Example Blocks for WinCC V7 and STEP 7 V5
(for S7-300 in SCL and S7-400 with CFC, SCL)

WinCC

Configuration Example • April 2010
Warranty, Liability and Support

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 sound 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 this application example 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”). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health. The above provisions do not imply a change of the burden of proof to your detriment.

It is not permissible to transfer or copy these Application Examples or excerpts of them without first having prior authorization from Siemens Industry Sector in writing.

For questions about this document please use the following e-mail address:

online-support.automation@siemens.com
Preface

Objective of the application
This document describes how STEP 7 and WinCC are used to solve an automation task.

The sensors (binary and analog values) and the actuators (valves, motors) of a plant are the basic components for each automation project. Technology blocks required for the configuration of basic automation tasks are not included as standard features in the scope of delivery of STEP 7 and WinCC. Basic automation tasks include the following functions:

- Processing of the information received from sensors and actuators
- Control of the actuators

Configuration of the basic automation functions is the prerequisite for the configuration of higher-level automation tasks, e.g. control functions or sequencers.

Configuration of the application
This application includes:

- an example project that demonstrates the use of STEP 7 blocks and faceplates
- the individual configuration examples of the blocks for usage in your own projects.

The table below shows all blocks included in this application:

<table>
<thead>
<tr>
<th>Block</th>
<th>No</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BST_DIGITAL</td>
<td>FB650</td>
<td>Visualization and operator control of digital signals</td>
</tr>
<tr>
<td>BST_ANALOG</td>
<td>FB640</td>
<td>Visualization and operator control of analog signals</td>
</tr>
<tr>
<td>BST_COUNT</td>
<td>FB654</td>
<td>Numerical rectangle integrator, e.g. simulation of a fill level (without faceplate)</td>
</tr>
<tr>
<td>BST_FF</td>
<td>FB653</td>
<td>Visualization and operator control of a FlipFlop</td>
</tr>
<tr>
<td>BST_ILOCK</td>
<td>FB651</td>
<td>Visualization and operator control of a logic AND/OR gate with 8 inputs</td>
</tr>
<tr>
<td>BST_VALVE</td>
<td>FB630</td>
<td>Visualization and operator control of a binary valve</td>
</tr>
<tr>
<td>BST_MOTOR</td>
<td>FB620</td>
<td>Visualization and operator control of a motor with fixed speed and rotational direction</td>
</tr>
</tbody>
</table>
### Main contents

This application describes the creation and configuration of basic automation functions. It provides detailed information on the configuration of technology blocks, for example, blocks for the processing of binary and analog values and for the control of valves and motors. The structure of these technology blocks enables plant operation in different modes (local, manual, automatic and simulation mode).

Furthermore, it describes how these technology blocks are called and interlinked in STEP 7 and WinCC. To do so, an example plant has been configured which includes several binary and analog values, valves and motors. This example project also shows how the individual technology blocks are interconnected (basic automation). Higher-level automation functions (two-step and PID control) are also configured.

With the help of the example plant, this application can be used to describe how automatic functions can be tested at minimum time and effort and without a real connection to the process. In this way, automatic functions can be tested already during the phase of configuration (e.g. in the office). For this purpose, the technology blocks are provided with a “Simulation ON” function. In this mode, the feedback signals from sensors and actuators are simulated directly by the technology block. It simulates, for example, the feedback “Valve OPEN” after output of the control command “Valve OPEN”. Furthermore, individual blocks for the simulation of different control functions are available, e.g. to simulate the temperature or filling level of a container.

WinCC V7.0 or a later version also includes the object type “Faceplate type”. This application describes how the object type “Faceplate type” can be used for the dynamization of block icons in process pictures.

### Reference to the Industry Automation and Drive Technologies Service & Support

This document is an article from the Internet Application Portal of Industry Automation and Drive Technologies Service & Support. The following link takes you directly to the download page of this document.

# Table of Contents

Warranty, Liability and Support.................................................................................. 2
Preface....................................................................................................................... 3
Application Description .......................................................................................... 7
1 Automation Task .................................................................................................. 7
   1.1 Prerequisites ................................................................................................ 7
   1.2 Requirements for the control program .................................................... 8
   1.3 Requirements to the visualization ......................................................... 8
2 Automation Solution ........................................................................................... 12
   2.1 WinCC in the SIMATIC Manager integrated with CFC ......................... 12
   2.2 Alternative solutions ............................................................................ 15
   2.2.1 WinCC integrated in the SIMATIC Manager without CFC ............... 15
   2.2.2 WinCC and SIMATIC Manager separate configuration ..................... 15
   2.2.3 Using the “Basic Process Control” option ......................................... 15
   2.2.4 Using PCS 7 ..................................................................................... 16
   2.3 Hardware requirements ......................................................................... 17
   2.4 Used software components ................................................................. 17
3 Integration of WinCC into the SIMATIC Manager ........................................... 20
   3.1 Installing WinCC .................................................................................. 20
   3.2 Creating message classes and message types in WinCC ...................... 20
   3.3 Creating the user text blocks in WinCC ............................................... 20
4 Creating the Blocks .......................................................................................... 21
   4.1 Interfaces .............................................................................................. 21
   4.2 High-byte, low-byte change ................................................................. 21
   4.3 Standard functions ............................................................................... 22
   4.4 Alarm messages .................................................................................... 23
   4.4.1 Chronological message procedure (S7-400) .................................... 23
   4.4.2 Bit message procedure (S7-300) ...................................................... 24
5 Configuration of the Screen Management Functions .................................. 25
   5.1 WinCC pictures for screen management ............................................. 25
   5.1.1 Start screen .................................................................................... 26
   5.1.2 Desktop .......................................................................................... 27
   5.1.3 Header ........................................................................................... 28
   5.1.4 Workspace ...................................................................................... 28
   5.1.5 Footer ............................................................................................. 29
   5.2 Picture change during runtime ............................................................. 30
6 General Configuration of Faceplates .............................................................. 31
   6.1 Loading and visualizing a faceplate ...................................................... 32
   6.2 General layout of the WinCC faceplates ............................................. 33
   6.3 Configuring the faceplates ................................................................. 34
   6.3.1 The pictures of a faceplate ............................................................. 34
   6.3.2 Configurating block views ............................................................ 35
   6.3.3 Configuring operator messages ..................................................... 35
   6.4 Configuring the block icons ............................................................... 36
   6.4.1 Dynamization with WinCC status displays .................................... 36
   6.4.2 Centrally changeable block icon by “faceplate types” ................... 37
   6.4.3 Configuring the process tag display .............................................. 40
7 Configuring Technology Subfunctions ......................................................... 41
# Table of Contents

7.1 Overview of data exchange between controller and WinCC.............. 41  
7.2 Display of process states in the WinCC process display............... 42  
7.3 Logging of messages in the WinCC Alarm Logging....................... 43  
7.4 Trigger switching commands in the WinCC process display............. 45  
7.5 Processing of switching commands in the control system.............. 47  

8 Description of the Example Project.................................................. 48  
8.1 S7 Program ................................................................................ 48  
8.1.1 Unit 100 .............................................................................. 49  
8.1.2 Unit 200 .............................................................................. 50  
8.1.3 Unit 300 .............................................................................. 51  
8.2 Process pictures ........................................................................ 51  
8.2.1 Plant process picture .............................................................. 52  
8.2.2 Alarm picture ........................................................................ 53  
8.2.3 Overview of the Blocks ......................................................... 54  
8.2.4 PC System Information .......................................................... 55  
8.3 Scenarios of the example system ................................................ 56  
8.3.1 Filling and emptying the containers ........................................ 56  
8.3.2 Temperature control Unit 100 .............................................. 56  
8.3.3 Transferring the medium ...................................................... 57  
8.3.4 Controlling actuator groups ................................................... 58  

Appendix ..................................................................................................... 60  
9 Links & Literature ................................................................................. 60  
9.1 Bibliography .............................................................................. 60  
9.2 Internet links .............................................................................. 60  
9.3 Further Example Blocks ............................................................... 61  

10 History ............................................................................................... 62
Application Description

1 Automation Task

1.1 Prerequisites

For automating a process engineering plant with STEP 7 and WinCC it makes sense to describe the individual automation functions in detail prior to project start. The following information for a process engineering plant should always exist before the project start:

- **Process flowsheet**
  
  In one or several process flowsheets the plant is represented with the individual process tags. The individual aggregates and the respective pipes are represented. In the process flowsheet the individual process tags are included with the tag names. The process flowsheets normally form the basis for generating the WinCC process pictures.

- **Process tag list**
  
  The process tag list contains all process tags with name and respective detailed information.

  **Example:**
  
  The process tag name is “TIC120”, for example. The name contains the following information:
  
  - “T” ➔ The recorded physical variable is a temperature.
  - “I” ➔ The temperature is recorded as analog value.
  - “C” ➔ The temperature is used for regulation.
  - “120” ➔ The process tag has the unique number “120”.

  The detailed information contains, for example, the following information:
  
  - Process tag comment (short description e.g. temperature of Tank 120)
  - Manufacturer of the field device
  - Type of measuring transmitter (e.g. 4..20mA)
  - Value range (value range top and bottom)
  - Unit of measurement
• **Interlock list**
  The interlock list contains the information of those process tags which must be interlocked under certain conditions. If a process tag is locked the respective device of the process tag takes on the safety position.

  **Example**
  The supply valve of a container must be closed if the filling level monitoring of the container sends the “Container full” signal or if the process tag for filling level monitoring has failed.

• **Signal flow charts, function descriptions**
  Signal flow charts or function descriptions often describe production processes. This information often provides the basis for creating automation functions or sequence charts.

### 1.2 Requirements for the control program

Here some requirements for the program of the controller:

**Simple program structure**
The control program should have a simple structure. An independent plan, which contains the program code for a process tag, shall be generated for each process tag. The name of the chart shall contain the actual process tag name. This enables the programmer or the later maintenance staff to find the program for a process tag.

**Reusable program code**
Recurring program code shall be programmed as a complete block. The completed block shall be called up where the program code is required. This ensures that always recurring tasks are solved in the same way. This saves time and avoids errors.

**Using operated and monitored blocks**
The control program shall be designed so that important program functions can be operated and monitored during WinCC runtime.

  **Example**
  During runtime the lock conditions of a valve or motor shall be displayed. This informs the operator of the reasons why the valve cannot be opened or why the motor cannot be switched on. In the commissioning phase it is very useful if the interlock conditions can be cancelled or set.

### 1.3 Requirements to the visualization

Normally, process technological engineering processes can be visualized in one or several process pictures. The process pictures usually correspond to the process flow charts.

The figure below shows a WinCC process picture in process automation.
Using this process picture some requirements regarding the visualization shall be explained.

**Centrally changeable block icons**

The dynamic parts of a process picture shall be executed as block icons. The block icons are generated separately, tested and then installed into the process pictures. It shall be possible to change the block icons centrally.

Centrally changeable means:
If a block icon shall be changed later on, it is not important to rework all process pictures in which the block icon has already been used. The modification shall occur once at a central location.

**Time-saving block icons**

Since as many process tags as possible shall be displayed in process pictures, the block icons must be configured with little space. Therefore, the block icons are represented as follows:

- No frame used for the process tag
- The process tag is not permanently displayed in the picture. The process tag name can be displayed and hidden again on demand during runtime. The process tag name additionally appears as tooltip text.
- The classic WinCC group display shall not be used in the user project. For the classic group display the icons for the different message types are displayed next to each other. It is therefore possible that valuable space is wasted. The display of a group warning, for example, is not important if simultaneously a group alarm or process tag error occurs. The individual information of a group display shall, if possible, be superimposed according to priority. This may considerable reduce the size of a block icon. For example, the display of a warning can be superimposed by the display of an alarm or process tag failure.
Multi-stage operating concept
Operator intervention into the process shall not be possible directly in the process picture but only by opening an additional faceplate. In the faceplate certain control elements (buttons, input fields) can be enabled for operation according to user authorization of the currently logged-in user. Additionally, a check of the operator authorization may occur directly when executing the action (e.g. in the script).

Uniform representation
Equal functions (e.g. simulation ON/OFF, manual/automatic, ...) in different blocks shall be displayed in the same way.

Generating WinCC configuration data
It shall be possible to generate the main configuration data of the WinCC project from the control program. In case of an “integrated” WinCC project the WinCC tags, alarms/messages, texts and archive tags are generated automatically by the compilation process in the WinCC project. This prevents configuration errors and saves configuration time.

Simple and time-saving configuration
Configuration parts, which are not automatically generated, shall be executed as simple as possible, to prevent errors and save time.

Reducing external WinCC tags
The number of external WinCC tags shall be kept as low as possible to minimize licensing costs.
Example:
The bits of a status word shall not be transferred individually to WinCC as a BOOL tag, but as a BYTE, WORD or DWORD tag.

Short picture opening times
The process pictures shall be configured so that the picture opening times are as low as possible. A picture opening time refers to the time from operating a button to change the picture, until the picture is downloaded and any dynamization has been updated. In practice, picture opening times of less than 2 seconds are required.

Supporting several screens in Runtime (Multi-VGA)
If the WinCC station has several screens (Multi-VGA), these shall be supported by Runtime.

Supporting WinCC multi-clients
The WinCC configuration shall be made so the WinCC clients (multi-clients) are supported.

Supporting WinCC web-clients
The WinCC configuration shall be made so the WinCC clients are supported.
Display of invalid process states

Invalid process states shall be represented clearly visible. Invalid process states may occur, for example, if:

- the connection to the controller has been interrupted.
- the address of an external WinCC tag has been configured wrong.
- the dynamization in the picture has errors.
- a picture is selected and the dynamization has not yet been updated.
2 Automation Solution

2.1 WinCC in the SIMATIC Manager integrated with CFC

In this application WinCC is used as integrated project. The WinCC project has been integrated in the SIMATIC Manager (STEP 7). The process tags of an automation project are configured in the CFC of STEP 7. Each process tag is generated in a separate CFC. In the CFC the message texts as well as texts for process tag names, process tag comments and units are configured. The information is then transferred to WinCC by means of a compilation process. Prior to the start of the configuration the existing process tags are grouped as according to different process tag types. For example, there are binary values, analog values, valves, pumps. A separate block (block type) is created for each process tag type. For most block types of this application an additional block icon and faceplate is created for operator control and monitoring.

In the course of this application the following technology blocks are created:

Table 2-1

<table>
<thead>
<tr>
<th>Block</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BST_DIGITAL</td>
<td>Binary value display</td>
</tr>
<tr>
<td></td>
<td>Display of a binary signal with the option of time delay, negation of the input signal and simulation.</td>
</tr>
<tr>
<td>BST_ANALOG</td>
<td>Analog value display</td>
</tr>
<tr>
<td></td>
<td>Preparation and display of an analog signal, limit value monitoring and simulation.</td>
</tr>
<tr>
<td></td>
<td>The block can be interconnected with the raw data of S7 analog modules or with S7 real values.</td>
</tr>
<tr>
<td></td>
<td>The analog signal can be monitored for exceeding or falling short of 4 limit values and a respective message be triggered.</td>
</tr>
<tr>
<td>BST_MOTOR</td>
<td>Operating and monitoring the motor</td>
</tr>
<tr>
<td></td>
<td>Monitoring, controlling and simulating a motor with fixed speed and rotational direction. The motor can be monitored for status feedback and motor protection. Monitoring a dry-running protection is planned. Feedback monitoring can be switched off.</td>
</tr>
<tr>
<td>BST_VALVE</td>
<td>Controlling and monitoring the valve</td>
</tr>
<tr>
<td></td>
<td>Monitoring, controlling and simulating a valve with two states (open/closed). The motor can be monitored for status feedback. Feedback monitoring can be switched off.</td>
</tr>
<tr>
<td>BST_SIMOREV</td>
<td>SIMOCODE reversing starter</td>
</tr>
<tr>
<td></td>
<td>Monitoring, controlling and simulating the motor management system SIMOCODE Pro as reversing starter.</td>
</tr>
<tr>
<td>BST_SIMODIR</td>
<td>SIMOCODE direct starter</td>
</tr>
<tr>
<td></td>
<td>Monitoring, controlling and simulating the motor management system SIMOCODE Pro as direct starter.</td>
</tr>
<tr>
<td>Block</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>BST_ILOCK</td>
<td>Visualizable AND/OR gate&lt;br&gt;Monitoring 8 binary input signals for AND/OR logic. Input and output can be simulated separately and negated. This block for example is called by other faceplates to display interlock conditions during runtime.</td>
</tr>
<tr>
<td>BST_FF</td>
<td>Visualizable RS-Flip-Flop&lt;br&gt;Controlling a binary signal with storage behavior.</td>
</tr>
<tr>
<td>BST_COUNT</td>
<td>Counter/integrator&lt;br&gt;This block can be used as simple counter as well as integrator. It has several counter and control inputs. It can be used, for instance, to simulate the filling level of a container based on different valve states (influx/drain).</td>
</tr>
<tr>
<td>BST_PIDCTRL</td>
<td>Visualizable software PID controller&lt;br&gt;Software PID controller for controlling analog actuators. The controller can monitor raw data of S7 analog modules or S7 real values. Actuators can be addressed directly via input of a manual setting. For the calculation the P, I and D part can be switched on and off separately.</td>
</tr>
<tr>
<td>BST_LAG</td>
<td>PT1 element (delay 1&lt;sup&gt;st&lt;/sup&gt; order)&lt;br&gt;This block can be used for simulation of self-regulating processes (e.g. temperature in a container). In addition to the constants of the PT1 element it has further inputs for convenient simulation of a controlled variable based on valves.</td>
</tr>
<tr>
<td>BST_SPLITR</td>
<td>SPLITRANGE&lt;br&gt;This block splits the output signal of a PID controller (0%..100%) into two analog or binary output signals. This enables, for example, using the output signal of a PID controller to control a controller with two actuators. (e.g. temperature control with heating and cooling circuit)</td>
</tr>
<tr>
<td>BST_MM4</td>
<td>MICROMASTER 440&lt;br&gt;Monitoring, controlling and simulating the frequency converter MICROMASTER 440.</td>
</tr>
<tr>
<td>BST_SINAG120</td>
<td>SINAMICS G120&lt;br&gt;Monitoring, controlling and simulating the frequency converter SINAMICS G120.</td>
</tr>
</tbody>
</table>

The technology blocks described above are provided with the respective sources.

**ATTENTION**<br>Before using the blocks in your own projects, check their proper functioning and adjust them to your individual requirements where necessary. The blocks described in this application are only intended as a template for the creating your own blocks.
The example project of this application shows the use of technology blocks by means of an example plant.

The example plant consists of several binary and analog values, valves and motors. For each process tag an independent CFC was generated in STEP 7 in which the respective block type is called and interconnected. Higher-level automation functions (two-step and PID control) are also configured.

The following figure gives an example of the work screen on a configuration system for STEP 7 and WinCC.

Figure 2-1

This screen contains the following components of the configuration:

- **SIMATIC Manager**
  In the workspace of the SIMATIC Manager the configured sources of the blocks are displayed.

- **Editor CFC**
  The right side of the screen contains an opened CFC. It displays the interconnection of two valve blocks (BST_VALVE) in test mode. The signals can be monitored and controlled online.

- **WinCC-Runtime**
  The WinCC runtime is visible in the background. Some block icons and pipes are displayed in the process picture. The bottom left corner contains an opened faceplate of the block type “BST_MOTOR”.

2.2 Alternative solutions

2.2.1 WinCC integrated in the SIMATIC Manager without CFC

Using the STEP 7 option CFC (Continuous Function Chart) is not necessary. The STEP 7 configuration may occur in the conventional way in STL, LAD or FBD. If the WinCC project has been integrated in the STEP 7 project, the following configuration data from the STEP 7 project can be transferred to the WinCC project:

- Tags (Tag Management)
- Alarms, messages (Alarm Logging)
- Curve configuration (Tag Logging)

Using SCL is not necessarily required. However, there are restrictions compared with the application of CFC:

- Some block attributes are only available when using the CFC (S7_comment, S7_unit). For example, the texts for process tags, process tag comments, units cannot be transferred to WinCC without greater workload. The FAQ with entry ID 27147567 describes how alternatively the S7 block attribute “S7_enum” can be used to transfer texts of enumerations from STEP 7 to WinCC.
- The “Charts” folder is not available without the STEP 7 option CFC. The process tags are programmed directly in the STEP 7 blocks in LAD, FBD or STL. CFC has the advantage that each process tag can be configured in its own chart. This is not possible without CFC.

2.2.2 WinCC and SIMATIC Manager separate configuration

The procedure for configuring the technology blocks described in this application can also be applied to projects where WinCC has not been integrated in the SIMATIC Manager. The main differences are:

- Configuration data (tags, messages, texts) for WinCC are not automatically transferred to WinCC by a compilation process.
- The chronological message procedure (e.g. ALARM_8P) cannot be used. The bit message procedure must be used.

2.2.3 Using the “Basic Process Control” option

The WinCC option “Basic Process Control” provides functions for picture management and for calling up faceplates. The “Basic Process Control” option is available free of charge as of WinCC V6.0, it can be selected during the installation of WinCC.

The layout of the runtime, for example, (number and resolution of the screens) can be configured. The process pictures can be filed in order in the picture tree. The configuration occurs as in PCS 7.

The WinCC option “Basic Process Control” (process control options) contains no technology blocks (e.g. motor, valve). These must be generated by the user. The procedure in this document can be used.
Note

The WinCC option “Basic Process Control” can also be used if WinCC has not been integrated in the SIMATIC Manager. The overview area of Runtime then contains WinCC group displays which are not supplied with valid values. A possible remedy is omitting the “Basic Process Control” or proceeding according to the FAQ with entry ID 17778440.

2.2.4 Using PCS 7

When using PCS 7 the user has additional functions:

PCS 7 standard library

The PCS 7 standard library already contains technology blocks. For example, there are blocks for display of a binary or analog value or for operating and monitoring a valve or motor.

Plant hierarchy

Without PCS 7 only the “Component view” is available to the user in the SIMATIC Manager. All charts for configuring the process tags are filed in the “Charts” folder.

When using PCS 7 there are additional views in the SIMATIC Manager. In the “Plant view”, for example, it is possible to create hierarchy folders. This enables structuring the plant, e.g. “Plant > Unit > Function”. The configured Plant Hierarchy is displayed by the picture tree (Picture Tree Manager) during runtime.

Graphic Object Update Wizard

Block icons can be automatically integrated into the process pictures or updated. During the configuration of the process tags in CFC it is already defined which block icons shall be used in the process tags.

Faceplate Designer

The faceplate designer is only available for PCS 7. It is an aid for configuring faceplates.

Additional functions in the Graphic Designer

- Advanced status display
  The advanced status display can be used alternatively to the WinCC group display for displaying alarms and messages in the process picture.
- Advanced analog display
2.3 Hardware requirements

Table 2-2

<table>
<thead>
<tr>
<th>Component</th>
<th>Qty</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development system</td>
<td>1</td>
<td>PC for the configuration of the control functions and WinCC. The usual hardware requirements for STEP 7 and WinCC apply.</td>
</tr>
<tr>
<td>S7-400 CPU</td>
<td>1</td>
<td>The technology blocks of this application use the chronological message procedure (ALARM_8P). Therefore an S7-400 CPU is required. As an alternative, you can also simulate system control with PLCSIM.</td>
</tr>
</tbody>
</table>

2.4 Used software components

Table 2-3

<table>
<thead>
<tr>
<th>Component</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>STEP 7 V5.4 SP3 Professional</td>
<td>S7-PLCSIM and S7-SCL are included in the scope of delivery.</td>
</tr>
<tr>
<td></td>
<td>• S7-PLCSIM can be used for simulation.</td>
</tr>
<tr>
<td></td>
<td>• S7-SCL is used for the creation of control blocks.</td>
</tr>
<tr>
<td>CFC V7.0 SP1</td>
<td>CFC is used for the easy interconnection of blocks. Additionally, CFC offers the transfer of texts (process tag comment, units of notation, interlock information, …) to WinCC.</td>
</tr>
<tr>
<td>WinCC V7.0 SP1</td>
<td></td>
</tr>
<tr>
<td>WinCC WebNavigator V7.0 SP1</td>
<td></td>
</tr>
<tr>
<td>SIMATIC PDM V6.0 SP3</td>
<td>Used to configure the special field devices, SIMOCODE pro, MICROMASTER 440 and SINAMICS G120.</td>
</tr>
<tr>
<td>SIMOCODE ES 2007 SP1</td>
<td>Used to configure a special field device SIMOCODE pro.</td>
</tr>
</tbody>
</table>
Sample project

The figure below shows the setup of the example project: Only an S7-400 and a SIMATIC PC station is used for configuring and starting WinCC Runtime. The S7-400 can alternatively be simulated with PLCSIM.

Figure 2-2

The following list contains all files and projects used in this example.

Table 2-4

<table>
<thead>
<tr>
<th>Component</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>BST_EXAMPLE.zip</td>
<td>This zip-archive contains the example project. The file was generated with the “Archive…” function of the SIMATIC Manager and can be extracted again with the “Retrieve…” function.</td>
</tr>
<tr>
<td>BST_EXAMPLE_DOC_d.pdf</td>
<td>This document.</td>
</tr>
<tr>
<td>BST_FILES.zip</td>
<td>This ZIP-archive contains the block library for STEP 7 as well as all necessary WinCC pictures and scripts to be used in own projects.</td>
</tr>
<tr>
<td>BST_DOCU_d.zip</td>
<td>This ZIP-archive contains a description in pdf-format for all blocks. In this document the interface, operation as well as configuration in own projects is explained.</td>
</tr>
</tbody>
</table>
To use the example project proceed as follows:

- Unzip the archived project “BST_EXAMPLE.zip” with the SIMATIC Manager (File > Retrieve…).
- Open the dearchived project.

The following screenshot shows the opened STEP 7 project “BST_Example” in the SIMATIC Manager.

Figure 2-3

The project contains the following stations:

- **SIMATIC 400 Station “PLC400”**
  This station contains the program of the controller including the source code for S7-400 CPUs.

- **SIMATIC PC Station “HMI_SRV”**
  This station contains the WinCC server project “BST”. The WinCC configuration is explained using this project.

- **S7 program “BST_PRG_300”**
  This program contains the source code and the compiled blocks for S7-300 CPUs. These blocks are not further used in the example program.

**Notes**

The “Source” folder of the S7 programs contains the sources of the technology blocks.

The “Blocks” folder contains the completely compiled and preconfigured blocks (with alarms, messages and units).

The “Charts” folder contains the CFCs. The CFCs represent the configuration of an example plant. It is shown how the technology blocks are interconnected to configure a plant.
3 Integration of WinCC into the SIMATIC Manager

This application requires the integration of WinCC into the SIMATIC Manager.

3.1 Installing WinCC

To be able to use WinCC as integrated project in the SIMATIC Manager you have to install the WinCC components AS-OS Engineering and Object Manager.

**Note**

General notes on the integration of WinCC into the SIMATIC Manager are available:

- in the FAQ with entry ID 11841504
- in the FAQ with entry ID 22272911
- in the WinCC Information System at “Working with WinCC > Integration of WinCC into the SIMATIC Manager >”

3.2 Creating message classes and message types in WinCC

This application does not use the OS project editor. Therefore, proceed as described in entry ID 31622970 to create the message classes.

**Note**

To determine the correlation between message classes in STEP 7 and WinCC the independent PC station “GETMSGCLAS” was created the SIMATIC Manager.

3.3 Creating the user text blocks in WinCC

This application does not use the OS project editor. Therefore, proceed as described in entry ID 30550239 to create the message texts.
Creating the Blocks

This chapter describes how the blocks were created using SCL. When creating the blocks, it was ensured that equal functions (manual/automatic, simulation) were also programmed equally. For this reason, a standard source (“BST_DUMMY_400”) was generated. This source already contains tags and functions which on demand can be used for all blocks.

4.1 Interfaces

The following interfaces have been standardized for all blocks and can be used on demand:

<table>
<thead>
<tr>
<th>Name</th>
<th>In</th>
<th>I/O</th>
<th>Out</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOCK</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>Interlock</td>
</tr>
<tr>
<td>ERR_EXTERN</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>External error</td>
</tr>
<tr>
<td>LIOP_SEL</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>Connection or operation selected</td>
</tr>
<tr>
<td>L_AUT</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>(linked) manual/automatic mode</td>
</tr>
<tr>
<td>L_REMOTE</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>(linked) local/remote mode</td>
</tr>
<tr>
<td>L_SIM</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>(linked) process/simulation</td>
</tr>
<tr>
<td>L_RESET</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>(linked) reset error</td>
</tr>
<tr>
<td>SAMPLE_T</td>
<td>x</td>
<td></td>
<td></td>
<td>REAL</td>
<td>Sampling time</td>
</tr>
<tr>
<td>MSG1_EVID</td>
<td>x</td>
<td></td>
<td></td>
<td>DWORD</td>
<td>Alarm 8P ID (only S7-400)</td>
</tr>
<tr>
<td>MSG2_EVID</td>
<td>x</td>
<td></td>
<td></td>
<td>DWORD</td>
<td>Notify 8P ID (only S7-400)</td>
</tr>
<tr>
<td>OP_dwCmd</td>
<td>x</td>
<td></td>
<td></td>
<td>DWORD</td>
<td>Command tag WinCC</td>
</tr>
<tr>
<td>QdwState</td>
<td>x</td>
<td></td>
<td></td>
<td>DWORD</td>
<td>Status tag WinCC</td>
</tr>
<tr>
<td>QMAN_AUT</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>Status manual/automatic</td>
</tr>
<tr>
<td>QREMOTE</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>Status local/remote</td>
</tr>
<tr>
<td>QSIM</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>Status process/simulation</td>
</tr>
<tr>
<td>QLOCK</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>Interlock error</td>
</tr>
<tr>
<td>QERR</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>General error</td>
</tr>
<tr>
<td>QERR_EXT</td>
<td>x</td>
<td></td>
<td></td>
<td>BOOL</td>
<td>External error</td>
</tr>
<tr>
<td>QwAlarm</td>
<td>x</td>
<td></td>
<td></td>
<td>WORD</td>
<td>Alarm tag (only S7_300)</td>
</tr>
</tbody>
</table>

4.2 High-byte, low-byte change

WinCC treats the bytes of a 32-Bit tag (DWORD) in opposite sequence. To ensure the function in a simple way, the bytes were exchanged internally.

One internal tag each was created in PLC and HMI format:

Figure 4-1

```
OP_dwCmdHMI : DWORD := 16H0;
OPabyCmdHMI AT OP_dwCmdHMI : ARRAY [0..3] OF BYTE;
OP_dwCmdPLC : DWORD := 16H0;
OPabyCmdPLC AT OP_dwCmdPLC : ARRAY [0..3] OF BYTE;
```
The bytes are changed via the following instructions:

```plaintext
// Change lowbyte to highbyte for HMI command word
OPcCmdHMI := OP_dwCmd;
OPabyCmdPLC[0] := OPabyCmdHMI[3];
OPabyCmdPLC[3] := OPabyCmdHMI[0];
```

Thereafter the individual bits can be used in the usual order of Bit[0] to Bit[31].

4.3 Standard functions

The following functions have been standardized for all blocks and can be used on demand: The used inputs as well as the bits of the HMI command tag are identical for all blocks:

<table>
<thead>
<tr>
<th>Manual / automatic operation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output “QMAN_AUT” is set or reset by setting the “L_AUT” input or by operating the operator “OP_dwCmd[Bit16]/[Bit17]”. For further processing in the source code the value of “QMAN_AUT” is used.</td>
<td></td>
</tr>
</tbody>
</table>

```plaintext
// MANUAL / AUTOMATIC Operation -----------------------------------------------
IF (LIOP_SEL AND NOT L_AUT) OR (OPabyCmdPLC[16] AND NOT LIOP_SEL) THEN
QMAN_AUT := FALSE;
ELSIF (LIOP_SEL AND L_AUT) OR (OPabyCmdPLC[17] AND NOT LIOP_SEL) THEN
QMAN_AUT := TRUE;
END_IF;
```

<table>
<thead>
<tr>
<th>Local/ remote operation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output “QREMOTE” is set or reset by setting the “L_REMOTE” input or by operating the operator “OP_dwCmd[Bit18]/[Bit19]”. For further processing in the source code the value of “QREMOTE” is used.</td>
<td></td>
</tr>
</tbody>
</table>

```plaintext
// LOCAL / REMOTE Operation -----------------------------------------------
IF (LIOP_SEL AND NOT L_REMOTE) OR (OPabyCmdPLC[18] AND NOT LIOP_SEL) THEN
QREMOTE := FALSE;
ELSIF (LIOP_SEL AND L_REMOTE) OR (OPabyCmdPLC[19] AND NOT LIOP_SEL) THEN
QREMOTE := TRUE;
END_IF;
```

<table>
<thead>
<tr>
<th>Process / simulation mode</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Output “QSIM” is set or reset by setting the “L_SIM” input or by operating the operator “OP_dwCmd[Bit20]/[Bit21]”. For further processing in the source code the value of “QSIM” is used.</td>
<td></td>
</tr>
</tbody>
</table>

```plaintext
// SIMULATION ON / OFF -----------------------------------------------
IF (LIOP_SEL AND NOT L_SIM) OR (OPabyCmdPLC[20] AND NOT LIOP_SEL) THEN
QSIM := FALSE;
ELSIF (LIOP_SEL AND L_SIM) OR (OPabyCmdPLC[21] AND NOT LIOP_SEL) THEN
QSIM := TRUE;
END_IF;
```
4 Creating the Blocks

4.4 Alarm messages

The blocks for S7-400 CPUs use the “chronological message procedure” and the blocks for S7-300 CPUs “bit message procedure”.

4.4.1 Chronological message procedure (S7-400)

Blocks “Alarm8P” and “Notify8P” are called for the chronological message procedure. Signals 5 to 8 of “Alarm8P” are reserved for standard error messages of “Notify8P” for standard status messages. Signals 1 to 4 respectively are available.

Figure 4-3

```plaintext
// Alarm8P ---------------------------------------------------------------
ACP(  
    EN_R := 1,
    SIG_1 := 0,
    SIG_2 := 0,
    SIG_3 := 0,
    SIG_4 := 0,
    SIG_5 := CLOCK,  // Interlock Error
    SIG_6 := 0,
    SIG_7 := ERR_EXT,  // External Error
    SIG_8 := ERR,  // General Error
    ID := w16#0000,
    EV_ID := MSG1_EVID,
    SEVERITY := w16#40
);

MSG1_bDone := ACP_DONE;
MSG1_bError := ACP_ERROR;
MSG1_uStatus := ACP_STATUS;
MSG1_uAck := ACP_ACK_STATUS;
```
4 Creating the Blocks

Figure 4-4

```c
// Notify_SP
N8P(
    SIG_1 := 0,
    SIG_2 := 0,
    SIG_3 := 0,
    SIG_4 := 0,
    SIG_5 := LOCK,   // Interlock
    SIG_6 := QREMOTE, // Remote
    SIG_7 := QMAN_AUT; // Automatic
    SIG_8 := QSIM;    // Simulation
    ID := "W16#0000",
    EV_ID := MSG2_EVId,
    SEVERITY := w16#40
);
MSG2_bDone := N8P_DONE;
MSG2_bError := N8P_ERROR;
MSG2_wState := N8P_STATUS;
```

4.4.2 Bit message procedure (S7-300)

For the bit message procedure the signals are transmitted to the HMI alarm tag “QwAlarm”. The messages must be created in the WinCC Alarm Logging and be configured with the respective bits of “QwAlarm”.

Bits 4-7 of “QwAlarm” are reserved for standard error messages and bits 12-15 of “QwAlarm” are reserved for standard status messages. Bits 0-3 (error) and bits 8-11 (status) are available.

Figure 4-5

```c
// Bit alarm procedure
CabAlarmPLC[0] := 0;
CabAlarmPLC[1] := 0;
CabAlarmPLC[2] := 0;
CabAlarmPLC[3] := 0;
CabAlarmPLC[4] := QLOCK;   // Interlock Error
CabAlarmPLC[5] := 0;
CabAlarmPLC[7] := 0;
CabAlarmPLC[8] := 0;
CabAlarmPLC[9] := 0;
CabAlarmPLC[10] := 0;
CabAlarmPLC[11] := 0;
CabAlarmPLC[12] := QLOCK;  // Interlock
CabAlarmPLC[14] := QMAN_AUT; // Automatic
CabAlarmPLC[15] := QSIM;   // Simulation

QabyAlarm[0] := QabyAlarmPLC[1];
QabyAlarm[1] := QabyAlarmPLC[0];
```
5 Configuration of the Screen Management Functions

This section describes how to commission WinCC Runtime so it is divided into Overview area, Workspace and Keyset. The functions for screen navigation (screen change) are provided. It is also described how systems with several monitors are used. The functions for calling faceplates are described in the next section.

**Note**

This section is only significant if you are not using the OS project editor. When using the OS project editor, screen management functions (change process picture, open faceplate) are available as a standard.

5.1 WinCC pictures for screen management

The WinCC Runtime of a WinCC station is realized by the “Graphics Runtime“ application (“pdlrt.exe”). For realizing the screen management the following WinCC pictures are configured in the Graphics Designer.

- @BST_Main_1M.pdl (Start picture 1 monitor operation)
- @BST_Main_1M.pdl (Start picture 2 monitor operation)
- @BST_DESK.pdl
- @BST_HEAD.pdl
- @BST_WORK.pdl
- @BST_BOTTOM.pdl

**Note**

The “Alarm.pdl” does not serve the actual picture management. It can be loaded into the workspace to display messages.
5.1.1 Start screen

In the WinCC ControlCenter you define the WinCC picture, downloaded during the start of Graphics Runtime, at "Computer properties > Graphics Runtime". Select the picture "@BST_Main_1M.pdl" for the one monitor operation or the picture "@BST_Main_2M.pdl" for the two monitor operation as start picture.

Figure 5-1

When using several monitors, the Graphics Runtime is not executed separately for each monitor, but one Graphics Runtime supplies all monitors. This means that the picture size of the start screen must fill the entire display area of all monitors.

Example:

If two monitors are used next to each other and if each monitor has a resolution of 1280x1024 picture points, then the start screen should have a resolution of 2560x1024. In this case, the following figure shows the configuration of the start screen "@BST_Main_2M.pdl" in the Graphics Designer.

Figure 5-2

Note

The WinCC Web client represents an exception. For a WinCC web client the WinCC Runtime runs completely within the Internet Explorer, whereby the Internet Explorer can be executed several times (also with WinCC).
5.1.2 Desktop

The start screen “@BST_Main_1M.pdl” shows the picture window “@BST_DESK.pdl”. Start screen “@BST_Main_2M.pdl” is divided so that for each monitor a separate picture window “@BST_DESK.pdl” is displayed. Picture “BST_DESK.pdl” in return is divided into several picture windows to represent the

- Overview area “@BST_HEAD.pdl”, the
- Workspace “@BST_WORK.pdl” and the
- Keyset “@BST_BOTTOM.pdl”.

Figure 5-3
5 Configuration of the Screen Management Functions

5.1.3 Header

The “@BST_HEAD.pdl” screen displays important information during runtime in the overview area. The information in the overview area are always displayed independent of the just selected process picture and are never superimposed by opened faceplates.

In this application the following information is displayed in the overview area:

- WinCC Alarm Control for display of the last three messages
- Output field for display of the currently registered user.
- Output field for display of the current computer name.
- WinCC Digital Clock for display of the current data and time
- WinCC Logo

Figure 5-4

5.1.4 Workspace

The “@BST_WORK.pdl” in return contains a picture window into which the actual process picture can be downloaded. The following figure shows the “@BST_WORK.pdl” picture. During runtime start the “Overview.pdl” process picture is displayed. In the “@BST_WORK.pdl” you define the WinCC process picture to be displayed during Runtime start.

Figure 5-5

During picture change only the picture in the workspace is changed “@BST_WORK.pdl”, the pictures in the overview area and keyset remain. This procedure offers the following advantage:

- Reduced configuration workload, since overview area and keyset are not configured in each process picture but only once (centrally).
5 Configuration of the Screen Management Functions

- Improved performance at picture change since fewer screen elements must be loaded.

Note
The "@BST_WORK.pdl" screen contains several superimposed picture windows. The superimposed picture windows display WinCC faceplates. The functions for faceplate management are described in the next section.

5.1.5 Footer

The "@BST_BOTTOM.pdl" represents the buttons for frequently required functions as keyset. The buttons in the keyset exist for the following functions:

- Calling the plant display ("Overview.pdl")
- Calling the WinCC message list ("Alarm.pdl")
- Calling the faceplate overview ("General.pdl")
- Calling the PC diagnosis ("PCDiagSysInfo.pdl")
- Show/hide process tag characteristic
- Change language (planned)
- User login/logoff (planned)
- Hardcopy
- Terminate WinCC Runtime

Figure 5-6
5.2 Picture change during runtime

The following function is called for the picture change in the workspace:

```
int BST_WorkfieldOpen(char*, char*)
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>char *lpszPictureName</td>
<td>Absolute name of the picture from which the function was called.</td>
</tr>
<tr>
<td>char *lpszPictureNameNew</td>
<td>The WinCC picture to be displayed in the workspace.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returned value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>When successful the function returns the value (int) 0, otherwise a value unequal 0 (error code)</td>
</tr>
</tbody>
</table>

The “BST_WorkfieldOpen()” is called by clicking a button in the keyset which executes a picture change.

The functions is used to:
- determine the screen from where the button was clicked,
- set the “Picture name” property of the picture window which displays the new picture in the workspace.

It is possible to configure further functions for picture management, e.g.
- Previous picture
- Next picture
- Memorize picture
- Restore memorized picture
6 General Configuration of Faceplates

This section describes the configuration of faceplates and block icons.

Using faceplates recurring display and operator functions are configured in a picture. During Runtime the dynamic screen elements of a faceplate with the process tags of a measuring location.

You can open a faceplate during runtime by clicking a block icon. A faceplate usually represents a movable, closable picture window which does not cover the entire workspace.

Separate faceplates are configured for different functions.

Opening a faceplate mainly consists of the following steps:

- Loading and visualizing a faceplate template
- Supplying the dynamic picture elements of the faceplate template with process data

**Note**

This section is only important if you are **not** using the OS project editor. When using the OS project editor, picture management functions (open a process picture, open a faceplate) are available as a standard. You can then open a faceplate as in FAQ with entry ID 24193022.
6.1 Loading and visualizing a faceplate

The “@BST_WORK.pdl” contains several picture windows.

The picture windows have the object names “TOP01” to “TOP10”. The following function is called when clicking a block icon:

```c
int BST_TopfieldOpen(char*, char*, char*)
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>char *lpszPictureName</td>
<td>Absolute name of the picture from which the function was called.</td>
</tr>
<tr>
<td>char *lpszObjectName</td>
<td>Name of the clicked object.</td>
</tr>
<tr>
<td>char *lpszTopPictureName</td>
<td>The picture window to be displayed as a faceplate. The picture name of the faceplate to be opened is defined in the header file “BST_FPDEF.h”.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Returned value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>When successful the function returns the value (int) 0, otherwise a value unequal 0 (error code)</td>
</tr>
</tbody>
</table>

The “BST_TopfieldOpen()” is used to
- determine the object names of the next free (not displayed) picture window (“TOP01” to “TOP10”),
- set the “Picture name” of the picture window with the name of the block template (WinCC picture),
- determine the tag to which the called object is linked,
General Example WinCC V7, S7 V5, CFC, SCL, S7-400

6 General Configuration of Faceplates

- set the “Tag prefix” property of the picture window with the prefix of the determined tag and
- set the “Display” property of the picture window to “TRUE”.

6.2 General layout of the WinCC faceplates

The WinCC faceplates have a uniform basic framework. Figure 6-2

- Process tag comment
  The tag prefix is displayed to which the picture window is connected.
- Button for closing
  In each corner of the faceplate buttons for closing the faceplate are displayed.
- Keyset
  At the bottom area of the block icon a keyset is displayed. The keyset contains buttons to display other views of the faceplate. The keyset is expandable individually for each picture block.
- Workspace
  The workspace is located between the comment line (process tag comment) and the keyset. The individual views of a faceplate are displayed in the workspace. When opening the faceplate the standard view is displayed.
6.3 Configuring the faceplates

A faceplate is used for detailed display and control of an object (e.g. analog value, motor, valve, ...) of the control program.

6.3.1 The pictures of a faceplate

Using the “ANALOG” faceplate the figure below illustrates which WinCC pictures compose the faceplate.

Figure 6-3

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>WinCC Pictures</th>
<th>Button</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Standard view</td>
<td>BST_&lt;Typical&gt;_STANDARD.pdl</td>
<td>![Icon]</td>
</tr>
<tr>
<td>2.</td>
<td>Message view</td>
<td>BST_&lt;Typical&gt;_MSG.pdl</td>
<td>![Icon]</td>
</tr>
<tr>
<td>3.</td>
<td>Curve view</td>
<td>BST_&lt;Typical&gt;_TREND.pdl</td>
<td>![Icon]</td>
</tr>
<tr>
<td>4.</td>
<td>Service view</td>
<td>BST_&lt;Typical&gt;_SERVICE.pdl</td>
<td>![Icon]</td>
</tr>
<tr>
<td>5.</td>
<td>Limit value view</td>
<td>BST_&lt;Typical&gt;_LIMITS.pdl</td>
<td>![Icon]</td>
</tr>
<tr>
<td>6.</td>
<td>Diagnostic view</td>
<td>BST_&lt;Typical&gt;_DIAG.pdl</td>
<td>![Icon]</td>
</tr>
<tr>
<td>7.</td>
<td>Statistic view</td>
<td>BST_&lt;Typical&gt;_STAT.pdl</td>
<td>![Icon]</td>
</tr>
</tbody>
</table>

Picture “BST_<TYPICAL>_Main.pdl” represents the basic picture of a faceplate. It contains the comment line, the keyset, the button for closing as well as the picture window which displays the actual views of the faceplate. The number of views is variable through adding or removing the buttons as well as the respective text objects.

A separate picture is configured for each view of a faceplate and an individual button is provided, e.g.:

Table 6-1
6.3.2 Configuring block views

The graphic elements of a block are configured in the individual views of a block. The daynamization types “direct tag connection” or “dynamic dialog” are used for dynamization. The following figure shows the dynamization of the display of an error bit with the Dynamic dialog.

Figure 6-4

This picture illustrates the configuration using the tag prefix. For dynamizations in the faceplate no complete tag names are given, but only the names of the individual structure elements of a block type. When opening the faceplate the tag prefix of the faceplate is set via the “BST_TopFieldOpen()” script. This accesses valid process tags during runtime.

6.3.3 Configuring operator messages

Operator logging is not realized in the example project. A possible procedure for configuration is described in the FAQ entry 24325381 “How to create user-defined operator messages in WinCC?”.
6.4 Configuring the block icons

A faceplate is used displaying the most important features of an object (e.g. analog value, motor, valve, ...) in the process picture. A mouse-click on the block icon opens the respective faceplate.

6.4.1 Dynamization with WinCC status displays

In the block icon mainly WinCC status displays are used for dynamization.

For the various characteristics (states) of a block, there are separate status displays configured. Although individual blocks differ in their main function, they often contain similar functions that are displayed in the same way:

- Display local / remote operation
  - Operating mode “local”

- Display manual / automatic operation
  - Operating mode “manual”
  - Operating mode “automatic”

- Display simulation on / off
  - Simulation “on”

- Display failure / warning
  - General failure
  - Warning
  - Alarm

- Display interlock
  - Interlock pending
  - Interlock error

For the configuration of the status displays of the block icon a separate picture “BST_<TYPICAL>_ICON_Define.pdf” has been created in the Graphics Designer for each block type in this example application.

All standard symbols valid for all blocks as well as special symbols of the status displays of a block type are represented (configured) in this picture. The export name is displayed for each symbol.
6 General Configuration of Faceplates

6.4.2 Centrally changeable block icon by “faceplate types”

This application uses a WinCC object of the “Faceplate type” to display a block icon. WinCC V7.0 or a later version also includes the object type “Faceplate type”. Alternatively, a block icon can also be configured as “user object”. The main advantage of a “Faceplate type” as opposed to a “User object” is the central changeability.

For each block icon of a block there is a WinCC picture of the “FPT” type. The file name has the structure “BST_<TYPICAL>_ICON.FPT”.

The following picture shows the faceplate types existing in the example project.
If it is necessary to take into account the alignment in the process picture, two icons were created. e.g.: "BST_VALVE_ICON.fpt" and "BST_VALVEH_ICON.fpt". These block icons only differ in the alignment of the status displays. The function is identical for both.

The faceplate types are configured with different objects: Status displays, graphic objects, text fields, etc. For each faceplate type an Input/Output field named “Trigger_QdwState” has been created. The “Output value” property of the object is placed outside below the property name “QdwState”.
With this property, the faceplate tag “QdwState” is connected. If the value of “QdwState” changes, a VB script is executed internally which transmits the individual bits or bit groups from “QdwState” to the internal faceplate tags. The script has been configured at the event “Selected Properties > Input/Output > Output value” of the “TRIGGER_QdwState” object.

The following tags of faceplate types are configured using the example of “BST_ANALOG_ICON.fpt”:

![Figure 6-8](image)

![Figure 6-9](image)
Note

In a faceplate type only the dynamization types “Direct tag connection” and VB script are possible.

To keep the number of process tags as low as possible (optimizing the licensing costs) all binary information of a process object are transferred to WinCC in a 32 bit tag (“QdwState”).

6.4.3 Configuring the process tag display

In order to save space there shall be no static display of the process tag name at the block icon. The process tag name shall be shown on demand for all block icons. For example, a screen shot of a process picture can then be created in which all configured process tags are labeled. Persons with little background knowledge of the actual process can with this function quickly find certain process tags in the process pictures! If the process tag names are no longer required, they can be hidden again.

For this reason, a “szTagName” text object is configured at each faceplate type and the properties “Display” and “Text” are placed on the outside.

The process tag name is configured at the “szTagNameText” property of the faceplate type. The “szTagNameVisible” property is connected with an internal WinCC tag, which is responsible for showing/hiding the text field. In the example project, the value of the tag is inverted via the “Paper clip” button.

Figure 6-10
7 Configuring Technology Subfunctions

7.1 Overview of data exchange between controller and WinCC

This section gives information on data exchange of controller and WinCC. Mainly the exchange of binary signals is discussed. The signal can be distinguished regarding the signal flow. There are status signals, control signals and telegrams.

Figure 7-1

Status signals

The technology blocks (FBs) of this application use the “QdwState” (DWORD) output to summarize different binary states in a tag (e.g. feedback “Valve OPEN”, “Valve CLOSED”, “Valve execution time monitoring”) and to transfer it to WinCC. Using the “QdwState” tag 32-binary signals (DWORD) can be transferred. The “QdwState” tag is used for dynamization of the process pictures.

Telegrams for messages

The blocks for S7-400 controllers use the ALARM_8P block (SFB35, “A8P”), to report failures, alarms or errors in the WinCC Alarm Logging. These messages requiring acknowledgement can be displayed and acknowledged during runtime in the WinCC Alarm Control.

The blocks for S7-400 controllers use the NOTITY_8P block (SFB35, “N8P”), to report operating messages (e.g. feedback (“Valve OPEN”) in the WinCC Alarm Logging. These messages not requiring acknowledgement can be displayed during runtime in the WinCC Alarm Control.
Control signals

The blocks use the Input/output signal “QdwState” (DWORD) to transfer binary control commands from the WinCC runtime (switching commands of the operator) to the control program (e.g. commands, “Valve OPEN”, “Valve CLOSED”, “Simulation On/Off”). In the “OP_dwCmd” tag up to 32 commands can be summarized.

7.2 Display of process states in the WinCC process display

The technology blocks (FBs) of this application use the output “QdwState” (DWORD) to show the different binary states (e.g. “Valve OPEN”) in the WinCC process display.

The bit assignment of the “QdwState” tag has been selected for different block types so that similar signals have the same bit position.

The following bits of “QdwState” are reserved:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>QdwState [16]</td>
<td>0 = manual 1 = automatic</td>
</tr>
<tr>
<td>QdwState [17]</td>
<td>0 = local 1 = remote</td>
</tr>
<tr>
<td>QdwState [18]</td>
<td>0 = process 1 = simulation</td>
</tr>
<tr>
<td>QdwState [24]</td>
<td>General error</td>
</tr>
<tr>
<td>QdwState [25]</td>
<td>External error</td>
</tr>
<tr>
<td>QdwState [26]</td>
<td>Error interlock</td>
</tr>
<tr>
<td>QdwState [27]</td>
<td>Interlock pending</td>
</tr>
</tbody>
</table>
7.3 Logging of messages in the WinCC Alarm Logging

The blocks for S7-400 controllers use the chronological message procedure to store messages in WinCC Alarm Logging. These messages can be displayed and acknowledged in runtime under WinCC Alarm Control.

The blocks call the block “ALARM_8P” (SFB 35) for messages requiring acknowledgement and the block “NOTIFY_8P” (SFB 31) for messages which do not require acknowledgement. The signal inputs are interconnected with the various status signals in the S7 program.

Figure 7-3

```
// Alarm_8P ============
A8P {
    EN_R := 1,
    SIG_1 := QMON_ERR,
    SIG_2 := 0,
    SIG_3 := 0,
    SIG_4 := 0,
    SIG_5 := QLOCK,
    SIG_6 := 0,
    SIG_7 := QERR_EXT,
    SIG_8 := QERR,
    ID := w#16##eee,
    EV_ID := MSG1_EVID,
    SEVERITY := w#16##40
} ;
```

```
// Notify_8P ===========
N8F {
    SIG_1 := QCLOSE,
    SIG_2 := OPENING,
    SIG_3 := OPEN,
    SIG_4 := QCLOSE,
    SIG_5 := LOCK,
    SIG_6 := QREMOTE,
    SIG_7 := QMAN_AUT,
    SIG_8 := IQSIM,
    ID := w#16##eee,
    EV_ID := MSG2_EVID,
    SEVERITY := w#16##40
} ;
```

The following signals are assigned identical for all blocks:

Table 7-2

<table>
<thead>
<tr>
<th>Signal</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>A8P [SIG_5]</td>
<td>Interlock error</td>
</tr>
<tr>
<td>A8P [SIG_6]</td>
<td>&lt;no signal&gt;</td>
</tr>
<tr>
<td>A8P [SIG_7]</td>
<td>External error</td>
</tr>
<tr>
<td>A8P [SIG_8]</td>
<td>General error</td>
</tr>
<tr>
<td>N8P [SIG_5]</td>
<td>Interlock pending</td>
</tr>
<tr>
<td>N8P [SIG_6]</td>
<td>Remote operation</td>
</tr>
<tr>
<td>N8P [SIG_7]</td>
<td>Automatic operation</td>
</tr>
<tr>
<td>N8P [SIG_8]</td>
<td>Simulation operation</td>
</tr>
</tbody>
</table>

Signals 1-4 are configured individually for each block.
The blocks for S7-300 controllers use the bit message procedure to store messages in WinCC Alarm Logging. These messages can be displayed and acknowledged in runtime under WinCC Alarm Control.

At the output of the blocks the tag “QwAlarm” (WORD) and internally a 16 bit array “QabAlarmPLC” were created for this.

Figure 7-4

```
// Bit alarm procedure  -----------------------------
QabAlarmPLC[0] :=QM0N_ERR;    // Feedback Error
QabAlarmPLC[1] :=0;
QabAlarmPLC[2] :=0;
QabAlarmPLC[3] :=0;
QabAlarmPLC[4] :=QLOCK;      // Interlock Error
QabAlarmPLC[5] :=0;
QabAlarmPLC[6] :=QERR_EXT;   // External Error
QabAlarmPLC[7] :=QERR;       // General Error
QabAlarmPLC[8] :=QCLOSE;     // CLOSE
QabAlarmPLC[9] :=QOPENING;   // OPENING
QabAlarmPLC[10] :=QOPEN;     // OPEN
QabAlarmPLC[12] :=QLOCK;     // Interlock
QabAlarmPLC[14] :=QMAN_AUT;  // Automatic
QabAlarmPLC[15] :=QSIM;      // Simulation
```

The following signals are assigned identical for all blocks:

Table 7-3

<table>
<thead>
<tr>
<th>Bit</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>QwAlarm [4]</td>
<td>Interlock error</td>
</tr>
<tr>
<td>QwAlarm [5]</td>
<td>&lt;no signal&gt;</td>
</tr>
<tr>
<td>QwAlarm [6]</td>
<td>External error</td>
</tr>
<tr>
<td>QwAlarm [7]</td>
<td>General error</td>
</tr>
<tr>
<td>QwAlarm [12]</td>
<td>Interlock pending</td>
</tr>
<tr>
<td>QwAlarm [13]</td>
<td>Remote operation</td>
</tr>
<tr>
<td>QwAlarm [14]</td>
<td>Automatic operation</td>
</tr>
<tr>
<td>QwAlarm [15]</td>
<td>Simulation operation</td>
</tr>
</tbody>
</table>

Bits 0-3 and 8-11 are configured individually for each block. With this method the messages are not transferred to WinCC and must therefore be created manually there for each block.

**Note**

Compared to the bit message procedure, the use of the STEP 7 message blocks “ALARM_8P” and “NOTIFY_8P” considerably reduces the time and effort required for WinCC message configuration. The messages are generated in the course of OS compilation with WinCC Alarm Logging.
7.4 Trigger switching commands in the WinCC process display

The WinCC control tag “OP_dwCmd” of a block is used to transmit commands from WinCC to the PLC.

The different control commands of a block type are managed centrally within WinCC Global Script in the header file “BST_HEADER.h”. The identifiers in “General BST Commands” are universal and are used by all blocks. They assign bits 16-31 of the control word. Bits 0-15 of the control word can be used for individual commands.

```c
// General BST Commands
#define BST_CMD_MANUAL 0x00010000 // DPdwCmd Bit 16
#define BST_CMD_AUTO 0x00020000 // DPdwCmd Bit 17
#define BST_CMD_LOCAL 0x00040000 // DPdwCmd Bit 18
#define BST_CMD_REMOTE 0x00080000 // DPdwCmd Bit 19
#define BST_CMD_PROCESS 0x01000000 // DPdwCmd Bit 20
#define BST_CMD_SIMULATION 0x02000000 // DPdwCmd Bit 21
#define BST_CMD_RESET 0x04000000 // DPdwCmd Bit 24

// Specific BST Commands

// BST_MOTOR_CMD (command)
#define BST_MOTOR_CMD_STOP 0x00000001 // DPdwCmd Bit 00
#define BST_MOTOR_CMD_START 0x00000002 // DPdwCmd Bit 01
#define BST_MOTOR_CMD_MONON 0x00000100 // DPdwCmd Bit 08
#define BST_MOTOR_CMD_MONOFF 0x00000200 // DPdwCmd Bit 09
```
If the operator presses the button of a faceplate in WinCC Runtime the following function is called up:

```c
int BST_COMMAND(DWORD)
```

### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DWORD dwCmd</td>
<td>Contains the respective command bit for the command to be executed. The identifiers of the command bits are defined in the header file “BST_FPDEF.h”.</td>
</tr>
</tbody>
</table>

### Returned value

<table>
<thead>
<tr>
<th>Returned value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>The function always returns the value (int) 0.</td>
</tr>
</tbody>
</table>

The following actions are performed in the script:

- The transferred command bit is written to the control tag “OP_dwCmd”. When writing the tag the prefix of the faceplate from which the call is made is always used.

**Example:**

When clicking the “Simulation ON” button of a faceplate the following C script is executed:

```c
#include "apdefap.h"
#include "BST_HEADER.h"

void OnButtonDown(char* pszPictureName, char* pszObjectName, char* pszPropertyName, UINT* nFlags, int x, int y)
{
    include "BST_HEADER.h"
    BST_COMMAND(BST_CMD_SIMULATION);
}
```

The “BST_COMMAND();” function is called and the bit (bit 21) contained in the identifier “BST_CMD_SIMULATION” is transferred.

```c
#include "apdefap.h"
#include "BST_HEADER.h"

int BST_COMMAND(DWORD dwCmd)
{
    DWORD dwCmdOld;
    dwCmdOld = GetTagDWordWait("OP_dwCmd");
    dwCmdOld |= dwCmd;
    SetTagDWordWait("OP_dwCmd", dwCmdOld);
    return (0);
}
```

The command bit is written to the “OP_dwCmd” tag.
7.5 Processing of switching commands in the control system

The PLC block analyses the control command, executes the desired action and then initiates a reset of all pending control commands by setting the value of the control tag “OP_dwCmd” to “0”.

Example:
The control program contains the instruction “Simulation ON” of an operator of WinCC (control tag “OP_dwCmd” bit 21 was set).

The following program code is executed.

```plaintext
// SIMULATION ON / OFF

IF (LIOP_SEL AND NOT L_SIM) OR (OPCmdPLC[20] AND NOT LIOP_SEL) THEN
  QSIM := FALSE;
ELSIF (LIOP_SEL AND L_SIM) OR (OPCmdPLC[21] AND NOT LIOP_SEL) THEN
  QSIM := TRUE;
QCMD_OP := FALSE;
END IF:
```

Your block is in the simulation mode (QSIM := TRUE). At the end of the block the control word is reset.

Figure 7-10

```plaintext
// reset commands

OP_dwCmd := 16#0;
```
8 Description of the Example Project

The example project shows, how the blocks in a project can be used for automating a plant. An example plant consisting of three plant sections was configured. The units consist of containers with agitators, pipes, sensors, actuators and controls.

8.1 S7 Program

The program for the automating system was configured using CFC.

A CFC chart was created for each unit. An additional “General” chart was created to show an overview of all created blocks in WinCC.

Figure 8-1
8 Description of the Example Project

The figure below shows a unit of the CFC chart “Unit_100_Tank”:

Figure 8-2

8.1.1 Unit 100

Unit 100 represents a simulated tank. The medium in the tank can be filled, drained, mixed, heated and cooled.

The following respective objects are configured in CFC chart “Unit_100_Tank”:

Table 8-1

<table>
<thead>
<tr>
<th>Object</th>
<th>Process tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BST_FF_400</td>
<td>Unit_100_Sim</td>
<td>Switching simulation mode of the actuators for Tank 100 on/off</td>
</tr>
<tr>
<td>Unit_100_Auto</td>
<td></td>
<td>Switching automatic mode of the actuators for Tank 100 on/off</td>
</tr>
<tr>
<td>Unit_100_Lock</td>
<td></td>
<td>Blocking actuators for Tank 100.</td>
</tr>
<tr>
<td>BST_ILOCK</td>
<td>LIOP_SEL</td>
<td>Blocking the HMI operation of the actuators for Tank 100.</td>
</tr>
<tr>
<td>BST_VALVE_400</td>
<td>V101</td>
<td>Supply valve</td>
</tr>
<tr>
<td>V102</td>
<td></td>
<td>Drain valve</td>
</tr>
<tr>
<td>V103</td>
<td></td>
<td>Valve heating circuit</td>
</tr>
<tr>
<td>V104</td>
<td></td>
<td>Valve cooling circuit</td>
</tr>
<tr>
<td>BST_MOTOR_400</td>
<td>E101</td>
<td>Agitators</td>
</tr>
<tr>
<td>E103</td>
<td></td>
<td>Pump heating circuit</td>
</tr>
<tr>
<td>E104</td>
<td></td>
<td>Pump cooling circuit</td>
</tr>
<tr>
<td>BST_ANALOG_400</td>
<td>LEVAN101</td>
<td>Analog value display for filling level</td>
</tr>
<tr>
<td>BST_DIGITAL_400</td>
<td>WH_T100</td>
<td>Upper filling level mark</td>
</tr>
<tr>
<td></td>
<td>WL_T100</td>
<td>Lower filling level mark</td>
</tr>
</tbody>
</table>
8 Description of the Example Project

8.1.2 Unit 200

Unit 200 represents a simulated tank. The medium in the tank can be filled, drained, mixed and heated.

The following respective objects are configured in CFC chart “Unit_200_Tank”:

<table>
<thead>
<tr>
<th>Object</th>
<th>Process tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BST_FF_400</td>
<td>Unit_200_Sim</td>
<td>Switching simulation mode of the actuators for Tank 200 on/off</td>
</tr>
<tr>
<td></td>
<td>Unit_200_Auto</td>
<td>Switching automatic mode of the actuators for Tank 200 on/off</td>
</tr>
<tr>
<td></td>
<td>Unit_200_Lock</td>
<td>Blocking actuators for Tank 200.</td>
</tr>
<tr>
<td>BST_ILOCK</td>
<td>LIOP_SEL</td>
<td>Blocking the HMI operation of the actuators for Tank 200</td>
</tr>
<tr>
<td>BST_VALVE_400</td>
<td>V201</td>
<td>Supply valve</td>
</tr>
<tr>
<td></td>
<td>V202</td>
<td>Drain valve</td>
</tr>
<tr>
<td>BST_MOTOR_400</td>
<td>E201</td>
<td>Agitators</td>
</tr>
<tr>
<td>BST_ANALOG_400</td>
<td>LEVAN201</td>
<td>Analog value display for filling level</td>
</tr>
<tr>
<td>BST_DIGITAL_400</td>
<td>WH_T200</td>
<td>Upper filling level mark</td>
</tr>
<tr>
<td></td>
<td>WL_T200</td>
<td>Lower filling level mark</td>
</tr>
<tr>
<td>BST_COUNT</td>
<td>LEV201</td>
<td>Filling level simulation Unit 100</td>
</tr>
<tr>
<td>BST_PIDCTRL_400</td>
<td>CTU200</td>
<td>Control for container temperature</td>
</tr>
<tr>
<td>BST_LAG</td>
<td>U200_TEMP</td>
<td>Simulation of the temperature for Unit 200 (PT1 – controlled system)</td>
</tr>
</tbody>
</table>
8.1.3 Unit 300

Unit 300 represents a pump station for transferring the medium from Unit 200 to Unit 100.

The following respective objects are configured in CFC chart “Unit_300_Balance”: Table 8-3

<table>
<thead>
<tr>
<th>Object</th>
<th>Process tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BST_VALVE_400</td>
<td>V301</td>
<td>Valve</td>
</tr>
<tr>
<td>BST_MOTOR_400</td>
<td>E301</td>
<td>Pump</td>
</tr>
</tbody>
</table>

8.2 Process pictures

In the example project the following WinCC pictures were created which are displayed in the workspace of WinCC Runtime:

Figure 8-3
8.2.1 **Plant process picture**

The picture shows, how the blocks created for this application can be used in a plant. The process picture is displayed at Runtime start or can be called via the “Home” button.

The picture is saved in the WinCC project under the name “Overview.pdl”.

Figure 8-4
8.2.2 Alarm picture

This screen displays all currently pending alarm messages. These can be acknowledge here at a central location.

The picture is saved in the WinCC project under the name “Alarm.pdl”.

Figure 8-5
8.2.3 Overview of the Blocks

This screen displays all blocks of this example application. For a process connection the CFC chart “General” was created and all S7 blocks were also configured. In this screen the blocks are also represented which were not adopted in the process picture. These are:

- BST_SIMODIR
- BST_SIMOREV
- BST_MM4
- BST_SINAG120

These blocks here can only run in simulation mode.

In this screen you can also have the interfaces and icon configurations displayed for all blocks.

The picture is saved in the WinCC project under the name “General.pdl”.

Figure 8-6
8.2.4 PC System Information

This screen is used for system diagnosis. The current values use the special WinCC driver channel “SYSTEM INFO”. You receive information on the processor load, storage load as well as load of the data carrier.

The picture is saved in the WinCC project under the name “PCDiagSysInfo.pdl”.

Figure 8-7

Note

This process picture is taken from the example application for monitoring the SIMATIC PC hardware. The example project is also offered to you in this documentation as a download. However, please note that a complete function requires special SIMATIC PC hardware.
8 Description of the Example Project

8.3 Scenarios of the example system

8.3.1 Filling and emptying the containers

The filling levels of both containers can be regulated via the inlet valves (V101, V201) and the outlet valves (V102, V202). As illustrated in the process picture opening the valve “V101” causes a filling of the container “Unit 100”. The current filling level is displayed at the bar and as decimal value at the analog block. If the container is full, this is displayed at the digital block “U100HL”. “U100LL” shows if the container is empty.

Figure 8-8

The figure displays the faceplate of the valve (V101). Valve (V101) is opened, valve (V102) is closed. The tank is filled.

8.3.2 Temperature control Unit 100

The temperatur of the medium is simulated using the block (U100_TEMP). The controller (CTU100) uses the block (U100_SPLIT) to control the binary control of the valves (V103, V104) and pumps (E103, E104). If the setpoint value (SP) is changed at the faceplate of the control, it controls the valve and the motor accordingly for cooling or heating. The valves and pumps of the heat exchanger are for this reason not enabled for operator control (local operation and automatic operation).

Raising the temperature opens the valve (V103) and starts the pump (E103). If the heating circuit is activated, the actuators (V104, E104) of the cooling circuit are locked (Interlock). They can neither be activated by another process.
8.3.3 Transferring the medium

Using the valve (V301) and the pump (E301) of “Unit 300” the medium can be transferred from “Unit 200” to “Unit 100”. The requirement for transferring the medium is that the tank (Unit 100) is not full and the tank (Unit 200) is not empty. If one of both cases has occurred, the valve (V301) cannot be opened (Interlock). The motor (E301) can only be started if the valve (V301) was opened.
In the figure the valve (V301) is opened and the motor (E301) is started. The medium is transferred from “Unit 200” to “Unit 300” until either the pump has stopped, the tank “Unit 200” is empty, or the tank “Unit 100” is full.

The analog displays (U100_VALUE) and (U200_VALUE) show the filling level as decimal value and are configured with alarm and warning limits. The messages are visible in the alarm view of the respective faceplates or in the “Alarm picture” of the project.

If one of the states reaches “Unit 100” full or “Unit 200” empty, valve (V301) and pump (E301) are blocked and go to the state “Interlock error”. The error must be reset prior to renewed operation.

### 8.3.4 Controlling actuator groups

Higher-level automatic functions (controllers, step chains) usually control (valves, pumps) to affect the process.

Section “Process Mode Unit 100” has configured switches which together can switch the groups of actuators to different operating states (e.g. simulation, automatic).

The actuators (V101, V102, E101) are interconnected so that the operating mode can be changed via the switches “Simulation” (Unit_100_Sim), “Automatic” (Unit_100_Auto) and “Interlock” (Unit_100_Lock).
The switches Simulation and Automatic are active, the valves (V101, V102) and the motor (E101) are therefore in simulated automatic. As long as one of the switches is active, the actuators for operation in WinCC are blocked, the “Local mode” is displayed again with signal “HMI Settings locked”. If all states (Simulation, Automatic and Interlock) have been reset again, the respective blocks must be set manually to remote operation at the faceplate so they can be operated again.
Appendix

9 Links & Literature

9.1 Bibliography

This list is not complete and only represents a selection of relevant literature.

Table 9-1

<table>
<thead>
<tr>
<th>Topic</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>/1/ STEP7 SIMATIC S7-300/400</td>
<td>Automating with STEP7 in STL and SCL Author: Hans Berger Publicis Publishing ISBN: 978-3895784125</td>
</tr>
</tbody>
</table>

9.2 Internet links

The following list is by no means complete and only provides a selection of appropriate sources.

Table 9-2

<table>
<thead>
<tr>
<th>Topic</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>/1/ Reference to this entry</td>
<td><a href="http://support.automation.siemens.com/WW/view/en/68679830">http://support.automation.siemens.com/WW/view/en/68679830</a></td>
</tr>
<tr>
<td>/2/ Siemens Industry Online Support</td>
<td><a href="http://support.automation.siemens.com">http://support.automation.siemens.com</a></td>
</tr>
<tr>
<td>/3/ Reference to this entry</td>
<td>How can an existing WinCC project be integrated into a STEP 7 project? <a href="http://support.automation.siemens.com/WW/view/en/11841504">http://support.automation.siemens.com/WW/view/en/11841504</a></td>
</tr>
<tr>
<td>/4/ Reference to this entry</td>
<td>How to proceed if “chronological alarm reporting” shall be used but first WinCC and then STEP 7 were installed? <a href="http://support.automation.siemens.com/WW/view/en/22272911">http://support.automation.siemens.com/WW/view/en/22272911</a></td>
</tr>
<tr>
<td>/5/ Reference to this entry</td>
<td>How are message classes used if WinCC is integrated in the STEP 7 project? <a href="http://support.automation.siemens.com/WW/view/en/31622970">http://support.automation.siemens.com/WW/view/en/31622970</a></td>
</tr>
<tr>
<td>/6/ Reference to this entry</td>
<td>How are message texts used if WinCC is integrated in the STEP 7 project? <a href="http://support.automation.siemens.com/WW/view/en/30550239">http://support.automation.siemens.com/WW/view/en/30550239</a></td>
</tr>
<tr>
<td>/7/ Reference to this entry</td>
<td>How can the faceplate which is part of a user object be opened during Runtime? <a href="http://support.automation.siemens.com/WW/view/en/24193022">http://support.automation.siemens.com/WW/view/en/24193022</a></td>
</tr>
<tr>
<td>/8/ Reference to this entry</td>
<td>How can you have hardware diagnostics information (hard disk status, temperature, fan status, UPS and WinAC RTX) of SIMATIC PCs of the “B generation” displayed in WinCC Runtime? <a href="http://support.automation.siemens.com/WW/view/en/29855065">http://support.automation.siemens.com/WW/view/en/29855065</a></td>
</tr>
<tr>
<td>/9/ Reference to this entry</td>
<td>How can user-defined operating messages be created? <a href="http://support.automation.siemens.com/WW/view/en/24325381">http://support.automation.siemens.com/WW/view/en/24325381</a></td>
</tr>
<tr>
<td>/10/ Reference to this entry</td>
<td>How can texts of enumerations (in Shared Declarations in the SIMATIC Manager) be used for display in WinCC? <a href="http://support.automation.siemens.com/WW/view/en/27147567">http://support.automation.siemens.com/WW/view/en/27147567</a></td>
</tr>
<tr>
<td>/11/ Reference to this entry</td>
<td>How can messages in the process picture be displayed with the smart object “Group display” if the bit message or analog message procedures are used? <a href="http://support.automation.siemens.com/WW/view/en/17778440">http://support.automation.siemens.com/WW/view/en/17778440</a></td>
</tr>
</tbody>
</table>
### 9.3 Further Example Blocks

The following list is by no means complete and only provides a selection of appropriate sources.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Title</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>\1\</td>
<td>Example Blocks for WinCC (TIA Portal) and STEP 7 (TIA Portal)</td>
<td><a href="http://support.automation.siemens.com/WW/view/en/68839614">http://support.automation.siemens.com/WW/view/en/68839614</a></td>
</tr>
<tr>
<td>\2\</td>
<td>Example Blocks for WinCC V7 and STEP 7 (TIA Portal)</td>
<td><a href="http://support.automation.siemens.com/WW/view/en/31624179">http://support.automation.siemens.com/WW/view/en/31624179</a></td>
</tr>
</tbody>
</table>
## History

Table 10-1

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Modifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>04.11.2008</td>
<td>First issue</td>
</tr>
<tr>
<td>V1.1</td>
<td>17.11.2008</td>
<td>Chapter 5.5 revised.</td>
</tr>
<tr>
<td>V1.2</td>
<td>27.04.2009</td>
<td>Foreword “Structure of the Application” added</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Name of divisions changed (A&amp;D → IIA)</td>
</tr>
<tr>
<td>V2.0</td>
<td>18.09.2009</td>
<td>Adjusting the example project. Using the new block with standardized interface.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Adjusting the documentation to the new example project</td>
</tr>
<tr>
<td>V2.1</td>
<td>27.04.2010</td>
<td>Adding the block “BST_SIPIART”</td>
</tr>
</tbody>
</table>