Procedure

CREATION OF A SIEMENS S7 UNICOS-CPC 6 APPLICATION

Abstract

This procedure explains how to create a Siemens S7 UNICOS CPC 6 application from the specifications using the UAB (UNICOS Application Builder) generation tool.
## HISTORY OF CHANGES

<table>
<thead>
<tr>
<th>REV. NO.</th>
<th>DATE</th>
<th>PAGES</th>
<th>DESCRIPTIONS OF THE CHANGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1.0</td>
<td>2011-12-15</td>
<td>All</td>
<td>First version (Silvia Izquierdo)</td>
</tr>
<tr>
<td>V1.1</td>
<td>2011-12-20</td>
<td>All</td>
<td>General corrections by E. Viñuela</td>
</tr>
<tr>
<td>V1.2</td>
<td>2012-02-24</td>
<td>All</td>
<td>General corrections after J. Ortola revision</td>
</tr>
<tr>
<td>V1.3</td>
<td>2012-03-15</td>
<td>All</td>
<td>General corrections after I. Prieto revision</td>
</tr>
<tr>
<td>V1.4</td>
<td>2012-03-16</td>
<td>All</td>
<td>General corrections after B. Bradu revision</td>
</tr>
<tr>
<td>V1.5</td>
<td>2012-03-20</td>
<td>All</td>
<td>General corrections after E. Viñuela revision</td>
</tr>
<tr>
<td>V1.6</td>
<td>2012-03-27</td>
<td>18</td>
<td>Information about WinCCFlex generator added</td>
</tr>
<tr>
<td>V1.7</td>
<td>2012-05-08</td>
<td>All</td>
<td>References added</td>
</tr>
<tr>
<td>V1.8</td>
<td>2012-05-08</td>
<td>22</td>
<td>WinCC OA output file name changed</td>
</tr>
<tr>
<td>V1.9</td>
<td>2012-06-27</td>
<td>36</td>
<td>References updated</td>
</tr>
<tr>
<td>V2.0</td>
<td>2012-06-29</td>
<td></td>
<td>Change references to point to EDMS for release v1.3.1</td>
</tr>
<tr>
<td>V2.1</td>
<td>2013-06-27</td>
<td></td>
<td>Redundant PLC case added (Tiago Rocha)</td>
</tr>
<tr>
<td>V2.2</td>
<td>2013-10-16</td>
<td>36</td>
<td>Error in _CL description corrected</td>
</tr>
<tr>
<td>V2.3</td>
<td>2013-05-30</td>
<td>9</td>
<td>Comment added for Connection resource 03</td>
</tr>
</tbody>
</table>
# Table of Contents

1. INTRODUCTION ..................................................................................................... 5  
2. REQUIREMENTS .................................................................................................... 6  
3. PROCEDURE .......................................................................................................... 6 
   3.1 INSTALLATION OF UAB GENERATOR TOOL ......................................................... 6  
   3.2 FILL IN SPECIFICATIONS .................................................................................... 6  
   3.3 CONFIGURATION ............................................................................................... 7  
      3.3.1 ARCHITECTURE OVERVIEW ............................................................................. 7  
      3.3.2 PLC CONFIGURATION ..................................................................................... 8  
      3.3.3 WINCC OA CONFIGURATION ........................................................................ 10  
      3.3.4 REDUNDANT PLC CONFIGURATION ............................................................... 11  
   3.4 UAB GENERATION TOOL ................................................................................... 13  
      3.4.1 GENERATION PRINCIPLES .............................................................................. 13  
      3.4.2 GENERATION PROCEDURE .......................................................................... 14  
      3.4.3 UAB USER REPORT ....................................................................................... 21  
      3.4.4 COMMON ERRORS ......................................................................................... 21  
      3.4.5 LOG ............................................................................................................. 24  
      3.4.6 GENERATED FILES ....................................................................................... 24  
      3.4.7 SEMANTIC RULES ......................................................................................... 25  
      3.4.8 TEMPLATES .................................................................................................. 25  
   3.5 IMPORTATION INTO SIEMENS S7 ........................................................................... 26  
   3.6 BASELINE IMPORTATION .................................................................................. 26  
      3.6.1 IMPORT INSTANCE AND LOGIC FILES ............................................................ 27  
      3.6.2 IMPORT SYMBOL TABLE ................................................................................ 30  
   3.7 PLC SOURCES COMPILATION ............................................................................. 31  
   3.8 SPECIFIC LOGIC CREATION .............................................................................. 32  
      3.8.1 CREATION OF A GRAFCET IN S7 ................................................................. 32  
      3.8.2 COMPLETE LOGIC FILES ............................................................................. 36  
   3.9 CREATION OF WINCC OA PROJECT ................................................................... 36  
      3.10 USAGE OF LOCAL PANELS .............................................................................. 36  
4. TROUBLESHOOTING ............................................................................................. 37  
5. SUPPORT ............................................................................................................. 38  
6. REFERENCES ......................................................................................................... 38
## TABLE OF FIGURES

Figure 1 - General procedure .............................................................. 5  
Figure 2 - Siemens Front End Device IO Configuration ............................ 6  
Figure 3 - FE Encoding Type and PLC channel ...................................... 7  
Figure 4 - Example of a decentralized configuration ............................... 7  
Figure 5 - New connection .................................................................. 8  
Figure 6 - Properties S7 connection ...................................................... 8  
Figure 7 - Address Details .................................................................... 9  
Figure 8 - Fields matching between the generator and the PLC configuration ... 9  
Figure 9 - WINCC OA final configuration ............................................. 10  
Figure 10 - UAB Generator window for S7-400H with redundant mode .... 11  
Figure 11 - WINCC OA Final Redundant Configuration ......................... 12  
Figure 12 - UAB CPC Wizard ............................................................. 13  
Figure 13 - UAB Bootstrap ................................................................. 14  
Figure 14 - UNICOS Application Builder ............................................. 14  
Figure 15 - Application General Data .................................................. 15  
Figure 16 - SIEMENS PLC Specifications ............................................. 16  
Figure 17 - SIEMENS Generators ........................................................ 16  
Figure 18 - S7 Instance generator ........................................................ 17  
Figure 19 - S7 Logic Generator ........................................................... 18  
Figure 20 - WINCC OA Instance Generator ......................................... 19  
Figure 21 - WinCC Flexible Instance Generator .................................... 20  
Figure 22 - UAB User Report ............................................................... 21  
Figure 23 - Succeed status .................................................................. 21  
Figure 24 - Failure status ..................................................................... 22  
Figure 25 - Required value is not specified ......................................... 22  
Figure 26 - Empty mandatory field ...................................................... 22  
Figure 27 - Exceeding maximum name length ..................................... 22  
Figure 28 - Not mandatory fields can become mandatory ..................... 23  
Figure 29 - Object does not exist ....................................................... 23  
Figure 30 - S7 Instance generator output ............................................ 24  
Figure 31 - Project Resources folders .................................................. 25  
Figure 32 - Opening library ................................................................. 26  
Figure 33 - Baseline sources importation ............................................. 27  
Figure 34 - Baseline blocks importation .............................................. 27  
Figure 35 - Insert external source ....................................................... 28  
Figure 36 - Importing instances files ................................................... 28  
Figure 37 - Project structure ............................................................... 29  
Figure 38. Open symbols table ............................................................ 30  
Figure 39. Importing symbols ............................................................. 30  
Figure 40. Importing symbols ............................................................. 30  
Figure 41. Baseline compilation .......................................................... 31  
Figure 42 - Grafcet: Change block number ........................................ 32  
Figure 43 - Grafcet: Block settings ...................................................... 32  
Figure 44 - Grafcet: Application Settings ............................................ 33  
Figure 45 - Grafcet: creation of transition variables ............................ 33  
Figure 46 - Grafcet: transitions ........................................................... 33  
Figure 47 - Grafcet: Changing the DB name ....................................... 34  
Figure 48 - Grafcet .......................................................... 34  
Figure 49 - Set time of day window in PLC ....................................... 37  
Figure 50 - Opening Customize options ............................................. 37  
Figure 51 - Customize window ............................................................ 37
1. INTRODUCTION

The goal of this document is to provide a procedure of how to create a Siemens S7 UNICOS application using the Siemens S7 PLC (S7-400 or S7-300 or S7-300 PN/DP).

The general architecture of a UNICOS S7 application can be found in the “UNICOS CPC6 on Siemens S7” document [1].

The general steps for the creation of a S7 UNICOS-CPC project are summarized as:
1. Filling in the specification file according to the application (Spec.xml).
2. Generation
   2.1. Running S7 Instance Generator to get the instances code for the PLC.
   2.2. Running S7 Logic Generator to get the control logic code for the PLC.
   2.3. Running WinCC OA Instance Generator to get the importation file for the supervision.
   2.4. Running WinCC Flexible Instance Generator to get importation files for a touch panel (if needed).
3. Importation of the object instances source files into the PLC project (Simatic Manager)
4. Importation of logic source files into the PLC project (Simatic Manager)
5. Completion of the Logic in the PLC project (It can also be done by using User Templates)
6. Downloading project to the PLC
7. Creation of the WinCC OA project
8. Importation of instances into WinCC OA
9. Completion of panels for supervision

Figure 1 - General procedure
2. REQUIREMENTS

Developing an S7 UNICOS application implies the usage of several software packages (minimum versions required are specified):

- Specification and Generation tools: **MS Office (Excel v2008)**, **Java SE Runtime Environment** (v6) and **UAB** (v1.3.x)
- PLC: Simatic **Step 7** (V5.5)
- SCADA: **PVSS/WinCCOA**¹ (v3.8)

3. PROCEDURE

3.1 INSTALLATION OF UAB GENERATOR TOOL

The different tools specified in the section “REQUIREMENTS” must be installed. For the installation of UAB tool follow the instructions described in the following link.

http://cern.ch/enice/Automatic+Generation+tools+%28UAB%29

3.2 FILL IN SPECIFICATIONS

The specification file is an *xml* file where all the UNICOS objects must be defined and parameterized. To know about how to fill the specification file check the document “UCPC Spec Documentation” [2].

If you were a UNICOS user, you will notice there are some changes in the specifications. To convert a specification file from CPC5 to CPC6 you can use the “CPC5 to CPC6 Spec Conversion Tool” [3].

Concerning the PLC I/O periphery addressing, is necessary to look at document “FEEncodingType.xlsx”, you can find it inside UAB project in “...\UAB_Project\Specs”. You can see a screenshot of this document in Figure 2.

![Siemens FE Device IO Configuration](image)

Figure 2 - Siemens Front End Device IO Configuration

¹ The software used for the supervision **PVSS** is nowadays known as **WinCC Open Arquitecture** (WinCC OA), in general in this tutorial we will be referred to that software as WinCC OA
I.e. for AI reading from periphery, we would fill the specification as can be seen in the Figure 3.

<table>
<thead>
<tr>
<th>DeviceIdentific</th>
<th>FE Encoding Type</th>
<th>InterfaceParam1</th>
<th>InterfaceParam2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 3 - FE Encoding Type and PLC channel

### 3.3 CONFIGURATION

#### 3.3.1 ARCHITECTURE OVERVIEW

The SCADA (WINCC OA) and the PLC are linked by an Ethernet Network, the exchange of data is done by a dedicated communications protocol, the so-called TSPP (Time-Stamp Push Protocol) \[4\]. This protocol allows the PLC to push the data only on change eliminating unnecessary data flow between the PLC and the supervision.

The Input/Output of the PLC could be on local cards (same rack/rail) and/or deported equipment as Profibus DP or Profibus PA (Figure 4).

![Diagram of SCADA and PLC architecture](attachment:image.png)

**SCADA (PVSS)**

- IP: 137.130.194.100
- Rack: 0
- Slot: 0
- Connection Resource: 03

**PLC**

- IP: 137.138.194.104
- Rack: 4
- Slot: 4
- Connection Resource: A0

**PROFIBUS DP**

**PROFIBUS PA**

Figure 4 - Example of a decentralized configuration
3.3.2 PLC CONFIGURATION

A new S7 connection must be created in Simatic Step 7 to have a network link with WINCC OA:

1. Open NetPro (Connections)
2. Select the CPU of the PLC and insert a new connection
3. Select ‘Unspecified station’ and a ‘S7 connection’ type
4. Validate clicking the OK button

5. Fill in the different fields according to the hardware which has been specified for your application

Create a S7 Connection

![Create a S7 Connection](image_url)

Figure 5 - New connection

![Properties S7 connection](image_url)

Type the ID of your connection

The connection must **NOT be active**. Uncheck this box

The partner must be the PVSS machine with its IP number

PLC information

Figure 6 - Properties S7 connection
The parameters chosen have to match with the ones that will be used in the UAB CPC wizard when generating the application; otherwise the communication between the PLC and PVSS won’t work. Note that "Connection Resource" 03 for Partner (WinCC OA) is reserved by Siemens and cannot be used for the communication.
3.3.3 WINCC OA CONFIGURATION

The WINCC OA configuration will be generated automatically according to parameters introduced during the generation procedure. Figure 9 only shows the final configuration in the WINCC OA S7 parameterization for a diagnostics purpose. Two connections will be automatically configured, one for the TSPP protocol and another one for the S7 polling/writing link.

Figure 9 - WINCC OA final configuration
3.3.4 REDUNDANT PLC CONFIGURATION

The procedure to configure a Redundant PLC (S7-400H) connected with one data server, in Step7 and WINCC OA will be explained in this section.

3.3.4.1 Step7 Redundant Configuration

1. Create a New Project: File>>New...
2. Insert SIMATIC H Station: Insert>>Station>>SIMATIC H Station
3. Select the Station created and double click the object Hardware to open HW Config
4. Select the components of your system and drag and drop them into the main window (If you don’t find the S7-400H CPU on the list, it might be necessary to install the HotFix1 for SIMATIC STEP 7, it can be downloaded here)
5. After configuring all the hardware, open NetPro
6. Follow the procedure presented in 3.3.2 for both PLC.

Figure 10 show the generated window when an S7-400H PLC is selected, the field should match the ones introduced in the HW Config and NetPro.
3.3.4.2 WinCC OA Redundant Configuration

The WINCC OA configuration will be **generated automatically** according to parameters introduced during the generation procedure. Figure 11 only shows the final configuration in the **WINCC OA S7 parameterization** for a diagnostics purpose. Two connections will be automatically configured, one for the TSPP protocol and another one for the S7 polling/writing link.

![Figure 11 – WINCC OA Final Redundant Configuration](image-url)
3.4 UAB GENERATION TOOL

3.4.1 GENERATION PRINCIPLES
The UAB CPC-Wizard (UNICOS Application Builder) consolidates all generation steps in one tool, making easier and intuitive the generation procedure.

The generators used for a S7 application are four:
- S7 Instance Generator: it generates the instances code for the PLC.
- S7 Logic Generator: it generates the control logic code (place holders and/or complete logic) for the PLC and the interconnections between PCOs and Field Objects.
- WinCC OA Instance Generator: it generates the importation file for the supervision.
- WinCC Flexible Instance Generator: it generates the instance files used for touch panel applications.
- Expert User Generator. It provides a way to execute user scripts to generate specific additional files.

In Figure 12 you can see the window where you can choose which generator you want to use.

![Figure 12 - UAB CPC Wizard](image)
3.4.2 GENERATION PROCEDURE

Run the UCPC-Wizard from the UAB Bootstrap by clicking on `cpc-wizard`.

![Figure 13 - UAB Bootstrap](image)

On *UAB CPC-Wizard* main window, choose ‘New UNICOS Application’, then choose *Siemens* as ‘Target Platform’ and finally choose the latest resource package (other resource package versions are available for compatibility with different projects). Finally, you must choose the project location and click ‘Run.’

![Figure 14 - UNICOS Application Builder](image)
In the ‘Application General Data’ window write a **name for the project** and a **name for the application**. Then we specify the **specification file**. The rest of the fields are optional. The ‘User Tag’ field introduces a tag to indentify and categorize your application visible at the SCADA level. Click ‘Next’ for next step.

![Figure 15 - Application General Data](image)

In the ‘SIEMENS PLC Specifications’ window we set the PLC configuration. Write a **name for the PLC** (please use the proper naming for the PLCs following CERN-wide recommendations, e.g.: CFP-864-MyPLC), write its **IP address** and choose the **PLC Type** (S7-300, S7-300 PN/DP or S7-400). Keep in mind that there is a difference within the S7-300 family whether it has the communications processor embedded or not.

Then set the **WINCC OA configuration** (Rack, Slot, Partner Con. Resource, Timeout) and the **PLC connection configuration**, and click ‘Next’. The PLC configuration must take into account the PLC settings (e.g. other connections may be already in place).
After this, we get the ‘SIEMENS Generators’ window, where we are able to choose between the four generators: S7 Instance Generator, S7 Logic Generator, WINCC OA Instance Generator and WinCC Flexible Instance Generator. To generate a complete application the four generators must be used (WinCC Flexible Instance Generator is needed just if you are using a touch panel for local application).
**S7 Instance Generator**

To get the PLC Instance source files it is necessary to run the *S7 Instance Generator*. Choose ‘S7 Instance Generator’ in Figure 17 and click “Next”. In Figure 18 you can see the generator and an explanation of the different options and fields of this generator.

It is possible to select all object types at once by clicking ‘Select All’ button, then you can click on ‘Generate’ to get the instance generation. The generation can be found on the folder “…\Project\Output\S7CodeGenerator\”.

It is recommendable to generate everything the first time and then use partial generation whenever you need to have individual generation of devices (e.g. Analog input only). In the case a partial generation is used, it is possible to select in the button “Global files scope” if the user wants the global files to be generated according to all objects (“All types” option) or simply taking into account the selected objects (“Selected Types” option).

In the “Post Process User Template” option, the user can execute an additional action for the instance generator by executing the template declared in the field “Post Process User Template”.

To do the commissioning of input/output signals (instrumentation, actuators) the option “Generate Commissioning File” has to be selected, then a check list is generated in your project. You can find the file in “…\Project\Output\S7CodeGenerator\S7InstanceGenerator\”.

It is possible to click on ‘Logic’ button to switch to *S7 Logic Generator*. 
S7 Logic Generator

In Figure 19 you can see the S7 Logic Generator window, and an explanation of its different options.

To generate all the logic files, it is necessary to select all object Types, masters, and finally all the logic sections. To do so, it is possible to use ‘Select All’ buttons. To get the logic generation you click on ‘Generate’ button. Generation can be found on “…\Project\Output\S7LogicGenerator\”.

In the “Post Process User Template” option, the user can execute an additional action for the Logic generator by executing the template declared in the field “Post Process User Template”.

As in the Instance Generator, the Logic Generator allows partial generation of logic files. In the case a partial generation is used, it is possible to select in the button “Global files scope” if the user wants the global files to be generated according to all objects (“All types” option) or simply taking into account the selected objects (“Selected Types” option).
**WINCC OA Instance Generator**

To get to *WINCC OA Instance Generator* you can click on ‘WINCC OA’ button on the previous window (Figure 19).

It is possible to select all device types by clicking ‘Select All’ button, and then just click ‘Generate’ button to get the generation started. The WINCC OA importation file generated can be found on “\Project\Output\WinCCOAInstanceGenerator\”.

![Figure 20 - WINCC OA Instance Generator](image-url)
**WinCC Flexible Instance Generator**

This generator is just used if you are using a touch panel for local operation. To get to *WinCC Flexible Instance Generator* you can click on ‘WinCC Flex’ button on the previous window (Figure 20).

It is possible to select all device types by clicking ‘Select All’ button, and then just click ‘Generate’ button to get the generation started. The WinCC Flexible importation files generated can be found on “…\Project\Output\WinCCFlexInstanceGenerator\”.

To know more about how to create touch panel applications take a look at the procedure “Creating and designing a local panel for a UNICOS application” [5].
3.4.3 UAB USER REPORT

To check the status of the different generation steps and the errors or warnings produced, you can check the UAB User Report.

It is possible to filter the different messages in the report by selecting/unselecting the message type buttons. The meaning of the buttons can be seen in figure 16. It is also possible to clear all messages appearing in the report by clicking ‘Clear’ button, and to copy the messages to clipboard using ‘Copy To Clipboard’ button.

Figure 22 - UAB User Report

If the generation has been successful, you can read next message in the report depending on the generator used:

“The exit status of the S7CodeGenerator plug-in is SUCCEED”
“The exit status of the S7LogicGenerator plug-in is SUCCEED”
“The exit status of the PVSSInstanceCodeGenerator plug-in is SUCCEED”

Figure 23 - Succeed status

3.4.4 COMMON ERRORS

If there is any error in the generation, the generation status is FAILURE, as you can see in figure 18. We will comment some of the most common errors.
There are different types of errors. If the user forgets to fill a mandatory field, he will get an error like the one in figure 19.

As you can see, the message indicates the object type, the name and the field. The object type in the figure is ‘ProcessControlObject’ called “DEMON_1_DemonPCO”, instance number 1, and the field is ‘FEDeviceParameters:ParReg:Manual Restart after Full Stop’, like this, it is very easy to identify the field in the spec and fill it to solve the error.

There is a maximum size for object names: for Siemens it is 17 letters for Alarms, 19 letters for PCO and 21 for all other objects, and it is not possible to use any special characters (space, !, &, ^, %, #...). If the user exceeds the maximum length, there will be an error message like the one in figure 21.
Some options transform not mandatory fields into mandatory. We can see an example in figure 22: when an Analog Alarm is a Full Stop, master definition is mandatory.

![Figure 28 - Not mandatory fields can become mandatory](image)

If the user commits a mistake writing object names in the specification file, he will get an error as the one in figure 23, which says that the object doesn’t exist. To solve this error check the device name.

![Figure 29 - Object does not exist](image)
3.4.5 LOG
Every time the user runs a generation, a log file is created with general information and the warning and error messages. This file can be found in “<User project path>\Log\”. There is also a log file called ‘UABLogging.html’ in which all generation messages are recorded, it is updated with every generation.

3.4.6 GENERATED FILES
All generated files can be found in “<user project path>\Output”. The output generated is distributed in four folders: WinCCOAInstanceGenerator, S7InstanceGenerator, S7LogicGenerator and WinCCFlexInstanceGenerator. The Instance Generation output is inside the folder called S7InstanceGenerator. The Instance Generator generates:
- The symbol table: “Symbol.sdf”
- three compilation files: “1_Compilation_Baseline.INP”, “2_Compilation_instance.INP” and “4_Compilation_OB.scl”
- recipes mechanism file “Recipes.SCL”
- one source file for each object type: “AA.SCL”, “ALSCL”, “ANADIG.SCL”, etc. It is necessary to import these files into S7.

![Figure 30 - S7 Instance generator output](image)

The Logic Generator output is inside the folder called S7LogicGenerator. It generates:
- a compilation file “3_Compilation_LOGIC.INP”
- all the logic files: “FC_CONTROLLER.SCL”, “FC_PCO_Logic.SCL”, “DB_ERROR_SIMU.SCL”, etc. It is necessary to import these files into S7.

The WinCC OA Instance Generator generates a file called “wincc_oa_db_file_ApplicationName.txt” inside WinCCOAInstanceGenerator, this is the importation file the user will import from WinCC OA.
The WinCC Flexible Instance Generator generates five files:
- an importation file for the alarms called “winccFlex_alarm.csv”
- a text file for the main script called “winccFlex_script.txt”
- a text file for the security settings script called “winccFlex_securitySettingsScript.txt”
- an importation file with the variables called “winccFlex_tags.csv”
- an importation file with text lists called “winccFlex_textLists.csv”

3.4.7 SEMANTIC RULES
The templates used for the semantic rules analyse the specifications file against the existing constraints (if applicable) to find out if there is any semantic error. These rules take into account the existing rules into PLC and WinCC OA.

You can check this constraint for the different fields in "UCPC Spec Documentation" [2].

Example: it checks the maximum length of names mentioned before in the section.

3.4.8 TEMPLATES
The templates used by generators are inside the folder “<user project folder>\Resources”. They are divided in several folders (Figure 31):
- **Device types**: description of the objects
- **IOCommissioning**: I/O commissioning check list

The following folders are the ones containing the templates used for the generation of code or importation files:
- **WinCCOAInstanceGenerator**: templates used to generate the WinCC OA importation file
- **S7InstanceGenerator**: templates used to generate instances generation
- **S7LogicGenerator**: templates used to generate logic generation
- **SemanticCheckRules**: templates used to check semantic rules
- **Upgrade**: templates used to upgrade projects
- **WinCCFlexInstanceGenerator**: templates used to generate the importation files for touch panel applications

![Figure 31 - Project Resources folders](image)

To know more about the usage of User Templates you can check the document “Logic Template Usage for UNICOS-CPC Applications” [6].
3.5 IMPORTATION INTO SIEMENS S7

Before importing source and configuration files which have been generated you must create a S7 Project in the Simatic Manager (Step 7) with your specific hardware configuration.

3.6 BASELINE IMPORTATION

You must import the **UNICOS Baseline** for S7 PLC which is common for all applications.

- In the Simatic Manager tool, click on 'File > Open'.
- Select **Libraries**, and look for the S7 baseline library. It is in "<user project path>\Baseline" (it needs to be unzipped first).

![Opening library](image)

**Figure 32 - Opening library**

- The communication mechanism between the PLC and the SCADA is automatically configured based on one of the three function blocks defined below:
  - FB12 "BSEND_S7300" in case the SCADA is communicating with a S7 300 with external Ethernet Communication Processor.
  - FB14 "BSEND_S7300PNDP" in case the SCADA is communicating with a S7 300 with embedded Ethernet Communicating Processor (typically S7300 CPU PN/DP).
  - SFB12 "BSEND" in case the SCADA is communicating with an S7 400 PLC.

The source files of the library are:

- **CPC_BASE_Unicos**: Contains declarations of data types and of basic functions.
- **CPC_FB_AA**: Analog Alarm FB.
- **CPC_FB_AI**: Analog Input FB.
- **CPC_FB_AIR**: Analog Input Real FB.
- **CPC_FB_ANADIG**: Analog Digital FB.
- **CPC_FB_ANALOG**: Analog FB.
- **CPC_FB_AO**: Analog Output FB.
- **CPC_FB_AOR**: Analog Output Real FB.
- **CPC_FB_APAR**: Analog Parameter FB.
- **CPC_FB_AS**: Analog Status FB.
- **CPC_FB_D4**: Digital Alarm FB.
- **CPC_FB_DI**: Digital Input FB.
- **CPC_FB_DO**: Digital Output FB.
- **CPC_FB_DPAR**: Digital Parameter FB.
- **CPC_FB_LOCAL**: Local FB.
- **CPC_FB_ONOFF**: On Off FB.
**CPC_FB_PCO**: Process Control Object FB.
**CPC_FB_PID**: Controller FB.
**CPC_FB_WPAR**: Word Parameter FB.
**CPC_FB_WS**: Word Status FB.

---

**Figure 33** - Baseline sources importation

The Blocks of the library are:
- **OB80, OB85, OB86, OB121, OB122**: Error treatment OBs.
- **FB9**: LAG1ST, First order filtering
- **FB12**: BSEND, exchange block of data.
- **FB13**: LMNGEN_C, come from the Modular PID library.
- **FB19**: PID, come from the Modular PID library.
- **FB20**: PULSEGEN, come from the Modular PID library.
- **FB22**: ROC_LIM, come from the Modular PID library.
- **FC1**: LP_SCHED, time scheduler managing the controller sampling times.
- **FC72**: Function for the execution of grafcet or sequencer.
- **VAT_TSPP**: visualization table which allows rapid troubleshooting of TSPP communications.
- **SFB3, SFB4, SFB5**: IEC timers.

---

**Figure 34** - Baseline blocks importation

**3.6.1 IMPORT INSTANCE AND LOGIC FILES**

In the *Simaic Manager Tool*, go to the Sources folder and click on ‘Insert > External Source...’
Look into the corresponding folder ‘(..\Project\Output\S7InstanceGenerator’) for instance generation files and select all of them, then click Open.

Now do the same with the logic generated files.
The source blocks in your project should have this structure now:
3.6.2 IMPORT SYMBOL TABLE
Open the symbol table, see Figure 38.

Figure 38. Open symbols table

Click on ‘Symbol Table > Import…’, see Figure 39.

Figure 39. Importing symbols

Select the symbols file in the project, it is in folder “…\Project\Output\S7InstanceGenerator”, and then click Open.

Figure 40. Importing symbols

Save the symbol table and now you can close the Symbol Editor.
Once these steps are done you could compile your source files and finally download them to the target PLC. This is the basic structure of the final application which must be already operative.
3.7 PLC SOURCES COMPILATION

To build the project we must compile four files:

1. Compilation_Baseline
2. Compilation_instance
3. Compilation_LOGIC
4. Compilation_OB

Open 1 Compilation_Baseline and click on ‘Compile’ button. See Figure 41.

Do the same with the rest of them.

Now you can download the project to the PLC.
3.8 SPECIFIC LOGIC CREATION

3.8.1 CREATION OF A GRAFCET IN S7

You can create a FB in graph language for sequential executions (Figure 48):

1. Create a FB in graph language in your project.
2. Click on ‘File > Change block number…’ to choose the FB number for the grafcet (Figure 42).

![Figure 42 - Grafcet: Change block number](image)

3. Click on ‘Options > Block settings…’ to choose the FC number: 72 (you can find this FC in the baseline). Make sure the options “Individual structures” and “Download to PLC” are chosen. See Figure 43.

![Figure 43 - Grafcet: Block settings](image)

4. In ‘Options > Application Settings…’ make sure the option “Include instance DB” is chosen, see Figure 44. It allows the usage of the variables generated in the instance DB.
5. Create boolean variables for the transitions in the Input of the Grafcet, see Figure 45.

6. Call the transition variables in the grafcet for the transitions, as you can see in Figure 46.

7. Choose meaningful names for the steps.
8. Choose an intuitive symbolic name for the DB, to be referred to the variables as “DBSymbolicName.variable” in the code.

<table>
<thead>
<tr>
<th>Object name</th>
<th>Symbolic name</th>
<th>Created in language</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB15</td>
<td>DemonPCO_SL</td>
<td>DB</td>
</tr>
</tbody>
</table>

Do not forget to include some comments in the different steps.

*Note: All steps are exploited in other functions (e.g. Dependent logic), and all transitions are calculated in the TL function of the PCO.*

In Figure 48 you can see an example of a grafcet.

To follow UNICOS logic:
- Use current *grafcet* active step in other PCO functions to do actions.
- Calculate transition Booleans in the transition logic function (FC NamePCO_TL).
- Modify logic file *PCO_SL* to add the grafcet call.
Example: There is a PCO called DemonPCO with a grafcet in a FB (Demon_Graphcet) and the instance DB of this FB is called DemonPCO_SL. The logic file DEMON_1_DemonPCO_SL should look like this:

```plaintext
(*Stepper Logic: DEMON_1_DemonPCO (The one and only) Application: AppDEMON *****)

FUNCTION DEMON_1_DemonPCO_SL : VOID

TITLE = 'DEMON_1_DemonPCO_SL'

AUTHOR: 'ICE/PLC'

NAME: 'Logic_TL'

FAMILY: 'TL'

BEGIN

Demon_Graphcet.DemonPCO_SL();

END_FUNCTION
```
3.8.2 COMPLETE LOGIC FILES

Each PCO contains 9 main functions (BL, CDOL, CL, DL, GL, IL, INST, SL, TL) and one function by Dependent Logic Object (DL). All these functions are generated by the Logic Generator.

It is necessary to fill some of the functions with the specific logic of the project:

The meanings of the functions filled automatically with the generator are:
- **BL**: Basic Logic, contains the basic logic of the PCO. It is not necessary to fill it.
- **CDOL**: Common Dependant Object Logic, calculate the auto requests of all dependant objects and calculate the alarm acknowledgment. It is not necessary to fill it.
- **INST**: Instantiation, PCO instantiation, contains a call to CPC_FB_PCO and it is executed once per each instance.

It is necessary to fill some of the functions with the specific logic of the project:
- **GL**: Global Logic, allow defining the global logic of the PCO.
- **IL**: Interlock Logic, alarm instantiation and calculation of the Interlocks of the PCO (Start, Temp, Stop Interlock).
- **SL**: Sequencer Logic, sequential behavior of the PCO (generally with Grafcet).
- **TL**: Transition Logic, contains a call to the process grafcet.
- **CL**: Configuration Logic, calculation of the On status and OFF status conditions of the PCO used mainly for animation purposes. In addition it computes the controlled stop finished conditions.
- **DL**: Dependent Logic of every object linked of the PCO. This function contains the behaviour of these objects according to the status (e.g. grafcet states). These objects can be: PCO, Controller, On/Off, Analog, AnaDig, AnaDO. All these objects have a Dependent Logic function associated to their PCO master.

*Note*: PCO can be its own master.

3.9 CREATION OF WINCC OA PROJECT

To create a new WinCC OA project for the supervision, you can check the procedure called “Procedure WinCC OA-CPC Application” [7].

You can design the panels for the supervision in the project and link it to the PLC project.

3.10 USAGE OF LOCAL PANELS

To design an application on a local touch panel, you can check the procedure called “Creating and designing a local panel for a UNICOS application” [5].
4. TROUBLESHOOTING

The time in the PLC and the SCADA application (PVSS) must be synchronized otherwise the frames pushed by the PLC could be discarded by the PVSS driver. The recommendation is to synchronize NTP the PLC whenever is possible. Otherwise the time must be set by hand in the PLC (UTC time) to ensure proper synchronization.

![Set Time of Day window in PLC](image)

Figure 49 - Set time of day window in PLC

The language Mnemonics option must be changed to English if German is chosen, otherwise the baseline won't compile properly. To change this option go to ‘Options > Customize…’ and choose English on the Customize window, see Figure 50 and Figure 51.

![Opening Customize options](image)

Figure 50 - Opening Customize options

![Customize window](image)

Figure 51 - Customize window
5. **SUPPORT**

Please address your questions to IceControls.Support@cern.ch

6. **REFERENCES**

[1] UNICOS CPC6 in Siemens S7 PLC
   

   
   https://edms.cern.ch/file/1228441/1.3.1/Specs_doc.xml

[3] CPC5 to CPC6 Spec Conversion Tool
   
   http://cern.ch/enice/Reverse+Engineering+Tools


[5] Creating and designing a local panel for a UNICOS application
   
   https://edms.cern.ch/file/1228441/1.3.1/Procedure_Touch_Panel_Application.pdf

[6] Logic Template Usage for UNICOS-CPC Applications
   
   https://edms.cern.ch/file/1228441/1.3.1/UCPC6_UserTemplate.pdf

[7] Procedure WinCC OA-CPC Application
   
   https://edms.cern.ch/file/1228441/1.3.1/Procedure_PVSS-UCPC_Application.pdf