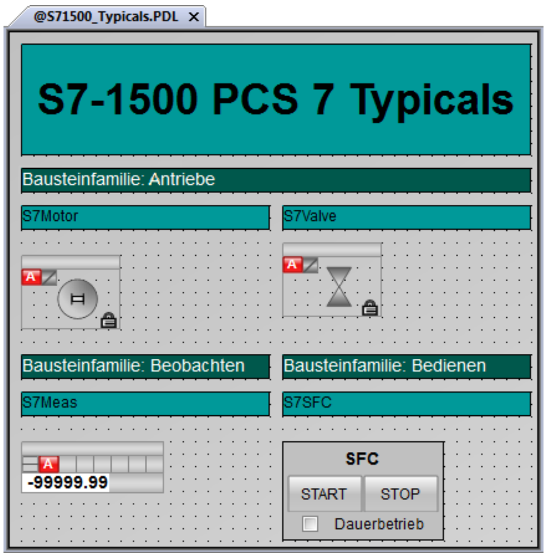
Use the collected data to generate the block icons and faceplates. Taking into consideration the naming conventions for Typicals pictures, use a meaningful name for the icon picture, e.g. "@S71500_Typicals.pdl".

Note When creating the block icons, please bear in mind the correct labeling for the object name. E.g., for the "S7MEAS" block, the object name for the associated block icon is "S7MEAS/1".

In the present example, these are the blocks for:

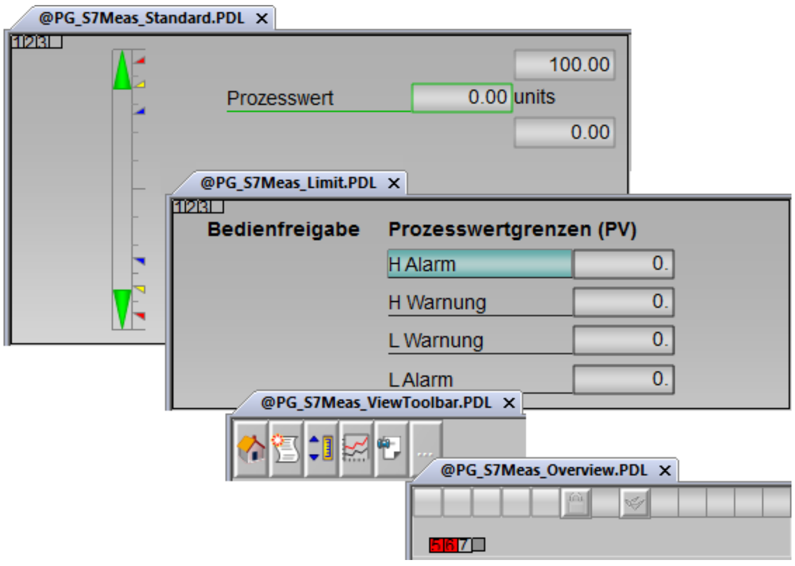
- Motor
- Valve
- Analog value
- SFC (no faceplates)

Figure 4–2



One faceplate set each was created for the motor, valve and analog value icons. The block icon for the sequence does not have a faceplate.

Figure 4–3



4.3 Generating the AS node with the AS Node Type Wizard

The AS node type is generated with the aid of the PCS 7 DBA AS Node Type Wizard. You can find the editor in the Start menu at: "Start > Siemens Automation > AS Node Wizard".

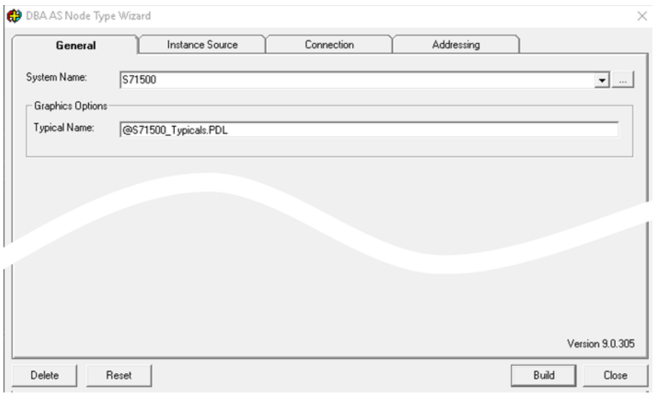
The AS node type is not part of a specific DBA project and is therefore stored in the installation directory of OPEN OS. You can use all the created AS node types in each of your DBA projects. The following sections explain how to create an AS node type.

4.3.1 General tab

In the "General" tab, you assign the name of the AS node type and define the name of the typical picture from which the OS block icons are copied during compilation.

1. Enter the name for the AS node type.
2. Enter the name of the WinCC screen that contains the generated picture symbols.

Figure 4–4



System Name: S71500

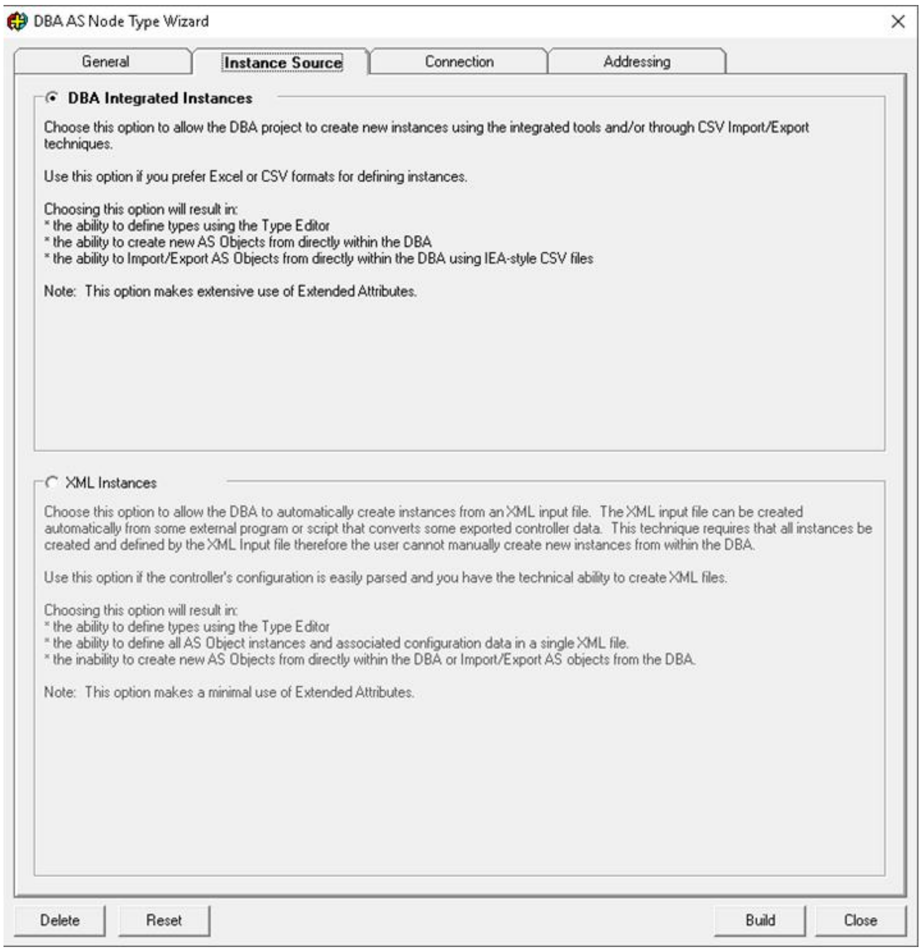
Typical Name: @S71500_Typicals.PDL

4.3.2 Instance Source tab

This document only describes the DBA Integrated Instances option.

1. Select the **DBA Integrated Instances** option.

Figure 4–5



(● DBA Integrated Instances)

Choose this option to allow the DBA project to create new instances using the integrated tools and/or through CSV Import/Export techniques.

Use this option if you prefer Excel or CSV formats for defining instances.

Choosing this option will result in:
* the ability to define types using the Type Editor
* the ability to create new AS Objects from directly within the DBA
* the ability to Import/Export AS Objects from directly within the DBA using IEA-style CSV files

Note: This option makes extensive use of Extended Attributes.

(○ XML Instances)

Choose this option to allow the DBA to automatically create instances from an XML input file. The XML input file can be created automatically from some external program or script that converts some exported controller data. This technique requires that all instances be created and defined by the XML Input file therefore the user cannot manually create new instances from within the DBA.

Use this option if the controller's configuration is easily parsed and you have the technical ability to create XML files.

Choosing this option will result in:
* the ability to define types using the Type Editor
* the ability to define all AS Object instances and associated configuration data in a single XML file.
* the inability to create new AS Objects from directly within the DBA or Import/Export AS objects from the DBA.

Note: This option makes a minimal use of Extended Attributes.

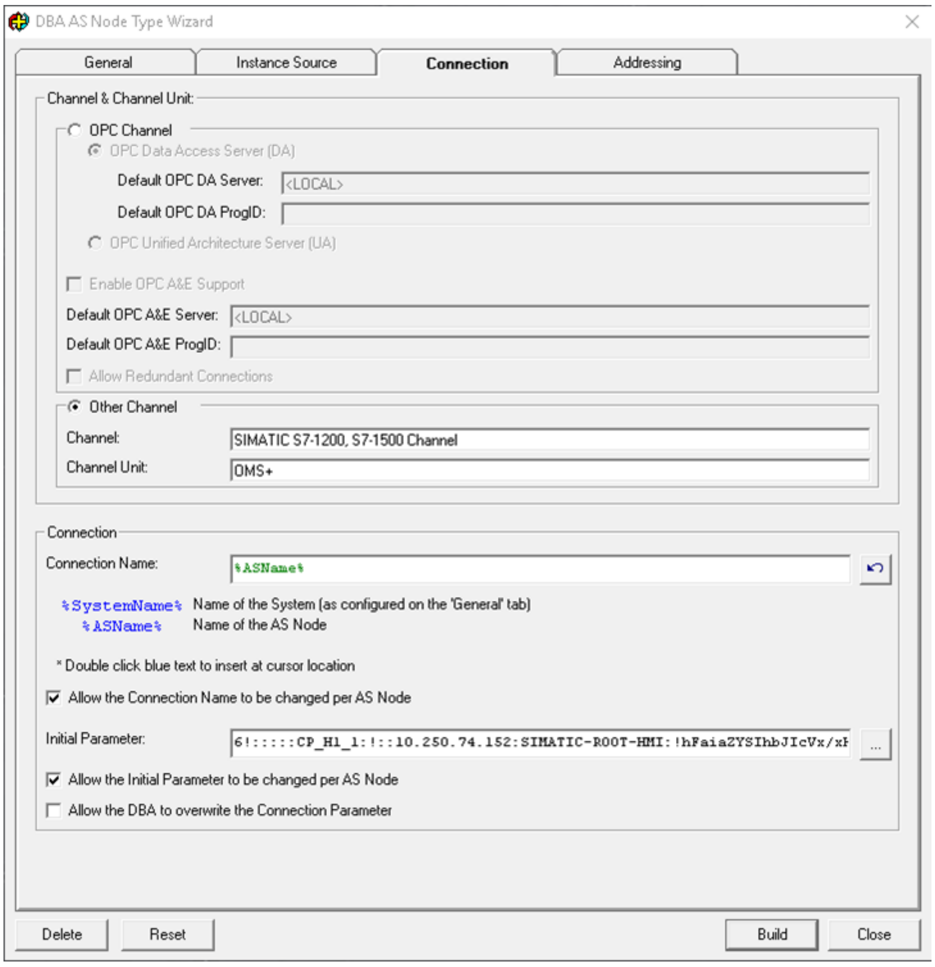
You can find a description of the use of XML instances in the following article: "How do you create an XML input file for automatic creation of instances in DBA?" (Entry ID: 78032500).

4.3.3 "Connection" tab

Because the S7-1500 is a SIMATIC product, using the compatible SIMATIC S7-1200 or S7-1500 channel is a logical choice.

1. Select the **Other Channel** option.
2. Enter the channel and the channel unit that are used.
3. Enter a permanent connection name or use the following possible tags:
   - %SystemName%: The name of the AS node type is used (General tab).
   - %ASName%: The name of the AS node is used that is entered in the DBA.
4. Enter the connection string used by WinCC as the initial parameter.

Figure 4–6



Channel & Channel Unit:

( ) OPC Channel

(•) OPC Data Access Server (DA)

Default OPC DA Server: <LOCAL>

Default OPC DA ProgID:

( ) OPC Unified Architecture Server (UA)

☐ Enable OPC A&E Support

Default OPC A&E Server: <LOCAL>

Default OPC A&E ProgID:

☐ Allow Redundant Connections

(•) Other Channel

Channel: SIMATIC S7-1200, S7-1500 Channel

Channel Unit: OMS+

Connection

Connection Name: %ASName%

%SystemName% Name of the System (as configured on the 'General' tab)

%ASName% Name of the AS Node

* Double click blue text to insert at cursor location

☑ Allow the Connection Name to be changed per AS Node

Initial Parameter: 6!:::::CP_H1_1:!:::10.250.74.152:SIMATIC-ROOT-HMI:!hFaiaZYSIhbJIcVx/xF

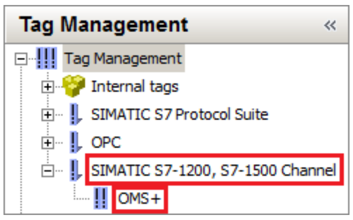
☑ Allow the Initial Parameter to be changed per AS Node

☐ Allow the DBA to overwrite the Connection Parameter

The text in the "Channel" and "Channel Unit" fields must be identical to the corresponding WinCC channel. You can find the necessary inputs in the WinCC Tag Management. Follow the steps below:

5. Open an existing or new OS project.
6. Open the WinCC Tag Management.
7. Click on "Tag management" with the right hand mouse button and add the new "SIMATIC S7-1200, S7-1500 Channel" driver.

Figure 4–7

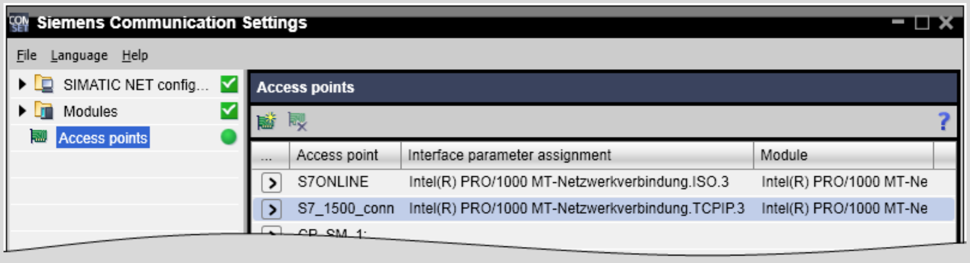


The "initial parameter" corresponds to the string that WinCC creates for a connection. If you do not know the exact string syntax, you can read the parameter out of WinCC. Proceed as follows:

8. Open an existing or new OS project.
9. Open the WinCC Tag Management.
10. In the desired channel unit, temporarily create a new connection with the desired connection parameters (it can be deleted later).

Note

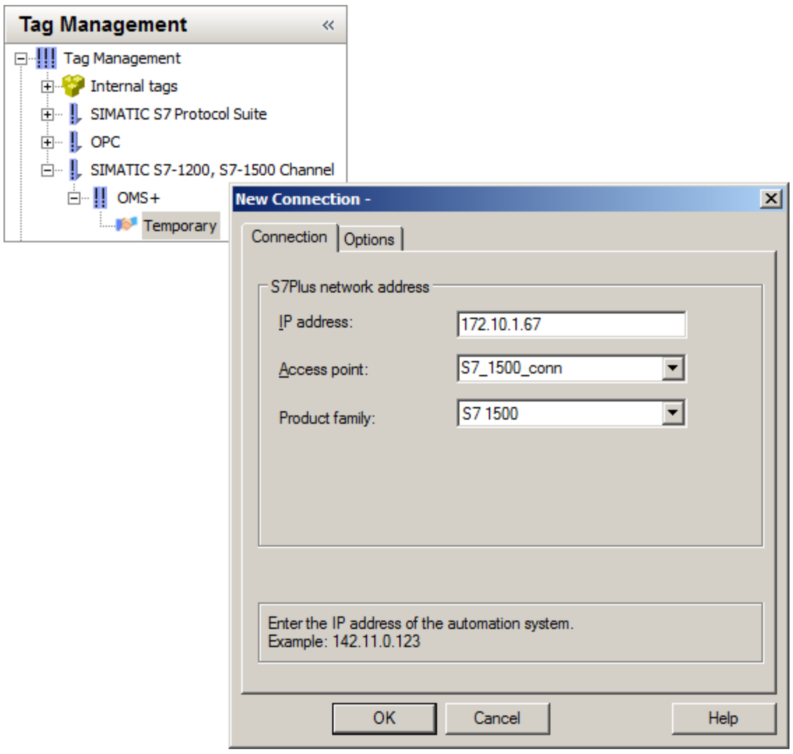
Since the existing access points might be used by other channels, it is advisable to create a new access point for the TCP/IP protocol for the S7-1500 channel in the SIMATIC NET "Communications settings" editor.



...	Access point	Interface parameter assignment	Module
>	S7ONLINE	Intel(R) PRO/1000 MT-Netzwerkverbindung.ISO.3	Intel(R) PRO/1000 MT-Ne
>	S7_1500_conn	Intel(R) PRO/1000 MT-Netzwerkverbindung.TCPIP.3	Intel(R) PRO/1000 MT-Ne

11. Select the created access point in the "New Connection" window.

Figure 4–8



S7Plus network address

IP address: 172.10.1.67

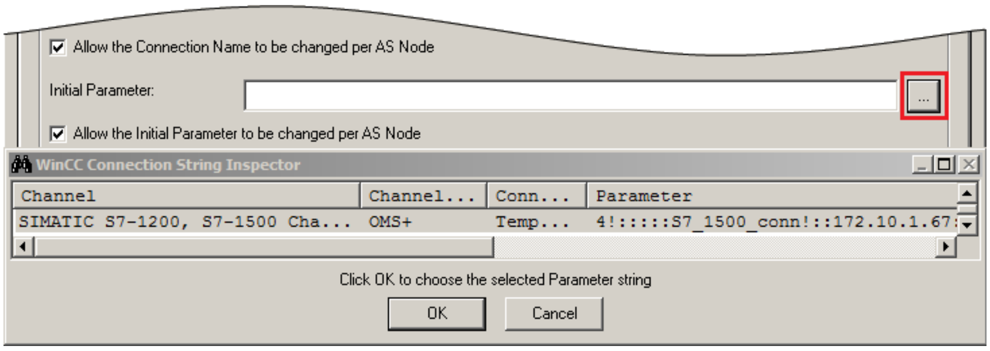
Access point: S7_1500_conn

Product family: S7 1500

Enter the IP address of the automation system.
Example: 142.11.0.123

12. Use the option button next to the text field for the initial parameter to start the WinCC Connection String Inspector. It is vital that the WinCC project with the corresponding connections is open. Otherwise, no connections will be displayed.

Figure 4–9



☑ Allow the Connection Name to be changed per AS Node

Initial Parameter:

☑ Allow the Initial Parameter to be changed per AS Node

Channel	Channel...	Conn...	Parameter
SIMATIC S7-1200, S7-1500 Cha...	OMS+	Temp...	4!:::::S7_1500_conn!::172.10.1.67:

Click OK to choose the selected Parameter string

13. In the event that multiple connections are available, select the appropriate connection and click OK in the dialog window to confirm the selection.
14. Do not delete the temporary connection yet. You will need it again at a later point to determine the address syntax.

4.3.4 Addressing tab

In the Addressing tab, you can specify the structure of the AS and variable addressing that will be used when the project is compiled at a later point.
The address structure depends on the communications driver that is set.
Here, the DBA offers the option of assigning parameters for all conceivable constellations. These range from a structured approach and the free assignment of addresses for all tags, to the use of scripts that can calculate the addresses individually.

The following tags are available for the assignment of ASO address parameters:

- %ASName% – name of the AS type
- %ASOName% – name of the AS instance
- %Instance% – assigned during instantiation
- %Attribute:Name% – value of the specified attribute
- %UID% – unique identifier generated by the DBA

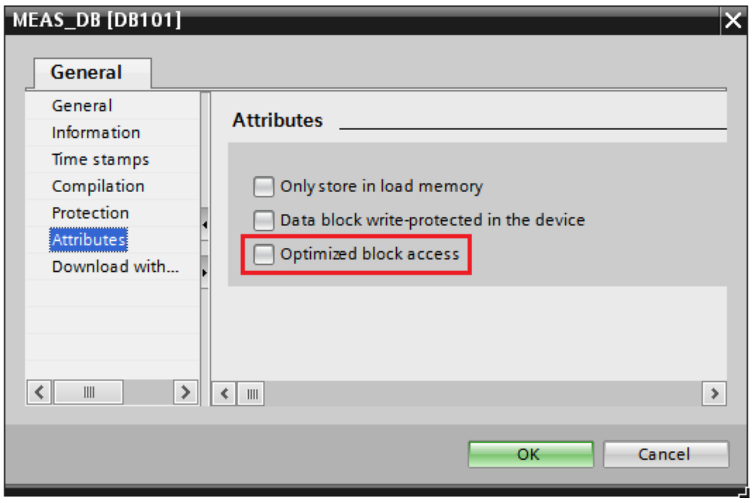
The following tags are available for the assignment of variable address parameters:

- %ASName% – name of the AS type
- %ASOName% – name of the AS instance
- %Instance% – assigned during instantiation
- %Tagname% – name of the tags for which the address is calculated
- %Address% – value of the address field of the tags
- %Field#TagAddress% – value of the extended attribute. Input during parameter assignment
- %Attribute:Name% – value of the extended attribute with the name "Name"
- %UID% – unique identifier generated by the DBA
- %OPCDataType% - OPC data type number for the tag type
- %UANamespace% - OPC UA adress namen space
- %UAType% - OPC UA identifier type
- %UAValue% - OPC UA identifier

Special features when addressing an S7-1500 PU

Access by the PCS 7 OS to S7-1500 DBs basically only functions if the "Optimized block access" attribute is deactivated in the TIA project (Figure 4–10). Otherwise it is not possible to recognize which data are in which area of the corresponding DB.

Figure 4–10



Only store in load memory
Data block write-protected in the device
Optimized block access

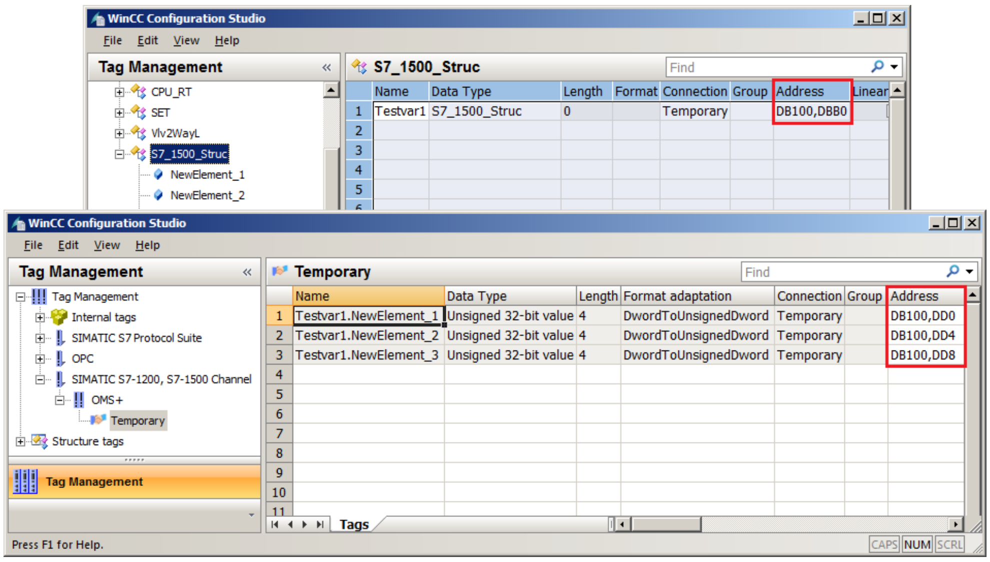
Note also that when an S7-1500 CPU is used, the tag addresses in WinCC and in the SQL database are different. DBA must create the addresses as they appear in the SQL database.

Proceed as follows to read the correct address from the SQL database:

1. Create a temporary connection in the WinCC tag manager or use the temporary connection from chapter "4.3.3 "Connection" tab".
2. Create a new structure with at least one element.
3. Generate an instance of the structure and select the temporary connection.

4. In the "Address" field you can see the syntax of the tag connection in WinCC (Figure 4–11).

Figure 4–11

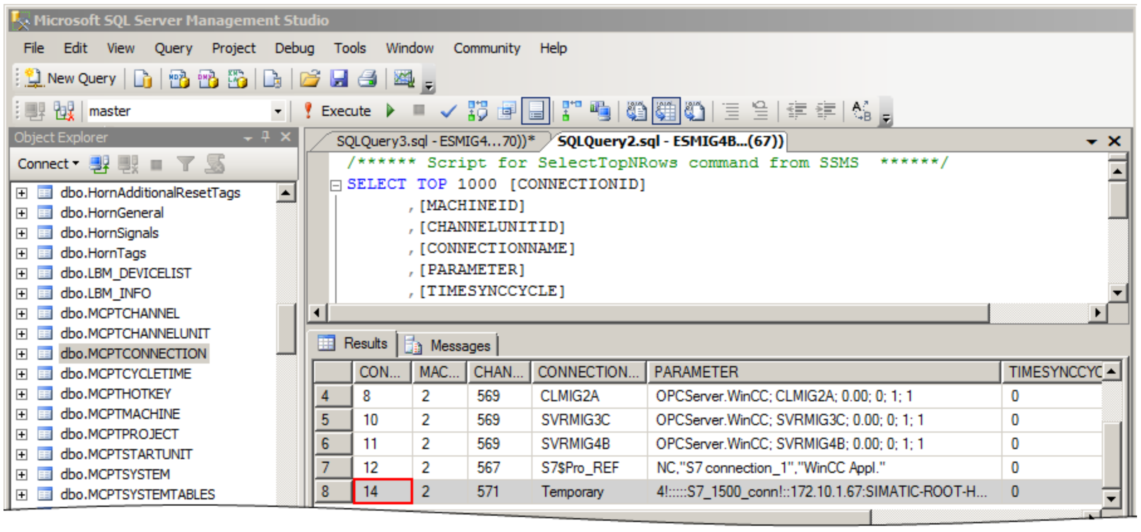


	Name	Data Type	Length	Format	Connection	Group	Address	Linear
1	Testvar1	S7_1500_Struc	0		Temporary		DB100,DBB0	

	Name	Data Type	Length	Format adaptation	Connection	Group	Address
1	Testvar1.NewElement_1	Unsigned 32-bit value	4	DwordToUnsignedDword	Temporary		DB100,DD0
2	Testvar1.NewElement_2	Unsigned 32-bit value	4	DwordToUnsignedDword	Temporary		DB100,DD4
3	Testvar1.NewElement_3	Unsigned 32-bit value	4	DwordToUnsignedDword	Temporary		DB100,DD8

5. Open the "Microsoft SQL Server Management Studio".
6. In your corresponding WinCC project, click on "Select Top 1000 Rows" in the context menu for the "dbo.MCPTCONNECTION" table.
   You can read the "CONNECTIONID" for the S7-1500 connection from the results table (Figure 4–12).

Figure 4–12



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [CONNECTIONID]
      ,[MACHINEID]
      ,[CHANNELUNITID]
      ,[CONNECTIONNAME]
      ,[PARAMETER]
      ,[TIMESYNCCYCLE]
```

	CON...	MAC...	CHAN...	CONNECTION...	PARAMETER	TIMESYNCCYC
4	8	2	569	CLMIG2A	OPCServer.WinCC; CLMIG2A; 0.00; 0; 1; 1	0
5	10	2	569	SVRMIG3C	OPCServer.WinCC; SVRMIG3C; 0.00; 0; 1; 1	0
6	11	2	569	SVRMIG4B	OPCServer.WinCC; SVRMIG4B; 0.00; 0; 1; 1	0
7	12	2	567	S7$Pro_REF	NC,"S7 connection_1","WinCC Appl."	0
8	14	2	571	Temporary	4!:::::S7_1500_conn!::172.10.1.67:SIMATIC-ROOT-H...	0

7. Select the command "Select Top 1000 Rows" from the context menu for the "dbo.MCPTVARIABLEDESC" table.

   At the end of the "SELECT TOP 1000" script, insert the following syntax with the CONNECTIONID just determined in Step 6 (in our example: 14), (Figure 4–13): where CONNECTIONID like '14'

8. When the "Execute" button is clicked, the entries in the results table will be filtered according to the CONNECTIONID and you will be able to see the address of the S7-1500 structure in the "ADDRESSPARAMETER" column (Figure 4–13):

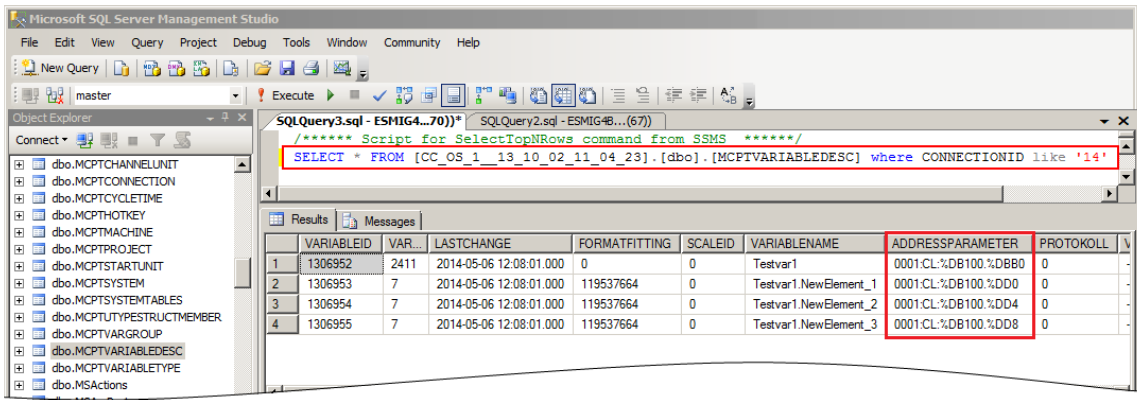
   0001:CL:%DBxxx.%Dxxx

Note

You will need this addressing syntax for any unstructured program design in the S7-1500 PU (see further on in this chapter).

For the structured program design in the S7-1500 PU in this example we will use a script to calculate the correct tag addresses.

Figure 4–13



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT * FROM [CC_OS_1__13_10_02_11_04_23].[dbo].[MCPTVARIABLEDESC] where CONNECTIONID like '14'
```

	VARIABLEID	VAR...	LASTCHANGE	FORMATFITTING	SCALEID	VARIABLENAME	ADDRESSPARAMETER	PROTOKOLL	V
1	1306952	2411	2014-05-06 12:08:01.000	0	0	Testvar1	0001:CL:%DB100.%DBB0	0	-
2	1306953	7	2014-05-06 12:08:01.000	119537664	0	Testvar1.NewElement_1	0001:CL:%DB100.%DD0	0	-
3	1306954	7	2014-05-06 12:08:01.000	119537664	0	Testvar1.NewElement_2	0001:CL:%DB100.%DD4	0	-
4	1306955	7	2014-05-06 12:08:01.000	119537664	0	Testvar1.NewElement_3	0001:CL:%DB100.%DD8	0	-

CAUTION

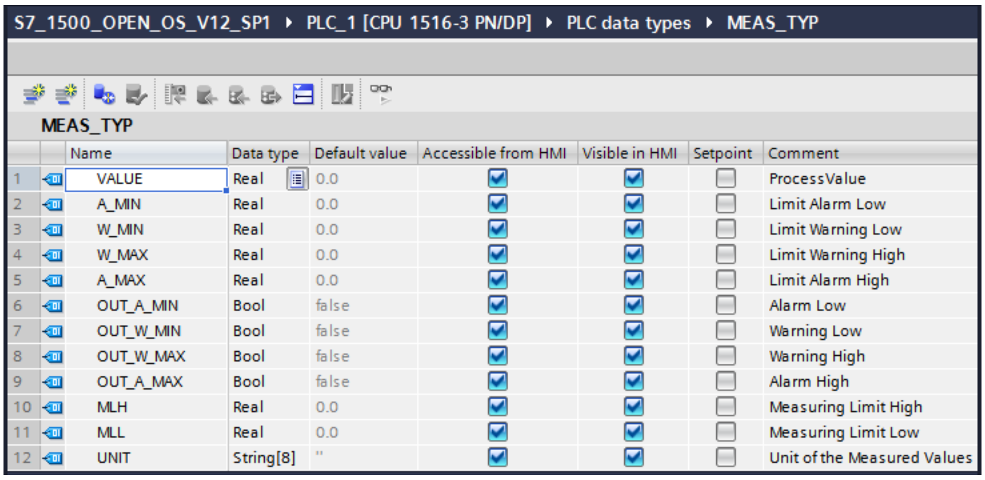
The PCS 7 OS project can be damaged

Only use the SQL database and filtering options to check the tag addresses. Do not make any changes to the database and use only reading operations.

Structured program design in the S7-1500 PU

In this example, the S7-1500 program of the PU is designed in a structured manner. There exist structure objects from which instances which interact with the field have been created. These structure objects are defined as data types (UDT) (Figure 4–14), which communicate with the PCS 7 OS by means of data blocks. These data blocks reproduce the structure of the data types and the addresses of the tags can be precisely calculated from the offset of the structure address (Figure 4–15).

Figure 4–14

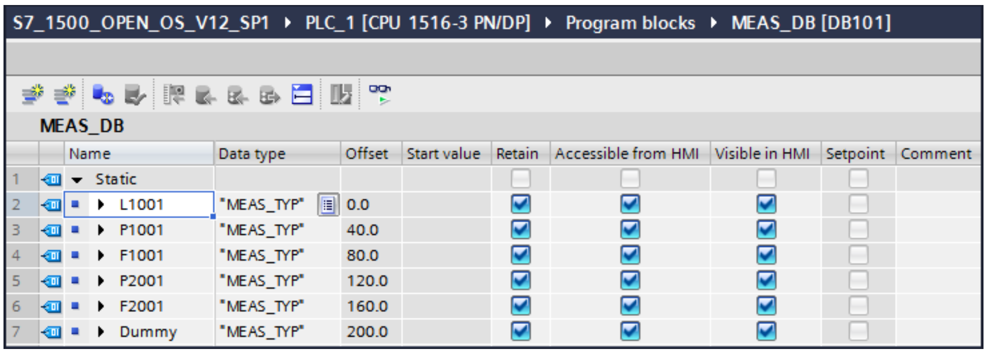


S7_1500_OPEN_OS_V12_SP1 ▸ PLC_1 [CPU 1516-3 PN/DP] ▸ PLC data types ▸ MEAS_TYP

MEAS_TYP

	Name	Data type	Default value	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	VALUE	Real	0.0	✓	✓	☐	ProcessValue
2	A_MIN	Real	0.0	✓	✓	☐	Limit Alarm Low
3	W_MIN	Real	0.0	✓	✓	☐	Limit Warning Low
4	W_MAX	Real	0.0	✓	✓	☐	Limit Warning High
5	A_MAX	Real	0.0	✓	✓	☐	Limit Alarm High
6	OUT_A_MIN	Bool	false	✓	✓	☐	Alarm Low
7	OUT_W_MIN	Bool	false	✓	✓	☐	Warning Low
8	OUT_W_MAX	Bool	false	✓	✓	☐	Warning High
9	OUT_A_MAX	Bool	false	✓	✓	☐	Alarm High
10	MLH	Real	0.0	✓	✓	☐	Measuring Limit High
11	MLL	Real	0.0	✓	✓	☐	Measuring Limit Low
12	UNIT	String[8]	''	✓	✓	☐	Unit of the Measured Values

Figure 4–15



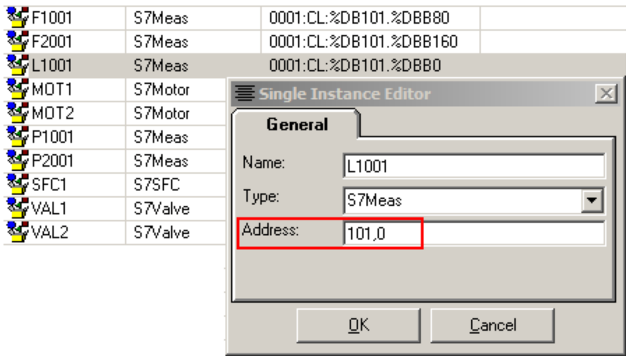
S7_1500_OPEN_OS_V12_SP1 ▸ PLC_1 [CPU 1516-3 PN/DP] ▸ Program blocks ▸ MEAS_DB [DB101]

MEAS_DB

	Name	Data type	Offset	Start value	Retain	Accessible from HMI	Visible in HMI	Setpoint	Comment
1	▾ Static				☐	☐	☐	☐	
2	▸ L1001	"MEAS_TYP"	0.0		✓	✓	✓	☐	
3	▸ P1001	"MEAS_TYP"	40.0		✓	✓	✓	☐	
4	▸ F1001	"MEAS_TYP"	80.0		✓	✓	✓	☐	
5	▸ P2001	"MEAS_TYP"	120.0		✓	✓	✓	☐	
6	▸ F2001	"MEAS_TYP"	160.0		✓	✓	✓	☐	
7	▸ Dummy	"MEAS_TYP"	200.0		✓	✓	✓	☐	

In this case, based on the structure of the object, the offset in the DB and the various data types of the tags, the addresses of the individual tags can be automatically calculated using scripts.
Then only the DB number and the offset of the structure within the DB are entered for the definition of instances (Figure 4‑16).

Figure 4–16



F1001	S7Meas	0001:CL:%DB101.%DBB80
F2001	S7Meas	0001:CL:%DB101.%DBB160
L1001	S7Meas	0001:CL:%DB101.%DBB0
MOT1	S7Motor	
MOT2	S7Motor	
P1001	S7Meas	
P2001	S7Meas	
SFC1	S7SFC	
VAL1	S7Valve	
VAL2	S7Valve	

Single Instance Editor

General

Name: L1001

Type: S7Meas

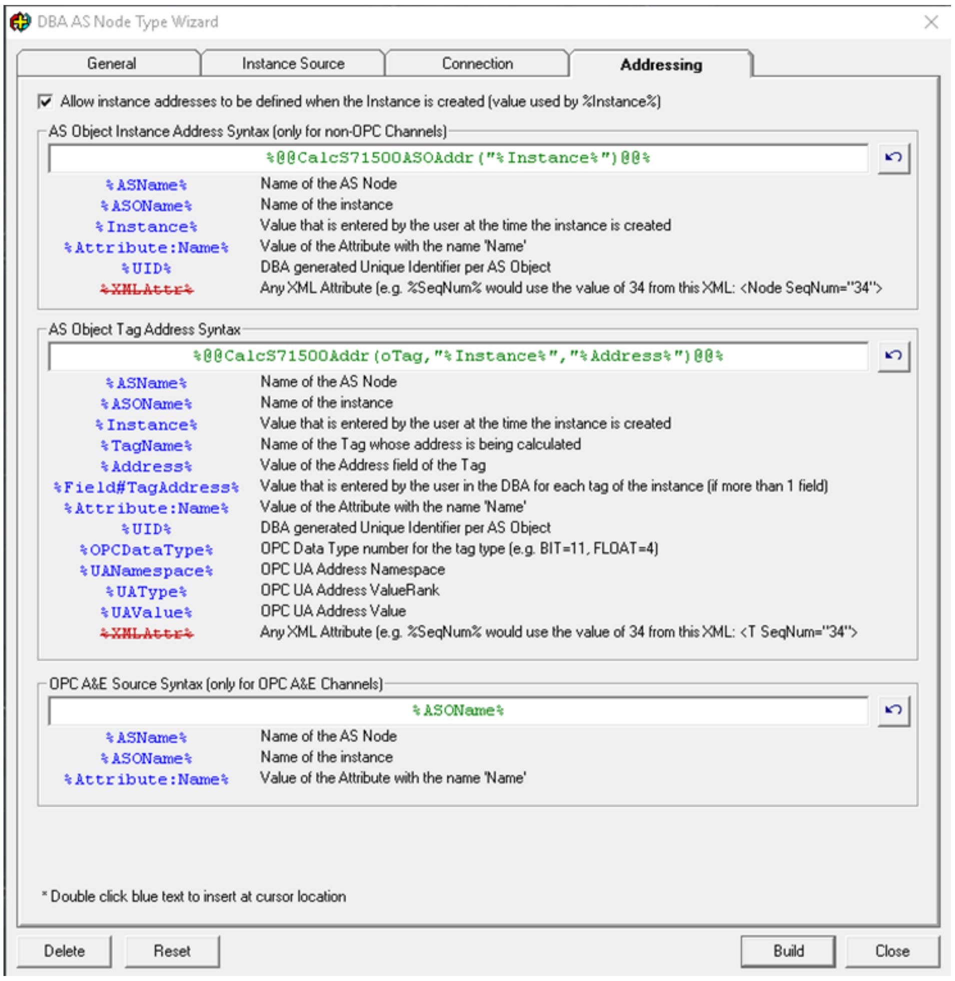
Address: 101,0

The scripts for calculating the tag addresses must be contained in the AS node types in DBA. They are only run for the definition of the ASO instance or during an updating run of the AS objects, and do not appear in the OS runtime.

Note Scripts in this context are a later topic. They will not be dealt with here, rather only their effects will be considered.

1. Enter the following syntax in the AS object instance address text field:
   %@@CalcS71500ASOAddr(„%Instance%“)@@%
2. Enter the following syntax in the tag address text field:
   %@@CalcS71500Addr(oTag,“%Instance%“,“%Address%“)@@%

Figure 4–17



☑ Allow instance addresses to be defined when the Instance is created (value used by %Instance%)

AS Object Instance Address Syntax (only for non-OPC Channels)

```
%@@CalcS71500ASOAddr("%Instance%")@@%
```

%ASName%	Name of the AS Node
%ASOName%	Name of the instance
%Instance%	Value that is entered by the user at the time the instance is created
%Attribute:Name%	Value of the Attribute with the name 'Name'
%UID%	DBA generated Unique Identifier per AS Object
~~%XMLAttr%~~	Any XML Attribute (e.g. %SeqNum% would use the value of 34 from this XML: <Node SeqNum="34">

AS Object Tag Address Syntax

```
%@@CalcS71500Addr(oTag,"%Instance%","%Address%")@@%
```

%ASName%	Name of the AS Node
%ASOName%	Name of the instance
%Instance%	Value that is entered by the user at the time the instance is created
%TagName%	Name of the Tag whose address is being calculated
%Address%	Value of the Address field of the Tag
%Field#TagAddress%	Value that is entered by the user in the DBA for each tag of the instance (if more than 1 field)
%Attribute:Name%	Value of the Attribute with the name 'Name'
%UID%	DBA generated Unique Identifier per AS Object
%OPCDataType%	OPC Data Type number for the tag type (e.g. BIT=11, FLOAT=4)
%UANamespace%	OPC UA Address Namespace
%UAType%	OPC UA Address ValueRank
%UAValue%	OPC UA Address Value
~~%XMLAttr%~~	Any XML Attribute (e.g. %SeqNum% would use the value of 34 from this XML: <T SeqNum="34">

OPC A&E Source Syntax (only for OPC A&E Channels)

```
%ASOName%
```

%ASName%	Name of the AS Node
%ASOName%	Name of the instance
%Attribute:Name%	Value of the Attribute with the name 'Name'

* Double click blue text to insert at cursor location

Here, the labels "CalcS71500ASOAddr" and "CalcS71500Addr" are the names of the two scripts which calculate exactly the addresses of the tags.
These scripts can be found in Chapter "6 A". Copy these into the corresponding script file for your AS node type. The scripts can be inserted at the end of the script file, in the "USER DEFINED FUNCTIONS" area.

The script file can be found in the following directory:

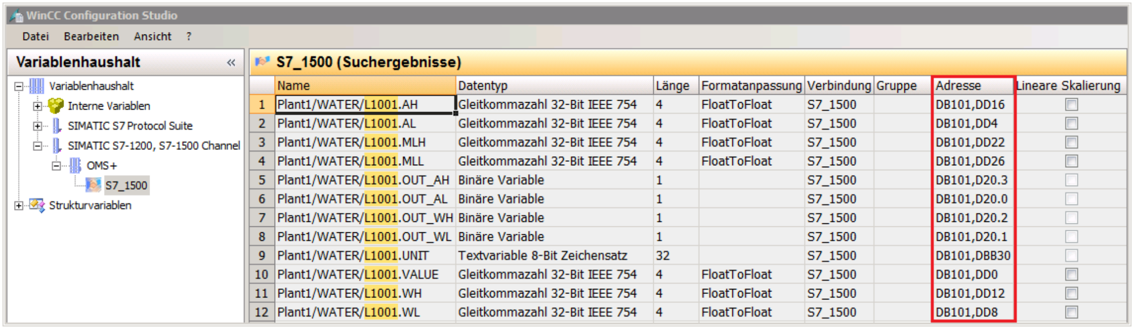
"C:\Program Files (x86)\SIEMENS\DBA\<Name of AS node type>\<Name of AS node type>Scripts.txt"

In our example:

„C:\Program Files (x86)\SIEMENS\DBA\S71500\S71500Scripts.txt“

Without going into the details of these scripts, it is possible to see the result in the WinCC tag manager after successful configuration (Figure 4–18). It can be seen that, based on the DB offset value of 0 entered during the instance definition (Figure 4–16), the byte offset for each tag has been calculated and entered as an address.

Figure 4–18



	Name	Datentyp	Länge	Formatanpassung	Verbindung	Gruppe	Adresse	Lineare Skalierung
1	Plant1/WATER/L1001.AH	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD16	☐
2	Plant1/WATER/L1001.AL	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD4	☐
3	Plant1/WATER/L1001.MLH	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD22	☐
4	Plant1/WATER/L1001.MLL	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD26	☐
5	Plant1/WATER/L1001.OUT_AH	Binäre Variable	1		S7_1500		DB101,D20.3	☐
6	Plant1/WATER/L1001.OUT_AL	Binäre Variable	1		S7_1500		DB101,D20.0	☐
7	Plant1/WATER/L1001.OUT_WH	Binäre Variable	1		S7_1500		DB101,D20.2	☐
8	Plant1/WATER/L1001.OUT_WL	Binäre Variable	1		S7_1500		DB101,D20.1	☐
9	Plant1/WATER/L1001.UNIT	Textvariable 8-Bit Zeichensatz	32		S7_1500		DB101,DBB30	☐
10	Plant1/WATER/L1001.VALUE	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD0	☐
11	Plant1/WATER/L1001.WH	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD12	☐
12	Plant1/WATER/L1001.WL	Gleitkommazahl 32-Bit IEEE 754	4	FloatToFloat	S7_1500		DB101,DD8	☐

Note

Note that the addresses in WinCC and in the SQL database are different.
With the aid of the scripts, DBA creates the addresses just as they appear in the SQL database (e.g., "0001:CL:%DB101.%DD16"). They are then automatically displayed correctly in WinCC (e.g., "DB101,DD16").

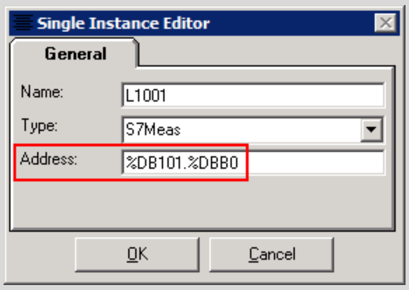
In cases where the address syntax in your SQL database differs from our example, you must adapt the scripts at the appropriate points ("sAddr" tag).

Unstructured program design in the S7-1500 PU

If the S7-1500 program to be integrated has been created in an unstructured manner and no rules for addressing can be derived, you can set the parameters for the addresses of all the tags when an instance is being created.

1. Enter the following syntax in the AS object instance address text field (Figure 4–19):
   0001:CL:%Instance%

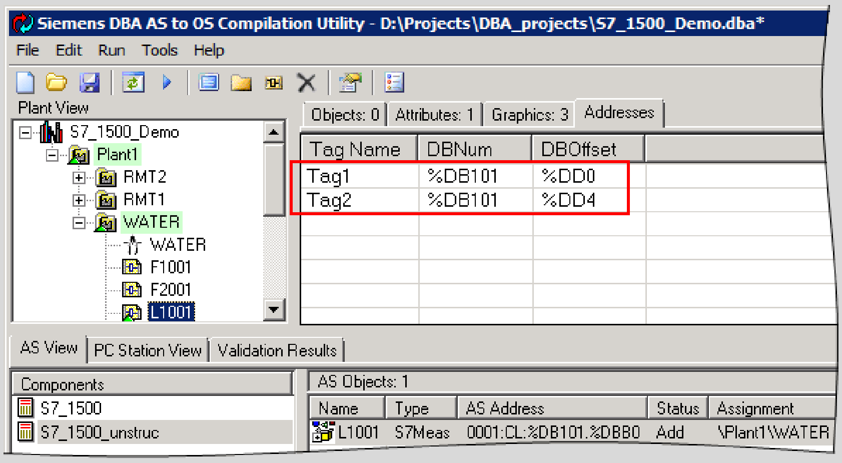
Note With this setting, the address must be entered as follows when creating an AS object in DBA at a later time (Chapter "4.6 Creating ASO instances"): %DBxxx.%Dxxx.



Name:	L1001
Type:	S7Meas
Address:	%DB101.%DBB0

2. Enter the following syntax in the tag address text field (Figure 4–19):
   0001:CL:%DBNum#TagAddress%.%DBOffset#TagAddress%

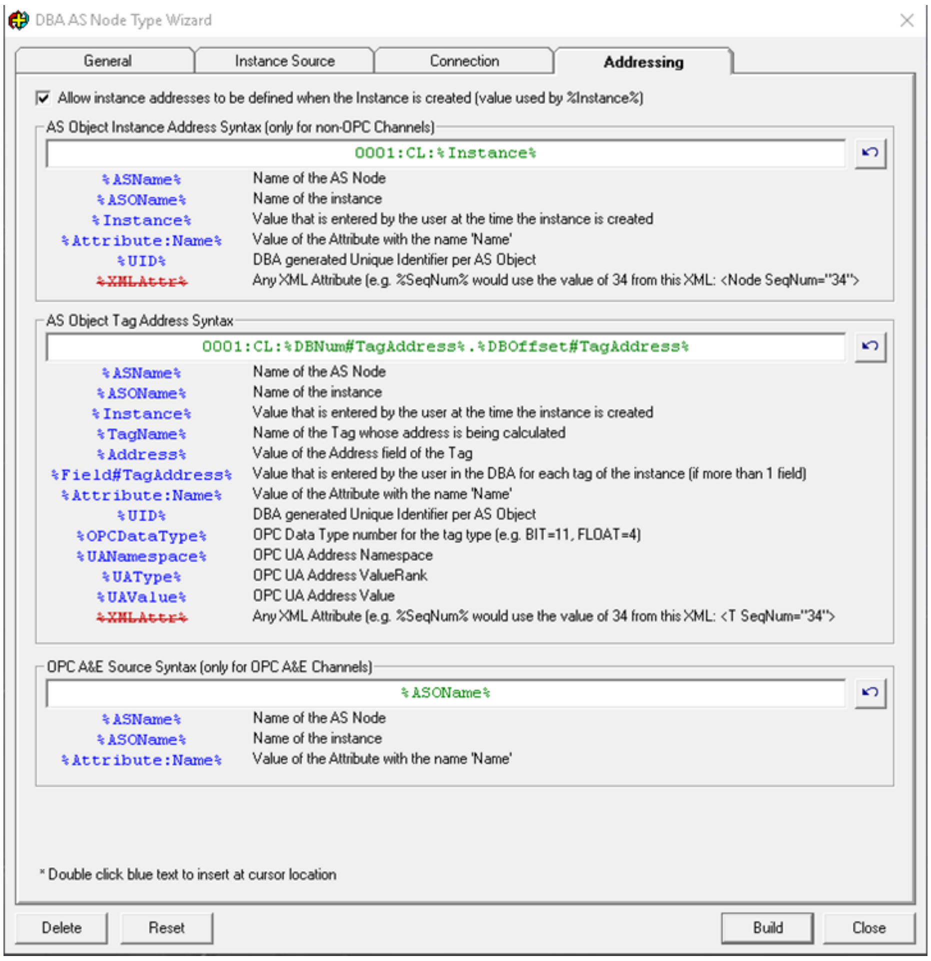
Note With this setting, a new "Address" tab with the columns TagName, DBNum and DBOffset is available when you assign the ASO instance parameters at a later time. Here, you can individually assign the address parameters for each tag



Tag Name	DBNum	DBOffset
Tag1	%DB101	%DD0
Tag2	%DB101	%DD4

Name	Type	AS Address	Status	Assignment
L1001	S7Meas	0001:CL:%DB101.%DBB0	Add	\Plant1\WATER

Figure 4–19



☑ Allow instance addresses to be defined when the Instance is created (value used by %Instance%)

AS Object Instance Address Syntax (only for non-OPC Channels)

`0001:CL:%Instance%`

%ASName%	Name of the AS Node
%ASOName%	Name of the instance
%Instance%	Value that is entered by the user at the time the instance is created
%Attribute:Name%	Value of the Attribute with the name 'Name'
%UID%	DBA generated Unique Identifier per AS Object
~~%XMLAttr%~~	Any XML Attribute (e.g. %SeqNum% would use the value of 34 from this XML: <Node SeqNum="34">

AS Object Tag Address Syntax

`0001:CL:%DBNum#TagAddress%.%DBOffset#TagAddress%`

%ASName%	Name of the AS Node
%ASOName%	Name of the instance
%Instance%	Value that is entered by the user at the time the instance is created
%TagName%	Name of the Tag whose address is being calculated
%Address%	Value of the Address field of the Tag
%Field#TagAddress%	Value that is entered by the user in the DBA for each tag of the instance (if more than 1 field)
%Attribute:Name%	Value of the Attribute with the name 'Name'
%UID%	DBA generated Unique Identifier per AS Object
%OPCDataType%	OPC Data Type number for the tag type (e.g. BIT=11, FLOAT=4)
%UANamespace%	OPC UA Address Namespace
%UAType%	OPC UA Address ValueRank
%UAValue%	OPC UA Address Value
~~%XMLAttr%~~	Any XML Attribute (e.g. %SeqNum% would use the value of 34 from this XML: <T SeqNum="34">

OPC A&E Source Syntax (only for OPC A&E Channels)

`%ASOName%`

%ASName%	Name of the AS Node
%ASOName%	Name of the instance
%Attribute:Name%	Value of the Attribute with the name 'Name'

* Double click blue text to insert at cursor location

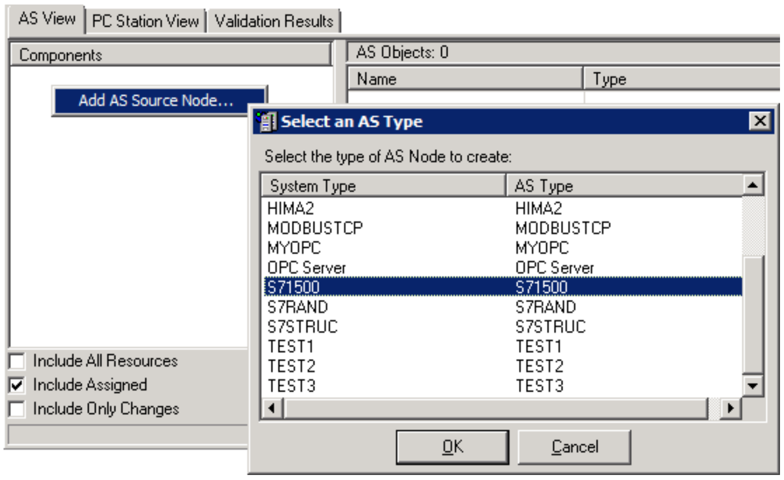
4.4 Creating the AS node

All the created AS node types are stored in the OPEN OS system folder.
From these AS node types, you can now create instances in DBA that map the controllers of the third-party system.

Once you have generated the AS node type, you can then create the AS node in the DBA.

1. Open a new or existing project in DBA.
2. In the "AS View" tab of the component view, add a new AS node.
   Select the "Add AS Source Node..." command in the shortcut menu.
3. Select the AS node type that was generated.

Figure 4-20

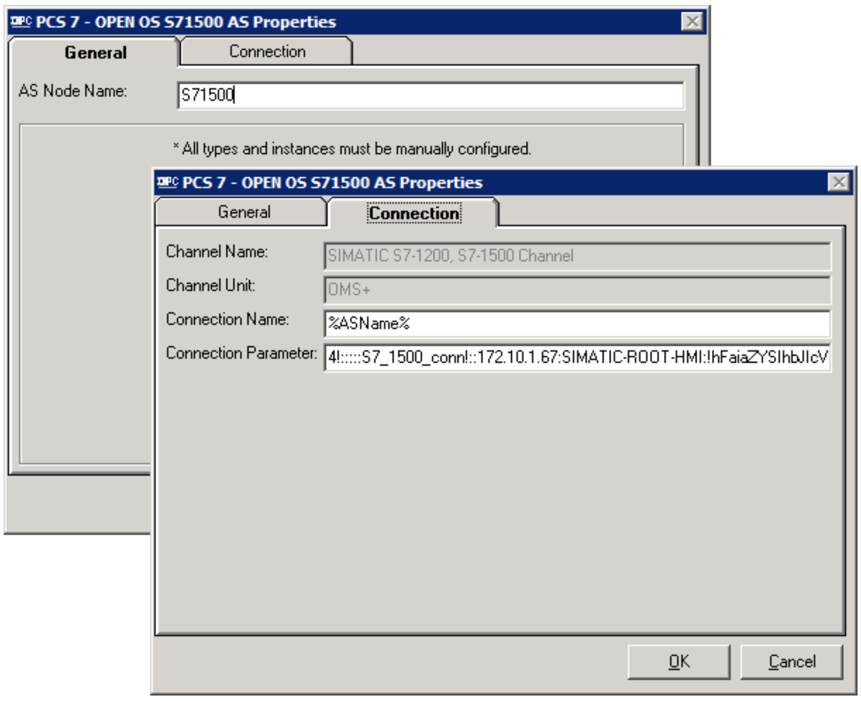


Select the type of AS Node to create:

System Type	AS Type
HIMA2	HIMA2
MODBUSTCP	MODBUSTCP
MYOPC	MYOPC
OPC Server	OPC Server
S71500	S71500
S7RAND	S7RAND
S7STRUC	S7STRUC
TEST1	TEST1
TEST2	TEST2
TEST3	TEST3

4. Assign a name for the station.
5. Depending on the setting that was chosen during the generation of the AS node type, you may still be able to adjust the connection name and the addressing.

Figure 4–21



AS Node Name: S71500

* All types and instances must be manually configured.

Channel Name: SIMATIC S7-1200, S7-1500 Channel

Channel Unit: OMS+

Connection Name: %ASName%

Connection Parameter: 4!::::S7_1500_conn!::172.10.1.67:SIMATIC-ROOT-HMI:!hFaiaZYSlhbJIcV

The new AS node has now been crated in the DBA AS View.

4.5 Generating AS object types

The AS object types in the DBA map the function blocks from the AS. An ASO type can be instantiated in the DBA as often as needed. Changes made to a type effect all instances of the type. The following steps use a block for an analog value measuring point to demonstrate how to generate this type. These tasks should then be carried out for all required blocks.

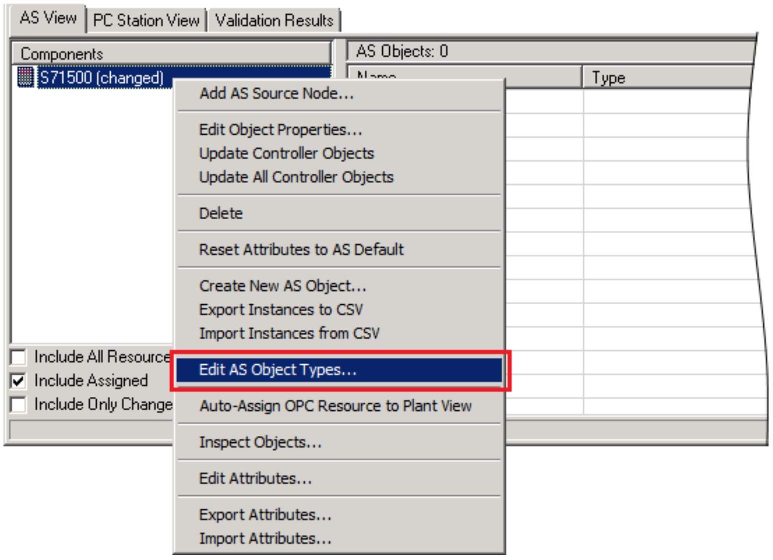
The data established in the chapter "4.1 Acquiring data" are the basis for generating the ASO types.

An ASO type can contain the following objects:

- Tags (type and address)
- Messages (message text, triggering event, accompanying values)
- Enumerations
- Attributes
- Runtime scripts

Start the DBA Type Editor by selecting the "Edit AS Object Types..." command from the AS node shortcut menu.

Figure 4–22



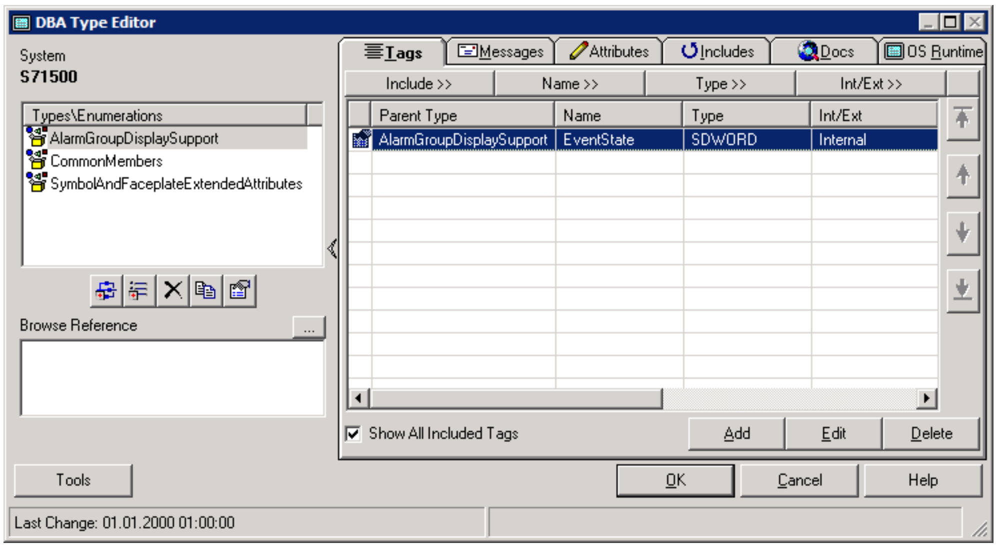
Note The AS object parameters are assigned for an AS node but are project specific. If you wish to reuse the objects in another DBA project, which uses the same AS node type, you can use the Export/Import function for ASO types.

Some attributes and tags in DBA must be added to the AS objects so that the third-party system objects behave like PCS 7 objects at a later stage. For this, the following types have already been created by default in DBA:

- AlarmGroupDisplaySupport
- CommonMembers
- SymbolAndFaceplateExtendedAttributes

These types introduce tags or attributes which are required for managing alarms or for managing block icons or faceplates. They can be included in any type that these functionalities are intended to work with.

Figure 4–23



Parent Type	Name	Type	Int/Ext
AlarmGroupDisplaySupport	EventState	SDWORD	Internal

The following sections contain more detailed descriptions of the "Variables, Messages, Attributes, Includes, Docs and OS Runtime" tabs. In the "Docs" tab you can link descriptions to your types in HTML format, and in the "OS runtime" tab you can add scripts for the OS runtime. You can find more detailed information about this topic in the "PCS 7 Open OS DBA Type Editor" manual.

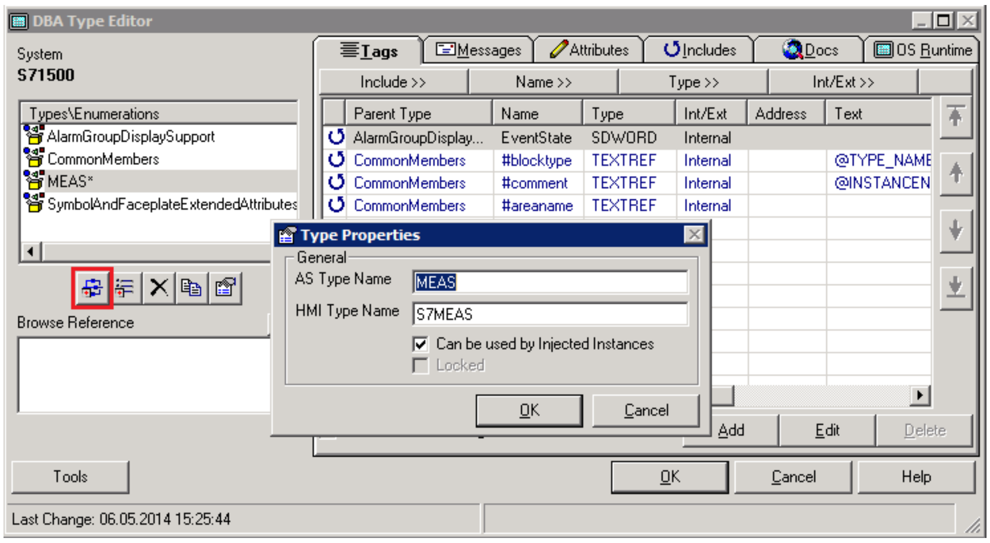
4.5.1 Creating a new type

To create a new type in the DBA Type Editor, click the "Add New Type" button.

The dialog window that is then displayed prompts you to enter the following data:

- AS Type Name – Choose a name that creates a relation to the object in the AS.
- HMI Type Name – Choose the same name that you assigned for the HMI symbol. It is recommended additionally to work in a reference to the name of the third-party system.

Figure 4–24



System
S71500

Types\Enumerations
- AlarmGroupDisplaySupport
- CommonMembers
- MEAS*
- SymbolAndFaceplateExtendedAttributes

Parent Type	Name	Type	Int/Ext	Address	Text
AlarmGroupDisplay...	EventState	SDWORD	Internal		
CommonMembers	#blocktype	TEXTREF	Internal		@TYPE_NAME
CommonMembers	#comment	TEXTREF	Internal		@INSTANCEN
CommonMembers	#areaname	TEXTREF	Internal		

Type Properties

General

AS Type Name: MEAS

HMI Type Name: S7MEAS

[✓] Can be used by Injected Instances

[ ] Locked

OK Cancel

Last Change: 06.05.2014 15:25:44

4.5.2 Including default types

With type inclusion, it is possible to add variables and attributes to the new types to achieve specific functionalities.

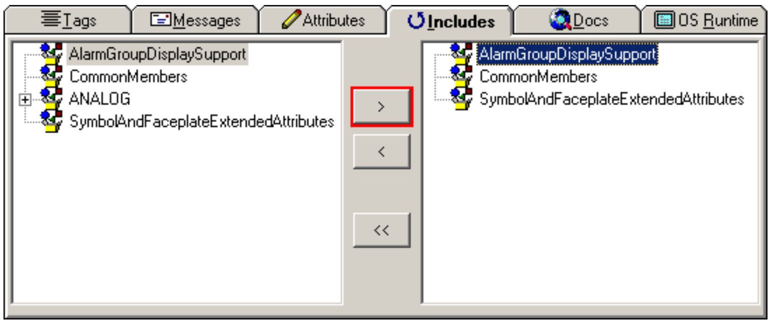
The following types are included by default:

- CommonMembers
- SymbolAndFaceplateExtendedAttributes

All available types are displayed in the left-hand window. The right-hand window contains the types that are already included. If the block also features an alarm function, include the "AlarmGroupDisplaySupport" type.

1. Select the "AlarmGroupDisplaySupport" type in the left-hand window.
2. Include the type by clicking the button (right arrow).

Figure 4–25



AlarmGroupDisplaySupport
CommonMembers
ANALOG
SymbolAndFaceplateExtendedAttributes

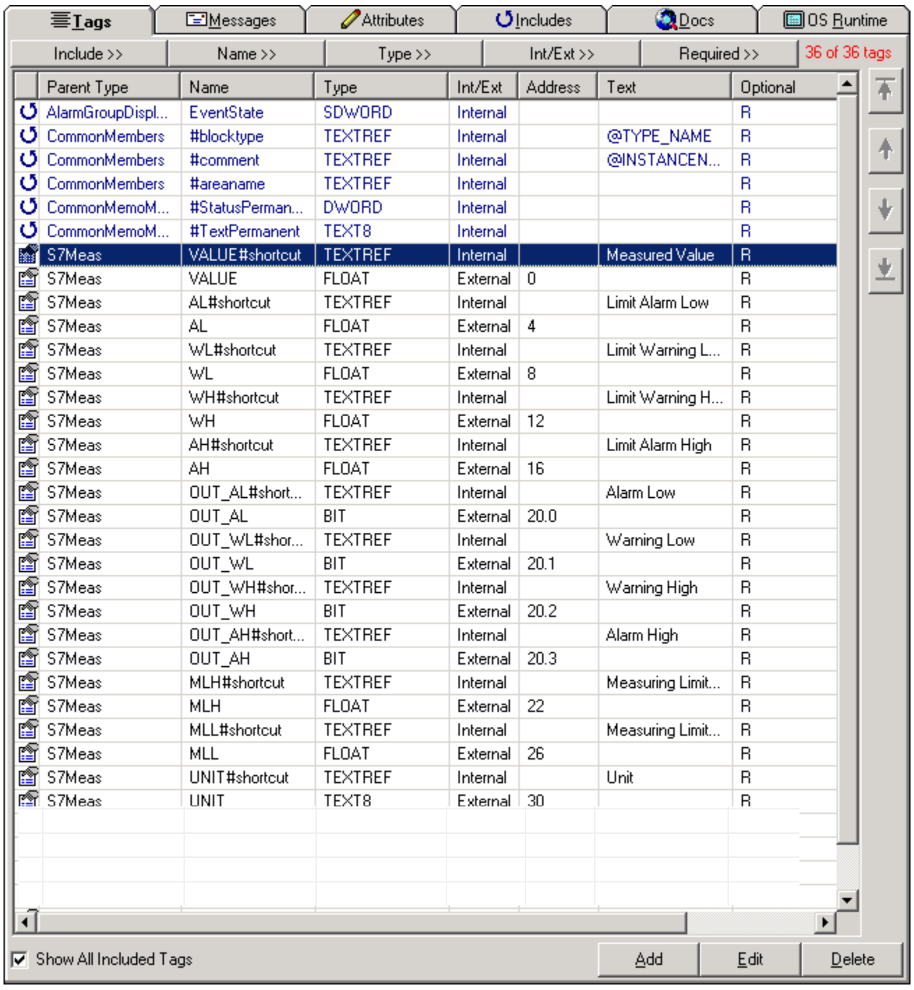
AlarmGroupDisplaySupport
CommonMembers
SymbolAndFaceplateExtendedAttributes

The new type is now equipped with the basic functionality for PCS 7 blocks.
You can remove the functionality of the included types by clicking the "Left arrow" button.

4.5.3 Creating tags

The tags form the interface between the OS and the automation system.
You can configure the tags in the "Tags" tab in the DBA Type Editor.

Figure 4-26



Parent Type	Name	Type	Int/Ext	Address	Text	Optional
AlarmGroupDispl...	EventState	SDWORD	Internal			R
CommonMembers	#blocktype	TEXTREF	Internal		@TYPE_NAME	R
CommonMembers	#comment	TEXTREF	Internal		@INSTANCEN...	R
CommonMembers	#areaname	TEXTREF	Internal			R
CommonMemoM...	#StatusPerman...	DWORD	Internal			R
CommonMemoM...	#TextPermanent	TEXT8	Internal			R
S7Meas	VALUE#shortcut	TEXTREF	Internal		Measured Value	R
S7Meas	VALUE	FLOAT	External	0		R
S7Meas	AL#shortcut	TEXTREF	Internal		Limit Alarm Low	R
S7Meas	AL	FLOAT	External	4		R
S7Meas	WL#shortcut	TEXTREF	Internal		Limit Warning L...	R
S7Meas	WL	FLOAT	External	8		R
S7Meas	WH#shortcut	TEXTREF	Internal		Limit Warning H...	R
S7Meas	WH	FLOAT	External	12		R
S7Meas	AH#shortcut	TEXTREF	Internal		Limit Alarm High	R
S7Meas	AH	FLOAT	External	16		R
S7Meas	OUT_AL#short...	TEXTREF	Internal		Alarm Low	R
S7Meas	OUT_AL	BIT	External	20.0		R
S7Meas	OUT_WL#shor...	TEXTREF	Internal		Warning Low	R
S7Meas	OUT_WL	BIT	External	20.1		R
S7Meas	OUT_WH#shor...	TEXTREF	Internal		Warning High	R
S7Meas	OUT_WH	BIT	External	20.2		R
S7Meas	OUT_AH#short...	TEXTREF	Internal		Alarm High	R
S7Meas	OUT_AH	BIT	External	20.3		R
S7Meas	MLH#shortcut	TEXTREF	Internal		Measuring Limit...	R
S7Meas	MLH	FLOAT	External	22		R
S7Meas	MLL#shortcut	TEXTREF	Internal		Measuring Limit...	R
S7Meas	MLL	FLOAT	External	26		R
S7Meas	UNIT#shortcut	TEXTREF	Internal		Unit	R
S7Meas	UNIT	TEXT8	External	30		R

To add new tags, click the "Add" button. An additional dialog window is opened.
In the "General" tab, you can enter the following data:

- Name of the tag
- Data type
- WinCC type (default is structure tag)
- Source (external, internal, indirect)
- Address
- Runtime options
- Inclusion rules

Use the data type as illustrated in Table 4–6 in the "Address Syntax" column below:

Table 4–6

WinCC data type	Address Syntax	AS Data Type
Binary	D	Bit
Unsigned 8-bit	DBB	Byte
Signed 8-bit	DBB	Byte
Unsigned 16-bit	DBW	Word
Signed 16-bit	DBW	Word
Unsigned 32-bit	DD	Double word
Signed 32-bit	DD	Double word
Float 32-bit	DD	Double word
Float 64-bit	DD	Double word
Text 8-bit	DBB	Byte
Text 16-bit	DBW	Word

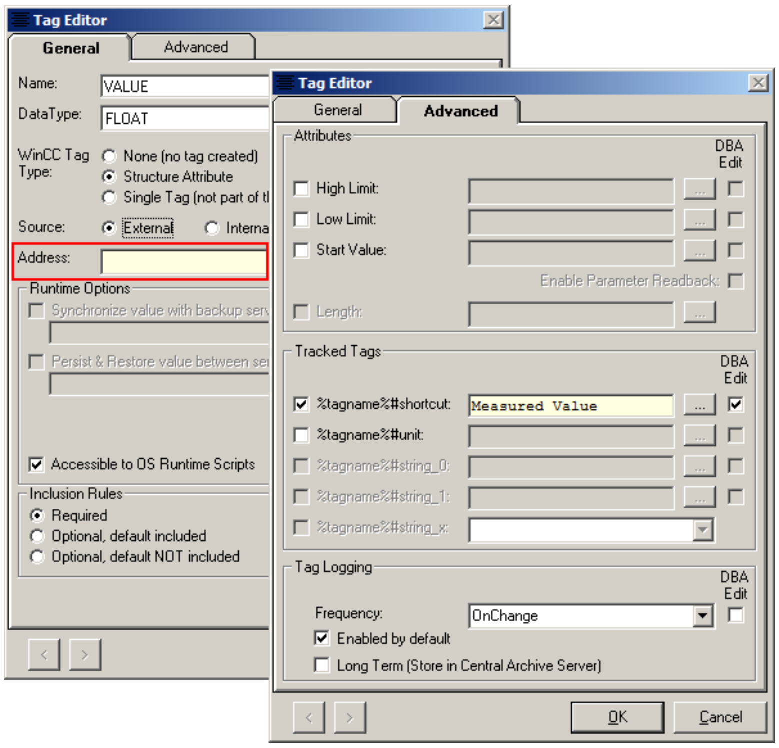
The entry in the "Address" field (Figure 4-27) is used in the script referred to in Chapter "4.3.4 Addressing tab" to calculate the tag addresses automatically.

In the "Advanced" tab, you can configure additional settings for the tag:

- Upper and lower limit
- Start value
- Auxiliary tags (description, unit, ...)
- Archiving

The “DBA Edit” option allows you to customize the specified start values at instantiation of the type.

Figure 4–27


Tag Editor

General | Advanced

Name: VALUE

DataType: FLOAT

WinCC Tag Type: None (no tag created) [✓ Structure Attribute] Single Tag (not part of t

Source: [✓ External] Interna

Address:

Runtime Options

☐ Synchronize value with backup serv

☐ Persist & Restore value between se

☑ Accessible to OS Runtime Scripts

Inclusion Rules

[✓ Required] Optional, default included Optional, default NOT included

Tag Editor

General | Advanced

Attributes

		DBA Edit
☐ High Limit:		☐
☐ Low Limit:		☐
☐ Start Value:		☐
Enable Parameter Readback:		☐
☐ Length:		

Tracked Tags

		DBA Edit
☑ %tagname%#shortcut:	Measured Value	☑
☐ %tagname%#unit:		☐
☐ %tagname%#string_0:		☐
☐ %tagname%#string_1:		☐
☐ %tagname%#string_x:		

Tag Logging

		DBA Edit
Frequency:	OnChange	☐

☑ Enabled by default

☐ Long Term (Store in Central Archive Server)

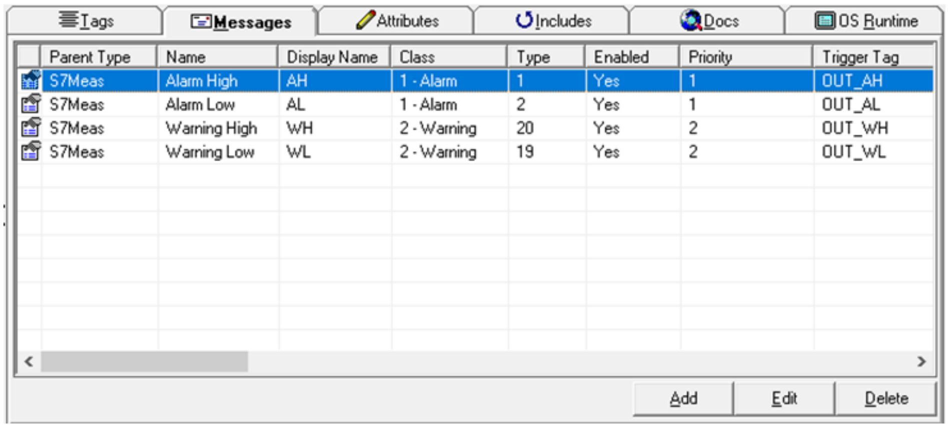
Click the "OK" button to close the dialog window and create the type tag.

4.5.4 Configuring alarms

In the "Messages" tab, you can configure the process messages of the AS block. The messages can be triggered by different tags or by means of an OPC A&E server. You can configure the following parameters:

- Alarm name
- Message text
- Message class
- Priority
- Trigger tag
- Associated message values

Figure 4–28



Parent Type	Name	Display Name	Class	Type	Enabled	Priority	Trigger Tag
S7Meas	Alarm High	AH	1 - Alarm	1	Yes	1	OUT_AH
S7Meas	Alarm Low	AL	1 - Alarm	2	Yes	1	OUT_AL
S7Meas	Warning High	WH	2 - Warning	20	Yes	2	OUT_WH
S7Meas	Warning Low	WL	2 - Warning	19	Yes	2	OUT_WL

Click "Add" to configure new messages of the ASO type. A dialog with the following tabs will open:

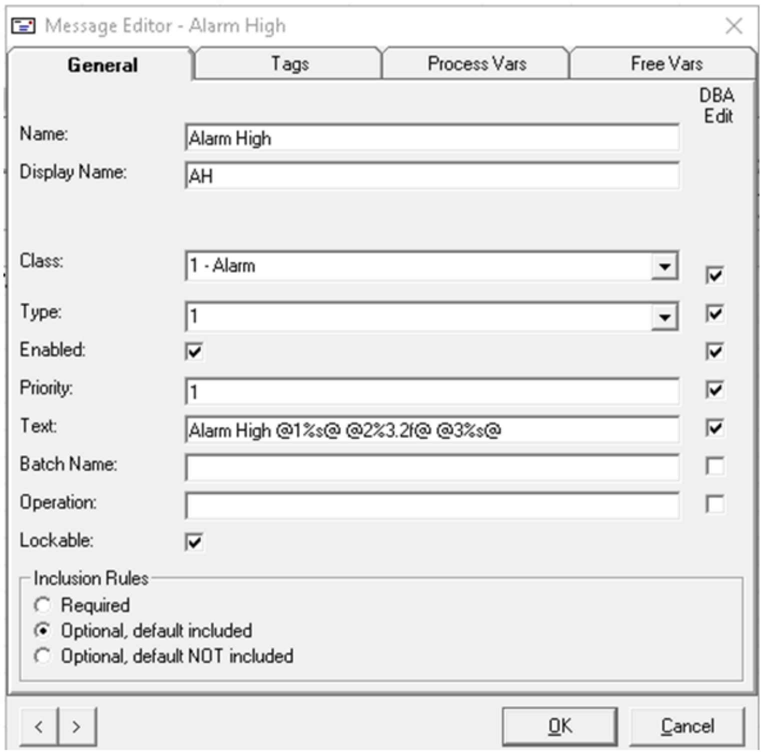
- General
- Tags
- Process Vars
- Free Vars

General tab

Assign the following parameters in the "General" tab:

- Name
- Display name
- Message class
- Priority
- Message text

Figure 4–29



Name:	Alarm High	
Display Name:	AH	
Class:	1 - Alarm	✓
Type:	1	✓
Enabled:	✓	✓
Priority:	1	✓
Text:	Alarm High @1%s@ @2%3.2f@ @3%s@	✓
Batch Name:		☐
Operation:		☐
Lockable:	✓	

Inclusion Rules
- ○ Required
- ● Optional, default included
- ○ Optional, default NOT included

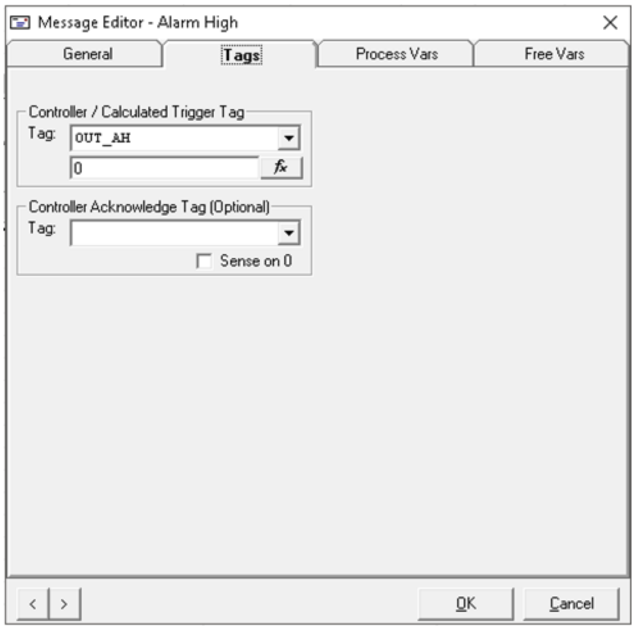
Associated message values are integrated using the control commands known from PCS 7. For example, associated value 2 of the floating-point number type: @2%3.2f@.

The associated values must be declared in the "Process Vars" tab.

Tags tab

In the "Tags" tab, the triggering tags for the message in WinCC and the initiating trigger in the AS are configured.

Figure 4–30


Message Editor - Alarm High

General | Tags | Process Vars | Free Vars

Controller / Calculated Trigger Tag
Tag: OUT_AH
0

Controller Acknowledge Tag (Optional)
Tag:
Sense on 0

Here you can configure the trigger and optional acknowledge in the controller for the OS – if the connected third-party system supports this. The message behavior then corresponds to that of the PCS 7 blocks. Messages are shown in the alarm group display and in the message line, and the "Loop in Alarm" function can be used.

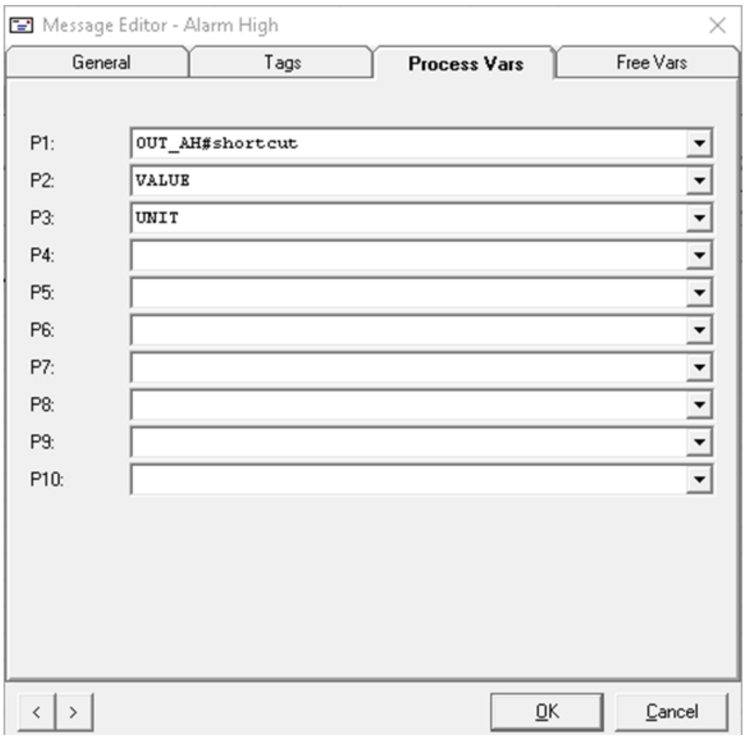
With button "fx" any desired terms can be build. Basing of variables and limit values these terms calculate and – if necessary - activate the trigger of a message.

Process Variables tab

In this tab, you can configure the tags that you wish to use as associated values in your message texts. The system provides 10 fields.

You may select any of the tags you created when generating the type.

Figure 4–31



P1:	OUT_AH#shortcut
P2:	VALUE
P3:	UNIT
P4:	
P5:	
P6:	
P7:	
P8:	
P9:	
P10:	

Free Variables tab

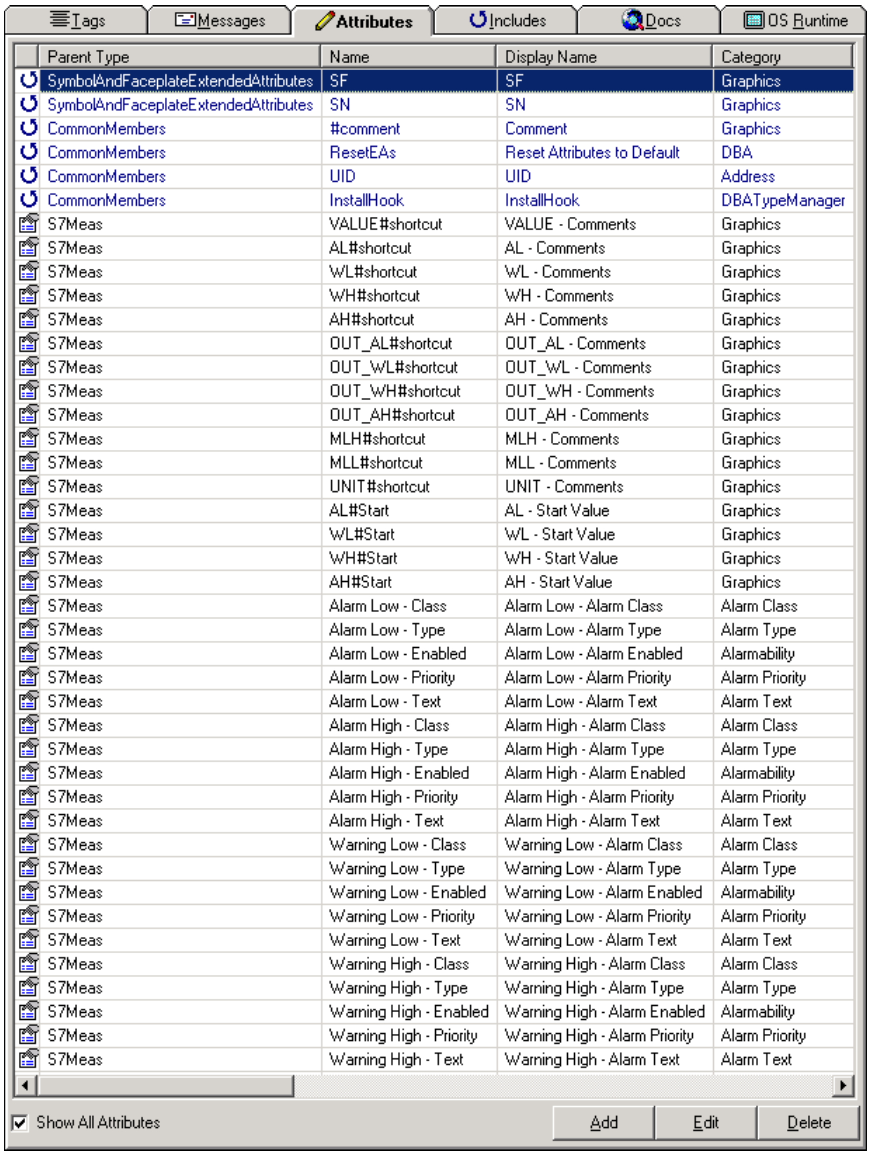
In the "Free Tags" tab, you can parametrize constantly recurring text blocks, for example. You can find further information in the WinCC Alarm Logging description.

4.5.5 Attributes

The attributes determine the behavior of a tag, a message or the instance of a type. Most attributes are automatically created during the generation of tags and messages or copied during the inclusion of other types. It is not necessary to make changes to these attributes. By default, attributes generated by the system are not displayed. You can display these attributes by activating the "Show All Attributes" selection box.

You will find detailed information about the attributes in the "PCS 7 Open OS DBA Type Editor" manual under chapter 6.5 "Working with the Attribute Editor".

Figure 4–32



Parent Type	Name	Display Name	Category
SymbolAndFaceplateExtendedAttributes	SF	SF	Graphics
SymbolAndFaceplateExtendedAttributes	SN	SN	Graphics
CommonMembers	#comment	Comment	Graphics
CommonMembers	ResetEAs	Reset Attributes to Default	DBA
CommonMembers	UID	UID	Address
CommonMembers	InstallHook	InstallHook	DBATypeManager
S7Meas	VALUE#shortcut	VALUE - Comments	Graphics
S7Meas	AL#shortcut	AL - Comments	Graphics
S7Meas	WL#shortcut	WL - Comments	Graphics
S7Meas	WH#shortcut	WH - Comments	Graphics
S7Meas	AH#shortcut	AH - Comments	Graphics
S7Meas	OUT_AL#shortcut	OUT_AL - Comments	Graphics
S7Meas	OUT_WL#shortcut	OUT_WL - Comments	Graphics
S7Meas	OUT_WH#shortcut	OUT_WH - Comments	Graphics
S7Meas	OUT_AH#shortcut	OUT_AH - Comments	Graphics
S7Meas	MLH#shortcut	MLH - Comments	Graphics
S7Meas	MLL#shortcut	MLL - Comments	Graphics
S7Meas	UNIT#shortcut	UNIT - Comments	Graphics
S7Meas	AL#Start	AL - Start Value	Graphics
S7Meas	WL#Start	WL - Start Value	Graphics
S7Meas	WH#Start	WH - Start Value	Graphics
S7Meas	AH#Start	AH - Start Value	Graphics
S7Meas	Alarm Low - Class	Alarm Low - Alarm Class	Alarm Class
S7Meas	Alarm Low - Type	Alarm Low - Alarm Type	Alarm Type
S7Meas	Alarm Low - Enabled	Alarm Low - Alarm Enabled	Alarmability
S7Meas	Alarm Low - Priority	Alarm Low - Alarm Priority	Alarm Priority
S7Meas	Alarm Low - Text	Alarm Low - Alarm Text	Alarm Text
S7Meas	Alarm High - Class	Alarm High - Alarm Class	Alarm Class
S7Meas	Alarm High - Type	Alarm High - Alarm Type	Alarm Type
S7Meas	Alarm High - Enabled	Alarm High - Alarm Enabled	Alarmability
S7Meas	Alarm High - Priority	Alarm High - Alarm Priority	Alarm Priority
S7Meas	Alarm High - Text	Alarm High - Alarm Text	Alarm Text
S7Meas	Warning Low - Class	Warning Low - Alarm Class	Alarm Class
S7Meas	Warning Low - Type	Warning Low - Alarm Type	Alarm Type
S7Meas	Warning Low - Enabled	Warning Low - Alarm Enabled	Alarmability
S7Meas	Warning Low - Priority	Warning Low - Alarm Priority	Alarm Priority
S7Meas	Warning Low - Text	Warning Low - Alarm Text	Alarm Text
S7Meas	Warning High - Class	Warning High - Alarm Class	Alarm Class
S7Meas	Warning High - Type	Warning High - Alarm Type	Alarm Type
S7Meas	Warning High - Enabled	Warning High - Alarm Enabled	Alarmability
S7Meas	Warning High - Priority	Warning High - Alarm Priority	Alarm Priority
S7Meas	Warning High - Text	Warning High - Alarm Text	Alarm Text

☑ Show All Attributes

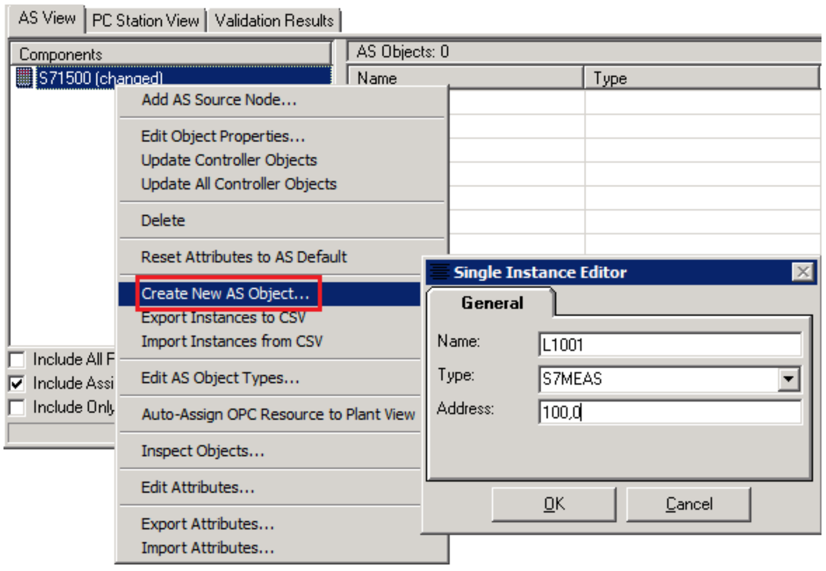
4.6 Creating ASO instances

The ASO types created in the last chapter emulate the structures (UDTs) of the S7-1500. Using the ASO types, instances will now be generated that emulate the instances in the data blocks of the AS.

As established in the chapter "4.1 Acquiring data", two motors, two valves, five analog value measuring points and one sequence (SFC) are used for our example PU. You can generate these instances by following the steps below:

Open the AS node shortcut menu and select the "Create New AS Object..." command.

Figure 4–33



Name: L1001
Type: S7MEAS
Address: 100,0

Enter the following data in the "Single Instance Editor" dialog window:

- Name of the instance (choose a name that you can associate with the AS program)
- ASO type
- AS address

Note

The "Address" text box is only available if you defined it when generating the AS type. In our case, the address string was as follows:

```
%@@CalcS71500ASOAddr("%Instance%")@@%
```

With %Instance%, we specified that the input was to occur in the "Address" field. By entering the DB number and the DB offset, the correct address can then be put together.

e.g.: DB100,DBB0 (WinCC) → 0001:CL:%DB100.%DBB0 (SQL database)

The addresses of the individual tags are calculated from the following address string:

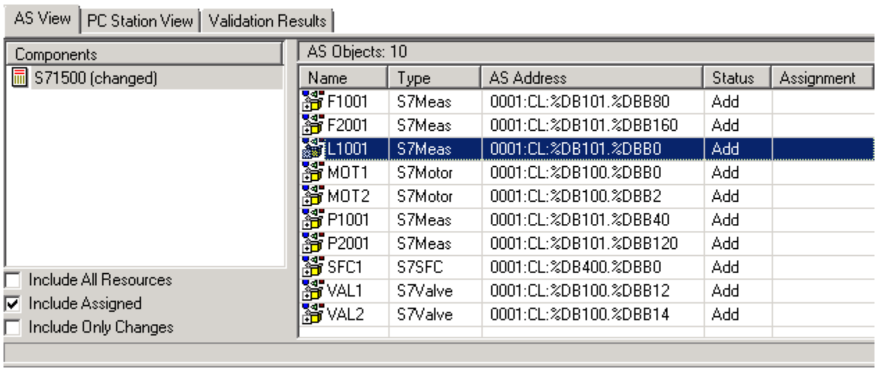
```
%@@CalcS71500Addr(oTag,"%Instance%","%Address%")@@%
```

As above, each individual tag address is calculated from the DB number, the DB offset and additionally the address that was entered in the ASO type (Figure 4–27).

In the event that you have an unstructured S7-1500 program, take note of the corresponding paragraph in Chapter "4.3.4 Addressing tab".

Carry out these steps for all required objects.

Figure 4–34



AS Objects: 10

Name	Type	AS Address	Status	Assignment
F1001	S7Meas	0001:CL:%DB101.%DBB80	Add	
F2001	S7Meas	0001:CL:%DB101.%DBB160	Add	
L1001	S7Meas	0001:CL:%DB101.%DBB0	Add	
MOT1	S7Motor	0001:CL:%DB100.%DBB0	Add	
MOT2	S7Motor	0001:CL:%DB100.%DBB2	Add	
P1001	S7Meas	0001:CL:%DB101.%DBB40	Add	
P2001	S7Meas	0001:CL:%DB101.%DBB120	Add	
SFC1	S7SFC	0001:CL:%DB400.%DBB0	Add	
VAL1	S7Valve	0001:CL:%DB100.%DBB12	Add	
VAL2	S7Valve	0001:CL:%DB100.%DBB14	Add	

☐ Include All Resources
☑ Include Assigned
☐ Include Only Changes

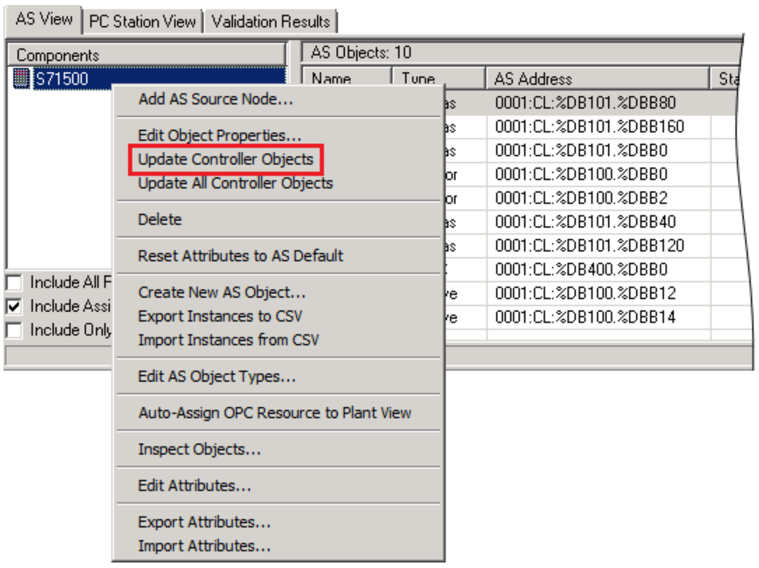
Whenever changes are made to the settings of the AS node, to the related types or to the created instances, the text "(changed)" is added to the AS node as a note.

Note

With the "DBA Object Inspector", you can read out information about which data was created on the OS. For example, you can already verify whether the structure of the address syntax is correct.
The command to open the "DBA Object Inspector" can be found in the shortcut menu of an AS node or an ASO instance.

By calling the "Update Controller Objects" function from the AS node shortcut menu, you can update all instances of the AS object.

Figure 4–35



Name	Type	AS Address	Sta
	as	0001:CL:%DB101.%DBB80	
	as	0001:CL:%DB101.%DBB160	
	as	0001:CL:%DB101.%DBB0	
	or	0001:CL:%DB100.%DBB0	
	or	0001:CL:%DB100.%DBB2	
	as	0001:CL:%DB101.%DBB40	
	as	0001:CL:%DB101.%DBB120	
	t	0001:CL:%DB400.%DBB0	
	ve	0001:CL:%DB100.%DBB12	
	ve	0001:CL:%DB100.%DBB14	

The configuration of the AS part for the S7-1500 package unit is now complete. The next steps can be found in the chapter "5 Configuring the PC station with the DBA".

5 Configuring the PC station with the DBA

The PC station in the DBA Editor serves as an interface to the operator system on which the third-party system is operated and monitored.

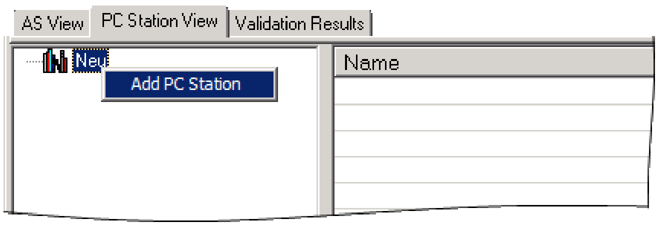
To create a PC station in the DBA, you first need a PCS 7 project complete with an operator station (OS) and a plant hierarchy. This can be a finished project that is to be expanded to include the functionality of the package unit, or you can create a new project.

5.1 Adding a PC Station

To create a new PC station in the DBA, follow the steps below:

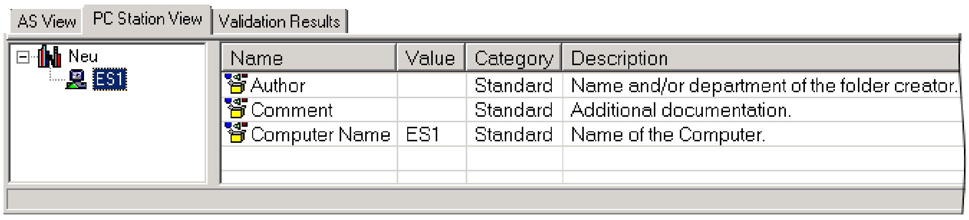
1. Switch to the "PC Station View" tab.
2. Select the "Add PC Station" command in the DBA project shortcut menu.

Figure 5–1



3. Assign the name of your ES to the PC station.
4. Also add the name of the ES (target OS) at the "Computer Name" line in the "Value" column.

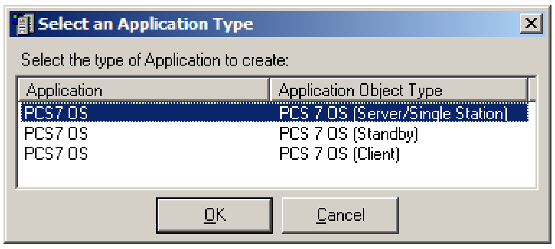
Figure 5–2



Name	Value	Category	Description
Author		Standard	Name and/or department of the folder creator.
Comment		Standard	Additional documentation.
Computer Name	ES1	Standard	Name of the Computer.

5. Select "Add Application" from the context menu of the PC station. A dialog window will then open.
6. Select the appropriate project type and click OK in the dialog window to confirm your selection. Name the application using the name of the corresponding OS project in the SIMATIC Manager.

Figure 5–3



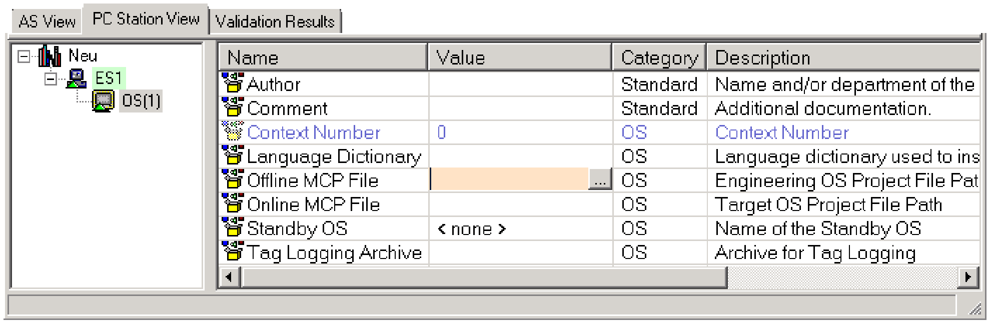
Select an Application Type

Select the type of Application to create:

Application	Application Object Type
PCS7 OS	PCS 7 OS (Server/Single Station)
PCS7 OS	PCS 7 OS (Standby)
PCS7 OS	PCS 7 OS (Client)

7. You can link the DBA project to the PCS 7 OS project by entering the path for the OS project at the "Offline MCP File" line in the "Value" column.
8. By clicking the "..." button you can navigate to the OS project with the aid of a file selection dialog window and then insert the path string.

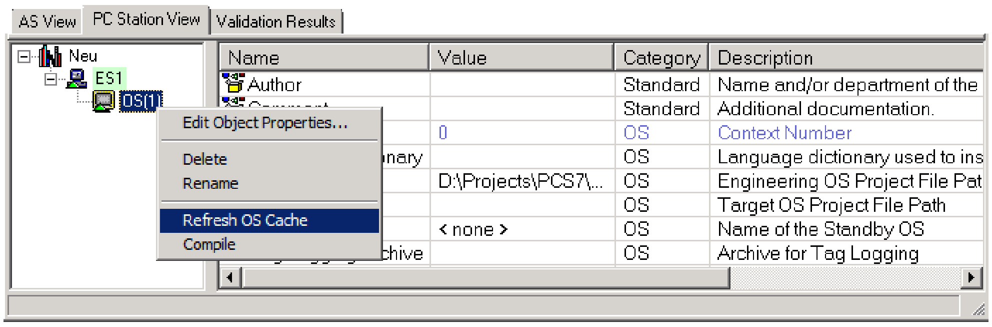
Figure 5–4



Name	Value	Category	Description
Author		Standard	Name and/or department of the
Comment		Standard	Additional documentation.
Context Number	0	OS	Context Number
Language Dictionary		OS	Language dictionary used to ins
Offline MCP File	...	OS	Engineering OS Project File Pat
Online MCP File		OS	Target OS Project File Path
Standby OS	< none >	OS	Name of the Standby OS
Tag Logging Archive		OS	Archive for Tag Logging

9. Update the project data in the DBA by selecting the "Refresh OS Cache" function in the shortcut menu. This function opens the PCS 7 OS project, reads the alarm classes, alarm types and tag logging archive information, and saves all of this information in the DBA project.

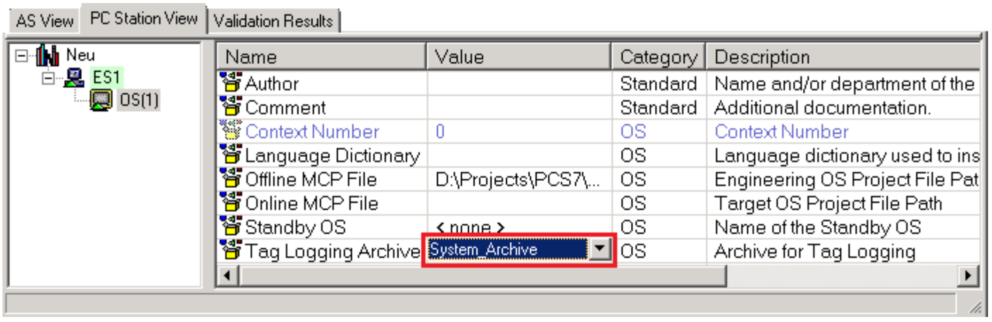
Figure 5–5



Name	Value	Category	Description
Author		Standard	Name and/or department of the
		Standard	Additional documentation.
	0	OS	Context Number
nary		OS	Language dictionary used to ins
	D:\Projects\PCS7\...	OS	Engineering OS Project File Pat
		OS	Target OS Project File Path
	< none >	OS	Name of the Standby OS
chive		OS	Archive for Tag Logging

10. Select a Tag Logging archive if one is present.

Figure 5–6



Name	Value	Category	Description
Author		Standard	Name and/or department of the
Comment		Standard	Additional documentation.
Context Number	0	OS	Context Number
Language Dictionary		OS	Language dictionary used to ins
Offline MCP File	D:\Projects\PCS7\...	OS	Engineering OS Project File Pat
Online MCP File		OS	Target OS Project File Path
Standby OS	< none >	OS	Name of the Standby OS
Tag Logging Archive	System_Archive	OS	Archive for Tag Logging

5.2 Creating a plant hierarchy

In the DBA, you can use the plant hierarchy to define the picture hierarchy as it appears on the OS. It is possible to use and extend an existing hierarchy from a PCS 7 project or to generate a new hierarchy.

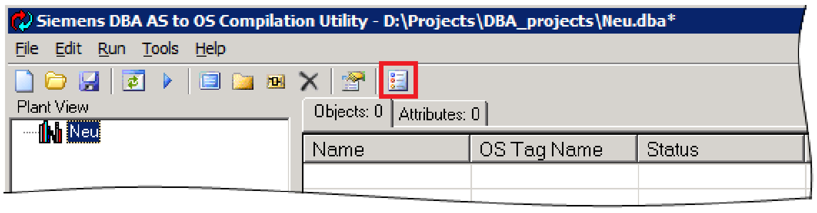
You will find detailed information in the PCS 7 Open OS DBA user manual.

In this example, an existing PH will be extended.

5.2.1 Setting project properties

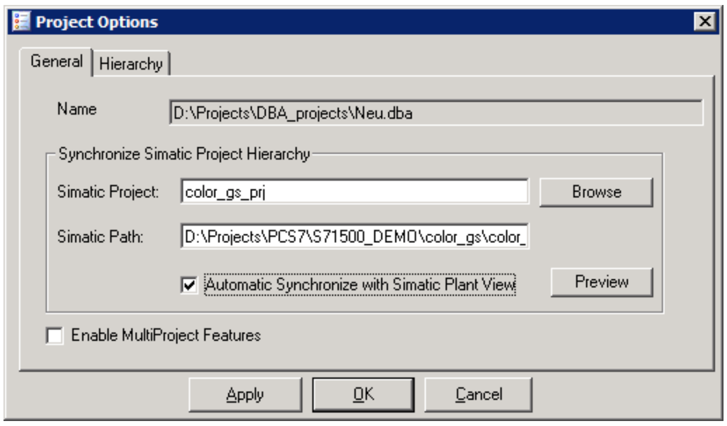
1. Click the "Change Project Properties" button.

Figure 5–7



2. In the "General" tab, you can define the project and the project path with which you intend to synchronize the PH. With the "Automatic Synchronize with Simatic Plant View" option, you can trigger an automatic synchronization with each compiling operation.

Figure 5–8



Project Options

General | Hierarchy

Name	D:\Projects\DBA_projects\Neu.dba

Synchronize Simatic Project Hierarchy

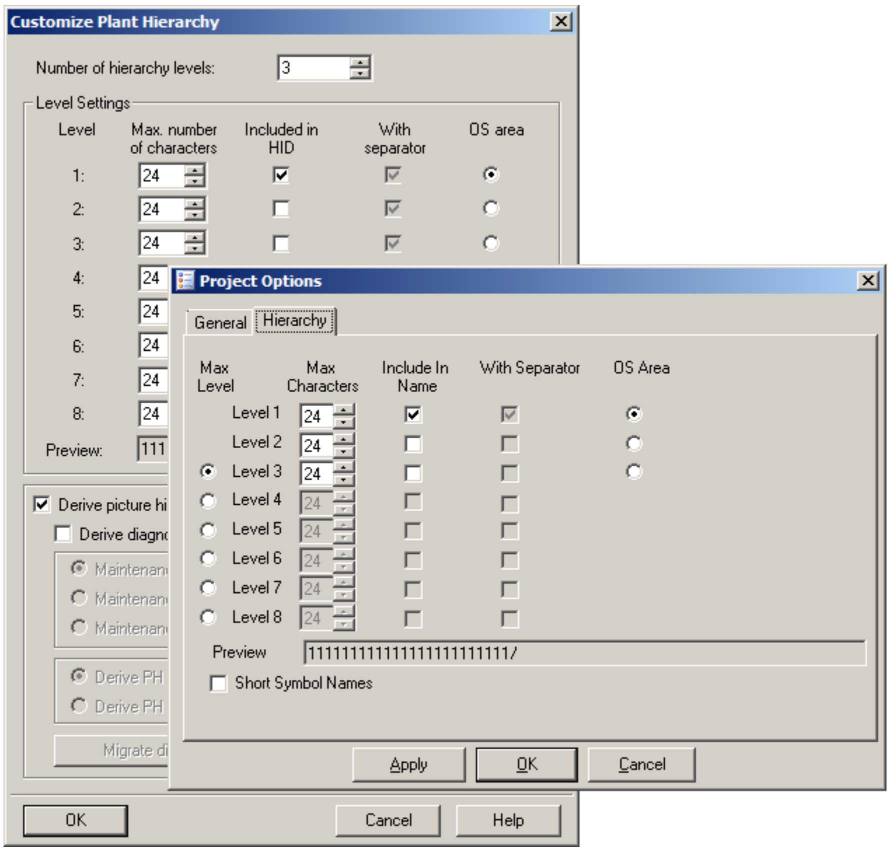
Simatic Project:	color_gs_prj
Simatic Path:	D:\Projects\PCS7\S71500_DEMO\color_gs\color_

[✓] Automatic Synchronize with Simatic Plant View

[ ] Enable MultiProject Features

3. In the "Hierarchy" tab, you can configure the PH settings. Here, you must choose the same settings as the ones in the PCS 7 project.

Figure 5–9



Customize Plant Hierarchy

Number of hierarchy levels: 3

Level Settings

Level	Max. number of characters	Included in HID	With separator	OS area
1:	24	✓	✓	●
2:	24		✓	○
3:	24		✓	○
4:	24			
5:	24			
6:	24			
7:	24			
8:	24			

Preview: 111

[✓] Derive picture hi

[ ] Derive diagn

Project Options

General | Hierarchy

Max Level		Max Characters	Include In Name	With Separator	OS Area
	Level 1	24	✓	✓	●
	Level 2	24			○
●	Level 3	24			○
○	Level 4	24			
○	Level 5	24			
○	Level 6	24			
○	Level 7	24			
○	Level 8	24			

Preview 111111111111111111111111/

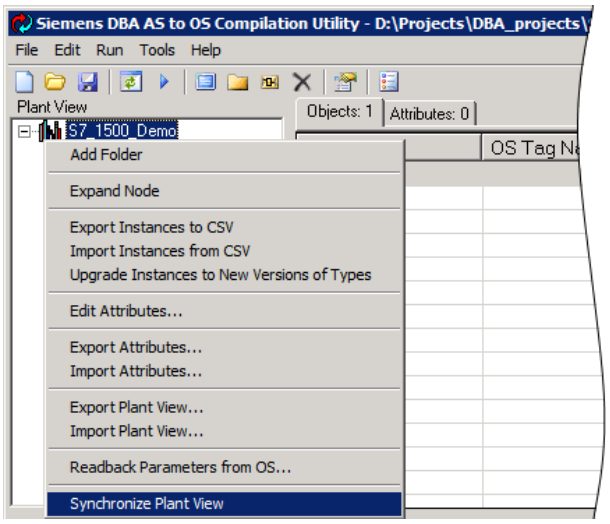
[ ] Short Symbol Names

4. Click the "OK" button to confirm the settings.

5.2.2 Synchronizing the plant hierarchy

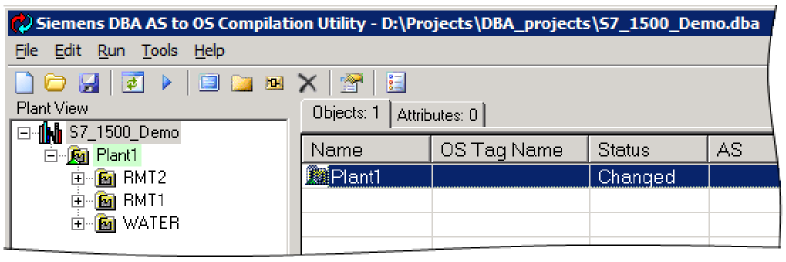
Select the "Synchronize Plant View" command in the DBA project shortcut menu.

Figure 5–10



In this case, the PH is read from the PCS 7 project and created in the DBA, since no hierarchy was present.

Figure 5–11



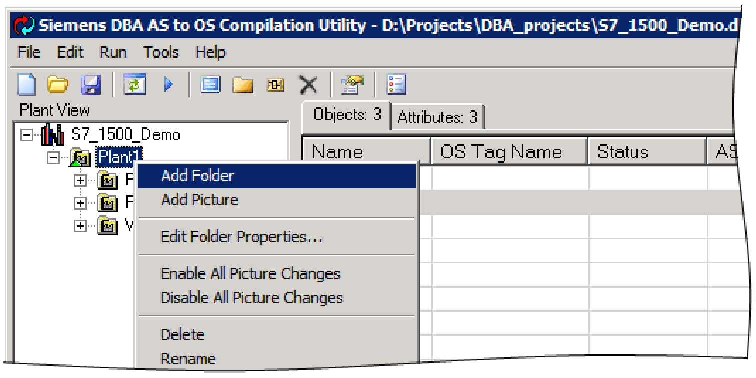
Name	OS Tag Name	Status	AS
Plant1		Changed	

5.2.3 Extending the plant hierarchy

You will now extend the loaded hierarchy to include the hierarchy of the third-party system. By selecting the "Add Folder" command in the shortcut menu, you can create additional hierarchy folders. Every new folder is automatically assigned an OS picture with the same name as the folder.

Add additional hierarchy folders according to your plant structure.

Figure 5–12


Siemens DBA AS to OS Compilation Utility - D:\Projects\DBA_projects\S7_1500_Demo.d
File Edit Run Tools Help
Plant View
S7_1500_Demo
Plant
Objects: 3
Attributes: 3
Name
OS Tag Name
Status
Add Folder
Add Picture
Edit Folder Properties...
Enable All Picture Changes
Disable All Picture Changes
Delete
Rename

If you have not enabled automatic synchronizing, you should synchronize the PH with the OS project again after expanding it in the DBA.

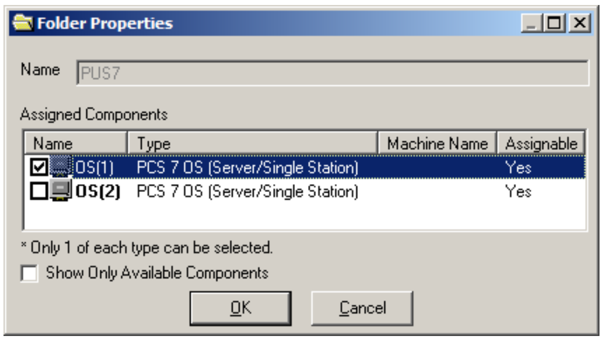
Note PH synchronization can only be used to add folders. If you wish to remove existing folders, you must delete them in the SIMATIC project and in the DBA.

5.2.4 Assigning hierarchy folders to an OS

If multiple OS projects have been created, it is possible to select a specific OS for each hierarchy folder. Lower-level hierarchy folders inherit the setting of higher-level folders.

To assign a hierarchy folder to an OS project, select the "Edit Folder Properties..." command in the shortcut menu of the relevant folder. In the dialog window that then opens, you can select an OS project for the chosen PH folder and its subfolder.

Figure 5–13


Folder Properties
Name
Assigned Components
* Only 1 of each type can be selected.
Show Only Available Components
OK
Cancel

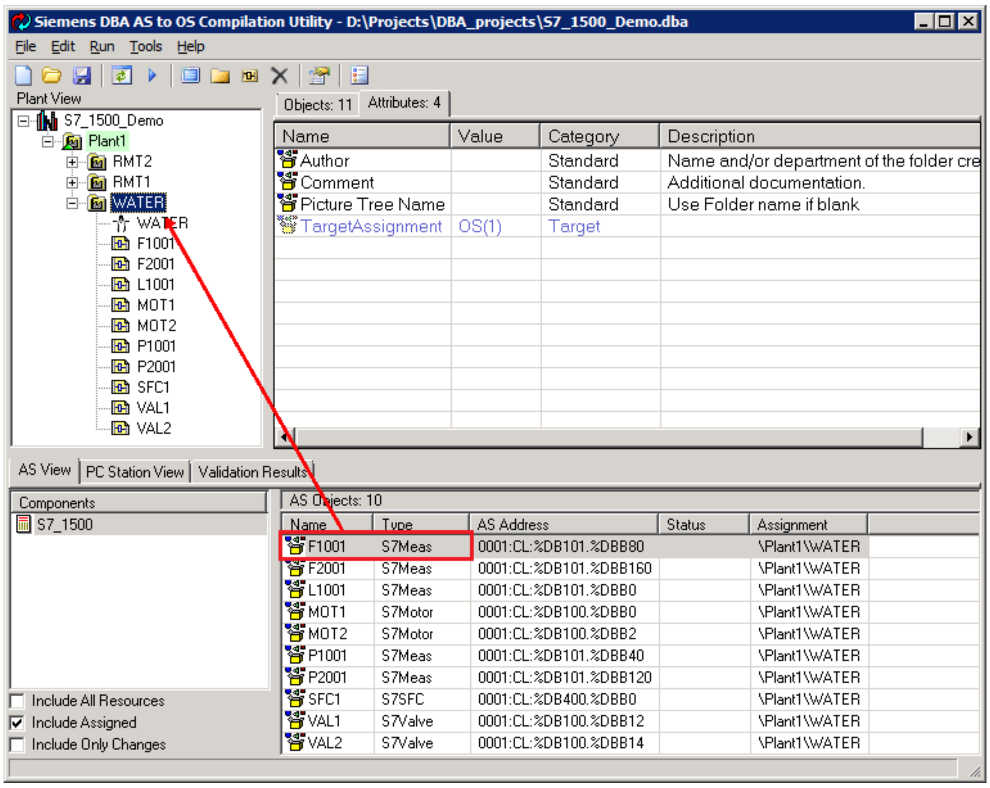
Name: PUS7

	Name	Type	Machine Name	Assignable
☑	OS(1)	PCS 7 OS (Server/Single Station)		Yes
☐	OS(2)	PCS 7 OS (Server/Single Station)		Yes

5.2.5 Assigning AS object instances

There are various ways to assign the process objects of the automation system to the PH. The simplest version is to drag-and-drop the objects from the AS View to the corresponding hierarchy folder.

Figure 5–14

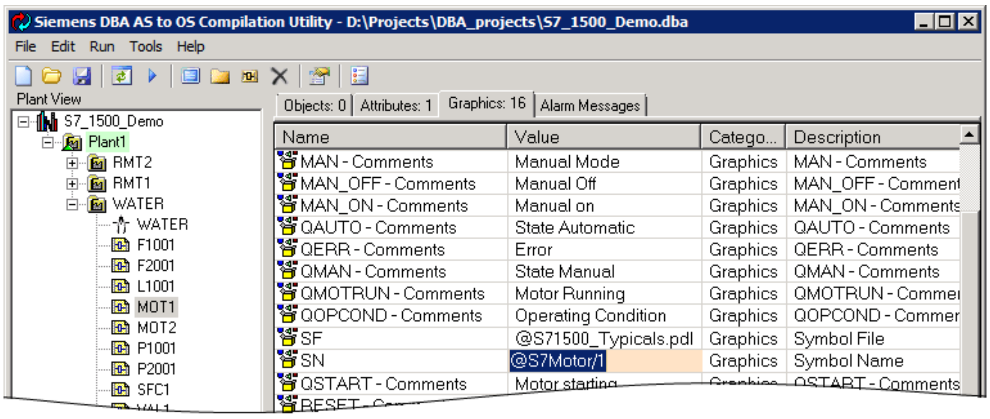


Name	Value	Category	Description
Author		Standard	Name and/or department of the folder cre
Comment		Standard	Additional documentation.
Picture Tree Name		Standard	Use Folder name if blank
TargetAssignment	OS(1)	Target	

Name	Type	AS Address	Status	Assignment
F1001	S7Meas	0001:CL:%DB101.%DBB80		\Plant1\WATER
F2001	S7Meas	0001:CL:%DB101.%DBB160		\Plant1\WATER
L1001	S7Meas	0001:CL:%DB101.%DBB0		\Plant1\WATER
MOT1	S7Motor	0001:CL:%DB100.%DBB0		\Plant1\WATER
MOT2	S7Motor	0001:CL:%DB100.%DBB2		\Plant1\WATER
P1001	S7Meas	0001:CL:%DB101.%DBB40		\Plant1\WATER
P2001	S7Meas	0001:CL:%DB101.%DBB120		\Plant1\WATER
SFC1	S7SFC	0001:CL:%DB400.%DBB0		\Plant1\WATER
VAL1	S7Valve	0001:CL:%DB100.%DBB12		\Plant1\WATER
VAL2	S7Valve	0001:CL:%DB100.%DBB14		\Plant1\WATER

Changes to the plant hierarchy are indicated in the Plant View by a green background. This means that the OS project still needs to be compiled at a later point in time.

Figure 5–15

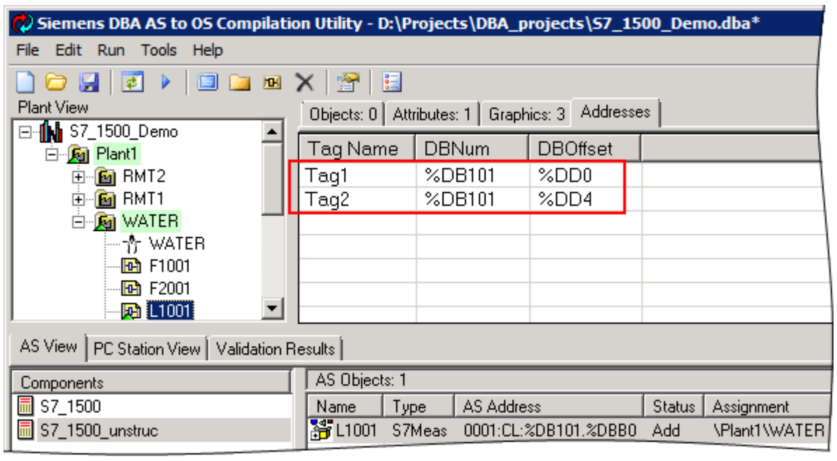


Name	Value	Catego...	Description
MAN - Comments	Manual Mode	Graphics	MAN - Comments
MAN_OFF - Comments	Manual Off	Graphics	MAN_OFF - Comment
MAN_ON - Comments	Manual on	Graphics	MAN_ON - Comments
QAUTO - Comments	State Automatic	Graphics	QAUTO - Comments
QERR - Comments	Error	Graphics	QERR - Comments
QMAN - Comments	State Manual	Graphics	QMAN - Comments
QMOTRUN - Comments	Motor Running	Graphics	QMOTRUN - Commer
QOPCOND - Comments	Operating Condition	Graphics	QOPCOND - Commer
SF	@S71500_Typicals.pdl	Graphics	Symbol File
SN	@S7Motor/1	Graphics	Symbol Name
QSTART - Comments	Motor starting	Graphics	QSTART - Comments

5.2.6 Editing the properties of the objects

After assigning the AS objects to the PH, you can still make changes to a few OS-relevant settings. For example, a different symbol version can be selected, or the alarm class can be changed for messages. If, as mentioned in the chapter "4.3.4 Addressing tab", the addressing parameters of the process variables were assigned individually, an additional tab labeled "Addresses" will still be present. In this tab, it is then possible to set the AS address individually for each variable (Figure 5–16).

Figure 5–16



Siemens DBA AS to OS Compilation Utility - D:\Projects\DBA_projects\S7_1500_Demo.dba*

File Edit Run Tools Help

Plant View: S7_1500_Demo > Plant1 > RMT2, RMT1, WATER > WATER, F1001, F2001, L1001

Objects: 0 | Attributes: 1 | Graphics: 3 | Addresses

Tag Name	DBNum	DBOffset
Tag1	%DB101	%DD0
Tag2	%DB101	%DD4

AS View | PC Station View | Validation Results

Components
S7_1500
S7_1500_unstruc

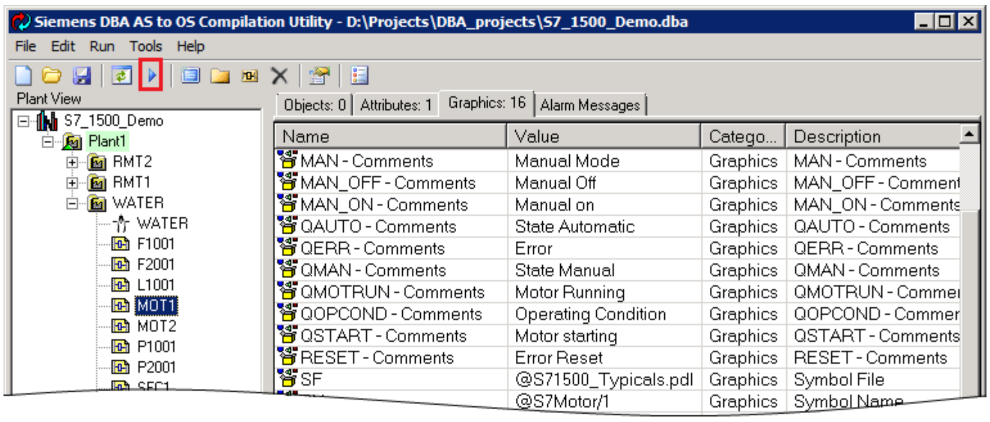
AS Objects: 1

Name	Type	AS Address	Status	Assignment
L1001	S7Meas	0001:CL:%DB101.%DBB0	Add	\Plant1\WATER

5.2.7 Compile OS

With the Compile OS function, all objects configured in the DBA (tags, symbols, messages) are created in the OS project. Click the button with the triangle pointing to the right:

Figure 5–17



Siemens DBA AS to OS Compilation Utility - D:\Projects\DBA_projects\S7_1500_Demo.dba

File Edit Run Tools Help

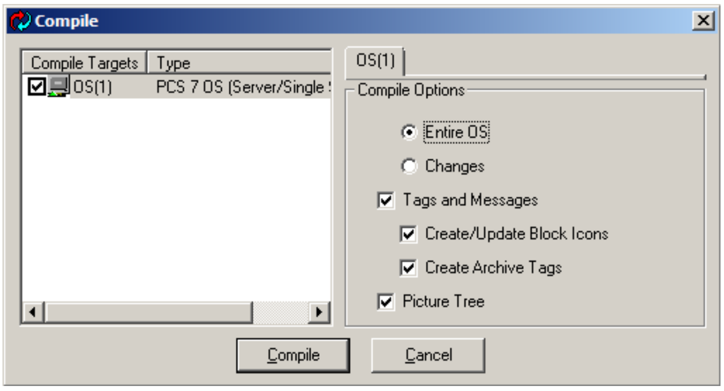
Plant View: S7_1500_Demo > Plant1 > RMT2, RMT1, WATER > WATER, F1001, F2001, L1001, MOT1, MOT2, P1001, P2001, SFC1

Objects: 0 | Attributes: 1 | Graphics: 16 | Alarm Messages

Name	Value	Catego...	Description
MAN - Comments	Manual Mode	Graphics	MAN - Comments
MAN_OFF - Comments	Manual Off	Graphics	MAN_OFF - Comment
MAN_ON - Comments	Manual on	Graphics	MAN_ON - Comments
QAUTO - Comments	State Automatic	Graphics	QAUTO - Comments
QERR - Comments	Error	Graphics	QERR - Comments
QMAN - Comments	State Manual	Graphics	QMAN - Comments
QMOTRUN - Comments	Motor Running	Graphics	QMOTRUN - Commei
QOPCOND - Comments	Operating Condition	Graphics	QOPCOND - Commer
QSTART - Comments	Motor starting	Graphics	QSTART - Comments
RESET - Comments	Error Reset	Graphics	RESET - Comments
SF	@S71500_Typicals.pdl	Graphics	Symbol File
	@S7Motor/1	Graphics	Symbol Name

In the dialog window that then opens, you can still configure the usual SIMATIC Manager settings for an entire compilation or the compilation of changes.

Figure 5–18



Compile

Compile Targets	Type
☑ OS(1)	PCS 7 OS (Server/Single S

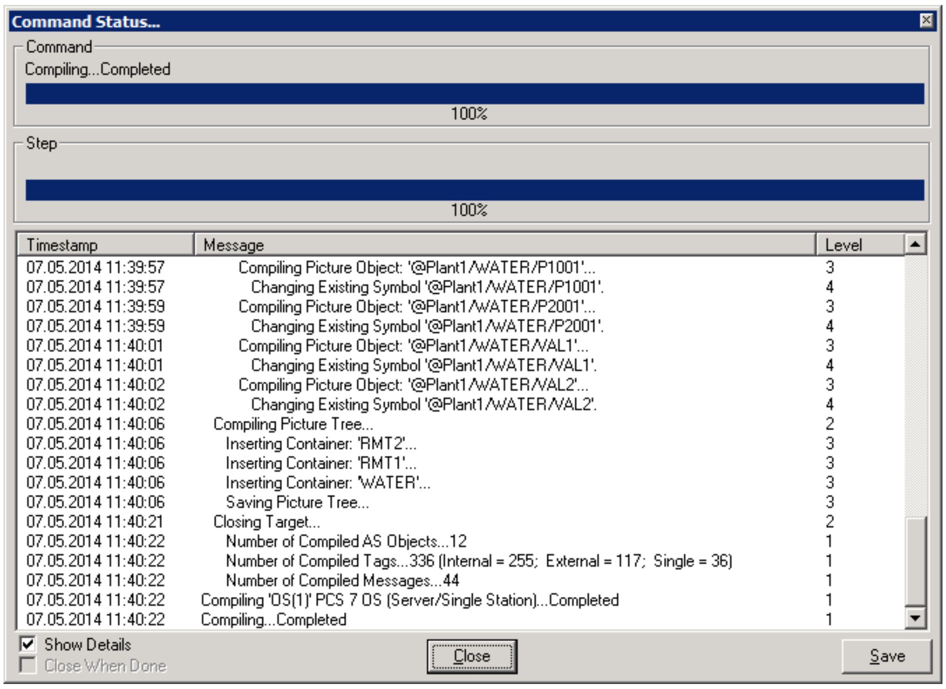
OS(1)

Compile Options

- [✓ Entire OS]
- Changes
- ☑ Tags and Messages
  - ☑ Create/Update Block Icons
  - ☑ Create Archive Tags
- ☑ Picture Tree

You can track the progress of the OS compilation in another dialog window. If the compiling process cannot be executed without errors, you can display the log by activating the "Show Details" option.

Figure 5–19



Command Status...

Command
Compiling...Completed
100%

Step
100%

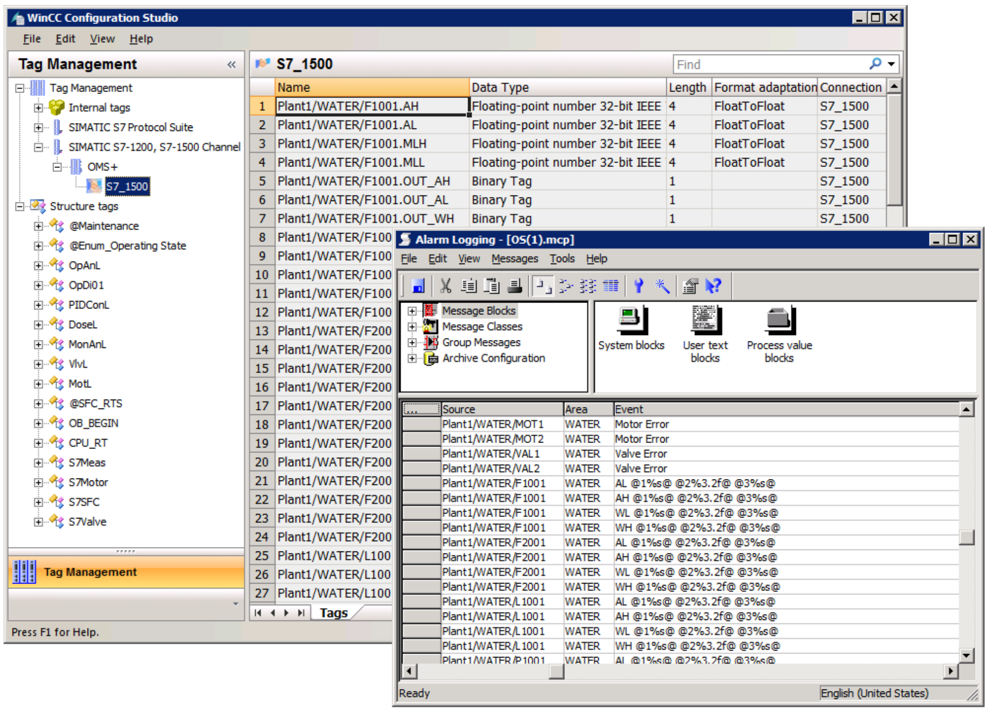
```
Timestamp            Message                                                                          Level
07.05.2014 11:39:57        Compiling Picture Object: '@Plant1/WATER/P1001'...                           3
07.05.2014 11:39:57          Changing Existing Symbol '@Plant1/WATER/P1001'.                            4
07.05.2014 11:39:59        Compiling Picture Object: '@Plant1/WATER/P2001'...                           3
07.05.2014 11:39:59          Changing Existing Symbol '@Plant1/WATER/P2001'.                            4
07.05.2014 11:40:01        Compiling Picture Object: '@Plant1/WATER/VAL1'...                            3
07.05.2014 11:40:01          Changing Existing Symbol '@Plant1/WATER/VAL1'.                             4
07.05.2014 11:40:02        Compiling Picture Object: '@Plant1/WATER/VAL2'...                            3
07.05.2014 11:40:02          Changing Existing Symbol '@Plant1/WATER/VAL2'.                             4
07.05.2014 11:40:06    Compiling Picture Tree...                                                        2
07.05.2014 11:40:06      Inserting Container: 'RMT2'...                                                 3
07.05.2014 11:40:06      Inserting Container: 'RMT1'...                                                 3
07.05.2014 11:40:06      Inserting Container: 'WATER'...                                                3
07.05.2014 11:40:06      Saving Picture Tree...                                                         3
07.05.2014 11:40:21    Closing Target...                                                                2
07.05.2014 11:40:22      Number of Compiled AS Objects...12                                             1
07.05.2014 11:40:22      Number of Compiled Tags...336 (Internal = 255;  External = 117;  Single = 36)  1
07.05.2014 11:40:22      Number of Compiled Messages...44                                               1
07.05.2014 11:40:22  Compiling 'OS(1)' PCS 7 OS (Server/Single Station)...Completed                     1
07.05.2014 11:40:22  Compiling...Completed                                                              1
```

☑ Show Details
☐ Close When Done

5.3 PCS 7 operator station

When you now open the OS project on the Engineering Station, all connections, tags, messages, process pictures and block icons should be created.

Figure 5–20

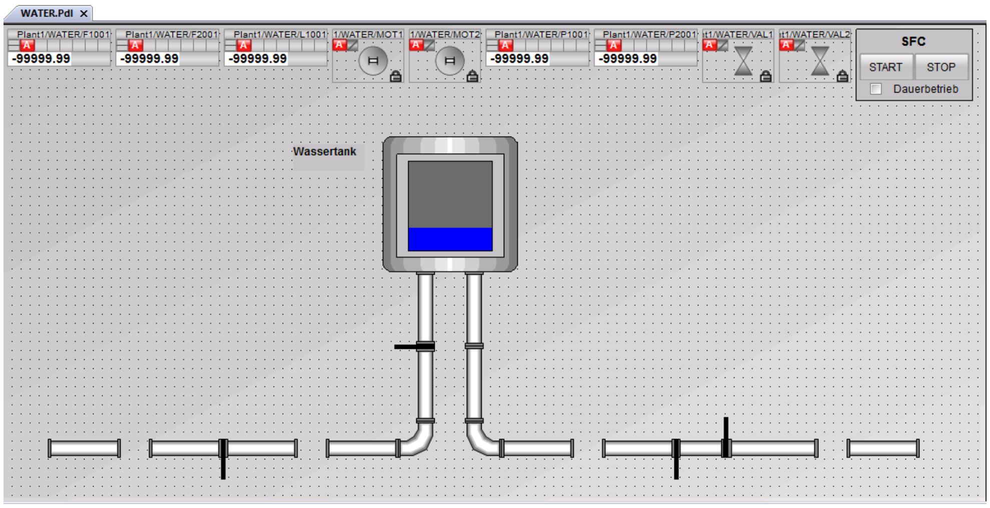


	Name	Data Type	Length	Format adaptation	Connection
1	Plant1/WATER/F1001.AH	Floating-point number 32-bit IEEE	4	FloatToFloat	S7_1500
2	Plant1/WATER/F1001.AL	Floating-point number 32-bit IEEE	4	FloatToFloat	S7_1500
3	Plant1/WATER/F1001.MLH	Floating-point number 32-bit IEEE	4	FloatToFloat	S7_1500
4	Plant1/WATER/F1001.MLL	Floating-point number 32-bit IEEE	4	FloatToFloat	S7_1500
5	Plant1/WATER/F1001.OUT_AH	Binary Tag	1		S7_1500
6	Plant1/WATER/F1001.OUT_AL	Binary Tag	1		S7_1500
7	Plant1/WATER/F1001.OUT_WH	Binary Tag	1		S7_1500
8	Plant1/WATER/F100				
9	Plant1/WATER/F100				
10	Plant1/WATER/F100				
11	Plant1/WATER/F100				
12	Plant1/WATER/F100				
13	Plant1/WATER/F200				
14	Plant1/WATER/F200				
15	Plant1/WATER/F200				
16	Plant1/WATER/F200				
17	Plant1/WATER/F200				
18	Plant1/WATER/F200				
19	Plant1/WATER/F200				
20	Plant1/WATER/F200				
21	Plant1/WATER/F200				
22	Plant1/WATER/F200				
23	Plant1/WATER/F200				
24	Plant1/WATER/F200				
25	Plant1/WATER/L100				
26	Plant1/WATER/L100				
27	Plant1/WATER/L100				

Source	Area	Event
Plant1/WATER/MOT1	WATER	Motor Error
Plant1/WATER/MOT2	WATER	Motor Error
Plant1/WATER/VAL1	WATER	Valve Error
Plant1/WATER/VAL2	WATER	Valve Error
Plant1/WATER/F1001	WATER	AL @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F1001	WATER	AH @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F1001	WATER	WL @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F1001	WATER	WH @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F2001	WATER	AL @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F2001	WATER	AH @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F2001	WATER	WL @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/F2001	WATER	WH @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/L1001	WATER	AL @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/L1001	WATER	AH @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/L1001	WATER	WL @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/L1001	WATER	WH @1%s@ @2%3.2f@ @3%s@
Plant1/WATER/P1001	WATER	AL @1%s@ @2%3.2f@ @3%s@

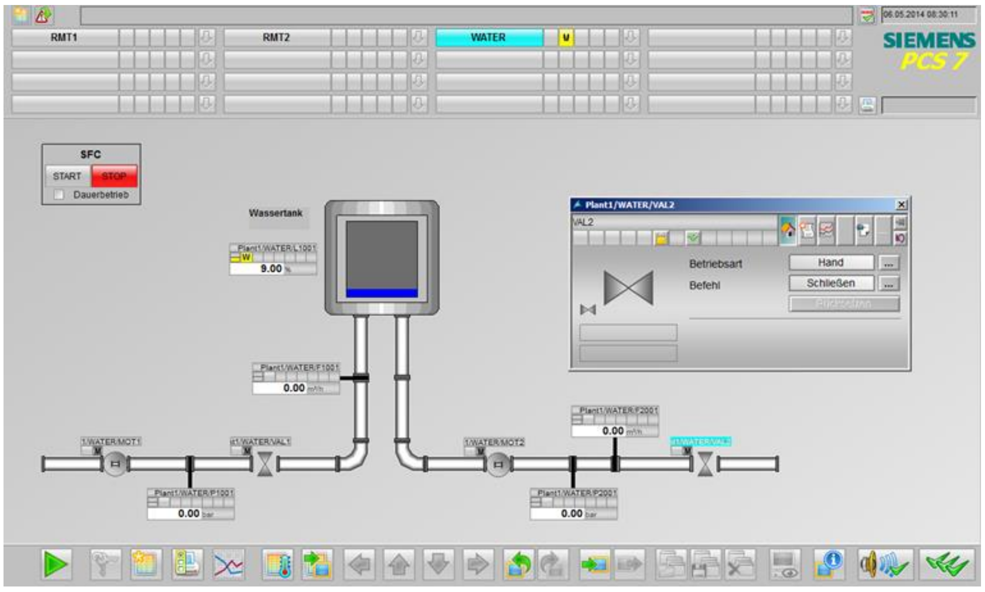
5. With the Graphics Designer, open the process pictures that were created using the DBA. The block icons for the AS objects will have been inserted next to one another.
6. Draw the process picture according to your requirements and move the block icons into the desired position.

Figure 5–21



7. In the SIMATIC Manager, execute the "Load OS" function and start the runtime at the OS server.

Figure 5–22



6 Appendix A

6.1 Skript „CalcS71500Addr“

```
Function CalcS71500Addr(ByVal oTag, ByVal sInstAddr, ByVal sTagAddr)  'V2.0
   Dim sInstAddrTokens, sTagAddrTokens, sTag, sType, sTagAddrOffsetPart
   Dim sDBNum, nDBStartOffset, fDBOffset, sQualifier, nNumDecPlaces
   Dim sAddr

   sTag = GetAttr(oTag,"Na","")
   sType = GetAttr(oTag,"Ty","")

   sInstAddrTokens = SPlit(sInstAddr, ",")
   sTagAddrTokens = SPlit(sTagAddr, ",")

   sDBNum = sInstAddrTokens(0)
   nDBStartOffset = CLng(sInstAddrTokens(1))
   sTagAddrOffsetPart = SPlit(sTagAddrTokens(0),".")

   fDBOffset = CDbl(sTagAddrOffsetPart(0))

   If UBound(sTagAddrTokens) > 0 Then
          sQualifier = sTagAddrTokens(1)
          If Len(sQualifier) = 0 Then
                 sQualifier = VarTypeToS7Type(sType)
          End If
   Else
          sQualifier = VarTypeToS7Type(sType)
   End If

   If UBound(sTagAddrTokens) > 1 Then
          nNumDecPlaces = sTagAddrTokens(2)
          If Len(nNumDecPlaces) = 0 Then
                 nNumDecPlaces = 0
          End If
   ElseIf sType = "BIT" Then
          nNumDecPlaces = 1
   Else
          nNumDecPlaces = 0
   End If

   sAddr = "0001:CL:%DB"
   sAddr = sAddr & sDBNum
   sAddr = sAddr & ".%" & sQualifier
   sAddr = sAddr & FormatNumber(nDBStartOffset + fDBOffset,0, , ,False)

If nNumDecPlaces > 0 Then
   sAddr = sAddr & "." & sTagAddrOffsetPart(1)
End If
   CalcS71500Addr = sAddr
End Function
```

6.2 Skript „CalcS71500ASOAddr“

```
Function CalcS71500ASOAddr(ByVal sInstAddr)
   Dim sInstAddrToken, sAddr

   sInstAddrToken = SPlit(sInstAddr, ",")

   sAddr = "0001:CL:%DB"
   sAddr = sAddr & sInstAddrToken(0)
   sAddr = sAddr & ".%DBB"
   sAddr = sAddr & sInstAddrToken(1)

   CalcS71500ASOAddr = sAddr
End Function
```

7 Appendix B

7.1 Service and support

Industry Online Support

Do you have any questions or need assistance?

Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions and services.

Product information, manuals, downloads, FAQs, application examples and videos – all information is accessible with just a few mouse clicks:

support.industry.siemens.com

Technical Support

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers – ranging from basic support to individual support contracts.

Please send queries to Technical Support via Web form:

support.industry.siemens.com/cs/my/src

SITRAIN – Digital Industry Academy

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that's tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:

siemens.com/sitrain

Service offer

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contracts

You can find detailed information on our range of services in the service catalog web page:

support.industry.siemens.com/cs/sc

Industry Online Support app

You will receive optimum support wherever you are with the "Siemens Industry Online Support" app. The app is available for iOS and Android:

support.industry.siemens.com/cs/ww/en/sc/2067

7.2 Industry Mall

The Siemens Industry Mall is the platform on which the entire siemens Industry product portfolio is accessible. From the selection of products to the order and the delivery tracking, the Industry Mall enables the complete purchasing processing – directly and independently of time and location:
mall.industry.siemens.com

7.3 Literature

Table 7–1

	Subject area	Title
\1\	Siemens Industry Online Support	https://support.industry.siemens.com
\2\	Article download page	https://support.industry.siemens.com/cs/ww/en/view/49740087
\3\	SIMATIC PCS 7 in Industry Online Support	https://support.industry.siemens.com/cs/ww/en/view/63481413
\4\	Product information PCS 7/OPEN OS	https://support.industry.siemens.com/cs/ww/en/view/109750265
\5\	Doenload PCS 7/OPEN OS V9.0 Update 1	https://support.industry.siemens.com/cs/ww/en/view/109781622
\6\	SIMATIC PCS 7 APL Style guide	https://support.industry.siemens.com/cs/ww/en/view/65601446

7.4 History

Table 7–2

Version	Date	Change
V1.0	09/2014	First edition
V1.1	07/2018	Correction in script "CalcS71500Addr"
V1.2	06/2021	Update to PCS 7/OPEN OS V9.0 and further correction in Script "CalcS71500Addr"
V1.3	09/2021	Fixing an error in chapter 6