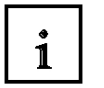
Training Document
for Comprehensive Automation Solutions
Totally Integrated Automation (T I A)

MODULE E09
PROFINET
with
2 x CPU 315F-2 PN/DP

We would like to thank the following: Michael Dziallas Engineering, the teachers at vocational schools, and all others who helped to prepare this document.

PAGE

The following symbols provide a guide through Module E09:

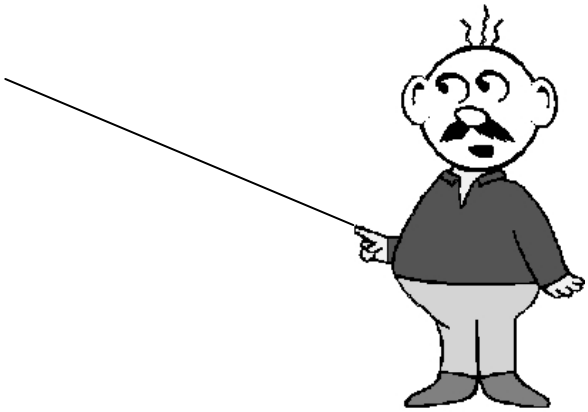


Information

Programming

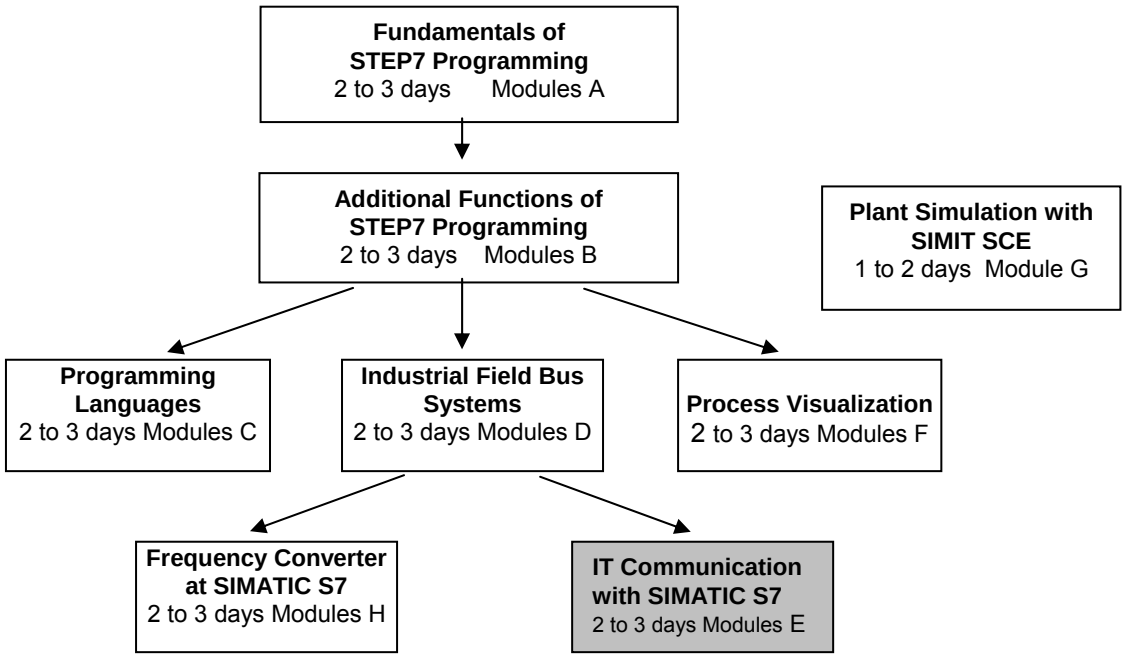
Sample Exercise

Notes



1. PREFACE

In terms of its contents, Module E09 is part of the teaching unit entitled **'IT Communication with SIMATIC S7'**.



Learning Objective:

In Module E09, the reader learns how two PLCs with the CPU 315F-2 PN/DP can communicate with each other via PROFINET. Module E09 shows the method for startup in principle, using a brief example.

Prerequisites:

To successfully work through Module E09, the following knowledge is assumed:

- Knowledge in handling Windows
- Fundamentals of PLC programming with STEP 7 (for example, Module A3 - 'Startup' PLC Programming with STEP 7)
- Fundamentals of network engineering (for example, Appendix V - Basics of Network Engineering)

Hardware and software required

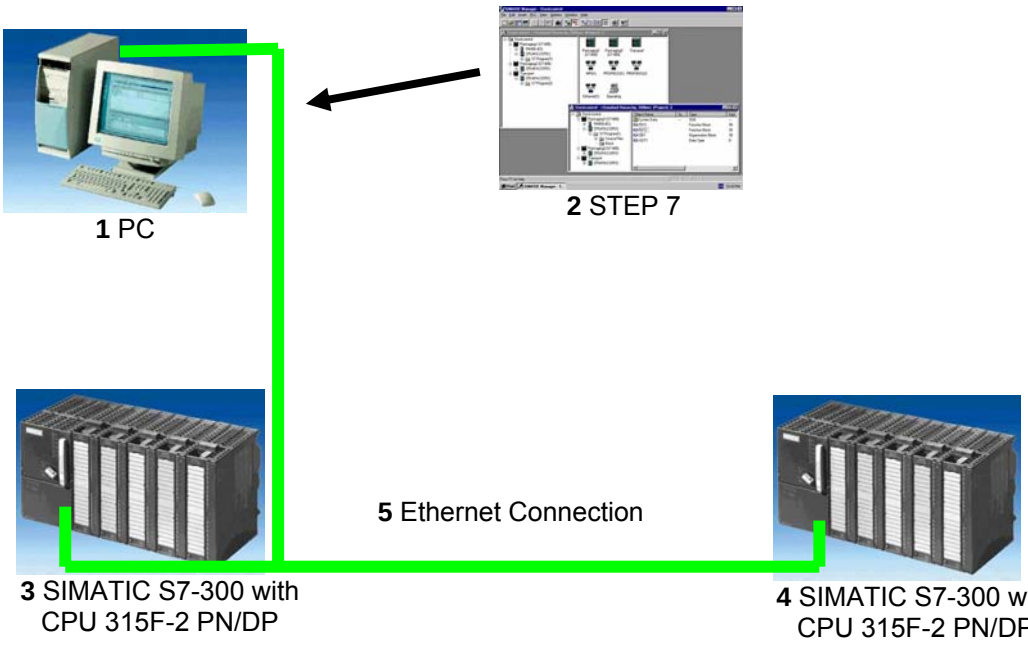
1 PC, operating system Windows 2000 Professional starting with SP4/XP Professional starting with SP1/Server 2003 with 600MHz and 512RAM, free hard disk storage 650 to 900 MB, MS Internet Explorer 6.0 and network card

2 Software STEP7 V 5.4

3 PLC SIMATIC S7-300 with CPU 315F-2 PN/DP and at least one digital input and output module
Sample configuration:
- Power supply: PS 307 2A
- CPU: CPU 315F-2 PN/DP
- Digital inputs: DI 16 x DC 24V
- Digital outputs: DO 16 x DC 24V/0.5A

4 PLC SIMATIC S7-300 with CPU 315F-2 PN/DP and at least one digital input and output module:
Sample configuration:
- Power supply: PS 307 2A
- CPU: CPU 315F-2 PN/DP
- Digital inputs: DI 16 xDC 24V
- Digital outputs: DO 16 x DC 24V/0.5A

5 Ethernet connection between PC and CPUs 315F-2 PN/DP



2. NOTES ON THE USE OF CPU 315F-2 PN/DP

The CPU 315F-2 PN/DP is a CPU that is shipped with 2 integrated interfaces.

- The first interface is a combined MPI/PROFIBUS-DP interface that can be used on the PROFIBUS DP as master or slave for connecting distributed IO/field devices with very fast response timing.
  In addition, the CPU can be programmed here by means of MPI or PROFIBUS DP.
- The second interface is an integrated PROFINET interface.
  It allows for using the CPU as a PROFINET IO controller for operating distributed IO on the PROFINET. Also, the CPU can be programmed by means of this interface!
- Fault tolerant IO devices can also be used on both interfaces.

Notes:

- In Module E09, the CPU 315F-2 PN/DP is used on the PROFINET.
- To operate this CPU, a micro-memory card is required!
- The addresses of the input and output modules can be parameterized at this CPU.

3. STARTUP OF THE PROFINET (2 X CPU 315F-2 PN/DP)

Below, the startup of a PROFINET network with 2 x CPU 315F-2 PN/DP is described.
To test the configuration, a program is written in which an input byte (SET) can be specified at the PLC in Station 1. Via PROFINET, this byte is transmitted to the other PLC in Station2, and can be read out there at a display byte (DISPLAY).

Assignment List Station1:

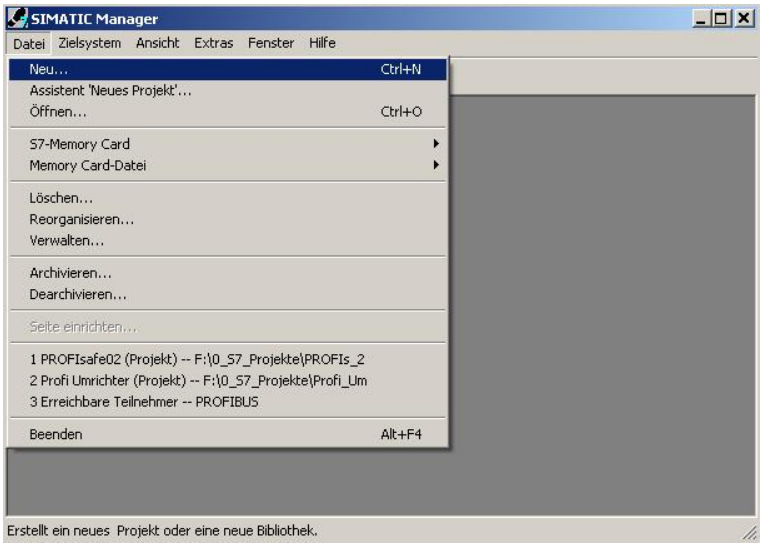
I1.0	REQ	Activation of the send request (with positive edge)
I1.1	R	Cancelling the send request (with positive edge)
IB 0	SET	Input byte
MB 40	Comm_A1	Output communication send buffer Byte1

Assignment List Station2:

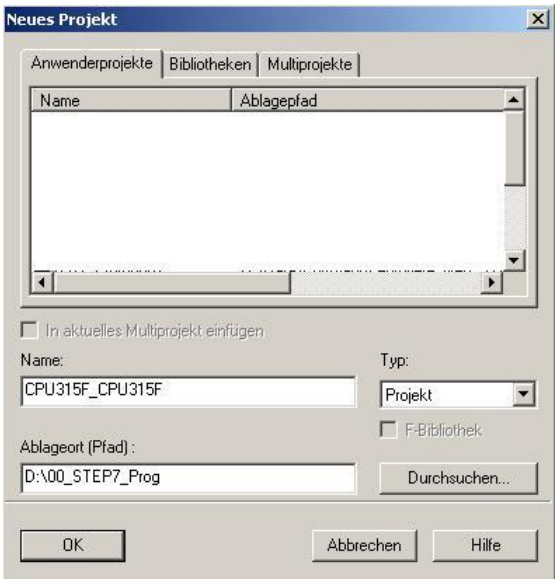
I1.0	EN_R	Activation of Ready to receive (if EN_R == 1)
MB 50	Comm_E1	Input communication receive buffer Byte1
QB 4	DISPLAY	Display byte

1. The central tool in STEP 7 is the **'SIMATIC Manager'**. It is called here with a double click. (→ SIMATIC Manager)

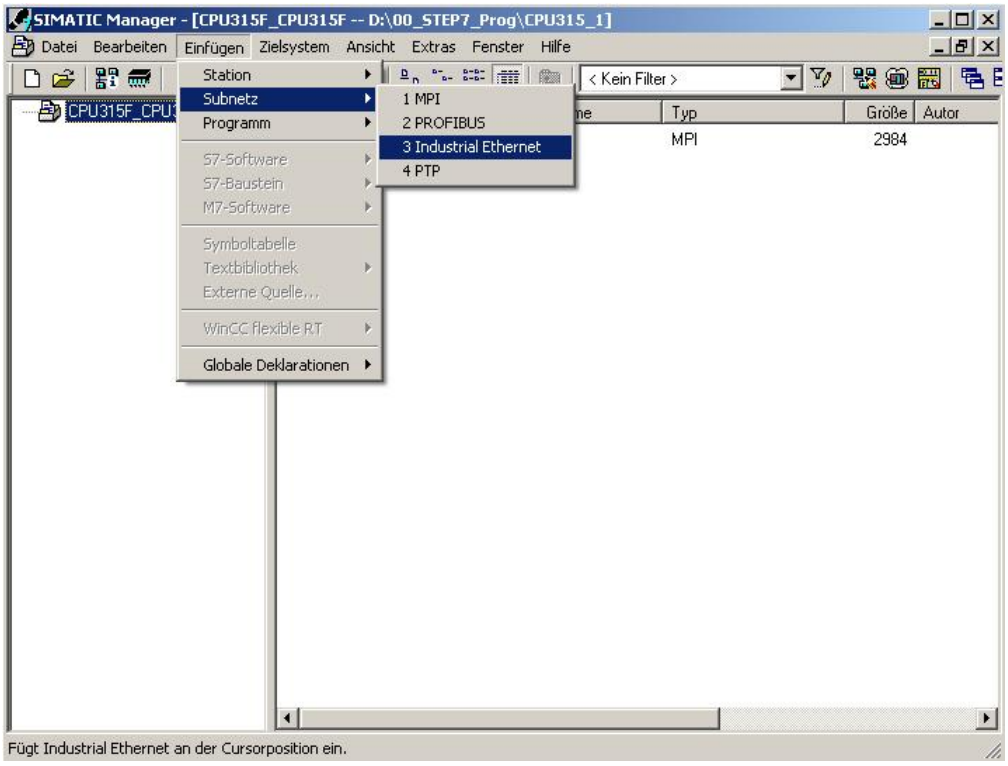
2. STEP7 programs are managed in projects. We are now setting up such a project. (→ File → New)



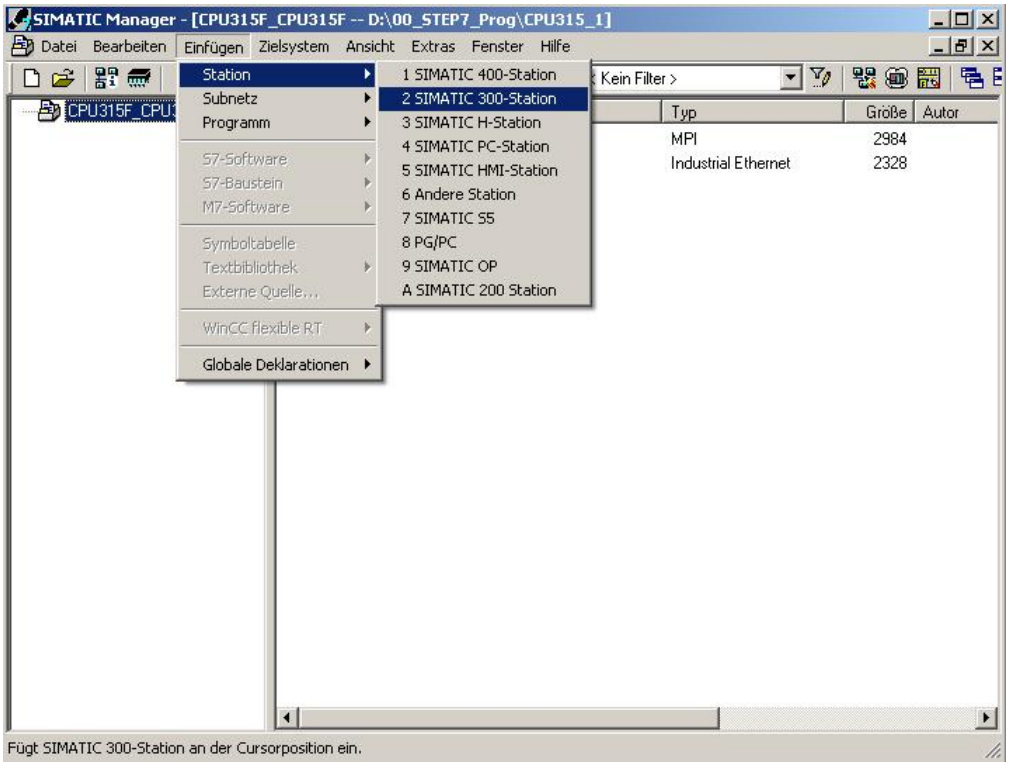
3. The project is now assigned the **'Name' 'CPU315F_CPU315F'** (→ CPU315F_CPU315F → OK)



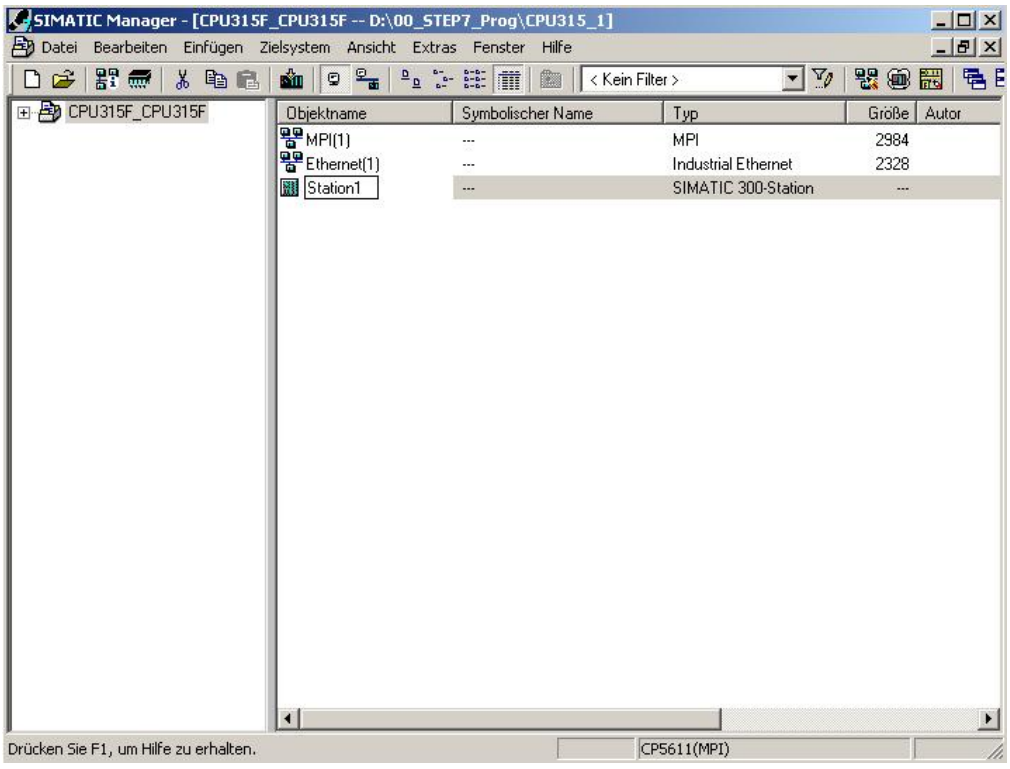
4. Highlight your project and insert an **'Industrial Ethernet Subnet'** (→ CPU315F_CPU315F → Insert → Subnet → Industrial Ethernet).



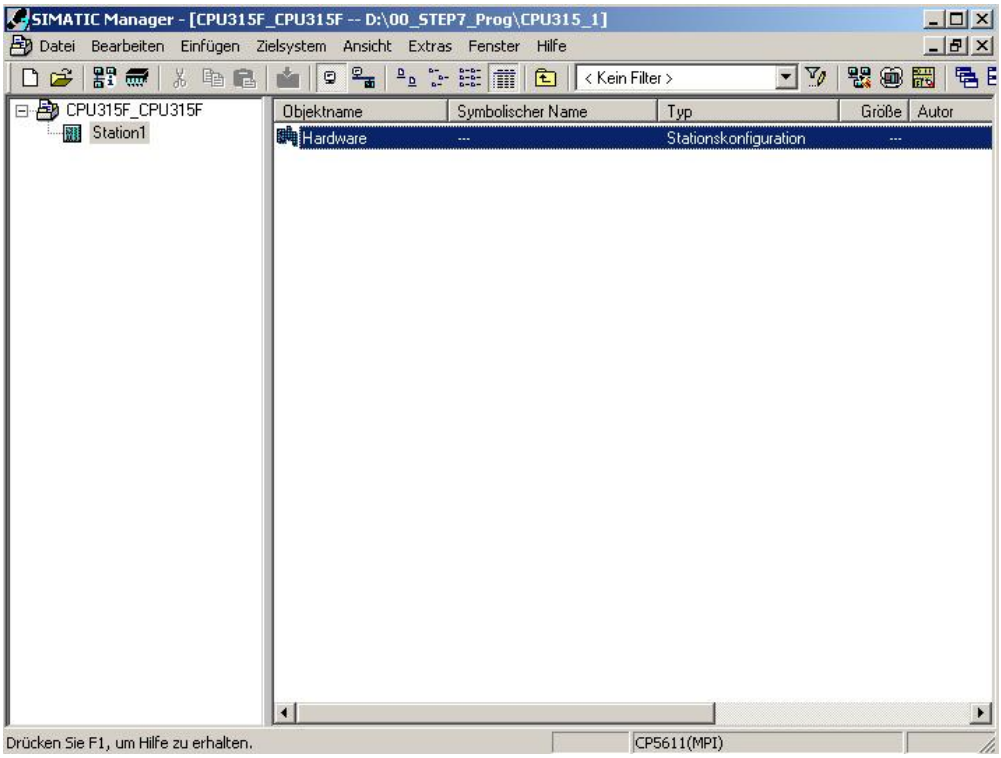
5. Then, a **'SIMATIC 300 Station'** is inserted. (→ Insert → Station → SIMATIC 300 Station)



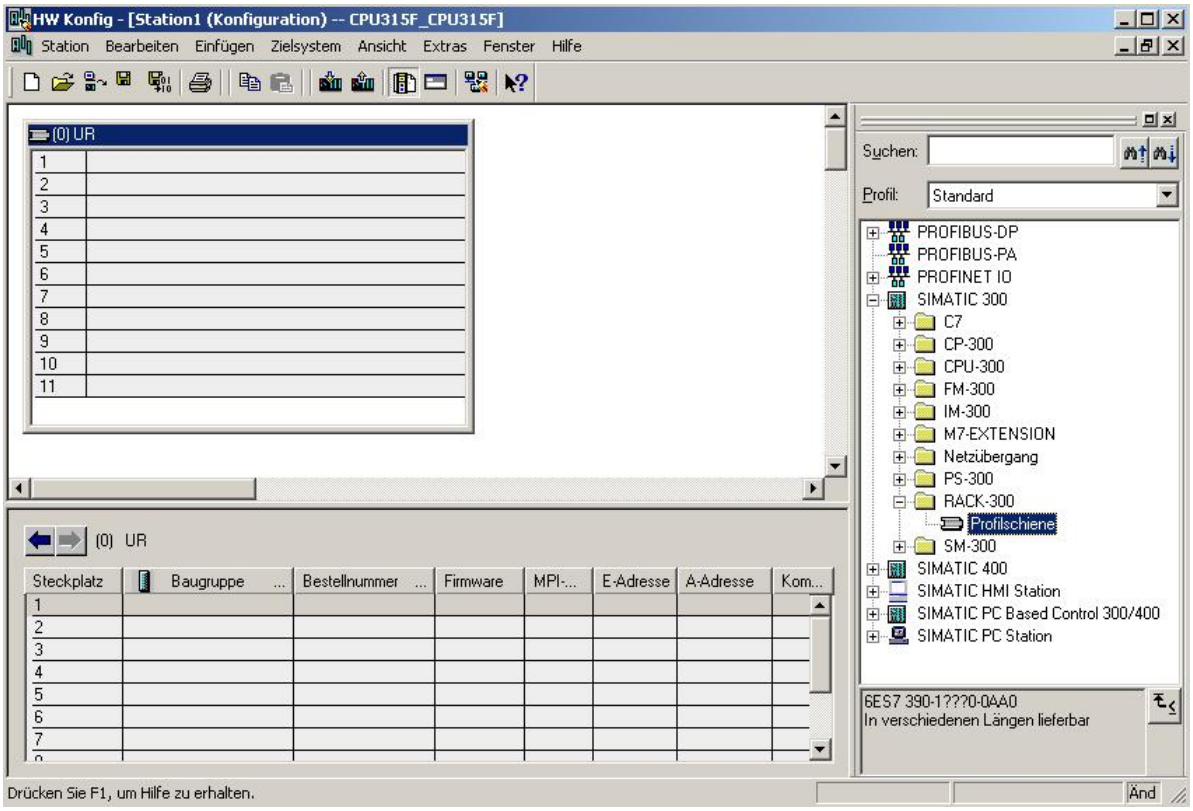
6. Change the name of the station to **'Station1'**. (→ Station1)



7. With a double click, open the configuration tool for the **'Hardware'**. (→ Hardware)

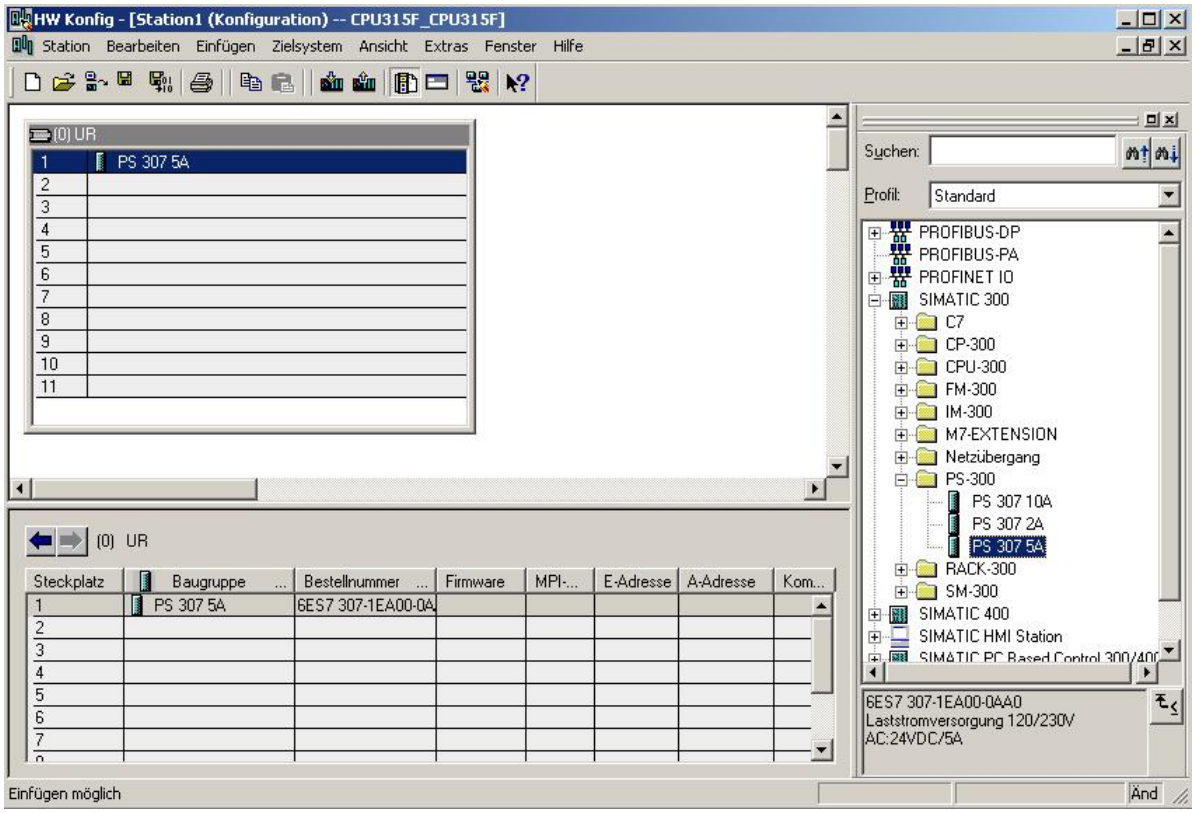


8. Open the hardware catalog by clicking on '  '. (→ )
   There, arranged in the following directories:
   PROFIBUS DP, PROFIBUS PA, PROFINET IO, SIMATIC 300, SIMATIC 400,
   SIMATIC PC Based Control, and SIMATIC PC Station,
   all racks, modules and interface modules are provided for configuring your hardware.
   Insert **'Rail'** with a double click. (→ SIMATIC 300 → RACK 300 → Rail)



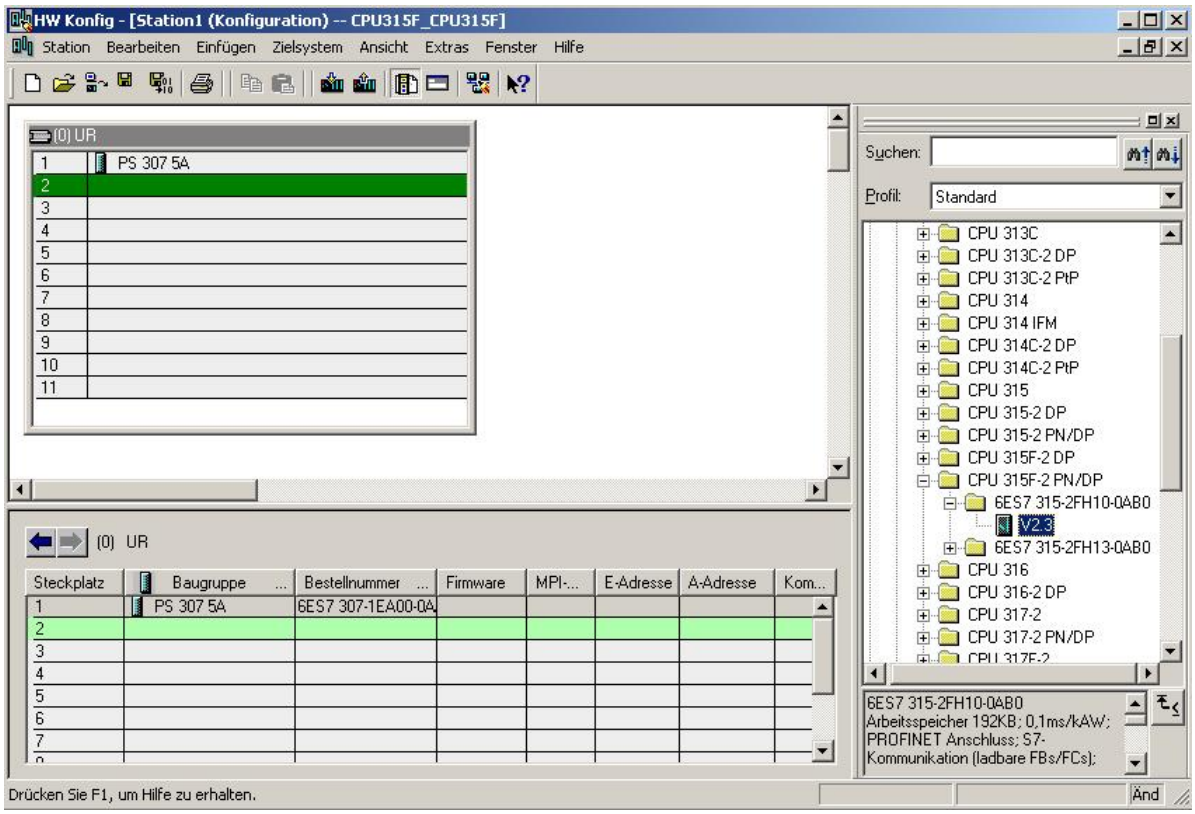
Note: After that, a configuration table is displayed automatically for configuring Rack 0.

9. From the hardware catalog, you can now select all modules that are also in your real rack, and insert them in the configuration table.
To this end, click on the name of the respective module, hold the mouse key and drag the module to a line in the configuration table.
We are starting with the power unit **'PS 307 2A'**. (→ SIMATIC 300 → PS-300 → PS 307 5A)

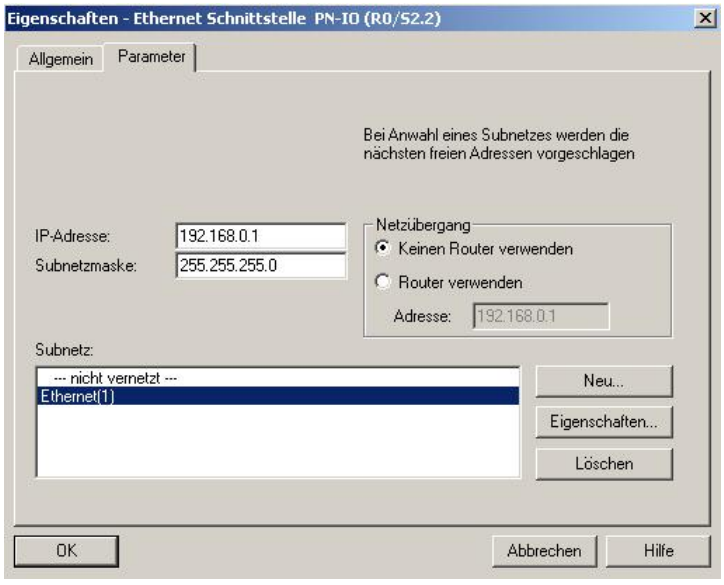


Note: If your hardware differs from the one displayed here, simply select the corresponding modules from the catalog and insert them in your rack. The order numbers of the individual modules -that are also indicated on the components- are displayed in the footer of the catalog.

10. Next, we are dragging the **'CPU 315F-2 PN/DP'** to the second slot.
The order number and the version of the CPU can be read off the front of the CPU.
(→ SIMATIC 300 → CPU-300 → CPU 315F-2 PN/DP → 6ES7 315-2FH10-0AB0 → V2.3)



11. When entering the CPU, the window below appears. In this window, do the following:
Assign an **'IP Address'** to the CPU 315F-2 PN/DP, specify the **'Subnet Screen Form'**, and select the **'Ethernet'** network that has already been generated.
Optional: a **'Router Address'** can also be selected for network-overreaching communication.
Confirm your input with **'OK'** (→ IP Address: 192.168.0.1 → Subnet Screen Form: 255.255.255.0 → Ethernet(1) → Don't use a router → OK)



Notes on Networking on the Ethernet (additional information is provided in Appendix V of the training document):

MAC Address:
The MAC address consists of a permanent and a variable part. The permanent part ("Basis MAC Address") identifies the manufacturer (Siemens, 3COM, ...). The variable part of the MAC address differentiates the different Ethernet stations, and should be assigned uniquely world-wide. On each module, a MAC address specified by the factory is imprinted.

Value range for the IP address:
The IP address consists of 4 decimal numbers from the value range 0 to 255 which are separated by a period; for example 141.80.0.16

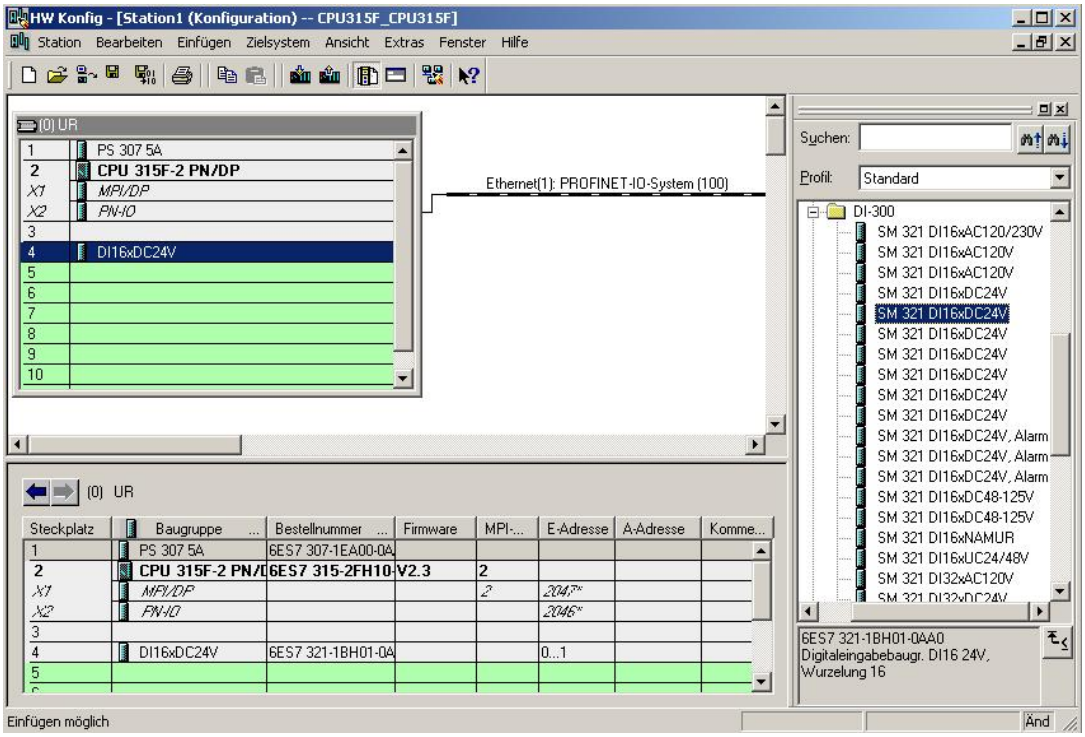
Value range for the subnet screen form:
This screen form is used in order to recognize whether a station or its IP address is part of the local subnet, or can be accessed only by means of a router.
The subnet screen form consists of 4 decimal numbers from the value range 0 to 255 which are separated by a period; for example, 255.255.0.0
In their binary representation, the 4 decimal numbers of the subnet screen form have to contain from the left a series of gapless values "1" and from the right a series of gapless values "0".
The values "1" determine the area of the IP address for the network number. The values "0" determine the area of the IP address for the station address.
Example:

Correct values:	255.255.0.0 decimal = 1111 1111.1111 1111.0000 0000.0000 0000 binary
	255.255.128.0 decimal = 1111 1111.1111 1111.1000 0000.0000 0000 binary
	255.254.0.0 decimal = 1111 1111.1111 1110.0000 0000.0000.0000 binary
Incorrect value:	255.255.1.0 decimal = 1111 1111.1111 1111.0000 0001.0000 0000 binary

Value range for the address of the network transition (Router):
The address consists of 4 decimal numbers from the value range 0 to 255 which are separated by a period; for example, 141.80.0.1.

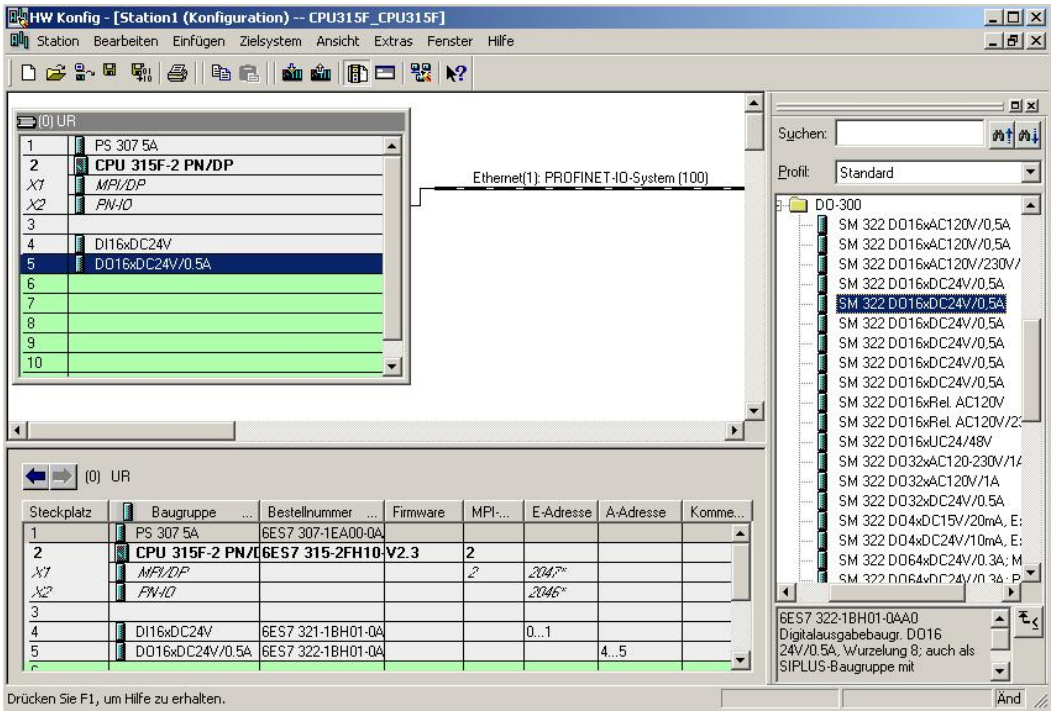
Relationship of IP addresses, router address, and subnet screen form:
The IP address and the address of the network transition may differ only in positions that have a "0" in the subnet screen form.
Example:
You entered: for the subnet screen form 255.255.255.0; for the IP address 141.30.0.5, and for the router address 141.30.128.1.
The IP address and the address for the network transition are to have a different value only in the 4th decimal number. In the example, however, the 3rd position already differs.
In the example, you have to change alternatively:
- the subnet screen form to: 255.255.0.0 or
- the IP address to: 141.30.128.5 or
- the address of the network transition to: 141.30.0.1

12. Next, we are dragging the input submodule for 16 inputs to the 4th slot. The order number of the submodule can be read off the front. (→ SIMATIC 300 → DI-300 → SM 321 DI16xDC24V)



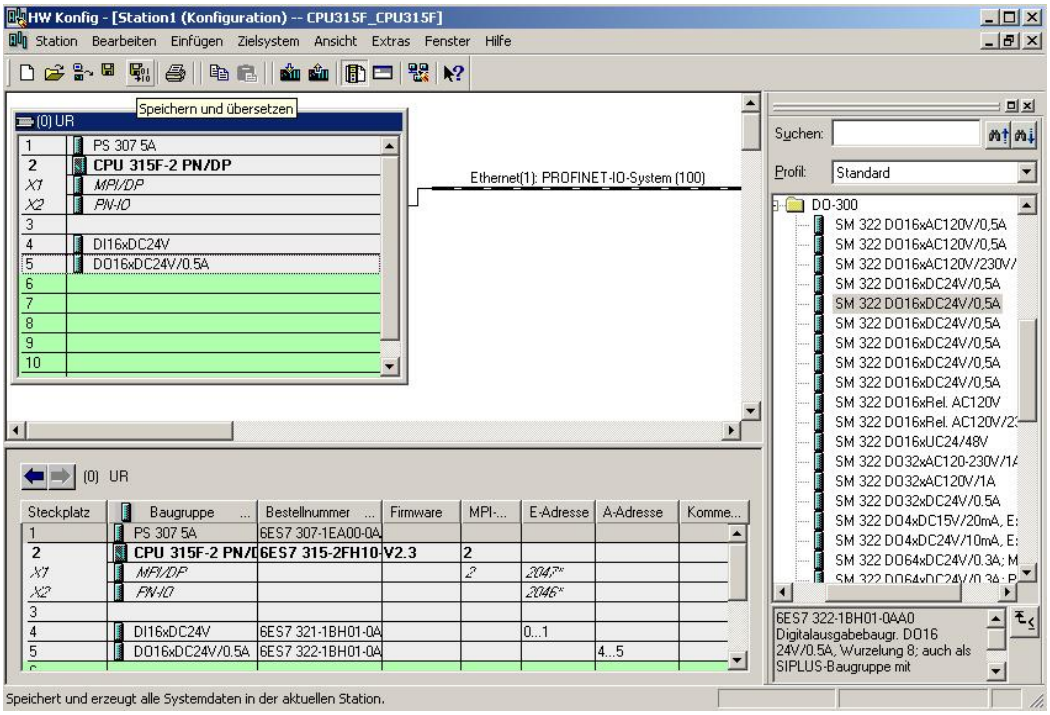
Note: Slot 3 is reserved for the interface modules and remains empty for that reason. The order number of the module is shown in the footer of the catalog.

13. Now we are dragging the output submodule for 16 outputs to the 5th slot. The order number of the submodule can be read off the front. (→ SIMATIC-300 → DO-300 → SM 322 DO16xDC24V/0.5A)

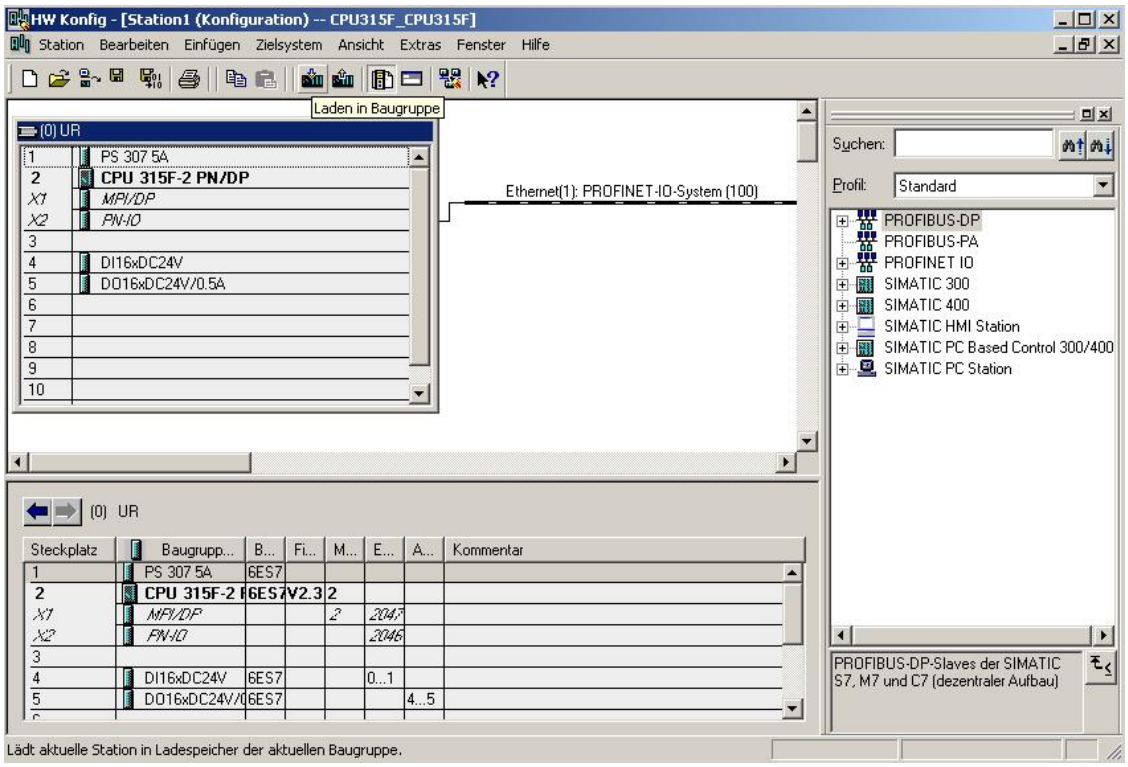


Note: The order number of the module is shown in the footer of the catalog.

14. By clicking on '  ', the configuration table is stored and compiled. (→ )



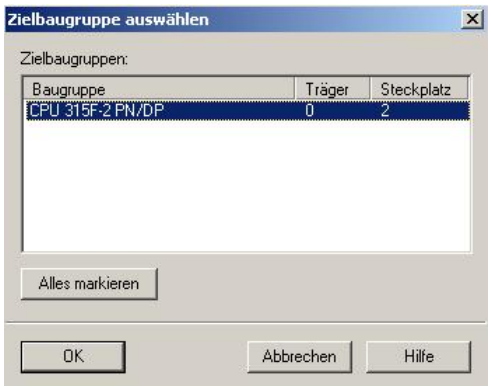
15. Then, the configuration table can be loaded to the PLC by clicking on '  '. The CPU's operating mode switch should be on Stop! (→ )



Note: A precondition for this is that the PG/PC interface is set to TCP/IP and the network card for the PC is configured correctly; for example: IP address 192.168.0.99, Subnet 255.255.255.0 and router address -.-.-.- (refer to module E02!)

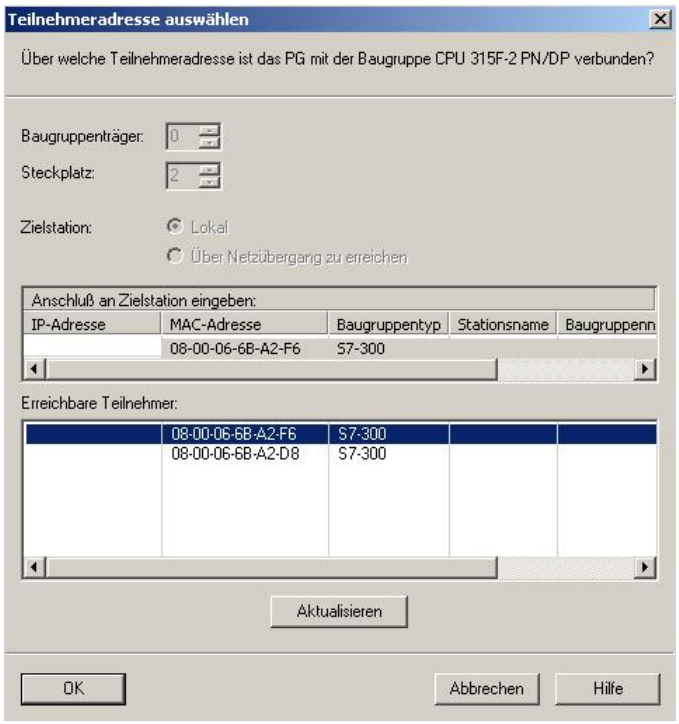
Note: Make sure that your programming device is connected to the CPU by means of the Ethernet!

16. The CPU 315F-2 PN/DP is confirmed as the destination module for loading the configuration. (→ OK)



17. In the dialog box below, you can have the devices on the network displayed **'Display'**. (→ display)

18. Then, the MAC address of the CPU on the Ethernet is selected. If you are connected to only one CPU, simply accept with **'OK'**. (→ OK)

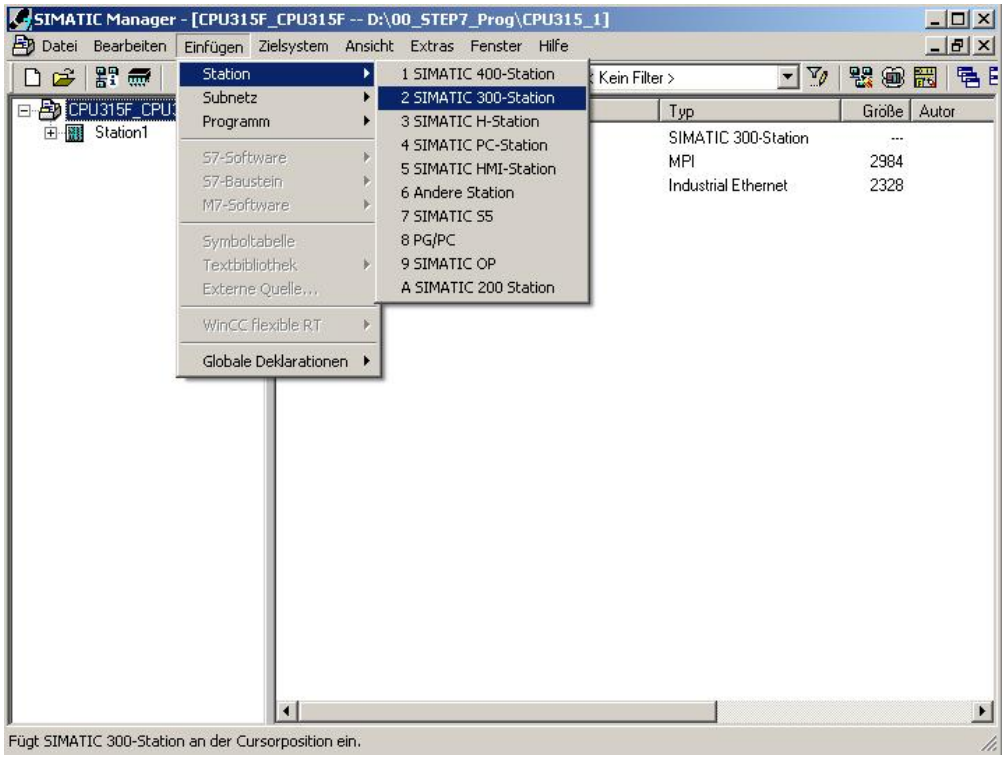


Note: If several IO controllers are on the network, the device can be identified with the MAC address imprinted on it.

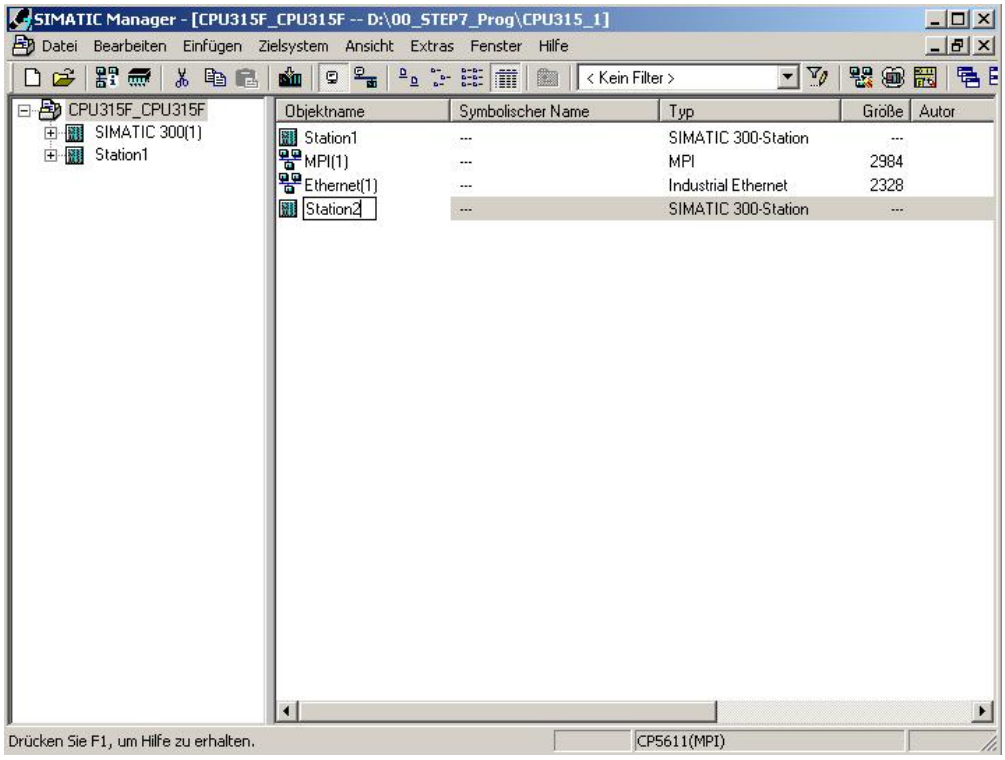
19. Now, the correct IP address has to be assigned to the IO controller if it has not been set correctly previously. Confirm this in the following dialog with **'Yes'**. (→ Yes)

<<The selected station does not have an IP address. Do you want to assign the address 192.168.0.1. now?>>

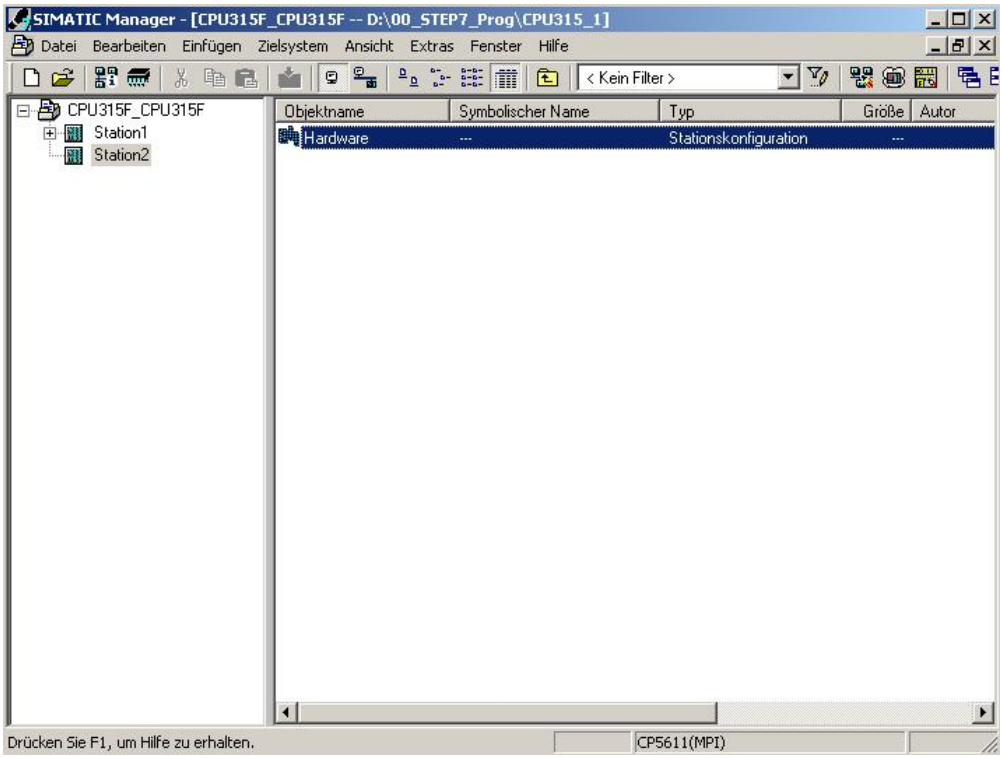
20. Then, a second **'SIMATIC 300 Station'** is inserted. (→ Insert → Station → SIMATIC 300 Station)



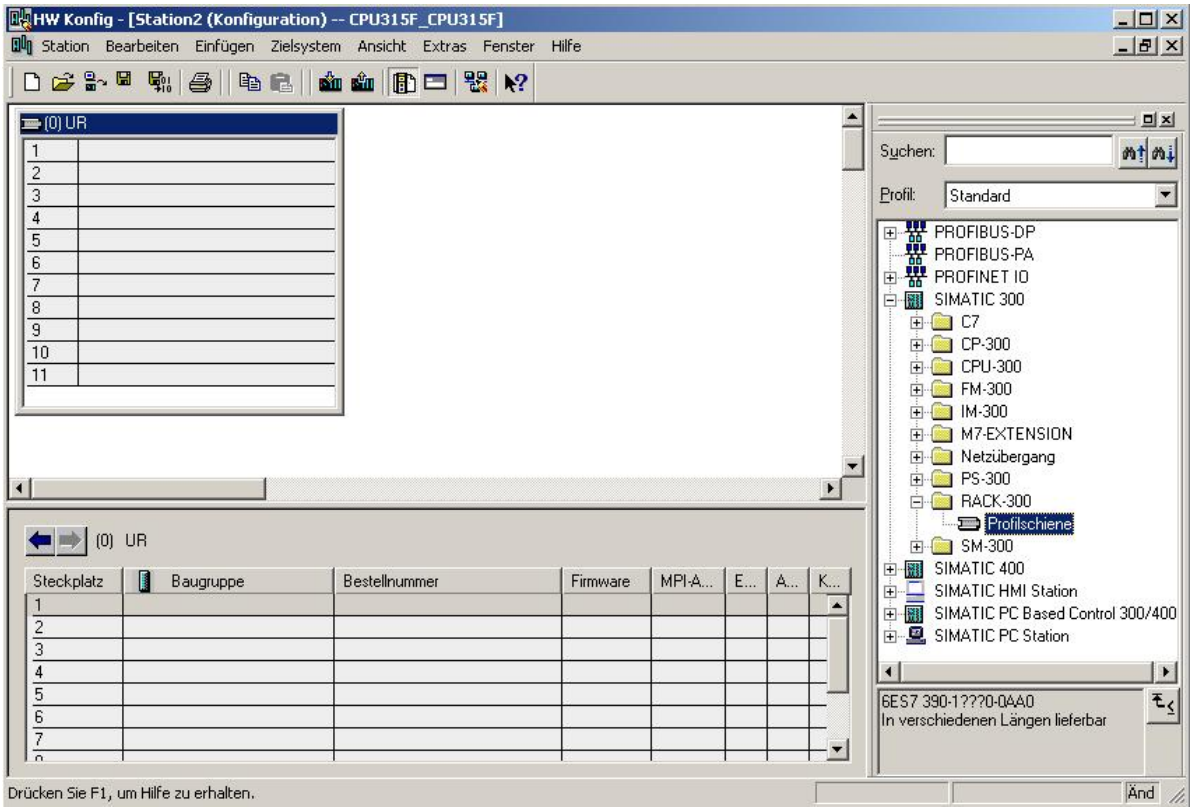
21. Change the name of the station to **'Station2'**. (→ Station2)



22. With a double click, open the configuration tool for the **'Hardware'**. (→ Hardware)

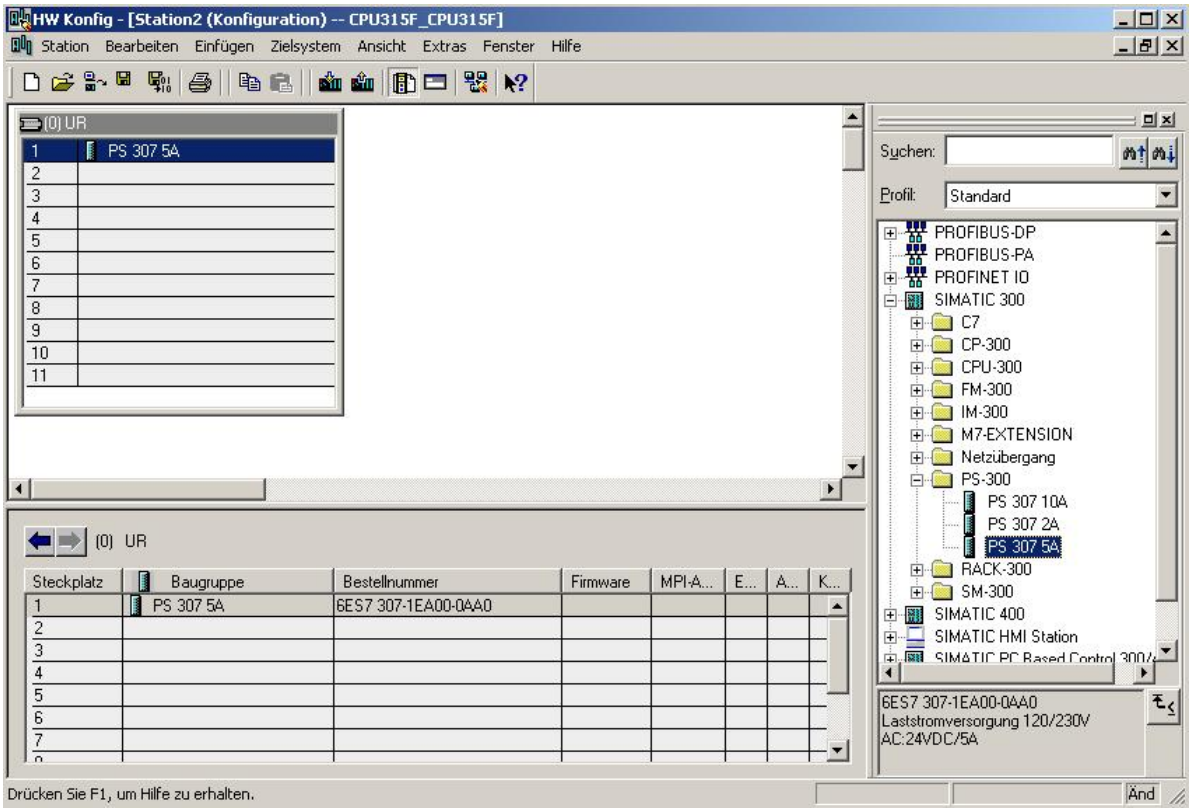


23. Open the hardware catalog by clicking on the symbol '  '. (→ )
    There, arranged in the following directories:
    PROFIBUS DP, PROFIBUS PA, PROFINET IO, SIMATIC 300, SIMATIC 400,
    SIMATIC PC Based Control, and SIMATIC PC Station,
    all racks, modules and interface modules are provided for configuring your hardware.
    Insert **'Rail'** with a double click. (→ SIMATIC 300 → RACK 300 → Rail)



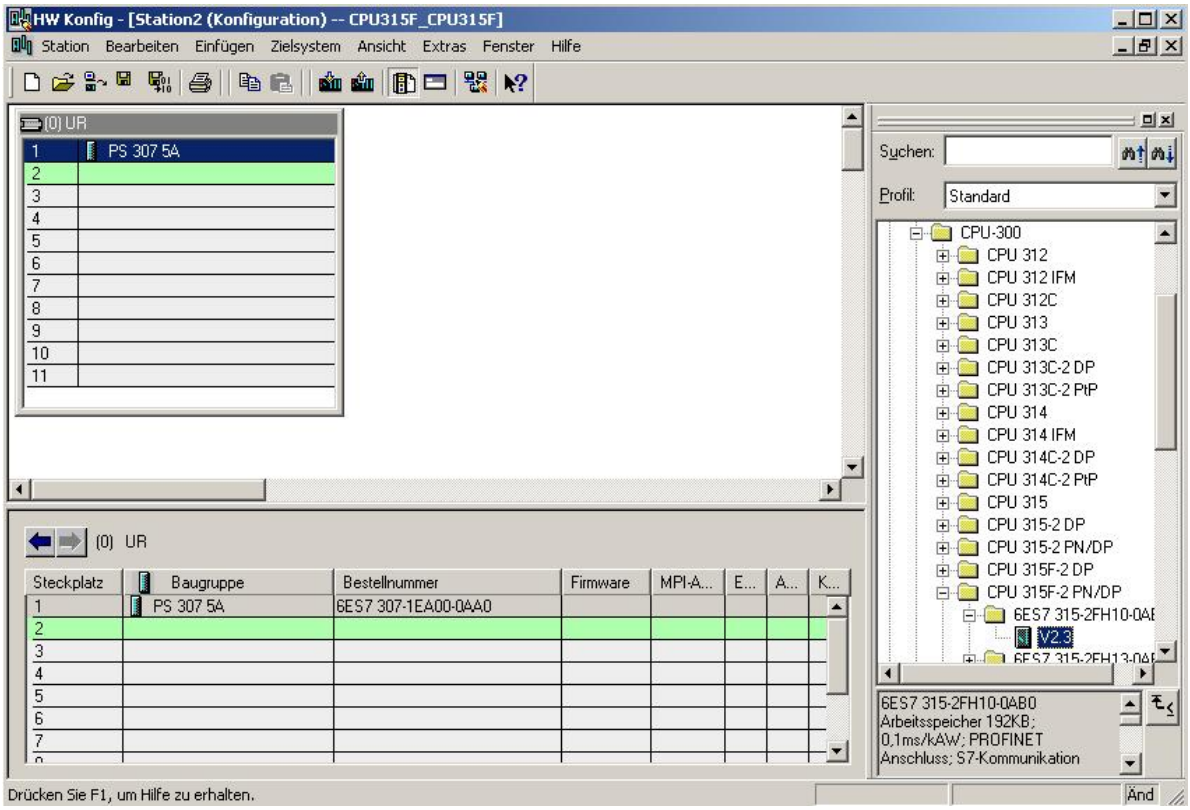
Note: After that, a configuration table for configuring Rack 0 is displayed automatically.

24. From the hardware catalog, you can now select all modules that are also in your real rack, and insert them in the configuration table.
    To this end, click on the name of the respective module, hold the mouse key and drag the module to a line in the configuration table.
    We are starting with the power unit **'PS 307 2A'**. (→ SIMATIC 300 → PS-300 → PS 307 5A)

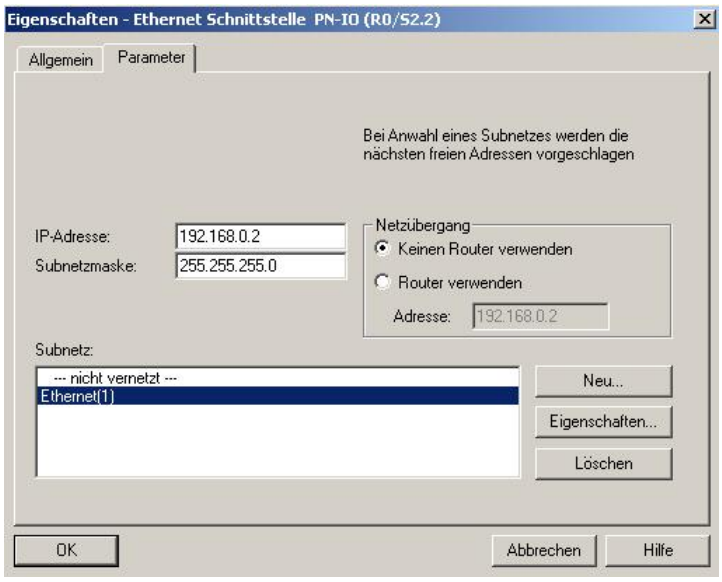


Note: If your hardware differs from the one displayed here, simply select the corresponding modules from the catalog and insert them in your rack. The order numbers of the individual modules -that are also indicated on the components- are displayed in the footer of the catalog.

25. Next, we are dragging the **'CPU 315F-2 PN/DP'** to the 2nd slot. The order number and the version of the CPU can be read off the front of the CPU.
(→ SIMATIC 300 → CPU-300 → CPU 315F-2 PN/DP → 6ES7 315-2FH10-0AB0 → V2.3)



26. When entering the CPU, the window below appears. In this window, do the following:
Assign an **'IP Address'** to the CPU 315F-2 PN/DP, specify the **'Subnet Screen Form'**, and select the **'Ethernet'** network that has already been generated.
Optional: a **'Router Address'** can also be selected for network-overreaching communication.
Confirm your input with **'OK'** (→ IP- Address: 192.168.0.2 → Subnet screen form: 255.255.255.0 → Ethernet(1) → Don't use a router → OK)



Notes on Networking on the Ethernet (additional information is provided in Appendix V of the training document):

MAC Address:
The MAC address consists of a permanent and a variable part. The permanent part ("Basis MAC Address") identifies the manufacturer (Siemens, 3COM, ...). The variable part of the MAC address differentiates the different Ethernet stations, and should be assigned uniquely world-wide. On each module, a MAC address specified by the factory is imprinted.

Value range for the IP address:
The IP address consists of 4 decimal numbers from the value range 0 to 255 which are separated by a period; for example 141.80.0.16

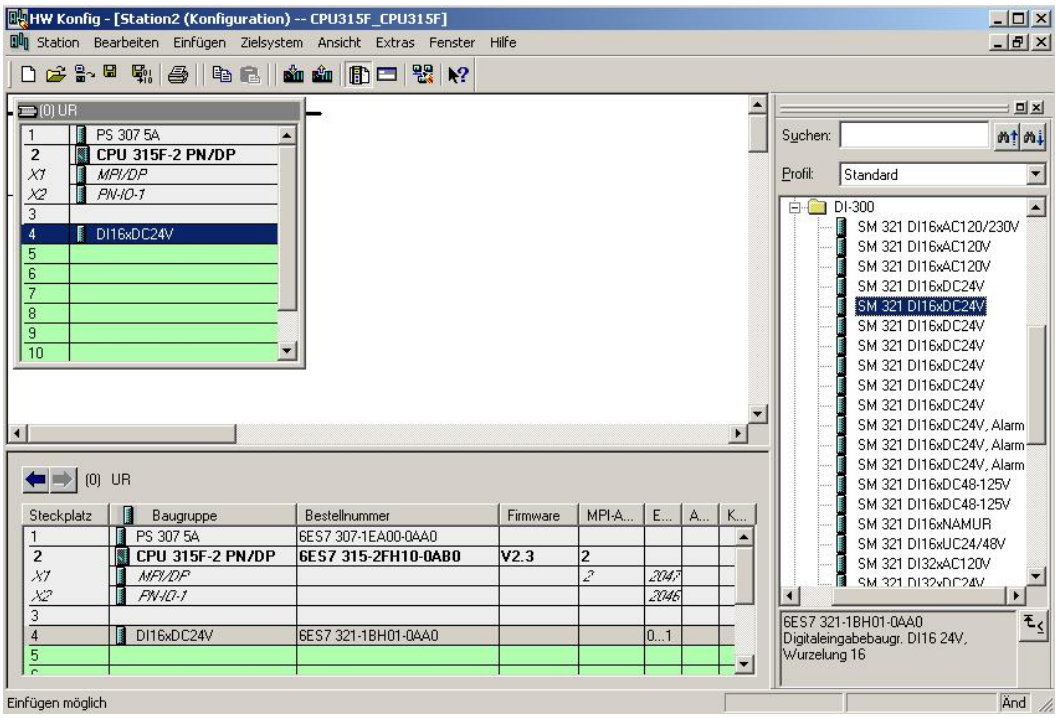
Value range for the subnet screen form:
This screen form is used in order to recognize whether a station or its IP address is part of the local subnet, or can be accessed only by means of a router.
The subnet screen form consists of 4 decimal numbers from the value range 0 to 255 which are separated by a period; for example, 255.255.0.0
In their binary representation, the 4 decimal numbers of the subnet screen form have to contain from the left a series of gapless values "1" and from the right a series of gapless values "0".
The values "1" determine the area of the IP address for the network number. The values "0" determine the area of the IP address for the station address.
Example:

Correct values:	255.255.0.0 decimal = 1111 1111.1111 1111.0000 0000.0000 0000 binary
	255.255.128.0 decimal = 1111 1111.1111 1111.1000 0000.0000 0000 binary
	255.254.0.0 decimal = 1111 1111.1111 1110.0000 0000.0000.0000 binary
Incorrect value:	255.255.1.0 decimal = 1111 1111.1111 1111.0000 0001.0000 0000 binary

Value range for the address of the network transition (Router):
The address consists of 4 decimal numbers from the value range 0 to 255 which are separated by a period; for example, 141.80.0.1.

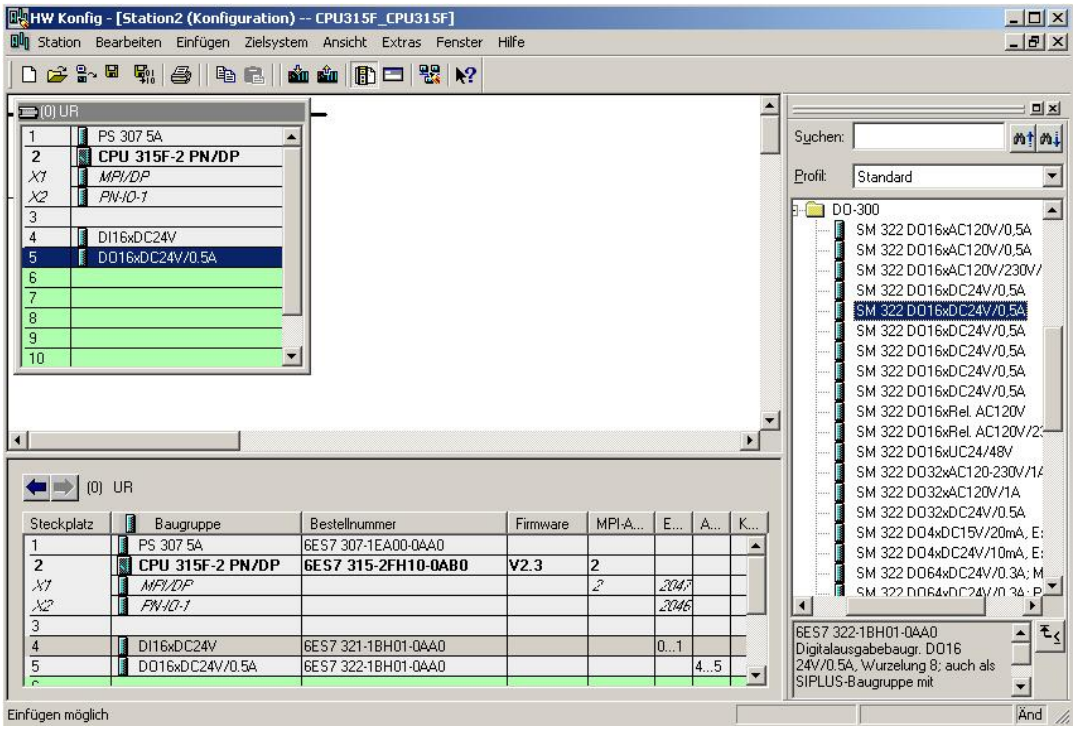
Relationship of IP addresses, router address, and subnet screen form:
The IP address and the address of the network transition may differ only in positions that have a "0" in the subnet screen form.
Example:
You entered: for the subnet screen form 255.255.255.0; for the IP address 141.30.0.5, and for the router address 141.30.128.1.
The IP address and the address for the network transition are to have a different value only in the 4th decimal number. In the example, however, the 3rd position already differs.
In the example, you have to change alternatively:
- the subnet screen form to: 255.255.0.0 or
- the IP address to: 141.30.128.5 or
- the address of the network transition to: 141.30.0.1

27. Next, we drag the input submodule for 16 inputs to the 4th slot. The order number of the submodule can be read off the front. (→ SIMATIC 300 → DI-300 → SM 321 DI16xDC24V)



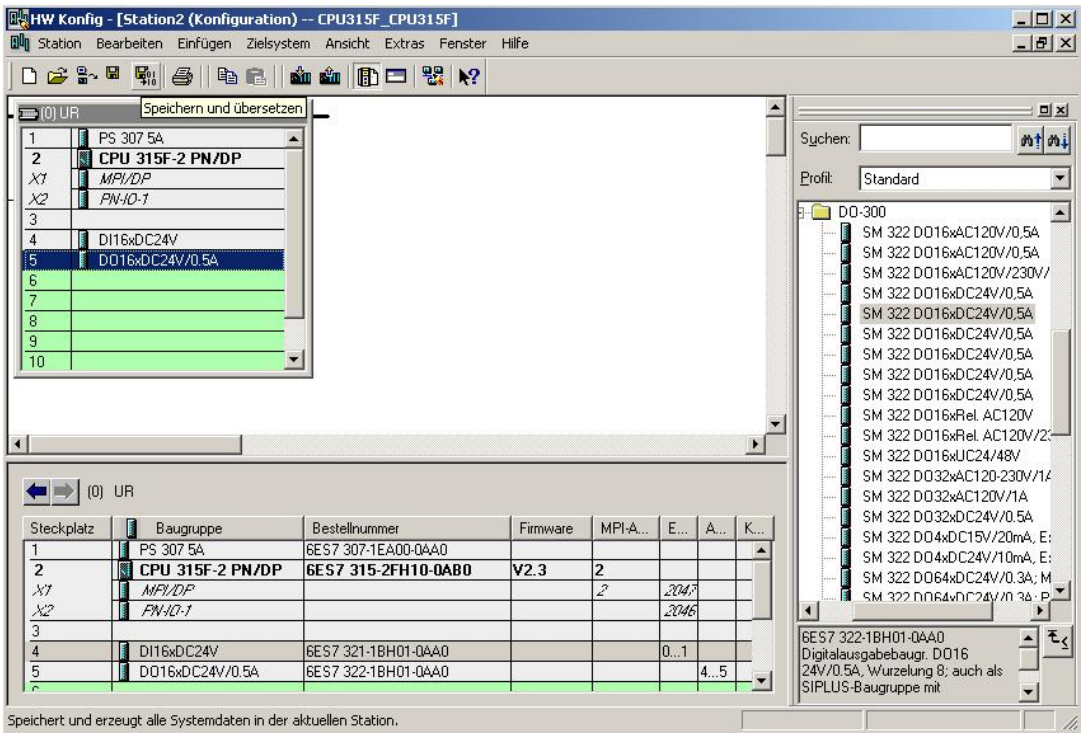
Note Slot 3 is reserved for interface modules and remains empty for that reason. The order number of the module is shown in the footer of the catalog.

28. Next, we are dragging the output submodule for the 16 outputs to the 5th slot.
The order number of the submodule can be read off the front.
(→ SIMATIC-300 → DO-300 → SM 322 DO16xDC24V/0.5A)

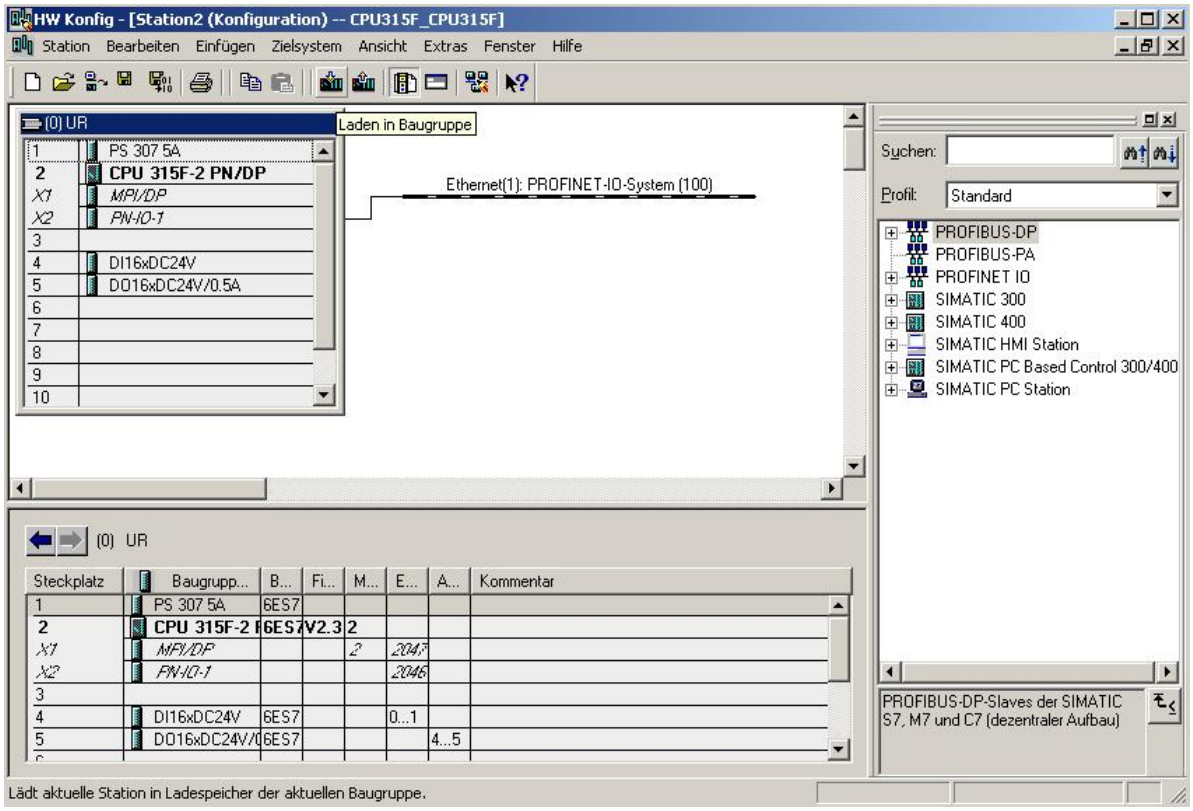


Note: The order number of the module is displayed in the footer of the catalog.

29. By clicking on '  ', the configuration table is saved and compiled. (→ )



30. By clicking on '  ', the configuration table can now be loaded to the PLC. The CPU's operating mode switch should be on Stop! (→ )

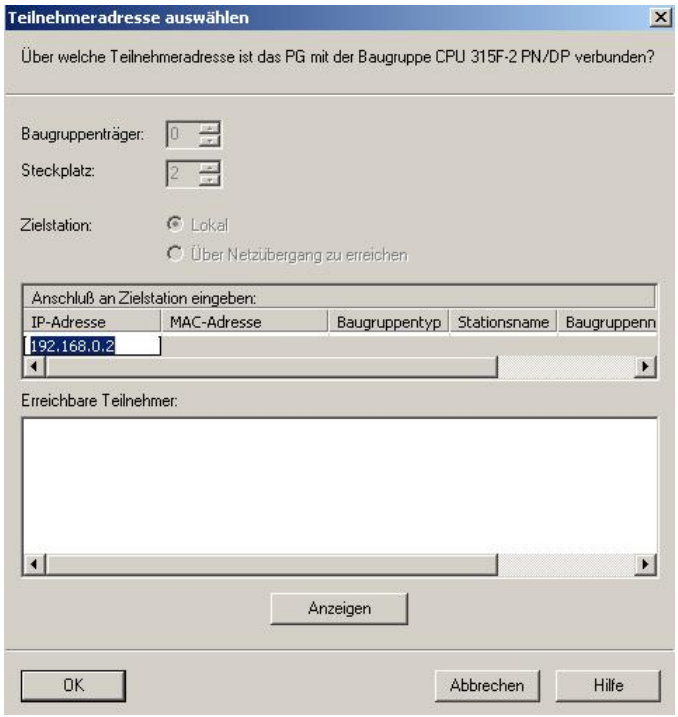


Note: A precondition for this is that the PG/PC interface is set to TCP/IP and the network card for the PC is configured correctly; for example: IP address 192.168.0.99, Subnet 255.255.255.0 and router address -.-.-.- (refer to module E02!)

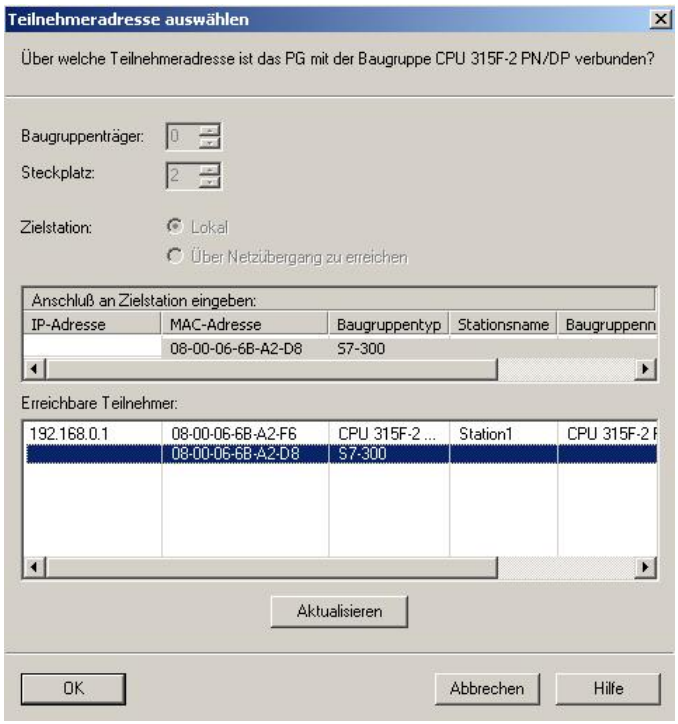
Note: Make sure that your programming device is connected to the CPU by means of the Ethernet!

31. The CPU 315F-2 PN/DP is confirmed as destination module for loading the configuration table. (→ OK)

32. In the following table, you can have the devices displayed that are connected to the network **'Display'**. (→ Display)

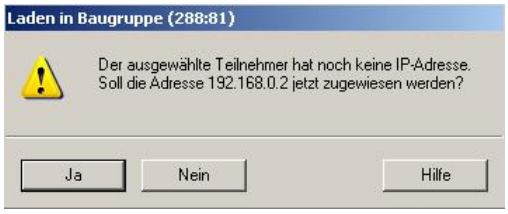


33. Then, the MAC address of the CPU on the Ethernet is selected. If you are connected to only one CPU, simply accept with **'OK'**. (→ OK)



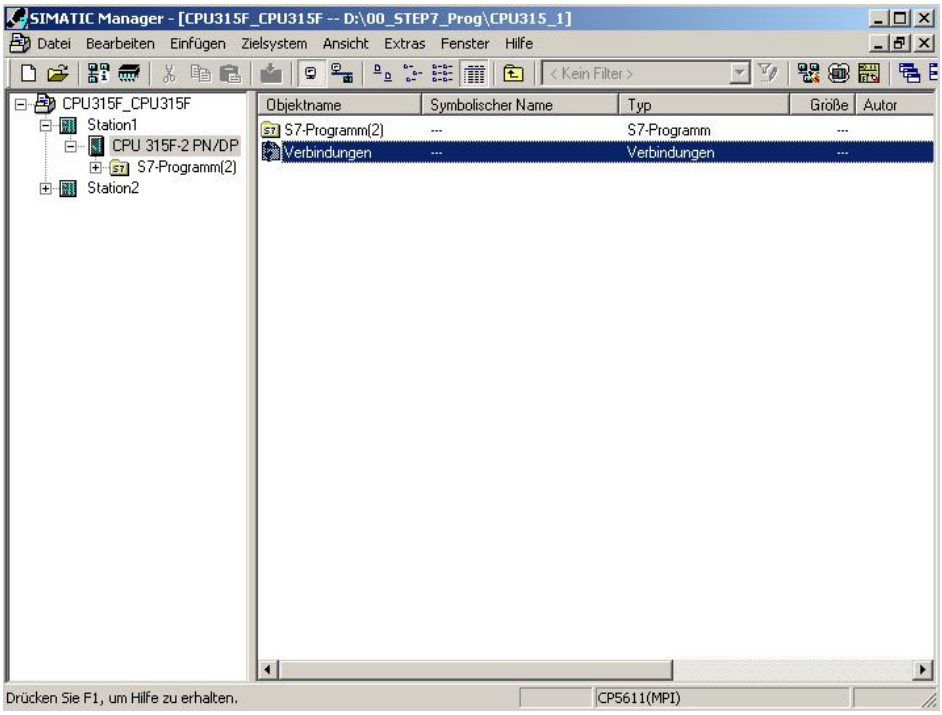
Note: If several IO controllers are on the network, the device can be identified with the MAC address imprinted on it.

34. Now, the correct IP address has to be assigned to the IO controller if it has not been set correctly previously. Confirm this in the following dialog with **'Yes'**. (→ Yes)

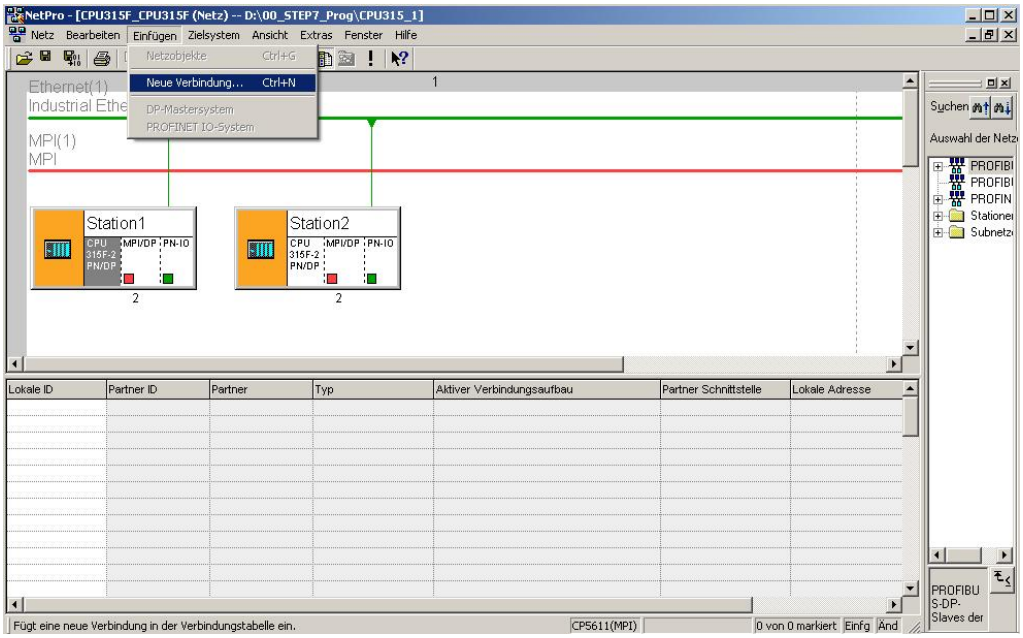


<<The selected station does not have an IP address. Do you want to assign the address 192.168.0.2. now?>>

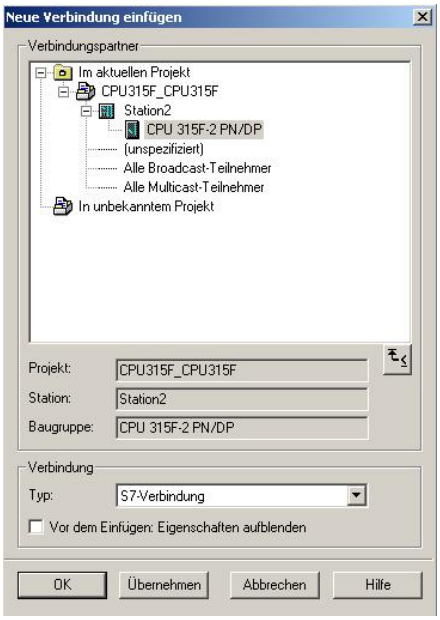
35. So that the two CPUs on the PROFINET can exchange data by using an S7 connection, first, with a double click, the folder **'Connections'** in one of the two stations (here: **'Station1'**) has to be opened in the **'SIMATIC Manager'**. (→ SIMATIC Manager → Connections)



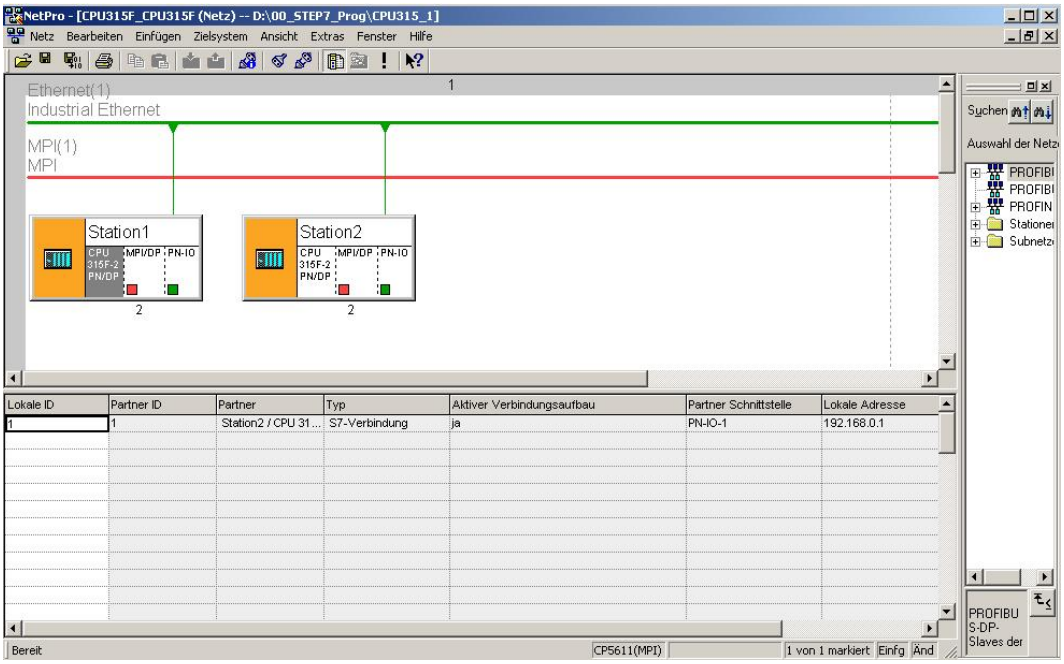
36. This opens the tool **'NetPro'**. There, we get a good overview over the networking of the available components MPI, PROFIBUS and Industrial Ethernet. One of the CPUs is already selected. Now, the **'New Connection'** is **'Insert'**ed. (→ Insert → New Connection)



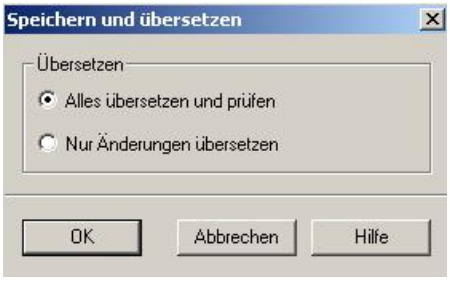
37. Then, from the current project, the other station **'Station2' -**with the CPU **'CPU 315F-2PN/DP'** inserted there- is selected as connection partner. The type of connection is an **'S7 Connection'**. (→ Station2 → CPU 315F-2PN/DP → S7 Connection →OK)



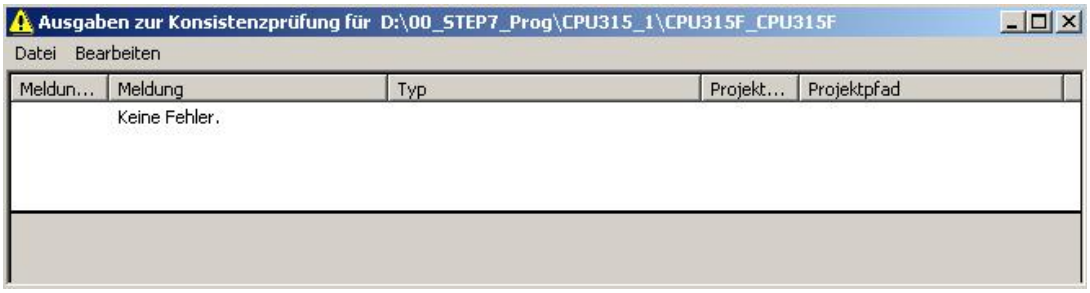
38. Now, the **'ID'** of the two connection partners is read off the connection table. (Here, both are on **'1'**!). Then, the connection table is saved and compiled by clicking on '  '. (→ )



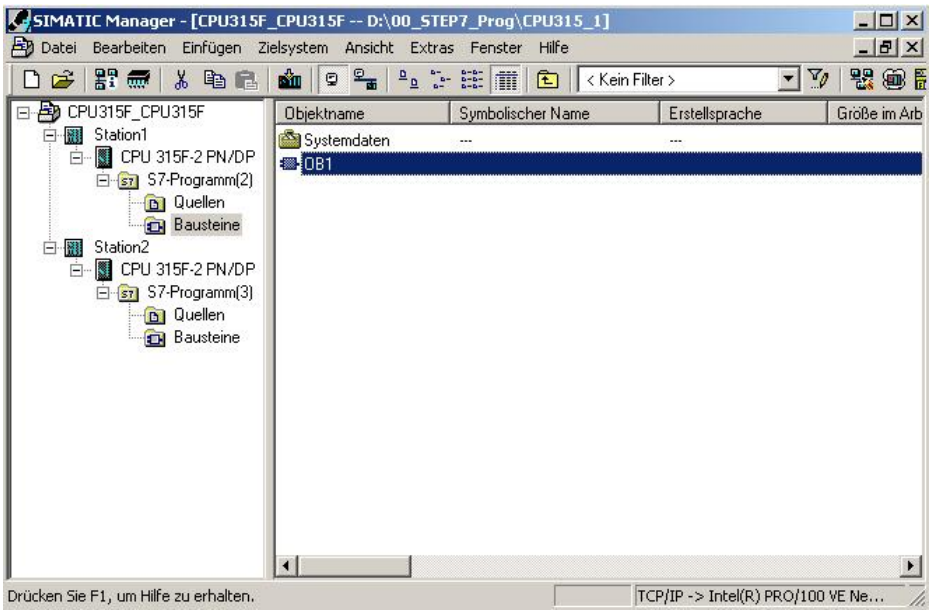
39. Now, select the compiling option **'Compile and check everything'**. (→ Compile and check everything →OK)



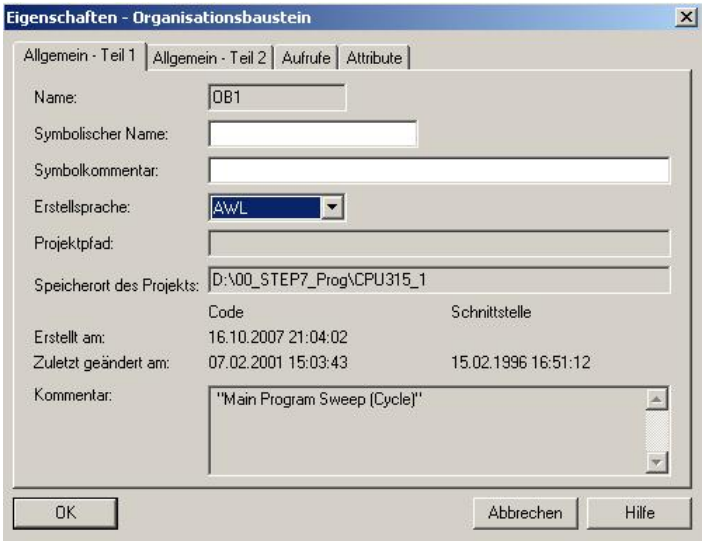
40. The subsequent window should display the message 'No errors'. Close the window by clicking on '☒'. (→ ☒)



41. After the connection was set up and compiled, we can start generating the program. With a double click, open **'OB1'** from the **'SIMATIC Manager'**. (→ OB1)



42. Select the **'Programming Language STL'** and accept with **'OK'**. (→ STL →OK)



Through programmed SFB block calls, the user program activates the transfer of the data areas for S7 communication, and monitors its successful execution.
The SFB blocks required for communication are stored in the **"Standard Library"** under the item **"Standard Function Blocks"**. To use these functions, they have to be incorporated in (copied to) your "own" project.

The **SFB12 "BSEND"** transfers data to the PROFINET for transmission over a configured S7 connection.
The specified data area can be a flag area or a data block area. Faultless execution is signaled.

The **SFB12 "BSEND"** sends data to a partner SFB of the type **"BRCV"** in the other CPU. During this data transfer, a larger data volume can be transported between the communication partners than is possible with all other communication SFBs for configured S7 connections. The following data volumes can be transmitted:

- 65534 bytes for the S7-400 and S7-300 over an integrated interface.

The data area to be sent is segmented. Each segment is sent individually to the partner. The last segment is acknowledged when it arrives, regardless of the associated call of the SFB "**BRCV**".

Sending is activated after the block is called and the edge is positive at control input REQ. The data from the user memory is sent asynchronously to user program processing.
The start address of the data to be sent is specified with SD_1. The length of the send data is specified request-related with LEN. Thus, LEN replaces the length share of SD_1.

The parameter R_ID has to be identical for associated SFBs.

If the edge is positive at control input R, current sending is canceled.

If sending was successful, it is indicated with a 1 at the status parameter DONE.

After the previous send request is completed, a new send request can be processed only if the status parameters DONE or ERROR have the value 1.

Because data transmission is asynchronous, transmitting data can be started again only after the previous data was fetched by calling the partner SFB. Until the data is fetched, the status value 7 is read out when the SFB "**BSEND**" is called.

When calling the **SFB12 "BSEND"**, the following parameters have to be entered one after the other:

Parameter	Declaration	Type	Memory Area	Description
REQ	INPUT	BOOL	I, O, F, D, L	Control parameter Request, activates the data exchange at rising edge
R	INPUT	BOOL	I, O, F, D, L, Const.	Control parameter Reset, activates the cancellation of a data exchange that is still running at rising edge
ID	INPUT	WORD	F, D, Const.	Addressing parameter ID
R_ID	INPUT	DWORD	I, O, F, D, L, Const.	Addressing parameter R_ID
DONE	OUTPUT	BOOL	I, O, F, D, L	Status parameter DONE: 0:Request not yet started or is still being executed 1:Request executed without fault
ERROR	OUTPUT	BOOL	I, O, F, D, L	Status parameter ERROR and STATUS
STATUS	OUTPUT	WORD	I, O, F, D, L	Error indication: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: warning, STATUS provides detailed information ERROR=1 An error occurred. STATUS provides detailed information about the type of error
SD_1	IN_OUT	ANY	F, D	Pointer to send area. Only the following data types are permissible: BOOL (not permitted: bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. Note: If the ANY pointer accesses a DB, the DB aways has to be specified; (for example: P# DB10.DBX5.0 Byte 10).
LEN	IN_OUT	WORD	I, O, F, D, L	Length of the data block to be sent in bytes

SFB13 "BRCV" accepts the data transmitted by PROFINET over a configured S7 connection.
The specified data area can be a flag area or a data block area. A faultless execution is signaled.

SFB13 "BRCV" receives the data from a partner SFB of the type "**BSEND**" in the other CPU. After each data segment is received, an acknowledgement is sent to the partner SFB, and the parameter LEN is updated.

After being called, the block is ready to receive with the value 1 at the control input EN_R. With EN_R=0, a running request can be canceled.
The start address and the maximum length of the receive area are specified with RD_1. The length of the received data block is indicated in LEN.
With each positive edge at EN_R, the parameters R_ID, ID and RD_1 are accepted. After the request is completed, new values can be assigned to the parameters R_ID, ID and RD_1. To transmit segmented data, the block has to be called cyclically in the user program.

The data is received from the user memory asynchronously to the processing of the user program.

The parameter R_ID has to be identical for the associated SFBs.

A faultless receipt of all data segments is indicated in the status parameter NDR with the value 1. The received data remains unchanged until SFB 13 is called again with EN_R=1.

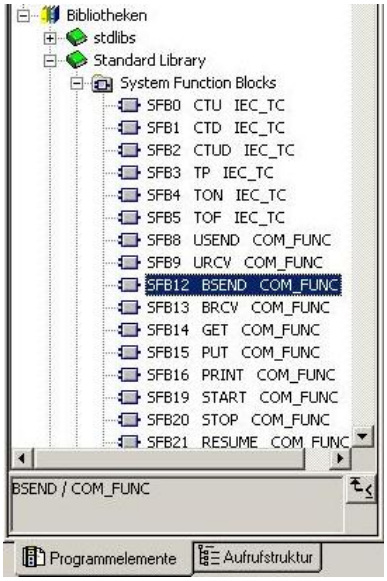
If the block is called again during asynchronous receiving, a warning is read out in the status parameter STATUS; if the call is made with EN_R=0, receiving is canceled, and the SFB returns to its basic status.

When callings **SFB13 "BRCV"**, the following parameters have to be entered one after the other:

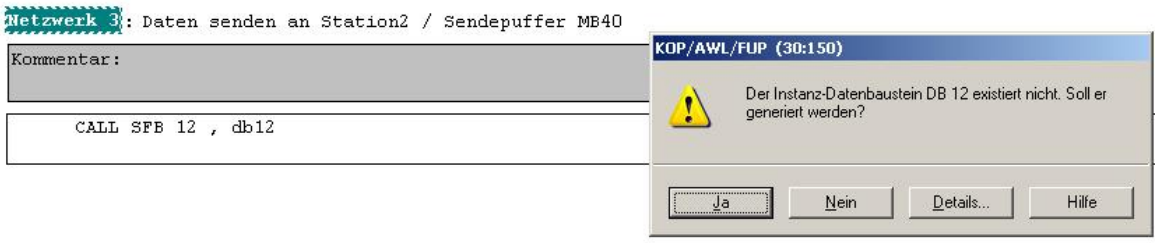
Parameter	Declaration	Type	Memory Area	Description
EN_R	INPUT	BOOL	I, O, F, D, L, Const.	Control parameter Enabled to receive; signals readiness to receive if the input is set
ID	INPUT	WORD	F, D, Const.	Addressing parameter ID
R_ID	INPUT	DWORD	I, O, F, D, L, Const.	Addressing parameter R_ID
NDR	OUTPUT	BOOL	I, O, F, D, L	Status parameter NDR: 0: Request not started or still running 1: Request was completed successfully
ERROR	OUTPUT	BOOL	I, O, F, D, L	Status parameter ERROR and STATUS
STATUS	OUTPUT	WORD	I, O, F, D, L	Error indication: ERROR=0 and STATUS has the value: 0000H: neither warning nor error <> 0000H: warning, STATUS provides detailed information ERROR=1 An error occurred. STATUS provides detailed information about the type of error
RD_1	IN_OUT	ANY	M, D	Pointer to receive area. The length indication specifies the maximum length of the block to be received. Only the following data types are permissible: BOOL (not permitted: bit field), BYTE, CHAR, WORD, INT, DWORD, DINT, REAL, DATE, TOD, TIME, S5TIME, DATE_AND_TIME, COUNTER, TIMER. Note: If the ANY pointer accesses a DB, the DB always has to be specified (for example: P# DB10.DBX5.0 Byte 10).
LEN	IN_OUT	WORD	I, O, F, D, L	Length in bytes of the data received so far

43. With **'LAD, STL, FBD – Program S7 Blocks'**, you now have an editor with which you can generate your STEP7 program accordingly. To this end, OB1 has already been opened with the first network. To generate your initial operations, highlight the first network. Now you can write your STEP7 program. In STEP7, individual programs are usually arranged in networks. A new network is opened by clicking on the network symbol '  '.

    In OB1, the content of the flag byte MB40 is sent to "Station2" with the block **SFB12 "BSEND"**. In the catalog, you can drag this block from the **'Library' 'Standard Library'** to your network. (→ Libraries → Standard Library → System Function Blocks → SFB12 BSEND)



44. The instance DB (here: DB12) can be generated automatically if the query is confirmed with **'Yes'**. (→ Call SFB12,DB12 → Yes)



<<The instance data block DB12 does not exist. Do you want to generate it?>>

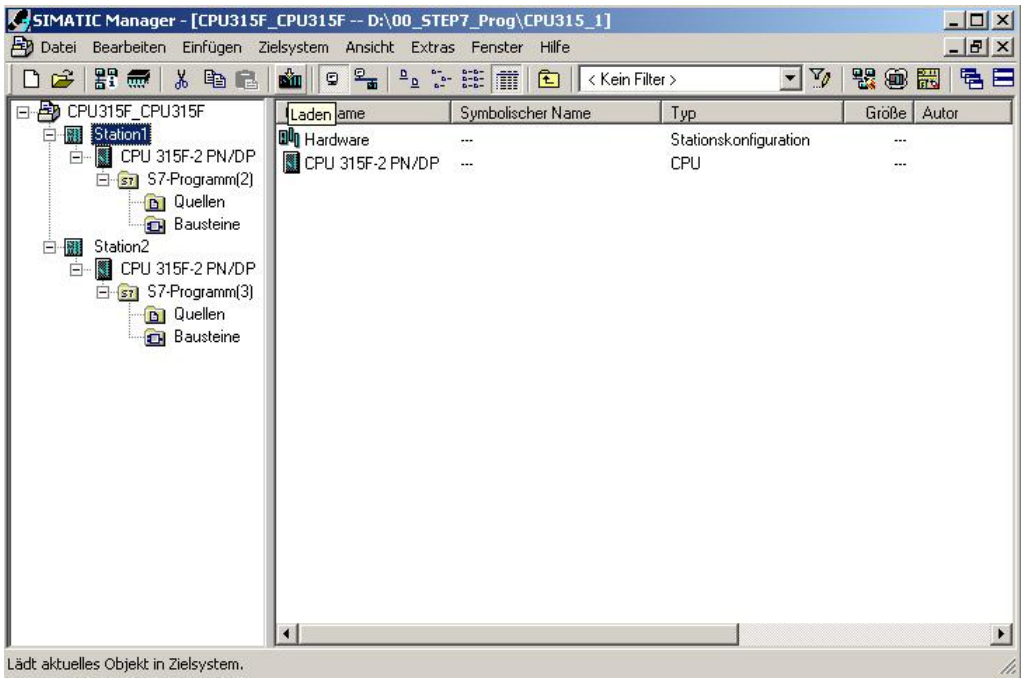
45. The complete program for sending the flag byte MB40 and for testing with the input byte IB0 looks like this:

```
OB1 :  "Main Program Sweep (Cycle)"
Netzwerk 1 :Specify length of send area
      L     W#16#1                  //Load Value 1 in hexa-decimal code
      T     MW    10                //Transfer in flag word MW10


Network 2: Write inputs to send buffer

      L     EB     0                 //Load input byte IB0
      T     MB    40                 //Transfer in send buffer flag word MW40


Network 3: Send data to Station2/Send-buffer MB40

      CALL  "BSEND" , DB12        //Call Send block SFB12 (BSEND) with instance DB DB12
       REQ   :=E1.0               //Activate Send request (with positive edge)
       R     :=E1.1               //Cancel Send request (with positive edge)
       ID    :=W#16#1             //ID of connection to other station (from NetPro)
       R_ID  :=DW#16#1            //R-ID on sender side has to match R_ID on receiver side
       DONE  :=M80.0              //Diagnostic info DONE indicates that request was executed without fault
       ERROR :=M80.1              //Diagnostic information ERROR indicates that an error occurred
       STATUS:=MW82               //Diagnostic info STATUS provides detail info about occurring errors
       SD_1  :=P#M 40.0 BYTE 1    //Data area that is to be sent (here: MB40)
       LEN   :=MW10               //Length indication of the send area in bytes (only I,O,F,D,L; no constants
      NOP   0
```

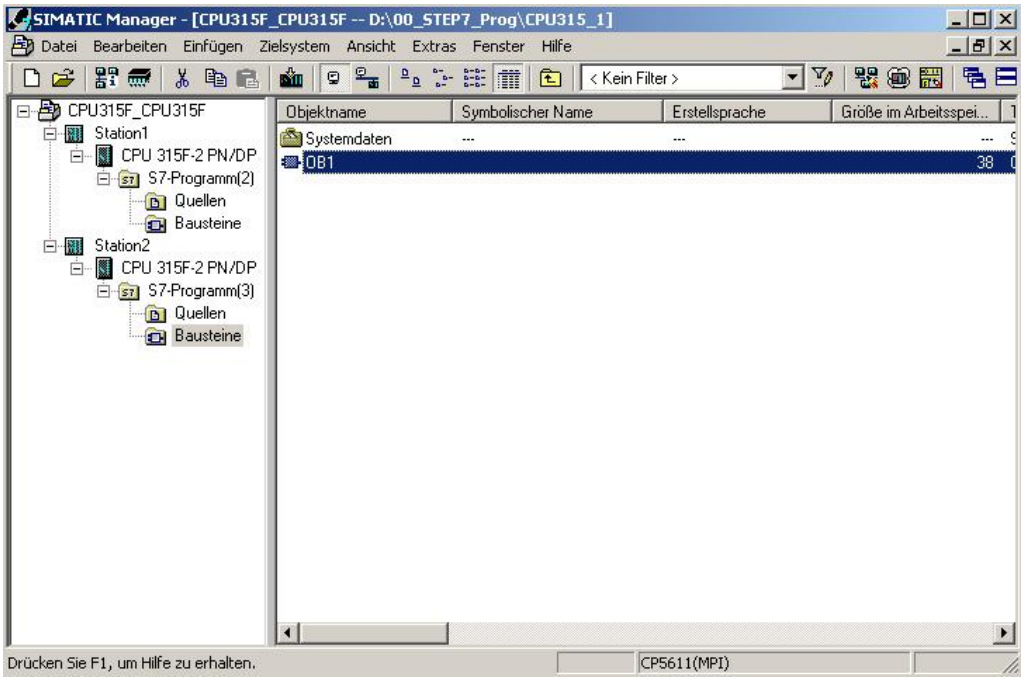
Note: More detailed information about **SFB12 "BSEND"** is provided in the descriptions in the preceding pages, or on Online Help

46. Save OB1 '  ' and close the **'LAD, STL, FBD Editor'** by clicking on '☒'. (→  → ☒)

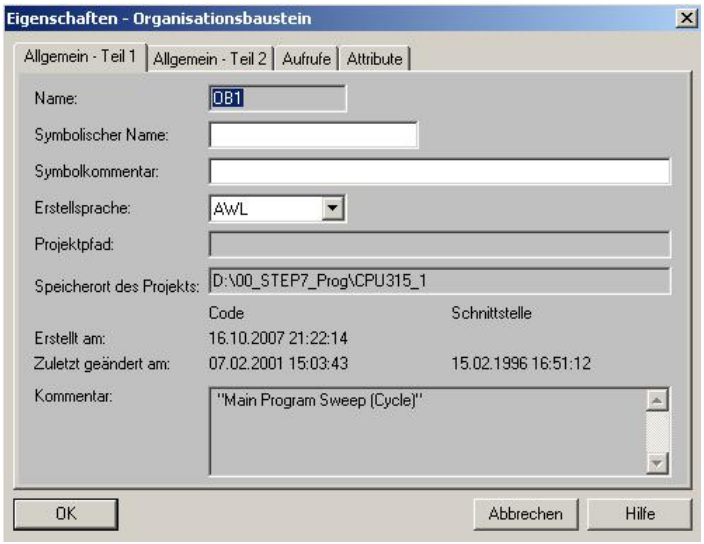
47. Now, the STEP 7 program has to be loaded to the PLC. In our case, we are doing this from the **'SIMATIC** Manager'. In the SIMATIC Manager, highlight the folder **'Station1'** and click on Load '  '. The CPU's key switch should be on STOP, and the programming device should be connected to the CPU by means of the Ethernet! (→ SIMATIC Manager → Station1 → )



48. For 'Station2' also, open **'OB1'** from the **'SIMATIC Manager'** with a double click. (→ OB1)

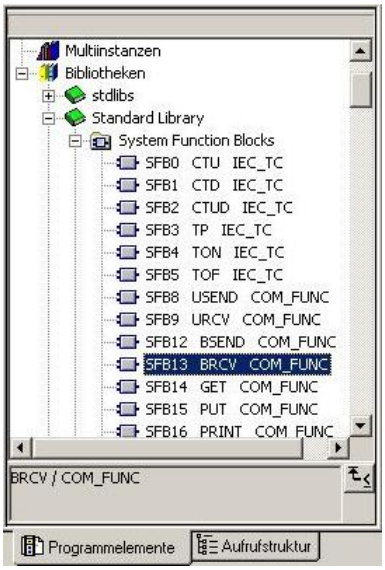


49. Optional: For documentation, enter the properties of OB1 and accept with **'OK'**. (→OK)

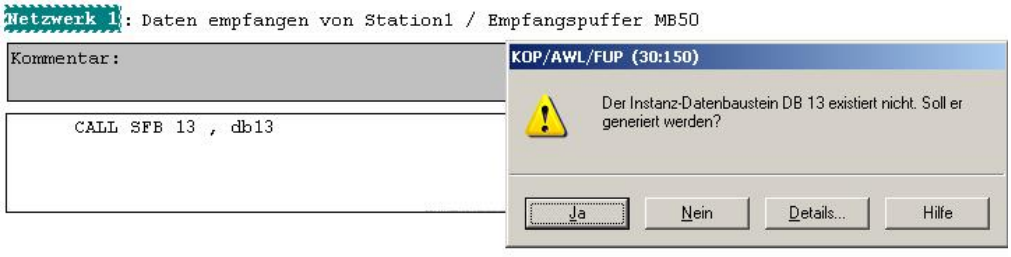


50. With **'LAD, STL, FBD – Program S7 Blocks'**, you now have an editor with which you can generate your STEP7 program accordingly. To this end, OB1 has already been opened with the first network. To generate your initial operations, highlight the first network. Now you can write your STEP7 program. In STEP7, individual programs are usually arranged in networks. A new network is opened by clicking on the network symbol '  '.

    With the block **SFB13 "BRCV"**, the data of Station1 is received in OB1 and written to the flag byte MB50. In the catalog, you can drag this block from the **'Library' 'Standard Library'** to your network. (→ Libraries → Standard Library → System Function Blocks → SFB13 BRCV)



51. The instance DB (here: DB13) can be generated automatically if the query is confirmed with **'Yes'**. (→ Call SFB13,DB13 → Yes)



<<The instance data block DB 13 does not exist. Do you want to generate it now?>>

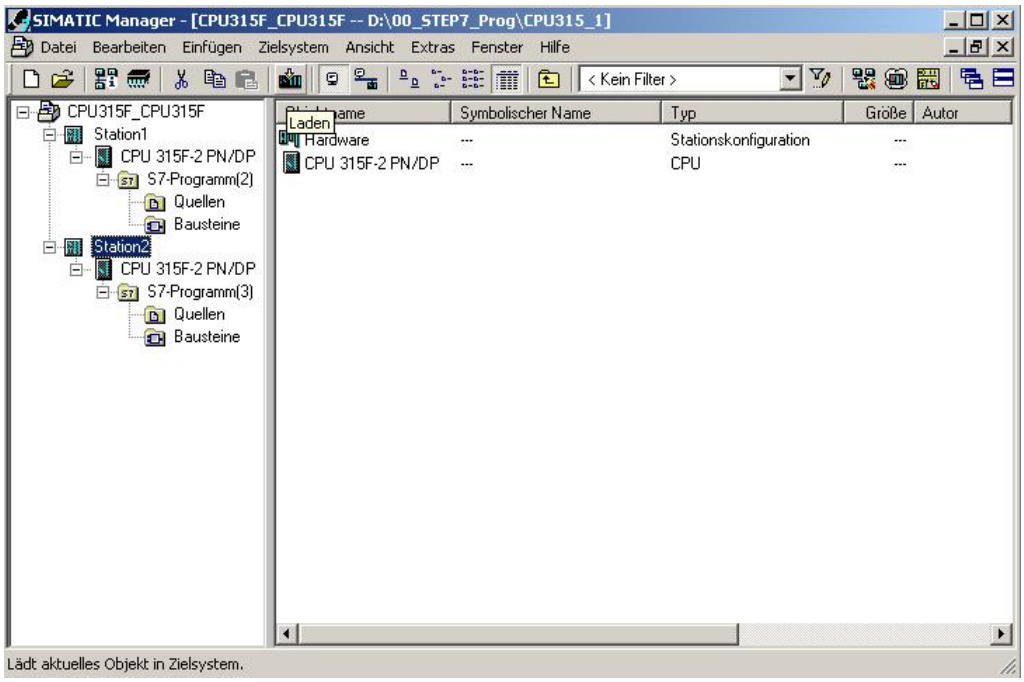
52. The complete program for receiving the data, for writing it to the receive buffer MB50, and for indication at output byte QB4 looks like this:

```
OB1 :  "Main Program Sweep (Cycle)"
Netzwerk 1: Receiving data from Station1/Receive buffer MB50
      CALL  "BRCV" , DB13              //Call of receive block SFB13 (BRCV) with instance DB DB 13
       EN_R  :=E1.0                    //Activation of Ready to receive (for EN_R == 1)
       ID    :=W#16#1                  //ID of connection to the other station (from NetPro)
       R_ID  :=DW#16#1                 //R_ID on receiver side has to match the R_ID on the sender side
       NDR   :=M90.0                   //Diagnostic info NDR indicates if request was completed successfully
       ERROR :=M90.1                   //Diagnostic information ERROR indicates that an error occurred
       STATUS:=MW92                    //Diagnostic info STATUS provides detail info about occurring errors
       RD_1  :=P#M 50.0 BYTE 1         //Data area that is to receive (here: MB50)
       LEN   :=MW12                    //Length of received data in bytes
      NOP   0

Network 2: Read out receive buffer and indicate at outputs
      L     MB    50                  //Load receive buffer flag byte MB50
      T     AB     4                  //Transfer to output byte QB4
```

Note: More detailed information about the **SFB13 "BRCV"** is provided in the descriptions in the preceding pages, or on Online Help.

53. Save OB1 '' and close the **'LAD, STL, FBD Editor'** by clicking on ' x'. (→ → x)

54. Now, the STEP7 program has to be loaded to the PLC. In our case, this is done from the **'SIMATIC Manager'**. In the SIMATIC Manager, highlight the folder **'Station2'** and click on Load ' '. The CPU's key switch should be on STOP and the programming device connected to the CPU by means of the Ethernet! (→ SIMATIC Manager → Station2 →)



55. The programs are started by switching the key switches on the CPUs to RUN. Test the programs according to the exercise described on Page 7.