Application Example • 02/2017

SIMATIC ODK 1500S Examples V2.0

STEP 7 V14, SIMATIC S7-1500S, SIMATIC ODK 1500S

Warranty and Liability

Note

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

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

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

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

Security information

Siemens provides products and solutions with industrial security functions that support the secure operation of plants, systems, machines and networks. In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens’ products and solutions only form one element of such a concept.

Customer is responsible to prevent unauthorized access to its plants, systems, machines and networks. Systems, machines and components should only be connected to the enterprise network or the internet if and to the extent necessary and with appropriate security measures (e.g. use of firewalls and network segmentation) in place.

Additionally, Siemens’ guidance on appropriate security measures should be taken into account. For more information about industrial security, please visit http://www.siemens.com/industrialsecurity.

Siemens’ products and solutions undergo continuous development to make them more secure. Siemens strongly recommends to apply product updates as soon as available and to always use the latest product versions. Use of product versions that are no longer supported, and failure to apply latest updates may increase customer’s exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed under http://www.siemens.com/industrialsecurity.
# Table of Contents

Warranty and Liability ........................................................................................................... 2

1 Task .................................................................................................................................. 4

2 Solution ................................................................................................................................. 5
   2.1 Overview ......................................................................................................................... 5
   2.2 Hardware and software components ............................................................................. 7
   2.2.1 Validity ....................................................................................................................... 7
   2.2.2 Components used ....................................................................................................... 7

3 Mode of Operation ............................................................................................................... 9
   3.1 General overview ........................................................................................................... 9
   3.2 Example overview ....................................................................................................... 13
   3.3 Variations of the examples ......................................................................................... 14

4 Installation and Commissioning ....................................................................................... 16
   4.1 Installing the hardware ................................................................................................. 16
   4.2 Installing the software ................................................................................................. 17
   4.3 Commissioning ............................................................................................................ 17
   4.3.1 STEP 7 Configuration .............................................................................................. 17
   4.3.2 Enabling the web server ......................................................................................... 20
   4.3.3 Transferring the ODK application ......................................................................... 21
   4.3.4 Completing STEP 7 project ................................................................................... 22

5 Operating the Application ................................................................................................ 24

6 Tips and Tricks .................................................................................................................... 26
   6.1 Changing the example project .................................................................................... 26
   6.2 Creating ODK application ......................................................................................... 27

7 Links & Literature .............................................................................................................. 30

8 History ............................................................................................................................... 30
1 Task

Introduction

The integration of PC functionality in the controller program often offers advantages for the entire automation task. That way, for example, compact automation solutions can be realized on one device. Furthermore, complex functions and solutions have already been created in high-level languages or are created in high-level languages.

For this purpose, for example, high-level language programs have to communicate or have to be integrated with the controller program via defined interfaces. Depending on the application case this may also be required as real-time application. The Open Developer Kit ODK 1500S is used as interface for the PC applications.

Overview of the automation task

The task is to integrate high-level language applications into the SIMATIC PLC.

Simple examples support the connection of PC programs via ODK 1500S to the PLC controller. The examples are implemented with different high-level languages.

Figure 1-1 – Overview of the automation task

Requirements to the automation task:

Table 1-1

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integration of complex algorithms</td>
<td>By integrating complex algorithms, applications can be developed and used in high-level languages.</td>
</tr>
<tr>
<td>Integration of already existing high-level language code</td>
<td>High-level language applications can be used by integrating the already existing high-level language code.</td>
</tr>
</tbody>
</table>
2 Solution

2.1 Overview

Schematic layout

The figure below shows a schematic illustration of the main components of this solution. In this application example the SIMATIC ET 200SP Open Controller is used with installed SIMATIC S7-1500 software. Alternatively, a CPU 1518(F)-4 PN/DP ODK can also be used.

![Schematic layout of the main components](image)

**Note**

S7-1500 Software Controller supports:
- ODK projects for Windows environment
- ODK projects for real-time environment

CPU 1518-4 PN/DP ODK supports:
- ODK projects for real-time environment

**SIMATIC ODK 1500S**

With ODK 1500S the S7-1500 software controller offers the option, to develop applications in high-level languages and to integrate them into the controller. ODK is a development package and is used as an interface to call independent high-level language programs from the controller program of the CPU.

ODK applications are explained by simple examples for two different environments (Windows and real-time environment). ODK applications generated with C/C++ shall be integrated into the control task.

**SIMATIC ET 200SP Open Controller**

The SIMATIC ET 200SP Open Controller, CPU 1515SP PC, is a robust, compact controller system that units the function of an ET 200SP controller with a PC-based platform.

The SIMATIC ET 200SP Open Controller can be expanded with ET 200SP modules. The configuration is done with STEP 7 V14.
2 Solution

SIMATIC S7-1500 Software Controller

The SIMATIC S7-1500 software controller is installed on the ET 200 SP Open Controller and implements the functionality of a SIMATIC S7-1500 controller on a PC. For this, the S7 1500 software controller uses standard PC resources.

Advantages

The solution presented here, offers you the following advantages

- Time and cost saving through fast familiarization with ODK
- Expandability of the STEP 7 project enclosed
- Integration of high-level language applications

Topics not covered by this application

This application example does not contain a description of STEP 7 V14. Basic knowledge of this subject is assumed.

Assumed knowledge

Basic knowledge of C/C++, STEP 7 V14 and PC based is assumed.
2.2 Hardware and software components

2.2.1 Validity

This application example is valid for
- STEP 7 V14 or higher
- S7-1500 Software Controller V2.0

2.2.2 Components used

The application example has been created with the following components:

Hardware components

Table 2-1

<table>
<thead>
<tr>
<th>Component</th>
<th>Qty.</th>
<th>Article number</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMATIC ET 200SP Open Controller</td>
<td>1</td>
<td>6ES7677-2AA41-0FB0</td>
<td>CPU 1515SP PC 4GB (WES7-P, 64-bit)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Alternatively, a SIMATIC IPC that fulfills the system requirements</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>for a CPU 1507S (F) can also be used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For the examples of the real-time environment, any S7-1500 controller</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>that supports ODK can be used.</td>
</tr>
</tbody>
</table>

Software components

Table 2-2

<table>
<thead>
<tr>
<th>Component</th>
<th>Qty.</th>
<th>Article number</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMATIC S7-1500 Software Controller V2.0</td>
<td>1</td>
<td>6ES7 672-5xC01-0Yx0</td>
<td>CPU 1505SP (F) is preinstalled on the SIMATIC ET 200SP Open Controller.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Alternatively, the CPU 1507S (F) can also be used on the SIMATIC IPCs</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>with the article numbers 6ES7 672-7xC01-0Yx0.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For the hardware variant (CPU 1518 ODK) this part is missing, since</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the firmware already exists on the hardware component and does not</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>have to be ordered separately.</td>
</tr>
<tr>
<td>SIMATIC ODK 1500S V2.0</td>
<td>1</td>
<td>6ES7 806-2CD02-0YA0</td>
<td>-</td>
</tr>
<tr>
<td>STEP 7 Professional V14</td>
<td>1</td>
<td>6ES7822-1..04+..</td>
<td>-</td>
</tr>
<tr>
<td>Microsoft Visual Studio</td>
<td>1</td>
<td>-</td>
<td>(Optional) Development of ODK applications for</td>
</tr>
</tbody>
</table>
Example files and projects

The following list includes all files and projects that are used in this example.

Table 2-3

<table>
<thead>
<tr>
<th>Component</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>106192387_ODK_1500S_Samples_DOCU_v20_en.pdf</td>
<td>This document.</td>
</tr>
<tr>
<td>106192387_ODK_1500S_Samples_PROJ_v20.zip</td>
<td>This zip file contains the STEP 7 project.</td>
</tr>
<tr>
<td>106192387_ODK_1500S_Samples_CODE_v20.zip</td>
<td>This zip file contains the ODK examples.</td>
</tr>
</tbody>
</table>
3 Mode of Operation

3.1 General overview

Here, you will find an explanation of the general mode of operation of a PLC program that uses high-level language applications via the ODK interface. SIMATIC ODK automatically generates function blocks for loading and unloading the ODK applications and for calling its functions. These blocks can be accepted in your S7 program.

ODK applications are loaded in the user program. After loading, the functions of the ODK applications can be called. ODK applications are unloaded in order to terminate them.

The following figure shows the sequence of a user program with an ODK application.

Figure 3-1
RUN environments

ODK applications can be created for Windows and for real-time environment.

Figure 3-2

Windows environment

For Windows environment the ODK application is executed in the Windows environment. In this variant Windows functions are available in the ODK application.

The ODK application is created as DLL file (dynamically linked library). A template for programming in Microsoft Visual Studio is included with ODK and supports you in the creation of an ODK application.

Note

ODK applications are carried out asynchronously to the CPU cycle in the Windows environment.

Real-time environment

For real-time environment the ODK application is executed in the real-time environment. In this variant Windows functions cannot be used in the ODK application.

The ODK application is created as SO file (shared object). A template and an additional class library for programming in Eclipse are included with ODK and support you in the creation of an ODK application.

Note

ODK applications are carried out synchronously to the CPU cycle in real-time environment.
Template

One template per environment is included by default with ODK. The template includes or creates functions that are required for the ODK application.

An automatically generated SCL file includes function blocks for the connection of the ODK application in the SIMATIC S7 program.

The template additionally includes helper classes that can be used, for example, for converting the data between SIMATIC S7 and high-level language application.

Figure 3-3
Data can be transferred directly to the user-specific function as IN, OUT, INOUT.

Figure 3-4

```c
/*
 * HelloWorld_win32() shows a dialog window with two strings, where one
 * is editable and will be returned after the call
 * @param[in]strupCaption String, which will be the name of the dialog window
 * @param[in,out]strMessage String, which will show in the text edit
 * field of the dialog and will be returned to the PLC
 * @return returns ODK_SUCCESS when the function finishes
 */
ODK_RESULT HelloWorld_win32 (const ODK_S7STRING strCaption[256],
                           ODK_S7STRING strMessage[256])
{

    // Convert ODK_S7STRING myString to char array
    Convert_S7STRING_to_SZSTR (strCaption, g_strCaption, 256);

    // Convert ODK_S7STRING myMessage to char array
    Convert_S7STRING_to_SZSTR (strMessage, g_strMessage, 256);

    // Create dialog
    LRESULT res = DialogBox(g_hInst, MAKEINTRESOURCE(IDD_DIALOGL), NULL, &DialogProc);

    // Convert char array to ODK_S7STRING
    Convert_SZSTR_to_S7STRING (g_strMessage, strMessage);

    return ODK_SUCCESS;
}
```

**Difference to WinAC ODK**

WinAC ODK has three types of interfaces: SMX, CCX and CMI. The CCX interface is comparable with the ODK 1500S. The management of the user-specific functions is taken on by the function blocks, i.e. for each function of the ODK application, a function block is generated.

The template enclosed for the ODK 1500S supports clearly more with the creation of the SCL file. Commissioning with SCL file prevents one or the other stumbling block.
3.2 Example overview

There are three examples in several variations.
1. “Hello World” (Windows environment)
2. “SimpleSort” (Windows or real-time environment)
3. “Trace” (real-time environment)

“Hello World”

This example shows the ODK application for Windows environment.
In this example, a Windows dialog is shown with an input field when calling the function. The text in the input field and the title of the dialog is transferred when calling it from the data block. The data is converted and used by SIMATIC S7 (S7) in accordance to the high-level language application (SZ). The text is converted to SIMATIC S7 when closing the dialog of the high-level language application and saved in the data block of the CPU.

Figure 3-5
3 Mode of Operation

“SimpleSort”

This example shows the ODK application for Windows environment twice and once for real-time environment.

In this example the values are sorted in an integer array with three values. One function sorts in ascending, another function in descending order.

In addition, values in the array are added up to a global tag with each call. When loading the ODK application, the total is set to 0.

- The “SortIntAscending” function sorts in ascending order.
- The “SortIntDescending” function sorts in descending order.

Note

In the variant for Windows environment with C# function a window is additionally shown that outputs the three values as sorted list. Once the window is closed, the values are saved in the CPU.

“Trace”

This example shows the ODK application in real-time environment.

In this example the trace functions are shown. Calling the “GetTrace” function block displays the entries of the trace buffer in the watch table.

- The “ReturnUserStatus” function creates a trace entry.
- The “CauseException” function creates a trace entry in the event of an error.

3.3 Variations of the examples

Windows environment:

- HelloWorld (Win32):
  in the example standard Win32 functions are used in the ODK application.
- HelloWorld (MFC):
  in the example, the Microsoft Foundation Class Library (MFC) is used in the ODK application.
- HelloWorld (Managed C++):
  in the example Managed C++ is used in the ODK application.
- HelloWorld (C#):
  in the example C# functions are used in the ODK application.

Note

The ODK application uses the external C# application. The C# application is in a separate DLL file (here HelloWorld_CSharp_CSharpLib.dll).

- SimpleSort (Win32):
  in this example, the sort function is solved in C++ with standard methods.
- SimpleSort (C#):
  in this example, the sort function is realized in C#. The ODK application uses the C# functions. The ODK application is developed in C++. An external C# application is used in the ODK application.
3 Mode of Operation

**Note**
The ODK application uses the external C# application. The C# application is in a separate DLL file (here SimpleSort_CSharp_CSharpLib.dll).

Real-time environment:
- **SimpleSort (Realtime):** in the example, functions for the real-time environment are used. Here the Eclipse open source software is used. ODK applications for real-time environment can be created with eclipse.
- **Trace (Realtime):** in the example, trace functions are used in the ODK application. In this case Eclipse is used.

**ODK projects for the examples**
The ODK projects included are stored in the structure.
**Figure 3-6**
4 Installation and Commissioning

This chapter describes the steps necessary to commission the example.

4.1 Installing the hardware

SIMATIC ET 200SP Open Controller

The SIMATIC ET 200SP Open Controller is delivered ready to use. In this application example the SIMATIC ET 200SP Open Controller is used with installed SIMATIC S7-1500 software. Alternatively, a CPU 1518(F)-4 PN/DP ODK can also be used.

Further information on SIMATIC ET 200SP Open Controller can be found in entry: https://support.industry.siemens.com/cs/ww/en/view/108741996

Note

Alternatively, a SIMATIC IPC that fulfills the system requirements for a CPU 1507S (F) can also be used.

First commissioning

Note

Only use standard input and output devices for the initial commissioning since additional drivers are only available after the first installation of the operating system.

When you switch on the SIMATIC ET 200SP Open Controller for the first time, the preinstalled operating system is set up first.

Table 4-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Make the basic settings.</td>
<td>-</td>
</tr>
<tr>
<td>2.</td>
<td>Create a user name and password.</td>
<td>-</td>
</tr>
<tr>
<td>3.</td>
<td>Enter the following IP address: &quot;192.168.1.1&quot;</td>
<td>This IP address is used in the STEP 7 project included. You can also freely select the IP address. Also change the IP address accordingly in the STEP 7 project.</td>
</tr>
<tr>
<td>4.</td>
<td>Wait until the SIMATIC ET 200SP Open Controller is ready.</td>
<td>The initial setup of the operating system is automatically completed.</td>
</tr>
</tbody>
</table>
4.2 Installing the software

Installing STEP 7
Install the current version of STEP 7 (as of STEP 7 V14) on your engineering PC. Follow the instructions before you carry out the installation. Further information on STEP 7 can be found in §6.

Installing the SIMATIC ODK 1500S
Install the SIMATIC ODK 1500S on your engineering PC. Follow the instructions before you carry out the installation. Further information on SIMATIC ODK 1500S can be found in §3.

Installing Eclipse
Eclipse is installed with the SIMATIC ODK 1500S.

Installing Microsoft Visual Studio
Optionally install Microsoft Visual Studio on your engineering PC. Follow the instructions during the installation. Further information on Microsoft Visual Studio can be found in §5.

4.3 Commissioning

4.3.1 STEP 7 Configuration

Initial configuration
The SIMATIC ET 200SP Open Controller has to be loaded via the X2 interface at the initial configuration. The background is that no hardware configuration is yet loaded on the device and the PROFINET interfaces cannot be used.

Loading the configuration
The configuration for the SIMATIC ET 200SP Open Controller is created with STEP 7. For each example listed in chapter 3.3 a configuration with the SIMATIC ET 200SP Open Controller is available in the STEP 7 project enclosed.

Table 4-2

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Start the STEP 7 (TIA Portal) and open the project in the zipped file.</td>
<td>&quot;106192387_ODK_1500S_Samples_PROJ_v20.zip&quot;</td>
</tr>
<tr>
<td>2.</td>
<td>Connect your engineering PC with the SIMATIC ET 200SP Open Controller.</td>
<td>&quot;directly to X2&quot; Ethernet interface</td>
</tr>
<tr>
<td>No.</td>
<td>Action</td>
<td>Remark</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>--------</td>
</tr>
</tbody>
</table>
| 3.  | • Select the CPU 1515SP PC of the respective example (here “HelloWorld_Win32”) in the project tree.  
• click the “Download to device” button in the menu bar. | The “loading” of the configuration into the CPU is started |
| 4.  | Confirm the download dialogs including the “Check before loading” dialog. | |
### 4 Installation and Commissioning

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.</td>
<td>When a new configuration is loaded, the PC systems needs to be restarted. In this case the following dialog “status and actions after downloading to device” is displayed. Before confirming the download dialog, wait until the PC system is restarted.</td>
<td>The PC system may be restarted automatically. This may take a little while.</td>
</tr>
</tbody>
</table>
| 6.  | - Continue the “loading” of the configuration when the PC system is ready.  
    | - Confirm the download dialogs.                                      | The “loading” of the configuration into the CPU is continued.                                                                   |
| 7.  | The CPU is in RUN.                                                   | You may have to set the operating mode switch to RUN.                                                                               |

**Note** If loading does not work as described, reload the configuration into the CPU.
4.3.2 Enabling the web server

**Note**

In the STEP 7 project included, the web server is enabled in every CPU configuration and the “admin” user is configured with the “admin” password.

If you are using the examples included, skip the steps in the following Table 4-3.

The SIMATIC ET 200SP Open Controller has an integrated web server. The web server enables access to the CPU.

The following table shows you how to enable the web server for the SIMATIC ET 200SP Open Controller.

### Table 4-3

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
</table>
| 1.  | • Select the CPU 1505SP.  
     • Open the “Properties” tab.  
     • Select the “Web server” property.  
     • Set the “Activate web server on this module” option.  
     • Please confirm the security note.  
     Note: the security note is only displayed with the initial activation of the web server.  
     | Set up the “Web server” option in the STEP 7 project.  
     | ![Image of web server setup](image1.png) |
| 2.  | • Create a user (here “admin” user, “admin” password).  
     • Specify the execution rights for the user.  
     | The execution rights have to be assigned for the user in the user management.  
     | ![Image of user management](image2.png) |
| 3.  | Load the configuration into the CPU as described in Table 4-2.  
     | After the loading the web server is activated.  
     | ![Image of configuration load](image3.png) |
4.3.3 Transferring the ODK application

Windows environment

Transfer the DLL file to the SIMATIC ET 200SP Open Controller. The DLL file is located in the “ODK_1500S_Samples_Code” in the “Release” directory of the respective example.

Table 4-4

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Copy the DLL file into the following subfolder of the target system: C:\ProgramData\Siemens\Automation\ODK1500S</td>
<td>for example, with a USB stick</td>
</tr>
</tbody>
</table>

Note
In the example with the external C# application it also has to be transferred to the target system (here HelloWorld_CSharp_CSharpLib.dll).

Real-time environment

Transfer the SO file with the help of the web server to the target system. The DLL file is located in the included “ODK_1500S_Samples_Code” in the “release_so” directory of the respective example (here “SimpleSort.so”).

Table 4-5

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>• Start your preferred web browser. In this application example the “Internet Explorer Version 11” web browser is used. • Connect with the integrated web server.</td>
<td>Use, for example, the PROFINET interface of the SIMATIC ET 200SP Open Controllers (X1). In the enclosed STEP 7 project the address is: <a href="http://192.168.2.10">http://192.168.2.10</a>. <strong>Note:</strong> If you are using a different web browser and the default web page of the web browser of your S7-1500 CPU cannot be opened, you can find more information this subject in the FAQ: <a href="https://support.industry.siemens.com/cs/ww/en/view/103528224">https://support.industry.siemens.com/cs/ww/en/view/103528224</a></td>
</tr>
<tr>
<td>2.</td>
<td>Log on as user. (Here: “admin” user, “admin” password).</td>
<td>If the function (file browser) is not available, check the execution rights of the user.</td>
</tr>
</tbody>
</table>
4. Navigate to the “ODK 1500S” directory as storage location for your SO file.

In this directory the SO files have to be stored.

5. Select your SO file (here “SimpleSort.so”).
   • Download the file into the load memory of the CPU.

The file is transferred to the SIMATIC ET 200SP Open Controller.

4.3.4 Completing STEP 7 project

Transferred ODK applications have to be additionally integrated in the S7 program. Complete your STEP 7 project with the help of the SCL file. The SCL file is located in the “ODK_1500S_Samples_Code” enclosed in the “STEP7” directory of the respective example.

Note

In the STEP 7 project enclosed, the respective ODK application is already integrated for each example.

If you are using the examples included, skip the steps in the following Table 4-6.
The following table shows you how you can integrate your OKD application in the STEP 7 project with the generated SCL file.

Table 4-6

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Add new external sources to your project (&quot;Add new external file&quot;).</td>
<td><img src="image1.png" alt="Image showing the SCL file being added" /></td>
</tr>
<tr>
<td>2.</td>
<td>Select your SCL file.</td>
<td><img src="image2.png" alt="Image showing the SCL file being selected" /></td>
</tr>
</tbody>
</table>
| 3.  | - Generate the blocks from the SCL file.  
   - Call the context menu for the SCL file (here "HelloWorld_Win32.scl").  
   - Select “Generate blocks from source”. | ![Image showing the blocks being generated](image3.png) |
| 4.  | Complete your program. | ![Image showing the blocks and their functions](image4.png) |
| 5.  | Load the configuration into the CPU as described in Table 4-2. | ![Image showing the configuration loading process](image5.png) |
Operating the Application

Operating the ODK application

In this application example you can control your ODK application with the respective watch table. The approach for the “HelloWorld_Win32” example is described in detail here.

Table 5-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Open the watch table of the respective example in the STEP 7 (TIA Portal) (here “HelloWorld_Win32_WT”).</td>
</tr>
<tr>
<td>2.</td>
<td>Click on the “Monitor all” button.</td>
</tr>
<tr>
<td>3.</td>
<td>Set “BoolReqHelloWorld_Win32_Load” to “1” or “TRUE”.</td>
</tr>
</tbody>
</table>

Your ODK application is loaded. Loading is successful when the status is “0”. The loading of the ODK applications for Windows environment may take a little while.
### Operating the Application

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td>Call your functions (here “BoolReqHello…..HelloWorld_Win32” to “1” or to “TRUE”).</td>
<td>The Windows window is displayed on the SIMATIC ET 200SP Open Controller.</td>
</tr>
<tr>
<td>5.</td>
<td>Enter a text in the input field of the Windows window and click “OK”.</td>
<td>The Windows window is closed and the entered text is saved in the data block.</td>
</tr>
<tr>
<td>6.</td>
<td>Set “BoolReqHelloWorld_Win32_Unload” to “1” or “TRUE”.</td>
<td>Your ODK application is unloaded.</td>
</tr>
</tbody>
</table>
6 Tips and Tricks

6.1 Changing the example project

General

If you want to change the example project you have to transfer the ODK application again to the target system, as described in chapter 4.3.3. You have to update the blocks in your STEP 7 project. You can use the SCL file for this.

Using the SCL file

When using the template, the SCL file is automatically generated with “Build project” with the ODK application. Use the SCL file as described in chapter 4.3.4 in order to integrate your ODK application.

Adding new functions

Functions are defined in the “<Project>. odk” file (here “HelloWorld_Win32.odk”). The template creates or automatically updates the files “ODK_Functions.h” and the “<Project>.scl” file with all required instructions. This is executed with “Build project”.

Figure 6-1

Implement your function in the “<Project>.cpp” file (here “HelloWorld_Win32.cpp”).
6.2 Creating ODK application

Note
When creating a new ODK application a “SampleFunction” example function is created. You can change this function or replace it with your one.

Microsoft Visual Studio
Use the Visual Studio template included, in order to create a new ODK application for Windows environment.

Table 6-1

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Start Visual Studio and create a new project.</td>
<td></td>
</tr>
</tbody>
</table>
| 2.  | • Select the ODK template for your project.  
     • Enter the project name.  
     • Adjust the storage location for your project. | A new project for the ODK application is created. It includes required methods and an example function. |
| 3.  | Expand or add your methods as described in chapter 6.1.                |                                                                        |
| 4.  | After the successful “Build project” your ODK application is ready.    | Normally a DLL file and a SCL file is created.                        |
| 5.  | Transfer the DLL file to the target system, as described in chapter 4.3.3. |                                                                        |
| 6.  | Use the SCL file as described in chapter 4.3.4 in order to create function blocks in the STEP 7 project. |                                                                        |
Eclipse

Use the included template for Eclipse in order to create a new ODK application for real-time environment.

Note: Java Runtime environment may be required on the engineering PC.

Table 6-2

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Start Eclipse and create a new project.</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>• Select the ODK template for your project.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Enter the project name.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Adjust the storage location for your project.</td>
<td></td>
</tr>
</tbody>
</table>

A new project for the ODK application is created. It includes required methods and an example function.
### 6 Tips and Tricks

<table>
<thead>
<tr>
<th>No.</th>
<th>Action</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.</td>
<td>Expand or add your methods as described in chapter 6.1.</td>
<td><img src="image1.png" alt="Image" /></td>
</tr>
<tr>
<td>4.</td>
<td>After the successful “Build project” your ODK application is ready. A SO file and a SCL file are created.</td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
<tr>
<td>5.</td>
<td>Transfer the SO file to the target system, as described in chapter 4.3.3.</td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
<tr>
<td>6.</td>
<td>Use the SCL file as described in chapter 4.3.4 in order to create function blocks in the STEP 7 project.</td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>
7 Links & Literature

Table 7-1

<table>
<thead>
<tr>
<th>Topic</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1\ Siemen Industry Online Support</td>
<td><a href="https://support.industry.siemens.com">https://support.industry.siemens.com</a></td>
</tr>
<tr>
<td>2\ Download page of the entry</td>
<td><a href="https://support.industry.siemens.com/cs/ww/en/view/106192387">https://support.industry.siemens.com/cs/ww/en/view/106192387</a></td>
</tr>
<tr>
<td>5\ Microsoft Visual Studio</td>
<td><a href="http://www.microsoft.com">http://www.microsoft.com</a></td>
</tr>
<tr>
<td>6\ STEP 7 Professional V14</td>
<td><a href="https://support.industry.siemens.com/cs/ww/en/view/109742272">https://support.industry.siemens.com/cs/ww/en/view/109742272</a></td>
</tr>
<tr>
<td>7\ Web browser of the S7-1500 CPU</td>
<td><a href="https://support.industry.siemens.com/cs/ww/en/view/103528224">https://support.industry.siemens.com/cs/ww/en/view/103528224</a></td>
</tr>
</tbody>
</table>

8 History

Table 8-1

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Modifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>V2.0</td>
<td>02/2017</td>
<td>First version for ODK 1500S V2.0</td>
</tr>
</tbody>
</table>