

# **SIEMENS**

## **SIMATIC**

### **S7 - OPEN MODBUS / TCP Communication via the integrated PN interface of the PLC**

**Manual**

# SIEMENS

## SIMATIC S7

### **S7 OPEN MODBUS / TCP Communication via the integrated PN interface of the PLC**

Manual

#### Preface, Table of Contents

---

Product Description	<b>1</b>
Getting Started	<b>2</b>
Commissioning	<b>3</b>
Parameterization	<b>4</b>
Licensing	<b>5</b>
FB MODBUSPN	<b>6</b>
Diagnosis	<b>7</b>
Sample Application	<b>8</b>
<b>Appendices</b>	
Literature	
Glossary	

---

## Safety Precautions and Warnings

This manual contains warnings, which you should note for your own safety as well as for the prevention of damage to property. These warnings are indicated by means of a warning triangle and are displayed as follows in accordance with the level of danger:



---

### Danger

indicates that loss of life, severe personal injury or substantial damage **will** result if proper precautions are not taken.

---



---

### Warning

indicates that loss of life, severe personal injury or substantial damage **can** result if proper precautions are not taken.

---



---

### Caution

indicates that minor personal injury or property damage can result if proper precautions are not taken.

---

---

### Note

represents an important information especially significant to the product, handling of the product or a specific part of this documentation.

---

## Qualified Personnel

The start-up and the operation of the device may only be carried out by **qualified personnel**. Qualified personnel in the sense of the security advices of this manual are any persons authorized to commission, ground and label devices, systems and electric circuits.

## Use as prescribed

Please note:



---

### Warning

This device may only be used for applications as prescribed in the catalogue and the technical description. Furthermore, they may only be used in conjunction with third-party devices and components recommended and authorized by Siemens.

A successful and safe operation of this product is depends on proper transport, and correct storage, installation and assembly as well as careful operation and maintenance.

---

SIMATIC<sup>®</sup> and SIMATIC NET<sup>®</sup> are registered trademarks of SIEMENS AG.

## Trademarks

Since any other brand names in this manual may refer to trademarks, the use of these names by third parties for their own purposes may infringe the rights of the owner.

## Copyright © Siemens AG 2008 All Rights Reserved

Passing on and reproducing this document as well as using and disclosing its contents is prohibited unless an explicit permission is given. Offenders will be liable for damages. All rights reserved, especially in the case of patent grant or registration of a utility model or design.

Siemens AG  
Industry Sector  
Engineering and Construction  
I IS IN E&C  
P.O. Box 3240, D- 91050 Erlangen  
IT4.Industry@siemens.com

## Exclusion of Liability

We have checked the contents of this document with regard to its conformity with the described hardware and software. Deviations, however, cannot be excluded. Therefore, we cannot guarantee its complete conformity. The information in this document is checked regularly and necessary corrections are contained in subsequent versions. We will be grateful for any suggestions for improvement.

Technical data subject to change.

# Preface

## Purpose of the Manual

The information in this manual allows you to set up and put in operation the connection between a PLC with integrated PN interface and a device that supports the Open MODBUS/TCP protocol.

## Contents of the Manual

This manual describes the function of the Modbus function block and its parameterization.

The manual contains the following topics:

- Production description
- Getting Started
- Commissioning
- Parameterization
- Licensing
- Function block MODBUSPN
- Diagnosis
- Sample application

## Scope of this Manual

This manual is valid for the following software:

Product	Identification number	From version
OPEN MODBUS/TCP	2XV9 450-1MB02	2.1
FB 102 "MODBUSPN"		3.1
FB 103 "TCP_COMM"		3.0
FB 104 "MOD_CLI"		1.1
FB 105 "MOD_SERV"		1.0

---

### Note

This manual contains the FB description valid at the time of publication.

---

## Additional Sources of Information

All additional information concerning PN PLCs and IM 151-8 PN/DP CPU (Startup, commissioning etc.) can be found in the manuals

SIEMENS  
 SIMATIC S7-300  
 CPU 31xC and CPU 31x: Installation  
 Operating Instructions  
 A5E00105491-07

SIEMENS  
SIMATIC S7-300  
CPU 31xC and CPU 31x, Technical Specifications  
Manual  
A5E00105474-07

SIEMENS  
SIMATIC S7-400  
Automation System S7-400 Hardware and Installation  
Operating Instructions  
A5E00850740-01

SIEMENS  
SIMATIC S7-400  
S7-400 Automation System, CPU Specifications  
Manual  
A5E00850746-06

SIEMENS  
SIMATIC  
Distributed I/Os ET 200S  
Interface Module IM151-8 PN/DP CPU  
Manual  
A5E02049033-01

SIEMENS  
Product Information on  
CPU315-2 PN/DP, 6ES7315-2EH13-0AB0  
CPU315F-2 PN/DP, 6ES7315-2FH13-0AB0  
CPU317-2 PN/DP, 6ES7317-2EK13-0AB0  
CPU317F-2 PN/DP, 6ES7317-2FK13-0AB0  
CPU317-2 DP, 6ES7317-2AJ10-0AB0  
CPU317F-2 DP, 6ES7317-6FF03-0AB0  
CPU319-3 PN/DP, 6ES7318-3EL00-0AB0  
CPU319F-3 PN/DP, 6ES7318-3FL00-0AB0  
A5E01103134-03

Additional information concerning STEP7 can be found in the following manuals:

SIEMENS  
SIMATIC Software  
Base software for S7 and M7  
STEP7 user manual  
C79000-G7000-C502-..

SIEMENS  
SIMATIC Software  
System software for S7-300/400  
System and standard functions  
Reference manual C79000-G7000-C503-02

<b>Additional Questions</b>	For further questions regarding the use of the FBs described in this manual, please contact your Siemens partner who provided you with this function block.
<b>Terminology</b>	This document uses the term PN PLC. The descriptions apply to <b>PN PLCs</b> of series <b>315, 317, 319, 414</b> and <b>416</b> as well as <b>IM 151-8 PN/DP CPU</b> .
<b>Scope of Application</b>	The function block described in this manual establishes a connection between a PN PLC and third-party MODBUS devices.

# Table of Contents

<b>1</b>	<b>Product Description</b> .....	<b>1-1</b>
1.1	Field of Applications.....	1-1
1.2	Hardware and Software Prerequisites.....	1-1
<b>2</b>	<b>Getting Started</b> .....	<b>2-1</b>
<b>3</b>	<b>Commissioning</b> .....	<b>3-1</b>
3.1	Installing the Library on the STEP7 PG/-PC .....	3-1
3.2	PLC – Assigning the IP Address .....	3-2
3.3	Insertion of the Function Blocks into the Program.....	3-4
<b>4</b>	<b>Parameterization of the Modbus Communication</b> .....	<b>4-1</b>
4.1	Parameterization with the Wizard.....	4-2
4.2	Editing the Parameterization .....	4-3
<b>5</b>	<b>Licensing</b> .....	<b>5-1</b>
<b>6</b>	<b>Function Block MODBUSPN</b> .....	<b>6-1</b>
6.1	Functionality of the FB.....	6-1
6.2	Parameters of the Function Block MODBUSPN .....	6-5
6.3	Example of Address Mapping.....	6-12
6.4	Data and Standard Functions used by the FB .....	6-15
6.5	Renaming Standard Functions.....	6-16
<b>7</b>	<b>Diagnosis</b> .....	<b>7-1</b>
7.1	Diagnosis via the Display Elements of the PLC .....	7-2
7.2	Verification by the FB MODBUSPN.....	7-3
7.3	Diagnosis Messages of the FB MODBUSPN.....	7-6
7.4	Diagnosis Messages of Called Blocks .....	7-12
7.5	Diagnosis Messages of SFC24 .....	7-12
<b>8</b>	<b>Sample Application</b> .....	<b>8-1</b>
<b>A</b>	<b>Literature</b> .....	<b>1</b>

# 1 Product Description

## 1.1 Field of Applications

**Placement in the System Environment** The function block described here is a software product for PLCs with integrated PN interface of Simatic S7-300, S7-400 and IM 151-8 PN/DP CPU.

**Function of the FBs** With these function blocks, communication link is established between a Simatic PLC with integrated PN interface and a device that supports the Open MODBUS/TCP protocol.

Data transmission is carried out in accordance with the client-server principle.

The SIMATIC S7 can act as both, a client and a server, during the data transmission.

**Use of Port Number 502** In general, the protocol uses the port 502. This port number has not been released for all PN PLCs yet and is only possible when using a 319 CPU, an IM 151-8 PN/DP CPU, a 414 CPU or a 416 CPU with the corresponding firmware version. You can find further information about the released port numbers in the relevant manual of your PLC.

The PLC only allows a **one-time use of a specific port number**. Therefore the PLC/FB cannot be addressed simultaneously from different devices on the same port number.

## 1.2 Hardware and Software Prerequisites

**Usable Modules** The function blocks have been tested on the PLCs with the product identifications

CPU315-2 PN/DP	6ES7315-2EG10-0AB0
CPU317-2 PN/DP	6ES7317-2EK13-0AB0
CPU319-3 PN/DP	6ES7318-3EL00-0AB0
CPU414-3 PN/DP	6ES7414-3EM05-0AB0
CPU416-3 PN/DP	6ES7416-3FR05-0AB0
IM151-8 PN/DP	6ES7151-8AB00-0AB0

You can find the latest hardware prerequisites on the internet:  
[www.siemens.com/s7modbus](http://www.siemens.com/s7modbus).

**Software Versions** The usage of the FB MODBUSPN is possible with **STEP7 Version 5.4 SP4** or higher.



**Memory requirements**

The FB MODBUSPN requires 6048 byte work memory and 7294 byte load memory.

The FB MOD\_CLI requires 11184 byte work memory and 12054 byte load memory.

The FB MOD\_SERV requires 10660 byte work memory and 11452 byte load memory.

The FB TCP\_COMM requires 1880 byte work memory and 2224 byte load memory.

## 2 Getting Started

### Procedure

1. Install "OpenModbusTCP PN CPU" and insert the Modbus function blocks into your SIMATIC project.  
=> Section 3.1 to 3.3
2. Parameterize the connection parameters regarding your requirements (IP-address, port number, etc.).  
=> Section 4.1 and 4.2
3. Parameterize the Parameter-DBs MODBUS\_PARAM regarding your requirements (client/server, connect at start-up, register numbers, DB numbers, etc.).  
=> Section 4.1 and 4.2
4. Call the Modbus block FB102 in the required OBs.  
=> Section 6.1
5. Parameterize the Modbus block for initialization and runtime.  
=> Section 6.2
6. Load the user program into the PLC and license the Modbus block for this CPU.  
=> Section 5

## 3 Commissioning

**General Information** The information below on STEP7 refers to Version 5.4 SP4. In later versions, the sequences, names and directories might be different.

**Requirements** Knowledge of AWL and basic knowledge of STEP7 and PLC is required.

### 3.1 Installing the Library on the STEP7 PG/PC

**What We Provide** The attached CD contains a setup which installs the library "Modbus\_PN\_CPU", the sample projects and the manuals in English and German in the corresponding STEP7 directories.

The manuals are also available as PDF files on CD.

**Requirements** **STEP7** has to be installed.

**Installation** Insert your Modbus CD into the CD ROM drive and follow step-by-step the instructions which are displayed automatically by the installation program. If the installation program does not start automatically, please proceed as follows:

1. In the Windows Explorer, navigate to the CD ROM drive. Double-click on the setup directory and then on **Setup.EXE** to start the installation.
2. Follow the step-by-step the instructions which are displayed by the installation program.

Now you can find

- the library in \Program Files\Siemens\Step7\S7libs,
- the sample project in \Program Files\Siemens\Step7\Examples,
- the manual in \Program Files\Siemens\Step7\S7manual\S7Comm,
- the software registration form in  
\Program Files\Siemens\Step7\S7libs\Modbus\_PN\_CPU.

To initially access the Modbus library, use the browse function of the "Open" dialog for libraries.

The manual can be accessed via short cut under \Program Files \Siemens\ Documentation as well.

## 3.2 PLC – Assigning the IP Address

### Introduction

Each node on the Ethernet network is identified by an internationally unique address. This so-called MAC address is preset by the manufacturer and cannot be changed.

Following the steps below, you can assign an IP address in the Ethernet to this physical address.

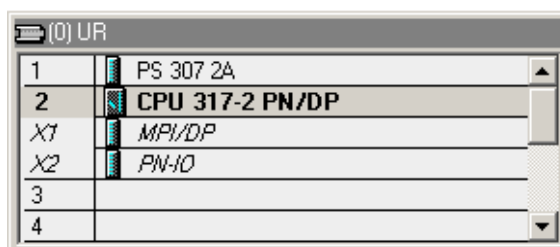
In this example, a PLC 317-2 PN/DP is inserted.

### Procedure

Before the configuration, it is necessary to create a new **S7 project** with STEP7.

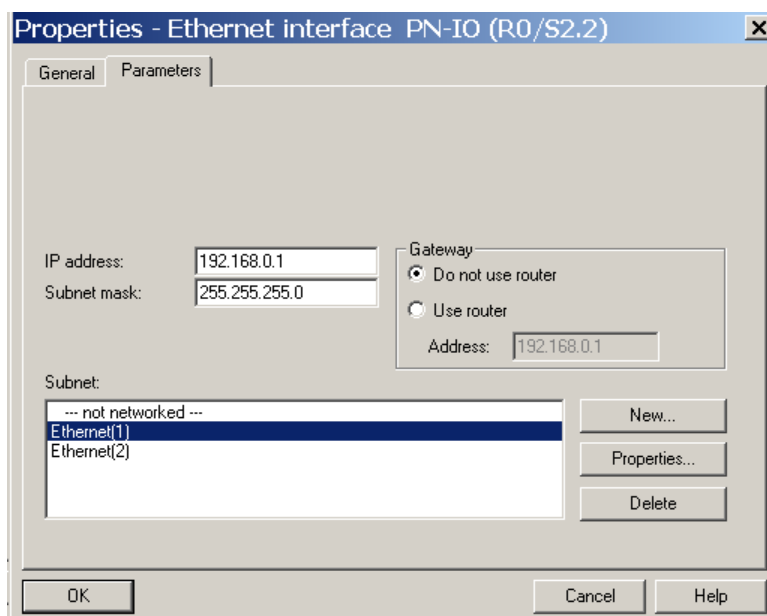
1. Open HWConfig

The PLC 317-2 PN/DP is inserted in slot 2 and the properties dialog box of the PN-IO interface X2 is mapped.



2. A double click on line X2 opens the object properties dialog of Ind. Ethernet.

3. Enter the IP address and the subnet mask.  
To establish a connection via a router, enter the address of the router as well.



4. Click on "New" to assign a name for a new Industrial Ethernet subnet.  
Confirm your entries with "OK".  
Result: You created a new Industrial Ethernet subnet.
5. Click on the "OK" button.  
Result: The properties window of Ethernet interface X2 for PLC 317-2 PN/DP closes.

### 3.3 Insertion of the Function Blocks into the Program

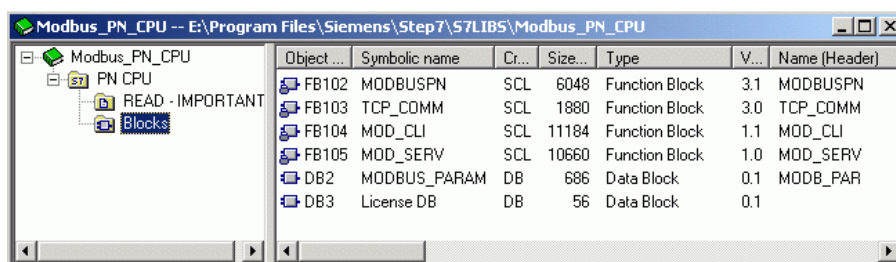
#### Content of the MODBUS library

The following blocks are required for the Modbus communication:

- MODBUSPN
- MOD\_CLI
- MOD\_SERV
- TCP\_COMM

The blocks have to be copied from the library into your project.

Additionally, the library contains the parameter data block MODBUS\_PARAM as a draft. They can be inserted into your project for assistance purposes.



Object ...	Symbolic name	Cr...	Size...	Type	V...	Name (Header)
FB102	MODBUSPN	SCL	6048	Function Block	3.1	MODBUSPN
FB103	TCP_COMM	SCL	1880	Function Block	3.0	TCP_COMM
FB104	MOD_CLI	SCL	11184	Function Block	1.1	MOD_CLI
FB105	MOD_SERV	SCL	10660	Function Block	1.0	MOD_SERV
DB2	MODBUS_PARAM	DB	686	Data Block	0.1	MODB_PAR
DB3	License DB	DB	56	Data Block	0.1	

#### Blocks of the Standard Library

The following blocks are required for the Modbus communication:

- TSEND (FB63)
- TRCV (FB64)
- TCON (FB65)
- TDISCON (FB66).

You can find these blocks in **Standard Library → Communication Blocks**. They have to be inserted into your project.

**Please note** that the following versions of the FBs are a prerequisite for the faultless function of the FB MODBUSPN:

TSEND	V2.1 or higher
TRCV	V2.2 or higher
TCON	V2.3 or higher
TDISCON	V2.1 or higher

Furthermore the block FC10 EQ\_STRNG is necessary. You can find this function in „**Standard Library → IEC Function Blocks**“.

## 4 Parameterization of the Modbus Communication

### General Information

For the communication via the integrated PN interface of the PLC, a network configuration in NetPro is not necessary. The connections are established and terminated by means of the function blocks TCON (FB65) und TDISCON (FB66).

Several connections to different communication partners can be parameterized and established at the same time. The maximal number of coexistent connections depends on the PLC.

### Connection Description DB MODBUS\_PARAM

The parameters to establish a connection and to execute the Modbus communication are defined as a structure in a parameter DB called MODBUS\_PARAM. At first the connection parameters are defined, subsequently, the Modbus parameters are defined.

For each logical connection, a **separate structure** is required. This structure contains the connection parameters of both communication partners and the Modbus parameters. For each additional connection, the parameter data block must be expanded by or a new data block created with the structure for connection parameters and the Modbus parameters.

The parameter data block can contain parameter information of all defined connections. It is also possible to use a separate parameter data block for each connection.

A prepared structure is part of the library "MODBUS\_PN\_CPU" and can be used as a sample.

#### Structure of DB MODBUS\_PARAM:

Address	Name
	STRUCT
+0.0	Connection 1: Connection parameters
+64.0	Connection 1: Modbus parameters
	END_STRUCT
	STRUCT
+650.0	Connection 2: Connection parameters
+714.0	Connection 2: Modbus parameters
	END_STRUCT
...	...
	STRUCT
650*i	Connection i+1: Connection parameters (65+i)
650*i+64	Connection i+1: Modbus parameters
	END_STRUCT

**Connection Parameters**

The parameters of the connection are defined in the first block, e.g. the used local interface and the IP address of the communication partner. The functions TCON and TDISCON can establish and terminate a connection by means of these parameters. A detailed structure can be found in section 4.2.

The structure of the connection parameter block is obligatory und may not be changed. Otherwise it becomes impossible to set up a connection.

**Modbus Parameters**

The Modbus parameters define the mode of communication and the address reference, e.g. how many register or bit areas are mapped to which DB or the differentiation between the S7 acting as server or as client. The structure of the parameter DB has to remain unchanged to ensure proper communication.

**Alternatives for Parameterization**

There are two alternatives for a parameterization of the connection and Modbus parameters. The first one is to use a Wizard, which offers an easy way to setup the connection parameters. The second possibility is to edit the parameters in the structure in the data block with the editor of STEP7.

The two opportunities are described in detail in the following sections 4.1 and 4.2.

## 4.1 Parameterization with the Wizard

**General Information**

With the wizard **Modbus TCP Wizard** you can easily parameterize the connection parameters as well as the Modbus parameters in the parameter data block MODBUS\_PARAM. The complete structure (connection parameters and Modbus parameters) is created.

It is recommended to use the wizard for the parameterization of the block MODBUS\_PARAM.

You can find the wizard at <http://support.automation.siemens.com/WW/view/en/31535566>.



## 4.2 Editing the Parameterization

**Procedure** Copy DB2 of the library **Modbus\_PN\_CPU** and insert them in your project. If the number is already used, rename the DB.

The parameter of data block MODBUS\_PARAM must not be changed during runtime. It is necessary to restart the PLC after the modification of MODBUS\_PARAM.

### Structure of and Adaption of Connection Parameters

For each logical connection one structure is required.

Address	Name	Type	Initial value
0.0		STRUCT	
+0.0	OUCW_1	STRUCT	
+0.0	block_length	WORD	W#16#40
+2.0	id	WORD	W#16#1
+4.0	connection_type	BYTE	B#16#1
+5.0	active_est	BOOL	FALSE
+6.0	local_device_id	BYTE	B#16#2
+7.0	local_tsap_id_len	BYTE	B#16#2
+8.0	rem_subnet_id_len	BYTE	B#16#0
+9.0	rem_staddr_len	BYTE	B#16#0
+10.0	rem_tsap_id_len	BYTE	B#16#0
+11.0	next_staddr_len	BYTE	B#16#0
+12.0	local_tsap_id	ARRAY[1..16]	B#16#D0, B#16#7, B#16#0, B#16#0,
*1.0		BYTE	
+28.0	rem_subnet_id	ARRAY[1..6]	B#16#0, B#16#0, B#16#0, B#16#0,
*1.0		BYTE	
+34.0	rem_staddr	ARRAY[1..6]	B#16#0, B#16#0, B#16#0, B#16#0,
*1.0		BYTE	
+40.0	rem_tsap_id	ARRAY[1..16]	B#16#0, B#16#0, B#16#0, B#16#0,
*1.0		BYTE	
+56.0	next_staddr	ARRAY[1..6]	B#16#0, B#16#0, B#16#0, B#16#0,
*1.0		BYTE	
+62.0	spare	WORD	W#16#0

#### **block\_length**

This parameter describes the length of the connection parameters and must not be changed.

Fixed value: W#16#40

#### **id**

A connection ID is assigned to each logical connection. This ID must be unique within the complete parameter data block and has to be parameterized during the call of FB MODBUSPN. It is used for the internal calls of the T blocks (TCON, TSEND, TRCV and TDISCON).

Value range: W#16#1 to W#16#FFF

<b>connection_type</b>	<p>The structure of the connection is defined here. It is used by the function TCON when establishing the connection. The value depends on the PLC.</p> <p>TCP (compatibility mode):      B#16#01 with CPU315 and 317 &lt;= FW V2.3          TCP:                                B#16#11 with CPU315 and 317 &gt;= FW V2.4,             IM 151-8 PN/DP CPU, CPU 319, CPU 414             and CPU416</p> <p>The values vary depending on the used firmware. You can find further information on the internet:  <a href="http://support.automation.siemens.com/WW/view/en/24294554">http://support.automation.siemens.com/WW/view/en/24294554</a></p>								
<b>active_est</b>	<p>This parameter defines the way the connection is established (active or passive). It is recommended that the Modbus client performs an active connection establishment while the Modbus server establishes passive connections.</p> <p>Active connection establishment:      TRUE          Passive connection establishment:    FALSE</p>								
<b>local_device_id</b>	<p>The <i>local_device_id</i> defines the IE interface of the used PN PLC. The following values are necessary with the different PLC types.</p> <table border="0"> <tr> <td>IM 151-8 PN/DP CPU</td> <td>B#16#1</td> </tr> <tr> <td>CPU315 or 317</td> <td>B#16#2</td> </tr> <tr> <td>CPU319</td> <td>B#16#3</td> </tr> <tr> <td>CPU414 or CPU416</td> <td>B#16#5</td> </tr> </table>	IM 151-8 PN/DP CPU	B#16#1	CPU315 or 317	B#16#2	CPU319	B#16#3	CPU414 or CPU416	B#16#5
IM 151-8 PN/DP CPU	B#16#1								
CPU315 or 317	B#16#2								
CPU319	B#16#3								
CPU414 or CPU416	B#16#5								
<b>local_tsap_id_len</b>	<p>The length of the parameter <i>local_tsap_id</i> (= local port number) is defined here.</p> <p>Active connection establishment:      0          Passive connection establishment:    2</p>								
<b>rem_subnet_id_len</b>	<p>This parameter is currently not used. Please assign the value B#16#0.</p>								
<b>rem_staddr_len</b>	<p>The length of <i>rem_staddr</i>, i.e. the IP address of the communication partner, is defined here. If an unspecified connection is to be used, no IP address is required for the partner.</p> <table border="0"> <tr> <td>Unspecified connection:</td> <td>B#16#0</td> </tr> <tr> <td>Specified connection:</td> <td>B#16#4</td> </tr> </table>	Unspecified connection:	B#16#0	Specified connection:	B#16#4				
Unspecified connection:	B#16#0								
Specified connection:	B#16#4								
<b>rem_tsap_id_len</b>	<p>This parameter defines the length of <i>rem_tsap_id</i>, the port number of the remote communication partner.</p> <p>Active connection establishment:      2          Passive connection establishment:    0</p>								
<b>next_staddr_len</b>	<p>This parameter defines the length of <i>next_staddr</i> as the distinction between the communication running via an external CP or via the integrated PN interface of the PLC.</p> <table border="0"> <tr> <td>PN interface of the PLC:</td> <td>B#16#0</td> </tr> </table>	PN interface of the PLC:	B#16#0						
PN interface of the PLC:	B#16#0								

**local\_tsap\_id** This parameter defines the local port number. The representation depends on the parameter *connection\_type*. The value range depends on the PLC. The port number has to be unique within the PLC.

With connection\_type B#16#01

local\_tsap\_id[1] low byte of the local port number in hexadecimal  
 local\_tsap\_id[2] high byte of the local port number in hexadecimal  
 local\_tsap\_id[3-16] B#16#00

With connection\_type B#16#11

local\_tsap\_id[1] high byte of the local port number in hexadecimal  
 local\_tsap\_id[2] low byte of the local port number in hexadecimal  
 local\_tsap\_id[3-16] B#16#00

**rem\_subnet\_id** This parameter is currently not used. Please assign the value B#16#0.

**rem\_staddr** In this array of bytes, the IP address of the remote communication partner is defined. When an unspecified connection is used, no IP address has to be entered. The representation depends on the parameter *connection\_type*.

Example: IP address 192.168.0.1:

With connection\_type B#16#01

rem\_staddr[1] = B#16#01 (1),  
 rem\_staddr[2] = B#16#00 (0),  
 rem\_staddr[3] = B#16#A8 (168),  
 rem\_staddr[4] = B#16#C0 (192),  
 rem\_staddr[5-6]= B#16#00 (reserved)

With connection\_type B#16#11

rem\_staddr[1] = B#16#C0 (192),  
 rem\_staddr[2] = B#16#A8 (168),  
 rem\_staddr[3] = B#16#00 (0),  
 rem\_staddr[4] = B#16#01 (1),  
 rem\_staddr[5-6]= B#16#00 (reserved)

**rem\_tsap\_id** This parameter defines the remote port number. The representation depends on the parameter *connection\_type*. The value range depends on the PLC.

With connection\_type B#16#01

local\_tsap\_id[1] low byte of the remote port number in hexadecimal  
 local\_tsap\_id[2] high byte of the remote port number in hexadecimal  
 local\_tsap\_id[3-16] B#16#00

With connection\_type B#16#11

local\_tsap\_id[1] high byte of the remote port number in hexadecimal  
 local\_tsap\_id[2] low byte of the remote port number in hexadecimal  
 local\_tsap\_id[3-16] B#16#00

**next\_staddr** This parameter defines rack and slot of the corresponding CP. When using the integrated PN interface of the PLC, assign 0 to this parameter.

next\_staddr[1-6] B#16#00

**spare** Reserved; assign 0 to this parameter.

**Adaption of the Modbus Parameters**

The Modbus parameters in the block MODBUS\_PARAM define the mode of operation of the Modbus communication and the address reference of Modbus addresses and SIMATIC addresses.

+64.0	server_client	BOOL	TRUE
+64.1	single_write	BOOL	FALSE
+64.2	connect_at_startup	BOOL	FALSE
+65.0	reserved	BYTE	B#16#0
+66.0	data_type_1	BYTE	B#16#3
+68.0	db_1	WORD	W#16#B
+70.0	start_1	WORD	W#16#1
+72.0	end_1	WORD	W#16#1F4
+74.0	data_type_2	BYTE	B#16#3
+76.0	db_2	WORD	W#16#C
+78.0	start_2	WORD	W#16#2D0
+80.0	end_2	WORD	W#16#384
+82.0	data_type_3	BYTE	B#16#4
+84.0	db_3	WORD	W#16#D
+86.0	start_3	WORD	W#16#2D0
+88.0	end_3	WORD	W#16#3E8
+90.0	data_type_4	BYTE	B#16#0
+92.0	db_4	WORD	W#16#4
+94.0	start_4	WORD	W#16#0
+96.0	end_4	WORD	W#16#64
+98.0	data_type_5	BYTE	B#16#1
+100.0	db_5	WORD	W#16#E
+102.0	start_5	WORD	W#16#280
+104.0	end_5	WORD	W#16#4E2
+106.0	data_type_6	BYTE	B#16#2
+108.0	db_6	WORD	W#16#F
+110.0	start_6	WORD	W#16#6A4
+112.0	end_6	WORD	W#16#8FC
+114.0	data_type_7	BYTE	B#16#1
+116.0	db_7	WORD	W#16#10
+118.0	start_7	WORD	W#16#6A4
+120.0	end_7	WORD	W#16#8FC
+122.0	data_type_8	BYTE	B#16#0
+124.0	db_8	WORD	W#16#8
+126.0	start_8	WORD	W#16#0
+128.0	end_8	WORD	W#16#64
+130.0	internal_send_buffer	ARRAY[1..260]	B#16#0
*1.0		BYTE	
+390.0	internal_recv_buffer	ARRAY[1..260]	B#16#0
*1.0		BYTE	
=650.0		END_STRUCT	
=650.0		END_STRUCT	

**server\_client**

TRUE: S7 is server  
FALSE: S7 is client

**single\_write**

In operating mode "S7 is client" and single\_write = TRUE write requests with length 1 are carried out with the function codes 5 and 6.

With single\_write = FALSE all write requests use the function codes 15 and 16.



***start\_x***                      *Start\_x* specifies the first register or bit address, which is stored in the data element 0 of the DB. *End\_x* defines the of the last MODBUS address.  
***end\_x***

When accessing registers, the number of the data element of the S7 DB in which the last register is mapped, can be calculated with the following formula:

$$\text{DBW number} = (\text{end\_x} - \text{start\_x}) * 2$$

When accessing coils or inputs, the number of the data element of the S7 DB in which the last bit is mapped, can be calculated with the following formula:

$$\text{DBB number} = (\text{end\_x} - \text{start\_x} + 7) / 8$$

The defined memory areas must not overlap. The parameter *end\_x* must not be smaller than *start\_x*. In case of an error occurring, the initialization of the FB is stopped with an error. When *start\_x* is equal to *end\_x*, one Modbus address (1 register or 1 bit) is allocated.

In section 6.3 you can find an example of the mapping of the MODBUS addresses to S7 memory areas.

*start\_x, end\_x*  
 MODBUS address                      0 to 65535 (W#16#0000 to W#16#FFFF)

***internal\_send\_buffer***                      This array is used internally for message data within the FB. Accessing or changing the array is inadmissible.

***internal\_rcv\_buffer***                      This array is used internally for the received data within the FB. Accessing or changing the array is inadmissible.

## 5 Licensing

### General

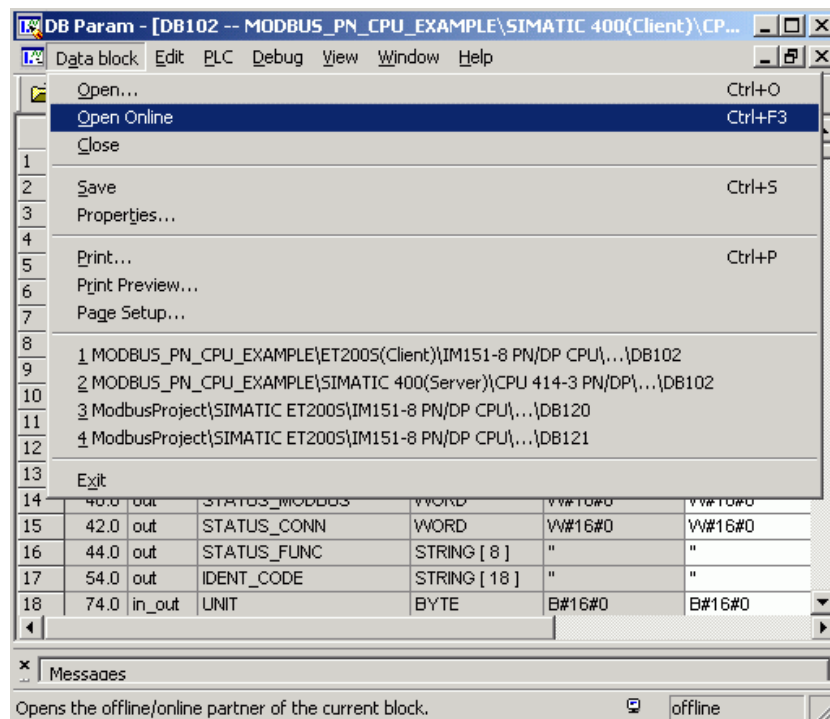
The block MODBUSPN must be licensed for each CPU individually. The licensing takes place in two steps: reading the IDENT\_CODE and declaring the registration key REG\_KEY.

### Read the IDENT\_CODE

To read the IDENT\_CODE please proceed as follows:

1. Parameterize the block in the cyclic OB (OB1 or cyclic interrupt OB), in OB100 and OB121 according to your requirements. Transfer the program to the PLC and turn it to RUN mode.
2. Open the instance DB of the block MODBUSPN. „Data block“ -> „Open Online“ to open the DB.

Monitoring the block via the button  is insufficient.







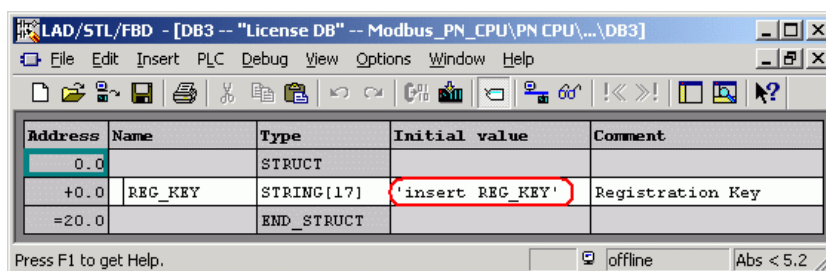
## Declaration of the Registration Key REG\_KEY

The registration key REG\_KEY must be declared for each call of the block MODBUSPN.

The registration key REG\_KEY should be stored in a global DB. Via this global DB all MODBUSPN blocks can receive the registration key (See also the following example).

Please proceed as follows to declare the registration key REG\_KEY:

1. Copy the prepared license block DB3 of the library "Modbus\_PN\_CPU" into your project. If the DB number is already used in your project, rename the license DB.
2. Open the license DB and copy the 17 digit registration key you received from IT4industry to the column "Initial value".



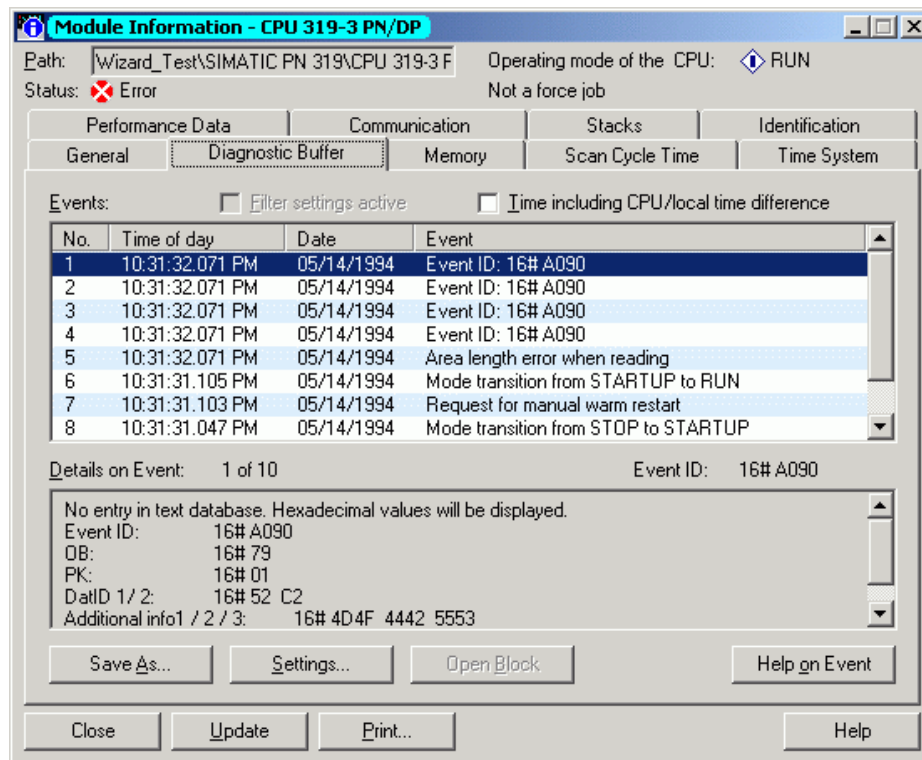
Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	REG_KEY	STRING[17]	'insert REG_KEY'	Registration Key
=20.0		END_STRUCT		

3. Declare the registration key in the data block as "initial value" to avoid a repeated insertion after reloading the PLC. Open the data block in the SIMATIC manager with the editor in the declaration view. Change over to the data view via the menu "View" -> "Data View". Choose in the menu "Edit > Initialize Data Block" – all values of the column "initial value" are copied to "actual values".
4. Assign the value "DB3.REG\_KEY" to the parameter REG\_KEY of the block MODBUSPN.
5. Transfer the changed blocks to the PLC. The registration key can be set at run time. A STOP -> RUN transition is not necessary.

The block is now licensed for this CPU.

## Missing or Wrong Licensing

When the registration key is missing or a wrong one is detected, the SF LED (S7-300 and IM151-8) or the INTF LED (S7-400) of the CPU is flashing. A cyclic error message regarding the missing license is displayed in the diagnostic buffer too. The error number of the missing license is W#16#A090.



Module Information - CPU 319-3 PN/DP

Path: \Wizard\_Test\SIMATIC PN 319\CPU 319-3 F Operating mode of the CPU: RUN  
 Status: Error Not a force job

Performance Data Communication Stacks Identification  
 General Diagnostic Buffer Memory Scan Cycle Time Time System

Events:  Filter settings active  Time including CPU/local time difference

No.	Time of day	Date	Event
1	10:31:32.071 PM	05/14/1994	Event ID: 16# A090
2	10:31:32.071 PM	05/14/1994	Event ID: 16# A090
3	10:31:32.071 PM	05/14/1994	Event ID: 16# A090
4	10:31:32.071 PM	05/14/1994	Event ID: 16# A090
5	10:31:32.071 PM	05/14/1994	Area length error when reading
6	10:31:31.105 PM	05/14/1994	Mode transition from STARTUP to RUN
7	10:31:31.103 PM	05/14/1994	Request for manual warm restart
8	10:31:31.047 PM	05/14/1994	Mode transition from STOP to STARTUP

Details on Event: 1 of 10 Event ID: 16# A090

No entry in text database. Hexadecimal values will be displayed.  
 Event ID: 16# A090  
 OB: 16# 79  
 PK: 16# 01  
 DatID 1 / 2: 16# 52 C2  
 Additional info1 / 2 / 3: 16# 4D4F 4442 5553

Save As... Settings... Open Block Help on Event  
 Close Update Print... Help

The entry in the diagnostic buffer is carried out by means of OB121 "Programming Error".



### Warning

**The CPU will turn to STOP mode, if the OB121 is not available.**

Modbus communication is carried out even with a missing or wrong registration key, but the output STATUS\_MODBUS is set to W#16#A090 "no valid license".

If this error code is shown although the registration key has been inserted, please check if the FC "EQ\_STRING" is copied into the project.

## 6 Function Block MODBUSPN

### 6.1 Functionality of the FB

#### General Information

The function block MODBUSPN enables a communication between a PLC with integrated PN interface and a partner which supports Open MODBUS/TCP.

The function codes 1, 2, 3, 4, 5, 6, 15 and 16 are supported.

Depending on the parameterization, the FB can be operated either in client or in server mode. In the server operating mode, the functionality multitasking in the sense of the MODBUS specification is not implemented.

The block MODBUSPN calls the blocks MOD\_CLI (FB104) and MOD\_SERV (FB105) internally. The block MOD\_CLI comprises the function of Modbus client, the block MOD\_SERV executes the function of Modbus server and the block TCP\_COMM handles the connection management.

The blocks provide the following functions:

- Connection and data handling by means of T blocks of the standard library
- Generation of MODBUS-specific telegram header before sending
- Verification of the MODBUS-specific telegram header when receiving
- Verification whether the memory areas requested by the client exist
- Generation of exception telegrams when errors occur (only when S7 is in server mode)
- Data transfer to and from the parameterized DB
- Time monitoring of the data reception as well as connection establishment and termination
- Verification of the registration key

#### Online-Help

The SIMATIC Manager provides an online help for the function block MODBUSPN. Mark the FB and press "F1" so that the online help is displayed. It contains the main information on the FB.

**Call of the FB**

For a correct program sequence the function block MODBUSPN has to be called in 3 organization blocks:

- OB100 Start-Up
- OB121 Programming Error
- Cyclic OB (OB1 or cyclic interrupt OB, e.g. OB35)

The subordinate blocks of the Modbus library MOD\_CLI, MOD\_SERV and TCP\_COMM must not be called additionally in an organization block.

A coexistent call of FB MODBUSPN in OB1 and a cyclic interrupt OB, e.g. OB35 is not permissible.

The Modbus block must be called in OB121. You will find more information regarding this matter in **section 5 “Licensing”**.

**Start-up of the FB**

The function block MODBUSPN is unconditionally called once in OB100.

- The initialization parameters must be set according to the station configuration.
- The initialization parameters are copied into the instance DB.
- The runtime parameters are not evaluated during the start-up.
- The values of the parameter data block MODBUS\_PARAM are evaluated.

**Cyclical Operation of the FB**

In case of cyclical operation, the FB MODBUSPN is called in OB1 or in a cyclic interrupt OB.

- Corresponding to the runtime parameters, the functions of the function block are activated.
- While a request is being processed, changes to the runtime parameters are ignored.
- In the cyclical operation mode, initialization parameters are ignored.

**OB121 “Programming Error”**

The block MODBUSPN must be called in OB121 with the same instance DB as in the start-up OB and the cyclic OB.

If the block has not been licensed yet, the OB121 is called.

**Warning**

**The CPU will turn to STOP mode, if the OB121 is not available.**

The call of MODBUSPN in OB121 executes an entry in the diagnostic buffer, which points out that the registration key is missing. The LED SF or INTF of the CPU flashes simultaneously.

**Connection Handling**

Active connection establishment can be carried out by the Modbus client as well as the Modbus server. It is recommended explicitly that the Modbus client executes the active establishment. The relevant information is read from the connection parameters of DB MODBUS\_PARAM.

With a parameter of the connection parameter block (*active\_est*), it is possible to define whether the PLC performs active or passive connection establishment.

During runtime the function TCON establishes the connection to the communication partner for both types of connections, active and passive.

The point in time at which the connection is established is defined in DB MODBUS\_PARAM (parameter *connect\_at\_startup*).

The termination of the connection is defined with the parameter DISCONNECT of FB MODBUSPN.

**Multiple Communication Partners**

A PN PLC can establish multiple connections to various communication partners. To ensure a proper operation and data transfer, a strict division of the different connections to the corresponding communication partners is inevitable.

Correspondingly, there are the following requirements for each connection:

- one connection parameter block and the related Modbus parameter in DB MODBUS\_PARAM
- call of FB MODBUSPN in OB100
- call of FB MODBUSPN in OB121
- call of FB MODBUSPN in OB1 or cyclic interrupt OB

In this case, the calls of the FB MODBUSPN in the OB100, OB121 and in the OB1 receive the same instance DB for one connection. A different instance DB is required for each additional connection.

**Initiate Request  
S7 is Client**

A rising edge at the trigger input ENQ\_ENR initiates a request. Depending on the input parameters UNIT, DATA\_TYPE, START\_ADDRESS, LENGTH, TI and WRITE\_READ, a MODBUS request telegram is generated and sent to the partner station via the TCP/IP connection. The client waits for the parameterized monitoring time RECV\_TIME for a response from the server. When the monitoring time elapses (no response from the server), the activated request is terminated with an error. A new request can be initiated.

After the receipt of the response telegram, a validity check is carried out. If the result is positive, the necessary actions are taken and the request is terminated without error. The output DONE\_NDR is set. When an error is recognized during verification, the request is terminated with an error, the ERROR bit is set and an error number is returned at the output STATUS.

**Activation of the  
Function Block  
S7 is Server**

With the signal TRUE at the trigger input ENQ\_ENR, the FB is ready to receive a request telegram from the client. The server remains passive and waits for a telegram from the client. The received telegram is verified. If the verification result is positive, the response telegram is sent. The completed transmission is reported to the user by setting the DONE\_NDR bit. At this point, the completed function is indicated at the outputs UNIT, DATA\_TYPE, START\_ADDRESS, LENGTH, TI and WRITE\_READ.

An erroneous request telegram causes an error message and the ERROR bit is set. The error number is returned in STATUS\_MODBUS. The request of the client is not answered.

## 6.2 Parameters of the Function Block MODBUSPN

Parameter	Decl.	Type	Description	Value range	Init
ID	IN	WORD	Connection ID, must be identical to the associated parameter in the local connection description	1 to 4095 W#16#1 to W#16#FFF	yes
DB_PARAM	IN	BLOCK_DB	Number of the parameter DB	depending on PLC	yes
RECV_TIME	IN	TIME	Monitoring Time: wait for data from communication partner  Shortest adjustable time: 20 ms.	T#20ms to T#+24d20h31 m23s647ms	no
CONN_TIME	IN	TIME	Monitoring Time: wait for establishing or termination of the connection  Shortest adjustable time: 100 ms.	T#20ms to T#+24d20h31 m23s647ms	no
KEEP_ALIVE	IN	TIME	Not used		
ENQ_ENR	IN	BOOL	S7 is Client: Initiate request at positive edge  S7 is Server: Ready to receive at TRUE signal	TRUE FALSE	no
DISCONNECT	IN	BOOL	S7 is Client: TRUE: connection is terminated after reception of response  S7 is Server: TRUE: connection is terminated when ENQ_ENR = FALSE	TRUE FALSE	no
REG_KEY	IN	STRING [17]	Registration key to activate the license	Character	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 T functions (TCON, TDISCON, TSEND or TRCV)  Job processing  No job processing active	TRUE FALSE	no
CONN_ESTABLISHED	OUT	BOOL	Connection established Connection terminated	TRUE FALSE	no
DONE_NDR	OUT	BOOL	S7 is Client: TRUE: Active request finished without errors  S7 is Server: TRUE: Request from the client was executed and answered	TRUE FALSE	no

Parameter	Decl.	Type	Description	Value range	Init
ERROR	OUT	BOOL	An error has occurred No error has occurred	TRUE FALSE	no
STATUS_MODBUS	OUT	WORD	Error number for protocol errors when evaluating a Modbus telegram	0 to FFFF	no
STATUS_CONN	OUT	WORD	Error number for connection errors during execution of the T blocks (TCON, TSEND, TRCV, TDISON)	0 to FFFF	no
STATUS_FUNC	OUT	STRING [8]	Name of the function, which causes the error at STATUS_MODBUS or STATUS_CONN	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 W#16#0000 to W#16#FFFF	no
LENGTH	IN/ OUT	WORD	Number of values 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 Register</u> Reading function Writing function  <u>Input Register</u> Reading 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 if in CLIENT mode, OUTPUT if in SERVER mode)	0 to 65535 W#16#0 to W#16#FFFF	no
WRITE_READ	IN/ OUT	BOOL	Write access or Read access (INPUT if in CLIENT mode, OUTPUT if in SERVER mode)	TRUE FALSE	no



**General Information**

The parameters of the FB MODBUSPN can be divided into two groups:

- Initialization parameters
- Runtime parameters

**Initialization parameters** are evaluated only during the call of OB100 and are adopted into the instance DB. They are marked with “yes“ in the column “INIT“ in the table displayed above.

A modification of the initialization parameters during run mode has no impact. After a modification of these parameters (e.g. during the test phase), the instance DB must be initialized again via a STOP → RUN transition of the PLC.

**Runtime parameters** can be modified during the cyclical operation. In the mode **S7 is server**, it is not advisable to modify the input parameters while a request is active. Wait with the next request and the change of the parameters until the previous request ends with DONE\_NDR or ERROR.

In the operation mode **S7 is server**, the output parameters may only be evaluated when DONE\_NDR is TRUE.

The output parameters are **displayed dynamically**, i.e. they are only available for **one PLC cycle**. They have to be copied to an additional memory area if you need to process them or to display the values in a VAT (STEP7 variable table).

**Range of Values**

For the range of values of the different parameters, PLC-specific restrictions must be taken into consideration.

- ID** To each connection between the PN PLC and a communication partner a connection ID is assigned. When operating various connections an individual ID is required for each logical connection. This connection ID is defined in the connection parameter block, which is part of the parameter data block MODBUS\_PARAM. The connection ID unambiguously specifies the connection between the PLC and the communication partner. The ID can be set to values between 1 and 4095.  
The connection ID defined in the connection parameter block has to be entered here and must be unique within the PLC.
- DB\_PARAM** The parameter DB\_PARAM assigns the number of the data block MODBUS\_PARAM. This parameter data block includes the connection parameters and the Modbus-specific parameters, which are necessary for the communication between the PN PLC and the Modbus device.  
  
This parameter is declared in plain text: "DBxy".  
  
The range of values for this parameter depends on the PLC. 0 cannot be used as a DB number since it is reserved for system functions.  
  
The parameter data block can contain a sequence of parameters for several connections. It is also possible to use different parameter data blocks for multiple connections.
- RECV\_TIME** The monitoring time RECV\_TIME observes the data input from the communication partner. The shortest adjustable time is 20 ms.  
  
In the operating mode **S7 is client**, an indication RECV\_TIME < 20 ms causes an error message and the request is rejected. When the time RECV\_TIME elapses, the active request is cancelled with an error. In order to receive a Modbus telegram, TRCV has to be called twice. With each call of TRCV, the RECV\_TIME is triggered. The monitoring time is started with the call of TRCV and stopped after the receipt of the telegram segment.  
  
In the operating mode **S7 is server**, an indication RECV\_TIME < 20 ms causes the use of the default time of 1,2s. If the monitoring time elapses, an error is reported. The RECV\_TIME monitors the duration of the TCP stream. The interval between two requests is not taken into consideration.
- CONN\_TIME** The monitoring time CONN\_TIME observes the connection establishment and the connection termination. The shortest adjustable time is 100 ms.  
  
When CONN\_TIME elapses, the corresponding error code is displayed at the output STATUS\_CONN.  
  
In the operating mode **S7 is server**, an indication CONN\_TIME < 100 ms causes the use of the default time 5s. This is also true for the operating mode **S7 is client** with *connect\_at\_startup* = TRUE.  
  
During cyclic operation in the operating mode **S7 is client**, CONN\_TIME < 100ms causes an error message and the request is rejected.

<b>ENQ_ENR</b>	<p>Operating mode <b>S7 is Client:</b> The data transfer is initiated with a TRUE edge at the input. The request telegram is generated with the values of the input parameters UNIT, DATA_TYPE, START_ADDRESS, LENGTH, TI and WRITE_READ. A new request may only be initiated when the previous one is ended with DONE_NDR or ERROR. If the connection is not established (CONN_ESTABLISHED = FALSE), the connection establishment is carried out first and then the data transfer takes places.</p> <p>Operating mode <b>S7 is Server:</b> The FB is activated with a TRUE signal at the input. Telegrams from the client can be received. If the connection is not established (CONN_ESTABLISHED = FALSE), the connection establishment is activated. When the parameter ENQ_ENR turns from TRUE to FALSE, the connection may be terminated depending on the parameter DISCONNECT. With a FALSE signal at the input and an established connection, data is received from the client and discarded.</p>
<b>DISCONNECT</b>	<p>In the operating mode <b>S7 is client</b>, the parameter DISCONNECT = TRUE indicates that the connection is terminated after the completed data transfer.</p> <p>In the operating mode <b>S7 is server</b>, the parameter DISCONNECT = TRUE indicates that the connection is terminated when the parameter ENQ_ENR is set to FALSE.</p> <p>This parameter is a runtime parameter and can be set optionally according to your requirements.</p>
<b>REG_KEY</b>	<p>The block MODBUSPN must be licensed for each CPU individually to permit a correct program sequence.</p> <p>With the registration key REG_KEY the block MODBUSPN is licensed and the Modbus communication runs without any restraint.</p> <p>You can find further information in <b>section 5 “Licensing”</b>.</p>
<b>BUSY</b>	<p>If this output is TRUE, one of the T functions TCON, TDISCON, TSEND or TRCV is running.</p>
<b>CONN_ESTABLISHED</b>	<p>CONN_ESTABLISHED indicates that a connection to the communication partner is established and data can be transferred.</p> <p>If CONN_ESTABLISHED is set to FALSE, the connection to the communication partner is not established.</p>

<b>DONE_NDR</b>	<p>The parameter DONE_NDR indicates an error-free execution of the request.</p> <p>In the operating mode <b>S7 is Client</b>, the activated request was executed without an error. With a reading function, the response data from the server has already been entered into the DB. With a writing function, the response to the request telegram has been received from the server.</p> <p>In the operating mode <b>S7 is Server</b>, this output indicates a telegram exchange without errors. In the parameters UNIT, DATA_TYPE, START_ADDRESS, LENGTH, TI and WRITE_READ the request parameters of the client are displayed. These outputs are only available and valid as long as DONE_NDR is TRUE.</p>
<b>ERROR</b>	<p>When this output is set, an error was recognized.</p> <p>In the operating mode <b>S7 is Client</b>, the activated request was ended with an error. The error number is displayed in the STATUS_MODBUS or STATUS_CONN output.</p> <p>In the operating mode <b>S7 is Server</b>, an error was detected at a request telegram of the client or when sending a response telegram. The error number is displayed in the STATUS_MODBUS or STATUS_CONN output.</p>
<b>STATUS_MODBUS</b>	<p>When ERROR is TRUE, the STATUS_MODBUS output displays the error number regarding the processing of Modbus telegrams. The error numbers are described in section 7.</p>
<b>STATUS_CONN</b>	<p>When ERROR is TRUE, the STATUS_CONN output displays the error number regarding the processing of the T functions. The error numbers are described in section 7 and in the STEP7 online help of the functions TCON, TDISCON, TSEND and TRCV.</p> <p>Status information like "job processing active" is provided by STATUS_CONN as well. In this case, the ERROR bit is set to FALSE.</p>
<b>STATUS_FUNC</b>	<p>This parameter shows the name of the function, which caused the error occurred.</p>
<b>IDENT_CODE</b>	<p>With the identification string IDENT_CODE you can order the registration key at IT4industry. After start-up of the PLC an 18 character string is displayed, which can be read by means of a variable table?</p> <p>You can find further information in <b>section 5 "Licensing"</b>.</p>
<b>UNIT</b>	<p>In mode <b>S7 is Client</b>, the parameter UNIT is an input parameter. This input has to be set according to your requirements. The FB copies this value to the request telegram and verifies when receiving the respond telegram.</p> <p>In mode <b>S7 is Server</b>, the parameter UNIT is an output parameter. The FB copies this value from the request telegram to the respond telegram. The output is set with the received value when the job is finished without an error.</p>

**DATA\_TYPE**

The parameter DATA\_TYPE defines which Modbus data type is to be accessed with the current job. The following data types are available:

Coils	B#16#1
Inputs	B#16#2
Holding Register	B#16#3
Input Register	B#16#4

In the operating mode **S7 is Client**, DATA\_TYPE is an input parameter. In the operating mode **S7 is Server**, DATA\_TYPE is an output parameter.

The different data types are related directly to the used function codes.

Data type	DATA_TYPE	Function	Length	single_write	Function code
Coils	1	read	any	irrelevant	1
Coils	1	write	1	TRUE	5
Coils	1	write	1	FALSE	15
Coils	1	write	>1	irrelevant	15
Inputs	2	read	any	irrelevant	2
Holding Register	3	read	any	irrelevant	3
Holding Register	3	write	1	TRUE	6
Holding Register	3	write	1	FALSE	16
Holding Register	3	write	>1	irrelevant	16
Input Register	4	read	any	irrelevant	4

**START\_ADDRESS**

The parameter START\_ADDRESS specifies the first MODBUS address that is read or written.

In the operating mode **S7 is Client**, this is an input parameter. In the operating mode **S7 is Server**, this is an output parameter.

**LENGTH**

The parameter LENGTH specifies the number of MODBUS values that are read or written.

In the operating mode **S7 is Client**, this is an input parameter. In the operating mode **S7 is Server**, this is an output parameter.

With a reading function, a maximum of 125 registers or 2000 bits is possible per telegram. With a writing function, a maximum of 100 registers or 800 bits is possible.

For each telegram, all registers or bits have to be in the same DB.

- TI** The parameter TI (Transaction Identifier) is copied by the server from the request telegram to the respond telegram according to the MODBUS specification.
- In the operating mode **S7 is Client**, this is an input parameter. The FB copies this value to the request telegram and verifies it when receiving the respond telegram.
- In the operating mode **S7 is Server**, this is an output parameter. The FB copies this value from the request telegram into the respond telegram.
- The Transaction Identifier is used for the identification of telegrams and the unambiguous allocation of the requests to the corresponding responds. The FB MODBUSPN can only perform this function properly if the TI is changed with each transaction. Only a change of the TI ensures a reliable operation of the FB.
- Therefore, we recommend increment the TI by 1 with any request.
- WRITE\_READ** This parameter defines whether a reading or writing function is to be carried out. If the value of the input/output is FALSE, the reading mode is specified. The value TRUE specifies the writing mode.
- In the operating mode **S7 is Client**, this is an input parameter. In the operating mode **S7 is Server**, this is an output parameter.

### 6.3 Example of Address Mapping

- Interpretation of MODBUS Register Addresses** The MODBUS data model is based on a series of storage areas with distinct properties. Some systems, e.g. MODICON PLCs, distinguish these areas by the register or bit addresses. The Holding Register with offset 0, for example, is called 40001 (memory type 4xxxx, reference 0001).
- A potential source of confusion is the varying interpretation of the register address in different manuals. Sometimes the register or bit address is defined as the address of the application layer and in other manuals it is the actually transferred address.
- The **FB MODBUSPN uses the register or bit address transferred** for its parameters *start\_x*, *end\_x* und START\_ADDRESS. Therefore, it is possible to use register or bit addresses from von 0000<sub>H</sub> to FFFF<sub>H</sub> for each function code.

**Example:**  
**Parameterization  
of the Memory  
Areas**

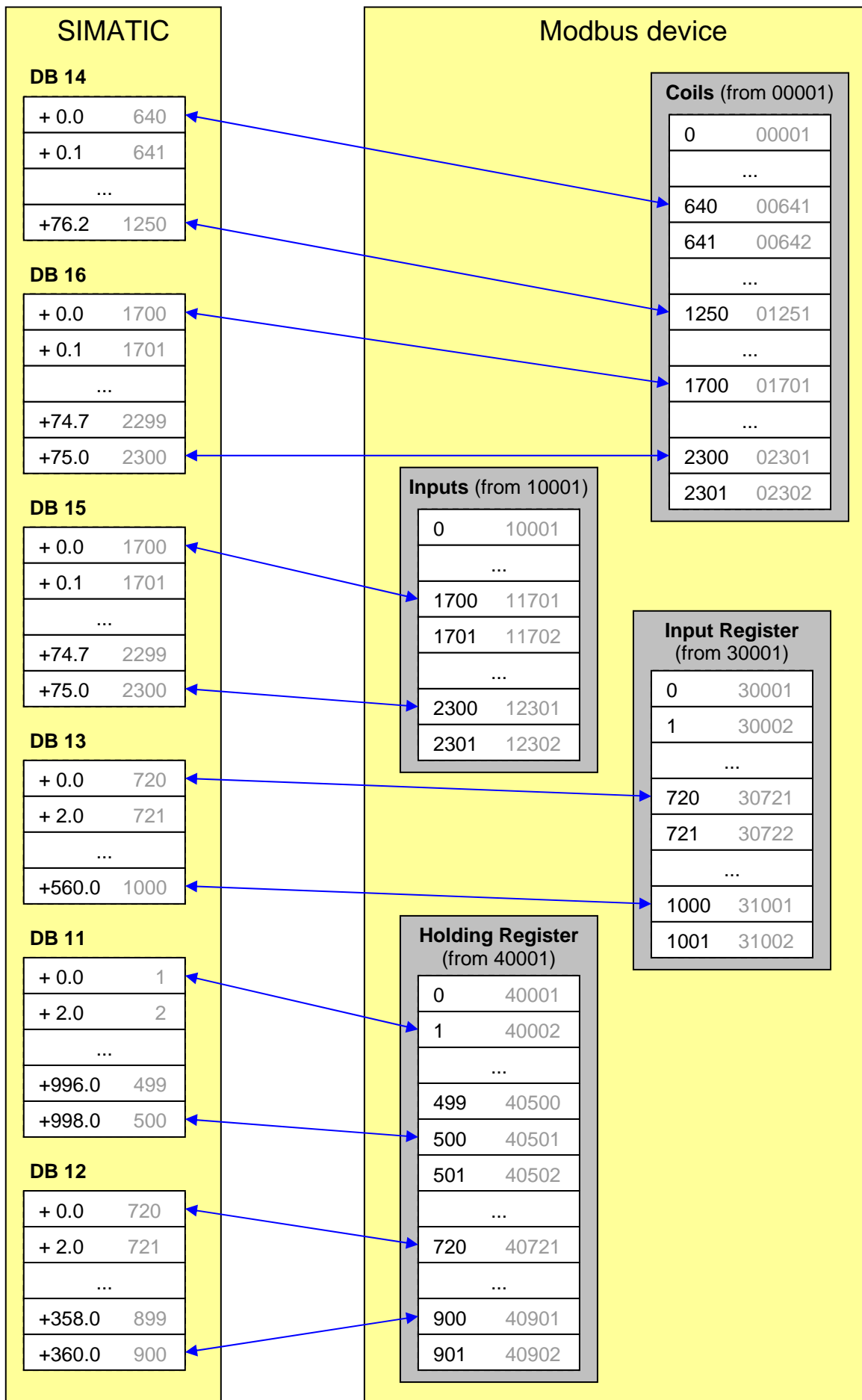
<i>data_type_1</i> <i>db_1</i> <i>start_1</i> <i>end_1</i>	B#16#3 W#16#B W#16#1 W#16#1F4	Holding Register DB 11 Start address: 1 End address: 500
<i>data_type_2</i> <i>db_2</i> <i>start_2</i> <i>end_2</i>	B#16#3 W#16#C W#16#2D0 W#16#384	Holding Register DB 12 Start address: 720 End address: 900
<i>data_type_3</i> <i>db_3</i> <i>start_3</i> <i>end_3</i>	B#16#4 W#16#D W#16#2D0 W#16#3E8	Input Register DB 13 Start address: 720 End address: 900
<i>data_type_4</i> <i>db_4</i> <i>start_4</i> <i>end_4</i>	B#16#0 0 0 0	Not used 0 0 0
<i>data_type_5</i> <i>db_5</i> <i>start_5</i> <i>end_5</i>	B#16#1 W#16#E W#16#280 W#16#4E2	Coils DB 14 Start address: 640 End address: 1250
<i>data_type_6</i> <i>db_6</i> <i>start_6</i> <i>end_6</i>	B#16#2 W#16#F W#16#6A4 W#16#8FC	Inputs DB 15 Start address: 1700 End address: 2300
<i>data_type_7</i> <i>db_7</i> <i>start_7</i> <i>end_7</i>	B#16#1 W#16#10 W#16#6A4 W#16#8FC	Coils DB 16 Start address: 1700 End address: 2300
<i>data_type_8</i> <i>db_8</i> <i>start_8</i> <i>end_8</i>	B#16#0 0 0 0	Not used 0 0 0

**Address Mapping**

The following diagram shows a comparison of the SIMATIC memory area and the Modbus-typical register oriented storage arrangement. The diagram is based on the parameterization in the table above.

The Modbus addresses printed in black refer to the Data Link Layer and the ones printed in grey refer to the Application Layer.

The SIMATIC addresses printed in black are the offset in the DB. Additionally, you can find the Modbus register numbers printed in grey.





## 6.4 Data and Standard Functions used by the FB

**Instance DB** The function block MODBUS stores its data in an instance DB. This instance DB is created by STEP7 when initially calling the FB.

The instance data block contains parameters of the types input, output and input/output as well as static variables required for its execution. These variables are remanent and keep their validity between FB calls. The variables control the internal process flow of the FB.

Required memory of the instance-DBs:

Instance-DB	work memory	load memory
MODBUSPN	830 Byte	1852 Byte

**Local Variables** The FB requires 96 bytes of local variables. Additionally, MOD\_CLI (110 Byte) or MOD\_SERV (84 Byte) and TCP\_COMM require 2 bytes local variables and TCON, TDISCON, TSEND or TRCV use up to 28 bytes depending on the used function block. That is a total of max. 236 bytes of local data for one FB MODBUSPN call.

**Parameter DB** The connection- and Modbus-specific parameters are stored in the parameter data block MODBUS\_PARAM.

**Timers** The FB does not use any timers.

**Flags** The function block does not use any flags.

**Standard FBs for Connection Handling** The function block TCP\_COMM, called by MODBUSPN, uses the blocks TCON and TDISCON of the standard library to establish and terminate the connection between the PLC and the communication partner.

**Standard FBs for Data Transfer** The function block TCP\_COMM, called by MODBUSPN, uses the blocks TSEND and TRCV of the standard library for the data transfer between the PLC and the communication partner.

**MODBUSPN: SFCs for Miscellaneous Functions** The FB MODBUSPN uses the following SFCs from the standard library:

- FC10 "EQ\_STRING"
- SFC6 "RD\_SINFO"
- SFC20 "BLKMOV"
- SFC24 "TEST\_DB"
- SFC51 "RDSYST"
- SFC52 "WR\_USMSG"

**MOD\_CLI and  
MOD\_SERV:  
SFCs for  
Miscellaneous  
Functions**

The FBs MOD\_CLI and MOD\_SERV use the following SFCs from the standard library:

- SFC20 „BLKMOV“
- SFC24 “TEST\_DB”

**TCP\_COMM:  
SFCs for  
Miscellaneous  
Functions**

The FB TCP\_COMM uses the following SFB from the standard library in addition to the T-blocks:

- SFB4 „TON“

## 6.5 Renaming Standard Functions

**Inducement**

If the numbers of the standard functions are already assigned within your project or if the number interval is reserved for a different application, it is possible to rename the internally called function blocks FB63, FB64, FB65 and FB66 of FB TCP\_COMM or the called function blocks MOD\_CLI, MOD\_SERV und TCP\_COMM.

It is not possible to rename the system functions SFC6, SFC20, SFC24, SFC51, SFC52 and SFB4.

**Renaming**

To rename the blocks proceed as described subsequently:

1. Change the numbers of the function blocks in the desired numbers in your program in SIMATIC Manager.
2. Add the modifications in the symbolism table supplementary.

A modification of the FB numbers via “Extras -> Re-wire” is not necessary.

## 7 Diagnosis

<b>Diagnostic Function</b>	<p>The diagnostic functions of the PN PLC enable a fast localization of errors. The following diagnostic features are available:</p> <ul style="list-style-type: none"><li>• Diagnosis via the display elements of the PLC</li><li>• Diagnosis via the STATUS_MODBUS and STATUS_CONN output of the MODBUSPN function block.</li></ul>
<b>Display Elements (LED)</b>	<p>The display elements inform you about the operating mode or about the error conditions of the PLC. The display elements provide an overview of internal errors, external errors and interface-specific errors.</p>
<b>STATUS Outputs of the FB MODBUSPN</b>	<p>For an error diagnosis, the MODBUSPN function block has 3 STATUS outputs.</p> <p>When reading the STATUS_MODBUS output, you are provided with a general indication of errors that occurred during the Modbus-specific telegram processing.</p> <p>The STATUS_CONN output displays status information and error codes with regard to the processing of the T functions.</p> <p>The output STATUS_FUNC shows the name of the function, which caused the error at STATUS_MODBUS or STATUS_CONN.</p> <p>When reading the STATUS outputs, you are provided with a general indication of errors that occurred during the telegram processing and connection handling. The STATUS parameters can be evaluated in the user program.</p>

## 7.1 Diagnosis via the Display Elements of the PLC

- Display Functions**      The display elements of the PLC provide information on the status of the component. There are two types of display functions:
- **Group Error Displays**
    - PN CPU 300**
      - SF      Group errorWhen this LED is flashing, the Modbus block has not been licensed yet. You can find further information in **section 5 “Licensing”**.
    - PN-CPU 400**
      - INTF      Internal errorWhen this LED is flashing, the Modbus block has not been licensed yet. You can find further information in **section 5 “Licensing”**.
  - **Special Displays**
    - PN CPU 300, PN CPU 400 and IM 151-8 PN/DP CPU:**
      - RX/TX a telegram is transferred via the integrated interface

A detailed description of the display elements can be found in the device manual of the PLC.

## 7.2 Verification by the FB MODBUSPN

### During Start-up

- Unambiguousness of the *id* parameter in the connection parameters of DB MODBUS\_PARAM
- Parameter group *data\_type\_x*, *db\_x*, *start\_x*, *end\_x*. (x = 1 to 8)
  1. With *data\_type\_x* = 0 the Modbus area is disabled and not further verified. At least the first area (*\_1*) has to be parameterized.
  2. Validity check of *data\_type*
  3. *db\_x* <> 0
  4. Test *end\_x* >= *start\_x*
  5. Modbus addresses for the same data type defined in two *db\_x* lead to an error message (address overlap).

Errors during start-up provoke the ERROR bit to remain set. In the cyclical operation no requests are executed. A correction of the parameterization and a STOP → RUN transition of the PLC are necessary.

### Cyclical Operation S7 is Client

#### Verification when the FB is called:

- Valid registration key
- DB MODBUS\_PARAM is available with the necessary length
- Range of values monitoring time RECV\_TIME and CONN\_TIME
- Range of values START\_ADRESS
- Range of values LENGTH
- When executing a request, there also is a check whether the data block specified by the register or bit address is available and has the necessary length. The DB number must not be 0 and/or identical to the number of MODBUS\_PARAM or the instance DB.
- Receipt of the response telegram within the monitoring time

The monitoring time can also elapse if less data than specified in the MODBUS telegram header is received. Subsequent errors with loss of telegrams can occur.

**Verification in the response telegram:**

- Received transaction identifier is equal to the sent one.
- Protocol identifier = 0
- Length is between 3 and 253  
Additionally, the length in the header of the response telegram is checked for plausibility regarding the request.
- Sent UNIT is equal to the received one
- Sent FC is equal to the received one
- Response is an exception code telegram
- For write requests, the start address and number of registers/bits have to match with the request telegram.
- FC 5 or FC6:  
Echo in the respond is equal to the request
- Receipt of the second part of the request telegram within the monitoring time

The monitoring time can also elapse if less data than specified in the MODBUS telegram header is received. Subsequent errors with loss of telegrams can occur.

- Protocol Identifier = 0
- Length between 6 and 207
- Received function code is verified. If the function code is not equal to 1, 2, 3, 4, 5, 6, 15 or 16 an exception telegram is sent.
- For write requests, the length in the header, the number of registers or bits and the byte count in the telegram must match.
- The number of registers or bits is verified. If the number is too large, an exception telegram is sent.
- During the execution of a request, there is also a check whether the data block specified by the register or bit address is available and has the necessary length. The DB number must not be 0 or identical to the number of MODBUS\_PARAM or the instance DB. In case of an error, an exception telegram is sent.

**Cyclical Operation  
S7 is Server**

**Termination of the  
Connection in  
Case of Error**

In special error situations, the FB terminates the connection:

- Monitoring time for connection establishment exceeded
- Monitoring time for receive exceeded
- PI  $\neq$  0
- received TI different to sent TI
- Length in the header does not match the length information in the telegram

### 7.3 Diagnosis Messages of the FB MODBUSPN

**Messages at the STATUS Outputs of the FB**

The block MODBUSPN has 3 status outputs: STATUS\_MODBUS, STATUS\_CONN and STATUS\_FUNC. STATUS\_MODBUS displays the error numbers regarding the processing of Modbus telegrams, whereas STATUS\_CONN displays the error numbers regarding the connection handling. STATUS\_FUNC shows the name of the function, which caused the error. STATUS is valid when ERROR is TRUE. Below you can find a list of FB-specific error messages.

**Error Messages of the called SFCs and FBs**

The FBs MODBUSPN, MOD\_CLI and MOD\_SERV use the standard functions SFC6, SFC20, SFC24, SFC51 und SFC52. The error messages of these blocks are passed on to STATUS\_MODBUS without any changes.

The block TCP\_COMM, called by MOD\_CLI or MOD\_SERV, uses the standard blocks SFB4, FB63, FB64, FB65 and FB66. The error messages of these blocks are passed on to STATUS\_CONN without any changes.

For further details on the error messages, please consult the diagnosis buffer or the online help of the SIMATIC Manager.

Error messages of FB MODBUSPN at the output STATUS_MODBUS		
STATUS (Hex)	Event text	Remedy
A001	The parameter data block MODBUS_PARAM is too short.	Correct the length of MODBUS_PARAM
A002	The parameter <i>end_x</i> is less than <i>start_x</i> .	Correct the parameterization in DB MODBUS_PARAM
A003	The DB to which MODBUS addresses shall be mapped is too short. Minimum length: - register values $(end\_x - start\_x + 1) * 2$ - bit values $(end\_x - start\_x + 7) / 8$  Other possible reasons: <ul style="list-style-type: none"> <li>• Wrong initialization parameter (S7 is client)</li> <li>• Wrong address area in the request telegram of the client (S7 is server)</li> </ul>	Extend the DB. S7 is Client: Correct the parameters START_ADDRESS or LENGTH  S7 is Server: Modify the request of the client.
A004	Applies only if S7 is client: An invalid combination of DATA_TYPE and WRITE_READ is given.	Correct the parameters. Only data type 1 or 3 can be written.



Error messages of FB MODBUS MODBUSPN at the output STATUS_MODBUS		
STATUS (Hex)	Event text	Remedy
A005	S7 is client: An invalid value for the parameter LENGTH is given. S7 is Server: The number of registers/bits in the request telegram is invalid. Range of values: Read coils/inputs: 1 to 2000 Write coils: 1 to 800 Read registers: 1 to 125 Write holding registers: 1 to 100	S7 is Client: Correct the parameter LENGTH. S7 is Server: Modify the number in the request telegram.
A006	The given range of registers defined with DATA_TYPE, START_ADDRESS and LENGTH does not exist in <i>data_type_1</i> to <i>data_type_8</i> .	S7 is Client: Correct the parameter combination DATA_TYPE, START_ADDRESS, LENGTH. S7 is Server: Modify the request of the client or correct the parameterization of <i>data_type_x</i> .
A007	Only if S7 is client: An invalid monitoring time RECV_TIME or CONN_TIME is parameterized. The RECV_TIME has to be at least 20 ms, CONN_TIME 100ms	Correct the parameterization.
A009	Only if S7 is client: The received transaction identifier TI is not equal to the sent one. The connection is terminated.	Verify the data of the communication partner with the help of a telegram trace.
A00A	Only if S7 is client: The received UNIT is not equal to the sent one.	
A00B	S7 is client: Received function code is not equal to the sent one. S7 is server: An invalid function code was received.	S7 is client: Verify the data of the communication partner with the help of a telegram trace. S7 is server: The FB MODBUS only supports the function codes 1, 2, 3, 4, 5, 6, 15 and 16.
A00C	The received byte count does not match the number of registers. The connection is terminated.	Verify the data of the communication partner with the help of a telegram trace.
A00D	Only if S7 is client: The register or bit address respectively the number of registers or bits in the response telegram is not equal to the one in the request telegram.	
A00E	The length indicated in the MODBUS-specific telegram header does not match the number of registers/bits or the byte count in the telegram. The FB ignores all data. The connection is terminated.	

Error messages of FB MODBUSPN at the output STATUS_MODBUS		
STATUS (Hex)	Event text	Remedy
A00F	A protocol identifier $\neq$ 0 was received. The connection is terminated.	Verify the data of the communication partner with the help of a telegram trace.
A010	In the parameters <i>db_1</i> to <i>db_8</i> a DB number is used twice.	Correct the parameterization in DB MODBUS_PARAM.
A011	An invalid value for DATA_TYPE is given (Value range: 1 to 4).	Correct the parameters.
A012	The parameterized areas <i>data_type_1</i> and <i>data_type_2</i> overlap.	Correct the parameterization in DB MODBUS_PARAM.
A013	The parameterized areas <i>data_type_1</i> and <i>data_type_3</i> overlap.	The data areas must not contain any overlapping register areas.
A014	The parameterized areas <i>data_type_1</i> and <i>data_type_4</i> overlap.	
A015	The parameterized areas <i>data_type_1</i> and <i>data_type_5</i> overlap.	
A016	The parameterized areas <i>data_type_1</i> and <i>data_type_6</i> overlap.	
A017	The parameterized areas <i>data_type_1</i> and <i>data_type_7</i> overlap.	
A018	The parameterized areas <i>data_type_1</i> and <i>data_type_8</i> overlap.	
A019	0 is assigned to one of the parameters <i>db_x</i> while the according <i>data_type_x</i> is $\neq$ 0. DB 0 can't be used; it is reserved for system functions.	Correct the parameterization in DB MODBUS_PARAM.
A01A	Wrong length in the Modbus header (1 to 253 byte are valid). The connection is terminated.	Verify the data of the communication partner with the help of a telegram trace.
A01F	The FB MODBUSPN has turned to an invalid state.	Please contact the product support.
A023	The parameterized areas <i>data_type_2</i> and <i>data_type_3</i> overlap.	Correct the parameterization in DB MODBUS_PARAM
A024	The parameterized areas <i>data_type_2</i> and <i>data_type_4</i> overlap.	The data areas must not contain any overlapping register areas.
A025	The parameterized areas <i>data_type_2</i> and <i>data_type_5</i> overlap.	
A026	The parameterized areas <i>data_type_2</i> and <i>data_type_5</i> overlap.	
A027	The parameterized areas <i>data_type_2</i> and <i>data_type_5</i> overlap.	
A028	The parameterized areas <i>data_type_2</i> and <i>data_type_5</i> overlap.	
A034	The parameterized areas <i>data_type_3</i> and <i>data_type_4</i> overlap.	

Error messages of FB MODBUSPN at the output STATUS_MODBUS		
STATUS (Hex)	Event text	Remedy
A035	The parameterized areas <i>data_type_3</i> and <i>data_type_5</i> overlap.	Correct the parameterization in DB MODBUS_PARAM  The data areas must not contain any overlapping register areas.
A036	The parameterized areas <i>data_type_3</i> and <i>data_type_6</i> overlap.	
A037	The parameterized areas <i>data_type_3</i> and <i>data_type_7</i> overlap.	
A038	The parameterized areas <i>data_type_3</i> and <i>data_type_8</i> overlap.	
A045	The parameterized areas <i>data_type_4</i> and <i>data_type_5</i> overlap.	
A046	The parameterized areas <i>data_type_4</i> and <i>data_type_6</i> overlap.	
A047	The parameterized areas <i>data_type_4</i> and <i>data_type_7</i> overlap.	
A048	The parameterized areas <i>data_type_4</i> and <i>data_type_8</i> overlap.	
A056	The parameterized areas <i>data_type_5</i> and <i>data_type_6</i> overlap.	
A057	The parameterized areas <i>data_type_5</i> and <i>data_type_7</i> overlap.	
A058	The parameterized areas <i>data_type_5</i> and <i>data_type_8</i> overlap.	
A067	The parameterized areas <i>data_type_6</i> and <i>data_type_7</i> overlap.	
A068	The parameterized areas <i>data_type_6</i> and <i>data_type_8</i> overlap.	
A078	The parameterized areas <i>data_type_7</i> and <i>data_type_8</i> overlap.	
A079	The connection ID of parameter ID is not defined in DB MODBUS_PARAM.	Correct the parameterization at the input ID.
A07A	An invalid value ID is parameterized. Range of values is 1 to 4095.	
A07B	The parameter ID exists twice in the parameter DB.	Correct the parameterization in DB MODBUS_PARAM.
A07C	An invalid value <i>data_type_x</i> was given. The value range is 0 to 4.	
A07D	Parameter <i>data_type_1</i> is not defined. The parameter area <i>_1</i> is the default area and must be defined.	
A07E	The DB number of <i>db_x</i> is identical to the number of the parameter DB MODBUS_PARAM or to the instance DB.	
A07F	The data block defined at DB_PARAM is no parameter DB for Modbus communication. The length information in DBW0 was changed or a wrong DB was defined.	
A080	The parameter DB_PARAM was changed without a restart of the CPU.	DB_PARAM is an initialization parameter. Restart the CPU after changing this parameter.

<b>Error messages of FB MODBUSPN at the output STATUS_MODBUS</b>		
<b>STATUS (Hex)</b>	<b>Event text</b>	<b>Remedy</b>
A081	Only if S7 is client and function code 5: The received coil status is not equal to the sent one.	Verify the data of the communication partner with the help of a telegram trace.
A082	Only if S7 is client and function code 6: The received register value is not equal to the sent one.	Verify the data of the communication partner with the help of a telegram trace.
A083	Only if S7 is client: A request was initiated prior to the completion of the previous one.	Wait with the initiation of a new request until the previous one was finished either with DONE/NDR = TRUE or ERROR = TRUE.
A084	The character string for identification "IDENT_CODE" could not be determined.	Please contact the Product Support.
A090	The block MODBUSPN is not licensed for this CPU.	Read the identification string IDENT_CODE for this CPU and order the registration key at IT4industry. See also section 5 "Licensing".
A091	An exception telegram with exception code 1 was received (only if CP is client)	The communication partner does not support the requested function.
A092	An exception telegram with exception code 2 was received (only if CP is client) An attempt to an invalid or non existing address at the communication partner was made.	Correct LENGTH or START_ADDRESS at the call of the FB.
A093	An exception telegram with exception code 3 was received (only if CP is client)	Check the error message of the communication partner.
A094	An exception telegram with exception code 4 was received (only if CP is client)	Check the error message of the communication partner.
A095	An exception telegram with an unknown exception code was received (only if CP is client).	Check the error message of the communication partner and verify the data with a telegram trace if needed.

Error messages of FB MODBUSPN at the output STATUS_CONN		
STATUS (Hex)	Event text	Remedy
A100	The monitoring time CONN_TIME or RECV_TIME was exceeded when executing a job. When RECV_TIME is exceeded, the connection is terminated.	Check the parameterization of the connection.
A101	The monitoring time of the TDISCON is exceeded.	Please contact the product support.

## 7.4 Diagnosis Messages of Called Blocks

Error messages of SFC 6 and SFC 20 at the output STATUS_MODBUS		
STATUS (Hex)	Event text	Remedy
7xxx	For detailed information please refer to the online help of SIMATIC Manager.	See online help (SIMATIC manager -> mark block -> key F1)
8xxx	For detailed information please refer to the online help of SIMATIC Manager.	See online help (SIMATIC manager -> mark block -> key F1)

Error messages of FB 63, FB 64, FB 65 and FB 66 at the output STATUS_CONN		
STATUS (Hex)	Event text	Remedy
7xxx	For detailed information please refer to the online help of SIMATIC Manager.	See online help (SIMATIC manager -> mark block -> key F1)
8xxx	For detailed information please refer to the online help of SIMATIC Manager.	See online help (SIMATIC manager -> mark block -> key F1)

## 7.5 Diagnosis Messages of SFC24

Error messages of SFC24 at the output STATUS_MODBUS		
STATUS (Hex)	Event text	Remedy
80A1	DB Number = 0 or too large for the PLC This error code is also reported, when FB MODBUSPN is called with different instance DBs in OB1 or cyclic interrupt OB and OB100.	Choose a valid DB number. Use the same IDB for the call of MODBUSPN in OB1 or cyclic interrupt OB and OB100.
80B1	The DB does not exist in the PLC.	All data blocks that are specified in DB_x must be created and copied into the PLC.
80B2	DB UNLINKED	DB must not be created as UNLINKED.

## 8 Sample Application

### General Information

The following simple programming example illustrates the use of FB MODBUSPN.

**Please note, the provided example project is meant for information purposes only. It displays the handling of the Modbus blocks and is not to be understood as a solution for a customer-specific installation configuration.**

### Example Project on the CD

On the CD you can find an extensive example project which offers all varieties of parameterization possibilities for the Simatic stations.

- Simatic Station is S7-300, S7-400 or IM 151-8 PN/DP CPU
- Simatic Station is client or server

### Sample Program

The programming example consists of the blocks:

- Start-Up OB100 with call of FB102
- Programming error OB121 with call of FB102
- Cyclic program processing OB1 with call of FB102
- Global DBs for job trigger (e.g. with variable table) and for licensing
- Data blocks for register and bit values

### Used Blocks

The following blocks are used in the provided sample project for S7 stations with FB MODBUSPN.

Block	Symbol	Comment
OB 1	CYCL_EXC	cyclic program processing
OB 100	COMPLETE RESTART	start-up OB for restart
OB 121	PROG_ERR	programming error OB
FB 102	MODBUSPN	FB MODBUSPN
FB 103	TCP_COMM	FB TCP_COMM
FB 104	MOD_CLI	FB MOD_CLI
FB 105	MOD_SERV	FB MOD_SERV
DB 1	CONTROL_DAT	work DB CONTROL DAT for FB MODBUSPN
DB 2	MODBUS_PARAM	parameter DB PARAM_DB for FB MODBUSPN
DB 3	LICENSE_DB	license data block for FB MODBUSPN
DB 102	IDB_MODBUS	instance DB for FB MODBUSPN
DB 11	DATA_AREA_1	value DB for area 1
DB 12	DATA_AREA_2	value DB for area 2
DB 13	DATA_AREA_3	value DB for area 3
DB 14	DATA_AREA_4	value DB for area 4
DB 15	DATA_AREA_5	value DB for area 5

# A Literature

**MODBUS IDA**

MODBUS APPLICATION PROTOCOL SPECIFICATION  
V1.1b, December 28, 2006

<http://www.modbus-IDA.org>



# Glossary

## A

**Address** The address identifies a physical storage location. If the address is known, the operand stored there can be directly accessed.

**Automation System** An automation system is a programmable logic controller that contains at least a PLC, different input and output devices as well as HMI devices.

## B

**Baud Rate** - > transmission rate

**Block Call** A block call occurs when program processing branches to the called block

**Block Parameter** Block parameters are variables within multiple-use blocks, which are replaced with actual values when the relevant block is called.

**Blocks** Blocks are elements of the user program which are defined by their function, structure, or purpose. With STEP7 there are

- Code blocks (FB, FC, OB, SFB, SFC)
- Data blocks (DB, SDB)
- User-defined data types (UDT)

**Bus Segment** Part of a -> subnet. Subnets can consist of bus segments and connectivity devices such as repeaters and bridges. Segments are transparent for addressing.

## C

**Client** A client is a device or, in general terms, an object that requests a service from a -> server.

**Communications Processor** Communications processors are modules for point-to-point connections and bus connections.

**Configuration** The configuration is the set up of individual modules of the PLC in the configuration table.

**Connection Parameterization** The specification of a connection ID in the system function block. With the help of a connection ID the system function blocks can communicate between two communication points.

**CP** Communications Processor. Module for communications tasks.

CPU	Central processing unit of the S7 programmable logic controller with control and arithmetic unit, memory, operating system, and interfaces to I/O modules.
CRC	Cyclic Redundancy-Check = Checksum which guarantees a high probability of error recognition.
Cycle Time	The cycle time is the time the PLC needs to execute the user program once.
Cyclic Program Processing	In cyclic program processing, the user program is executed in a constantly repeating program loop, called a cycle.

## D

Data Block (DB)	These are blocks containing data and parameters with which the user program works. Unlike all other blocks, data blocks do not contain instructions. They are subdivided into global data blocks and instance data blocks. The data held in the data blocks can be accessed absolutely or symbolically. Complex data can be stored in structured form.
Data Type	Data types allow users to define how the value of a variable or constant is to be used in the user program. They are classified into elementary and structured data types.
Default Setting	The default setting is a basic setting which is always used if no other value is specified.
Diagnostic Buffer	Every PLC has a diagnostic buffer, in which detailed information on diagnostic events are stored in the order in which they occur.
Diagnostic Event	Diagnostic events are, for example, errors on a module or system errors in the PLC, which are caused by, say, a program error or by operating mode transitions.
Diagnostic Functions	The diagnostics functions cover the entire system diagnosis and include detection, analysis and reporting of errors within the automation system.
Download	Downloading means loading objects (e.g. code blocks) from the programming device into the load memory of the PLC.

## F

Function Block (FB)	Function blocks are components of the user program and, in accordance with the IEC standard, are "blocks with memory". The memory for the function block is an assigned data block, a so called "instance data block". Function blocks can be parameterized but can also be used without parameters.
---------------------	--

## H

Hardware	Hardware is the term given to all the physical and technical equipment of a PLC.
----------	--

## I

Industrial Ethernet	A LAN system complying with IEEE 802.3 (ISO 8802-2)
Instance Data Block	An instance data block is a block assigned to a function block and contains data for this special function block.
Interface Module	On the interface module the physical conversion of signals takes place. By exchanging the pluggable interface module you can adapt the communications processor to the physical interface of the communications partner.
Interrupt	Interrupt is a name for a break of the program processing in the processor of an automation system by an external alarm.

## M

MAC-Address	Address to distinguish between different stations connected to a common transmission medium (Industrial Ethernet).
Media Access Control (MAC)	Mechanisms for controlling access by a station to a common transmission medium shared with other stations.
Module	Modules are pluggable printed circuit boards for programmable logic controllers
Module Parameters	Module parameters are used to set the module behaviors. A distinction is made between static and dynamic module parameters.

## N

NCM S7 for Industrial Ethernet	Configuration software for configuration and diagnostic functions on an Ethernet CP.
--------------------------------	--

## O

Online / Offline	Online means that a data connection exists between PLC and programming device. Offline means that no such data connection exists.
Online Help	STEP7 allows you to display contextual help texts on the screen while working with the programming software.
Operand	An operand is part of a STEP7 instruction and states with what the processor is to do something. It can be both absolutely and symbolically addressed.
Operating Mode	The SIMATIC S7 programmable controllers have three different operating modes: STOP, START UP and RUN. The functionality of the PLCs varies in the individual operating modes.
Operating System of the PLC	The operating system of the PLC organizes all functions and operations of the PLC which are not connected to a specific control task.

## P

Parameter	Parameters are values that can be assigned. A distinction is made between block parameters and module parameters.
Parameterization	Parameterization means setting the behavior of a module.
Procedure	The execution of a data interchange operation according to a specific protocol is called a procedure.
Process image	This is a special memory area in the PLC. At the beginning of the cyclic program, the signal states of the input modules are transferred to the process image input table. At the end of the cyclic program, the process image of the outputs is transferred to the output modules as output signals.
Protocol	The communications partners involved in a data interchange must abide by fixed rules for handling and implementing the data traffic. These rules are called protocols.

## R

Rack	A rack is a rail containing slots for mounting modules.
------	---

## S

Server	A server is a device, or in general terms, an object that provides certain services. A service is started at the instigation of a -> client.
Software	Software is the term given to all programs used on a computer system. These include the operating system and the user programs.
START UP	The operating mode START UP is active when the PLC transits from operating mode STOP to operating mode RUN.
STEP7	STEP7 is the programming software of SIMATIC S7.
Subnet	A subnet is part of a -> network whose parameters must be matched. The subnet includes bus components and all the attached stations. Subnets can, for example, be connected together by -> gateways to form a network. A system consists of several subnets with unique subnet numbers. A subnet consists of several stations with unique -> MAC addresses.
System Block	System blocks differ from the other blocks in that they are already integrated into the S7-400 system and are available for already defined system functions. They are classified into system data blocks, system functions, and system function blocks.
System Function (SFC)	System functions are software modules <i>without</i> memory which are already integrated into the operating system of the S7-PLC and can be called by the user as required.
System Function Block (SFB)	System function blocks are software modules <i>with</i> memory which are already integrated into the operating system of the S7-PLC and can be called up by the user as required.

## **T**

**Tool** A tool is a piece of software that is capable of accessing operating system functions in a programming device.

**Transmission Rate** According to DIN 44302, this is the number of binary decisions transmitted per time unit. The unit is bps. The set or selected transmission rate depends on various conditions, for example the distance across

## **U**

**Upload** Uploading means loading objects (e.g. code blocks) from the load memory of the PLC into the programming device.

**User Program** The user program contains all instructions and declarations for signal processing, by means of which a system or a process can be controlled. The user program for SIMATIC S7 is structured and is divided into smaller units called blocks.

## **V**

**Variable** A variable is an operand (e.g. E 1.0) which can have a symbolic name and can therefore also be addressed symbolically.

## **W**

**Work Memory** The work memory is a RAM on the PLC which the processor accesses while processing the user program.

---

## **Customer Support**

Siemens AG  
Industry Sector  
I IS IN E&C  
Werner-von-Siemens-Str. 60  
91052 Erlangen  
Tel: ++49 9131 7-46111  
Fax: ++49 9131 7-44757  
Mail: it4.industry@siemens.com

<http://www.siemens.com/s7modbus>

**Siemens Aktiengesellschaft**

Subject to change without prior notice.

Stand: 06/2009