

SIEMENS



Application Example • 09/2016

Exchange of large data volumes between S7-1500 control system and WinCC

S7-1500, WinCC V7.4



<https://support.industry.siemens.com/cs/ww/de/view/37873547>

Warranty and Liability

Note

The Application Examples are not binding and do not claim to be complete with regard to configuration, equipment or any contingencies. The Application Examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for the correct operation of the described products. These Application Examples do not absolve you from responsibility for safe and professional use, installation, operation and servicing. 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 and 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 shall 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 fundamental contractual obligations (“wesentliche Vertragspflichten”). The compensation for damages due to a breach of a fundamental contractual obligation is, 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 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 secure plants, systems, machines and networks against cyber threats it is necessary to implement (and to continuously maintain) a holistic, state-of-the-art Industrial Security concept. With this in mind, Siemens' products and solutions are only one part of such a concept.

It is the client's responsibility to prevent unauthorized access to his plants, systems, machines, and networks. Systems, machines, and components should only be connected to the company's network or the Internet if and to the extent to which this is required and the appropriate protective actions (for example, use of firewalls and network segmentation) have been taken.

In addition, Siemens' recommendations regarding appropriate protective action should be followed. For more information on Industrial Security, please visit <http://www.siemens.com/industrialsecurity>.

Siemens' products and solutions undergo continuous development to make them even more secure. Siemens explicitly recommends updating the software as soon as new updates are available and always using current product versions only. Continuing to use product versions which became obsolete or are no longer supported may increase customer's exposure to cyber threats.

In order to stay informed about product updates, please subscribe to the Siemens Industrial Security RSS Feed at <http://www.siemens.com/industrialsecurity>.

Table of Contents

	Warranty and Liability	2
1	Task.....	4
	1.1 Overview.....	4
2	Solution.....	5
	2.1 Configuration	5
	2.2 Hardware and software components	6
	2.2.1 Applicability	6
	2.2.2 Components used	6
3	Basic information	7
	3.1 Building communication	7
	3.2 Difference between the functions GetTagRawWait and GetTagRaw	8
	3.3 Quality Code.....	9
4	PLC Configuration and Design	10
	4.1 Creating and configuring the CPU	10
	4.2 Creating the data block	11
	4.3 Setting the clock time	13
5	HMI Configuration and Design	14
	5.1 Preparing the design environment in SIMATIC WinCC	14
	5.2 Establishing a connection to the S7-1500 control system	15
	5.3 Setting the PG/PC interface	18
	5.4 Creating C scripts.....	19
6	Converting raw data to other data types.....	23
	6.1 Example from Byte to Word	23
7	Using the Application.....	26
	7.1 Commission the example project	26
	7.2 Using the example project.....	26
8	Further Notes, Tips & Tricks, etc.	27
9	Links & References	28
10	History.....	28

1 Task

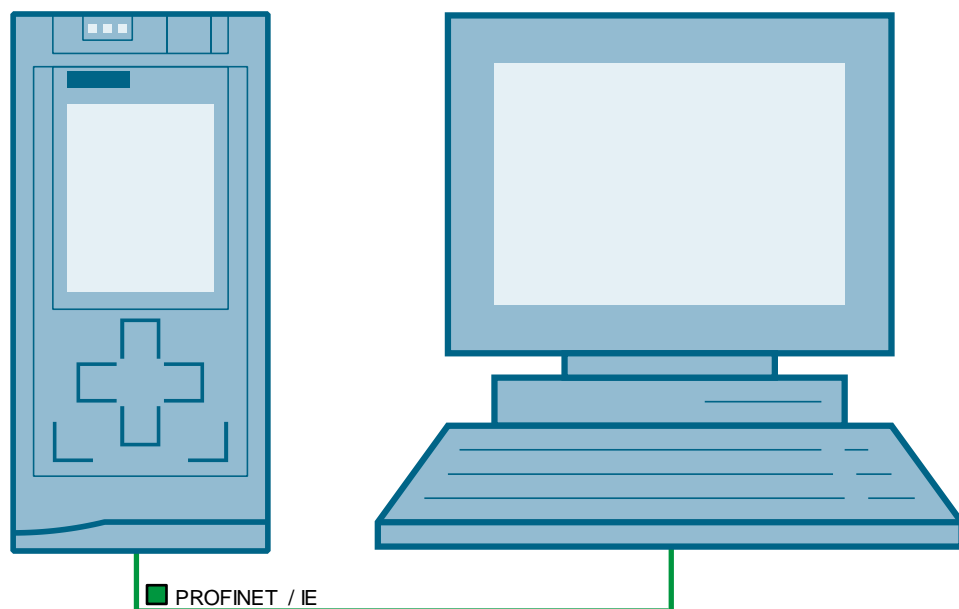
Description of the automation task

The present application example shows you how to transmit large data volumes from an automation control system to WinCC. This functionality will be explained using the example of the S7-1500.

Raw data are used to transmit large data volumes from the control system to WinCC or from WinCC to the control system, respectively. In this type of communication, the data volume is split into single segments which are sent individually to the related partners.

1.1 Overview

Figure 1-1



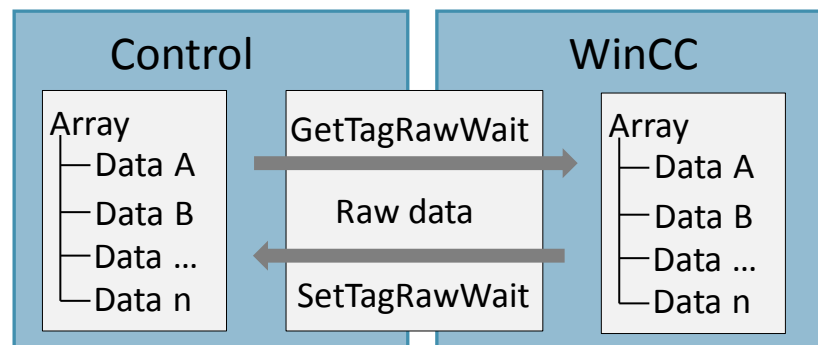
2 Solution

2.1 Configuration

A standard WinCC configuration is supposed. The present application example considers the connection from an automation control system to WinCC.

Diagram

Figure 2-1



A sender and a receiver are necessary for the bilateral data exchange.

Advantage

The advantage is that a Raw Data Variable is licensed as external variable. As a raw data variable contains an array of "n" values, the latter will not be counted apart. Hence, one licensed external variable is enough to transmit 8000 byte values from the control system to WinCC, as an example.

Disadvantage

Extended time and effort are necessary to configure the raw data communication.

Required knowledge

To implement the solution described in the present application example, basic knowledge in the following branches is necessary:

- Automation technology

2.2 Hardware and software components**2.2.1 Applicability**

This application example is valid for

- TIA Portal V13 SP1 Update 7
- WinCC V7.4

2.2.2 Components used

The application example has been created using the following components:

Hardware components

Table 2-1

Component	Qty	Article number	Note
SIMATIC S7-1500	1x	6ES7 516-3AN00-0AB0	Firmware: V1.7 The control system is given as an example; other control systems may be used considering the software requirements.
Industrial PC SIMATIC IPC647D	1	6AG4112-2....-....	The IPC is given as an example; other IPCs may be used considering the software requirements.

Software components

Table 2-2

Component	Qty	Article number	Note
WinCC V7.4	1	6AV63.1-....7-4...	
STEP 7 Professional V13 SP1	1	6ES7822-1..03	

Example files and projects

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

Table 2-3

Component	Note
37873547_Rohdaten_WinCC_V74.zip	This zip file contains the WinCC project.
37873547_Rohdaten_TIA_S7-1500.zip	This zip file contains the PLC project.
37873547_Rohdaten_WinCC_V74_TIA_S7-1500_de.pdf	The present document.

3 Basic information

3.1 Building communication

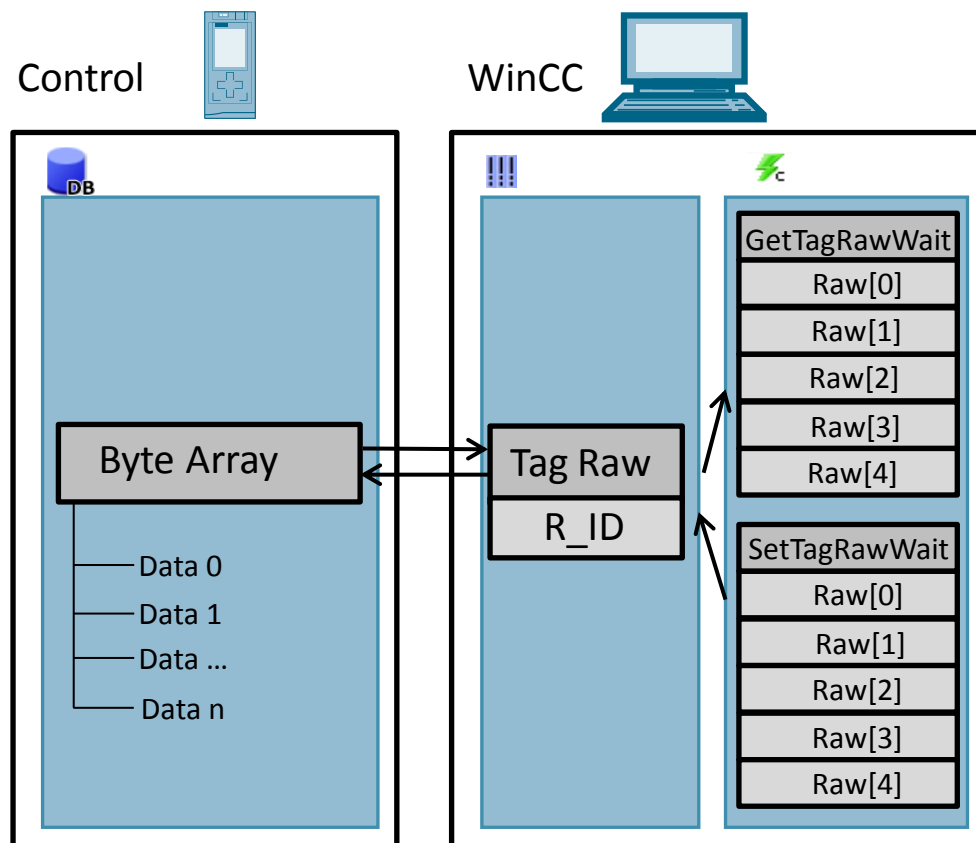
To implement the functionality, a data memory must be created within the control system.

This data memory is used to save data in structured form as an array. That array is used as data basis in the control system. Data from this array are written with the help of WinCC. WinCC in turn is able to interpret these data and read / display the data from this tag.

The procedure in the other direction also works in the same way. The WinCC functionality allows the modification of the raw data tags. These data are now transmitted to the control system and written into the data block array. The data within the array modified by this procedure are then re-transmitted to WinCC on the next operation.

Sending / receiving data from the control to WinCC

Figure 3-1

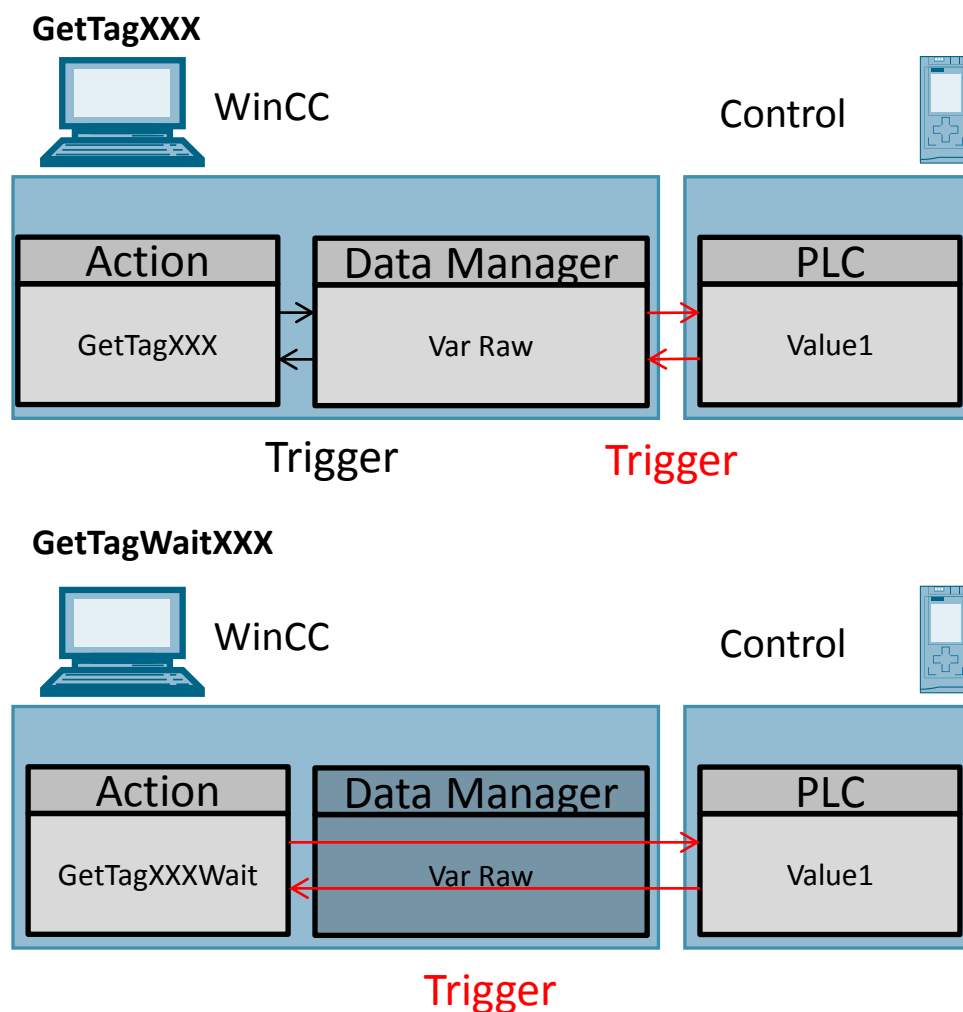


3.2 Difference between the functions GetTagRawWait and GetTagRaw

To read variables from / write variables to the AS, the C scripts provide the TAG object function.

Reading the variables only when they are necessary in the script is an error that we can often discover. As a consequence, the variable is only cyclically registered in the variables map of the data manager with 1 second standard cycle on it's first reading and thus will rise the basic load. The GetTagXXXWait function will remedy this problem. This function is used to bypass the data manager and the variable is not registered in this case.

Figure 3-2



The GetTagWait call is necessary in the following cases:

- Synchronizing quick write/read processes
- Reading explicitly one value out of the automation device
- By-passing deliberately the registering

The GetTagWait call shall be avoided in cyclical C actions.

3.3 Quality Code

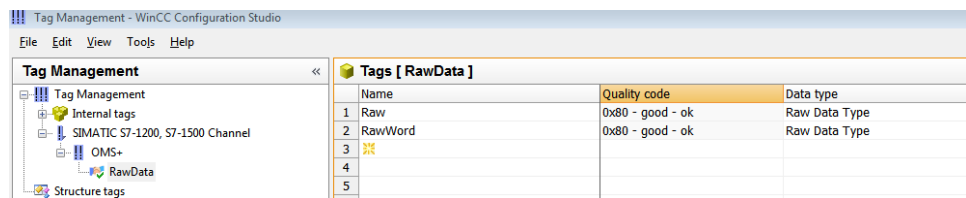
The Quality Code is used to check state and quality of a tag. This information is completed by quality-related information regarding the partners which acquire and process the tags. Potential partners are the following:

- Automation systems
- Automation systems with field devices
- OPC servers
- OPC servers with lower-level automation systems

Access “Tag Management” to display the Quality Code of the tags.

The following requirements must be met:

- The WinCC project is activated
- The “Quality Code” column is visible among others in the “Tag Management” data area.



The Quality Code “good” for example means that this tag may be used. For detailed information on the Quality Code elements, please refer to the table in [WinCC Handbuch Kommunikation](#). The table starts with the worst Quality Code and ends with the best one.

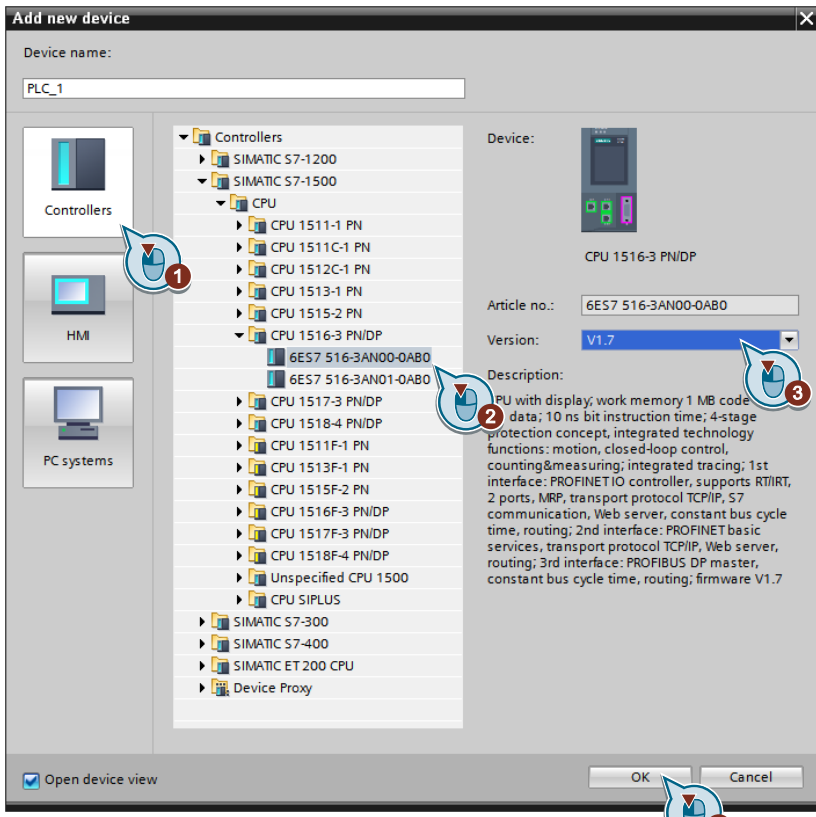
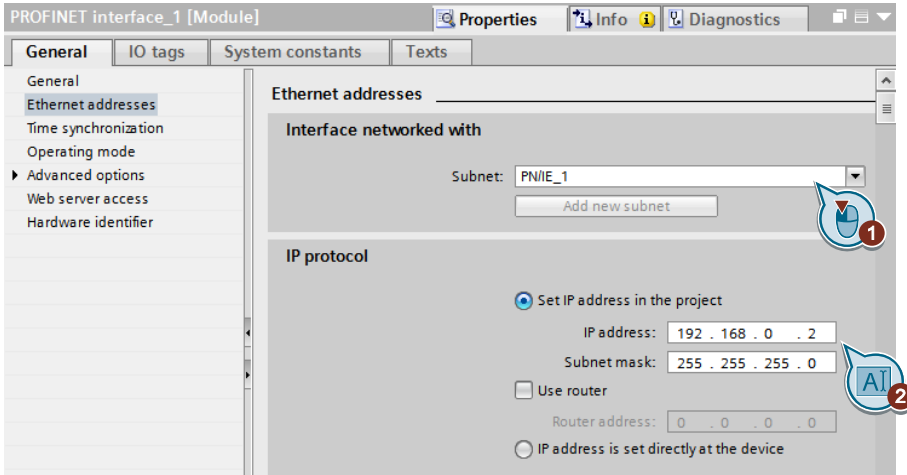
Note

Owing to the asynchronous read operation, the Quality Code is always “bad” when a raw data tag is read for the first time. The status will switch to “good” for any further read operation.

4 PLC Configuration and Design

4.1 Creating and configuring the CPU

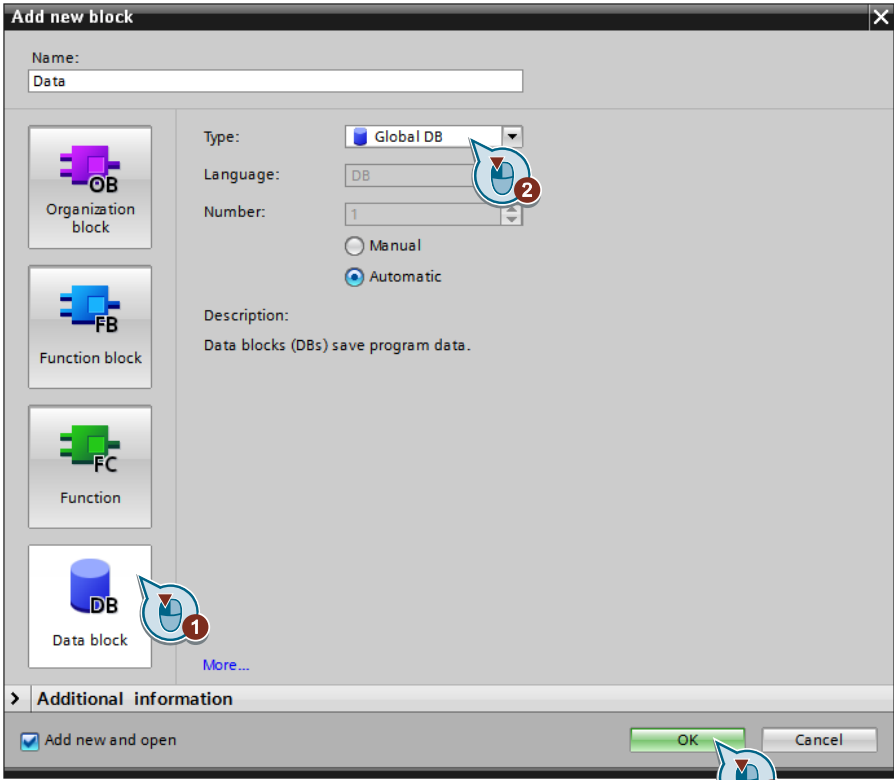
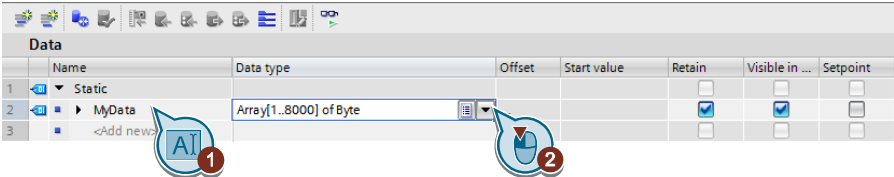
Table 4-1

Item	Action
1.	<p>Insert a new S7-1500 control system into your project.</p> 
2.	<p>Assign an IP address and a subnet to the new control system.</p> 

4.2 Creating the data block

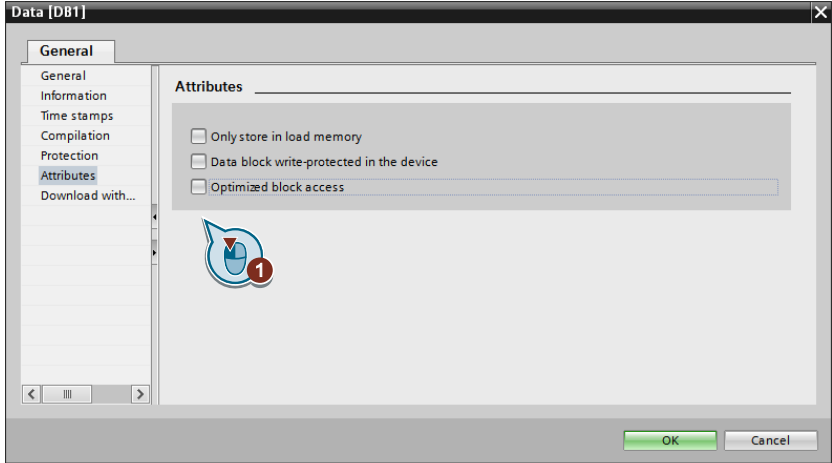
The "Data" data block is used to save the data to the PLC in the form of an array.

Table 4-2

Item	Action
1.	<p>Create a new "Data" data block.</p> 
2.	<p>Create a byte array with 8000 values, as an example. These values will be sent later as raw data to WinCC.</p> 

4 PLC Configuration and Design

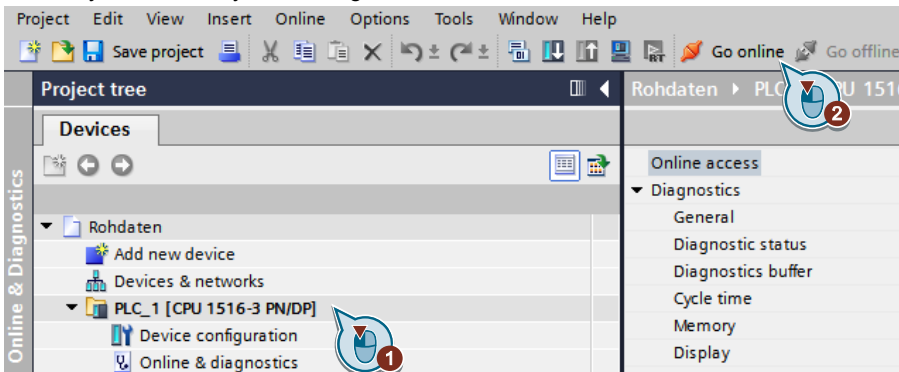
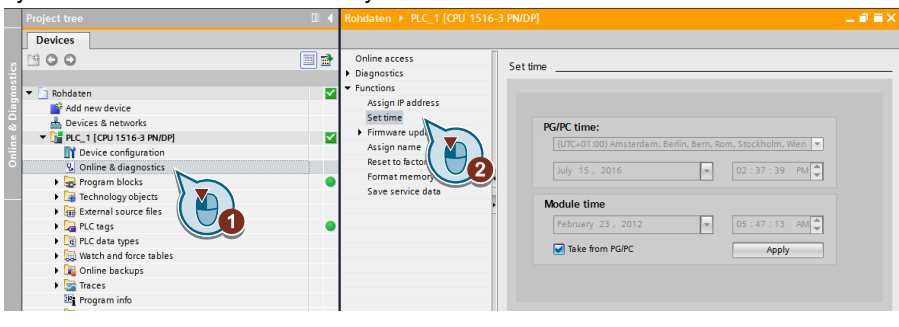
4.2 Creating the data block

Item	Action
3.	<p>Go to "Attributes" of the data block. In "Attributes", select "Optimized block access".</p>  <p>The screenshot shows a software window titled "Data [DB1]". On the left is a navigation pane with tabs: "General", "Information", "Time stamps", "Compilation", "Protection", "Attributes", and "Download with...". The "Attributes" tab is selected. The main area shows three checkboxes: "Only store in load memory", "Data block write-protected in the device", and "Optimized block access". A blue callout bubble with a red "1" is positioned over the "Optimized block access" checkbox. At the bottom right are "OK" and "Cancel" buttons.</p>

4.3 Setting the clock time

Make sure that the CPU and WinCC are set to the same clock time (UTC or local time).

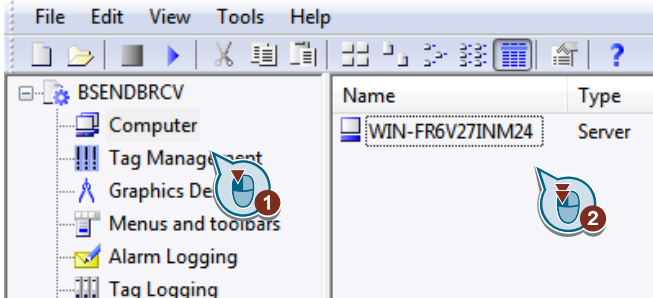
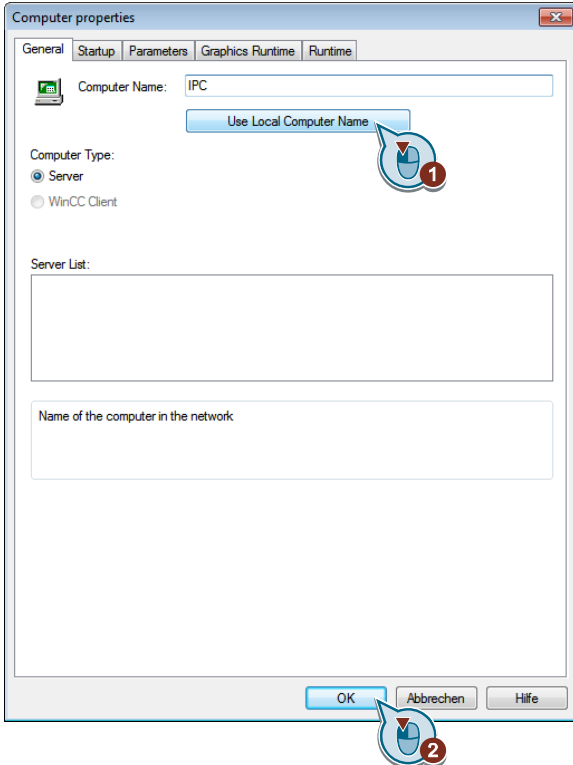
Table 4-3

Item	Action
1.	<p>Access your control system and go “online”.</p> 
2.	<p>In the project navigation of your CPU, select “Online & Diagnostics”. Adjust your system to the same clock time as your WinCC station.</p> 

5 HMI Configuration and Design

5.1 Preparing the design environment in SIMATIC WinCC

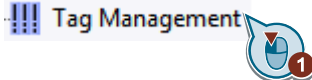
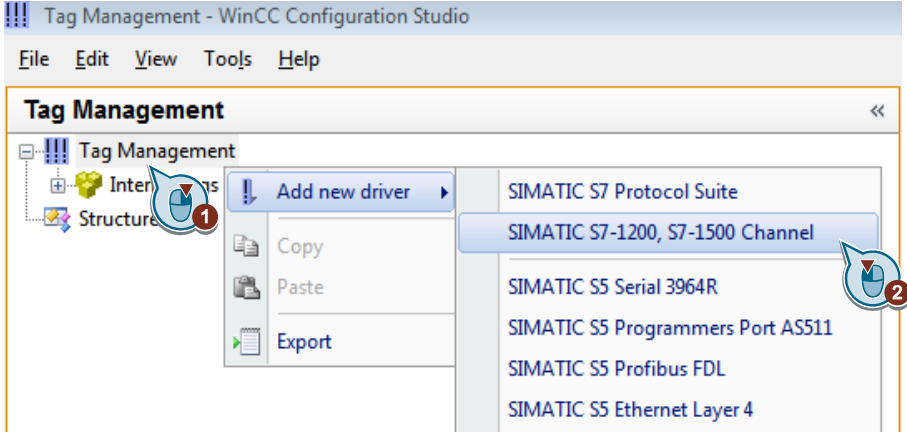
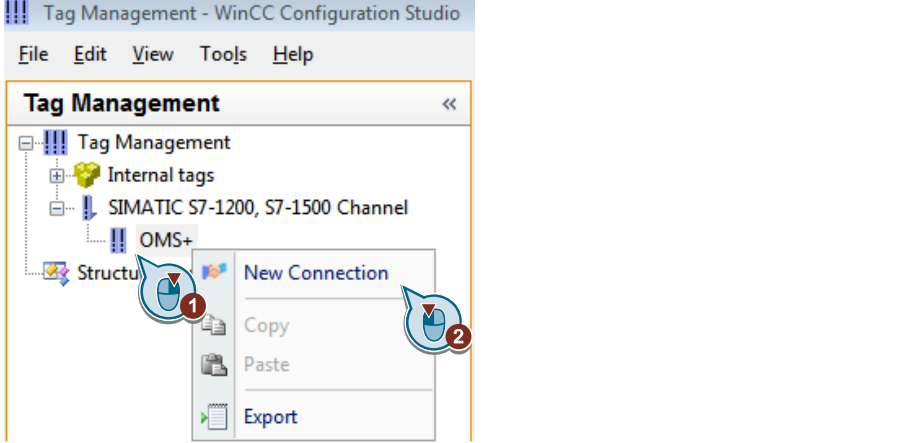
Table 5-1

Item	Action
1.	Start the SIMATIC WinCC explorer.
2.	<p>In the WinCC explorer, click the “Computer” icon. Then click the computer name.</p> 
3.	<p>In “Computer properties → General” click “Use local computer name” and confirm with “OK”.</p> 

5.2 Establishing a connection to the S7-1500 control system

Make sure that the interface conforms to your Programming device / PC interface in Windows.

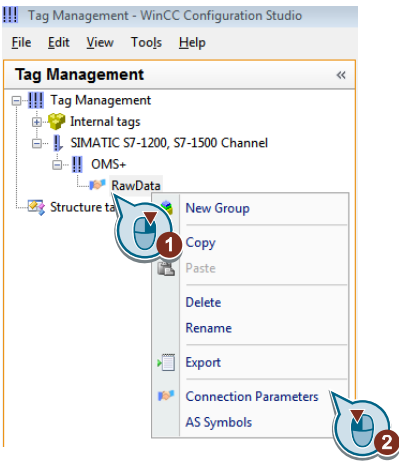
Table 5-2

Item	Action
1.	Open the “Tag Management” in WinCC. 
2.	In the Tag Management, select “Add new driver” → SIMATIC S7 1200, S7-1500 Channel”. 
3.	Create a new “RawData” connection. 

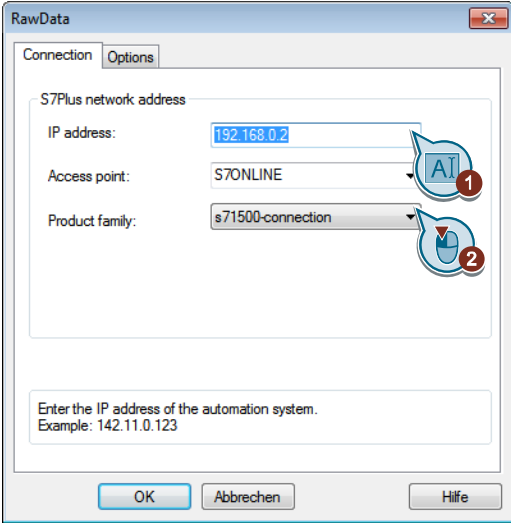
5 HMI Configuration and Design

5.2 Establishing a connection to the S7-1500 control system


4. Open the Connection Parameters of the new connection.



5. Enter the IP address of your S7-1500. Select S7ONLINE as access point. Select an S71500 connection as product family.



6. Create a "Raw" tag of data type „Raw Data Type“ using the new connection. Click "Address".

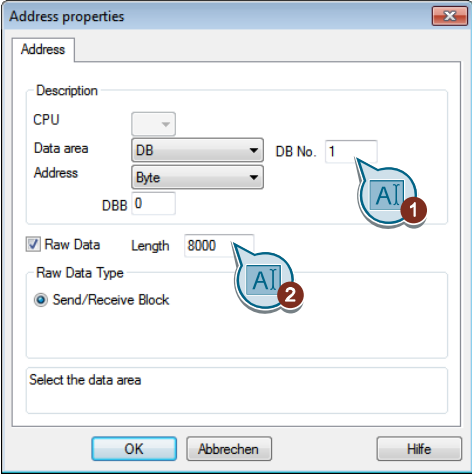


Tags [RawData]					
	Name	Data type	Connection	Address	Linear scaling
1	Raw	Raw Data Type	RawData		
2					

5 HMI Configuration and Design

5.2 Establishing a connection to the S7-1500 control system

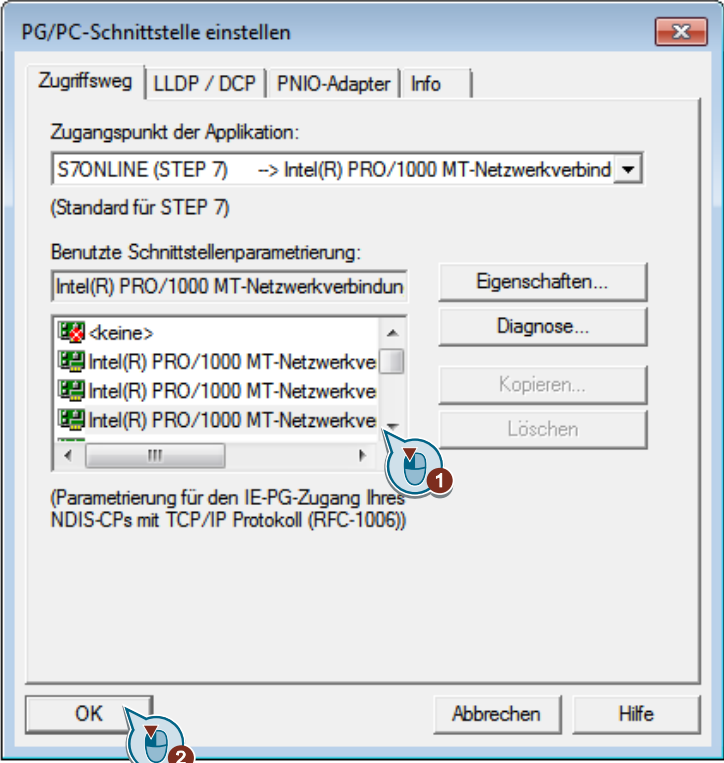
7. Enter the DB block number you have assigned in the PLC. Enter the raw data length.



5.3 Setting the PG/PC interface

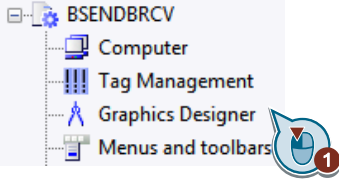
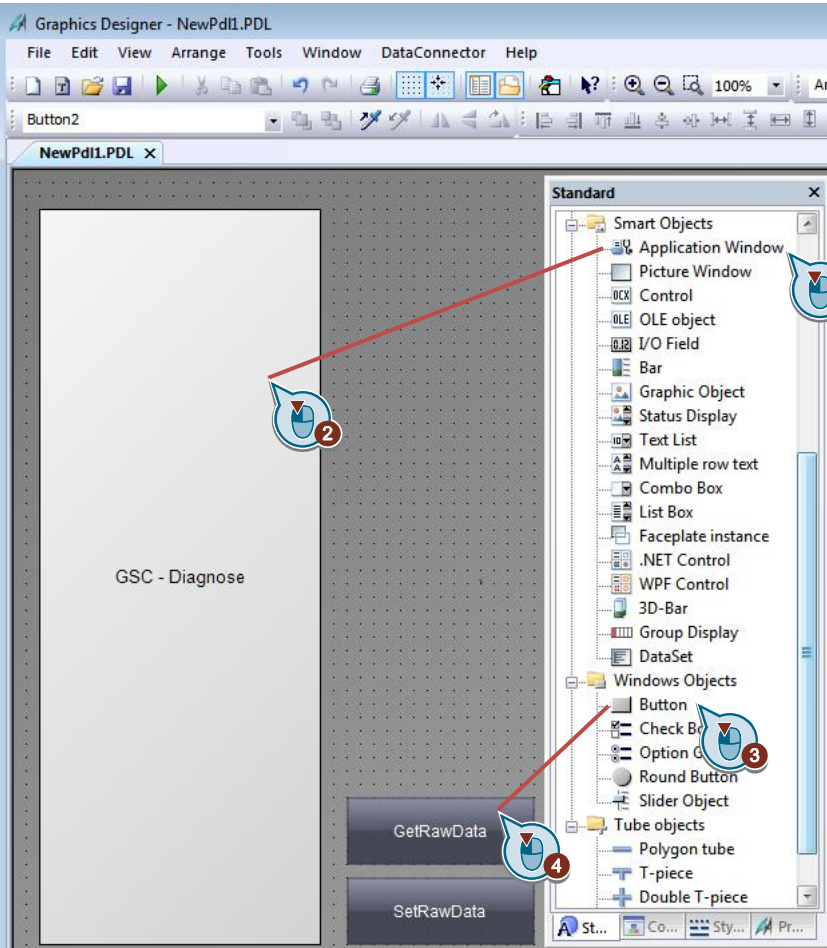
Further information on the PG/PC interface is available in entry ID: [79689088](#).

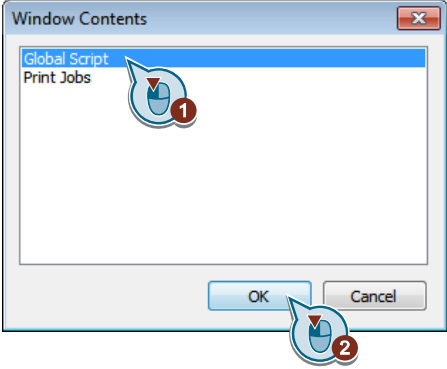
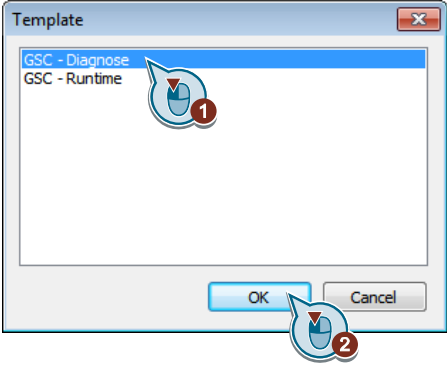
Table 5-3

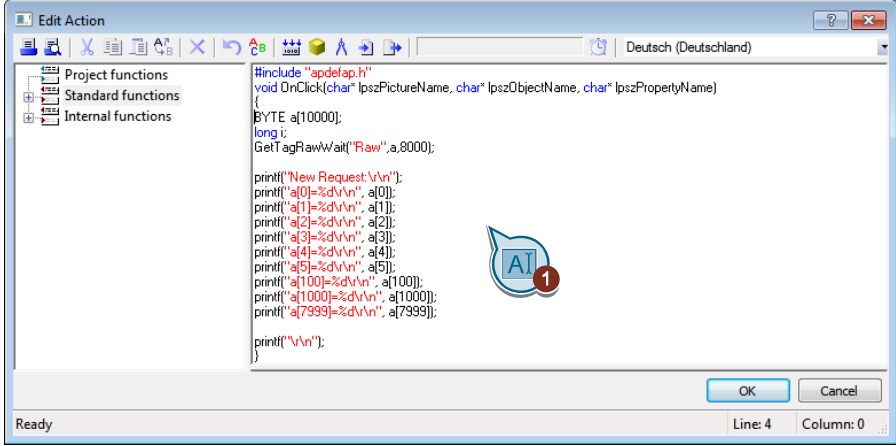
Item	Action
1.	In Windows, select the PG/PC Interface entry in "Start → Control Panel → PG/PC Interface Setting".
2.	<p>In the "PG/PC Interface Setting" window, select the desired interface in the "Use interface parameterizing" option.</p>  <p>Confirm the dialog with OK.</p>

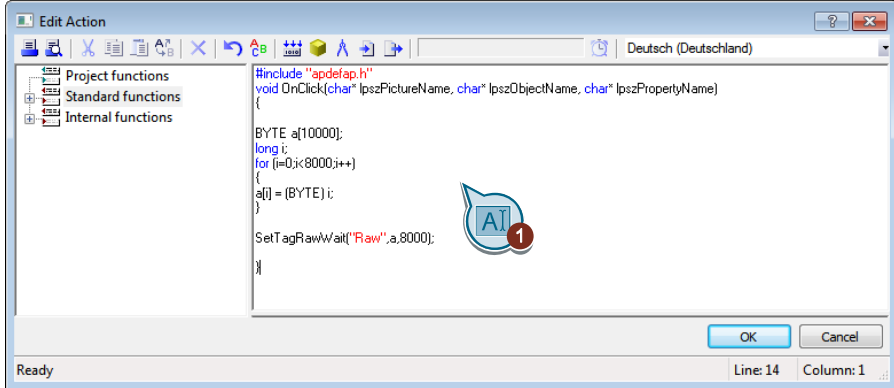
5.4 Creating C scripts

Table 5-4

Item	Action
1.	<p>Open the Graphics Designer.</p> 
2.	<p>Insert the „GetRawData“ and „SetRawData“ buttons into your screen 2. Insert an application window into your screen.</p> 

Item	Action
3.	<p>Once the application window inserted the pop-up window below opens. Select "Global Script" here.</p> 
4.	<p>In the next pop-up window, select "GSC-Diagnose".</p> 

Item	Action
5.	<p data-bbox="486 309 1013 336">Add the script below to the „GetRawData“ button.</p> <div data-bbox="486 340 1385 784" style="border: 1px solid gray; padding: 5px;">  <pre data-bbox="734 403 1204 694"> #include "apdefap.h" void OnClick(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName) { BYTE a[10000]; long i; GetTagRawWait("Raw",a,8000); printf("New Request:\r\n"); printf("a[0]=%d\r\n", a[0]); printf("a[1]=%d\r\n", a[1]); printf("a[2]=%d\r\n", a[2]); printf("a[3]=%d\r\n", a[3]); printf("a[4]=%d\r\n", a[4]); printf("a[5]=%d\r\n", a[5]); printf("a[100]=%d\r\n", a[100]); printf("a[1000]=%d\r\n", a[1000]); printf("a[7999]=%d\r\n", a[7999]); printf("\r\n"); } </pre> </div> <p data-bbox="486 817 1356 907">A byte array is being created within the script. The values from the “Raw” tag (chapter 5.2) are written into the array created by means of the „GetTagRawWait“ order. The single elements of the array are output using the “printf” function.</p> <pre data-bbox="486 940 837 1456"> BYTE a[10000]; long i; GetTagRawWait("Raw",a,8000); printf("New Request:\r\n"); printf("a[0]=%d\r\n", a[0]); printf("a[1]=%d\r\n", a[1]); printf("a[2]=%d\r\n", a[2]); printf("a[3]=%d\r\n", a[3]); printf("a[4]=%d\r\n", a[4]); printf("a[5]=%d\r\n", a[5]); printf("a[100]=%d\r\n", a[100]); printf("a[1000]=%d\r\n", a[1000]); printf("a[7999]=%d\r\n", a[7999]); printf("\r\n"); </pre>

Item	Action
6.	<p data-bbox="485 309 1046 338">Insert the script below into the „SetRawData“ button.</p> <div data-bbox="485 371 1382 757" style="border: 1px solid gray; padding: 5px;">  <pre data-bbox="735 439 1206 645"> #include "apdefap.h" void OnClick(char lpszPictureName, char lpszObjectName, char lpszPropertyName) { BYTE a[10000]; long i; for (i=0;i<8000;i++) { a[i] = (BYTE) i; } SetTagRawWait("Raw",a,8000); } </pre> </div> <p data-bbox="485 797 1353 882">A byte array is being created within the script. A "For" loop is used to fill the array with values. The values from the array created are written into the tag variable "Raw" using the „SetTagRawWait“ order.</p> <pre data-bbox="485 920 831 1178"> BYTE a[10000]; long i; for (i=0;i<8000;i++) { a[i] = (BYTE) i; } SetTagRawWait("Raw",a,8000); </pre>

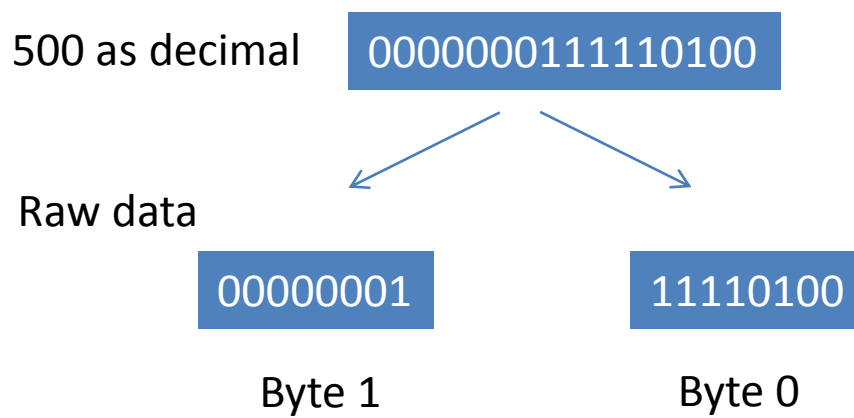
6 Converting raw data to other data types

6.1 Example from Byte to Word

Raw data are available in byte format in WinCC. To integrate other data types, like for example, Word, some single bytes must be pooled to build the data type Word.

Example: The figure 500 is saved as Word

Figure 6-1



You must use a script to execute the separating and merging process.

6 Converting raw data to other data types

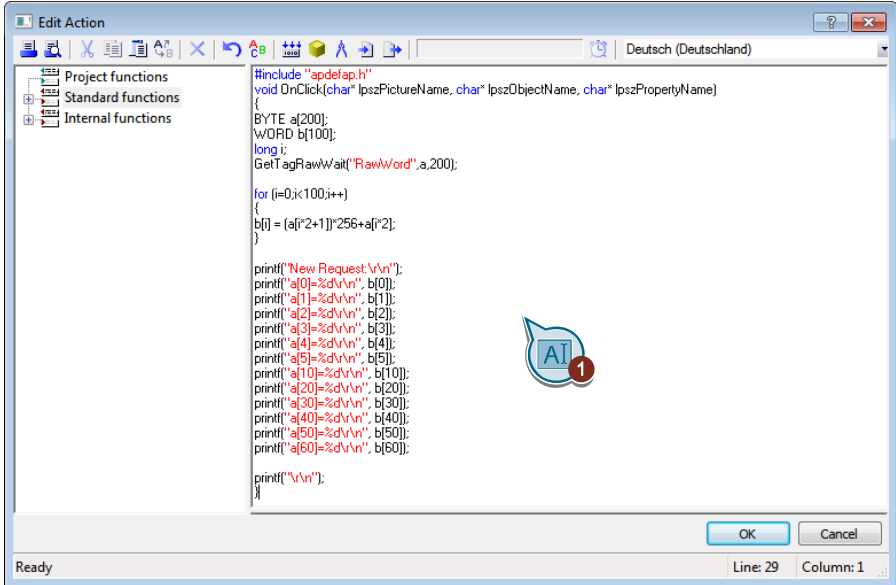
6.1 Example from Byte to Word

Table 6-1

Item	Action																																										
1.	Add a new data block. Create a byte array like in chapter 4.2.																																										
2.	Establish a new connection to the data block in the Tag Management. <div data-bbox="486 443 1385 566" data-label="Table"> <table border="1"> <thead> <tr> <th colspan="7">Tags [RawData]</th> </tr> <tr> <th>Name</th> <th>Data type</th> <th>Length</th> <th>Connection</th> <th>Address</th> <th>Linear scaling</th> <th>AS value range</th> </tr> </thead> <tbody> <tr> <td>1 Raw</td> <td>Raw Data Type</td> <td>8000</td> <td>RawData</td> <td>0001:CY:4101:8000:%D81.%D8B0</td> <td><input type="checkbox"/></td> <td></td> </tr> <tr> <td>2 RawWord</td> <td>Raw Data Type</td> <td>8000</td> <td>RawData</td> <td>0001:CY:4101:8000:%D82.%D8B0</td> <td><input type="checkbox"/></td> <td></td> </tr> <tr> <td>3</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>4</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table> </div>	Tags [RawData]							Name	Data type	Length	Connection	Address	Linear scaling	AS value range	1 Raw	Raw Data Type	8000	RawData	0001:CY:4101:8000:%D81.%D8B0	<input type="checkbox"/>		2 RawWord	Raw Data Type	8000	RawData	0001:CY:4101:8000:%D82.%D8B0	<input type="checkbox"/>		3							4						
Tags [RawData]																																											
Name	Data type	Length	Connection	Address	Linear scaling	AS value range																																					
1 Raw	Raw Data Type	8000	RawData	0001:CY:4101:8000:%D81.%D8B0	<input type="checkbox"/>																																						
2 RawWord	Raw Data Type	8000	RawData	0001:CY:4101:8000:%D82.%D8B0	<input type="checkbox"/>																																						
3																																											
4																																											
3.	Insert the following script into your project for sending data. <div data-bbox="486 645 1385 1019" data-label="Code-Block"> <pre> #include "apdefap.h" void OnClick(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName) { BYTE a[200]; WORD b[100]; long i; BYTE hilf; for (i=0;i<100;i++) { b[i] = (WORD)i*50; a[i*2] = (BYTE) (b[i] & 0xFF); a[i*2+1] = (BYTE) ((b[i] >> 8) & 0xFF); } SetTagRawWait("RawWord",a,200); } </pre> </div> <p>The script generates 100 tags in 50 steps with the help of a “for” loop. Byte 0 and byte 1 are masked out and written into a new array. This array is written as raw data to the control system using the „SetTagRawWait“ command.</p> <pre> BYTE a[200]; WORD b[100]; long i; BYTE hilf; for (i=0;i<100;i++) { b[i] = (WORD)i*50; a[i*2] = (BYTE) (b[i] & 0xFF); a[i*2+1] = (BYTE) ((b[i] >> 8) & 0xFF); } SetTagRawWait("RawWord",a,200); </pre>																																										

6 Converting raw data to other data types

6.1 Example from Byte to Word

Item	Action
4.	<p data-bbox="486 304 1129 338">Insert the following script into your project for receiving data.</p> <div data-bbox="486 338 1385 920"><pre data-bbox="735 405 1209 846">#include "apdefap.h" void OnClick(char* lpszPictureName, char* lpszObjectName, char* lpszPropertyName) { BYTE a[200]; WORD b[100]; long i; GetTagRawWait("RawWord",a,200); for (i=0;i<100;i++) { b[i] = (a[i*2+1])*256+a[i*2]; } printf("New Request:\r\n"); printf("a[0]=%d\r\n", b[0]); printf("a[1]=%d\r\n", b[1]); printf("a[2]=%d\r\n", b[2]); printf("a[3]=%d\r\n", b[3]); printf("a[4]=%d\r\n", b[4]); printf("a[5]=%d\r\n", b[5]); printf("a[10]=%d\r\n", b[10]); printf("a[20]=%d\r\n", b[20]); printf("a[30]=%d\r\n", b[30]); printf("a[40]=%d\r\n", b[40]); printf("a[50]=%d\r\n", b[50]); printf("a[60]=%d\r\n", b[60]); printf("\r\n"); }</pre></div> <p data-bbox="486 927 1385 987">The function „GetTagRawWait“ is used to read the raw data. Byte[0] and byte[1] are merged with the help of a “for” loop and output afterward.</p> <pre data-bbox="486 1021 879 1834">BYTE a[200]; WORD b[100]; long i; GetTagRawWait("RawWord",a,200); for (i=0;i<100;i++) { b[i] = (a[i*2+1])*256+a[i*2]; } printf("New Request:\r\n"); printf("a[0]=%d\r\n", b[0]); printf("a[1]=%d\r\n", b[1]); printf("a[2]=%d\r\n", b[2]); printf("a[3]=%d\r\n", b[3]); printf("a[4]=%d\r\n", b[4]); printf("a[5]=%d\r\n", b[5]); printf("a[10]=%d\r\n", b[10]); printf("a[20]=%d\r\n", b[20]); printf("a[30]=%d\r\n", b[30]); printf("a[40]=%d\r\n", b[40]); printf("a[50]=%d\r\n", b[50]); printf("a[60]=%d\r\n", b[60]); printf("\r\n");</pre>

7 Using the Application

Make sure that the CPU and WinCC are set to the same clock time (UTC or local time) before you start the application project for the first time (see chapter 4.3).

7.1 Commission the example project

Table 7-1

Item	Action
1.	Unzip the file „37873547_Rohdaten_TIA_S7-1500.zip“
2.	Start the TIA Portal.
3.	Retrieve the „Rohdaten.zap13“ project.
4.	Download the project to the control system.
5.	Unzip the file „37873547_Rohdaten_WinCC_V74.zip“.
6.	Start the SIMATIC WinCC explorer.
7.	Open the file “BSENBRCV.MCP”.
8.	Make fit the IP addresses of your PC station and your control system.
9.	Start Runtime.

7.2 Using the example project

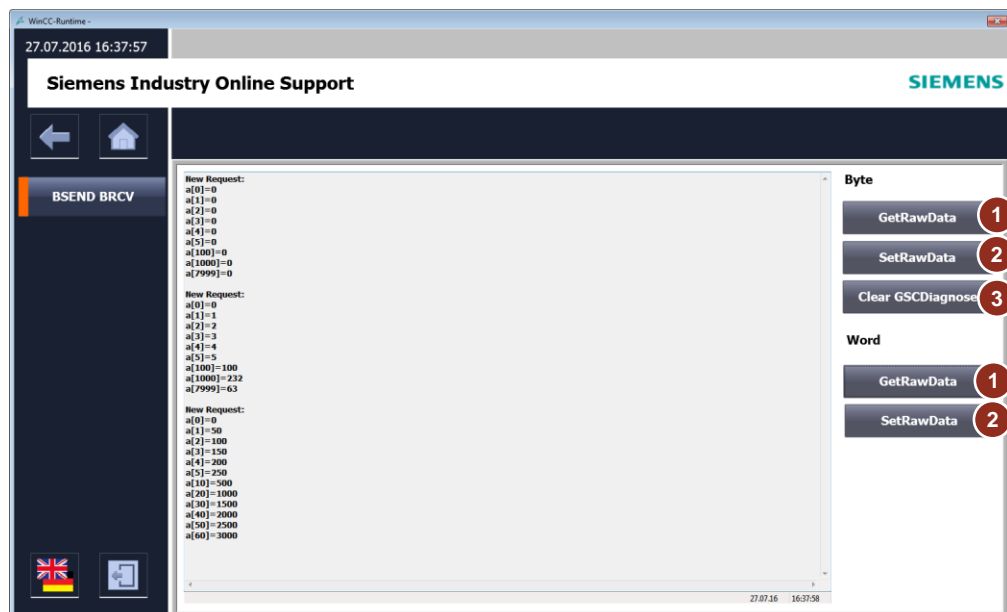


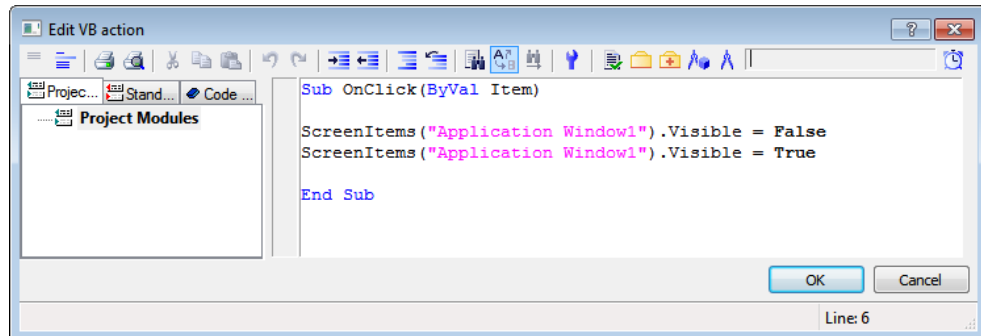
Table 7-2

Item	Action
1.	Use the „GetRawData“ button to receive raw data.
2.	Use the „SetRawData“ button to send raw data.
3.	Use the “Clear GSCDiagnose” button to erase the contents of GSCDiagnostics.

8 Further Notes, Tips & Tricks, etc.

Erasing the GSC Diagnostics contents

GSC Diagnostics does not provide a button to erase text from the printf function in the control system. As a result, the continuous printf orders build a scroll bar in GSC Diagnostics and are filled with contents. If you wish to erase the contents, you can do it by showing/hiding for a short time the GSC Diagnostics window by means of a script. For this purpose insert the VB script below to a button in your project.



9 Links & References

Table 9-1

	Topic
\1\	Siemens Industry Online Support https://support.industry.siemens.com
\2\	Download page of the entry https://support.industry.siemens.com/cs/ww/de/view/37873547

10 History

Table 10-1

Version	Date	Modifications
V1.0	09/2016	First version