

Communication between SIMATIC S7-300 and Modicon M340 PLC

Modbus TCP Connection

Application Description • January 2010

Applications & Tools

Answers for industry.

SIEMENS

Industry Automation and Drives Technologies Service & Support Portal

This article is taken from the Service Portal of Siemens AG, Industry Automation and Drives Technologies. The following link takes you directly to the download page of this document.

<http://support.automation.siemens.com/WW/view/en/38586568>

If you have any questions concerning this document please e-mail us to the following address:

online-support.automation@siemens.com

SIMATIC

Modbus TCP Connection

Automation Task	1
Automation Solution	2
Basics	3
Function Mechanisms of this Application	4
Installation	5
Startup of the Application	6
Operation of the Application	7
Related Literature	8
History	9

Warranty and Liability

Note

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

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

We do not accept any liability for the information contained in this document.

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

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

Table of Contents

Warranty and Liability	4
1 Automation Task	7
2 Automation Solution	8
2.1 Overview of overall solution	8
2.2 Description of the core functionality	10
2.3 Hardware and software components used.....	14
3 Basics	16
3.1 Basics on Modbus TCP.....	16
3.2 S7 function blocks for Modbus TCP	20
4 Function Mechanisms of this Application	23
4.1 Program structure of S7 CPU and ET200S CPU with integrated PN interface.....	24
4.1.1 Program details on Modbus PN blocks	24
4.1.2 Configuration explanations.....	31
4.2 Program structure of S7 CPU with CP	32
4.2.1 Program details on MODBUS CP blocks	32
4.2.2 Configuration explanations.....	40
4.3 Program structure of Modicon M340	40
4.3.1 Program details on Modicon M340 blocks	41
4.3.2 Configuration explanations.....	41
5 Installation	42
5.1 Installation of the hardware	42
6 Startup of the Application	45
6.1 Configuration of CPU319-3 PN/DP	45
6.1.1 Hardware configuration	45
6.1.2 Insert Modbus TCP blocks into project	47
6.1.3 Configuring Modbus TCP connections.....	47
6.1.4 Project download.....	50
6.2 Configuration of IM151-8 PN/DP CPU	51
6.2.1 Hardware configuration	51
6.3 Configuring the CPU315-2 PN/DP with CP343-1 Lean	53
6.3.1 Hardware configuration	53
6.3.2 Creating a project for Modbus TCP.....	56
6.3.3 Configuring Modbus TCP connections.....	57
6.3.4 Downloading project.....	59
6.4 Configuration of Modicon M340	60
6.4.1 Using application example	60
6.4.2 Hardware configuration	62
6.4.3 Configuring an Ethernet interface for Modbus TCP	64
6.4.4 Creating a project for Modbus TCP.....	66
7 Operation of the Application	68
7.1 Operation of CPU319-3 PN/DP and IM151-8 PN/DP CPU.....	68
7.1.1 S7 station is client	68
7.1.2 S7 station is server.....	71
7.2	71
7.3 Operation of CPU315-2 PN/DP + CP343-1 Lean	72
7.3.1 S7 station is client	72
7.3.2 S7 station is server.....	74
7.4 Operation of Modicon M340	76
7.4.1 Modicon M340 as client.....	76
7.4.2 Modicon M340 as server.....	77
8 Related Literature.....	79

Table of Contents

8.1	Bibliography.....	79
8.2	Internet Links.....	79
9	History	79

1 Automation Task

Introduction

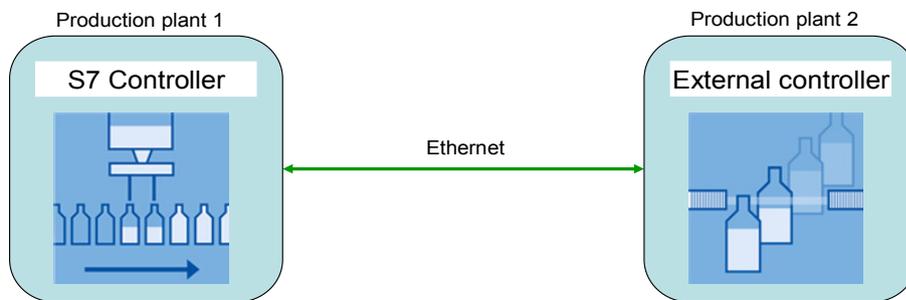
Protocols are necessary to exchange data between communication partners. One of these protocols which is mainly used in the industrial environment, is Modbus TCP. The specification of this protocol is open and can therefore be used by anyone. Thus, there is a multitude of components from different manufacturers with Modbus TCP interface.

To enable a simple and fast connection to such third-party devices they also have to support Modbus TCP.

Overview of the automation task

The figure below provides an overview of the automation task.

Figure 1-1



Description of the automation task

There are two production plants in a product line which carry out intermediate and final assembly of a serial production. Between those two manufacturing plants production data is to be exchanged. One of the two plants is equipped with a controller of a third-party manufacturer. For the communication with other components this external controller offers a Modbus TCP interface. Data exchange is to be realized via this interface.

The controllers of the two production plants and the panel are located in the same IP subnet. Therefore a gateway is not needed.

Quantity frameworks of the data transmission

In the application example the volume of data which is transferred via Modbus TCP is limited to 64 Bit. The transfer is word by word. I.e., per client job four words are read from the server or are written into the server.

The table below shows the maximum possible data transfer per Modbus TCP job.

Table 1-1 : Maximum data volume per job

	Transfer bit-by-bit	Transfer word-by-word
Read job	250 Byte	250 Byte
Write job	100 Byte	200 Byte

2 Automation Solution

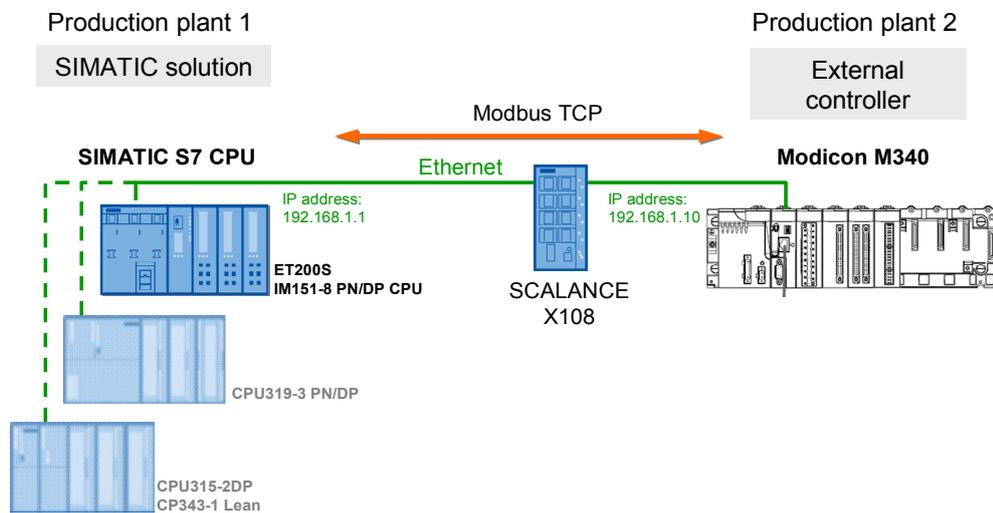
2.1 Overview of overall solution

There are several possibilities to link an external controller via Modbus TCP to a SIMATIC system because there are different peers on the SIMATIC side which support this protocol, using different function blocks for Modbus TCP. Below, on the example of three different SIMATIC components it is shown how a link to an external controller is realized via Modbus TCP.

Schematic layout

The figures below display the components of three different solution alternatives:

Figure 2-1 :Connection Modicon M340 ← → with S7-CPU's



Design of the SIMATIC system

In each solution alternative the SIMATIC components are physically connected via an industrial Ethernet cable. SCALANCE X108 enables the connection of the PG/PC and the SIMATIC station, and the connection to the external system. Via the physical connection, production data is exchanged with an external system, using Modbus TCP. The PG/PC will be used to start the send and receive jobs.

The following SIMATIC stations are available:

- CPU319-3 PN/DP
- ET200S (IM151-8 PN/DP)
- CPU315-2 PN/DP with CP 343-1 Lean

Design of the external system

The external system is a Modicon M340 controller. Via an industrial Ethernet cable it is physically linked with the SCALANCE X108.

Topics not covered by this application

This application does not

- contain an introduction to STEP7
- contain an introduction to Unity Pro XL
- contain an introduction to WinCC flexible

Basic knowledge of these topics is assumed.

Scope of the application example

This application example will describe:

- basics on Modbus TCP
- configuration and structure of a Modbus TCP connection between a CPU319-3 PN/DP and a Modicon M340, as well as data transmission via Modbus TCP between the two peers (variant 1: Modicon M340 acts as client, CPU319-3 PN/DP acts as server; variant 2: Modicon M340 acts as server, CPU319-3 PN/DP acts as client)
- configuration and structure of a Modbus TCP connection between an ET200S (IM151-8 PN/DP) and a Modicon M340, as well as data transmission via Modbus TCP between the two peers (variant 1: Modicon M340 acts as client, ET200S acts as server; variant 2: Modicon M340 acts as server, ET200S acts as client)
- configuration and structure of a Modbus TCP connection between a CPU315-2 PN/DP with CP343-1 Lean and a Modicon M340, as well as data transmission via Modbus TCP between the two peers (variant 1: Modicon M340 acts as client, CPU315-2 PN/DP with CP acts as server; variant 2: Modicon M340 acts as server, CPU315-2 PN/DP with CP acts as client)
- handling of Modbus TCP Wizard

2.2 Description of the core functionality

Sequence of the core functionality

A Modbus TCP communication between a SIMATIC controller and a Modicon M340 is created to exchange data between the two nodes afterwards. There are different hardware solutions on the SIMATIC side that need different software solutions for Modbus TCP. The SIMATIC station and Modicon M340 are alternately client or server. I.e. when the SIMATIC station acts as server the Modicon M340 station is the client and visa versa.

The figures below show the solution variants of this application example.

Connection Modicon M340 ← → CPU319-3 PN/DP

Figure 2-2: CPU319-3 PN/DP as Modbus TCP client; Modicon M340 as Modbus TCP server

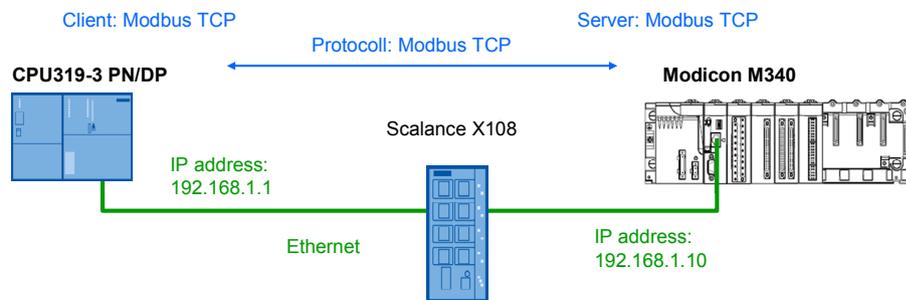
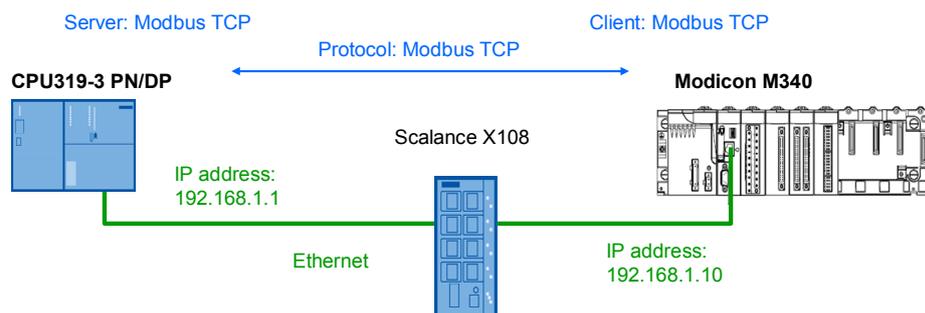


Figure 2-3: CPU319-3 PN/DP as Modbus TCP server; Modicon M340 as Modbus TCP client



Connection Modicon M340 ← → CPU315-2 PN/DP + CP343-1 Lean

Figure 2-4: CPU315-2 PN/DP + CP343-1 Lean as Modbus TCP client; Modicon M340 as Modbus TCP server

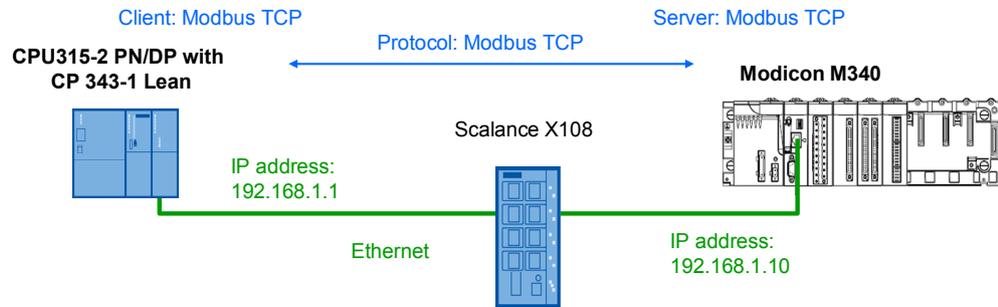
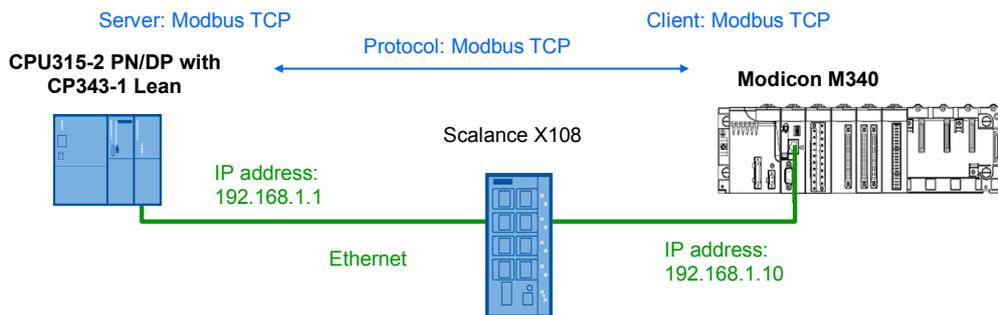


Figure 2-5: CPU315-2 PN/DP + CP343-1 Lean as Modbus TCP server; Modicon M340 as Modbus TCP client



Connection Modicon M340 ← → IM151-8 PN/DP CPU

Figure 2-6: IM151-8 PN/DP CPU as Modbus TCP client; Modicon M340 as Modbus TCP server

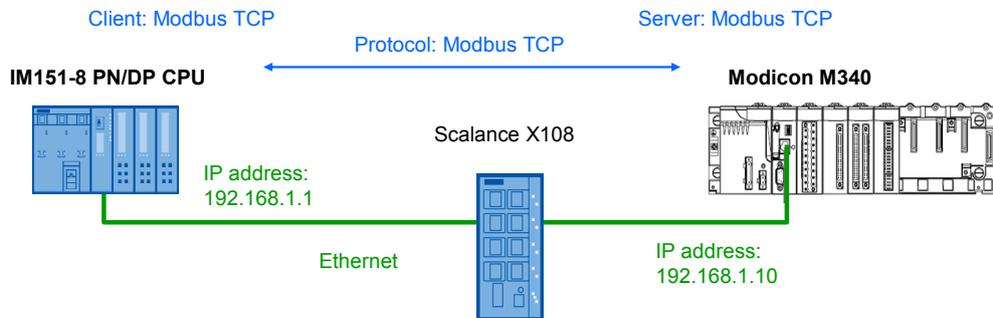
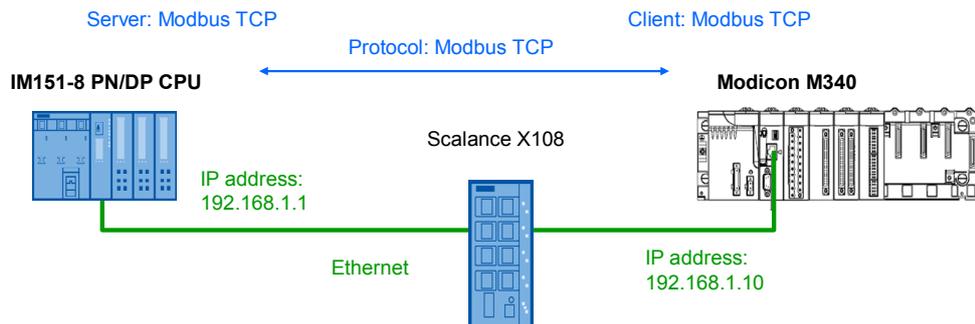


Figure 2-7: IM151-8 PN/DP CPU as Modbus TCP server; Modicon M340 as Modbus TCP client



Advantages of this solution

The solution introduced here offers the following advantages:

- option to connect Modbus TCP-capable external controllers
- extensibility of existing systems with SIMATIC S7 modules
- few changes necessary in SIMATIC component when exchanging a Modbus TCP peer
- option to select client or server functionality

Typical fields of application

The devices displayed here are typical Modbus TCP Siemens peers.

Table 2-1

Machine or branch	Task	Figure
Multi function measuring instrument Sentron PAC3200	U,I,R,f measurement	
e.g. TP177B, MP 277 8", MP 377 12" Touch	Visualization	

Other Modbus TCP components by third-party manufacturers:

- Modicon M340 by Schneider Electric
- Modbus/TCP (UDP) bus link by Phoenix Contact
- TwinCAT Modbus TCP server by Beckhoff

2.3 Hardware and software components used

The application was generated with the following components:

Hardware components

Table 2-2

Component	No.	MLFB / order number	Note
CPU319-3 PN/DP	1	6ES7318-3EL00-0AB0	
IM 151-8 PN/DP CPU	1	6ES7151-8AB00-0AB0	
CPU315-2 PN/DP	1	6ES7315-2EG10-0AB0	
CP343-1 Lean	1	6GK7343-1CX10-0XE0	
SCALANCE X108	1	6GK5108-0AB00-2AA3	
PS307 24 V/5 A	1	6ES7307-1EA00-0AA0	
Modicon M340 CPS2010	1		
P34 2030	1		Modicon M340 communication module
DDO 1602	1		Modicon M340 digital output module
DDI 1602	1		Modicon M340 digital input module

Standard software components

Table 2-3

Component	No.	MLFB / order number	Note
SIMATIC Manager V5.4+SP4	1	6ES7810-4CC08-0YA5	Configuration software for S7 CPUs
UnityProXL	1		Configuration software for Modicon M340
Modbus TCP Wizard	1		Program for configuring the Modbus TCP communication via PN - CPUs
Function blocks for Modbus TCP via PN CPU	1	2XV9 450-1MB02	These function blocks are not suitable for Modbus TCP via CP
Function blocks for Modbus TCP via CP	1	2XV9 450-1MB00	These function blocks are not suitable for Modbus TCP via PN CPU

Sample files and projects

The list below includes all files and projects used in this example. All program samples include not licensed Modbus TCP blocks. You get a valid license on the homepage of S7 OpenModbus/TCP (see [6](#)).

Table 2-4

Component	Note
38586568_Modbus_TCP_Kopplung_CODE_v11_de.zip	
38586568_Modbus_TCP_Kopplung_Doku_v11_de.pdf	

3 Basics

3.1 Basics on Modbus TCP

Modbus TCP is a client/server communication that uses the TCP/IP as transmission medium. In this architecture the client establishes a connection and sends request frames to the server. The server processes this request and sends the appropriate answer to the request back to the client. Depending on the job, data is either read from the memory area of the server or written into it. To be able to distinguish the type of jobs and the memory areas, there are clearly defined function codes for Modbus TCP. They are sent to the server by the client in the request frame, including the address of the memory area, the so called register address.

Frame structure

A Modbus frame consists of PDU (Protocol Data Unit) and ADU (Application Data Unit). In the PDU, function code and data belonging to the function are transmitted. ADU is for addressing and error detection. Whilst the PDU is identical for all Modbus variants (Modbus TCP, Modbus RTU, etc.) there are some differences for ADU. Addressing and error detection for Modbus TCP, for example, is undertaken by the subordinate TCP protocol. At the same time the TCP protocol is responsible for the integrity check of the data packets and if necessary also for troubleshooting. The information for addressing and error detection is transferred in the "MBAP Header" (Modbus Application Header).

Figure 3-11: Modbus standard frame

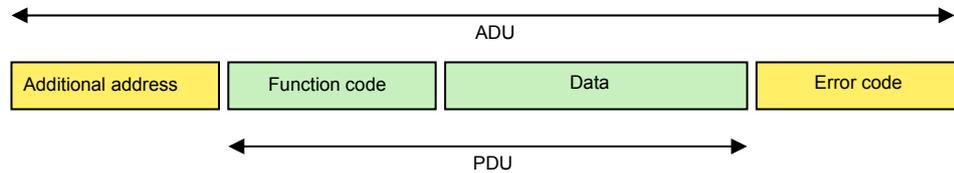
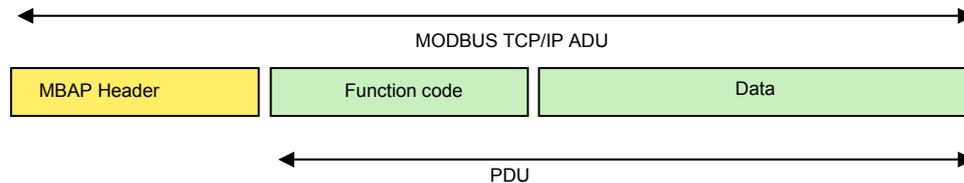


Figure 3-12: Modbus TCP frame



Modbus register and memory areas

Modbus is based on a series of memory areas with various characteristics. The addresses within this memory area are called register. Depending on the Modbus memory area these registers are either word- or bit-oriented. The table below displays all memory areas a Modbus component may contain.

Table 3-6

Memory area	Register size	Access	Note
Inputs	1 bit	read	can be changed by I/O system
Coils (outputs)	1 bit	read and write	can be changed by user program
Input register (input data)	16 bit word	read	can be changed by I/O system
Holding register (output data)	16 bit word	read and write	can be changed by user program

Functions codes

In the Modbus frame, the function code defines whether it is a read or write job and which memory area is to be accessed.

Precise addressing in the memory areas is via a register number and the number of registers to be processed. This information is also contained in the Modbus frame. The table below displays the function codes supported by S7 Modbus TCP.

All S7 MODBUS variants (PN-CPU's and TCP-CP's) support function codes 1, 2, 3, 4, 5, 6, 15 and 16.

Table 3-7

Function code	Function	Memory area
01	reads several bits	Coils (outputs)
02	reads several bits	Inputs
03	reads any number of registers	Holding register (output data)
04	reads any number of registers	Input register (input data)
05	writes individual bits	Coils (outputs)
06	writes individual register	Holding register (output data)
15	writes several bits	Coils (outputs)
16	writes more than one register	Holding register (output data)

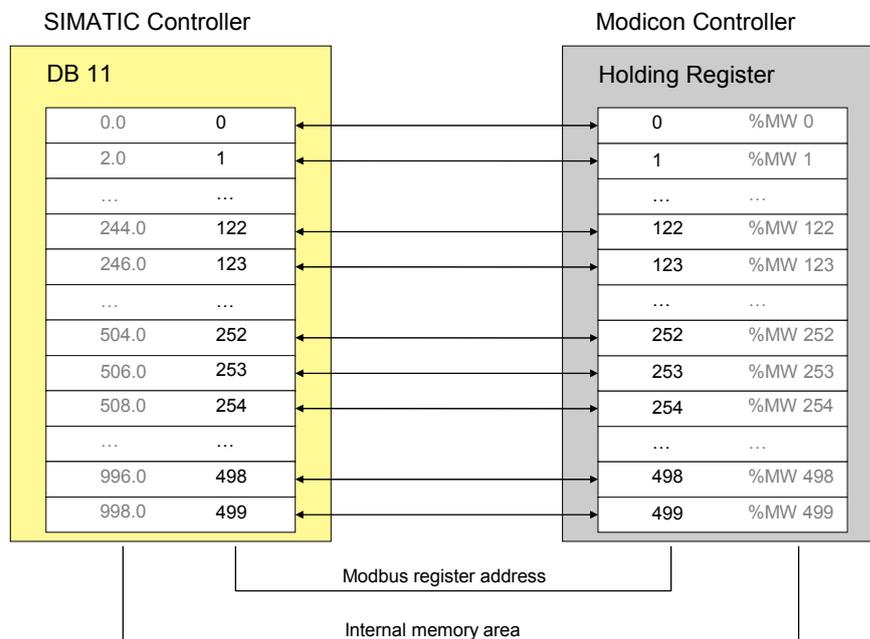
Display of address

Modbus addresses can vary from the internal address of a Modbus component. For this reason it is necessary to have information on the display of address. I.e. it has to be known which Modbus address displays an internal address of a Modbus component.

This is now explained using a simple example. This example shows the address display between a Siemens and a Modicon controller during data exchange via a holding register.

The Modbus register addresses are displayed in black. The internal addresses of the controllers are displayed in gray.

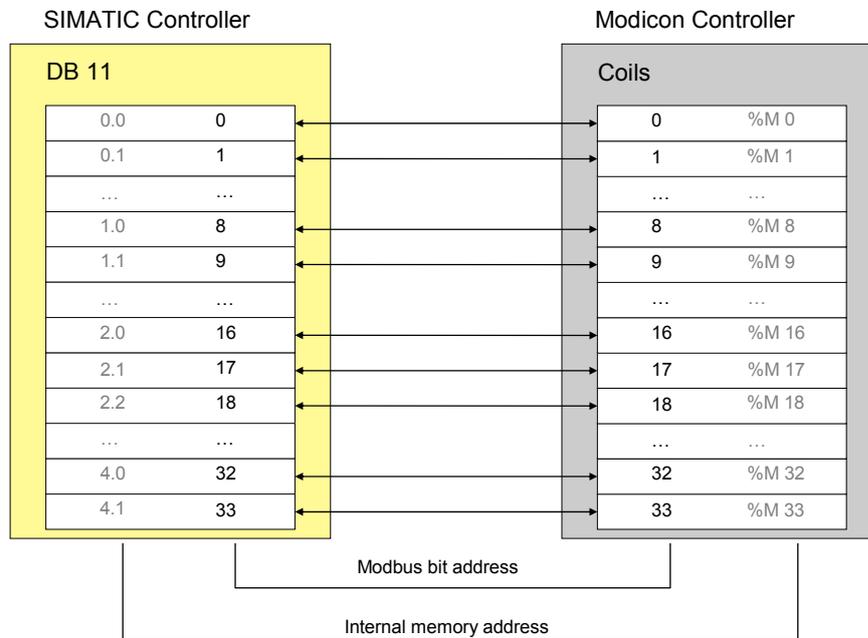
Figure 3-13:



Another example shows the address display between a Siemens and a Modicon controller during data exchange via coils.

The Modbus bit addresses are displayed in black. The internal addresses of the controllers are displayed in gray.

Figure 3-14:



Copyright © Siemens AG 2009. All rights reserved.

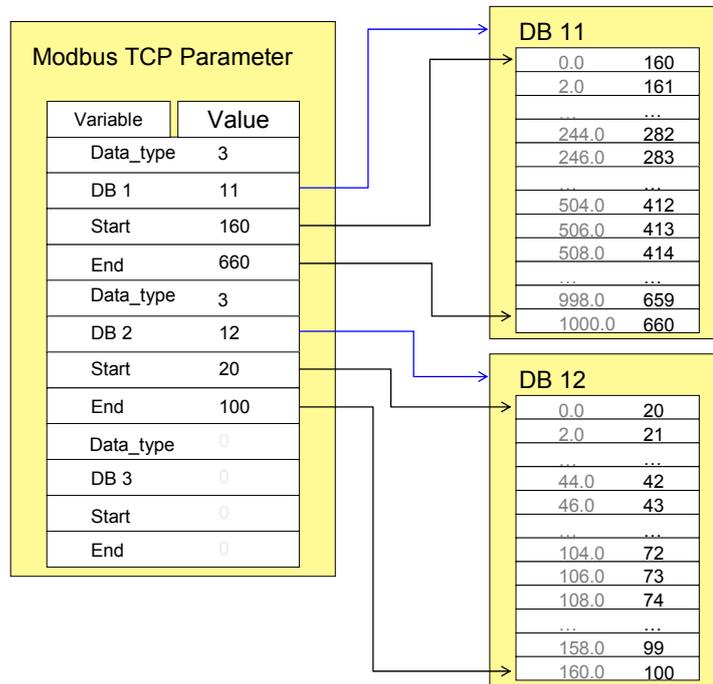
Configuring the Modbus addresses

When configuring the Modbus register addresses, attention has to be paid that there is no overlap on the SIMATIC side in the data areas. I.e. if there are, for example, two holding registers, one each configured in DB11 and DB12, no Modbus register addresses may be assigned twice.

The figure below shows the example of two DBs as holding register with clear Modbus register addresses. Configuration is via the Modbus parameter DB (MODBUS_PARAM) or the OB100.

S7 function blocks for Modbus TCP

Figure 3-15: Realization of address display by configuring the S7 Modbus TCP blocks (parameters are displayed in simplified form)



3.2 S7 function blocks for Modbus TCP

There are several solutions for a S7 controller to communicate with an external system which supports the Modbus TCP protocol. Each of these solutions is a software product which differs depending on the existing Ethernet interface. For a Modbus TCP communication via a CP (Communication Processor) a different software product is necessary than for the Modbus TCP communication via the integrated PN interface of a S7-CPU. This is due to the different internal data processing between the interface used and the user program in the S7 CPU. Each of these software solutions is basically a user program for a S7 CPU. This user program consists of different function blocks and data blocks which can simply be implemented in an existing S7 project.

The Modbus TCP Wizard is available for the configuration of a Modbus TCP communication via integrated PN interface of a S7 CPU. The Modbus TCP communication via a CP has to be configured manually.

Assigning a Modbus register address to an internal address of a SIMATIC CPU data block depends on the respective configuration of the Modbus TCP function blocks (MODBUS_PARAM or OB100).

Modbus TCP via CP

If an existing S7 CPU does not have an integrated PN interface, then it can be connected to an existing Ethernet network using a CP.
 Whilst the CP and the CPU, which are linked via a backplane bus, are exchanging data with each other, the CP simultaneously handles communication to the other Ethernet nodes.
 For this hardware variant the Modbus TCP solution "S7 Open Modbus TCP via CP343-1 and CP443-1" is available. This Modbus solution enables Modbus TCP communication for S7-300 and S7-400 CPUs in combination with a CP.
 The following hardware and software requirements are necessary for the use of the Modbus TCP CP solution. There are no firmware restrictions.

Table 3-1

Hardware	MLFB	Firmware
CP343-1	6GK7 343-1CX00-0XE0	
	6GK7 343-1CX10-0XE0	
	6GK7 343-1EX11-0XE0	
	6GK7 343-1EX20-0XE0	
	6GK7 343-1EX21-0XE0	
	6GK7 343-1EX30-0XE0	
	6GK7 343-1GX11-0XE0	
	6GK7 343-1GX20-0XE0	
	6GK7 343-1GX21-0XE0	
	6GK7 343-1GX30-0XE0	
CP443-1	6GK7 443-1EX11-0XE0	
	6GK7 443-1EX20-0XE0	
	6GK7 443-1EX40-0XE0	
	6GK7 443-1EX41-0XE0	
	6GK7 443-1GX11-0XE0	
	6GK7 443-1GX20-0XE0	

Table 3-2

Software	MLFB	Version
SIMATIC Manager	6ES7810-4CC07-0YA5	as of version 5.3

Modbus TCP via integrated interface of a CPU

If the integrated interface of the S7-CPU is to be used instead of the CP, another Modbus TCP solution is necessary, as already mentioned earlier. "S7 Open Modbus TCP PN" enables the Modbus TCP communication via the integrated PN interface for all S7-CPU's and ET200S-CPU's.

The following hardware and software requirements are necessary for the use of the Modbus TCP PN solution.

Table 3-3

Hardware	MLFB	Firmware
CPU315-2 PN/DP	6ES7 315-2EH13-0AB0	V2.6.7
	6ES7 315-2EG10-0AB0	V2.3.4
CPU317-2 PN/DP	6ES7317-2EJ10-0AB0	V2.3.4
	6ES7317-2EK13-0AB0	V2.6.7
CPU319-3 PN/DP	6ES7319-3EL00-0AB0	V2.7.2
CPU414-3 PN/DP	6ES7414-3EM05-0AB0	V5.2
CPU416-3 PN/DP	6ES7416-3FR05-0AB0	V5.2
IM151-8 PN/DP	6ES7151-8AB00-0AB0	V2.7.1

Table 3-4

Software	MLFB	Version
SIMATIC Manager	6ES7810-4CC08-0YA5	as of version 5.4 SP4

4 Function Mechanisms of this Application

This chapter shows the program structures of the S7 Modbus solution and the Modicon M340.

The Modicon M340 controller can be used as Modbus server without additional functions in the user program at any time. As soon as a Modbus request is received, it is processed and immediately answered. A Modbus server usually expects such a request on port 502. Several requests from different clients can also be processed on this port.

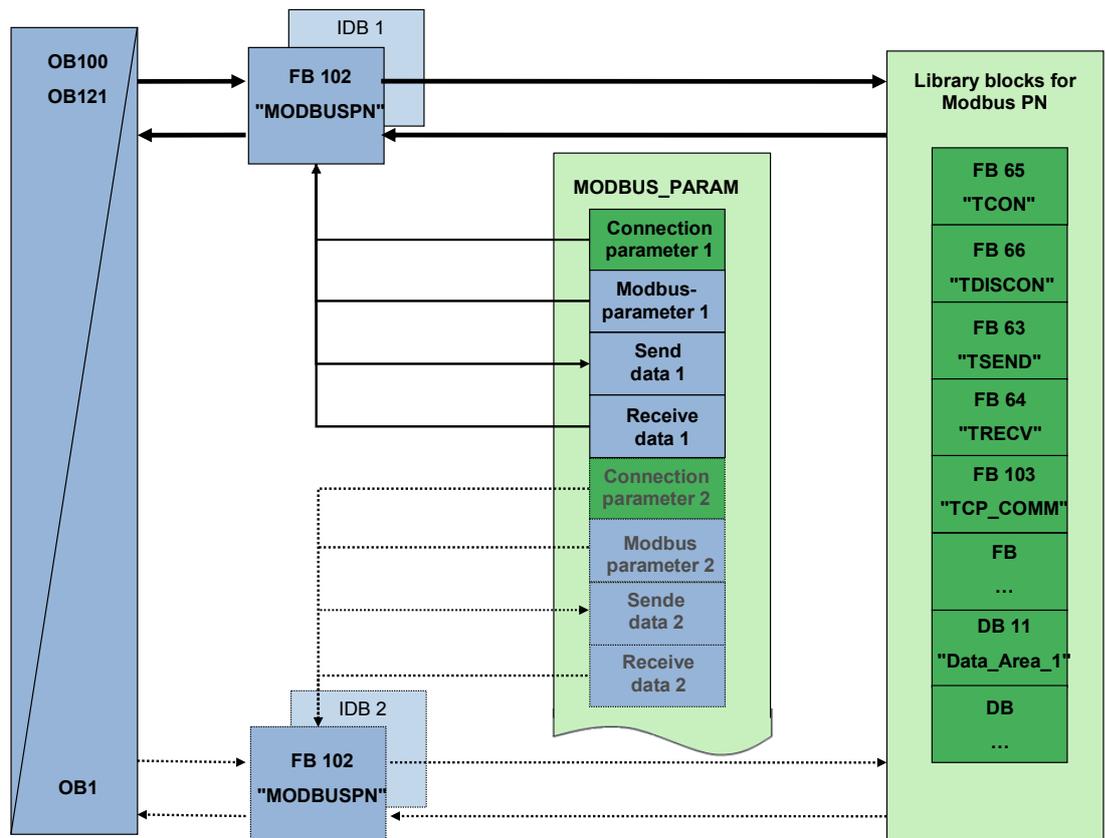
If the Modicon M340 controller is to be used as client, different function calls are necessary in the user program depending on the type of the job to be sent. Here, it has to be distinguished whether it is a read or a write job.

When a SIMATIC S7 CPU is either Modbus server or Modbus client, all the corresponding functions or function blocks have to be called in the user program. In the process, a further call of the Modbus function block with individual instance and the respective connection parameters is necessary for each Modbus connection. As opposed to the M340 CPU the S7 Modbus server can only process one client request per port. Therefore it is necessary that an extra server is to be configured with another port for each extra client which is supposed to be connected simultaneously with station S7 via Modbus TCP.

4.1 Program structure of S7 CPU and ET200S CPU with integrated PN interface

To be able to create one or several Modbus connections with a S7 CPU or an ET200S CPU with integrated PN interface it is necessary to configure the calls of the Modbus library blocks correctly. Doing this, a certain program structure has to be met. This program structure is displayed for two connections in the figure below. However, the application example only contains one Modbus TCP connection since the connection to the touch panel is realized via a S7 communication.

Figure 4-1



Copyright © Siemens AG 2009. All rights reserved.

4.1.1 Program details on Modbus PN blocks

The Modbus PN blocks have a Modbus DB parameter (Modbus_Param). Each Modbus connection is configured in this DB parameter and the Modbus register addresses as well as the respective memory areas are assigned in S7 (e.g. DB11 "Data_Area_1"). The corresponding send and receive data of each connection is also stored in this DB parameter.

To configure the DB parameter the Modbus TCP Wizard can be used or a direct value input in DB.

A separate call of the "MODBUSPN" function block with individual DB instance is necessary for each client or server connection. Please note that the call of the "MODBUSPN" function block has to take place in OB1, OB100 and in OB121 with the same DB instance per connection.

Modbus PN library blocks

The figure below displays all S7 program blocks which are needed for Modbus TCP via the integrated PN interface of a S7-CPU. They can be inserted and configured in any project from the examples for Modbus TCP for PN CPUs or directly from the Modbus TCP library for PN CPUs.

To be able to use Modbus TCP for PN CPUs in a program it is only necessary to call the "MODBUSPN" function block. The call of the other blocks is internally in the "MODBUSPN" function block.

Figure 4-17: Modbus TCP PN blocks from the sample program

Object name	Symbolic name	Created in language	Size in the work me...	Type	Version
Systemdaten	---	---	---	SDB	---
OB1	CYCL_EXC	STL	870	Organization Block	0.1
OB100	COMPLETE RESTART	STL	404	Organization Block	0.1
OB121	PROG_ERR	STL	70	Organization Block	0.1
FB63	TSEND	STL	292	Function Block	2.1
FB64	TRCV	STL	348	Function Block	2.2
FB65	TCON	STL	1018	Function Block	2.3
FB66	TDISCON	STL	230	Function Block	2.1
FB102	MODBUSPN	SCL	5950	Function Block	3.0
FB103	TCP_COMM	SCL	1880	Function Block	3.0
FB104	MOD_CLI	SCL	11132	Function Block	1.0
FB105	MOD_SERV	SCL	10660	Function Block	1.0
FC10	EQ_STRNG	STL	152	Function	1.1
DB1	CONTROL_DAT	DB	142	Data Block	0.1
DB2	MODBUS_PARAM	DB	686	Data Block	0.1
DB3	LICENSE_DB	DB	56	Data Block	0.1
DB11	DATA_AREA_1	DB	1236	Data Block	0.1
DB12	DATA_AREA_2	DB	436	Data Block	0.1
DB13	DATA_AREA_3	DB	636	Data Block	0.1
DB14	DATA_AREA_4	DB	118	Data Block	0.1
DB15	DATA_AREA_5	DB	118	Data Block	0.1
DB16	DATA_AREA_6	DB	118	Data Block	0.1
DB102	IDB_MODBUS	DB	830	Instance data block ...	0.0
Client_Job	Client_Job	---	---	Variable Table	0.1
SFB4	TON	STL	---	System function block	1.0
SFC24	TEST_DB	STL	---	System function	1.0

Copyright © Siemens AG 2009. All rights reserved.

OB1 call

In OB1 the cyclical calls of the "MODBUSPN" function block are made as well as of other user specific functions and function blocks. This is where runtime parameters are transferred to the "MODBUSPN" function block.

OB100 call

In OB100 the "MODBUSPN" function block is initialized. OB100 is completed once when there is a STOP → RUN transfer or when restarting the CPU in case of a power failure.

Function Mechanisms of this Application

Program structure of S7 CPU and ET200S CPU with integrated PN interface

OB121 call

In this OB the "MODBUSPN" function block also has to be called.
The OB121 makes an entry in the diagnostic buffer when there is no valid licence.
Additionally, without a valid licence the SF (system error) error LED of the CPU will flash. If OB121 was not configured the CPU will go to stop mode after start-up.

Structure of Modbus DB parameter

In the Modbus DB parameter the Modbus parameters, the send and receive data buffer and the connection parameters for the TCP/IP connection are stored. These four areas of the DB parameter can be seen in the table below.

For each Modbus connection all four areas each have to be created in the DB parameter.

Table 4-1

Address	Comment
+0.0	Start of structure for connection 1
+0.0	Connection parameters
+64.0	Modbus parameters
+130.0	Internal send buffer
+390.0	Internal receive buffer
+650.0	End of structure for connection 1
+650.0	Start of structure for connection 2
+650.0	Connection parameters
+714.0	Modbus parameters
+780.0	Internal send buffer
+1040.0	Internal receive buffer
+1300.0	End of structure for connection 2
	...
$(i-1)*650$	Start of structure for connection i
	Connection parameters
	Modbus parameters
	Internal send buffer
	Internal receive buffer
$i*650$	End of structure for connection i

Parameters of "MODBUSPN" function block

The figure below shows all input and output parameters of the "MODBUSPN" function block.

Figure 4-18

```
CALL "MODBUSPN" , DB1000
  ID           :=
  DB_PARAM    :=
  RECV_TIME   :=
  CONN_TIME   :=
  KEEP_ALIVE  :=
  ENQ_ENR     :=
  DISCONNECT  :=
  REG_KEY     :=
  LICENSED    :=
  BUSY        :=
  CONN_ESTABLISHED :=
  DONE_MDR    :=
  ERROR       :=
  STATUS_MODEBUS :=
  STATUS_CONN :=
  STATUS_FUNC :=
  IDENT_CODE  :=
  UNIT        :=
  DATA_TYPE  :=
  START_ADDRESS :=
  LENGTH      :=
  TI          :=
  WRITE_READ  :=
```

Copyright © Siemens AG 2009 All rights reserved

Table 4-2

Parameters	Decl.	Type	Description	Value range	Initial-ization
ID	IN	WORD	Connection ID has to be identical with corresponding <i>id</i> parameter in DB parameter MODBUS-PARAM	1 to 4095 W#16#1 to W#16#FFF	yes
DB_PARAM	IN	BLOCK_DB	Number of DB parameter MODBUS_PARAM	CPU dependent	yes
RECV_TIME	IN	TIME	Monitoring time for the receipt of data from peer The minimum time that can be set is 100ms	T#100ms to T#+24d20h3 1m23s647ms	no
CONN_TIME	IN	TIME	Monitoring time for setting or clearing connection The minimum time that can be set is 100ms	T#100ms to T#+24d20h3 1m23s647ms	no
KEEP_ALIVE	IN	TIME	not used		
ENQ_ENR	IN	BOOL	S7 is client: trigger job at positive edge	TRUE FASLE	no

Function Mechanisms of this Application

Program structure of S7 CPU and ET200S CPU with integrated PN interface

Parameters	Decl.	Type	Description	Value range	Initial-ization
			S7 is server: ready to receive at positive level		
DISCONNECT	IN	BOOL	S7 is client: TRUE: after receiving the response frame the connection is cleared down S7 is server: TRUE: for ENQ_ENR = FALSE the connection is to be cleared down	TRUE FALSE	no
REG_KEY	IN	STRING [17]	Registration key (activation code) for licencing	Character	no
LICENSED	OUT	BOOL	License status of block Block is lincensed Block is not licensed	TRUE FALSE	no
BUSY	OUT	BOOL	Processing status of T functions (TCON, TDISCON, TSEND or TRCV) In process Not in process	TRUE FALSE	no
CONN_ESTABLISHED	OUT	BOOL	Connection established Connection cleared down	TRUE FASLE	no
DONE_NDR	OUT	BOOL	S7 is client: TRUE: enabled job was completed without errors S7 is server: TRUE: requirement was carried out and answered by client	TRUE FALSE	no
ERROR	OUT	BOOL	An error has occurred. There was no error	TRUE FALSE	no
STATUS_MODBUS	OUT	WORD	Error number for protocol error for processing Modbus frames	0 to FFFF	no
STATUS_CONN	OUT	WORD	Error number for connection error when processing T blocks (TCON, TSEND, TRCV, TDISCON)	0 to FFFF	no
STATUS_FUNC	OUT	STRING [8]	Name of function which caused error on STATUS_MODBUS or STATUS_CONN	Character	no
IDENT_CODE	OUT	STRING [18]	Identification number for licensing With this identifier you can apply for the REG_KEY activation code for your license.	Character	no
UNIT	IN/ OUT	BYTE	Unit identifier (INPUT for CLIENT function, OUTPUT for server function)	0 to 255 B#16#0 to B#16#FF	no
DATA_TYPE	IN/	BYTE	Data type to be processed:		no

Function Mechanisms of this Application
Program structure of S7 CPU and ET200S CPU with integrated PN interface

Parameters	Decl.	Type	Description	Value range	Initial-ization
	OUT		(INPUT for CLIENT function, OUTPUT for SERVER function) Coils Inputs Holding Register Input register	1 2 3 4	
START_ADDRESS	IN/ OUT	WORD	MODBUS start address (INPUT for CLIENT function, OUTPUT for SERVER function)	0 to 65535 W#16#0000 to W#16#FFFF	no
LENGTH	IN/ OUT	WORD	Number of values to be processed (INPUT for CLIENT function, OUTPUT for SERVER function) <u>Coils</u> read function write function <u>Inputs</u> read function <u>Holding register</u> read function write function <u>Input register</u> read function	1 to 2000 1 to 800 1 to 2000 1 to 125 1 to 100 1 to 125	no
TI	IN/ OUT	WORD	Transaction identifier (INPUT for CLIENT function, OUTPUT for server function)	0 to 65535 W#16#0 to W#16#FFFF	no
WRITE_READ	IN/ OUT	BOOL	Write access or read access (INPUT for CLIENT function, OUTPUT for SERVER function)	TRUE FALSE	no

The parameters of the MODBUSPN function block are divided into two groups:

- initialization parameters
- runtime parameters

The initialization parameters are only evaluated for calls in OB100 and are transferred to the instance DB. Initialization parameters are marked with "yes" in the table above in the initialization column. A change of initialization parameters during operation is of no consequence. When these parameters are changed, e.g. during test mode, the instance DB (I-DB) of the CPU has to be reinitialized by STOP -> RUN.

The runtime parameters can be changed during cyclical operation. However, in "S7 is client" mode, it does not make sense to change the input parameters while a job is being processed. The preparations for the next job and the thus connected changes of parameters should only be carried out when the previous job was completed with DONE_NDR or ERROR. In the "S7 is server" mode the output parameters can only be evaluated when DONE_NDR was set.

Function Mechanisms of this Application

Program structure of S7 CPU and ET200S CPU with integrated PN interface

Output parameters are dynamical displays and they are therefore only pending for one CPU cycle. For possible further processing or for display in the variable table they have to be copied to other memory areas.

The generation of the function code is via the parameters DATA_TYPE, LENGTH and WRITE_READ. Possible combinations to generate a function code are shown in the table below.

Table 4-3

Data type	DATA_TYPE	Function	LENGTH	WRITE_READ	Function code
Coils	1	read	1 to 2000	false	1
Coils	1	write	1	true	5
Coils	1	write	1	true	15
Coils	1	write	>1 to 800	true	15
Inputs	2	read	1 to 2000	false	2
Holding register	3	read	1 to 125	false	3
Holding register	3	write	1	true	6
Holding register	3	write	1	true	16
Holding register	3	write	>1 to 100	true	16
Input register	4	read	1 to 125	false	4

4.1.2 Configuration explanations

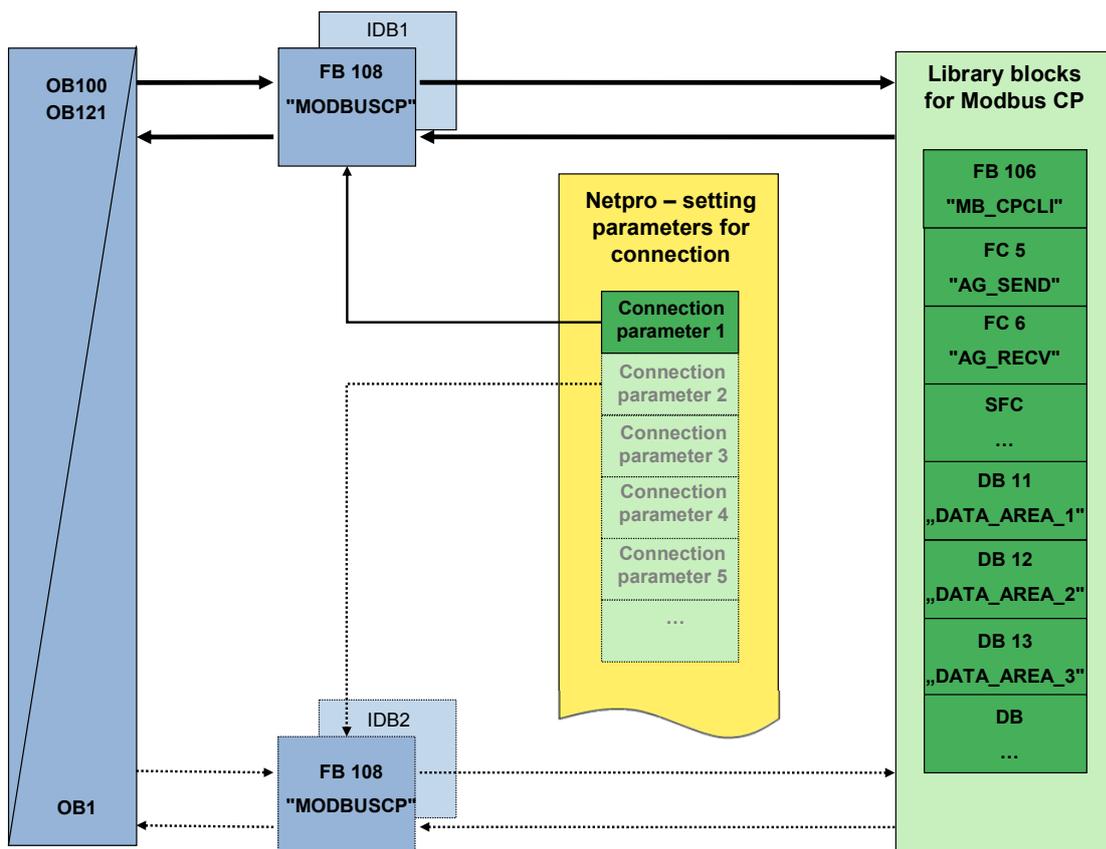
In this application example read and write data exchange is to be realized between Modbus client and server. When the S7 CPU with integrated PN interface takes on the client function, data is taken from the server via read access. This data is copied internally to other memory spaces and then given back to the Modbus server via write access.

When the S7 CPU with integrated PN interface takes on the server function, it's not possible to start client or server jobs. In this case the S7 CPU is waiting for requests from the Modbus client (M340).

4.2 Program structure of S7 CPU with CP

The main difference of the program structure of the Modbus TCP solution via a CP and the PN CPU is the configuration of the connection parameters. It is exclusively done in STEP7 NetPro. There is no data structure in which the connection parameters or the Modbus parameters could be entered. The Modbus parameters are transferred in OB1.

Figure 4-19



Copyright © Siemens AG 2009. All rights reserved.

4.2.1 Program details on MODBUS CP blocks

A separate call of the "MODBUSCP" function block with individual DB instance is necessary for each connection. Please note that the call of the "MODBUSCP" function block has to take place in OB1, OB121 and in OB100 with the same DB instance per connection.

Modbus library blocks

The figure below displays all S7 program blocks which are necessary for Modbus TCP via CP.

To be able to use Modbus TCP for CPs in a program, only the "MODBUSCP" block has to be called. The other blocks are called internally in the "MODBUSCP" function block.

Figure 4-20: Modbus TCP CP blocks from the sample program

Object name	Symbolic name	Created in language	Size in the work me...	Type	Versio
System data	---	---	---	SDB	---
OB1	CYCL_EXC	STL	840	Organization Block	0.1
OB100	COMPLETE RESTART	STL	1176	Organization Block	0.0
OB121	PROG_ERR	STL	38	Organization Block	0.1
FB106	MB_CPCLI	SCL	10324	Function Block	1.0
FB107	MB_CPSRV	SCL	10126	Function Block	1.0
FB108	MODBUSCP	SCL	5548	Function Block	1.0
FC5	AG_SEND	STL	1664	Function	4.2
FC6	AG_RECV	STL	1206	Function	4.7
FC10	EQ_STRNG	STL	152	Function	1.1
DB1	CONTROL DAT	DB	200	Data Block	0.1
DB3	LICENSE_DB	DB	56	Data Block	0.1
DB11	DATA_AREA_1	DB	1036	Data Block	0.1
DB12	DATA_AREA_2	DB	436	Data Block	0.1
DB13	DATA_AREA_3	DB	636	Data Block	0.1
DB14	DATA_AREA_5	DB	114	Data Block	0.1
DB15	DATA_AREA_6	DB	114	Data Block	0.1
DB16	DATA_AREA_7	DB	114	Data Block	0.1
DB108	IDB_MODBUS	DB	1354	Instance data block ...	0.0
Server_Job	Server_Job	---	---	Variable Table	0.1
SFB4	TON	STL	---	System function block	1.0
SFC6	RD_SINFO	STL	---	System function	1.0
SFC20	BLKMOV	STL	---	System function	1.0
SFC24	TEST_DB	STL	---	System function	1.0
SFC51	RDSYSST	STL	---	System function	1.0
SFC52	WR_USMSG	STL	---	System function	1.0

OB1 call

In OB1 the cyclical calls of the "MODBUSCP" function block are made as well as of other user specific functions and function blocks. This is where runtime parameters are transferred to the "MODBUSCP" function block.

OB100 call

In OB100 the "MODBUSCP" function block is initialized. OB100 is completed once when there is a STOP → RUN transfer or when restarting the CPU in case of a power failure.

OB121 call

In this OB the "MODBUSCP" function block also has to be called. The OB121 makes an entry in the diagnostic buffer when there is no valid licence. Additionally, without a valid licence the SF (system error) error LED of the CPU will flash. If OB121 was not configured the CPU will go to stop mode after start-up.

Parameters of the "MODBUS" function block

This section describes the input and output parameters of the "MODBUSCP" function block.

Figure 4-21: "MODBUSCP" function block call in STL

```
CALL "MODBUSCP" , DB1000
  id      :=
  laddr   :=
  MONITOR :=
  REG_KEY :=
  server_client:=
  single_write :=
  data_type_1 :=
  db_1    :=
  start_1 :=
  end_1   :=
  data_type_2 :=
  db_2    :=
  start_2 :=
  end_2   :=
  data_type_3 :=
  db_3    :=
  start_3 :=
  end_3   :=
  data_type_4 :=
  db_4    :=
  start_4 :=
  end_4   :=
  data_type_5 :=
  db_5    :=
  start_5 :=
  end_5   :=
  data_type_6 :=
  db_6    :=
  start_6 :=
  end_6   :=
  data_type_7 :=
  db_7    :=
  start_7 :=
  end_7   :=
  data_type_8 :=
  db_8    :=
  start_8 :=
  end_8   :=
  ENQ_ENR :=
  LICENSED :=
  BUSY    :=
  DONE_MDR :=
  ERROR   :=
  STATUS  :=
  STATUS_FUNC :=
  IDENT_CODE :=
  UNIT    :=
  DATA_TYPE :=
  START_ADDRESS:=
  LENGTH  :=
  TI      :=
  WRITE_READ :=
```

Function Mechanisms of this Application
Program structure of S7 CPU with CP

Table 4-15

Parameters	Decl.	Type	Description	Value range	Initializ ation
id	IN	WORD	Connection ID in accordance with the configuration	1 to 64 W#16#1 to W#16#40	yes
laddr	IN	WORD	Input-Address of the CP in HW Config	PLC dependent	yes
MONITOR	IN	TIME	Monitoring Time: Wait for data from communication partner Shortest adjustable time is 20ms	T#20ms to T#+24d20h31m2 3s647ms	no
REG_KEY	IN	STRING [17]	Registration key to activate the licence	Character	no
server_client	IN	BOOL	CP/FB operates in server mode CP/FB operates in client modet	TRUE FALSE	yes
single_write	IN	BOOL	Write 1 Coil/Register: Function code 5 or. 6 Function code 15 or. 16	TRUE FALSE	yes
data_type_1	IN	BYTE	1st data area: daten type Coils Inputs Holding Register Input Register	1 2 3 4	yes
db_1	IN	WORD	1st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_1	IN	WORD	1st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_1	IN	WORD	1st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
data_type_2	IN	BYTE	2st data area: daten type Coils Inputs Holding Register Input Register	0 to 4	yes
db_2	IN	WORD	2st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_2	IN	WORD	2st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_2	IN	WORD	2st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
data_type_3	IN	BYTE	3st data area: daten type Coils Inputs	0 to 4	yes

Function Mechanisms of this Application

Program structure of S7 CPU with CP

Parameters	Decl.	Type	Description	Value range	Initializ ation
			Holding Register Input Register		
db_3	IN	WORD	3st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_3	IN	WORD	3st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_3	IN	WORD	3st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
data_type_4	IN	BYTE	4st data area: daten type Coils Inputs Holding Register Input Register	0 to 4	yes
db_4	IN	WORD	4st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_4	IN	WORD	4st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_4	IN	WORD	4st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
data_type_5	IN	BYTE	5st data area: daten type Coils Inputs Holding Register Input Register	0 to 4	yes
db_5	IN	WORD	5st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_5	IN	WORD	5st data area: first Modbus address in this DB	0 to 65535 W#16#0000 bis W#16#FFFF	yes
end_5	IN	WORD	5st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
data_type_6	IN	BYTE	6st data area: daten type Coils Inputs Holding Register Input Register	0 to 4	yes
db_6	IN	WORD	6st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_6	IN	WORD	6st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_6	IN	WORD	6st data area:	0 to 65535	yes

Function Mechanisms of this Application
Program structure of S7 CPU with CP

Parameters	Decl.	Type	Description	Value range	Initializ ation
			last Modbus address in this DB	W#16#0000 to W#16#FFFF	
data_type_7	IN	BYTE	7st data area: daten type Coils Inputs Holding Register Input Register	0 to 4	yes
db_7	IN	WORD	7st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_7	IN	WORD	7st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_7	IN	WORD	7st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
data_type_8	IN	BYTE	8st data area: daten type Coils Inputs Holding Register Input Register	0 to 4	yes
db_8	IN	WORD	8st data area: data block number	1 to 65535 W#16#1 to W#16#FFFF	yes
start_8	IN	WORD	8st data area: first Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
end_8	IN	WORD	8st data area: last Modbus address in this DB	0 to 65535 W#16#0000 to W#16#FFFF	yes
ENQ_ENR	IN	BOOL	CP is client: Initiate request at TRUE signal CP is server: Ready to receive at TRUE signal	TRUE FALSE	no
LICENSED	OUT	BOOL	License state of the function block: Block is licensed Block is not licensed	TRUE FALSE	no
BUSY	OUT	BOOL	Operating state of the functions AG_SEND and AG_RECV Job processing No job processing active	TRUE FALSE	no
DONE_NDR	OUT	BOOL	CP is client: Active request finished without errors CP is server: Request from the client was executed and answered	TRUE FALSE	no
ERROR	OUT	BOOL	An error has occurred: yes:	TRUE FALSE	no

Function Mechanisms of this Application

Program structure of S7 CPU with CP

Parameters	Decl.	Type	Description	Value range	Initializ ation
			no:		
STATUS	OUT	WORD	Error number	0 to FFFF	no
STATUS_FUNC	OUT	STRING [8]	Name of the function, which causes the error at STATUS	Character	no
IDENT_CODE	OUT	STRING [18]	Identification for licensing Please order your license with this identification string.	Character	no
UNIT	IN/ OUT	BYTE	Unit identification (INPUT if in CLIENT mode, OUTPUT if in SERVER mode)	0 to 255 B#16#0 to B#16#FF	no
DATA_TYPE	IN/ OUT	BYTE	Data type to be accessed: (INPUT if in CLIENT mode, OUTPUT if in SERVER mode) Coils: Inputs: Holding registers: Input registers:	 1 2 3 4	no
START_ ADDRESS	IN/ OUT	WORD	MODBUS start address (INPUT if in CLIENT mode, OUTPUT if in SERVER mode)	0 to 65535	no
LENGTH	IN/ OUT	WORD	Number of registers to be processed (INPUT if in CLIENT mode, OUTPUT if in SERVER mode) <u>Coils:</u> reading function writing function. <u>Inputs:</u> reading function <u>Holding Reg.:</u> reading function writing function. <u>Input Reg.:</u> reading function	 1 to 2000 1 to 1968 1 to 2000 1 to 125 1 to 123 1 to 125	no
TI	IN/ OUT	WORD	Transaction Identifier (INPUT if in CLIENT mode, OUTPUT if in SERVER mode)	0 to 65535	no
WRITE_READ	IN/ OUT	BOOL	INPUT if in CLIENT mode, OUTPUT if in SERVER mode: Write access: Read access:	 TRUE FALSE	no

The parameters of the “MODBUSCP” function block are divided into two groups:

- initialization parameters
- runtime parameters

The initialization parameters are only evaluated for the initial run of the “MODBUSCP” function block and are transferred to the DB instance. Initialization parameters are marked with "yes" in the table above in the initialization column. A change of initialization parameters during operation is of no consequence. When these parameters are changed, e.g. during test mode, the instance DB (I-DB) of the CPU has to be reinitialized by STOP → RUN.

Runtime parameters can be changed during cyclical operation. It is not advisable to change the input parameters whilst a job is being processed. The preparations for the next job and the thus connected changes of parameters should only be carried out when the previous job was completed with DONE_NDR or ERROR. Output parameters should only be evaluated with set DONE_NDR.

The generation of the function code is via the parameters LENGTH, single_write and WRITE_READ. Possible combinations to generate a function code are shown in the table below.

Table 4-4

Data type	DATA_TYPE	Function	LENGTH	WRITE_READ	single_write	Function code
Coils	1	read	1 to 2000	false	irrelevant	1
Coils	1	write	1	true	TRUE	5
Coils	1	write	1	true	FALSE	15
Coils	1	write	>1 to 1968	true	irrelevant	15
Inputs	2	read	1 to 2000	false	irrelevant	2
Holding Register	3	read	1 to 125	false	irrelevant	3
Holding Register	3	write	1	true	TRUE	6
Holding Register	3	write	1	true	FALSE	16
Holding Register	3	write	>1 to 123	true	irrelevant	16
Input Register	4	read	1 to 125	false	irrelevant	4

4.2.2 Configuration explanations

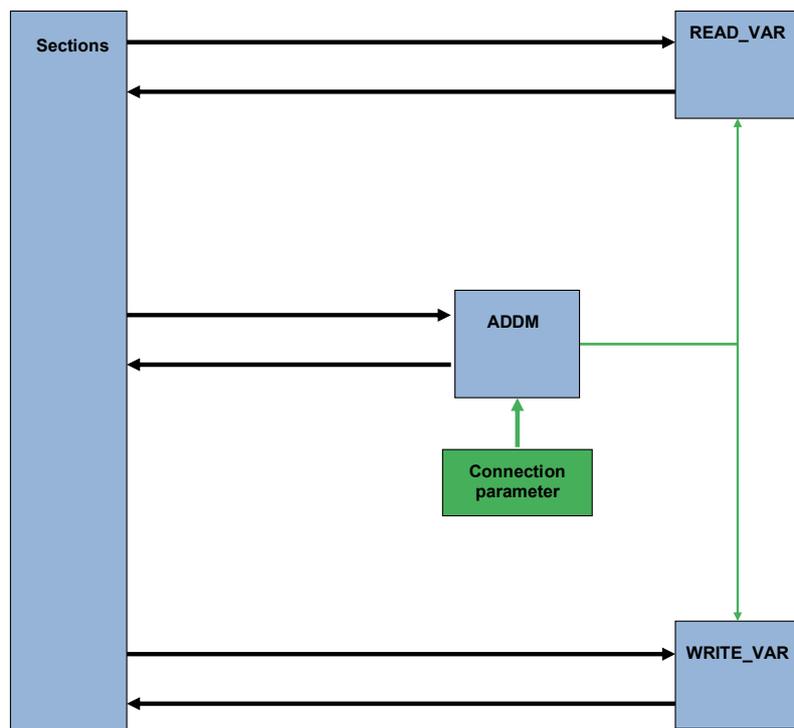
In this application example read and write data exchange is to be realized between Modbus client and server. When the S7 CPU with CP takes on the client function, data is taken from the server via read access. This data is copied internally to other memory spaces and then given back to the Modbus server via write access.

When the S7 CPU with CP takes on the server function, it's not possible to start client or server jobs. In this case the S7 CPU is waiting for requests from the Modbus client (M340).

4.3 Program structure of Modicon M340

The following program structure only shows the Modbus client application of Modicon M340. If the Modicon M340 is to act as Modbus server, no special block call is necessary in the user program.

Figure 4-22



4.3.1 Program details on Modicon M340 blocks

READ_VAR

This function can read data from a Modbus server. The Modbus server is addressed via IP address.

WRITE_VAR

The WRITE_VAR function can write data in a Modbus server. For this function the IP address of the Modbus server is also necessary.

ADDM

This function changes a string to an address which can be used by communication functions (READ_VAR, WRITE_VAR). The IP address is assigned to the string as initial value in the variable declaration.

4.3.2 Configuration explanations

In this application example read and write data exchange is to be realized between Modbus client and server. When the Modicon M340 takes on the client function, data is taken from the server via read access. This data is copied internally to other memory spaces and then given back to the Modbus server via write access.

When the Modicon M340 takes on the server function, it's not possible to start client or server jobs. In this case the M340 is waiting for requests from the Modbus client (S7).

5 Installation

This chapter describes which hardware and software components have to be installed. It is essential to read the instructions, manuals and any delivery information supplied with the respective products.

5.1 Installation of the hardware

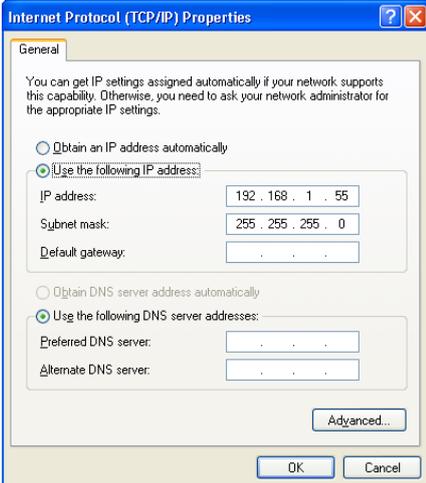
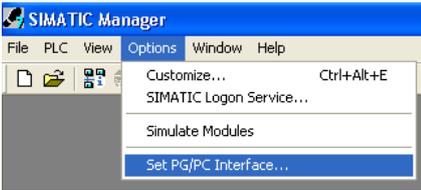
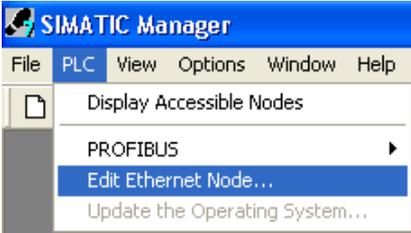
For the required hardware components, please refer to chapter 2.3 "Hardware and software components used". The table below gives an overview of the IP addresses as well as the devices which are used in the hardware setup of the application. Both S7 CPUs with integrated PN interface and the S7 CPU with CP have the same IP address. These components are never connected to the Ethernet network at the same time.

Table 5-17

Device	IP address
CPU319-3 PN/DP	192.168.1.1
IM 151-8 PN/DP CPU	192.168.1.1
CPU315-2 PN/DP	none
CP343-1 Lean	192.168.1.1
SCALANCE X108	none
PS307 24 V/5 A	none
Modicon M340	192.168.1.10
PG/PC	192.168.1.55

For the hardware configuration, please follow the instructions listed in the table below:

Table 5-1

No.	Instruction	Comment
1	Mount the modules on a DIN rail	
2	Connect the CPU315-2PN/DP with the CP343-1 lean via the backplane bus	
3	Connect all modules with 24V power supply	If necessary use terminal strips or several power supplies
4	Connect the following modules via Ethernet: * CP343-1 Lean or CPU319-3 PN/DP or IM 151-8 PN/DP CPU with Scalance X108 *Modicon M340 with Scalance X108 *PG/PC with Scalance X108	Use the corresponding S7 CPU depending on the application example.
5	Assign the following IP address 192.168.1.55 (Subnet mask: 255.255.255.0) to the Ethernet interface of the PG/PC. Start → Settings → Control Panel → Network Connections → LAN Properties → TCP/IP Properties	
6	Start the SIMATIC manager and select the TCP/IP interface Options → Set PG/PC interface	
7	Assign the IP addresses from table 5-15 to the S7 CPUs. To do this, use the SIMATIC manager and search the available nodes in the menu "PLC → Edit Ethernet Node".	

Note The guidelines for the components used here always have to be met.

Installation

Installation of the hardware

Installation of the software

Table 5-19

No.	Instruction	Remarks
1	Installing STEP7	Installation is automatic. Follow the setup instructions. Installation of STEP7 is also described in the manual "Programming with STEP7 V5.4" (see 3).
2	Installing the Modbus TCP library	Installation is automatic. Follow the setup instructions. A description of the installation of the Modbus TCP library can also be found in the manuals for Modbus TCP (see 4).
3	Installing the Modbus TCP Wizard	In the manual of the Modbus TCP Wizards you will find information on installing and configuring a Modbus TCP communication (see 5).
4	Installing of Unity Pro XL	Installation is automatic. Follow the setup instructions.

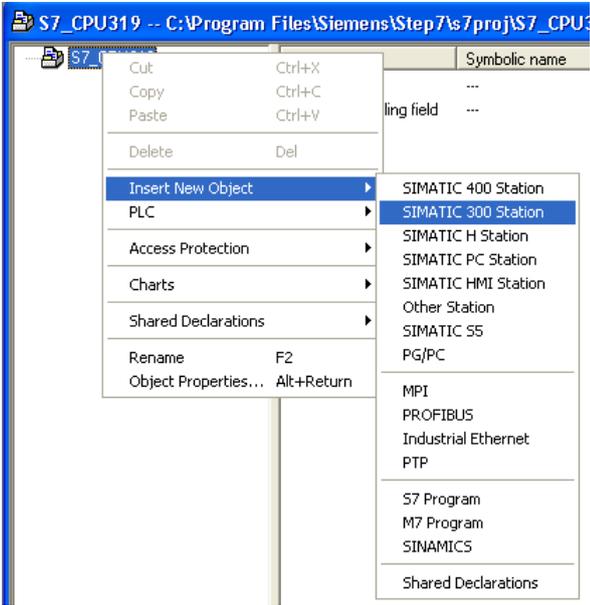
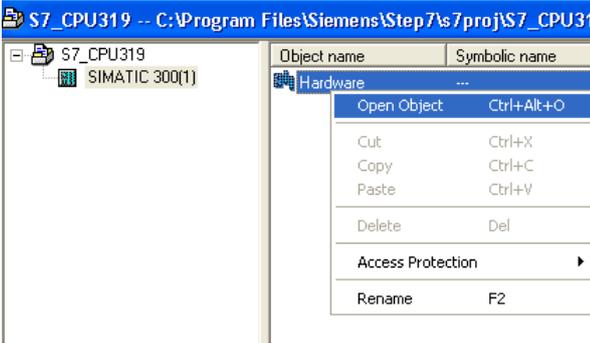
6 Startup of the Application

6.1 Configuration of CPU319-3 PN/DP

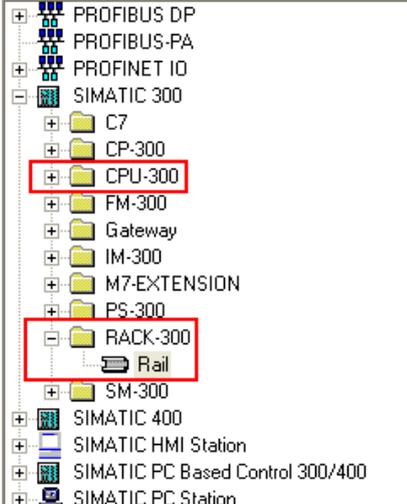
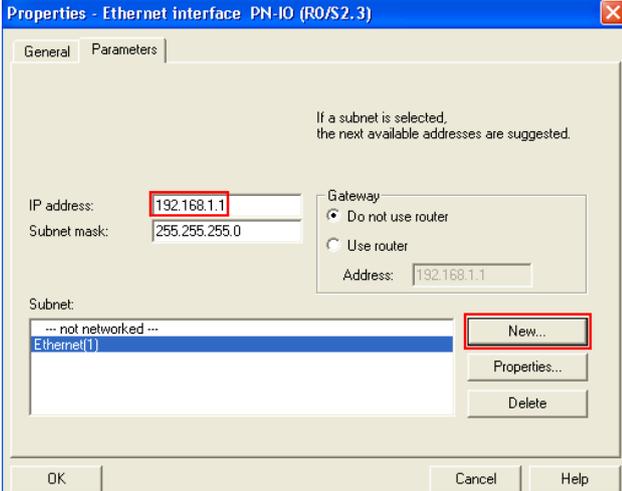
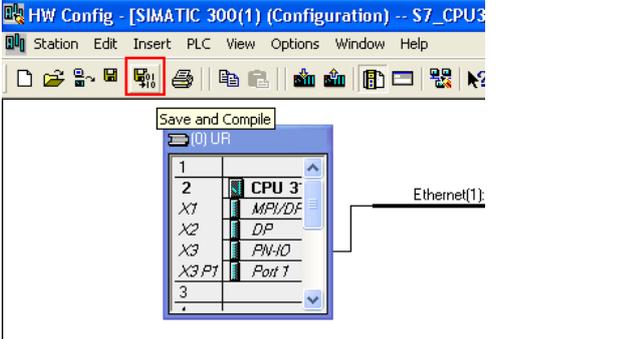
This chapter describes the configuration of the CPU319-3 PN/DP. Using this configuration description you can configure the application examples of the CPU319-3 PN/DP yourself and start up the CPU.

6.1.1 Hardware configuration

Table 6-20

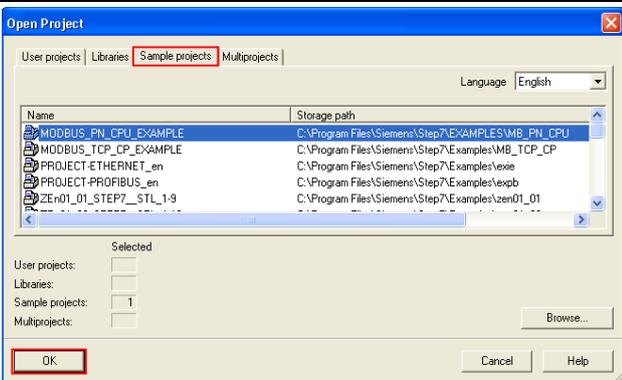
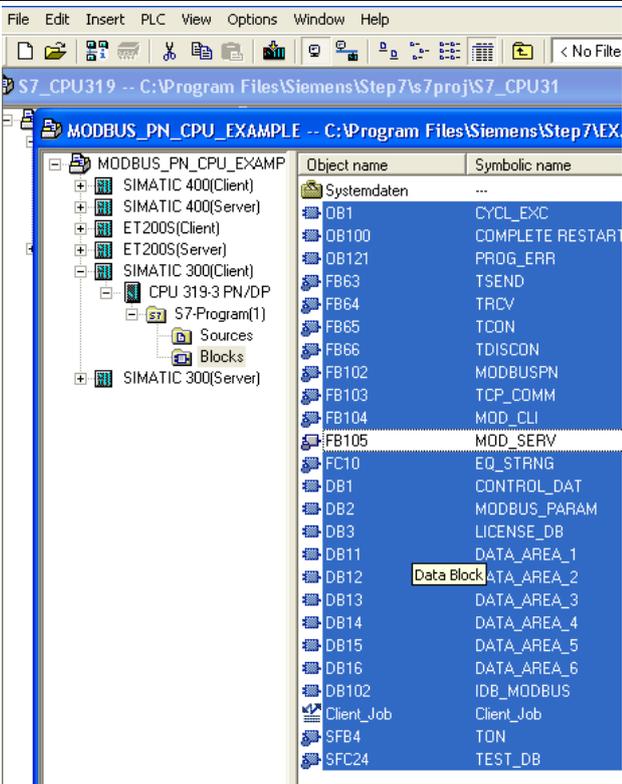
No.	Instruction	Comment
1	Create a new project in the SIMATIC manager.	File→new
2	Right click the just created project and insert the SIMATIC 300 station via the menu "Insert New Object → SIMATIC 300 Station".	
3	Right click "Hardware" and select "Open Object" to open the hardware configuration	

Startup of the Application
Configuration of CPU319-3 PN/DP

No.	Instruction	Comment
4	Insert a rack in the hardware configuration via drag&drop from the hardware catalog.	
5	Insert a CPU319-3 PN/DP V2.7 from the hardware catalog via drag&drop on slot 2 of the rack.	Hardware catalog: SIMATIC 300→CPU 300→CPU319-3 PN/DP→6ES7 318-3EL00-0AB0→V2.7
6	Configure the Ethernet interface of the CPU319-3 PN/DP: <ul style="list-style-type: none"> - Assign IP Address 192.168.1.1 - create new Ethernet network <p>Note: this menu opens automatically after inserting the module.</p> <p>A Profibus network does not need to be created.</p>	
7	Compile and save the hardware configuration.	

6.1.2 Insert Modbus TCP blocks into project

Table 6-21

No.	Instruction	Comment
1	Open the Modbus TCP examples from the library for PN CPUs: File → Open → Sample projects	
2	<p>Insert S7 300 CPU library blocks into S7 project (depending whether S7 CPU is to be client or server, FB104 or FB105 has to be copied)</p> <p>In the sample project the blocks can be marked and inserted in the block container of your S7 Modbus project via drag&drop.</p> <p>OB1 in the S7 Modbus project has to be overwritten with OB1 of the sample project.</p>	

Copyright © Siemens AG 2009. All rights reserved.

6.1.3 Configuring Modbus TCP connections

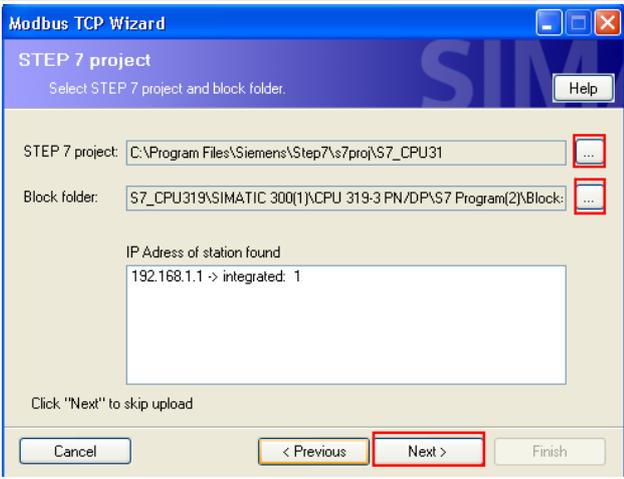
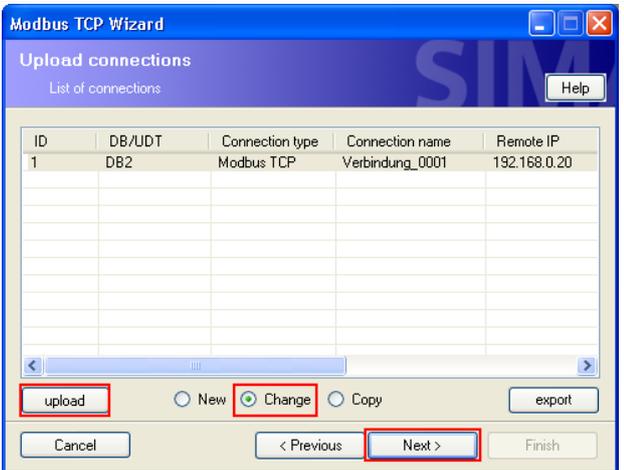
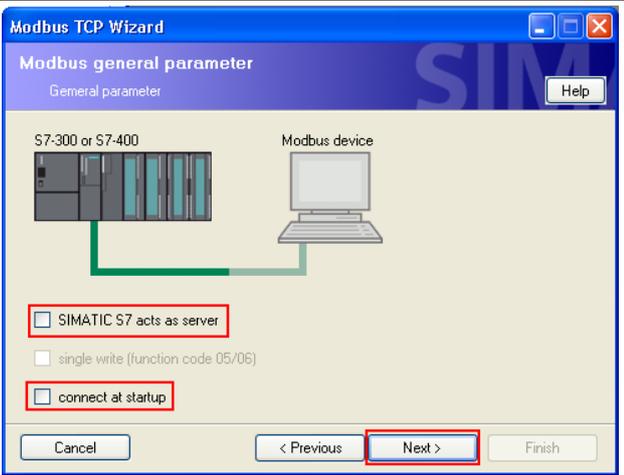
The Modbus TCP connection can either be configured with the Modbus TCP Wizard or by manually processing the DB parameter (MODBUS_PARAM). Manual processing of MODBUS_PARAM is described in the Modbus TCP manual for PN CPUs (see [4](#)).

Modbus TCP Wizard

This section explains how a Modbus connection for S7 CPUs with integrated PN interface is configured with the aid of the Modbus TCP Wizard. When using the Wizard, please make sure that the Modbus (MODBUS_PARAM) DB parameter is not open in the SIMATIC manager.

Startup of the Application Configuration of CPU319-3 PN/DP

Table 6-22

No.	Instruction	Comment
1	<p>Start the Modbus TCP Wizard (Start → SIMATIC → ModbusTCPWizard) and click the "Next" button to get to the "STEP 7 project" menu.</p> <p>In this dialog you select the S7 project just created.</p>	
2	<p>Read existing project</p> <p>Click the "upload" button to load a configured connection of your project into the Modbus TCP Wizard.</p> <p>You can process this connection by enabling the "Change" option.</p>	
3	<p>Select client/server mode</p> <p>Enable the function "SIMATIC S7 acts as server" when the S7 CPU is to act as Modbus TCP server.</p> <p>Enable the "connect at startup" function so that the CPU can automatically create a Modbus TCP connection to the remote communication partner after startup.</p>	

Startup of the Application
Configuration of CPU319-3 PN/DP

No.	Instruction	Comment																																													
4	<p>Select the Modbus TCP partner and the internal interface (select your hardware: e.g.: Integrated 319)</p> <p>Client application: enter the remote IP address (192.168.1.10). The client is always the active partner.</p> <p>Server application: the server is configured as passive communication partner. Doing this, each client can connect with the server when an unspecified connection partner was selected.</p>																																														
5	<p>Selection of port</p> <p>Client application: selection of remote port (port 502 for Modicon M340)</p> <p>Server application: selection of local port (port: 502)</p>																																														
6	<p>Definition of Modbus register</p> <p>Client application: 1 Holding Registers from 100 to 399 2 Input Registers from 400 to 600</p> <p>Server application: 1 Holding Registers from 0 to 500 2 Holding Registers from 720 to 900</p>	<table border="1"> <thead> <tr> <th>Data area</th> <th>Data type</th> <th>Start address</th> <th>End address</th> <th>DB number</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Holding Registers</td> <td>100</td> <td>399</td> <td>11</td> </tr> <tr> <td>2</td> <td>Input Registers</td> <td>400</td> <td>600</td> <td>12</td> </tr> <tr> <td>3</td> <td>not used</td> <td>720</td> <td>1000</td> <td>13</td> </tr> <tr> <td>4</td> <td>not used</td> <td>640</td> <td>1250</td> <td>14</td> </tr> <tr> <td>5</td> <td>not used</td> <td>1700</td> <td>2300</td> <td>15</td> </tr> <tr> <td>6</td> <td>not used</td> <td>1700</td> <td>2300</td> <td>16</td> </tr> <tr> <td>7</td> <td>not used</td> <td>0</td> <td>100</td> <td>7</td> </tr> <tr> <td>8</td> <td>not used</td> <td>0</td> <td>100</td> <td>8</td> </tr> </tbody> </table>	Data area	Data type	Start address	End address	DB number	1	Holding Registers	100	399	11	2	Input Registers	400	600	12	3	not used	720	1000	13	4	not used	640	1250	14	5	not used	1700	2300	15	6	not used	1700	2300	16	7	not used	0	100	7	8	not used	0	100	8
Data area	Data type	Start address	End address	DB number																																											
1	Holding Registers	100	399	11																																											
2	Input Registers	400	600	12																																											
3	not used	720	1000	13																																											
4	not used	640	1250	14																																											
5	not used	1700	2300	15																																											
6	not used	1700	2300	16																																											
7	not used	0	100	7																																											
8	not used	0	100	8																																											

Startup of the Application

Configuration of CPU319-3 PN/DP

No.	Instruction	Comment																				
7	Transfer Modbus TCP parameters to MODBUS_PARAM	Click the "Next" button to get to the dialogs that follow. With one click on the "Next" button in the "Connection list" dialog the Modbus TCP data is transferred to the MODBUS_PARAM DB. After transfer click the "Finish" button to exit the wizard.																				
8	Now you have to adjust the DBs for the Modbus data areas in the S7 project. Client application: DB11 ARRAY[0...300] DB12 ARRAY[0...300] Server application: DB11 ARRAY[0...600] DB12 ARRAY[0...300]	<table border="1"> <thead> <tr> <th>Address</th> <th>Name</th> <th>Type</th> <th>Initial value</th> </tr> </thead> <tbody> <tr> <td>0.0</td> <td></td> <td>STRUCT</td> <td></td> </tr> <tr> <td>+0.0</td> <td>DB_VAR</td> <td>ARRAY[0..300]</td> <td>W#16#0</td> </tr> <tr> <td>*2.0</td> <td></td> <td>WORD</td> <td></td> </tr> <tr> <td>=800.0</td> <td></td> <td>END_STRUCT</td> <td></td> </tr> </tbody> </table>	Address	Name	Type	Initial value	0.0		STRUCT		+0.0	DB_VAR	ARRAY[0..300]	W#16#0	*2.0		WORD		=800.0		END_STRUCT	
Address	Name	Type	Initial value																			
0.0		STRUCT																				
+0.0	DB_VAR	ARRAY[0..300]	W#16#0																			
*2.0		WORD																				
=800.0		END_STRUCT																				

6.1.4 Project download

Table 6-23

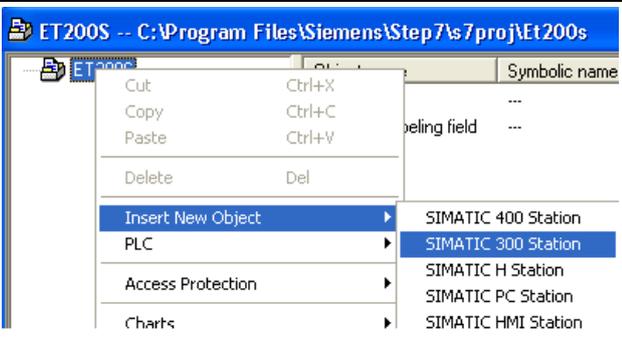
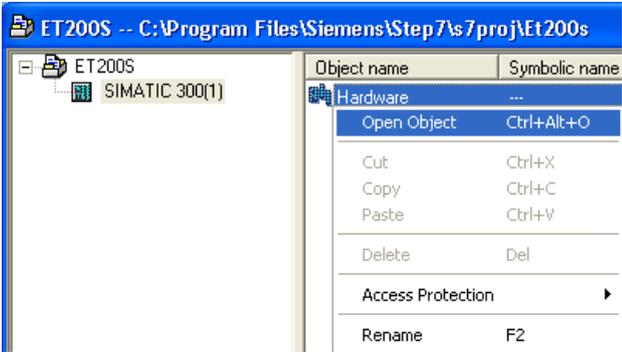
No.	Instruction	Comment						
1	Mark the SIMATIC 300 station in the SIMATIC manager and load the project into the controller.	 <p>The screenshot shows the SIMATIC Manager interface for project S7_CPU319. The menu bar includes File, Edit, Insert, PLC, View, Options, Window, and Help. The toolbar contains various icons, with the 'Download' icon (a computer with a downward arrow) highlighted by a red box. The project tree on the left shows 'S7_CPU319' expanded to 'SIMATIC 300(1)'. The right pane shows the hardware configuration table:</p> <table border="1"> <thead> <tr> <th>Object name</th> <th>Symb</th> </tr> </thead> <tbody> <tr> <td>Hardware</td> <td>...</td> </tr> <tr> <td>CPU 319-3 PN/DP</td> <td>...</td> </tr> </tbody> </table>	Object name	Symb	Hardware	...	CPU 319-3 PN/DP	...
Object name	Symb							
Hardware	...							
CPU 319-3 PN/DP	...							

6.2 Configuration of IM151-8 PN/DP CPU

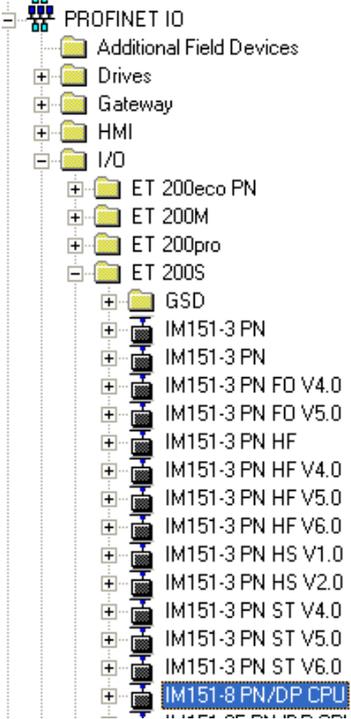
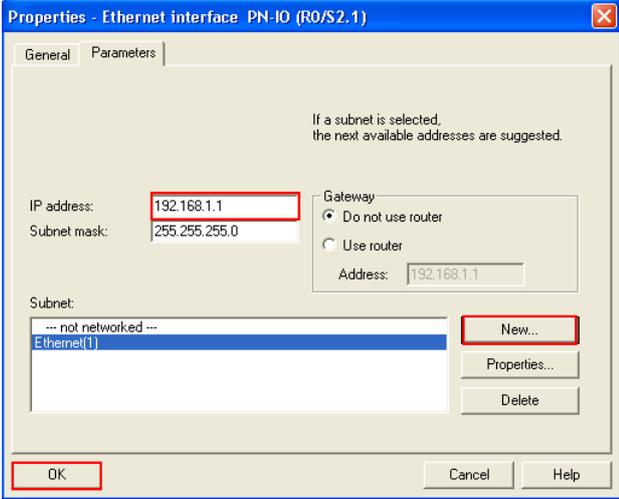
The configuration of the ET200S (IM151-8 PN/DP CPU) is similar to the configuration of the CPU319-3 PN/DP. This is why this chapter only deals with the hardware configuration of the ET200S. Creating and configuring a Modbus TCP project for the ET200S is analog to the project of the CPU319-3 PN/DP. When configuring the MODBUS_PARAM the respective local interface for the ET200S has to be selected in the Modbus TCP Wizard! The other configuration steps in the Modbus TCP wizard do not change.

6.2.1 Hardware configuration

Table 6-24

No.	Instruction	Comment
1	Create a new project in the SIMATIC manager.	File → New
2	Right click the STEP7 project and insert the SIMATIC 300 station via the menu "Insert New Object".	
3	Right click "Hardware" and select "Open Object" to open the hardware configuration of the S7-300 station.	

Startup of the Application
Configuration of IM151-8 PN/DP CPU

No.	Instruction	Comment
4	<p>Add the IM151-8 PN/DP CPU from the hardware catalog via drag&drop.</p> <p>Hardware catalog: SIMATIC 300 → PROFINET IO → I/O → ET200S → IM151-8 PN/DP CPU</p>	
5	<p>Configure the Ethernet interface of ET200S: assign the IP address 192.168.1.1 and create a new Ethernet network</p> <p>Note: this menu opens automatically after inserting the module.</p>	

Startup of the Application
Configuring the CPU315-2 PN/DP with CP343-1 Lean

No.	Instruction	Comment
6	Compile and save the hardware configuration.	

6.3 Configuring the CPU315-2 PN/DP with CP343-1 Lean

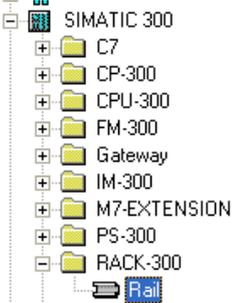
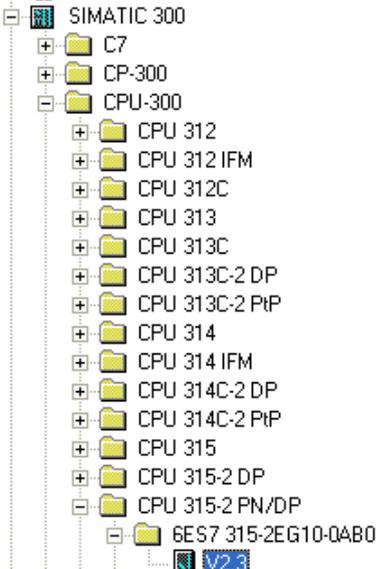
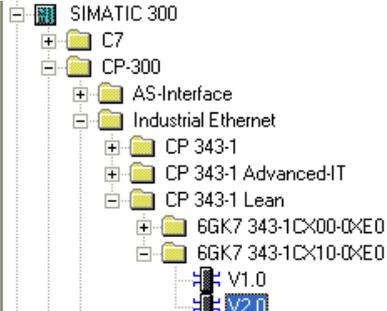
6.3.1 Hardware configuration

Table 6-25

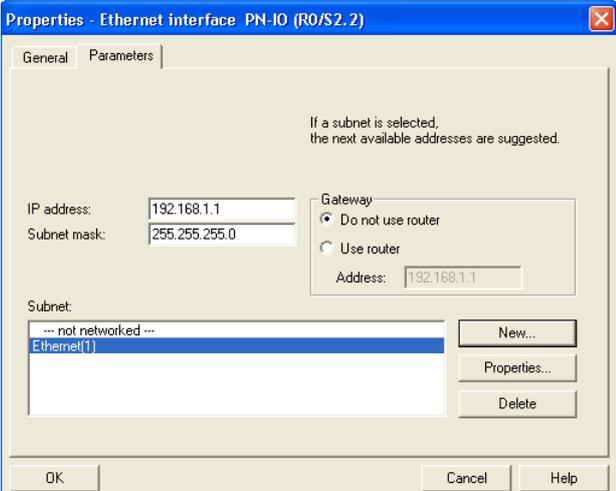
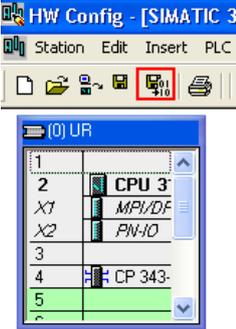
No.	Instruction	Comment
1	Create a new project in the SIMATIC manager.	File→new
2	Right click the STEP7 project and insert the SIMATIC 300 station via the menu "Insert New Object".	
3	Right click "Hardware" and select "Open Object" to open the hardware configuration of the S7-300 station.	

Startup of the Application

Configuring the CPU315-2 PN/DP with CP343-1 Lean

No.	Instruction	Comment
4	Insert a SIMATIC 300 via drag&drop.	
5	<p>Insert a CPU315-2 PN/DP from the hardware catalog via drag&drop on slot 2 of the rack.</p> <p>There is no need to define CPU interfaces. Close the configuration dialogs which open automatically after inserted the CPU.</p> <p>Hardware catalog: SIMATIC 300 → CPU-300 → CPU315-2 PN/DP → 6ES7 315-2EG10-0AB0 → V2.3</p>	
6	<p>Now insert the CP343-1.</p> <p>Hardware catalog: SIMATIC 300 → CP-300 → Industrial Ethernet → CP343-1 Lean → 6GK7 343-1CX10-0XE0 → V2.0</p>	

Startup of the Application
Configuring the CPU315-2 PN/DP with CP343-1 Lean

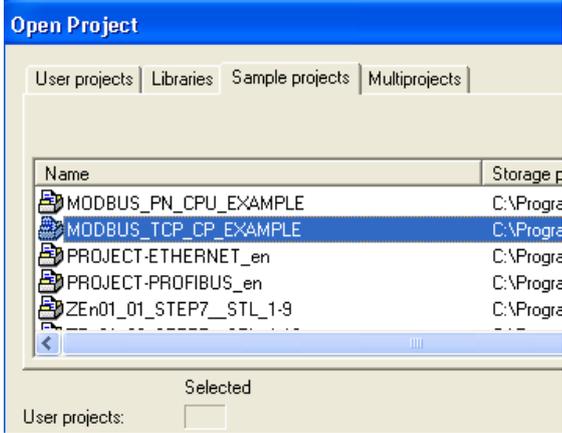
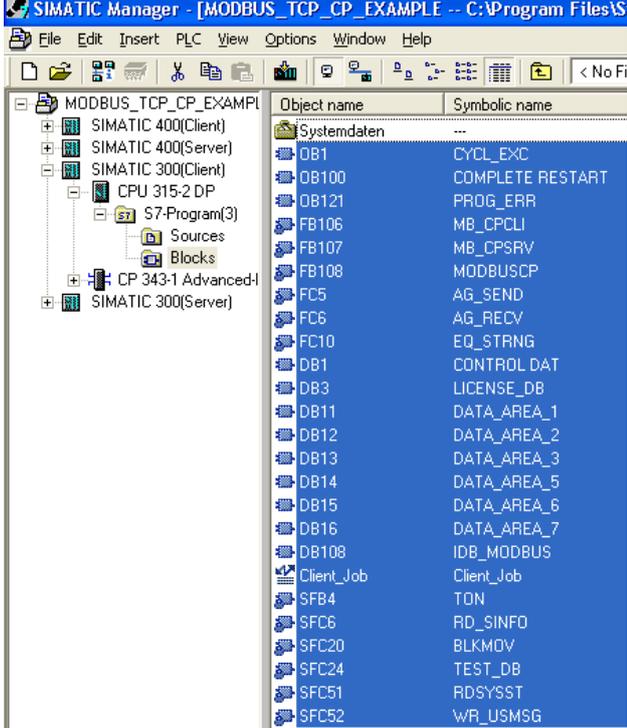
No.	Instruction	Comment
7	<p>Configure the Ethernet interface of the CP343-1:</p> <p>assign the IP address 192.168.1.1 and create a new Ethernet network</p> <p>Note: this menu opens automatically after inserting the module.</p>	
8	<p>Compile and save the hardware configuration.</p>	

Startup of the Application

Configuring the CPU315-2 PN/DP with CP343-1 Lean

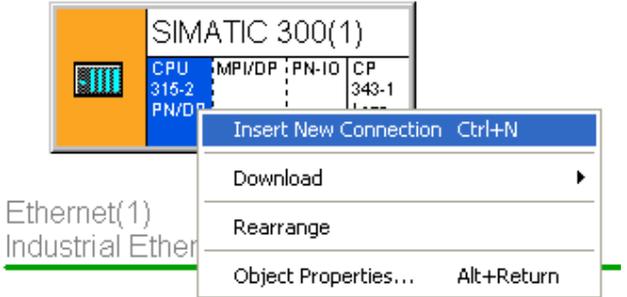
6.3.2 Creating a project for Modbus TCP

Table 6-26

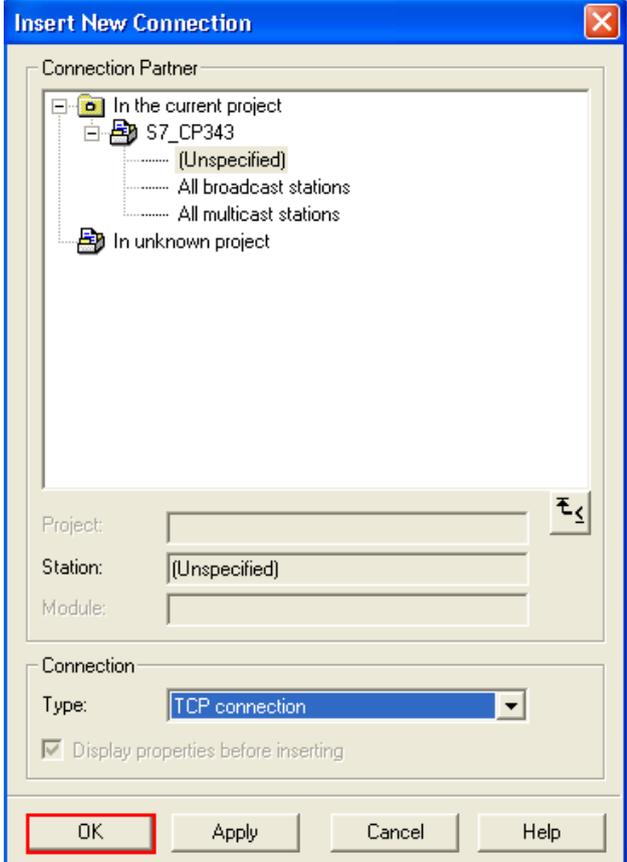
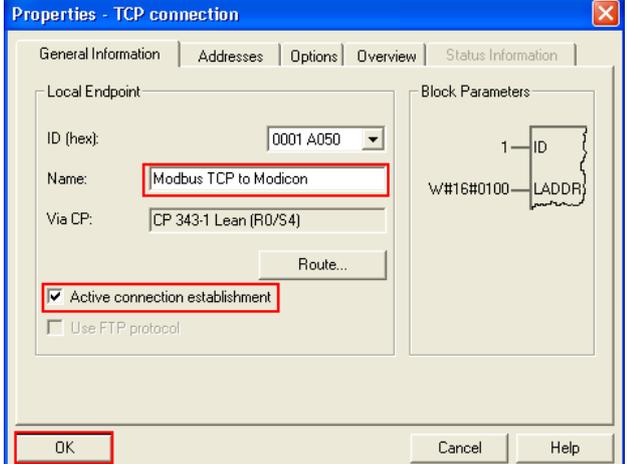
No.	Instruction	Comment
1	Open the sample projects for Modbus TCP CP via the "File → Open" menu.	
2	Copy the blocks from the sample projects for a server or client application and insert it in your Modbus project Overwrite the already existing OB1.	

6.3.3 Configuring Modbus TCP connections

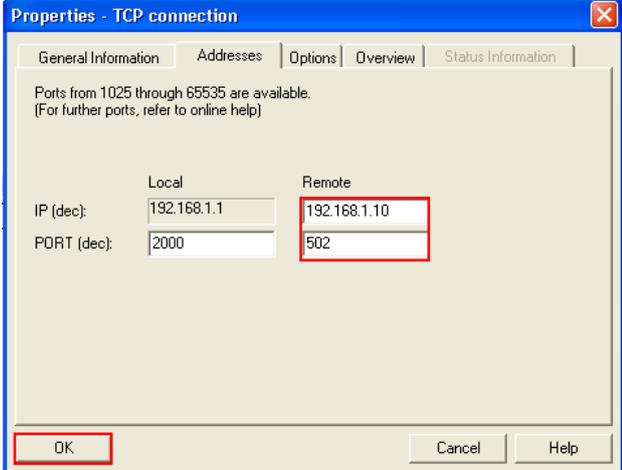
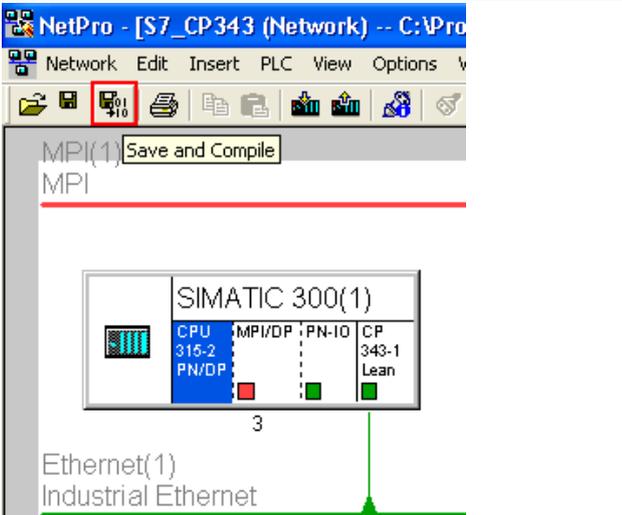
Table 6-27

No.	Instruction	Comment																				
1	<p>Open the OB100 and process the initialization parameters to adapt the Modbus data areas.</p> <p>For the application example described here the following parameters have to be assigned (see figure). For areas not used enter a zero.</p> <p>Client:</p> <ol style="list-style-type: none"> 1. DB: DB11 area: 100 – 399 as Holding register (data_type = 3) 2. DB: DB12 area: 400 – 600 as Input register (data_type = 4) 3. – 8. DB: not necessary <p>Server:</p> <ol style="list-style-type: none"> 1. DB: DB11 area: 0 – 500 as Holding register (data_type = 3) 2. DB: DB12 area: 720 – 900 as Holding register (data_type = 3) 3. – 8. DB: not necessary 	<pre>// Client configuration in S7-300 //adapt the initialisation parameters according to your configuration OPN "CONTROL DAT" L 1 //according to NETPRO T "CONTROL DAT".ID L W#16#100 //according to HW-Config T "CONTROL DAT".LADDR L 3 //first memory area T "CONTROL DAT".data_type_1 //Holding register L 11 T "CONTROL DAT".db_1 L 100 //from 100 to 399 T "CONTROL DAT".start_1 L 399 T "CONTROL DAT".end_1 L 4 //second memory area T "CONTROL DAT".data_type_2 //Input register L 12 T "CONTROL DAT".db_2 L 400 //from 400 to 600 T "CONTROL DAT".start_2 L 600 T "CONTROL DAT".end_2 L 0 //third memory area T "CONTROL DAT".data_type_3 //not used L 0 T "CONTROL DAT".db_3 L 0 T "CONTROL DAT".start_3 L 0 T "CONTROL DAT".end_3</pre>																				
2	<p>Adapt DBs for Modbus data areas in S7 project</p> <p>Client application: DB11 ARRAY[0...300] DB12 ARRAY[0...300]</p> <p>Server application: DB11 ARRAY[0...600] DB12 ARRAY[0...300]</p>	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Address</th> <th>Name</th> <th>Type</th> <th>Initial value</th> </tr> </thead> <tbody> <tr> <td>0..0</td> <td></td> <td>STRUCT</td> <td></td> </tr> <tr> <td>+0..0</td> <td>DB_VAR</td> <td>ARRAY[0..300]</td> <td>W#16#0</td> </tr> <tr> <td>*2..0</td> <td></td> <td>WORD</td> <td></td> </tr> <tr> <td>=800..0</td> <td></td> <td>END_STRUCT</td> <td></td> </tr> </tbody> </table>	Address	Name	Type	Initial value	0..0		STRUCT		+0..0	DB_VAR	ARRAY[0..300]	W#16#0	*2..0		WORD		=800..0		END_STRUCT	
Address	Name	Type	Initial value																			
0..0		STRUCT																				
+0..0	DB_VAR	ARRAY[0..300]	W#16#0																			
*2..0		WORD																				
=800..0		END_STRUCT																				
3	Open NetPro.																					
4	Insert a new connection and configure it in a way so that it can be used for the Modbus TCP communication to Modicon M340.																					

Startup of the Application
Configuring the CPU315-2 PN/DP with CP343-1 Lean

No.	Instruction	Comment
5	<p>In the dialog field "Insert New Connection".select "TCP connection".</p> <p>Under "Connection Partner" select "Unspecified".</p> <p>Note: once this has been confirmed the properties window of this connection will open automatically.</p>	
6	<p>Assign an unambiguous name for the connection in the properties dialog of the TCP connection (e.g. Modbus TCP to Modicon).</p> <p>If the S7 CPU is the Modbus client enable the function "Active connection establishment".</p>	

Startup of the Application
Configuring the CPU315-2 PN/DP with CP343-1 Lean

No.	Instruction	Comment
7	<p>Once the active connection establishment was selected an IP address and a port number of the communication partner (Modicon M340) has to be known. The local port is of no significance in this case.</p> <p>In the case of a passive connection establishment, IP address or port of the communication partner does not need to be known. Only the local port has to be configured. A connection query from a remote communication partner has to take place on this port (local port: 502).</p> <p>Confirm the entry by clicking "OK".</p>	
8	<p>Save and compile the changes made and afterwards exit NetPro.</p>	

Copyright © Siemens AG 2009. All rights reserved.

6.3.4 Downloading project

Table 6-28

No.	Instruction	Comment
1	<p>Mark the SIMATIC 300 station in the SIMATIC manager and load the project into the controller.</p> <p>Note: if asked via which interface you would like to created the connection to the CPU315 you have to select the IP of the CP.</p>	

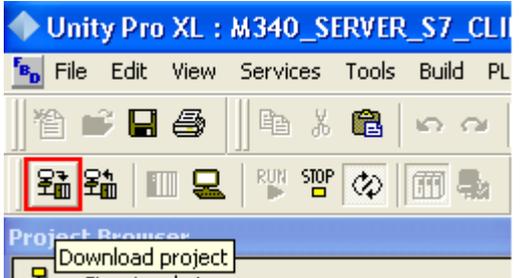
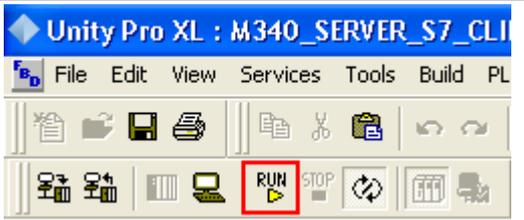
6.4 Configuration of Modicon M340

To be able to better comprehend the application example, the following points will explain how the Modicon M340 has to be configured to allow a Modbus TCP connection with a SIMATIC station.

In the application examples included these steps do not need to be carried out anymore.

6.4.1 Using application example

Table 6-29

No.	Instruction	Comment
1	Open client or server application. In the steps that follow, open the application example included in delivery for the Modicon. M340. Client: „m340_client_s7_server.stu“ Server: „m340_server_s7_client.stu“	File → Open...
2	Open the already created user program in the menu "Section".	A sample program with the corresponding data areas already exists.
3	Rebuild all project.	
4	Create an online connection between PG/PC and Modicon M340.	
5	Transfer the server or client program in the Modicon M340.	
6	Start the CPU	

Startup of the Application

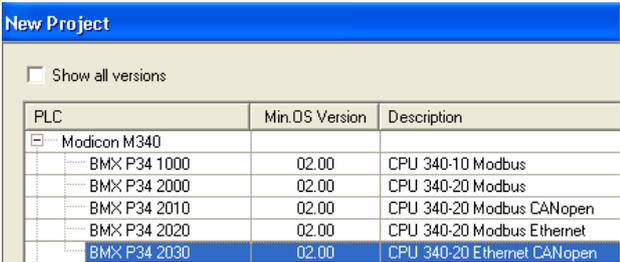
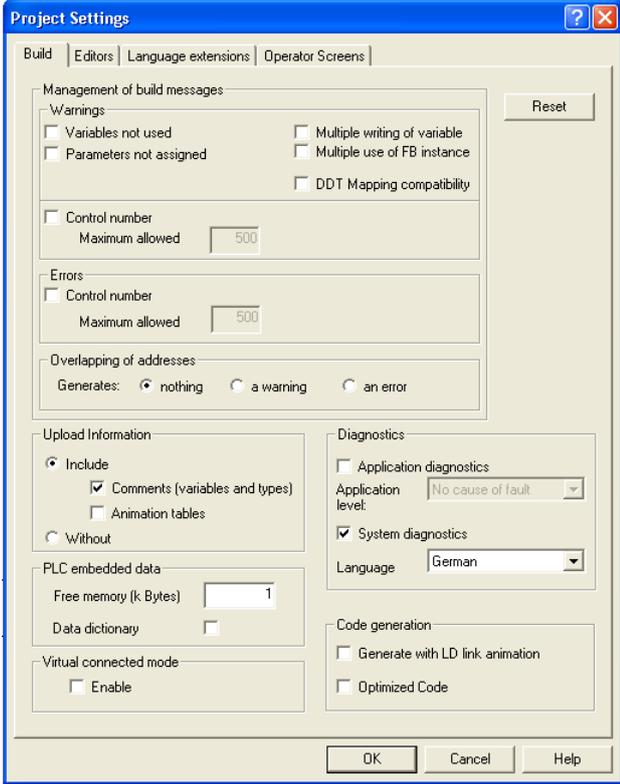
Configuration of Modicon M340

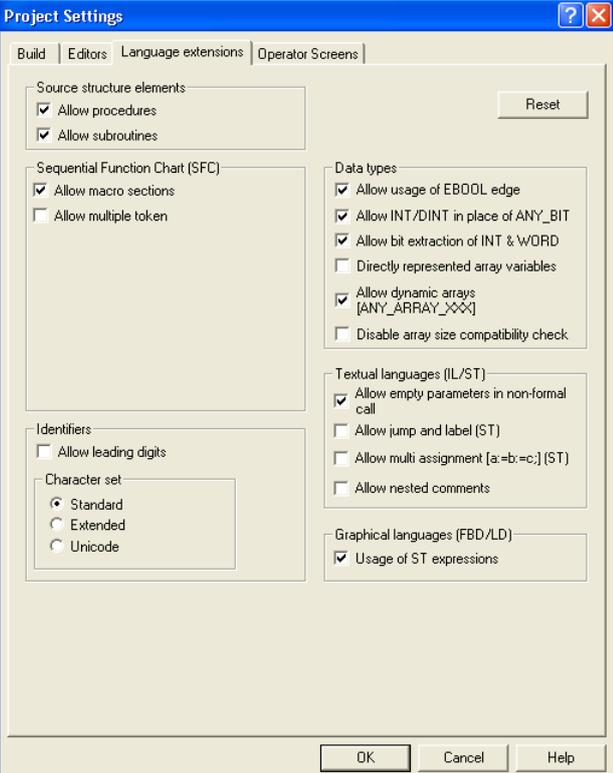
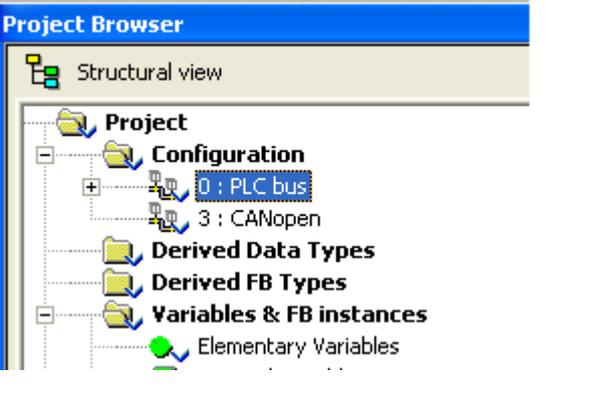
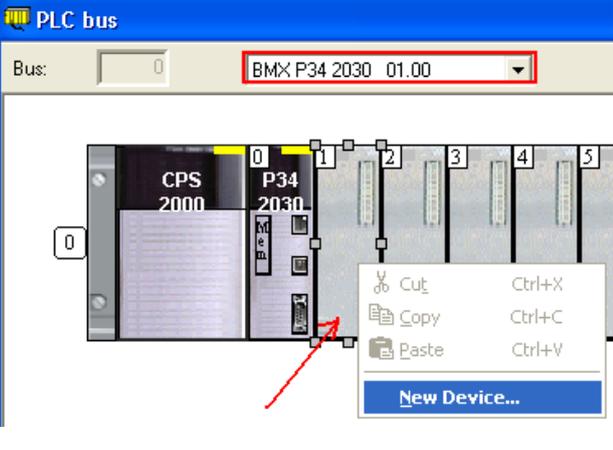
6.4.2 Hardware configuration

Execute the following steps to configure the Modicon M340 hardware.

The table deals with the hardware of the application example. If you are using a different hardware, you have to adjust the configuration accordingly.

Table 6-30

No.	Instruction	Comment
1	Create a new project in Unity Pro XL	File→New
2	Select the hardware.	
3	Configure the project settings under "Tools → Project Settings → Build":	

No.	Instruction	Comment
4	Configure the project settings under "Tools → Project Settings → Language extensions":	
5	Open the hardware configuration by double-clicking "PLC bus" in the "Project Browser" menu.	
6	<p>The hardware configuration is already equipped with power supply and the communication module.</p> <p>Right click onto a free slot in the rack to add I/Os.</p> <p>Insert a DDO1602 in position 1 and a DDI1602 in position 2.</p> <p>Change the PLC BUS to BMX P34 2030 01.00.</p>	

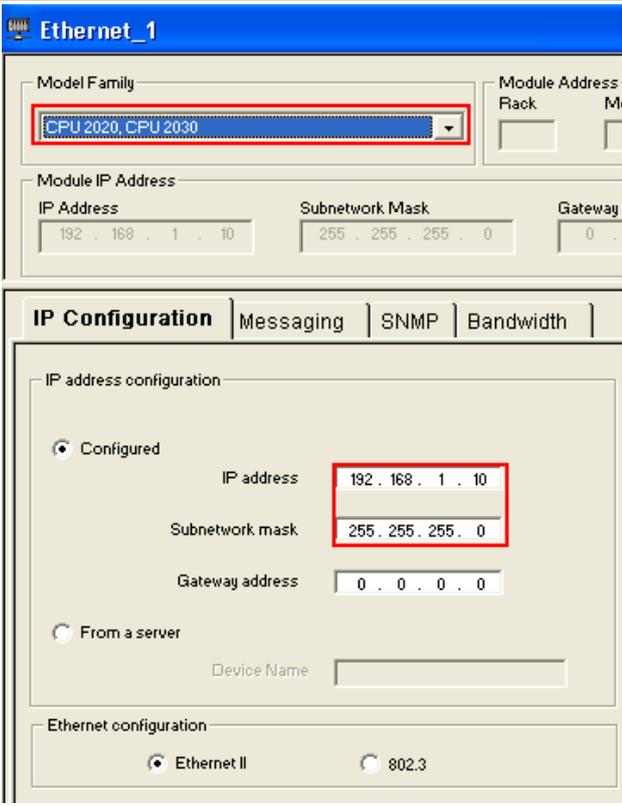
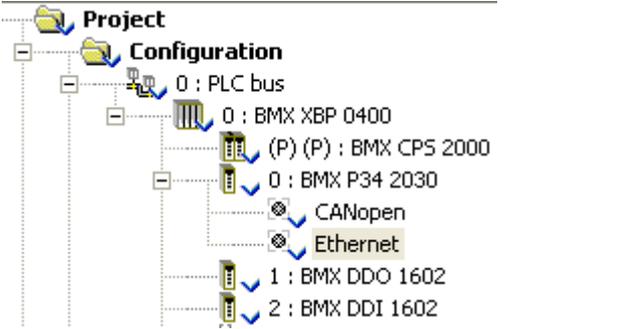
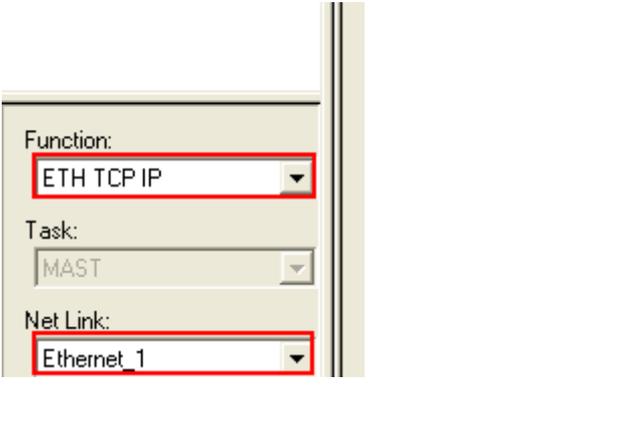
Startup of the Application
Configuration of Modicon M340

No.	Instruction	Comment
7	Right click the rack to adjust the existing hardware. In the application example a rack with 4 slots is used. Therefore the rack with 7 slots configured here has to be exchanged for a rack with 4 slots.	

6.4.3 Configuring an Ethernet interface for Modbus TCP

Table 6-31

No.	Instruction	Comment
1	Right click "Networks" in the "Project Browser" under "Communication" and select the "New Network" menu to insert a new network.	
2	Select Ethernet as network and assign the name "Ethernet_1".	

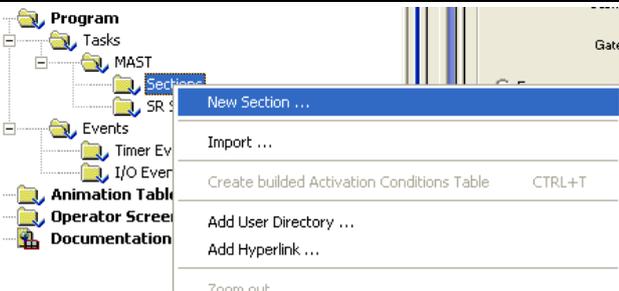
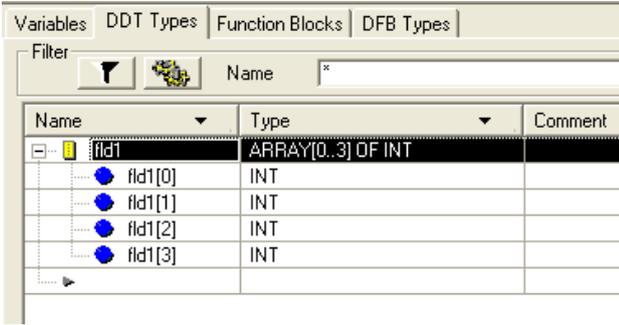
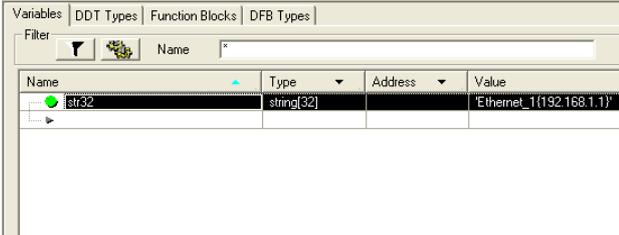
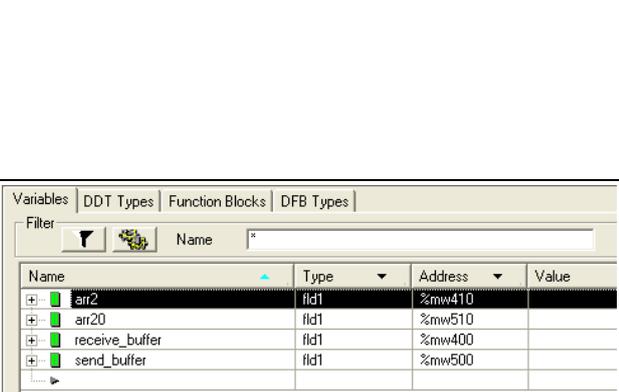
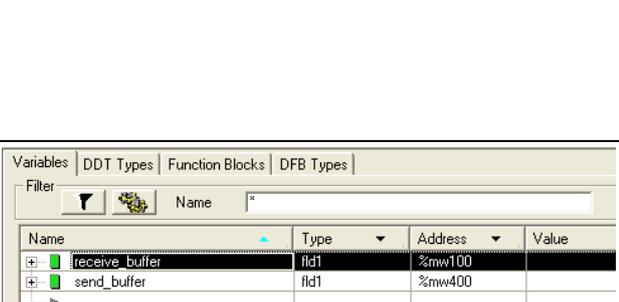
No.	Instruction	Comment
3	<p>Now the local Ethernet interface has to be configured.</p> <p>You get to this menu by left double-clicking on the just created network "Ethernet_1" under the "Networks" menu</p> <p>Change the "Model Family" to "CPU2020, CPU2030" and assign the IP address as well as the subnet mask.</p> <p>IP address: 192.168.1.10 Subnetwork mask: 255.255.255.0</p> <p>After successful configuration exit this menu and accept the changes.</p>	
4	<p>The newly created network now has to be assigned to the hardware.</p> <p>For this purpose open the Ethernet interface in the hardware configuration by double-clicking it "Project Browser → Project → Configuration → PLC bus → BMX XBP 0400 → BMX P34 2030 → Ethernet".</p>	
5	<p>Double-click the "Channel 3" field in the Window just opened up "0.0:Ethernet"</p> <p>Select "Function" ETH TCP IP from the drop-down list of the menu.</p> <p>In the "NET Link" menu select the previously configured network "Ethernet_1" from the drop-down list.</p> <p>Close the window and accept the changes.</p>	

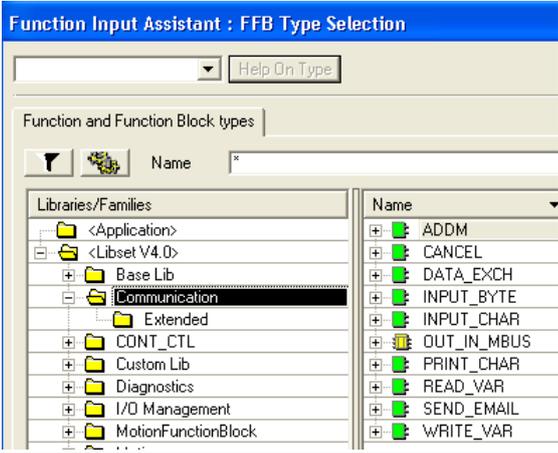
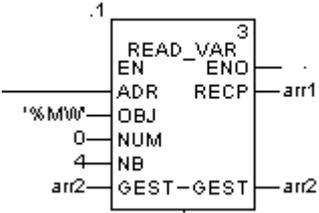
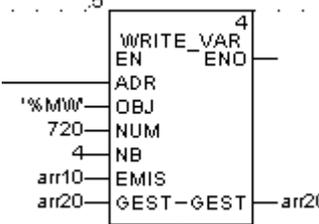
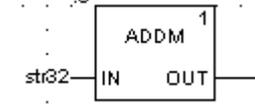
Startup of the Application

Configuration of Modicon M340

6.4.4 Creating a project for Modbus TCP

Table 6-32

No.	Instruction	Comment
1	<p>Create a new section.</p> <p>Project Browser → Station → Program → Tasks → MAST → Sections → right mouse click</p>	
2	<p>Define a structure in the "Derived Data Types" menu. This predefined structure can be used for the declaration of variables.</p> <p>Project Browser → Station → Derived Data Types</p>	
3	<p>Only in client application:</p> <p>create a variable with the initial value "Ethernet_1{192.168.1.1}" from data type string[32].</p> <p>Project Browser → Variables & FB instances → Elementary Variables</p> <p>Note: this variable is necessary for addressing the communication partner (S7 station) of Modicon M340.</p>	
4	<p>Only in client application:</p> <p>for the READ_VAR and WRITE_VAR functions variables are necessary which also serve as send or receive memory. Create the displayed variables for Modicon M340 as client:</p> <p>Project Browser → Variables & FB instances → Derived Variables</p>	
5	<p>Only in server application:</p> <p>declare variables in the server which can be accessed by a Modbus client. Read zone: from %MW400</p>	

No.	Instruction	Comment
	Write zone: %MW100 - %MW399 Project Browser → Variables & FB instances → Derived Variables	
6	Only in client application: with the aid of the FBB Input Assistant the functions "READ_VAR, WRITE_VAR and ADDM" are inserted. Right click an opened section in the editor program to open the assistant. Server application: the blocks are not needed.	
7	Only in client application: insert READ_VAR and supply with parameters: ADR → address OBJ → object type NUM → first object NB → object number GEST → management_param RECP → receiving array	
8	Only in client application: insert WRITE_VAR and supply with parameters ADR → address OBJ → object type NUM → first object NB → object number EMIS → data to write GEST → management parameter	
9	Only in client application: insert ADDM and supply with parameters IN → address string OUT → connect to ADR parameter	

7 Operation of the Application

The following chapter describes the operation of the application examples included in delivery.

7.1 Operation of CPU319-3 PN/DP and IM151-8 PN/DP CPU

For the CPU319-3 PN/DP and the IM151-8 PN/DP CPU the Modbus TCP client jobs are triggered via PG/PC. For this purpose the variable tables are used which enable control of individual DB variables.

When the S7 stations are configured as server the job only has to be started once. The server will then constantly monitor the configured port and will wait for an arriving client job.

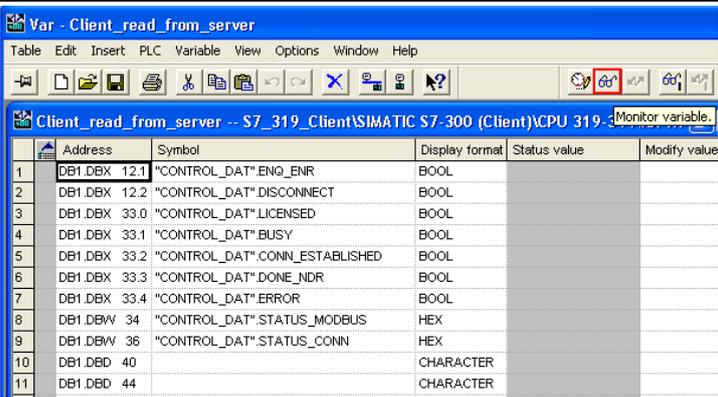
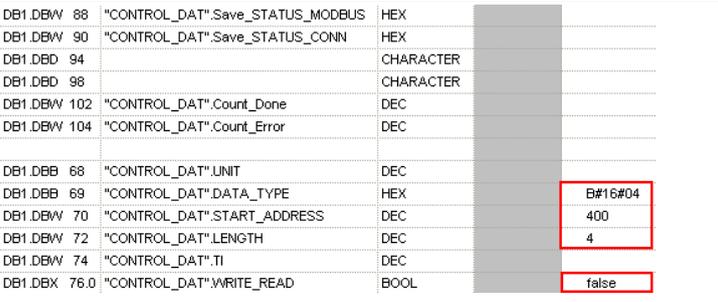
For the S7 client and server application there is the option to change the Modbus TCP data via a variable table ("VAT_DB11-15").

7.1.1 S7 station is client

Reading data from the server

To be able to read data from the Modicon M340 via Modbus TCP follow the instructions of the table below.

Table 7-33: Reading data from the server

No.	Instruction	Comment
1	Open the variable table "Client_read_from_server"	
2	In the menu bar click the "Monitor variable" icon to change to the online view.	
3	Modify the variables displayed in the screenshot Right mouse click → Modify DB1.DBB 69 → B#16#04 DB1.DBW 70 → 400 DB1.DBW 72 → 4 ... DB1.DBX 76.0 → false	

Operation of the Application

Operation of CPU319-3 PN/DP and IM151-8 PN/DP CPU

No.	Instruction	Comment
4	Modify ENQ_ENR Right mouse click → Modify to 1	
5	Display received data Open the variable table "VAT_DB11-15" and change to the online view.	

Writing data in the server

To be able to write data from the client in the Modbus TCP server (Modicon M340), please follow the instructions in the table below.

Table 7-34: Writing data in the server

No.	Instruction	Comment
1	Open the variable table "Client_write_to_server"	
2	In the menu bar click the "Monitor variable" icon to change to the online view.	
3	Modify the variables displayed in the screenshot Right mouse click → Modify DB1.DBB 69 → B#16#03 DB1.DBW 70 → 100 DB1.DBW 72 → 4 ... DB1.DBX 76.0 → true	

Operation of the Application

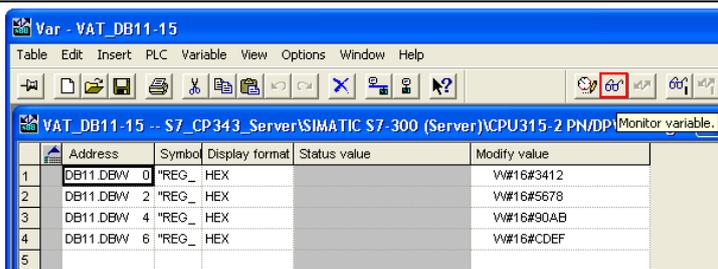
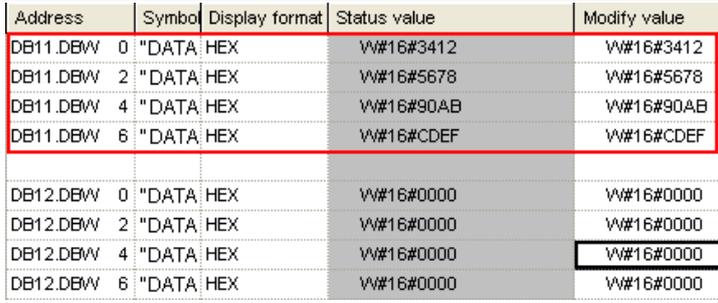
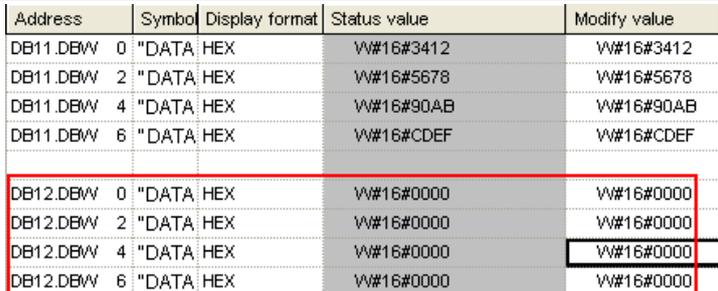
Operation of CPU319-3 PN/DP and IM151-8 PN/DP CPU

No.	Instruction	Comment																																																							
4	<p>Modify send data in DB11</p> <p>Open the variable table "VAT_DB11-15" and change to the online view.</p> <p>Modify the variables that are marked in screenshot.</p>	<table border="1"> <thead> <tr> <th>Address</th> <th>Symbol</th> <th>Display format</th> <th>Status value</th> <th>Modify value</th> </tr> </thead> <tbody> <tr> <td>DB11.DBW 0</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#1A62</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 2</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#5B9F</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 4</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0DEC</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 6</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#00EA</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 0</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#00EA</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 2</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0DEC</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 4</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#5B9F</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 6</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#1A62</td> <td>VW#16#0000</td> </tr> </tbody> </table>	Address	Symbol	Display format	Status value	Modify value	DB11.DBW 0	"DATA"	HEX	VW#16#1A62	VW#16#0000	DB11.DBW 2	"DATA"	HEX	VW#16#5B9F	VW#16#0000	DB11.DBW 4	"DATA"	HEX	VW#16#0DEC	VW#16#0000	DB11.DBW 6	"DATA"	HEX	VW#16#00EA	VW#16#0000	DB12.DBW 0	"DATA"	HEX	VW#16#00EA	VW#16#0000	DB12.DBW 2	"DATA"	HEX	VW#16#0DEC	VW#16#0000	DB12.DBW 4	"DATA"	HEX	VW#16#5B9F	VW#16#0000	DB12.DBW 6	"DATA"	HEX	VW#16#1A62	VW#16#0000										
Address	Symbol	Display format	Status value	Modify value																																																					
DB11.DBW 0	"DATA"	HEX	VW#16#1A62	VW#16#0000																																																					
DB11.DBW 2	"DATA"	HEX	VW#16#5B9F	VW#16#0000																																																					
DB11.DBW 4	"DATA"	HEX	VW#16#0DEC	VW#16#0000																																																					
DB11.DBW 6	"DATA"	HEX	VW#16#00EA	VW#16#0000																																																					
DB12.DBW 0	"DATA"	HEX	VW#16#00EA	VW#16#0000																																																					
DB12.DBW 2	"DATA"	HEX	VW#16#0DEC	VW#16#0000																																																					
DB12.DBW 4	"DATA"	HEX	VW#16#5B9F	VW#16#0000																																																					
DB12.DBW 6	"DATA"	HEX	VW#16#1A62	VW#16#0000																																																					
5	<p>Open the variable table "Client_write_to_server" again.</p> <p>Modify ENQ_ENR to send the data from DB11 to the Modbus TCP server.</p> <p>Right mouse click → Modify to 1</p>	<table border="1"> <thead> <tr> <th>Address</th> <th>Symbol</th> <th>Display format</th> <th>Status value</th> <th>Modify value</th> </tr> </thead> <tbody> <tr> <td>DB1.DBX 12.1</td> <td>"CONTROL_DAT" ENQ_ENR</td> <td>BOOL</td> <td>false</td> <td></td> </tr> <tr> <td>DB1.DBX 12.2</td> <td>"CONTROL_DAT" DISCONNECT</td> <td>BOOL</td> <td>false</td> <td>Monitor</td> </tr> <tr> <td>DB1.DBX 33.0</td> <td>"CONTROL_DAT" LICENSED</td> <td>BOOL</td> <td>false</td> <td>Modify</td> </tr> <tr> <td>DB1.DBX 33.1</td> <td>"CONTROL_DAT" BUSY</td> <td>BOOL</td> <td>false</td> <td></td> </tr> <tr> <td>DB1.DBX 33.2</td> <td>"CONTROL_DAT" CONN_ESTABLISHED</td> <td>BOOL</td> <td>true</td> <td>Update Monitor Values</td> </tr> <tr> <td>DB1.DBX 33.3</td> <td>"CONTROL_DAT" DONE_NDR</td> <td>BOOL</td> <td>false</td> <td>Activate Modify Value</td> </tr> <tr> <td>DB1.DBX 33.4</td> <td>"CONTROL_DAT" ERROR</td> <td>BOOL</td> <td>false</td> <td>Modify Address to 1</td> </tr> <tr> <td>DB1.DBW 34</td> <td>"CONTROL_DAT" STATUS_MODBUS</td> <td>HEX</td> <td>VW#16#A090</td> <td>Modify Address to 0</td> </tr> <tr> <td>DB1.DBW 36</td> <td>"CONTROL_DAT" STATUS_CONN</td> <td>HEX</td> <td>VW#16#0000</td> <td>Cut</td> </tr> <tr> <td>DB1.DBD 40</td> <td></td> <td>CHARACTER</td> <td>' '</td> <td>Copy</td> </tr> </tbody> </table>	Address	Symbol	Display format	Status value	Modify value	DB1.DBX 12.1	"CONTROL_DAT" ENQ_ENR	BOOL	false		DB1.DBX 12.2	"CONTROL_DAT" DISCONNECT	BOOL	false	Monitor	DB1.DBX 33.0	"CONTROL_DAT" LICENSED	BOOL	false	Modify	DB1.DBX 33.1	"CONTROL_DAT" BUSY	BOOL	false		DB1.DBX 33.2	"CONTROL_DAT" CONN_ESTABLISHED	BOOL	true	Update Monitor Values	DB1.DBX 33.3	"CONTROL_DAT" DONE_NDR	BOOL	false	Activate Modify Value	DB1.DBX 33.4	"CONTROL_DAT" ERROR	BOOL	false	Modify Address to 1	DB1.DBW 34	"CONTROL_DAT" STATUS_MODBUS	HEX	VW#16#A090	Modify Address to 0	DB1.DBW 36	"CONTROL_DAT" STATUS_CONN	HEX	VW#16#0000	Cut	DB1.DBD 40		CHARACTER	' '	Copy
Address	Symbol	Display format	Status value	Modify value																																																					
DB1.DBX 12.1	"CONTROL_DAT" ENQ_ENR	BOOL	false																																																						
DB1.DBX 12.2	"CONTROL_DAT" DISCONNECT	BOOL	false	Monitor																																																					
DB1.DBX 33.0	"CONTROL_DAT" LICENSED	BOOL	false	Modify																																																					
DB1.DBX 33.1	"CONTROL_DAT" BUSY	BOOL	false																																																						
DB1.DBX 33.2	"CONTROL_DAT" CONN_ESTABLISHED	BOOL	true	Update Monitor Values																																																					
DB1.DBX 33.3	"CONTROL_DAT" DONE_NDR	BOOL	false	Activate Modify Value																																																					
DB1.DBX 33.4	"CONTROL_DAT" ERROR	BOOL	false	Modify Address to 1																																																					
DB1.DBW 34	"CONTROL_DAT" STATUS_MODBUS	HEX	VW#16#A090	Modify Address to 0																																																					
DB1.DBW 36	"CONTROL_DAT" STATUS_CONN	HEX	VW#16#0000	Cut																																																					
DB1.DBD 40		CHARACTER	' '	Copy																																																					

7.1.2 S7 station is server

The following steps describe how to modify the data from the S7 Modbus TCP server. This data is read by the Modbus TCP client (Modicon M340).

Table 7-35

No.	Instruction	Comment																																													
1	Open the variable table "VAT_DB11-15"																																														
2	In the menu bar click the "Monitor variable" icon to change to the online view.																																														
3	<p>Modify the variables displayed in the screenshot</p> <p>Right mouse click → Modify</p> <p>DB11.DBW 0 → W#16#3412</p> <p>DB11.DBW 2 → W#16#5678</p> <p>DB11.DBW 4 → W#16#90AB</p> <p>DB11.DBW 6 → W#16#CDEF</p> <p>These values are read from the Modicon M340 via Modbus TCP.</p>	 <table border="1"> <thead> <tr> <th>Address</th> <th>Symbol</th> <th>Display format</th> <th>Status value</th> <th>Modify value</th> </tr> </thead> <tbody> <tr> <td>DB11.DBW 0</td> <td>"DATA</td> <td>HEX</td> <td>W#16#3412</td> <td>W#16#3412</td> </tr> <tr> <td>DB11.DBW 2</td> <td>"DATA</td> <td>HEX</td> <td>W#16#5678</td> <td>W#16#5678</td> </tr> <tr> <td>DB11.DBW 4</td> <td>"DATA</td> <td>HEX</td> <td>W#16#90AB</td> <td>W#16#90AB</td> </tr> <tr> <td>DB11.DBW 6</td> <td>"DATA</td> <td>HEX</td> <td>W#16#CDEF</td> <td>W#16#CDEF</td> </tr> <tr> <td>DB12.DBW 0</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> <tr> <td>DB12.DBW 2</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> <tr> <td>DB12.DBW 4</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> <tr> <td>DB12.DBW 6</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> </tbody> </table>	Address	Symbol	Display format	Status value	Modify value	DB11.DBW 0	"DATA	HEX	W#16#3412	W#16#3412	DB11.DBW 2	"DATA	HEX	W#16#5678	W#16#5678	DB11.DBW 4	"DATA	HEX	W#16#90AB	W#16#90AB	DB11.DBW 6	"DATA	HEX	W#16#CDEF	W#16#CDEF	DB12.DBW 0	"DATA	HEX	W#16#0000	W#16#0000	DB12.DBW 2	"DATA	HEX	W#16#0000	W#16#0000	DB12.DBW 4	"DATA	HEX	W#16#0000	W#16#0000	DB12.DBW 6	"DATA	HEX	W#16#0000	W#16#0000
Address	Symbol	Display format	Status value	Modify value																																											
DB11.DBW 0	"DATA	HEX	W#16#3412	W#16#3412																																											
DB11.DBW 2	"DATA	HEX	W#16#5678	W#16#5678																																											
DB11.DBW 4	"DATA	HEX	W#16#90AB	W#16#90AB																																											
DB11.DBW 6	"DATA	HEX	W#16#CDEF	W#16#CDEF																																											
DB12.DBW 0	"DATA	HEX	W#16#0000	W#16#0000																																											
DB12.DBW 2	"DATA	HEX	W#16#0000	W#16#0000																																											
DB12.DBW 4	"DATA	HEX	W#16#0000	W#16#0000																																											
DB12.DBW 6	"DATA	HEX	W#16#0000	W#16#0000																																											
4	In the opened variable table you can also check the data that is written in the server by the client.	 <table border="1"> <thead> <tr> <th>Address</th> <th>Symbol</th> <th>Display format</th> <th>Status value</th> <th>Modify value</th> </tr> </thead> <tbody> <tr> <td>DB11.DBW 0</td> <td>"DATA</td> <td>HEX</td> <td>W#16#3412</td> <td>W#16#3412</td> </tr> <tr> <td>DB11.DBW 2</td> <td>"DATA</td> <td>HEX</td> <td>W#16#5678</td> <td>W#16#5678</td> </tr> <tr> <td>DB11.DBW 4</td> <td>"DATA</td> <td>HEX</td> <td>W#16#90AB</td> <td>W#16#90AB</td> </tr> <tr> <td>DB11.DBW 6</td> <td>"DATA</td> <td>HEX</td> <td>W#16#CDEF</td> <td>W#16#CDEF</td> </tr> <tr> <td>DB12.DBW 0</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> <tr> <td>DB12.DBW 2</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> <tr> <td>DB12.DBW 4</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> <tr> <td>DB12.DBW 6</td> <td>"DATA</td> <td>HEX</td> <td>W#16#0000</td> <td>W#16#0000</td> </tr> </tbody> </table>	Address	Symbol	Display format	Status value	Modify value	DB11.DBW 0	"DATA	HEX	W#16#3412	W#16#3412	DB11.DBW 2	"DATA	HEX	W#16#5678	W#16#5678	DB11.DBW 4	"DATA	HEX	W#16#90AB	W#16#90AB	DB11.DBW 6	"DATA	HEX	W#16#CDEF	W#16#CDEF	DB12.DBW 0	"DATA	HEX	W#16#0000	W#16#0000	DB12.DBW 2	"DATA	HEX	W#16#0000	W#16#0000	DB12.DBW 4	"DATA	HEX	W#16#0000	W#16#0000	DB12.DBW 6	"DATA	HEX	W#16#0000	W#16#0000
Address	Symbol	Display format	Status value	Modify value																																											
DB11.DBW 0	"DATA	HEX	W#16#3412	W#16#3412																																											
DB11.DBW 2	"DATA	HEX	W#16#5678	W#16#5678																																											
DB11.DBW 4	"DATA	HEX	W#16#90AB	W#16#90AB																																											
DB11.DBW 6	"DATA	HEX	W#16#CDEF	W#16#CDEF																																											
DB12.DBW 0	"DATA	HEX	W#16#0000	W#16#0000																																											
DB12.DBW 2	"DATA	HEX	W#16#0000	W#16#0000																																											
DB12.DBW 4	"DATA	HEX	W#16#0000	W#16#0000																																											
DB12.DBW 6	"DATA	HEX	W#16#0000	W#16#0000																																											

Copyright © Siemens AG 2009. All rights reserved.

7.2

7.3 Operation of CPU315-2 PN/DP + CP343-1 Lean

For the CPU315-2 PN/DP + CP343-1 Lean the Modbus TCP client jobs are also triggered via PG/PC. Variable tables are also used which enable control of individual DB variables.

When the S7 stations are configured as server the job only has to be started once. The server will then constantly monitor the configured port and will wait for an arriving client job.

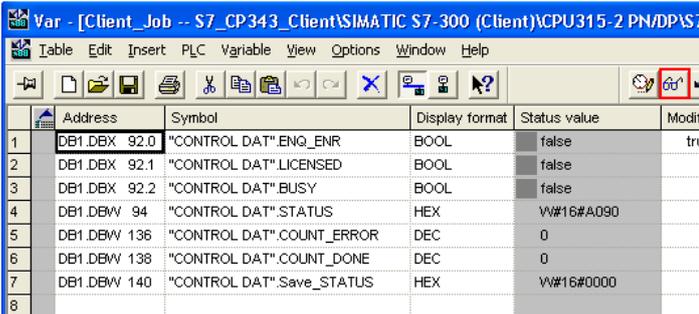
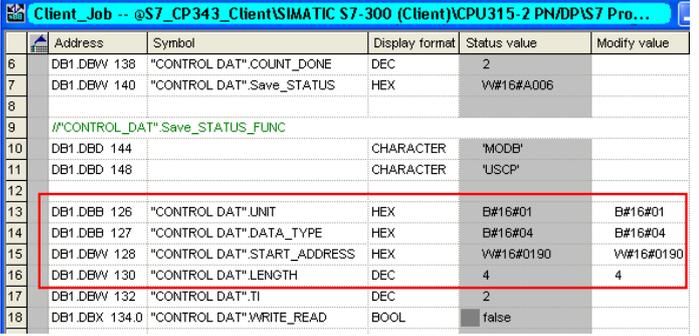
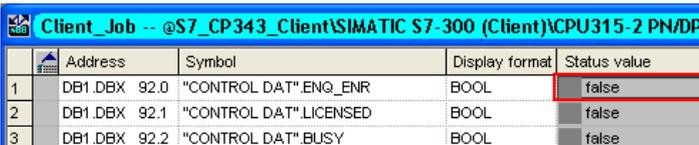
For the server application there is the option to change the Modbus TCP data via variable table.

7.3.1 S7 station is client

Reading data from the server

To be able to read data from the Modicon M340 via Modbus TCP follow the instructions of the table below.

Table 7-36: Reading data from the server

No.	Instruction	Comment
1	Open the variable table "Client_read_from_server"	
2	In the menu bar click the "Monitor variable" icon to change to the online view.	
3	Modify the variables displayed in the screenshot Right mouse click → Modify DB1.DBB 126 → B#16#01 DB1.DBB 127 → B#16#04 DB1.DBW 128 → W#16#0190 DB1.DBW 130 → 4 ... DB1.DBX 134.0 → false	
4	Modify ENQ_ENR (DB1.DBX 92.0) Right mouse click → Modify to 1	

Operation of the Application

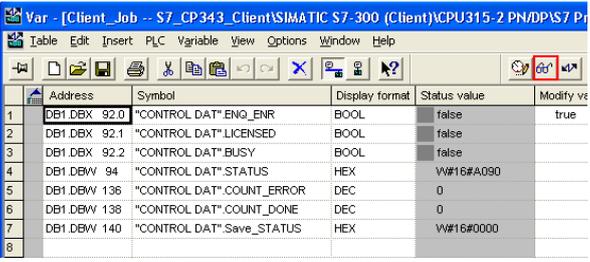
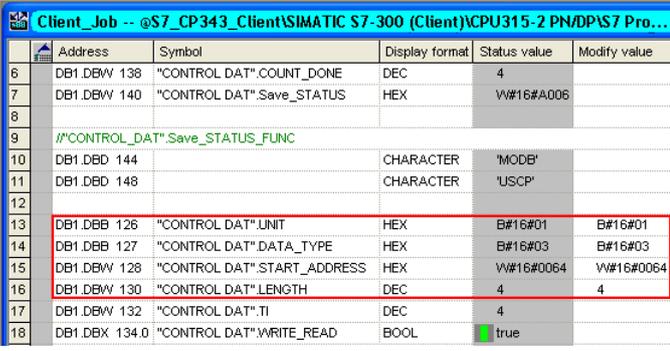
Operation of CPU315-2 PN/DP + CP343-1 Lean

No.	Instruction	Comment																																													
5	Display received data Open the variable table "VAT_DB11-15" and change to the online view.	<table border="1"> <thead> <tr> <th>Address</th> <th>Symbol</th> <th>Display format</th> <th>Status value</th> <th>Modify value</th> </tr> </thead> <tbody> <tr> <td>DB11.DBW 0</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0000</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 2</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0000</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 4</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0000</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 6</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0000</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 0</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#012E</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 2</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0016</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 4</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0098</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 6</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0002</td> <td>VW#16#0000</td> </tr> </tbody> </table>	Address	Symbol	Display format	Status value	Modify value	DB11.DBW 0	"DATA"	HEX	VW#16#0000	VW#16#0000	DB11.DBW 2	"DATA"	HEX	VW#16#0000	VW#16#0000	DB11.DBW 4	"DATA"	HEX	VW#16#0000	VW#16#0000	DB11.DBW 6	"DATA"	HEX	VW#16#0000	VW#16#0000	DB12.DBW 0	"DATA"	HEX	VW#16#012E	VW#16#0000	DB12.DBW 2	"DATA"	HEX	VW#16#0016	VW#16#0000	DB12.DBW 4	"DATA"	HEX	VW#16#0098	VW#16#0000	DB12.DBW 6	"DATA"	HEX	VW#16#0002	VW#16#0000
Address	Symbol	Display format	Status value	Modify value																																											
DB11.DBW 0	"DATA"	HEX	VW#16#0000	VW#16#0000																																											
DB11.DBW 2	"DATA"	HEX	VW#16#0000	VW#16#0000																																											
DB11.DBW 4	"DATA"	HEX	VW#16#0000	VW#16#0000																																											
DB11.DBW 6	"DATA"	HEX	VW#16#0000	VW#16#0000																																											
DB12.DBW 0	"DATA"	HEX	VW#16#012E	VW#16#0000																																											
DB12.DBW 2	"DATA"	HEX	VW#16#0016	VW#16#0000																																											
DB12.DBW 4	"DATA"	HEX	VW#16#0098	VW#16#0000																																											
DB12.DBW 6	"DATA"	HEX	VW#16#0002	VW#16#0000																																											

Writing data in the server

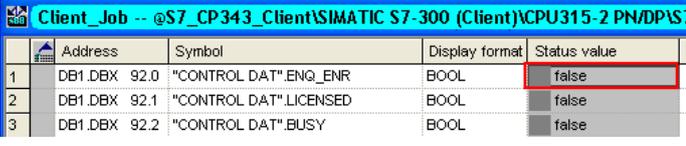
To be able to write data from the client in the Modbus TCP server (Modicon M340), please follow the instructions in the table below.

Table 7-37: Writing data in the server

No.	Instruction	Comment																																													
1	Open the variable table "Client_write_to_server"																																														
2	In the menu bar click the "Monitor variable" icon to change to the online view.																																														
3	Modify the variables displayed in the screenshot Right mouse click → Modify DB1.DBB 126 → B#16#01 DB1.DBB 127 → B#16#03 DB1.DBW 128 → W#16#0064 DB1.DBW 130 → 4 ... DB1.DBX 134.0 → true																																														
4	Modify send data in DB11 Open the variable table "VAT_DB11-15" and change to the online view. Modify the variables that are marked in the screenshot.	<table border="1"> <thead> <tr> <th>Address</th> <th>Symbol</th> <th>Display format</th> <th>Status value</th> <th>Modify value</th> </tr> </thead> <tbody> <tr> <td>DB11.DBW 0</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#1A62</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 2</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#5B9F</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 4</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0DEC</td> <td>VW#16#0000</td> </tr> <tr> <td>DB11.DBW 6</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#00EA</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 0</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#00EA</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 2</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#0DEC</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 4</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#5B9F</td> <td>VW#16#0000</td> </tr> <tr> <td>DB12.DBW 6</td> <td>"DATA"</td> <td>HEX</td> <td>VW#16#1A62</td> <td>VW#16#0000</td> </tr> </tbody> </table>	Address	Symbol	Display format	Status value	Modify value	DB11.DBW 0	"DATA"	HEX	VW#16#1A62	VW#16#0000	DB11.DBW 2	"DATA"	HEX	VW#16#5B9F	VW#16#0000	DB11.DBW 4	"DATA"	HEX	VW#16#0DEC	VW#16#0000	DB11.DBW 6	"DATA"	HEX	VW#16#00EA	VW#16#0000	DB12.DBW 0	"DATA"	HEX	VW#16#00EA	VW#16#0000	DB12.DBW 2	"DATA"	HEX	VW#16#0DEC	VW#16#0000	DB12.DBW 4	"DATA"	HEX	VW#16#5B9F	VW#16#0000	DB12.DBW 6	"DATA"	HEX	VW#16#1A62	VW#16#0000
Address	Symbol	Display format	Status value	Modify value																																											
DB11.DBW 0	"DATA"	HEX	VW#16#1A62	VW#16#0000																																											
DB11.DBW 2	"DATA"	HEX	VW#16#5B9F	VW#16#0000																																											
DB11.DBW 4	"DATA"	HEX	VW#16#0DEC	VW#16#0000																																											
DB11.DBW 6	"DATA"	HEX	VW#16#00EA	VW#16#0000																																											
DB12.DBW 0	"DATA"	HEX	VW#16#00EA	VW#16#0000																																											
DB12.DBW 2	"DATA"	HEX	VW#16#0DEC	VW#16#0000																																											
DB12.DBW 4	"DATA"	HEX	VW#16#5B9F	VW#16#0000																																											
DB12.DBW 6	"DATA"	HEX	VW#16#1A62	VW#16#0000																																											

Operation of the Application

Operation of CPU315-2 PN/DP + CP343-1 Lean

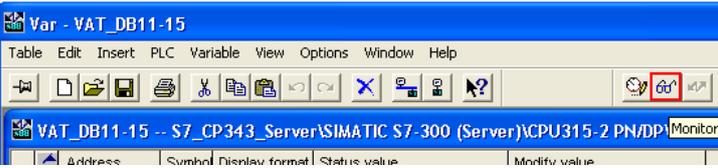
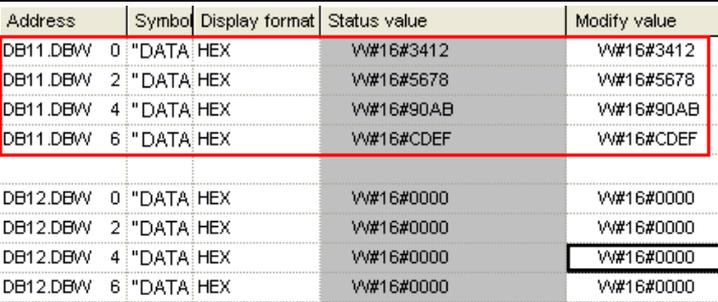
No.	Instruction	Comment
5	<p>Open the variable table "Client_write_to_server" again.</p> <p>Modify ENQ_ENR</p> <p>Right mouse click → Modify to 1</p>	

7.3.2 S7 station is server

The following steps describe how to modify the data from the S7 Modbus TCP server.

This data is read by the Modbus TCP client (Modicon M340).

Table 7-38: Viewing Modbus TCP server data

No.	Instruction	Comment
1	Open variable table "VAT_DB11-15"	
2	In the menu bar click the "Monitor variable" icon to change to the online view.	
3	<p>Modify the variables displayed in the screenshot</p> <p>Right mouse click → Modify</p> <p>DB11.DBW 0 → W#16#3412</p> <p>DB11.DBW 2 → W#16#5678</p> <p>DB11.DBW 4 → W#16#90AB</p> <p>DB11.DBW 6 → W#16#CDEF</p>	

Operation of the Application
Operation of CPU315-2 PN/DP + CP343-1 Lean

No.	Instruction	Comment				
4	In the opened variable table you can also check the data that is written in the server by the client.	Address	Symbol	Display format	Status value	Modify value
		DB11.DBW 0	"DATA	HEX	W#16#3412	W#16#3412
		DB11.DBW 2	"DATA	HEX	W#16#5678	W#16#5678
		DB11.DBW 4	"DATA	HEX	W#16#90AB	W#16#90AB
		DB11.DBW 6	"DATA	HEX	W#16#CDEF	W#16#CDEF
		DB12.DBW 0	"DATA	HEX	W#16#0000	W#16#0000
		DB12.DBW 2	"DATA	HEX	W#16#0000	W#16#0000
		DB12.DBW 4	"DATA	HEX	W#16#0000	W#16#0000
		DB12.DBW 6	"DATA	HEX	W#16#0000	W#16#0000

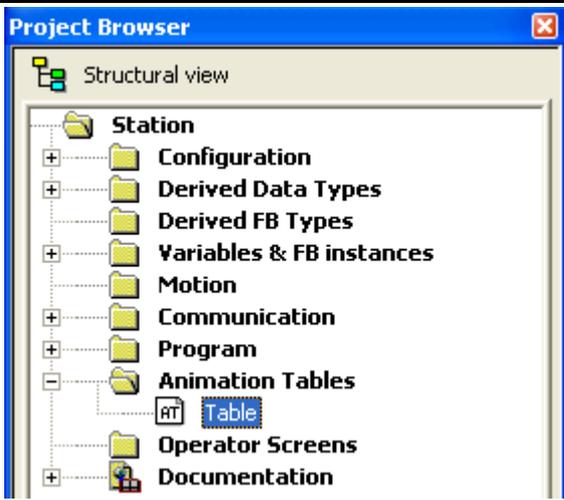
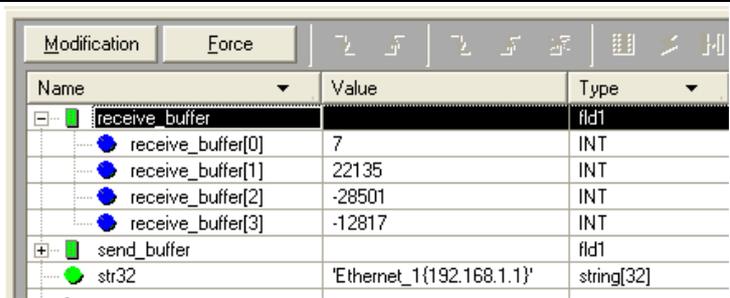
7.4 Operation of Modicon M340

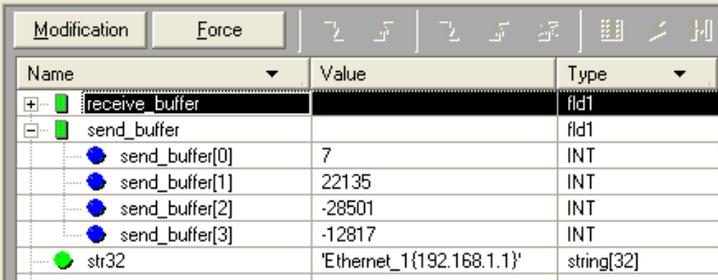
7.4.1 Modicon M340 as client

Reading data from the server and writing to the server

The following steps describe, how to view the Modicon M340 data, which will be read from and written to the S7. In the program of the Modicon M340 all the received data will be copied directly onto the sent data. If you want to modify the sent data yourself, you must delete the copy instructions in the M340 program, recompile the program and download it to the M340 PLC again.

Tabelle 7-39: Reading data from the server

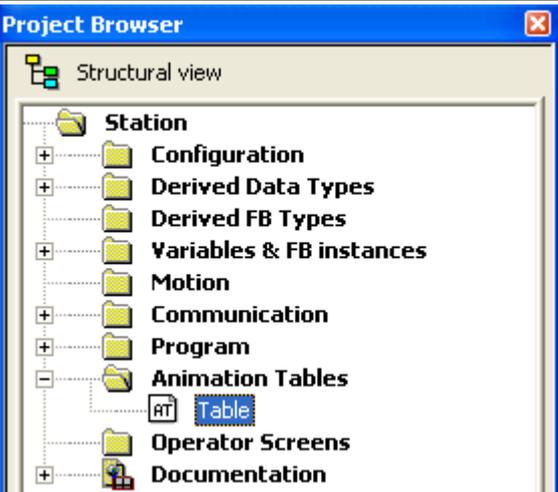
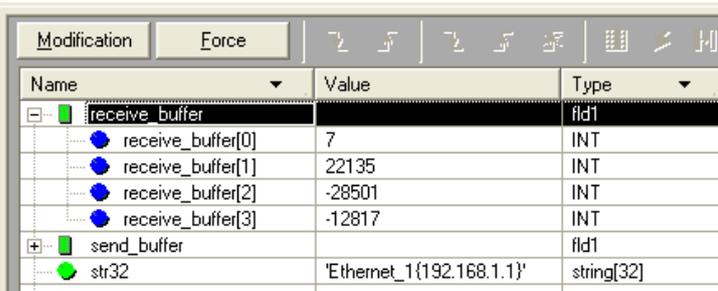
No.	Instruction	Comment																								
1	Open the variable table „Animation Tables → Table“																									
2	In the menu bar click the “Connect” icon to change to the online view.																									
3	Viewing the received data That data are read from the server(S7) by the client (M340)	 <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>receive_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td> receive_buffer[0]</td> <td>7</td> <td>INT</td> </tr> <tr> <td> receive_buffer[1]</td> <td>22135</td> <td>INT</td> </tr> <tr> <td> receive_buffer[2]</td> <td>-28501</td> <td>INT</td> </tr> <tr> <td> receive_buffer[3]</td> <td>-12817</td> <td>INT</td> </tr> <tr> <td>send_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td>str32</td> <td>'Ethernet_1{192.168.1.1}'</td> <td>string[32]</td> </tr> </tbody> </table>	Name	Value	Type	receive_buffer		fld1	receive_buffer[0]	7	INT	receive_buffer[1]	22135	INT	receive_buffer[2]	-28501	INT	receive_buffer[3]	-12817	INT	send_buffer		fld1	str32	'Ethernet_1{192.168.1.1}'	string[32]
Name	Value	Type																								
receive_buffer		fld1																								
receive_buffer[0]	7	INT																								
receive_buffer[1]	22135	INT																								
receive_buffer[2]	-28501	INT																								
receive_buffer[3]	-12817	INT																								
send_buffer		fld1																								
str32	'Ethernet_1{192.168.1.1}'	string[32]																								

No.	Instruction	Comment																								
4	Viewing the sent data That data are written to the server (S7) by the client. (M340).	 <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>receive_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td>send_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td> send_buffer[0]</td> <td>7</td> <td>INT</td> </tr> <tr> <td> send_buffer[1]</td> <td>22135</td> <td>INT</td> </tr> <tr> <td> send_buffer[2]</td> <td>-28501</td> <td>INT</td> </tr> <tr> <td> send_buffer[3]</td> <td>-12817</td> <td>INT</td> </tr> <tr> <td>str32</td> <td>'Ethernet_1{192.168.1.1}'</td> <td>string[32]</td> </tr> </tbody> </table>	Name	Value	Type	receive_buffer		fld1	send_buffer		fld1	send_buffer[0]	7	INT	send_buffer[1]	22135	INT	send_buffer[2]	-28501	INT	send_buffer[3]	-12817	INT	str32	'Ethernet_1{192.168.1.1}'	string[32]
Name	Value	Type																								
receive_buffer		fld1																								
send_buffer		fld1																								
send_buffer[0]	7	INT																								
send_buffer[1]	22135	INT																								
send_buffer[2]	-28501	INT																								
send_buffer[3]	-12817	INT																								
str32	'Ethernet_1{192.168.1.1}'	string[32]																								

7.4.2 Modicon M340 as server

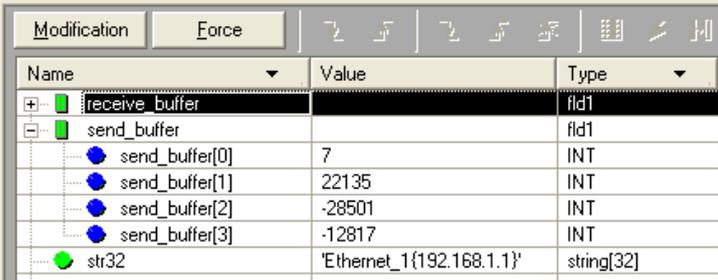
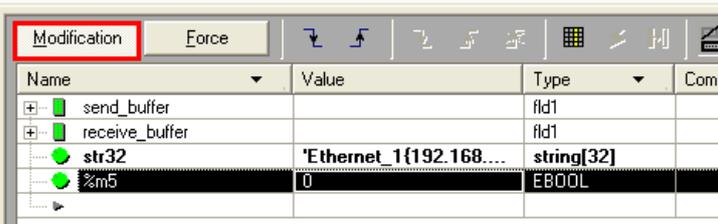
The following steps describe, how to change the data of the Modicon M340. These data will be read from the server (M340) by the client (S7). Furthermore you can view the data which are written to the server (M340) by the client (S7).

Tabelle 7-40: Viewing the Modbus TCP server data

No.	Instruction	Comment																								
1	Open the variable table „Animation Tables → Table“																									
2	In the menu bar click the “Connect” icon to change to the online view.																									
3	Viewing the received data That data are written to the server (M340) by the client. (S7).	 <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>receive_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td> receive_buffer[0]</td> <td>7</td> <td>INT</td> </tr> <tr> <td> receive_buffer[1]</td> <td>22135</td> <td>INT</td> </tr> <tr> <td> receive_buffer[2]</td> <td>-28501</td> <td>INT</td> </tr> <tr> <td> receive_buffer[3]</td> <td>-12817</td> <td>INT</td> </tr> <tr> <td>send_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td>str32</td> <td>'Ethernet_1{192.168.1.1}'</td> <td>string[32]</td> </tr> </tbody> </table>	Name	Value	Type	receive_buffer		fld1	receive_buffer[0]	7	INT	receive_buffer[1]	22135	INT	receive_buffer[2]	-28501	INT	receive_buffer[3]	-12817	INT	send_buffer		fld1	str32	'Ethernet_1{192.168.1.1}'	string[32]
Name	Value	Type																								
receive_buffer		fld1																								
receive_buffer[0]	7	INT																								
receive_buffer[1]	22135	INT																								
receive_buffer[2]	-28501	INT																								
receive_buffer[3]	-12817	INT																								
send_buffer		fld1																								
str32	'Ethernet_1{192.168.1.1}'	string[32]																								

Operation of the Application

Operation of Modicon M340

No.	Instruction	Comment																								
4	Viewing the sent data That data are read from the server (M340) by the client (S7).	 <p>The screenshot shows a variable declaration window with the following data:</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td>receive_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td>send_buffer</td> <td></td> <td>fld1</td> </tr> <tr> <td> send_buffer[0]</td> <td>7</td> <td>INT</td> </tr> <tr> <td> send_buffer[1]</td> <td>22135</td> <td>INT</td> </tr> <tr> <td> send_buffer[2]</td> <td>-28501</td> <td>INT</td> </tr> <tr> <td> send_buffer[3]</td> <td>-12817</td> <td>INT</td> </tr> <tr> <td>str32</td> <td>'Ethernet_1{192.168.1.1}'</td> <td>string[32]</td> </tr> </tbody> </table>	Name	Value	Type	receive_buffer		fld1	send_buffer		fld1	send_buffer[0]	7	INT	send_buffer[1]	22135	INT	send_buffer[2]	-28501	INT	send_buffer[3]	-12817	INT	str32	'Ethernet_1{192.168.1.1}'	string[32]
Name	Value	Type																								
receive_buffer		fld1																								
send_buffer		fld1																								
send_buffer[0]	7	INT																								
send_buffer[1]	22135	INT																								
send_buffer[2]	-28501	INT																								
send_buffer[3]	-12817	INT																								
str32	'Ethernet_1{192.168.1.1}'	string[32]																								
5	Switch on variable send data Click the button „Modification“ and set the variable %M5 to the value 1.	 <p>The screenshot shows the same variable declaration window, but with the 'Modification' button highlighted in red. The variable '%m5' is now visible and set to the value 0.</p> <table border="1"> <thead> <tr> <th>Name</th> <th>Value</th> <th>Type</th> <th>Comr</th> </tr> </thead> <tbody> <tr> <td>send_buffer</td> <td></td> <td>fld1</td> <td></td> </tr> <tr> <td>receive_buffer</td> <td></td> <td>fld1</td> <td></td> </tr> <tr> <td>str32</td> <td>'Ethernet_1{192.168.1.1}'</td> <td>string[32]</td> <td></td> </tr> <tr> <td>%m5</td> <td>0</td> <td>EB00L</td> <td></td> </tr> </tbody> </table>	Name	Value	Type	Comr	send_buffer		fld1		receive_buffer		fld1		str32	'Ethernet_1{192.168.1.1}'	string[32]		%m5	0	EB00L					
Name	Value	Type	Comr																							
send_buffer		fld1																								
receive_buffer		fld1																								
str32	'Ethernet_1{192.168.1.1}'	string[32]																								
%m5	0	EB00L																								

8 Related Literature

8.1 Bibliography

This list is not complete and only represents a selection of relevant literature.

Table 8-41 Bibliography

	Topic	Title
/1/	STEP7	Automating with STEP7 in STL und SCL Hans Berger Publisher: Vch Pub ISBN-10 3895783412 ISBN-13 9783895783418

8.2 Internet Links

This list is not complete and only represents a selection of relevant information.

Table 8-42 Internet links

	Topic	Title
\1\	Reference to the entry	http://support.automation.siemens.com/WW/view/en/38586568
\2\	Siemens I IA/DT Customer Support	http://support.automation.siemens.com
\3\	Programming with STEP 7 V5.4	http://support.automation.siemens.com/WW/view/en/18652056
\4\	Information OPEN MODBUS / TCP	http://support.automation.siemens.com/WW/view/en/22660304
\5\	Modbus TCP Wizard	http://support.automation.siemens.com/WW/view/en/31535566
\6\	S7 OpenModbus/TCP license/downloads	http://www.industry.siemens.com/industrial-services/it/de/products/simatic_add_ons/s7_open_modbus_tcp.htm

9 History

Table 9-43 History

Version	Date	Modifications
V1.1	29.01.2010	Second version