SIEMENS

Modbus/TCP with WinAC RTX (F) in TIA Portal V13 SP1

WinAC RTX, Modbus TCP

https://support.industry.siemens.com/cs/ww/en/view/109482560

This entry is from the Siemens Industry Online Support. The general terms of use (http://www.siemens.com/terms_of_use) apply.

Table of content

1 Introduction

1.1 Overview

These FAQs will provide you with a step-by-step instruction on how to configure the SIMATIC Modbus/TCP software under a WinAC RTX (F) Software PLC with TIA Portal V13 SP1.

Figure 1-1
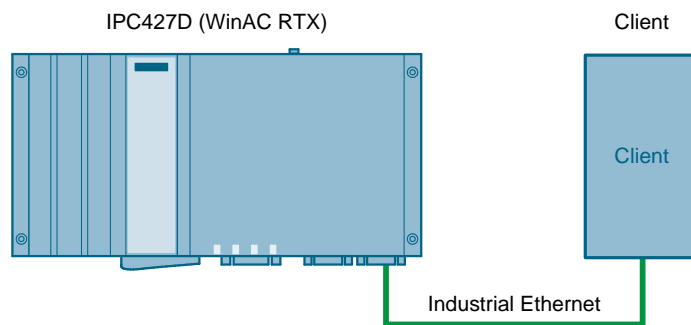
Note	This description only refers to the configuration of the SIMATIC Modbus/TCP software under WinAC RTX (F) in the TIA Portal V13 SP1. For detailed information on Modbus/TCP, WinAC RTX (F) or the TIA Portal, refer to the corresponding documentation.

SIMATIC WinAC RTX (F)

SIMATIC WinAC RTX (F) is the SIMATIC software controller for PC-based automation solutions and allows real-time deterministic control on the PC.

SIEMENS Modbus/TCP software

MODBUS is a worldwide communication protocol which is open to all users and is used by many manufacturers. On this basis, MODBUS/TCP has been developed for the use in modern networks. In today's industry, this protocol has become a de facto standard which has been introduced in the Internet Engineering Task Force (IETF) – an organization responsible for Internet standardization.

The SIMATIC Modbus/TCP software packages offer ready-to-use blocks for simple and quick implementation of a MODBUS communication in SIMATIC controllers.

1.2 Hardware and software components

1.2.1 Validity

This application is valid for STEP 7 as of V13 SP1.

1.2.2 Components used

The application has been created with the following components:

Hardware components

Table 1-1

Component	Qty	Article number	Note
Power supply	1	6EP1332-4BA00	PM 1507 70 W
SIMATIC Micro Box PC	1	6AG4140-8BL04-0GA0	IPC427D PN

Software components

All components for this Application Example are already included in STEP 7 V13 SP1. Licensing requires the "SIMATIC MODBUS/TCP PN-CPU" software package.

Table 1-2

Component	Qty	Article number	Note
STEP 7 Professional	1	6ES7822-1AA03-0YA5	V13 SP1
WinAC RTX Software PLC	1	6ES7611-4SB00-0YB7	
SIMATIC MODBUS/TCP PN-CPU	1	2XV9450-1MB02	The software package is already included in STEP 7 V13 and can be used without license for test purposes.

Example files and projects

The following list includes all files and projects that are used in this example.
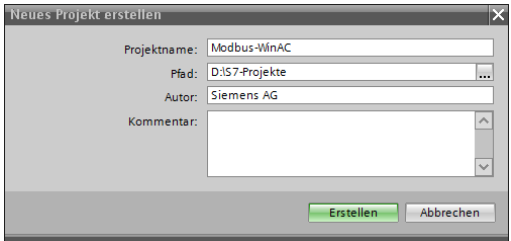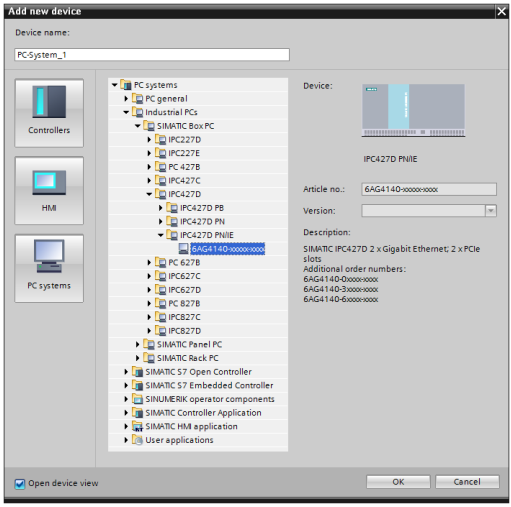
Table 1-3

Component	Note
109482560_WinAC_RTX_Modbus_DOC_V10_de.pdf	This document

2 Configuring Modbus/TCP under WinAC RTX (F)

2.1 Creating IPC427D in a new project

In this Application Example, a SIMATIC Micro Box PC IPC427D is used as PC. First, this hardware has to be added and configured.
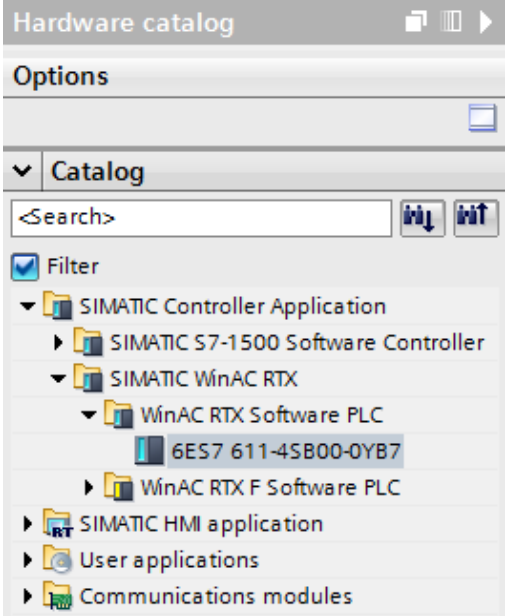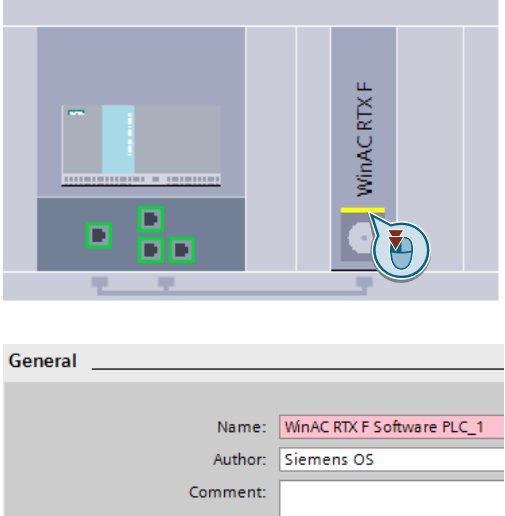
Table 2-1

No.	Action	Remark
1.	Open the TIA Portal V13 SP1.	
2.	Go to the project view.	
3.	Click "Project > New" and enter a project name and path. Then click "Create".	
4.	Click "Add new device" in the project navigation. Under "PC systems", select "Industrial PCs > SIMATIC Box PC > IPC427D > IPC427D PN > 6AG4140-xxxx-xxxx", and click "OK".	
5.	If the "Open device view" option is set, the device configuration will open after insertion.	

2.2 Creating WinAC RTX (F) in the IPC427D
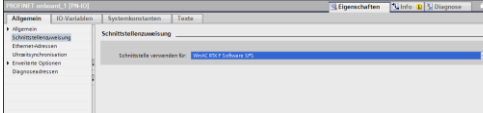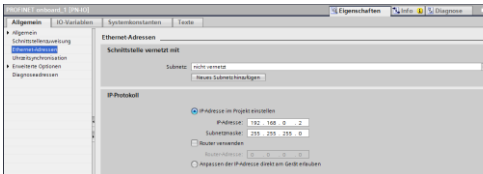
In this Application Example, a "WinAC RTX Software PLC" is used as PLC. It is created as software controller in the IPC427D.
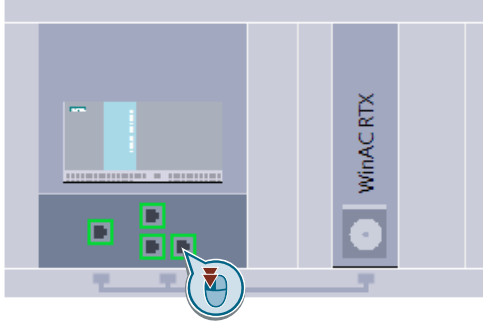
Table 2-2

No.	Action	Remark
1.	In the hardware catalog on the right, select "SIMATIC Controller Application > SIMATIC WinAC RTX > WinAC RTX Software PLC > 6ES7 611-4FB00-0YB7" and drag the component to index 2 of the device "PC-System_1".	
2.	If you are using the software controller WinAC RTX F, double-click "WinAC RTX F" in the workspace and change the name under "General", as the default name is too long.	

2.2 Creating WinAC RTX (F) in the IPC427D

No.	Action	Remark
3.	In the workspace, double-click on a port of the right PROFINET interface.	
4.	Then, select the software PLC under "Interface assignment".	
5.	Under "Ethernet addresses", assign the IP address of the WinAC RTX Software PLC.	
6.	Save the project. Thus, the hardware configuration is completed.	

2.3 Data block for the Modbus/TCP interface

To control the Modbus/TCP communication and to evaluate status information, a data block with the necessary interface tags is created.
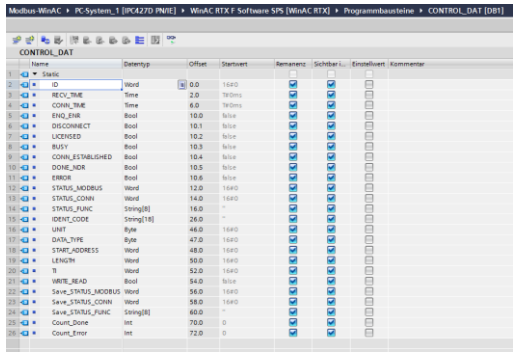
Table 2-3

No.	Action	Remark
1.	In the project navigation, open the folders "PC-System_1 > WinAC RTX PLC_1 > Program blocks". Add a new block by double-clicking on "Add new block".	
2.	Create a global data block with the name "CONTROL_DAT".	
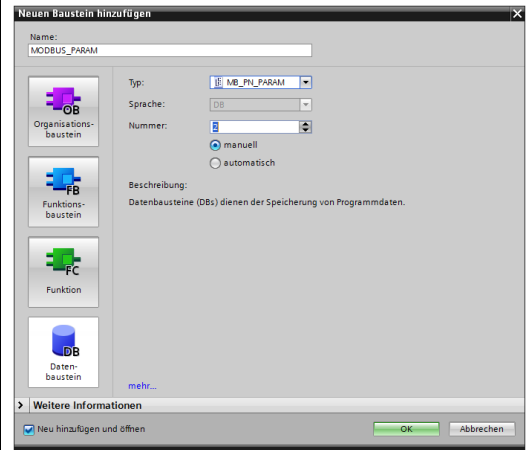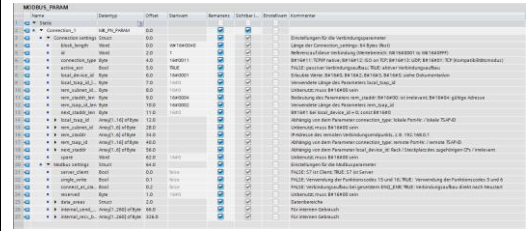
2.3 Data block for the Modbus/TCP interface

No.	Action	Remark
3.	Open the data block and create the tags as specified in the Appendix in chapter 5.1. You can add the names and data types in TIA Portal via copy & paste.	
4.	Assign start values to the tags as required.	For example: • ID = 1 • RECV_TIME = T#100ms • CONN_TIME = T#200ms • ENQ_ENR = TRUE
5.	Save the project and close the block.	

2.4 Data block for the Modbus/TCP parameters

The Modbus/TCP communication requires a parameter data block containing the communication and Modbus settings. The required structure is available as "PLC data type" and is used for DB creation.

Table 2-4
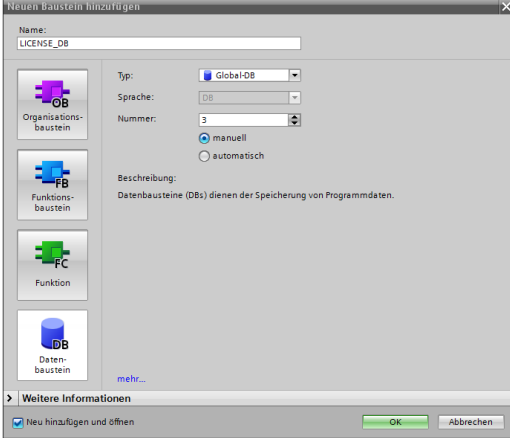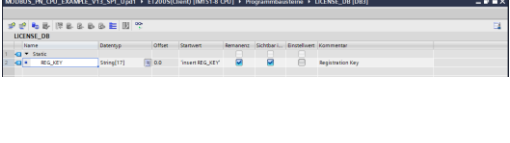
No.	Action	Remark
1.	Add a new block by double-clicking on "Add new block".	
2.	Create a data block with the name "MODBUS_PARAM" and select "MB_PN_PARAM" as type.	
3.	Open the block and parameterize the connection and Modbus settings. A precise description of the parameters is available in the online help or in the "MODBUS-PN-CPU" documentation.	

Note	The data block of type "MB_PN_PARAM" is write-protected. No other parameters can be added. However, editing of the available parameters is possible.

2.5   Data block for licensing

The "SIMATIC MODBUS PN-CPU" requires a valid license. For licensing, a "REG_KEY" registration key is required. This one is entered in the "License_DB" data block in order to make it available for all Modbus blocks.

Table 2-5

No.	Action	Remark
4.	Add a new block by double-clicking on "Add new block".	
5.	Create a global data block with the name "LICENSE_DB".	
6.	Open the block and create a tag with the name "REG_KEY" of the type "String[17]". If you have already done the licensing, enter the license key under "Start value".	
7.	Save the project and close the block.	

2.6 Data block for the Modbus communication data

The data that shall be sent/received via the Modbus/TCP communication will be stored in data blocks. In this Application Example, a data block is created in order to store 500 holding registers.
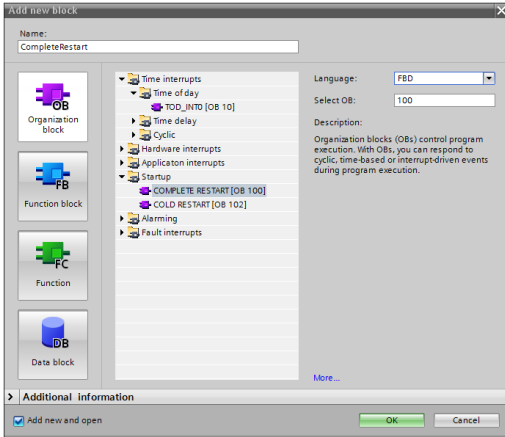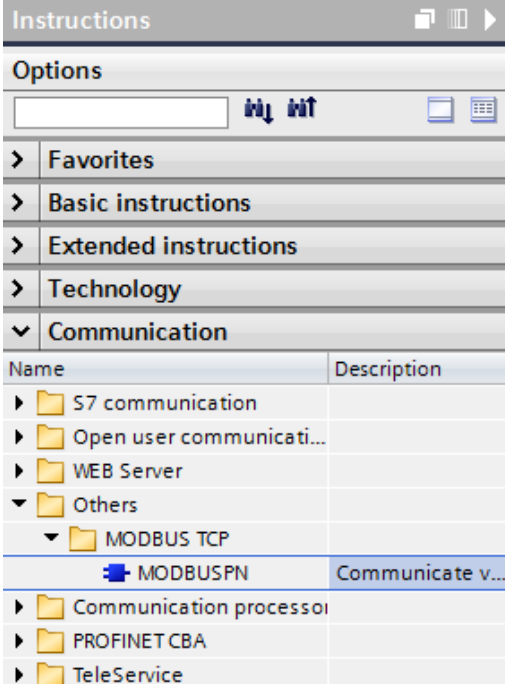
Table 2-6

No.	Action	Remark
8.	Add a new block by double-clicking on "Add new block".	
9.	Create a global data block with the name "DATA_AREA_1".	
10.	Open the block and create, for example, an array named "Holding_Register" of the type "Word" with 500 elements.	
11.	Create an additional tag of the data type "Word" with the name "reserved". This is required for internal purposes.	
12.	Save the project and close the block.	

2.7 Organization block for Modbus/TCP initialization
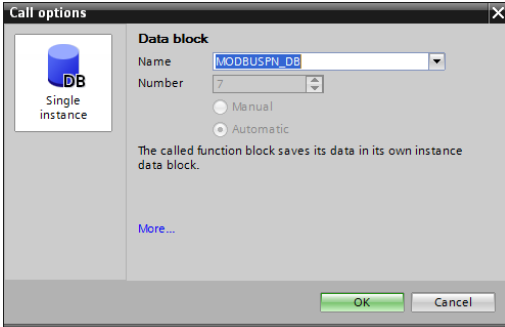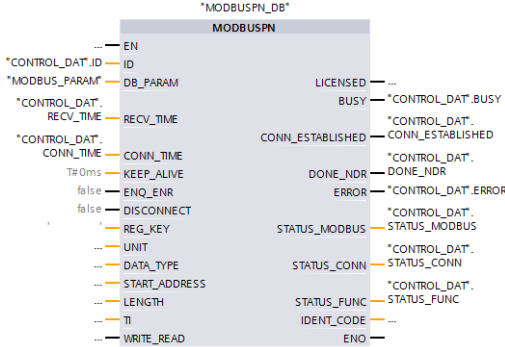
At the beginning, the Modbus block "MODBUSPN" has to be initialized. This is done with a warm start of the PLC in OB100.

Table 2-7

No.	Action	Remark
1.	Add a new block by double-clicking on "Add new block".	
2.	Select "Organization block > Startup > COMPLETE RESTART [OB100]" and set the programming language FBD. Then, click "OK".	
3.	Open the block. (If the option "Add new and open" is selected, the block will be opened automatically. Open the "Instructions" tab shown on the right and navigate to "Communication > Others > MODBUS TCP". Drag the "MODBUSPN" instruction into the empty network using drag & drop.	
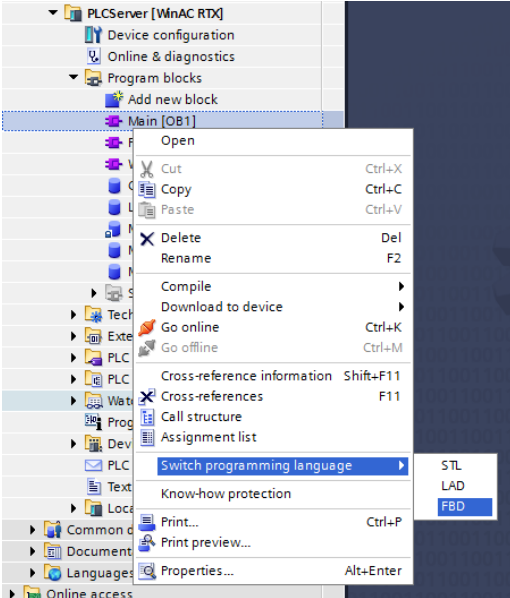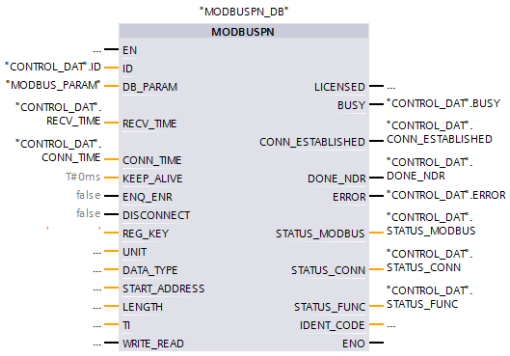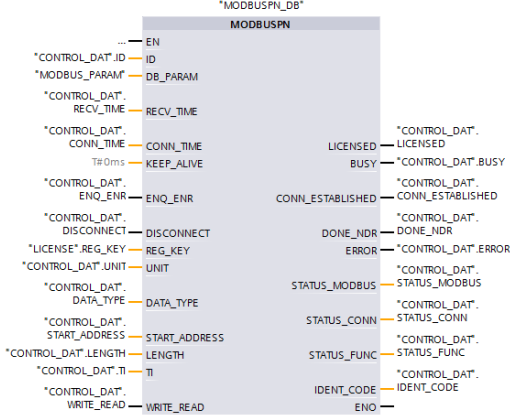
2.7 Organization block for Modbus/TCP initialization

No.	Action	Remark
4.	In the open dialog, create an instance data block with the name "MODBUSPN_DB".	
5.	Assign tags from the previously created data blocks to the following parameters of the function block: • ID • DB_PARAM • RECV_TIME • CONN_TIME • BUSY • CONN_ESTABLISHED • DONE_NDR • ERROR • STATUS_MODBUS • STATUS_CONN • STATUS_FUNC	
6.	Copy the network in which the function block is called.	
7.	Save the project and close the block.	All system blocks required for the Modbus/TCP communication are added automatically under "System blocks > Program resources".

2.8 Organization block for Modbus/TCP communication

For communication via Modbus/TCP, the "MODBUSPN" block has to be called in a cyclic block. In this Application Example, "OB1" is used as an example.

Table 2-8

No.	Action	Remark
1.	Switch the programming language of the "Main [OB1]" block to FBD. For this, right-click on the block and select "Switch programming language > FBD".	
2.	Open the "Main [OB1]" block.	
3.	Right-click on a free network and add the previously copied network.	
4.	Assign tags from the previously created data blocks to the remaining parameters of the function block:	
5.	Save the project and close the block.	

NOTE	The initialization of the instance of the Modbus block takes place in OB100, which is used later for the cyclic call in OB1. For this reason, in both OBs, the same instance data block (InstModbusPN) has to be used for the "MODBUSPN" instruction.
	However, for each communication partner, an own instance of the Modbus block is created which is used both in OB1 and OB100.

2.9 Organisation block "PROG_ERR [OB121]"

As long as the MODBUSPN block is not licensed, the organization block "PROG_ERR [OB121]" has to be added to the project in order to avoid a STOP status of the PLC.

Table 2-9

No.	Action	Remark
6.	Add a new block by double-clicking on "Add new block".	
7.	Select "Organization block > Fault interrupts > PROG_ERR [OB121]". Then, click "OK".	Modbus/TCP communication does not require any program in "PROG_ERR [OB121]". Save the project and close the block, if required.

2.10 Completion

Thus, the setup of a Modbus/TCP communication on a WinAC RTX (F) on an IPC427D under TIA Portal V13 SP1 is completed.

You can now compile the project and load it to the CPU.

Note	The OB100 is only called in case of a warm start of the PLC. Thus, in case of a cold start, the function block MODBUSPN will not be initialized and the A080 error message will be output.
	Make sure that a warm start is carried out after having modified any parameters or additionally add the OB102 with the same network as in OB100.
	A cold start is carried out by means of the button for starting a PLC in the toolbar of the TIA Portal.

3 Related Literature

Table 3-1

	Topic
\1\	Siemens Industry Online Support http://support.industry.siemens.com
\2\	Download page of the entry https://support.industry.siemens.com/cs/ww/en/view/109482560
\3\	Siemens industrial PCs http://siemens.com/ipc
\4\	Siemens Software Controller WinAC http://siemens.com/winac
\5\	Siemens SIMATIC Modbus/TCP software http://www.siemens.com/s7modbus
\6\	Modbus/TCP example projects (TIA Portal) https://support.industry.siemens.com/cs/ww/en/view/75312612
\7\	The Modbus Organization http://modbus.org

4 History

Table 4-1

Version	Date	Modifications
V1.0	06/2016	First version

5 Appendix

5.1 CONTROL_DAT

For the control and diagnostics of the Modbus communication, you will find the parameters of the data block "CONTROL_DAT" in the following.

Table 5-1

Name	Data type	Offset	Start value
Static			
ID	Word	0.0	16#0
RECV_TIME	Time	2.0	T#0ms
CONN_TIME	Time	6.0	T#0ms
ENQ_ENR	Bool	10.0	false
DISCONNECT	Bool	10.1	false
LICENSED	Bool	10.2	false
BUSY	Bool	10.3	false
CONN_ESTABLISHED	Bool	10.4	false
DONE_NDR	Bool	10.5	false
ERROR	Bool	10.6	false
STATUS_MODBUS	Word	12.0	16#0
STATUS_CONN	Word	14.0	16#0
STATUS_FUNC	String[8]	16.0	''
IDENT_CODE	String[18]	26.0	''
UNIT	Byte	46.0	16#0
DATA_TYPE	Byte	47.0	16#0
START_ADDRESS	Word	48.0	16#0
LENGTH	Word	50.0	16#0
TI	Word	52.0	16#0
WRITE_READ	Bool	54.0	false
Save_STATUS_MODBUS	Word	56.0	16#0
Save_STATUS_CONN	Word	58.0	16#0
Save_STATUS_FUNC	String[8]	60.0	''

5.2 MODBUS_PARAM

For Modbus communication, you will find an example of a data block "MODBUS_PARAM" for the connection and Modbus configuration in the following.

A precise description of the parameters is available in the online help or in the "MODBUS/TCP PN CPU" documentation.

Table 5-2

Name	Data type	Offset	Start value	Comment
Connection_1	MB_PN_PARAM	0.0		
Connection settings	Struct	0.0		Settings for the connection parameters
block_length	Word	0.0	W#16#0040	Length of the Connection_settings: 64 bytes (fixed)
id	Word	2.0	1	Reference to this connection (range of values: W#16#0001 to W#16#0FFF)
connection_Type	Byte	4.0	16#0011	B#16#11: TCP/IP native; B#16#12: ISO on TCP; B#16#13: UDP; B#16#01: TCP (compatibility mode)
active_est	Bool	5.0	TRUE	FALSE: passive connection establishment; TRUE: active connection establishment
local_device_id	Byte	6.0	16#0001	Allowed values: B#16#0, B#16#2, B#16#3, B#16#5; see documentation
local_tsap_id_len	Byte	7.0	16#0	Used length of the parameter local_tsap_id
rem_subnet_id_len	Byte	8.0	16#0	Idle; must be B#16#00
rem_staddr_len	Byte	9.0	16#0004	Meaning of the parameter rem_staddr: B#16#00: is irrelevant; B#16#04: valid address
rem_tsap_id_len	Byte	10.0	16#0002	Used length of the parameter rem_tsap_id
next_staddr_len	Byte	11.0	16#0	B#16#1 for local_device_id = 0; otherwise B#16#0
local_tsap_id	Array[1..16] of Byte	12.0		Depending on the parameter connection_type: local port no. / local TSAP-ID
local_tsap_id[1]	Byte	0.0	16#0	
local_tsap_id[2]	Byte	1.0	16#0	
local_tsap_id[3]	Byte	2.0	16#0	
local_tsap_id[4]	Byte	3.0	16#0	

5.2 MODBUS_PARAM

Name	Data type	Offset	Start value	Comment
local_tsap_id[5]	Byte	4.0	16#0	
local_tsap_id[6]	Byte	5.0	16#0	
local_tsap_id[7]	Byte	6.0	16#0	
local_tsap_id[8]	Byte	7.0	16#0	
local_tsap_id[9]	Byte	8.0	16#0	
local_tsap_id[10]	Byte	9.0	16#0	
local_tsap_id[11]	Byte	10.0	16#0	
local_tsap_id[12]	Byte	11.0	16#0	
local_tsap_id[13]	Byte	12.0	16#0	
local_tsap_id[14]	Byte	13.0	16#0	
local_tsap_id[15]	Byte	14.0	16#0	
local_tsap_id[16]	Byte	15.0	16#0	
rem_subnet_id	Array[1..6] of Byte	28.0		Idle; must be B#16#00
rem_subnet_id[1]	Byte	0.0	16#0	
rem_subnet_id[2]	Byte	1.0	16#0	
rem_subnet_id[3]	Byte	2.0	16#0	
rem_subnet_id[4]	Byte	3.0	16#0	
rem_subnet_id[5]	Byte	4.0	16#0	
rem_subnet_id[6]	Byte	5.0	16#0	
rem_staddr	Array[1..6] of Byte	34.0		IP address of the remote connection endpoint, e. g. 192.168.0.1
rem_staddr[1]	Byte	0.0	16#000A	
rem_staddr[2]	Byte	1.0	16#0	
rem_staddr[3]	Byte	2.0	16#0	
rem_staddr[4]	Byte	3.0	16#0006	
rem_staddr[5]	Byte	4.0	16#0	
rem_staddr[6]	Byte	5.0	16#0	
rem_tsap_id	Array[1..16] of Byte	40.0		Depending on the parameter connection_type: remote port no. / remote TSAP-ID
rem_tsap_id[1]	Byte	0.0	16#0001	
rem_tsap_id[2]	Byte	1.0	16#00F6	
rem_tsap_id[3]	Byte	2.0	16#0	
rem_tsap_id[4]	Byte	3.0	16#0	
rem_tsap_id[5]	Byte	4.0	16#0	
rem_tsap_id[6]	Byte	5.0	16#0	
rem_tsap_id[7]	Byte	6.0	16#0	
rem_tsap_id[8]	Byte	7.0	16#0	
rem_tsap_id[9]	Byte	8.0	16#0	
rem_tsap_id[10]	Byte	9.0	16#0	
rem_tsap_id[11]	Byte	10.0	16#0	
rem_tsap_id[12]	Byte	11.0	16#0	
rem_tsap_id[13]	Byte	12.0	16#0	
rem_tsap_id[14]	Byte	13.0	16#0	
rem_tsap_id[15]	Byte	14.0	16#0	
rem_tsap_id[16]	Byte	15.0	16#0	
next_staddr	Array[1..6] of Byte	56.0		Depending on the parameter local_device_id: Rack /

5.2 MODBUS_PARAM

Name	Data type	Offset	Start value	Comment
				slot of the corresponding CPs / irrelevant
next_staddr[1]	Byte	0.0	16#0	
next_staddr[2]	Byte	1.0	16#0	
next_staddr[3]	Byte	2.0	16#0	
next_staddr[4]	Byte	3.0	16#0	
next_staddr[5]	Byte	4.0	16#0	
next_staddr[6]	Byte	5.0	16#0	
spare	Word	62.0	16#0	Idle; must be B#16#00
Modbus settings	Struct	64.0		Settings for the Modbus parameters
server_client	Bool	0.0	false	FALSE: S7 is client; TRUE: S7 is server
single_write	Bool	0.1	false	FALSE: Use of the function codes 15 and 16; TRUE: Use of the function codes 5 and 6
connect_at_startup	Bool	0.2	false	FALSE: Connection establishment for ENQ_ENR being set; TRUE: Connection establishment directly after restart
reserved	Byte	1.0	16#0	Idle; must be B#16#00
data_areas	Struct	2.0		Data areas
data_area_1	Struct	0.0		Data area 1
data_type	Byte	0.0	3	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	11	DB number for data storage
start	Word	4.0	1	First register/bit address stored in the data block
end	Word	6.0	500	Last register/bit address stored in the data block
data_area_2	Struct	8.0		Data area 2
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block
data_area_3	Struct	16.0		Data area 3
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data

5.2 MODBUS_PARAM

Name	Data type	Offset	Start value	Comment
				storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block
data_area_4	Struct	24.0		Data area 4
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block
data_area_5	Struct	32.0		Data area 5
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block
data_area_6	Struct	40.0		Data area 6
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block
data_area_7	Struct	48.0		Data area 7
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block

5.2 MODBUS_PARAM

Name	Data type	Offset	Start value	Comment
data_area_8	Struct	56.0		Data area 8
data_type	Byte	0.0	16#0	1: Coils; 2: Inputs; 3: Holding register; 4: Input register
db	Word	2.0	16#0	DB number for data storage
start	Word	4.0	16#0	First register/bit address stored in the data block
end	Word	6.0	16#0	Last register/bit address stored in the data block
internal_send_buffer	Array[1..260] of Byte	66.0		For internal use
internal_recv_buffer	Array[1..260] of Byte	326.0		For internal use