Integrating a Foxboro DCS with PCS 7/OPEN OS

SIMATIC PCS 7/OPEN OS

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

Note

The Application Examples are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The Application Examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These Application Examples do not relieve you of the responsibility to use safe practices in application, installation, operation and maintenance. When using these Application Examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these Application Examples at any time without prior notice.

If there are any deviations between the recommendations provided in these Application Examples and other Siemens publications – e.g. Catalogs – the contents of the other documents have priority.

We do not accept any liability for the information contained in this document.

Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). The damages for a breach of a substantial contractual obligation are, however, limited to the foreseeable damage, typical for the type of contract, except in the event of intent or gross negligence or injury to life, body or health. The above provisions do not imply a change of the burden of proof to your detriment.

Any form of duplication or distribution of these Application Examples or excerpts hereof is prohibited without the expressed consent of the Siemens AG.

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, solutions, machines, equipment and/or networks. They are important components in a holistic industrial security concept. With this in mind, Siemens’ products and solutions undergo continuous development. Siemens recommends strongly that you regularly check for product updates.

For the secure operation of Siemens products and solutions, it is necessary to take suitable preventive action (e.g. cell protection concept) and integrate each component into a holistic, state-of-the-art industrial security concept. Third-party products that may be in use should also be considered. For more information about industrial security, visit http://www.siemens.com/industrialsecurity.

To stay informed about product updates as they occur, sign up for a product-specific newsletter. For more information, visit http://support.industry.siemens.com.
Table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Task</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>Solution</td>
<td>6</td>
</tr>
<tr>
<td>2.1</td>
<td>Overview</td>
<td>6</td>
</tr>
<tr>
<td>2.2</td>
<td>Description of the core functionality</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>Basics</td>
<td>8</td>
</tr>
<tr>
<td>3.1</td>
<td>Configuration guide</td>
<td>8</td>
</tr>
<tr>
<td>3.2</td>
<td>Editors</td>
<td>11</td>
</tr>
<tr>
<td>3.2.1</td>
<td>PCS 7 DBA AS Node Type Wizard</td>
<td>11</td>
</tr>
<tr>
<td>3.2.2</td>
<td>PCS 7 DBA</td>
<td>12</td>
</tr>
<tr>
<td>3.2.3</td>
<td>PCS 7 DBA Type Editor</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>Linking an existing Foxboro FreelanceDCS</td>
<td>14</td>
</tr>
<tr>
<td>4.1</td>
<td>Acquiring the data</td>
<td>15</td>
</tr>
<tr>
<td>4.2</td>
<td>Generating OS block icons and faceplates</td>
<td>17</td>
</tr>
<tr>
<td>4.3</td>
<td>Determining OPC data</td>
<td>18</td>
</tr>
<tr>
<td>4.4</td>
<td>Creating the AS node with the AS Node Type Wizard</td>
<td>19</td>
</tr>
<tr>
<td>4.4.1</td>
<td>General Tab</td>
<td>19</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Instance Source tab</td>
<td>20</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Connections tab</td>
<td>21</td>
</tr>
<tr>
<td>4.4.4</td>
<td>“Addressing tab”</td>
<td>22</td>
</tr>
<tr>
<td>4.5</td>
<td>Creating AS nodes</td>
<td>23</td>
</tr>
<tr>
<td>4.6</td>
<td>Creating AS object types</td>
<td>25</td>
</tr>
<tr>
<td>4.6.1</td>
<td>Creating a new ASO type</td>
<td>27</td>
</tr>
<tr>
<td>4.6.2</td>
<td>Including default types</td>
<td>28</td>
</tr>
<tr>
<td>4.6.3</td>
<td>Creating tags</td>
<td>29</td>
</tr>
<tr>
<td>4.6.4</td>
<td>Configuring messages</td>
<td>31</td>
</tr>
<tr>
<td>4.6.5</td>
<td>“Attributes” tab</td>
<td>34</td>
</tr>
<tr>
<td>4.7</td>
<td>Creating ASO instances</td>
<td>35</td>
</tr>
<tr>
<td>5</td>
<td>Configuring the PC station using DBA</td>
<td>38</td>
</tr>
<tr>
<td>5.1</td>
<td>Adding a PC Station</td>
<td>38</td>
</tr>
<tr>
<td>5.2</td>
<td>Creating a technological hierarchy</td>
<td>41</td>
</tr>
<tr>
<td>5.2.1</td>
<td>Configuring the Project properties</td>
<td>41</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Synchronizing the plant hierarchy</td>
<td>43</td>
</tr>
<tr>
<td>5.2.3</td>
<td>Extending the plant hierarchy</td>
<td>44</td>
</tr>
<tr>
<td>5.2.4</td>
<td>Assigning a plant hierarchy to an OS</td>
<td>45</td>
</tr>
<tr>
<td>5.2.5</td>
<td>Assigning AS object instances</td>
<td>46</td>
</tr>
<tr>
<td>5.2.6</td>
<td>Editing the properties of the instances</td>
<td>47</td>
</tr>
<tr>
<td>5.2.7</td>
<td>Compiling the OS</td>
<td>48</td>
</tr>
<tr>
<td>5.3</td>
<td>PCS 7 Operator Station</td>
<td>49</td>
</tr>
<tr>
<td>6</td>
<td>Related Literature</td>
<td>51</td>
</tr>
<tr>
<td>7</td>
<td>History</td>
<td>51</td>
</tr>
</tbody>
</table>
1 Task

Introduction

System landscapes that have grown over time often consist of heterogeneous automation technologies. This results in the requirement that devices which contain controllers from competitors 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 do not belong to the spectrum of SIMATIC PCS 7 system components can be integrated into the PCS 7 process control.

PCS 7/OPEN OS makes it possible to exchange data 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 data exchange with the controllers via OPC DA and OPC A&E.

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 database 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 existing third-party controllers into process control using the PCS 7 operator system. This gives operators the option of operating and monitoring the entire system from a single operator system, even if it contains subareas, with controllers from third-party manufacturers.
Overview of the automation task

An Invensys Foxboro DCS is to be operated and monitored using the SIMATIC PCS 7 operator system. The system is integrated into the PCS 7 landscape using SIMATIC PCS 7/OPEN OS and the OPC server for Foxboro DCS by Matrikon.

For more information about Matrikon's OPC server, see the manufacturer's website.

http://www.matrikonopc.com/
2 Solution

2.1 Overview

Using the PCS 7 SIMATIC PCS 7/OPEN OS option, this application example will show you how you can effectively integrate third-party controllers into the PCS 7 system landscape.

2.1 Overview

The example contained in this document describes how to integrate an existing plant section with an Invensys Foxboro DCS into an existing PCS 7 landscape. The system is integrated using the OPC channel of the PCS 7 operator station and the manufacturer-independent OPC server by Matrikon.

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
- A step by step guide to configuration using PCS 7/OPEN OS
- Common 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
- Process visualization using PCS 7 OS engineering
- A basic knowledge of Matrikon’s OPC server
- A basic knowledge of the third-party system

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.

Amongst other things, DBA generates the following OS data:

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

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

- SIMATIC 505
- Allen Bradley
- Mitsubishi

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

In this application example, we will link a third-party system manufactured by Invensys using OPC and Matrikon’s manufacturer-independent OPC server.
3 Basics

3.1 Configuration guide

In this chapter, you will find 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

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Collecting data</td>
</tr>
<tr>
<td></td>
<td>Before you start configuring, you need the following information in order to operate and monitor the third-party system:</td>
</tr>
<tr>
<td></td>
<td>• Access type (S7 connection, OPC DA, OPC A&amp;E, …)</td>
</tr>
<tr>
<td></td>
<td>• Network addresses (OPC server, automation systems)</td>
</tr>
<tr>
<td></td>
<td>• Syntax of tag addresses</td>
</tr>
<tr>
<td></td>
<td>• Tags, interfaces, alarms</td>
</tr>
<tr>
<td></td>
<td>• Project file path of the target operator system</td>
</tr>
<tr>
<td></td>
<td>• Plant hierarchy</td>
</tr>
<tr>
<td>2.</td>
<td>Generating OS block icons and faceplates</td>
</tr>
<tr>
<td></td>
<td>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.</td>
</tr>
<tr>
<td></td>
<td>For more information about creating icons and faceplates, refer to the “SIMATIC SIMATIC PCS 7 Process Control System APL Style Guide” manual.</td>
</tr>
<tr>
<td></td>
<td>The item entitled “Creating technological functions with PCS 7 TeleControl or PCS 7/OPEN OS” shows an example of creating a block using the DBA Type Editor and WinCC.</td>
</tr>
<tr>
<td></td>
<td>Note</td>
</tr>
<tr>
<td></td>
<td>You must create the symbol displays and the faceplates before compiling the plant hierarchy for the first time.</td>
</tr>
<tr>
<td>3.</td>
<td>Creating AS node types</td>
</tr>
<tr>
<td></td>
<td>You create 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.</td>
</tr>
<tr>
<td>4.</td>
<td>Creating AS nodes</td>
</tr>
<tr>
<td></td>
<td>In the AS View of the DBA, you create an instance of the generated AS node type. Depending on the settings, you may still be able to adapt the connection parameters and assign an instance name.</td>
</tr>
<tr>
<td>5.</td>
<td>Creating AS object types</td>
</tr>
<tr>
<td></td>
<td>The AS objects (ASO) represent the objects that are present in the automation system, e.g. motors, valves, closed-loop controllers, analog and digital values, and they are linked to the AS node type.</td>
</tr>
<tr>
<td></td>
<td>The ASO types can contain tag addresses, the tag format, tag attributes, and tag alarms and messages.</td>
</tr>
</tbody>
</table>
## 3 Basics
### 3.1 Configuration guide

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
</table>
| 6.   | **Creating ASO instances**  
In this step, you 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 define the OS project to which the data will be compiled. In addition, you set here 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**  
In this step, you drag the ASO instance to the corresponding hierarchy folder in the Plant View. In this way, you create the relation of the AS objects to the operator system.  
Depending on the type of configuration, you may still be able to assign or adjust tag addresses and attributes.  
| 10.  | **Compiling the DBA project**  
This process 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 runtime**  
If all the sequences have been executed properly, the OS project can be loaded and 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
3 Basics

3.2 Editors

PCS 7/OPEN OS includes the following editors:

- 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 generate the AS node types using the AS Node Type Wizard. The following information is stored:

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

You can start the AS Node Type Wizard at "Start > Programs > Siemens Automation > SIMATIC > DBA > PCS 7 DBA ASNodeWizard".

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

![Figure 3-2](image-url)
3.2 Editors

3.2.2 PCS 7 DBA

The DBA Editor is the main configuration software of PCS 7/OPEN OS. Using the DBA Editor, you configure the AS structures and connect them to the PCS 7 operator system.

This includes the following steps:

- Creating AS nodes and instances of AS objects (ASO)
- Creating a PC Station and connecting it to the OS project
- Creating a technological hierarchy and connecting it to the SIMATIC project
- Assigning AS object instances of the technological hierarchy and parameterizing them
- Compiling the OS using the DBA Editor

You start the DBA Editor by clicking on "Start > Programs > Siemens Automation > SIMATIC > DBA > PCS 7 DBA".

You can find detailed information in the "PCS 7 Open OS DBA" manual. The manuals are available on your system when you install OPEN OS.

Figure 3-3
3 Basics

3.2 Editors

3.2.3 PCS 7 DBA Type Editor

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

- Tags possibly with addresses
- Lists
- Alarms and messages
- Attributes
- Runtime scripts

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

You can find detailed information in the "PCS 7 Open OS DBA Type Editor" manual. The manuals are available on your system when you install OPEN OS.
4 Linking an existing Foxboro FreelanceDCS

The following chapters describe how to integrate an Invensys system with PCS 7 using OPEN OS.

Drawing up the application example was based on Matrikon's OPC server for simulation and testing. The technological functions of the automation system were mapped using an alias.

Configuration of a real system using the Matrikon OPC server for Foxboro is virtually identical. However, you do not need to use an alias when carrying out configuration on an OPC server, since addressing in DBA is possible directly.

In this application example, we only configured one analog measuring point and one pump drive.

Figure 4-1

- Fill level sensor (1)
- Pump drive (2)
4.1 Acquiring the data

In this application example, we used a few proprietary functions that were only configured as aliases in the OPC server. If you use the existing default blocks in your Foxboro system, use the documentation of these blocks to help you.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pump drive</td>
<td>DRV</td>
<td>Drive</td>
</tr>
<tr>
<td>Fill level sensor</td>
<td>ANA</td>
<td>Analog value acquisition</td>
</tr>
</tbody>
</table>

**Pump drive**

The following aliases were parameterized for the pump drive:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCK</td>
<td>BOOL</td>
<td>IN</td>
<td>RW Interlock</td>
</tr>
<tr>
<td>MAN_ON</td>
<td>BOOL</td>
<td>IN</td>
<td>RW Start drive manually (pulse)</td>
</tr>
<tr>
<td>MAN_OFF</td>
<td>BOOL</td>
<td>IN</td>
<td>RW Stop drive manually (pulse)</td>
</tr>
<tr>
<td>FB</td>
<td>BOOL</td>
<td>IN</td>
<td>RW Feedback (0=OFF, 1=ON)</td>
</tr>
<tr>
<td>CMD_ON</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO Control command</td>
</tr>
<tr>
<td>AUT</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO Operating mode (0=Manual, 1=Automatic)</td>
</tr>
<tr>
<td>ST_ON</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO Status of drive ON</td>
</tr>
<tr>
<td>ST_OFF</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO Status of drive OFF</td>
</tr>
<tr>
<td>MON_FB</td>
<td>BOOL</td>
<td>Param</td>
<td>RW Feedback monitoring</td>
</tr>
<tr>
<td>MON_T</td>
<td>INT</td>
<td>Param</td>
<td>RW Feedback time in ms</td>
</tr>
</tbody>
</table>
4 Linking an existing Foxboro FreelanceDCS

4.1 Acquiring the data

**Analog value acquisition**

The following aliases were parameterized for analog value acquisition:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Type</th>
<th>Type</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAL</td>
<td>REAL</td>
<td>OUT</td>
<td>RO</td>
<td>Measured value</td>
</tr>
<tr>
<td>HL</td>
<td>REAL</td>
<td>IN</td>
<td>RW</td>
<td>Upper limit of measured value acquisition</td>
</tr>
<tr>
<td>LL</td>
<td>REAL</td>
<td>IN</td>
<td>RW</td>
<td>Lower limit of measured value acquisition</td>
</tr>
<tr>
<td>HLAL</td>
<td>REAL</td>
<td>IN</td>
<td>RW</td>
<td>Alarm high</td>
</tr>
<tr>
<td>HLWN</td>
<td>REAL</td>
<td>IN</td>
<td>RW</td>
<td>Warning high</td>
</tr>
<tr>
<td>LLWN</td>
<td>REAL</td>
<td>IN</td>
<td>RW</td>
<td>Warning low</td>
</tr>
<tr>
<td>LLAL</td>
<td>REAL</td>
<td>IN</td>
<td>RW</td>
<td>Alarm low</td>
</tr>
<tr>
<td>QAH</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO</td>
<td>Alarm high (signal)</td>
</tr>
<tr>
<td>QWH</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO</td>
<td>Warning high (signal)</td>
</tr>
<tr>
<td>QWL</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO</td>
<td>Warning low (signal)</td>
</tr>
<tr>
<td>QWA</td>
<td>BOOL</td>
<td>OUT</td>
<td>RO</td>
<td>Alarm low (signal)</td>
</tr>
<tr>
<td>Unit</td>
<td>String</td>
<td>Param</td>
<td>RW</td>
<td>Unit</td>
</tr>
</tbody>
</table>

*Table 4-3*
4.2 Generating OS block icons and faceplates

You can create 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 creating user-defined icons and faceplates can be found in the "SIMATIC Process Control System PCS 7 APL Style Guide" manual:

The application example entitled "Creating technological functions with PCS 7 TeleControl or PCS 7/OPEN OS" describes how to create the technological functions using DBA and the associated faceplate using a TeleControl type as an example.

With the collected data from the Foxboro automation system project, you can generate the dynamic objects in the WinCC icons and the faceplates with the correct tags. Later, when you generate the OS from the OPEN OS project, the block icons are linked to the correct tags of the OPC server.

For the Typicals picture, use a meaningful name in accordance with PCS 7 naming conventions. Choose the name "@Foxboro_Typicals.pdl", for example, for the icon picture.

Create the symbol displays and faceplates for all of the required technological functions, e.g.:

- Drive: FoxDRV
- Analog value acquisition: FoxANA

Figure 4-2
4.3 Determining OPC data

The following data is required for configuring with PCS 7/OPEN OS at a later stage:

- PC name on which the OPC server is started
- Name of the OPC server to be connected
- Tag syntax
- Triggering conditions for messages

To determine the OPC server data, you can, for example, use the SIMATIC NET OPC client “OPC Scout V10” or the Matrikon OPC Explorer. The OPC Scout can be found on a PC with a SIMATIC NET installation in the Start menu under the following path: "Start > All Programs > Siemens Automation > SIMATIC > SIMATIC NET > OPC Scout V10". At installation of the OPC server for Foxboro, the Matrikon OPC Explorer is also installed.

After starting the OPC client, you can search the entire network for available OPC servers. In this example, the “Matrikon.OPC.Foxboro.1” OPC server is linked to the OPC Scout:

**OPC Data Access**

With the OPC Scout, you can test the connection to the OPC server and view individual tag values. The output window also shows the syntax which is used by the OPC server to form the tag string. In the case of the input tags for the tank fill level, this is "ANA1.VALUE". In this case, a “.” is used as the separator for construction of the tags. Please note that this may differ from OPC server to OPC server. The OPC server also distinguishes between upper case and lower case.

Figure 4-3
4.4 Creating the AS node with the AS Node Type Wizard

The AS node type is created with the aid of the PCS 7 DBA AS Node Type Wizard. To execute the wizard, choose in the Start menu "Start > Programs > Siemens Automation > SIMATIC > DBA > PCS 7 DBA ASNodeWizard". Amongst other things, this AS node type type includes information about the connection to the AS and the tag syntax.

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.

4.4.1 General Tab

On 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 at compilation.

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

![DBA AS Node Type Wizard](image)
4.4 Creating the AS node with the AS Node Type Wizard

4.4.2 Instance Source tab

This document describes the "DBA Integrated Instances" option.

3. Select the "DBA Integrated Instances" option.

You can find a description of the use of XML instances in the item entitled "How do you create an XML input file for automatic creation of instances in DBA?".
4.4 Creating the AS node with the AS Node Type Wizard

4.4.3 Connections tab

In this example, we use the OPC channel to connect to the third-party CPU. Select the settings below in the “Channel & Channel Group” area:

4. Select the “OPC channel” option.
5. Enter the name of the computer on which the OPC server has been started.
6. Enter the name of the OPC server (DA).

Matrikon does not make available an OPC A & E server for Foxboro. In the "Connection" area you can enter a fixed connection name or use the following tags for dynamization in the DBA:

7. %SystemName%: The name of the AS node type is used ("General" tab)
8. %ASName%: The name of the AS node is used that is entered in the DBA at instantiation.

In the OS project, the connection is created under this name.

Figure 4-6
4.4 Creating the AS node with the AS Node Type Wizard

4.4.4 "Addressing tab"

On the Addressing tab, you can specify the structure of tag addressing that is used at OS compilation of the project. In the case of the OPC server, the AS address is not relevant and is ignored by the DBA.

Here, the DBA offers the option of configuring a wide range of different ways to parameterize tag addresses. 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 variables can be used to parameterize the tag addresses:

- `%ASName%` – name of the AS node
- `%ASOName%` – name of the AS instance
- `%Instance%` – Instance address that is allocated during creation of the instance
- `%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 at 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

The program is structured in the same way as was previously specified when determining the third-party system data. The tags can be derived from the used function blocks and are the same for each instance.

In WinCC, the address string for the OPC tags is composed of the following components:

- Object name
- Access path
- Data type

The WinCC tags are formed from the AS object instance and the tag name. A period "." is used to separate them both. The access path is already stored in the configured connection and therefore remains empty. The data type is appended as a numeric value.

This yields the following address string in WinCC of the type "VAL" process values of analog value acquisition "ANA01":

- "ANA01.VAL",".",4

For DBA to be able to structure the tag addresses correctly, the syntax below is used:

- `%ASOName%.%TagName%`
### 4.5 Creating AS nodes

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. To create the AS nodes in DBA, proceed as follows:

1. Open a new or existing project in DBA.
2. On 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 an AS node type that has been created using the "AS Node Type Wizard". In this case, it is the type that was created in Chapter 4.4 Creating the AS node with the AS Node Type Wizard for the Foxboro automation system.
4. Assign a name for the station.
5. If you have allowed the subsequent change of connection parameters during the creation of the AS node type, you can configure these settings in the "Connection" tab.

The new AS node has now been created in the DBA AS View.
4.6 Creating AS object types

The AS object types in the DBA map the function blocks used in the AS. An ASO type can be instantiated in the DBA as often as you like. Changes made to a type have an effect on all the 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 the different blocks that are necessary.

The data that was determined in Chapter "4.1 Acquiring the data" is the basis for creating the ASO types. An ASO type can contain the following objects:

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

You can start the DBA Type Editor by selecting the "Edit AS Object Types..." command from the AS node’s shortcut menu.

Figure 4-10

Note

The AS object parameters are assigned to one AS node; however, they are linked to the project. If you want to reuse the objects in another DBA project, 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 that are needed to handle alarms or to handle block icons or faceplates. The attributes can be included with any type for which these functionalities are intended.

The following sections contain more detailed descriptions of the “Tags”, “Messages”, “Attributes”, “Includes”, “Docs”, and “OS Runtime” tabs. On the “Docs” tab you can integrate descriptions for the created ASO types in HTML format. On the “Runtime Scripts” tab, you can add scripts that the TC server executes. You can find more detailed information about this topic in the “PCS 7 Open OS DBA Type Editor” manual.
4 Linking an existing Foxboro FreelanceDCS

4.6 Creating AS object types

4.6.1 Creating a new ASO type

1. To create a new ASO 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 reference to the object in the AS.
   - HMI Type Name – Choose a name that creates a reference to the object in the OS. Note that this must be the same name that you assigned to the HMI symbol.

   Figure 4-12

   ![DBA Type Editor](image)

   **Note**

   The symbol name is formed using a function. At instantiation of the type, the value "@FoxANA/1" is generated from the name "FoxANA". In the stated Typicals picture, an object with the object name ",@FoxANA/1" must be present at compilation of the OS.
4 Linking an existing Foxboro FreelanceDCS

4.6 Creating AS object types

4.6.2 Including default types

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

These types are already included by default in all new types:

- CommonMembers
- SymbolAndFaceplateExtendedAttributes

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

2. Switch to the "Includes" tab.
3. Select the "AlarmGroupDisplaySupport" type in the left-hand window.
4. Include the type by clicking the right arrow button.

Figure 4-13

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.
4.6.3 Creating tags

The tags form the interface from the OS to the OPC server. The tags of the selected type are displayed on the "Tags" tab of the DBA Type Editor.

5. Switch to the "Tags" tab.
6. Open the Tag Editor by clicking on the "Add" pushbutton (1).

Figure 4-14

On the "General" tab of the Tag Editor, you can enter the following data:

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

The OPC tag address is composed of the measuring point name, the tag name and the data type used. In the case of OPC connections, entries in the address field of the Tag Editor are ignored for construction of the tag addresses because the address syntax has been defined as %ASOName%.%TagName% without using this address field.
4 Linking an existing Foxboro FreelanceDCS

4.6 Creating AS object types

On the "Advanced" tab of the Tag Editor, 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 4-15

7. After parameterizing the tags, click on the "OK" pushbutton to create the type tag and close the dialog box.
4.6.4 Configuring messages

On 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

8. Switch to the "Messages" tab.
9. Open the Message Editor by clicking on the “Add” pushbutton (2).

Figure 4-16

```
<table>
<thead>
<tr>
<th>Parent Type</th>
<th>Name</th>
<th>Display Name</th>
<th>Number</th>
<th>Class</th>
<th>Type</th>
<th>Enabled</th>
<th>Priority</th>
<th>Tag</th>
<th>Trigger Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>FANA</td>
<td>GAH</td>
<td>Alarm High</td>
<td>1</td>
<td>-Alarm</td>
<td>1</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>High Limit Alarm</td>
</tr>
<tr>
<td>FANA</td>
<td>GWH</td>
<td>Warning High</td>
<td>2</td>
<td>-Warning</td>
<td>18</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>High Limit Warning</td>
</tr>
<tr>
<td>FANA</td>
<td>GWL</td>
<td>Warning Low</td>
<td>2</td>
<td>-Warning</td>
<td>18</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>Low Limit Warning</td>
</tr>
<tr>
<td>FANA</td>
<td>GAL</td>
<td>Alarm Low</td>
<td>1</td>
<td>-Alarm</td>
<td>1</td>
<td>Yes</td>
<td>0</td>
<td>0</td>
<td>Low Limit Alarm</td>
</tr>
</tbody>
</table>
```

"General" tab

On the "General" tab, set the following parameters:

- Name
- Displayed name
- Message class
- Priority
- Message text
4 Linking an existing Foxboro FreelanceDCS

4.6 Creating AS object types

Figure 4-17

Associated message values are integrated using the control commands you are familiar with from PCS 7, e.g. associated value 1 of type floating point number: \@1%3.2f\@.

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

Tags tab

On the "Tags" tab, you define the trigger tag of the message. You can choose all of the configured tags and specify a value or a function.

Figure 4-18
4 Linking an existing Foxboro FreelanceDCS

4.6 Creating AS object types

"Process Vars" tab

On this tab, you configure existing tags that you want to use as associated values in your message texts. The system provides 10 fields. You can select any of the tags you created when creating the ASO type.

Figure 4-19

```
<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P1:</td>
<td>VALUE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2:</td>
<td>VALUE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P9:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P10:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```

"Free Vars" tab

On the "Free Vars" tab, you can configure continually recurring text blocks, for example. You can find further information in the WinCC Alarm Logging description.

10. After parameterizing the message, click on the "OK" pushbutton to create the message of the type and close the dialog box.
4.6.5 "Attributes" tab

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

By default, the start values (HL#Start, LL#Start, VALUE#Units) are assigned to the "Graphics" category. To provide a better overview, you can change the values for "Category" and "Display Category" to the value "Attributes". This means that at later parameterization of the instance, the system will display these properties on the "Attributes" tab and not on the "Graphics" tab.

Assignment of the category does not affect functioning of the block, however.
4.7 Creating ASO instances

The ASO types that you created in the previous chapter map the function blocks of the Foxboro system. You can set up any number of instances from these ASO types.

1. Open the AS node shortcut menu and select the "Create New AS Object..." command (1).

2. 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

Note
The OPC tag addresses are generated as follows, depending on how they were configured in the DBA AS Node Type Wizard:
"<ASOName>.<TagName>", ",", <data type>

For example, the tag string for the "VALUE" tag of type Float of analog value measuring point "FANA" with instance name "ANA1" would look like this:
"ANA1.FANA", ",", 4
3. Carry out these steps for all of the configured functions in the AS that you want to monitor on the OS.

Fig. 4-22

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

With the "DBA Object Inspector", you can read out information about which data was created on the OS. You can, for example, verify here whether the address syntax is structured correctly.

The command to open the "DBA Object Inspector" can be found in the shortcut menu of an AS node or an ASO instance.

Figure 4-23
If you make changes to an ASO type later, you must then update the Controller objects.

By calling the "Update Controller Objects" (2) function from the AS node's shortcut menu, you can update all the instances of the AS objects.

**Figure 4-24**

Configuration of the AS part for the Foxboro subsystem is now complete. In the next chapter, you will learn how to configure the OS part with DBA.
5 Configuring the PC station using DBA

The PC station in the DBA Editor is an interface to the operator station 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 with an operator system and a plant hierarchy. This could be an already finished project that is extended to a plant section with a third-party system, or a newly created 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 (1).
2. Select the "Add PC Station" command in the DBA project shortcut menu (2).

3. Give the PC station the same name as your OS (3).
4. Also insert the computer name of the OS (4) on the "Computer Name" row in the "Value" column.

Figure 5-1

Figure 5-2
5 Configuring the PC station using DBA

5.1 Adding a PC Station

5. Select "Add Application" (5) from the shortcut menu of the PC station. After this, the system opens the "Select an Application Type" dialog box.

6. Select project type "PCS 7 OS (Server/Single Station)" (6) and confirm your selection by clicking on the "OK" pushbutton. Give the application the name of your corresponding OS project in the SIMATIC Manager, (e.g.: OS (1)).

7. Link the DBA project to the PCS 7 OS project by entering the path to the OS project on the “Offline MCP File” row in the “Value” column (7). You can enter the path string by using a file selection dialog to help you.
5 Configuring the PC station using DBA

5.1 Adding a PC Station

8. Update the project data in the DBA by selecting the "Refresh OS Cache" function in the shortcut menu (8). This function opens the PCS 7 OS project, reads the alarm class, alarm type and Tag Logging archive information, and saves all of this information in the DBA project.

Figure 5-5

9. Select a Tag Logging archive (9). The Tag Logging archive must have been created in the OS project.

Figure 5-6
5.2 Creating a technological hierarchy

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

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

In this example, an existing PH will be extended.

5.2.1 Configuring the Project properties

1. Click the "Change Project Properties" button (1).

   Figure 5-7

2. On the "General" tab, define the project and the project path with which you intend to synchronize the PH.

   Figure 5-8
Using the "Automatic Synchronize with Simatic Plant View" option, you can trigger automatic synchronization at each compiling operation. It is, however, adequate to create the TH once and then synchronize it manually.

3. On the "Hierarchy" tab, configure the PH settings (2). Choose the same settings here as the ones of the PCS 7 project in SIMATIC Manager (3).

4. Click the "OK" button to confirm the settings.
Synchronizing the plant hierarchy

5. Select the "Synchronize Plant View" (1) command in the DBA project shortcut menu.

The system reads the PH from the PCS 7 project and creates it in the DBA (2).
5 Configuring the PC station using DBA

5.2 Creating a technological hierarchy

5.2.3 Extending the plant hierarchy

In this step, you extend the plant hierarchy of the PCS 7 plant by adding the hierarchy of the third-party system.

6. Select the "Add Folder" command (1) in the shortcut menu to create additional hierarchy folders that match the plant structure of the Foxboro system.

Figure 5-12

In each new folder, the system automatically creates an OS picture with the name of the folder.

You can delete the process pictures at the lower hierarchy levels if you do not need them. Be sure to keep the plant hierarchy settings in mind. If you have not enabled automatic synchronizing, synchronize the PH with the PCS 7 project again after expanding it in DBA.

Note

PH synchronization can only be used to add folders. If you want to remove existing folders, you must delete them in the SIMATIC project and in the DBA before carrying out synchronization again.
5.2.4 Assigning a plant hierarchy to an OS

If you have created several OSes in your project, you can assign a different OS to each hierarchy folder. Lower-level hierarchy folders inherit the setting of higher-level folders.

7. Highlight a hierarchy folder choose the "Edit Folder Properties..." command in the shortcut menu or click on the corresponding pushbutton (1) on the toolbar. The system opens the "Folder Properties" dialog box.

8. Choose an OS for the highlighted PH folder.

Figure 5-13
5 Configuring the PC station using DBA

5.2 Creating a technological hierarchy

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 way is to drag-and-drop the objects from the AS View (1) to the corresponding hierarchy folder.

9. Drag all AS objects to the designated hierarchy folder. The 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 5-14

![Diagram showing the process of assigning AS object instances to the hierarchy folder.](image)
5 Configuring the PC station using DBA
5.2 Creating a technological hierarchy

5.2.6 Editing the properties of the instances

Each AS object has attributes assigned to it which determine its behavior at display in the OS. They can have values pre-assigned to them and can be edited in DBA for each instance if it has been provided appropriately in the type.

10. Edit the attributes by choosing one of the instances that has been assigned to the hierarchy and changing the parameters on the following tabs:
   - “Logged Tags” (1): Configuring the Tag Logging parameters
   - “Messages” (2): Configuring the alarms and messages
   - “Graphics” (3): Configuring the symbol picture and symbol variant
   - “Attributes” (4): Configuring the start values or unit

Figure 5-15

Integrating a Foxboro DCS with PCS 7/OPEN OS
Entry ID: 49740087, V1.0, 12/2015
5 Configuring the PC station using DBA

5.2 Creating a technological hierarchy

5.2.7 Compiling the OS

The Compile OS function creates in the OS project all the objects that are configured in the DBA like tags, symbols, and messages.

11. Click on the “Start Compilation” pushbutton (1). In a similar way to the SIMATIC Manager, you can then choose a few options for the compile process (2).

Figure 5-16

12. Start the compile process by clicking the “Compile” button (3). If the compile process cannot be executed in an error-free way, you can view the log using the “Show Details” option (4).

Figure 5-17
5.3 PCS 7 Operator Station

1. Open the OS project on the ES. If all the described steps have been carried out, the system must have created all the links (1), tags (2), messages (3), process pictures and block icons.

Figure 5-18
2. Open all the process pictures created by DBA.
3. Draw the process pictures according to your requirements and move the icons that DBA has created to their intended positions.

4. When all work has been completed, transfer the project to the OS using the SIMATIC Manager and start OS Runtime there.

   If the OPC connection has been established, the process of the third-party system is displayed on the OS and it can be operated there.
6 Related Literature

Table 6-1

<table>
<thead>
<tr>
<th>Topic</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1
Siemens Industry Online Support                                      | http://support.automation.siemens.com     |
| 
2
Download page of this entry                                           | https://support.industry.siemens.com/cs/ww/en/view/49740087 |
| 
3
SIMATIC PCS 7 Overview                                                | https://support.industry.siemens.com/cs/ww/en/view/63481413 |
| 
4
| 
5
SIMATIC PCS 7/OPEN OS V8.0+SP1 Download                               | https://support.industry.siemens.com/cs/ww/en/view/75781230 |
| 
6
| 
7
Creating technological functions with TeleControl and OPEN OS for a consistent Process Visualization in PCS 7 | https://support.industry.siemens.com/cs/ww/en/view/109478763 |
| 
8
Matrikon OPC Solutions                                                | http://www.matrikonopc.com/               |

7 History

Table 7-1

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>11/2015</td>
<td>First edition</td>
</tr>
</tbody>
</table>