Configuring RT communication between SIMATIC and SIMOTION (I-Device)

SIMATIC & SIMOTION

Application Example • June 2012

Applications & Tools

Answers for industry.

SIEMENS

Siemens Industry Online Support

This article originates from the Siemens Industry Online Support. The following link takes you directly to the download page for this document:

http://support.automation.siemens.com/WW/view/en/38494882

Caution
The functions and solutions described in this article confine themselves to the realization of the automation task predominantly. Please take into account furthermore that corresponding protective measures have to be taken up in the context of Industrial Security when connecting your equipment to other parts of the plant, the enterprise network or the Internet. Further information can be found under the Item-ID 50203404.

http://support.automation.siemens.com/WW/view/en/50203404

You can also actively use our Technical Forum from the Siemens Industry Online Support regarding this subject. Add your questions, suggestions and problems and discuss them together in our strong forum community:

http://www.siemens.com/forum-applications

S

SIMATIC & SIMOTION
RT I-Device

Application Example

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.

Table of contents

1 Problem

1.1 Overview

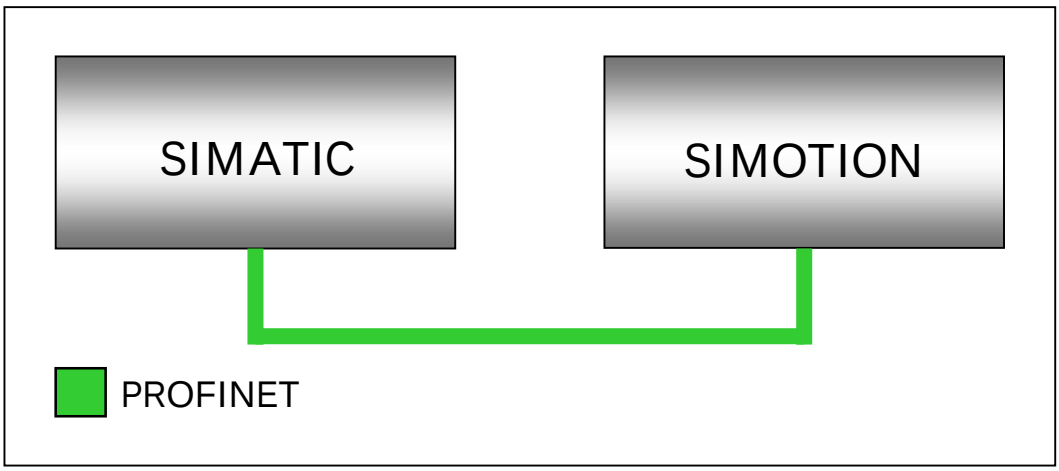
Introduction

The **I-Device** functionality has been available since SIMOTION V4.1.1 in order to exchange data between SIMATIC and a SIMOTION controllers via PROFINET. This means that in this case the SIMOTION controller acts as "intelligent IO device" = **I-Device**. This means that data is exchanged between the controllers via the I/O areas.

Overview of the automation task

The following figure provides an overview of the automation task.

Fig. 1-1



Description of the automation task

The SIMATIC CPU is the higher-level IO controller in the PROFINET IO system. The SIMOTION controller is configured as **I-Device** and is therefore an IO device of the SIMATIC CPU.

Communication can be established in both directions (bidirectional) via I/O areas using the **I-Device** functionality.

Further, this function allows both controllers to operate in separate projects.

Further, using the **I-Device**, a machine module can be encapsulated, so that this can then be used several times in a large plant or system.

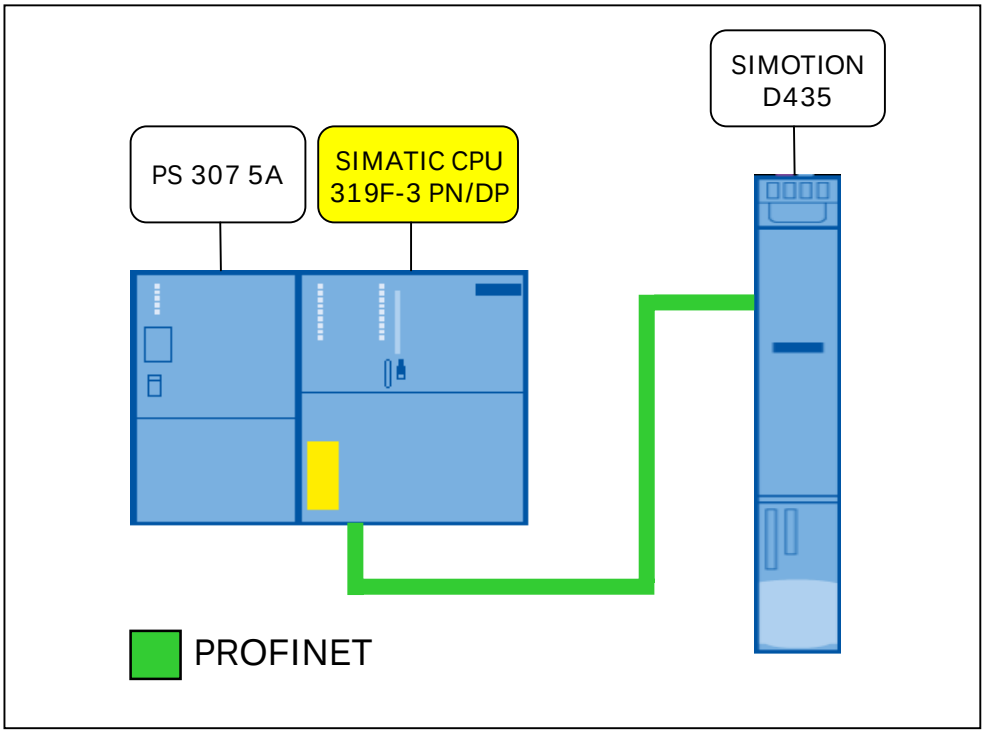
2 Solution

2.1 Overview of the overall solution

Schematic diagram

The following figure schematically shows the most important components of the solution:

Fig. 2-1



In this application, a description is given on how a SIMOTION **I-Device** connected to a SIMATIC CPU can be configured in order to exchange data between both controllers via PROFINET RT.

In the project example, 254 bytes are sent from the SIMOTION controller to the SIMATIC CPU. The SIMATIC controller then returns the received data, mirrored.

Required knowledge

It is assumed that the reader has a basic knowledge about configuring SIMATIC and/or SIMOTION CPUs using the STEP7 or SIMOTION SCOUT engineering system.

2.2 Hardware and software components used

The project example has been created with the following components.

Hardware components

Table 2-1

Component	Qty.	MLFB / order number	Note
SIMATIC 319F-3 PN/DP	1	6ES7318-3FL01-0AB0	Firmware V3.2
SIMOTION D435	1	6AU1435-0AA00-0AA1	Firmware V4.2
CBE30	1	6FC5312-0FA00-0AA0	Option board for D4x5

Note

The project example was created with the hardware components listed here.

Alternatively, other components with the same function may be used. A different parameter assignment and different wiring of the components may be required.

Regarding the firmware versions, please observe the note on page 24.

Standard software components

Table 2-2

Component	Qty.	MLFB / order number	Note
STEP7	1	6ES7810-4CC10-0YA5	V 5.5 HF4
SIMOTION SCOUT	1	6AU1810-1BA42-1XE0	V 4.2 SP1

Sample files and projects

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

Table 2-3

Component	Note
61449067_Example_RT_I-Device_V1_3.zip	ZIP archive contains project example, ST units and an address list
61449067_Application_Example_RT_I-Device_V1_3_en.pdf	This document

3 Basics

3.1 PROFINET communication

Not only the MAC address but also the device name is used to identify the devices for PROFINET. This device name must be unique across the PROFINET network.

During the commissioning phase, the HW Config or the Primary Setup Tool (PST) is used to make an initial online assignment of a device name for each PROFINET device (a so-called node initiation). This device name is stored retentively in the PROFINET device and must match the device name in the project.

If a device is replaced, e.g. because of a defect, the new device has a different MAC address. If it is initialized with the same device name as the replaced device (e.g. by reconnecting the removable medium that stores the device name retentively), it can take over the function of the replaced device without any changes in the configuration.

3.1.1 Rules for assigning names

A PROFINET device must be named with the corresponding device name. This name must match the device name in the project. The following rules apply in this case:

- The device name stored retentively in the device should only contain lowercase letters.

Note

The device name in the project must only contain uppercase letters. During the node naming, the engineering system replaces the uppercase letters with lowercase letters.

- Letters a-z and digits 0-9 may be used.
- Special characters are not permitted: ! " § $ % & / ( ) = ? * ' _ : ; > < , # + | ~ \ }] [{
- Blanks are also not permitted
- The total maximum length of a name is 240 characters.
- Reserved names that must not be used:
  "port-xyz" or "port-xyz-abcde" a, b, c, d, e, x, y, z = 0...9
- The minus character must not be used for a SIMOTION controller.
- The engineering system software replaces impermissible characters with an “x".

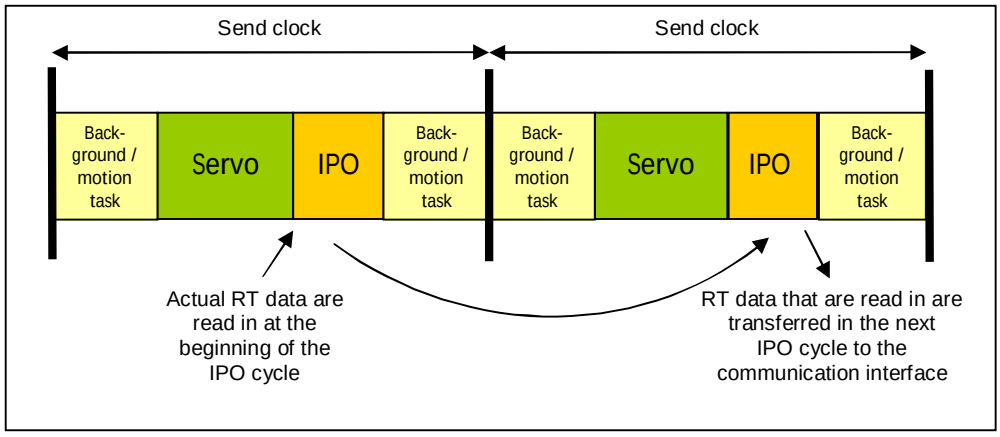
3.1.2 RT communication

On the SIMOTION controller side, the RT data are updated with each IPO clock cycle, i.e. in the IPO clock cycle, the actual RT data are read in and the RT data from the previous clock cycle are transferred to the communication interface.

Note

As from SIMOTION V4.2, the instant in time that RT data are updated can be set in SCOUT under system clock (IPO or IPO_2).
The IPO_2 clock cycle is the default setting.

Fig. 3-1



3.1.3 I-Device

With the PROFINET functionality I-Device, up to 1440 bytes of user data can be transferred in each direction between the IO controller and I-Device. The user data is made up of input or output data + user data supplement.

For each submodule (irrespective of input or output), one byte of user data supplement is automatically created in the input and output data area. The number of bytes for the user data supplement, is not directly obvious for the I-Device configuration, and is dependent on the number of submodules.

A submodule should always be selected to be as large as possible.

SIMATIC

As from STEP7 5.5 and higher, the functionality I-Device is also available for SIMATIC controller systems. Contrary to a SIMOTION CPU, a SIMATIC CPU can be linked to two higher-level I/O controllers as **Shared I-Device**. Data exchange between the controllers is realized via submodules. The maximum size of a submodule is 1024 bytes.

SIMOTION

A submodule of the SIMOTION I-Device can have a maximum size of 254 bytes. Copying a submodule in the controller itself involves a relatively high CPU load; this is the reason that it is recommended that only a few submodules are configured, which are then large.

Note

The SIMOTION I-Device can only be assigned to one I/O controller. The SIMOTION CPU does not support PROFINET **Shared I-Device** functionality.

3.1.4 Consistent data transfer

In many applications, data consistency is a precondition when transferring data. This means that for a contiguous data block (submodule), the input or output data are completely updated at a defined instant in time (consistent). Otherwise it is possible that a part of the data block already contains new data while the other part still has old data.

SIMATIC

On the SIMATIC CPU side, SFC14 and SFC15 are available for RT communication. When using these system functions, it is ensured that the data are consistently read or written.

Note As from STEP7 5.5 and higher, the SIMATIC CPU also supports IRT communication. In this case, it should be noted that to consistently read and write IRT data, the SFC126 and SFC127 system functions must be used.

SIMOTION

For the RT I-Device, when transferring RT data, data transfer is only consistent per submodule. For a SIMOTION controller, a submodule can have a maximum size of 254 bytes.

As the access to a submodule involves a relatively high CPU load (IPO or IPO_2 load), only as few submodules as possible should be configured. Instead of creating many small submodules, a higher performance is achieved by using a few large submodules.

Consistent data blocks > 254 bytes

If a larger data block (RT data), made up of several submodules, is to be consistently read, then data consistency must be ensured in the application itself. For this purpose, in the input data area, one byte for each submodule should be used as counter, for example. Data consistency is established by monitoring this counter in the application itself.

Note IRT data transfer is always consistent. If this data is processed in a task, which can be interrupted by a ServoTask (e.g. BackgroundTask), then this data must be allocated a process image (e.g. process image of the Background Task).

3.2 Consistent data processing

If the I/O data are processed in a cyclic task, then this I/O data should be allocated a process image. As a consequence, the I/O data are consistently updated at the beginning and at the end of each particular task.

For a SIMOTION controller, four process images are available:

- ServoSynchronousTask
- IPOsynchronousTask
- IPOsynchronousTask_2
- BackgroundTask

When accessing IO data in non-cyclic tasks (e.g. MotionTask), then it is preferable that the actual status is interrogated.
When required, you can create a process image in the application on local variables (consistent image by copying to temporary variables, e.g. at the instant that MotionTask starts).

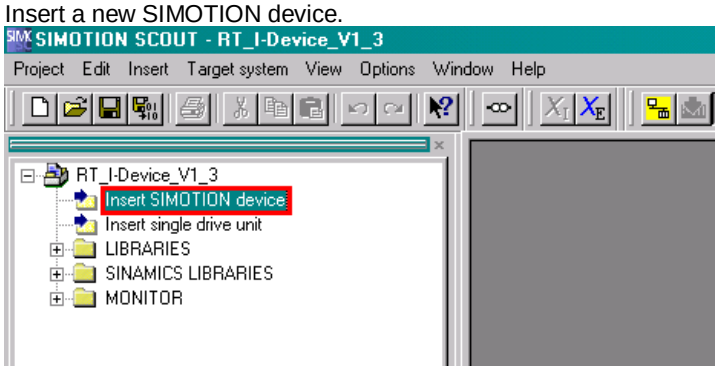
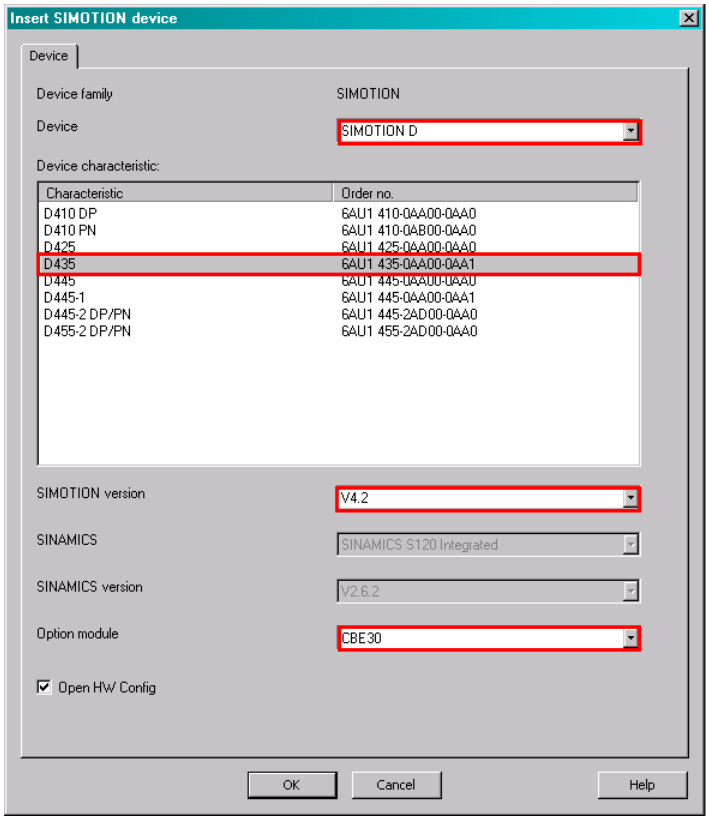
Note A process image is consistently accessed, and has a higher performance than direct access, which in turn reduces the system load.

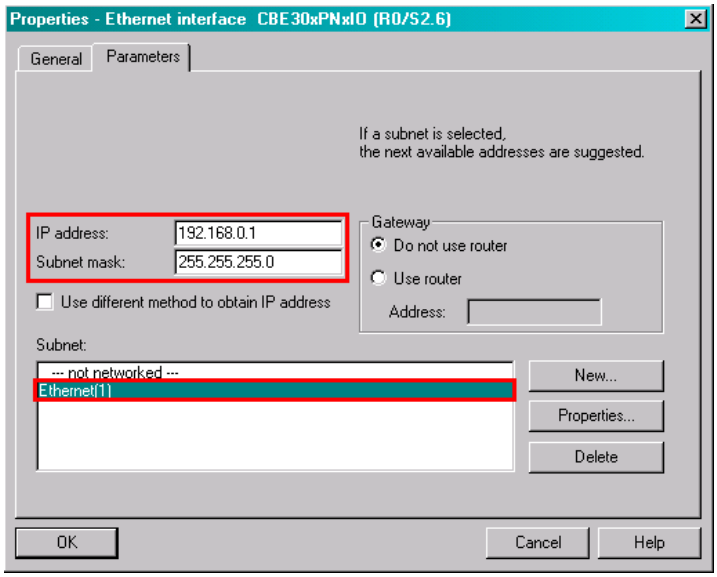
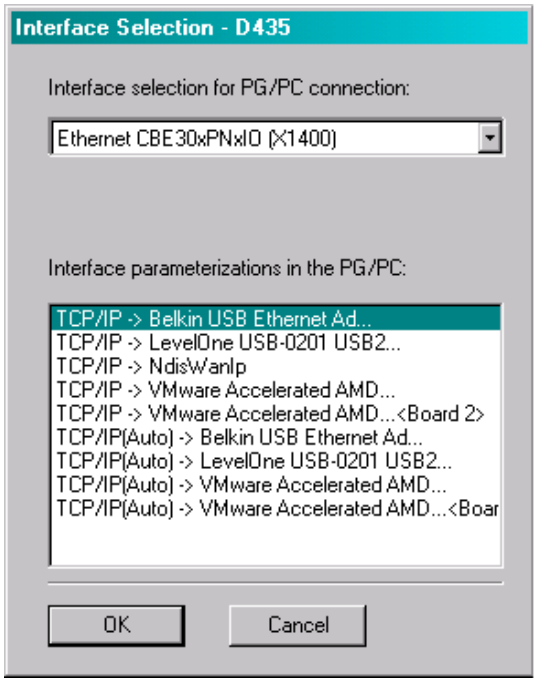
4 Configuration

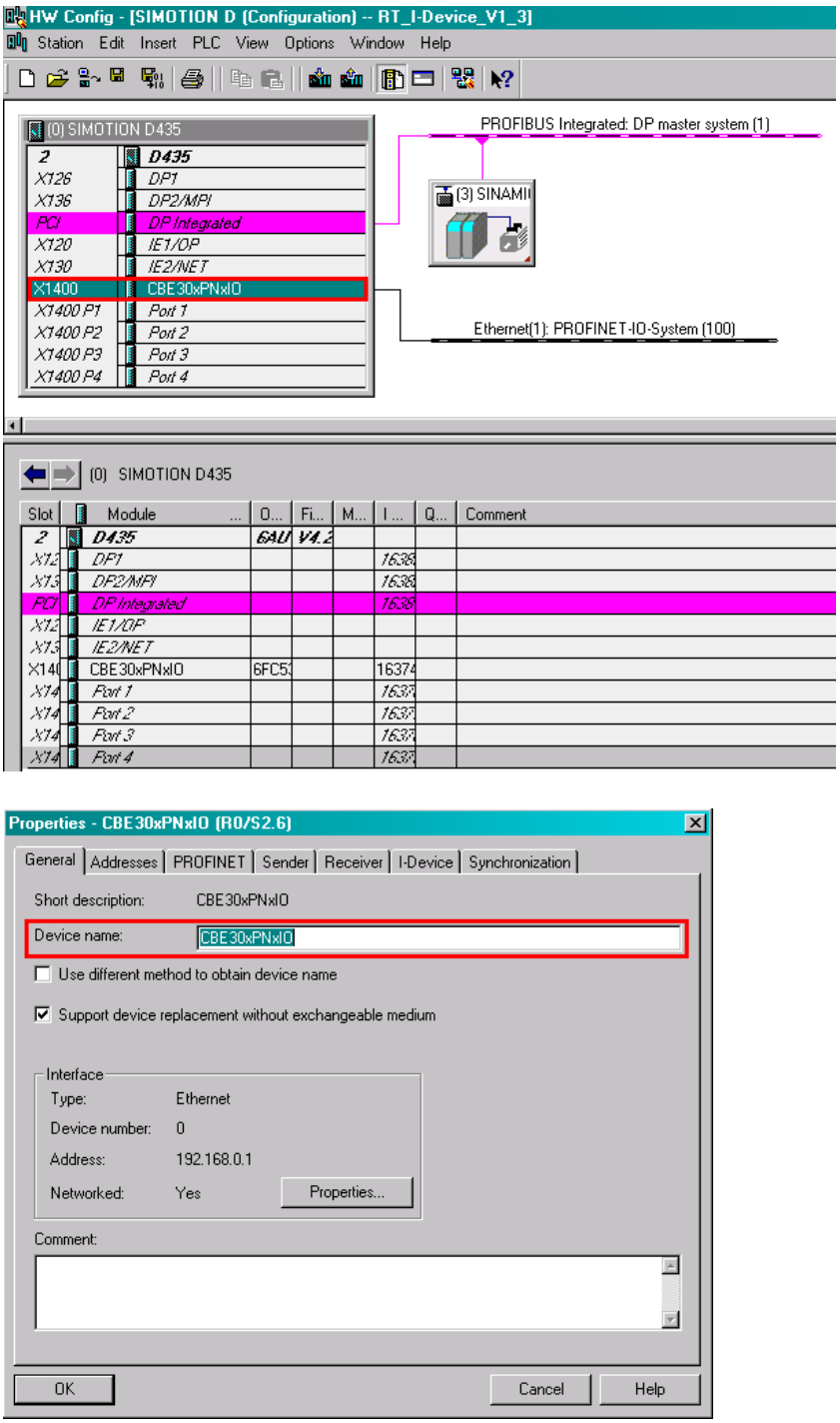
4.1 HW Config of SIMOTION

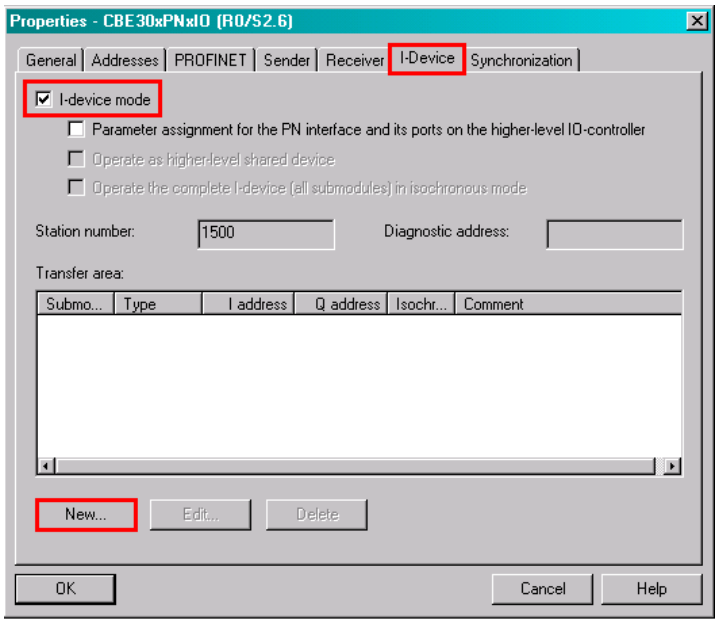
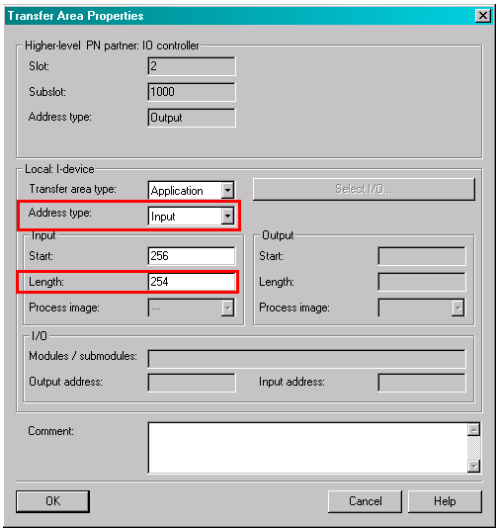
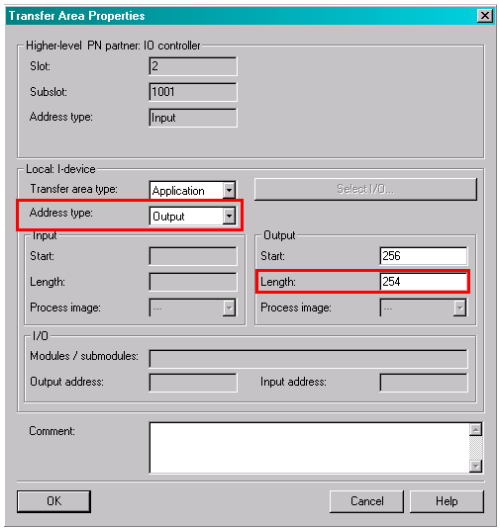
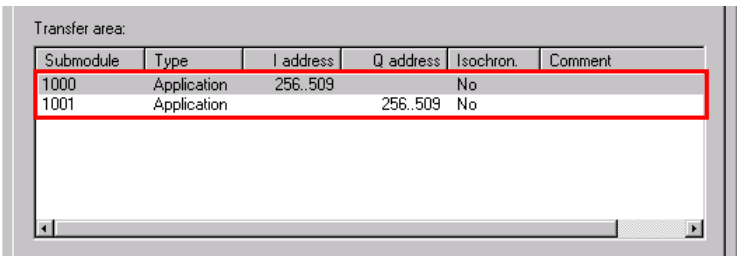
In the application, a SIMOTION D435 is used, which is configured as follows.

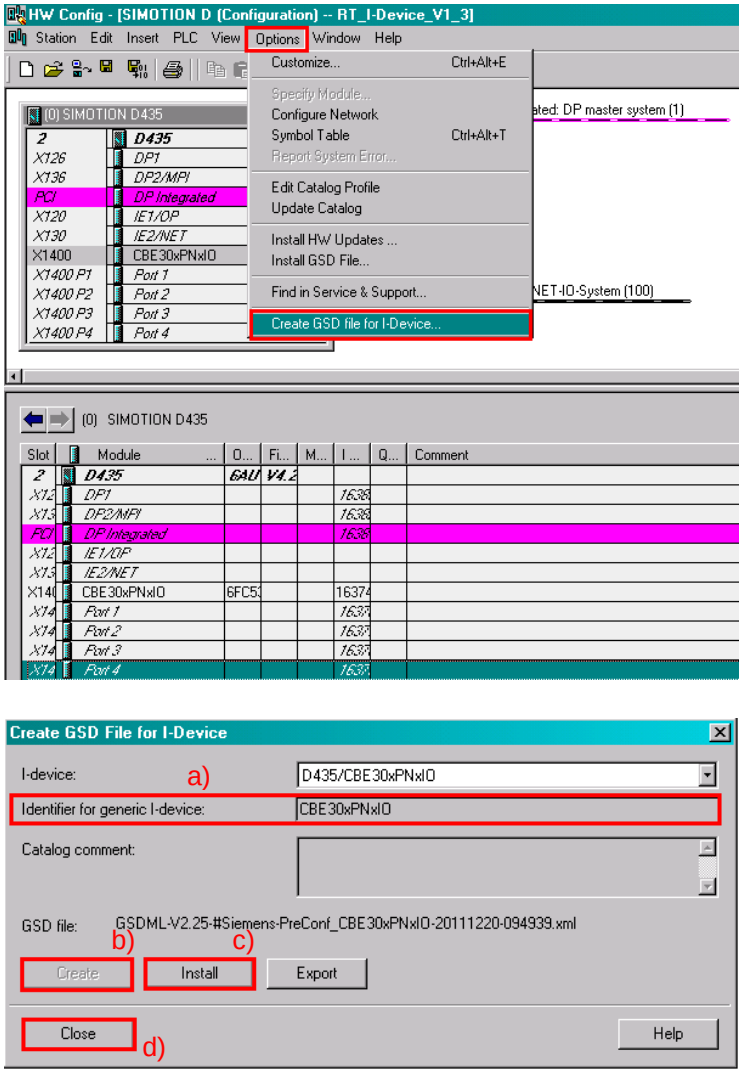
Table 4-1

No.	Action
1.	To create a new project, open SIMOTION SCOUT. Insert a new SIMOTION device. 
2.	Select the corresponding SIMOTION hardware platform and version (min. V4.1).  In the application example, communication to the SIMATIC CPU is established via the CBE30 option module (Communication Board Ethernet).

No.	Action
3.	Create a new Ethernet subnet and assign an IP address. 
4.	Configure the connection between the SIMOTION controller and the engineering system. 

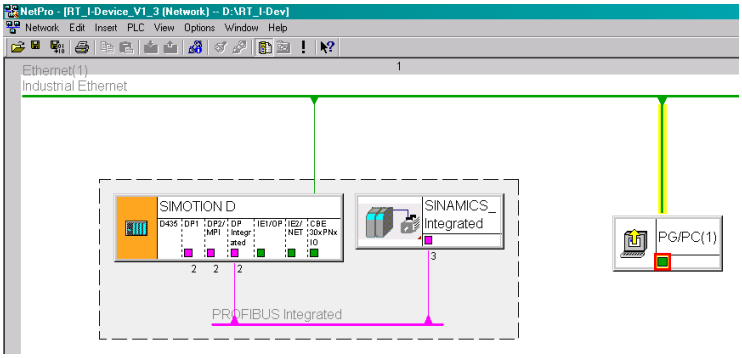
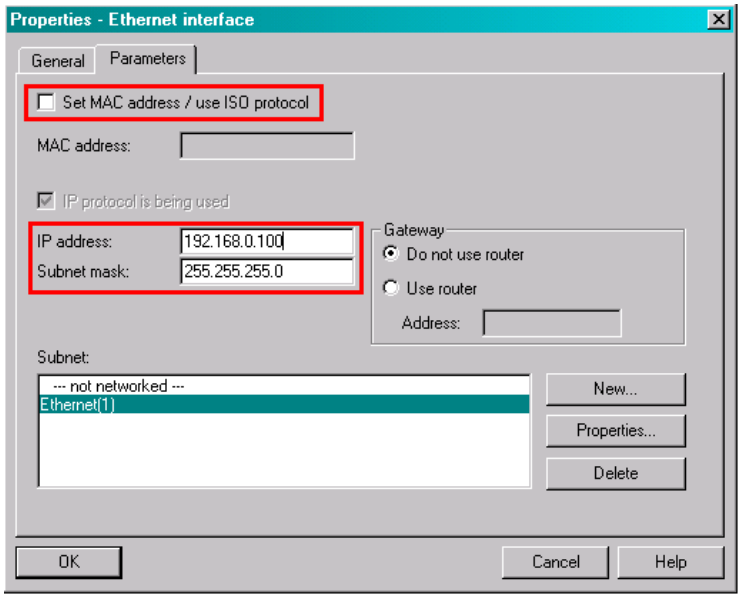
No.	Action
5.	Pressing key F4 automatically arranges the available modules in HW Config. The device name of the SIMOTION controller is "CBE30xPNxIO", and can be adapted by double-clicking on the PN interface. 

No.	Action
6.	Change to the "I-Device" tab and activate the "I-Device mode".  Create two new submodules. These can be subsequently used as data transfer areas for communication between the SIMATIC CPU and SIMOTION controller. A maximum of 1440 bytes can be configured for the send and receive directions (including the user data supplement). The maximum size of a submodule is 254 bytes.   

No.	Action
7.	Generate the GSD file for the previously configured I-Device. In order to do this, perform a click on "Options > Create GSD file for I-Device".  a) Allocate a unique name for the GSD file. The file can be found under this name in HW catalog afterwards. b) Perform a click on "Create" to generate the GSD file. c) Install the GSD file on the engineering system the higher-level I/O controller is configured with. If the controller is in the same project or is configured with the same engineering system, just perform a click on “Install”. d) Close the window. The HW catalog then gets updated automatically.

Note

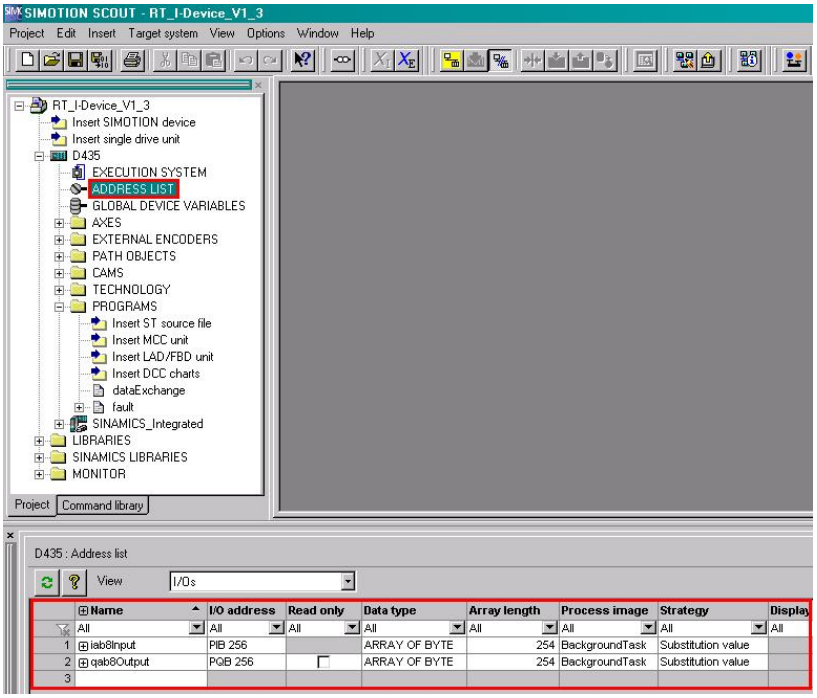
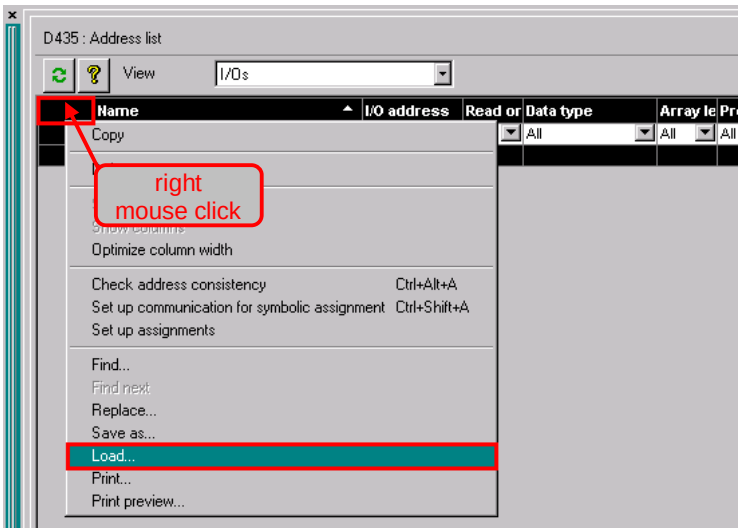
Every subsequent change to the configuration of the I-Device means that a new GSD file must be generated!
In the STEP7 installation folder GSD files that are no longer required can be deleted under "Step7 > S7data > GSD".

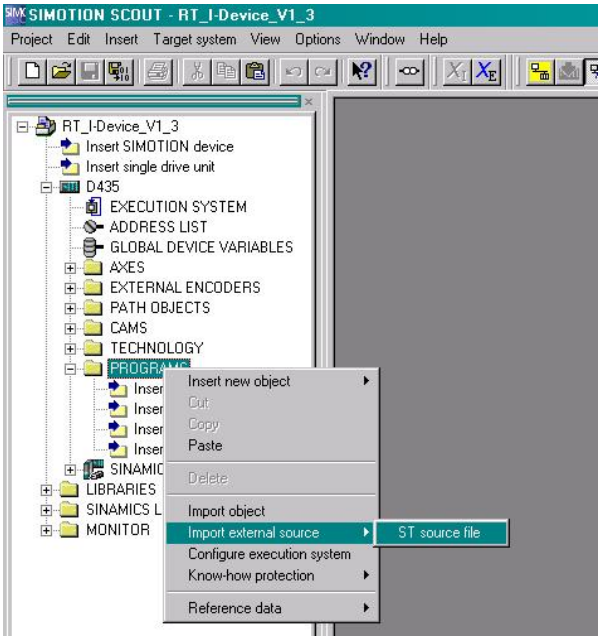
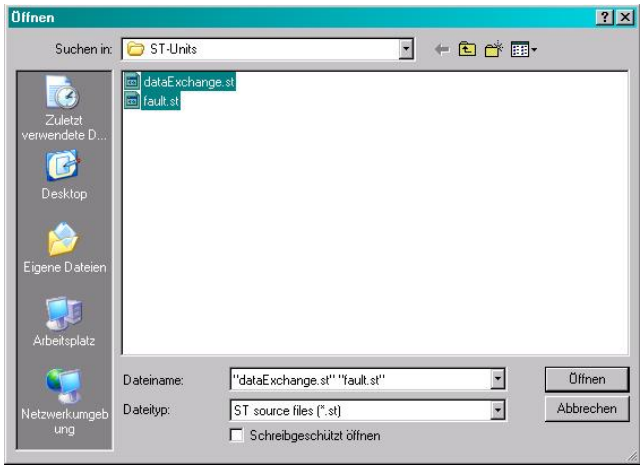
No.	Action
8.	Open NetPro in order to set the Ethernet interface of the engineering system. The yellow connection at the PG/PC interface indicates that the engineering system can access SINAMICS Integrated via S7 routing. Routing tables are automatically generated when saving and compiling; these must be saved in the device by downloading the HW Config.  Double-clicking on the PG/PC interface opens their properties.
9.	Deactivate the check mark "Set MAC address / use ISO protocol", as the SIMOTION controller does not support any ISO protocol.  In the application example, the engineering system has the following IP address: IP address: `192.168. 0 .100` Subnet mask: `255.255.255. 0`
10.	Save and compile the HW Config. 
11.	Load the HW Config to the target system. 

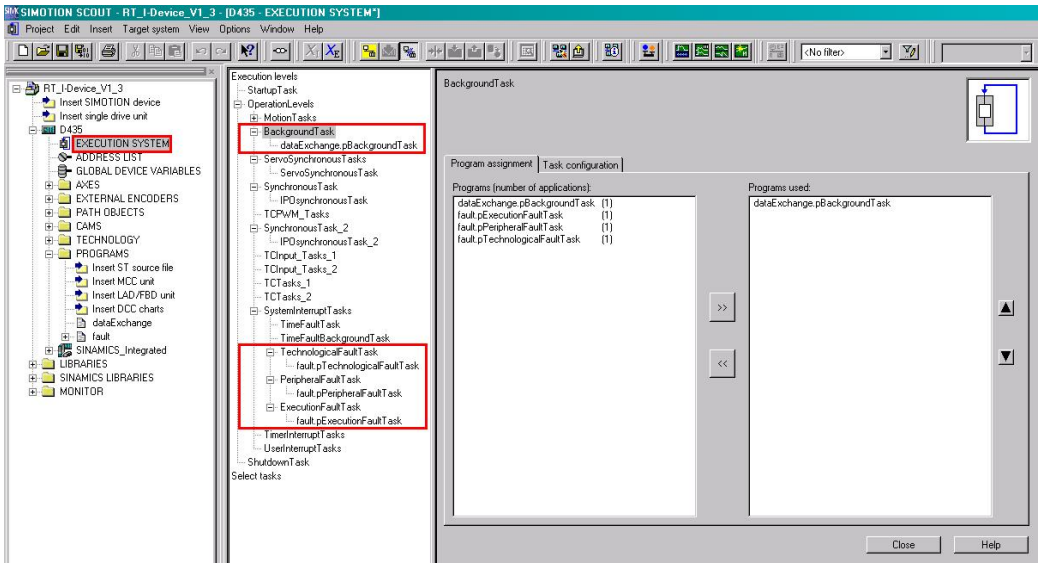
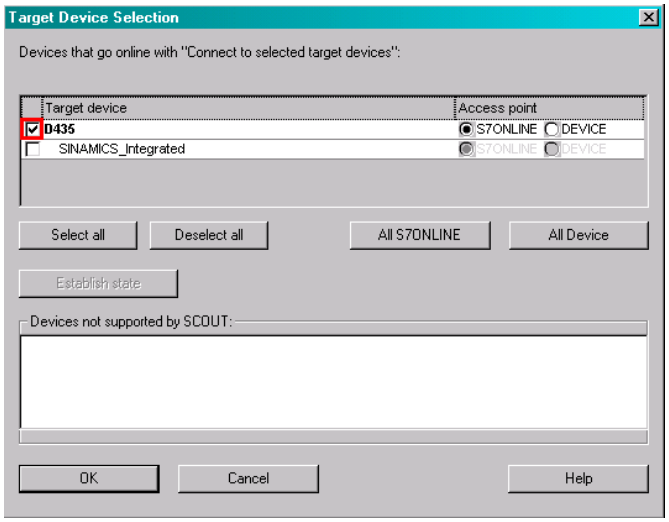
4.2 Configuring the SIMOTION controller

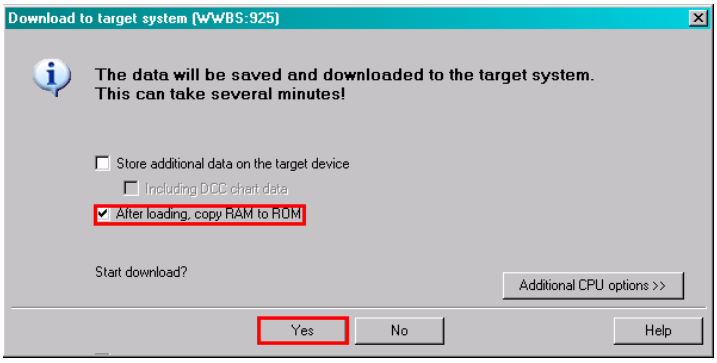
Configuring the SIMOTION controller using the SIMOTION SCOUT engineering system is shown in the following.

Table 4-2

No.	Action
12.	In the example project, data in the “iab8Input” array are incremented and copied into the “qab8Output” array. Data is sent to the SIMATIC CPU from up there.  In the folder "AddressList" of the ZIP archive "61449067_Example_RT_I-Device_V1_3.zip", there is the "IO_Variables.csv" file, which can be imported into the address list of the SIMOTION controller using the right-hand mouse key > "Load". 

No.	Action
13.	The "fault" source is provided in the example project. This source contains 3 programs, which are called in the particular FaultTask. The "dataExchange" source contains the "pBackgroundTask" program, in which the data are incremented and sent to the SIMATIC CPU. As the name implies, this program is called in the BackgroundTask. The sources can be imported from the "ST-Units" folder of the ZIP archive "61449067_Example_RT_I-Device_V1_3.zip".  Several ST sources can be simultaneously imported.  After the import, the SIMOTION project must be saved and compiled in order that the programs are visible in the execution system.

No.	Action
14.	Open the execution system of the SIMOTION controller and insert the example programs in the respective task. 
15.	Save and compile the SIMOTION project. 
16.	Establish an online connection to the SIMOTION controller. 
17.	Select the SIMOTION controller as target system in order to go online. 

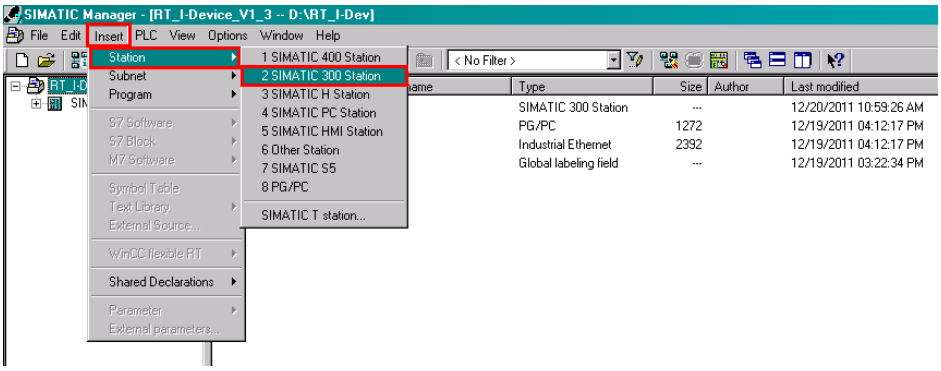
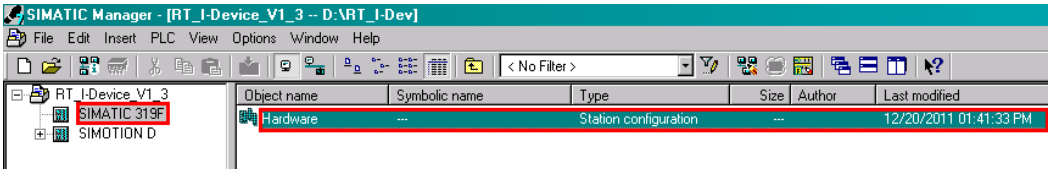
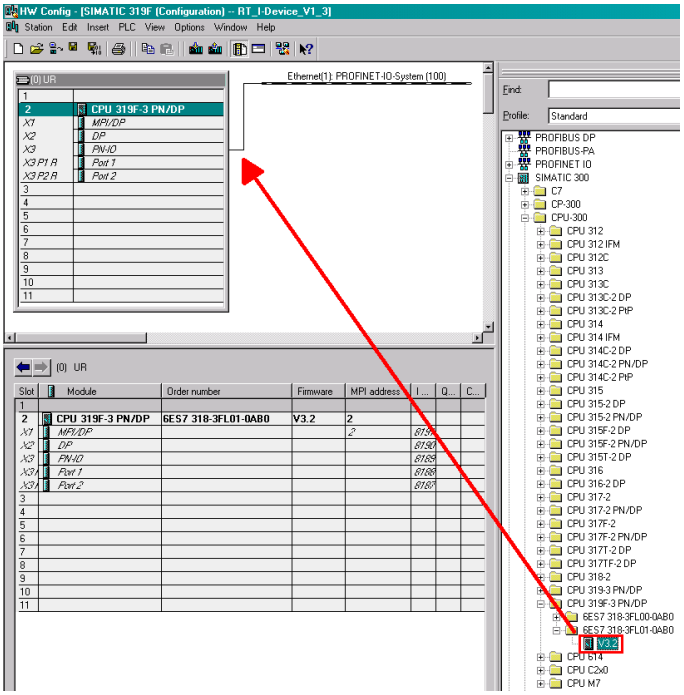
No.	Action
18.	Load the complete project into the SIMOTION controller. 
19.	Set the check mark "After loading, copy RAM to ROM" and start the download. 

4.3 HW Config of the SIMATIC CPU

In the application example, a SIMATIC CPU 319F-3 PN/DP is used as higher-level I/O controller, which can be configured as follows.

In the example, safety operation of the CPU is not activated.

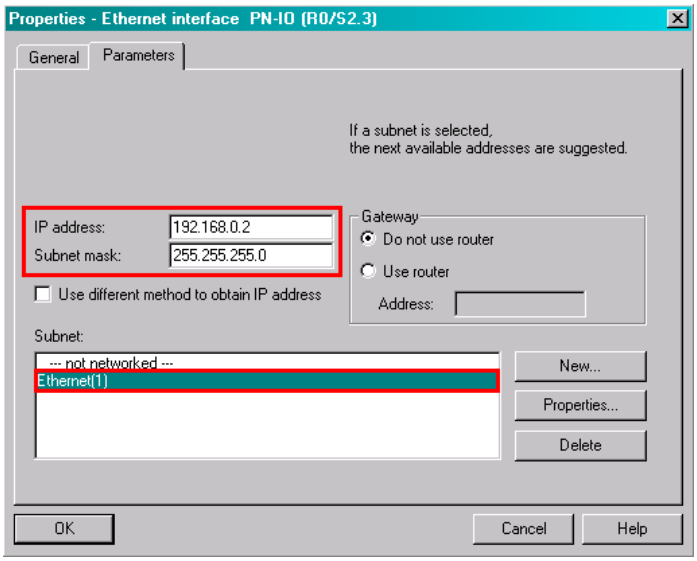
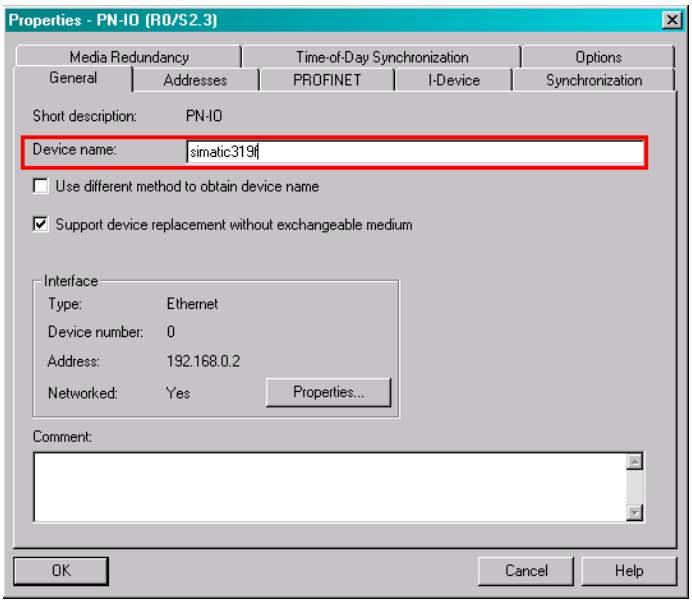
Table 4-3

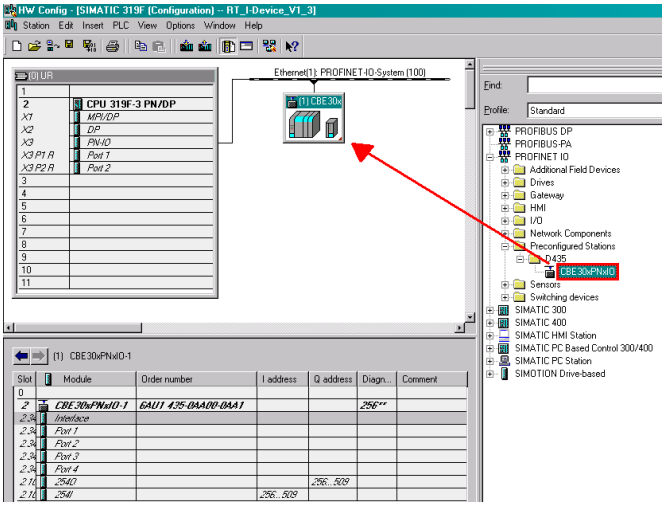
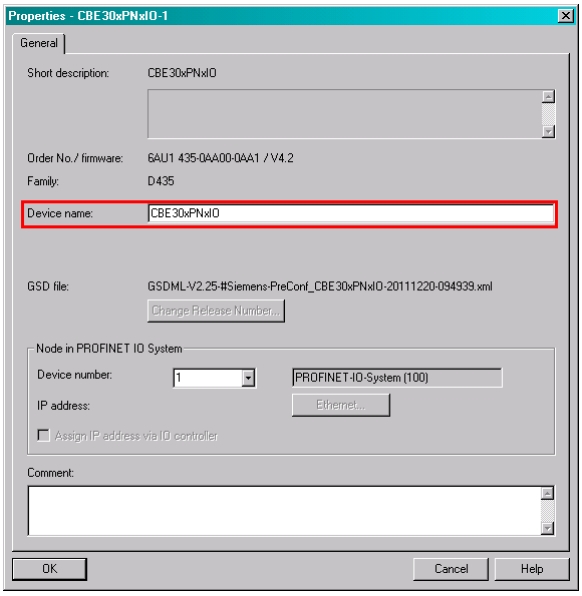
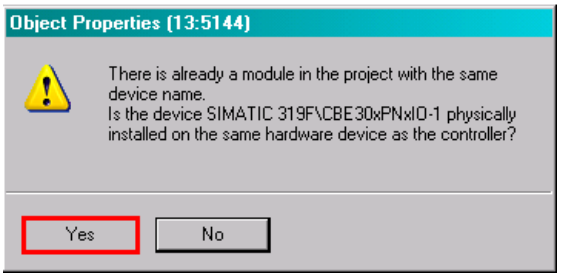
No.	Action
1.	Open the SIMOTION project with the SIMATIC Manager and insert a new S7-300 station. 
2.	Open the HW Config. 
3.	A SIMATIC CPU 319F-3 PN/DP is used in the project example.  Pressing the F4 key automatically arranges the modules present in the HW Config.

Note

For the I-Device of a SIMOTION V4.1, an S7-300 CPU must have at least firmware V2.6 or higher, an S7-400 CPU must have at least firmware V5.1.1 or higher.

For the I-Device of a SIMOTION V4.2 or higher, an S7-300 CPU must have at least firmware V3.2 or higher, an S7-400 CPU must have at least firmware V6.0 or higher.

No.	Action
4.	Select the existing Ethernet subnet and allocate an IP address for the higher-level I/O controller. 
5.	Double-clicking on the PROFINET interface (PN-IO) opens the properties window. Define a device name. In the example project the device name is "simatic319f". 

No.	Action
6.	Drag the previously generated GSD file of the SIMOTION controller to the PROFINET IO system. You will find the GSD file in the HW catalog under "PROFINET IO > Preconfigured Stations > D435 > CBE30xPNxIO".  Please note that the device name (X3) is "simatic319f" and not “PN-IO” as shown in the diagram.
7.	The properties window is opened by double-clicking on the inserted I-Device. Adapt the device name ("CBE30xPNxIO").  Acknowledge the following message with "Yes", as the I-Device substitute must have the same device name, as the associated controller. 

No.	Action
8.	Save and compile the HW Config. 
9.	Load the HW Config to the SIMATIC CPU. 

Note

Please ensure that the device name of the I-Device substitute corresponds to the device name of the associated I-Devices (the same hardware).

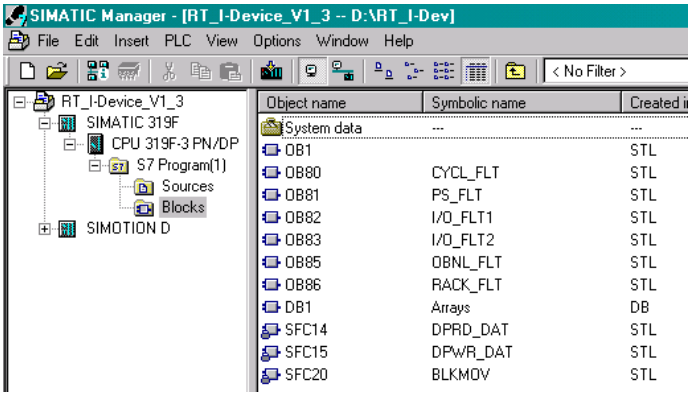
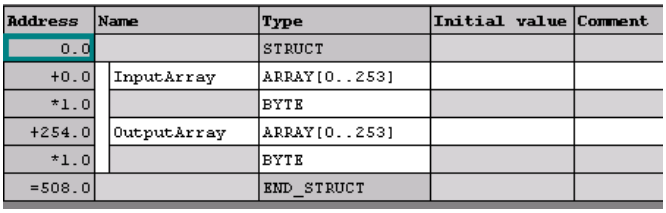
If the higher-level IO controller and the I-Device are in the same project, the engineering system automatically changes the device name when inserting the I-Device.

Actually, a device name must be unique within a PROFINET network. However, this rule does not apply, as in this case, it involves the same hardware!

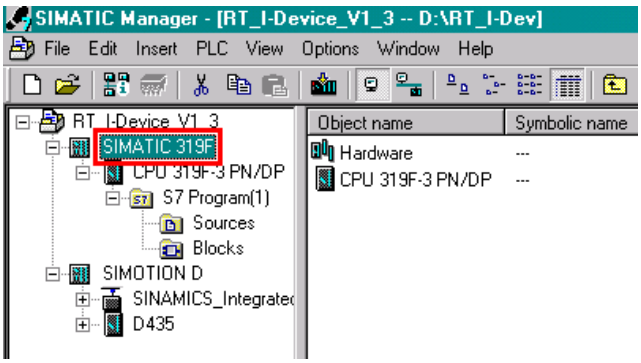
4.4 Configuring the SIMATIC CPU

Configuring the SIMATIC CPU using the SIMATIC Manager engineering system is shown in the following.

Table 4-4

No.	Action
1.	The blocks OB1, OB80-OB86, data block DB1 as well as the system functions SFC14/15 and SFC20 can be copied from the example project. This is located in the "Project" folder of the ZIP archive "61449067_Example_RT_I-Device_V1_3.zip". 
2.	Data block 1 contains two arrays with 254 bytes. Data, which the SIMATIC CPU receives from the SIMOTION controller, is written into the "InputArray". The "OutputArray" contains the data, which is sent from the SIMATIC CPU to the SIMOTION controller. 

No.	Action
3.	OB1 : "Main Program Sweep (Cycle)" Comment: Network 1: read consistent data of a standard dp slave Comment: CALL "DPRD_DAT" SFC14 -- Read Consistent LADDR :=W#16#100 RET_VAL:=MW10 RECORD :="Arrays".InputArray P#DB1.DBX0.0 Network 2: copy variables of 254 byte Comment: CALL "BLKMOV" SFC20 -- Copy Variables SRCBLK :="Arrays".InputArray P#DB1.DBX0.0 RET_VAL:=MW15 DSTBLK :="Arrays".OutputArray P#DB1.DBX254.0 Network 3: write consistens data of a standard dp slave Comment: CALL "DPWR_DAT" SFC15 -- Write Consisten LADDR :=W#16#100 RECORD :="Arrays".OutputArray P#DB1.DBX254.0 RET_VAL:=MW20 Reading and writing the IO data takes place in the example project in OB1. 1. Using the system function SFC14, the input data of logical address "LADDR" is consistently read and written into the "InputArray". • Parameter "LADDR" is assigned the start address of the input data of the I-Device substitute (W#16#100 corresponds to 256, see HW Config of the S7-CPU). • Parameter "RECORD" defines the target area in which the received data are written (in this case, the "InputArray" of data block DB1). 2. The read in data are copied from the "InputArray" to the "OutputArray" of the data block DB1 using the system function SFC20. 3. The output data of the “OutputArray” are finally written consistently to the logical address “LADDR” using the system function SFC15. • Parameter "LADDR" specifies the start address of the output data of the I-Device substitute (W#16#100 corresponds to 256, see HW Config of the S7-CPU). • The source data for the output data, which are sent to the SIMOTION controller, are available at parameter "RECORD" (in this case the "OutputArray" of data block DB1).

No.	Action
4.	Select the SIMATIC CPU in the STEP7 project after inserting the OBs, the DB and the system functions. 
5.	Perform a download to the CPU. 

5 Startup of the Application

The following steps must be performed to commission the example project.

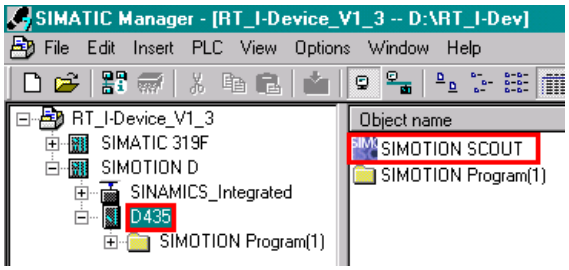
5.1 Preparation

Table 5-1

No.	Action
1.	All of the hardware components according to Table 2-1 (page 8) are available and upgraded to the corresponding firmware (see the note on page 24).
2.	All PROFINET components are networked and accessible from the engineering system.
3.	The Ethernet interface of the engineering system is configured correctly. IP address: `192.168. 0 .100` Subnet mask: `255.255.255. 0`

5.2 Startup

Table 5-2

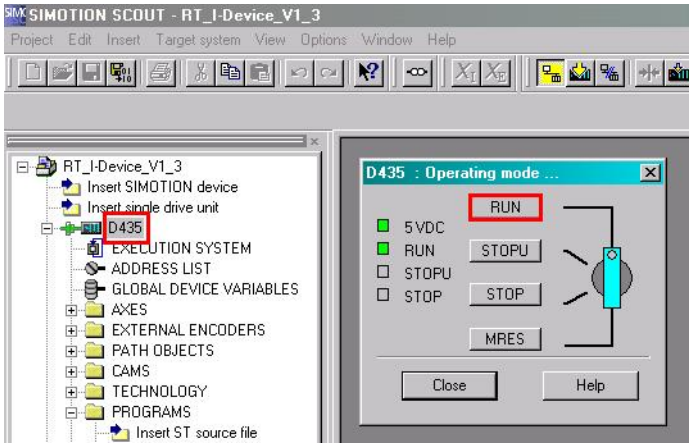
No.	Action
1.	Start the STEP7 SIMATIC Manager engineering system.
2.	Retrieve from archive the example project "61449067_Example_RT_I-Device_V1_3.zip". It is located in the "Project" folder of the ZIP archive "61449067_Example_RT_I-Device_V1_3.zip".
3.	Open the S7 project "RT_I-Device_V1_3".
4.	Load the configuration of the SIMATIC CPU into the associated device. In order to do this, first select the SIMATIC CPU and then perform a click on the "Download" button. 
5.	Open the SIMOTION SCOUT project out of the SIMATIC Manager. 
6.	Go online. 
7.	If it's not possible to establish a online connection with the SIMOTION controller, then first load the HW Config into the SIMOTION device in order to assign it the correct IP address and device name or assign the correct IP address to the PROFINET interface, over which you are connected with the engineering system, via "Target system > Ethernet > Edit Ethernet Node").
8.	Load the SIMOTION project into the target system. The example project can now be used. 

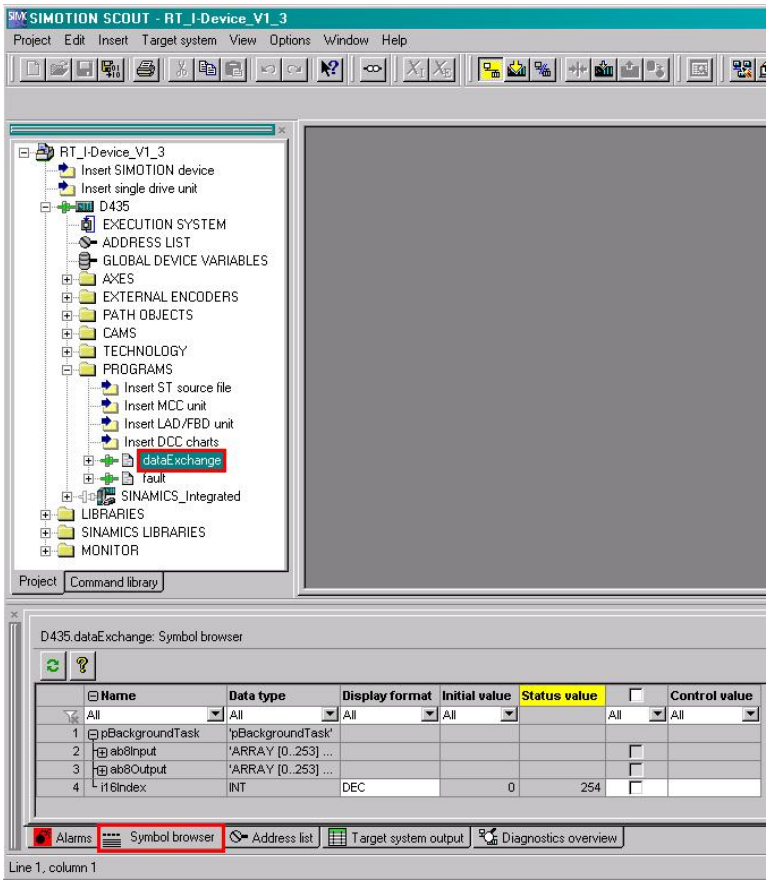
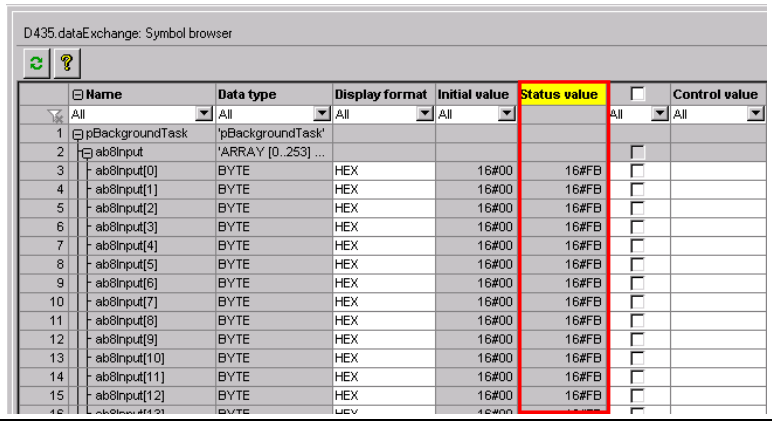
6 Operation of the Application

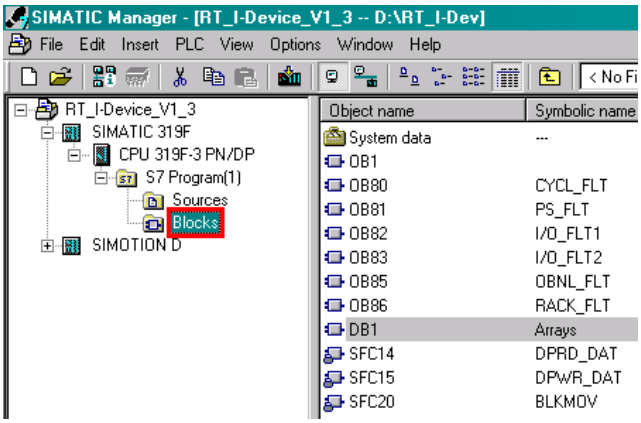
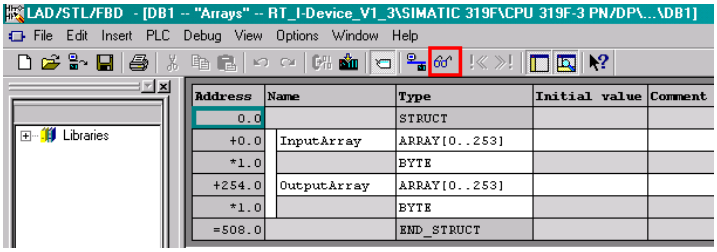
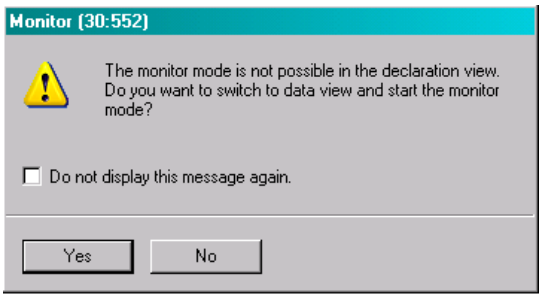
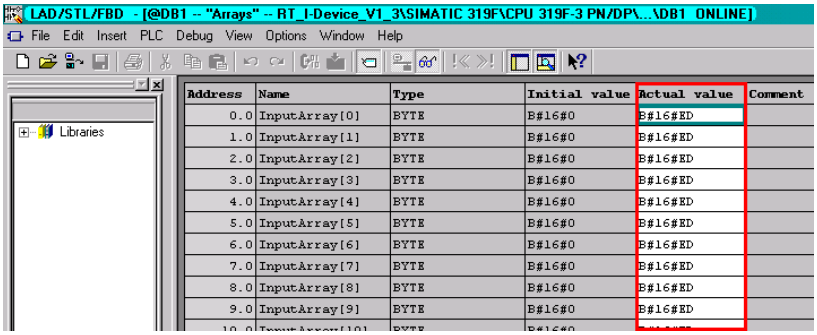
6.1 Overview

The example project is handled via the SIMATIC Manager and the SIMOTION SCOUT engineering system.

Table 6-1

No.	Action
1.	Switch the SIMATIC CPU to the operating state “RUN”.
2.	Switch to the SIMOTION SCOUT project and select the SIMOTION controller. The operating state window is opened using the shortcut key "Ctrl + I". In order to do this, you must establish an online connection with the SIMOTION controller. Switch the SIMOTION controller in the operating state “RUN”, too. 
3.	Data exchange can be monitored in the symbol browser of the SIMOTION SCOUT as well as in data block DB1 of the STEP7 project. Please note that both controllers must be in the operating state “RUN”, in order to copy the input data to the output data!

No.	Action
4.	Select the ST unit "dataExchange" in the SIMOTION SCOUT project and switch to the "Symbol browser" tab in the detail window. 
5.	In the column "Status value", the actual values for the elements of the "ab8Input" array and "ab8Output" array are displayed. 

No.	Action
6.	Switch to the "Blocks" folder in the STEP7 project. 
7.	Open data block DB1 with a double-click.
8.	Perform a click on the symbol for monitoring the block. 
9.	Confirm the following message with "Yes". 
10.	The current values for the elements of the data block are displayed in the "Actual value" column. 

7 Further Notes, Tips and Tricks, etc.

7.1 Multiple use of an I-Device GSD file

Using an I-Device GSD file, several I-Device substitutes, with the same GSD file, can be connected to the higher-level I/O controller.

This means that when using a GSD file several times, each associated I-Device substitute on the I-Device side uses the same I/O addresses in order to communicate with the higher-level I/O controller. It goes without saying that at the higher-level I/O controller, the I/O addresses at the I-Device substitutes are different.

The multiple use of an I-Device GSD file is used to implement modular machine concepts and offers the following advantages:

1. On one hand, the individual machine modules (I-Device) internally operate with the same I/O addresses.
2. On the other hand, a machine module (I-Device) can be activated and deactivated as required (e.g. for partial commissioning purposes).

To do this, each machine module requires its own controller (e.g. SIMOTION) and the complete machine, a higher-level controller (e.g. SIMATIC).

Note

Additional information on modular machine concepts is provided in the SIMOTION documentation “Modular_Machine_Concepts.pdf“.

Modular_Machine_Concepts.pdf

7.2 Efficient I/O definitions

I/O variables should always be defined as the largest possible data type, i.e. WORD instead of 2 BYTE, DWORD instead of 4 BYTE or 2 WORD. For large elements, we recommend utilizing the possibility of forming arrays, i.e. one 10-byte array is better than 10 individual I/O bytes. The advantage is the more efficient utilization of copy routines on the system side.

Further, address gaps should be avoided when creating I/O variables in the symbol browser as well as for telegrams for connecting drives. If this is used often, then this can have a significant negative impact on the efficiency of the implicitly called copy routines.

For read and write access to the individual bits in an I/O byte (or word or double word, Section 7-1) at the latest with the second access it is more efficient to access the complete I/O byte, and by using bit masks or corresponding system functions (e.g. the SIMOTION system function **_getBit()**), to evaluate the individual bits (Figure 7-2).

Fig. 7-1

Fig. 7-2

```
1  INTERFACE
2    VAR_GLOBAL
3      gboInput1 : BOOL;
4      gboInput2 : BOOL;
5    END_VAR;
6  END_INTERFACE
7
8  IMPLEMENTATION
9    PROGRAM pGetBits
10     VAR_TEMP
11       b8Input : BYTE; // local data storage of input byte
12     END_VAR;
13
14     // copy input byte to local data storage
15     b8Input := ib8Input;
16
17     // get single bits with bitmask
18     gboInput1 := BYTE_TO_BOOL (b8Input AND 16#01);
19     gboInput2 := BYTE_TO_BOOL (b8Input AND 16#02);
20
21     // get single bits with systemfuntion _getBit()
22     gboInput1 := _getBit(b8Input, 1);
23     gboInput2 := _getBit(b8Input, 2);
24   END_PROGRAM
25 END_IMPLEMENTATION
26
```

Note

The individual bits in the I/O area cannot be assigned to any process image. For this purpose, at least one byte should be defined at the same address, in order to assign this to the appropriate process image.

Different process images for individual bits are not possible.

8 Contact

Siemens AG

Industry Sector
I DT MC PMA APC
Frauenauracher Strasse 80
D - 91056 Erlangen
mailto: profinet.team.motioncontrol.i-dt@siemens.com

9 History

Table 9-1

Version	Date	Change
V1.1	05/2008	First edition
V1.2	12/2010	Revised Edition
V1.3	06/2012	Revised Edition