CPU-CPU Communication with SIMATIC Controllers

SIMATIC S7

Compendium • September 2010

Applications & Tools

Answers for industry.

SIEMENS

Warranty, Liability and Support

Note The application examples are not binding and do not claim to be complete regarding configuration, equipment and any eventuality. The application examples do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible for ensuring that the described products are used correctly. These application examples do not relieve you of the responsibility to use sound practices in application, installation, operation and maintenance. When using these Application Examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these Application Examples at any time without prior notice. If there are any deviations between the recommendations provided in these application examples and other Siemens publications – e.g. Catalogs – the contents of the other documents have priority.

We accept no liability for information contained in this document. Any claims against us – based on whatever legal reason – resulting from the use of the examples, information, programs, engineering and performance data etc., described in this Application Example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health. The above provisions do not imply a change in the burden of proof to your detriment. It is not permissible to transfer or copy these Application Examples or excerpts thereof without express authorization from Siemens Industry Sector.

If you have any questions about this document, please contact us at the following e-mail address:

online-support.automation@siemens.com

Industry Automation and Drives Technologies Service & Support Portal

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

http://support.automation.siemens.com/WW/view/en/20982954

Table of Contents

1 Preliminary Remarks on the Document

1.1 Subject of the document

Communication tasks

In the field of automation technology the communication of controllers plays a crucial part. Controllers perform various communication tasks. The table below illustrates these communication tasks.

Table 1-1

Communication task	Communication partner	Communication	Data (examples)	Network (examples)
Field and process communication	Controller distributed I/O (actuators, sensors)	within a network	Limit-switch positions temperature values	PROFINET/ Industrial Ethernet PROFIBUS
Data communication	Controller 1 Controller 2	within a network or across network boundaries	Setpoint values recipes	PROFINET/ Industrial Ethernet PROFIBUS
IT communication	Controller PC	worldwide	e-mail file	PROFINET/ Industrial Ethernet Internet

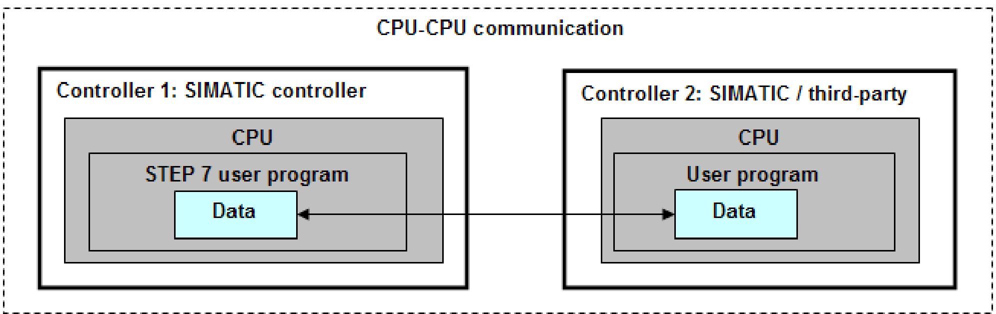
Subject of the document on hand is the data communication between the following communication partners:

- SIMATIC controller / SIMATIC controller
- SIMATIC controller / third-party controller

CPU-CPU communication

For data communication data is exchanged between controllers (data blocks, memory bit, ...). This data is located in the user programs of the CPUs. For clarity reasons the term “CPU-CPU communication” is used for the term “data communication”. The following figure illustrates this.

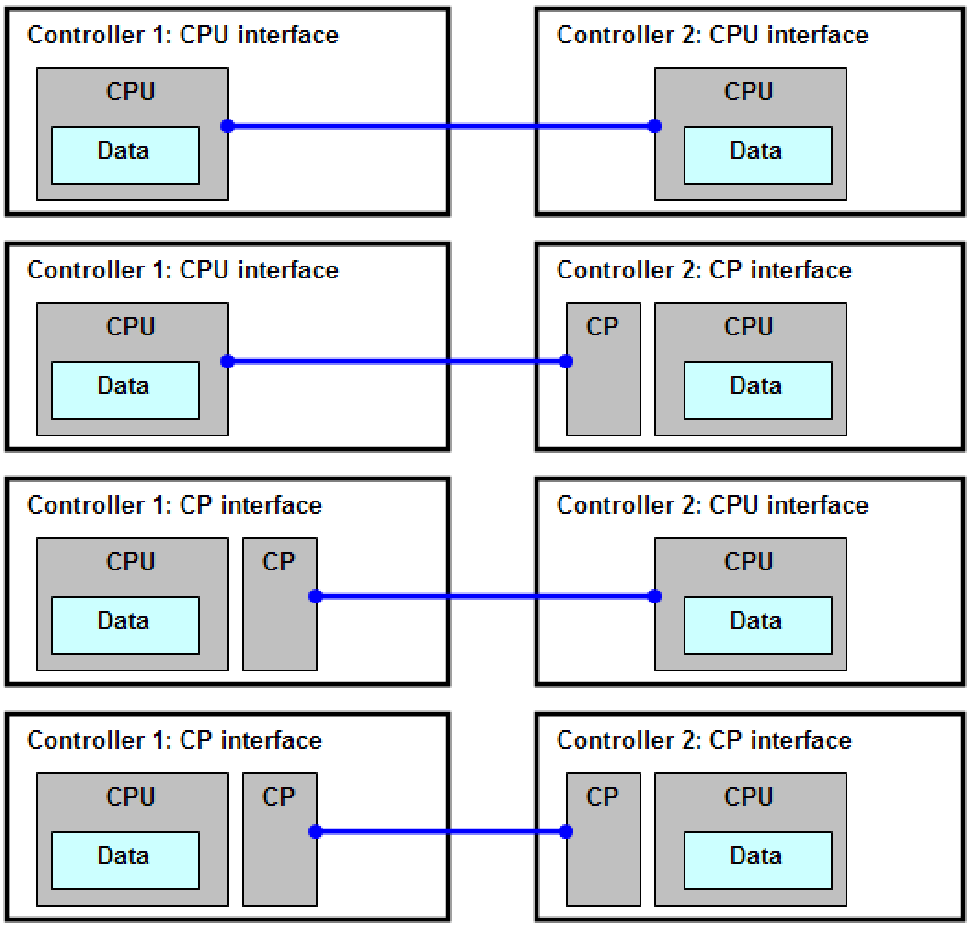
Figure 1-1



Communication paths

For the communication the integrated interface of a SIMATIC CPU, or a SIMATIC CP can be used. The following figure shows the two variants discussed in the document.

Figure 1-2



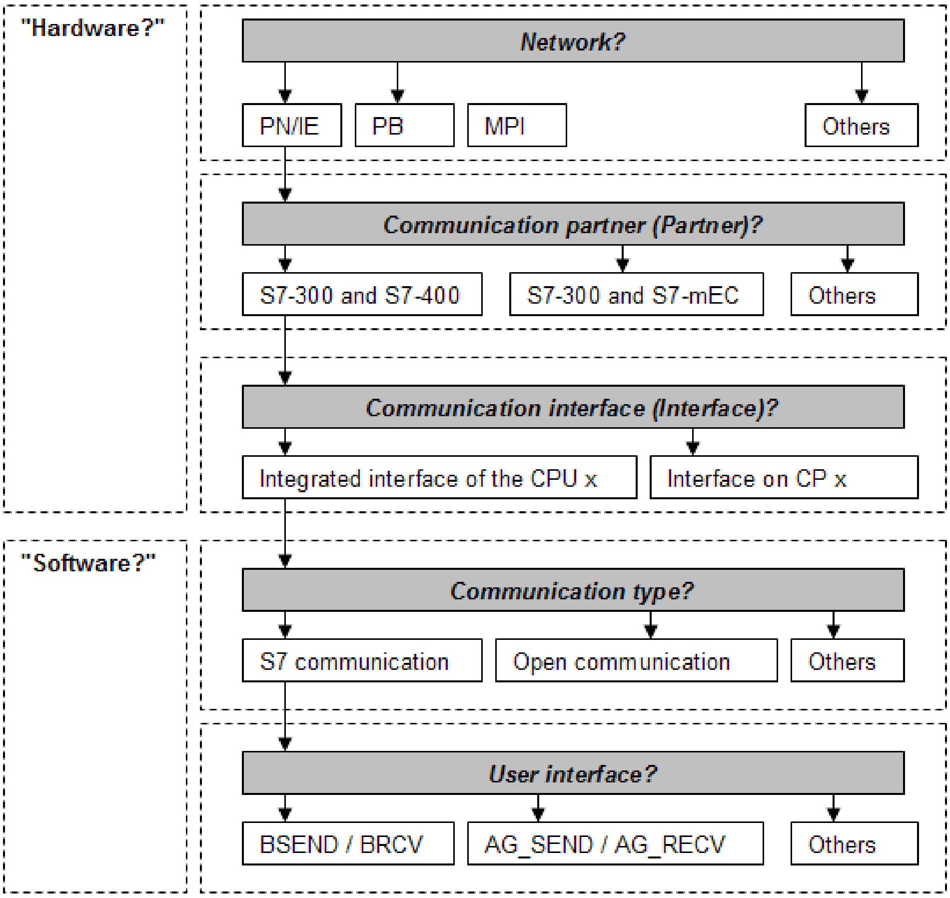
1.2 Purpose and objective of the document

Purpose

To realize a CPU-CPU communication there are a number of different options. In the course of searching for an optimal solution the user is faced with the following questions:

- Which solutions exist?
- In which way do the solutions differ?

Figure 1-3



Objective

The document helps you find an optimal solution for a CPU-CPU communication between two SIMATIC controllers or between a SIMATIC controller and a controller of another manufacturer (“third-party controller”).

In summary the document answers the question:

who can communicate with whom?

1.3 Properties and benefits of the document

Properties

The document has the following properties:

- Clear and compact structure
- Basic content
- No details also available in other documents are described here. The exact working of communication blocks is not described (*1).

(*1): Details on the communication blocks are available in

- STEP 7 online help
- Device manuals on the S7-CPUs and S7-CPs
- System and standard functions for SIMATIC S7-CPUs (/6/)
- Functions and function blocks for SIMATIC NET S7-CPs (/13/)

Benefits

The document provides the following benefits to the user:

- Support for planning and configuration
- Quick finding of information (reference book)
- Transfer of basic knowledge
- Cross-reference to further helpful information (manuals, application examples, FAQs, ...)

1.4 Scope of validity of this document

All statements in the document refer exclusively to the most current components of SIMATIC:

- as of September 2010
- Programming tool STEP 7 V5.5 (except S7-1200)
- Programming tool STEP 7 Basic V10.5 (for S7-1200)

In the document the following topics are discussed:

- Field and process communication (sensors, ...)
- IT communication (e-mail, …)
- Communication with standard PC (OPC, …)
- Communication via modem
- F-communication
- H-communication

1.5 Document classification

For SIMATIC there are a number of documents on the topic of communication. The following table shows how to classify the document on hand.

Table 1-2

Document	Objective of the document	Reference
Document on hand: CPU-CPU communication with SIMATIC controllers	Selection aid for CPU-CPU communication	---
Manuals on the components (S7-CPU, S7-CP, ...)	Technical documentation of the components. (Properties of the interfaces, ...)	/0/ Service & Support Portal
Application examples on the communication	Solutions on specific tasks (documentation and STEP 7 project)	
FAQs on communication	Answers to FAQs	
System manual Communication with SIMATIC	Basic information on industrial communication	/3/
Catalog Products for Totally Integrated Automation and Micro Automation	Ordering document for SIMATIC Controller	/4/
Catalog Industrial communication	Ordering document for SIMATIC Net Products (short description of the networks, ...)	/5/

2 Structure of the Document

The document consists of several parts (part 1 to part 5).
Objective and content of each of the parts is described briefly below.

2.1 Part 1: Introduction

Objective

Part 1 serves as an introduction into the topic of CPU-CPU communication:

- Compact introduction into the topic of communication with SIMATIC
- Explanation of terms and correlations necessary for understanding the document.

Content

The following topics are addressed:

- Functional models on CPU-CPU communication
- Connections for SIMATIC
- Data consistency with SIMATIC
- Overview of SIMATIC controllers
- Media on SIMATIC communication
- Interfaces of the SIMATIC families

The end of this part contains a chapter with references to further information.

2.2 Part 2: Selection aids

Objective

Part 2 is the central part of the document:

- Clear representation of all options for a CPU-CPU communication with SIMATIC controllers

Content

Pro Medium (PN/IE, PB, MPI, …) is described:

- For each SIMATIC family (ET 200 CPU, S7-300, …):
  - what interfaces (CPU, CP) are there?
  - what communication types (S7 communication, ...) are there?
- How can the SIMATIC families communicate with each other?
  - via which interfaces?
  - with which communication types?
  - what are the characteristics (client, server, …) to be considered?
- Comparison of all available communication types

The end of this part contains a chapter with references to further information.

2.3 Part 3: Communication types

Objective

Part 3 provides in-depth information:

- Detailed information on all communication types

Content

The following is described for each Pro communication type
(S7 communication, ...):

- Features
- Properties (table with uniform criteria)
- Principle application (configuration, programming)
- User interface (communication blocks)

The end of this part contains a chapter with references to further information.

2.4 Part 4: Communication with third-party controllers without using an open standard

Part 4 describes examples for communication of SIMATIC controllers with controllers of other manufacturers (third-party controller) via open protocols.

Example: communication with third-party controllers via Modbus/TCP.

The end of this part contains a chapter with references to further information.

2.5 Part 5: Appendix

Contents of Part 5:

- Literature
- Terms and abbreviations
- Background information on selected topics
- Overview of the components discussed in the document
- History of the document

3 Application of the document

This document can be used in different ways:

- Reading the document directly on the PC (online)
- Reading the print-out of the document (offline)

The following description considers this. It is noted in brackets whether the described action is possible online or offline.

3.1 Notes on handling the document

Navigation in the document

Since the document is very extensive, provisions have been made to facilitate handling the document.

Table of contents

The detailed table of contents enables specific selection of chapters (online, offline).

Jump distributor

At the beginning of part 2 of the documentation there is the chapter jump distributor. For each medium there is a page with a collection of cross-references. It lists all important chapters on the respective medium. Cross-references are marked in the document (shaded in gray or framed).

Clicking a cross-reference (online) leads to the respective chapter. At the end of this chapter there is a back jump which enables returning quickly to the jump distributor (online). A back jump is marked as a blue and underlined text.

Example: Back to jump distributor PN/IE

Literature

Bibliographic references are labeled in the text with /x/. Chapter 47 contains a collection of links to the respective sources. Clicking a link (online) takes you directly to the desired information.

Terms and abbreviations

Important terms are explained in chapter 48. Chapter 49 contains a description of all abbreviations.

Background information

Important correlations are explained in chapter 50.

3.2 Example on the application of the document

The application of the document is illustrated using a specific example.

3.2.1 Task

Known

Two SIMATIC controllers shall communicate via PN/IE network:

- Controller 1: from S7-300 family
- Controller 2: from S7-400 family

Sought

We are looking for the answers to the following questions:

- Question 1: which interfaces and communication types are available per family?
- Question 2: which components can communicate with each other and which communication types are possible?
- Question 3: which properties have the available communication types?
- Question 4: what do user interfaces (communication blocks) specifically look like?

Solution

For each medium (PNIE, PB, MPI, ...) there is a so-called jump distributor in the document.

The jump distributor consists of the following sections:

- Overview of interfaces and communication types
- Combination controller 1 / controller 2
- Communication types

With the jump distributor the above questions are answered quickly. This is shown in the following chapters using examples.

3.2.2 Overview of interfaces and communication types

The following figure shows the section “overview interfaces and communication types” from the jump distributor for PN/IE (chapter 18.1).

Figure 3-1

Family		Chapter
Modular controllers	ET 200 CPU	19.2.1
	S7-300	19.2.2
	S7-400	19.3.4
	S7-1200	19.2.4
Embedded controllers	S7-mEC	19.2.5
	Box PC	19.2.6
	Panel PC	19.2.7
	WinAC MP	19.2.8
PC-based controller	WinAC RTX	19.2.9

Procedure for answering question 1 (which interfaces and communication types are available per family?):

Click the chapter for the searched SIMATIC family (online) or open the respective chapter (offline).

Result:

The chapter contains a table which lists all interfaces and communication types of the respective family. The following figure shows a section of this table for the S7-300 family (Table 19-2).

Figure 3-2

Controller to PN/IE: S7-300		Communication type	
		SIMATIC specific	Open standard
		S7 communication	Open communication
CPU	all with interface: PN (*2)	(1) (IoT) (14) (IoT)	(3) (IoT, TCP, UDP) (13) (IoT, TCP) (*1)
CP	343-1 Lean	PUT, GET Server (IoT)	(8) (IoT, TCP, UDP) (13) (IoT, TCP)
	343-1	(1) (IoT, ISO)	(8) (IoT, TCP, UDP, ISO) (13) (IoT, TCP, ISO)
	343-1 Advanced	(1) (IoT, ISO)	(8) (IoT, TCP, UDP, ISO) (13) (IoT, TCP, ISO)
	343-1 ERPC (*3)	(1) (IoT)	(8) (IoT, TCP, UDP) (13) (TCP)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_
(3) TSEND/TRCV, TUSEND/TURCV
(6) PNIO_SEND, PNIO_RECV
(8) AG_SEND/AG_RECV
(13) Server for Fetch/Write

This is how to read the table (example in the red frame):

- S7-300 can be operated at the PN/IE using CP 343-1.
- One possible communication type is the S7 communication (server and client).
- The communication blocks listed in index (1) can be employed: USEND/URC, BSEND/BRCV, PUT, GET
- The protocols available here are in brackets:
  ISO on TCP, ISO

Note

Only the principle of the table shall be shown here. A detailed description of the table structure is contained in chapter (17.2).

3.2.3 Combination controller 1 / controller 2

The following figure shows the section “Combination controller 1 controller 2” from the jump distributor for PN/IE (chapter 18.1).

Figure 3-3

Controller 2		Controller 1								
		Modular				Embedded				PC-based
		ET 200 CPU	S7-300	S7-400	S7-1200	S7-mEC	Box PC	Panel PC	WinAC MP	WinAC RTX
Modular	ET 200 CPU	19.3.1	19.3.2	19.3.3	19.3.4	19.3.5	19.3.6	19.3.7	19.3.8	19.3.9
	S7-300	19.3.2	19.4.2	19.4.3	19.4.4	19.4.5	19.4.6	19.4.7	19.4.8	19.4.9
	S7-400	19.3.3	19.4.3	19.5.3	19.5.4	19.5.5	19.5.6	19.5.7	19.5.8	19.5.9
	S7-1200	19.3.4	19.4.4	19.5.4	19.6.4	19.6.5	19.6.6	19.6.7	19.6.8	19.6.9
Embedded	S7-mEC	19.3.5	19.4.5	19.5.5	19.6.5	19.7.5	19.7.6	19.7.7	19.7.8	19.7.9
	Box PC	19.3.6	19.4.6	19.5.6	19.6.6	19.7.6	19.8.6	19.8.7	19.8.8	19.8.9
	Panel PC	19.3.7	19.4.7	19.5.7	19.6.7	19.7.7	19.8.7	19.9.7	19.9.8	19.9.9
	WinAC MP	19.3.8	19.4.8	19.5.8	19.6.8	19.7.8	19.8.8	19.9.8	19.10.8	19.10.9
PC-based	WinAC RTX	19.3.9	19.4.9	19.5.9	19.6.9	19.7.9	19.8.9	19.9.9	19.10.9	19.11.9

<u>Procedure for answering question 2 (which components can communicate with each other and which communication types are possible?):</u>

Click the chapter for the searched combination of two SIMATIC families (online) or open the respective chapter (offline).

<u>Result</u>:

The chapter contains a table which lists all combinations of the interfaces (CPU, CP) of both families. The possible communication types have been entered for each combination. The following figure shows a section of this table for the S7-300 and S7-400 families (Table 19-23).

Figure 3-4

Controller 2: S7-400			**Controller 1: S7-300 to PN/IE**							
			CPU				CP			
			all with interface: PN				343-1 Lean			
			IOC, IOD				IOD			
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**	
					PNIO	CBA			PNIO	CBA
CPU	all with interface: PN	IOC, IOD	(1)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---
CP	443-1	IOC, IOD	(1)	(3)/(4)+(9)	(2)	---	(21)	(8)/(4)+(9)	(6)/(2)	---
	443-1 Advanced	IOC, IOD	(1)	(3)/(4)+(9)	(2)	x	(21)	(8)/(4)+(9)	(6)/(2)	---

<u>Back to jump distributor PN/IE</u>

<u>Communication blocks</u>

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_D
(3) TSEND/TRCV, TUSEND/TURCV
(4) TSEND/TRCV
(6) PNIO_SEND, PNIO_RECV
(21) Controller 1 is server (for PUT, GE
(8) AG_SEND/AG_RECV
(9) AG_SEND/AG_RECV, AG_LSEND

This is how to read the table (example in the red frame):

- All S7-300 CPUs with PN interface (controller 1) can communicate with CP 443-1 Advanced (controller 2).
- Possible communication types:
  - S7 communication (S7)
  - Open communication (OC)
  - PN communication (PN)
- Possible communication blocks for open communication:
  - Controller 1: T-blocks (3)
  - Controller 2: T-blocks (4) and send/receive blocks (9)
- Possible communication types for PN communication:
  - PNIO with the user interfaces (2)
  - CBA

Note

Only the principle of the table shall be shown here. A detailed description of the table structure is contained in chapter (17.3).

3.2.4 Communication types

The following figure shows the section “communication types” from the jump distributor for PN/IE (chapter 18.1).

Figure 3-5

Communication type		Chapter
Communication types (compact table)		19.12
Table with details	S7 communication	29.2
	Open communication with send/receive blocks	31.2
	Open communication with T blocks	32.2
	PNIO	34.2
Modbus/TCP (SIMATIC / third-party controller)		44

1
2

Overview of all communication types

Procedure for answering question 3 (which properties have the available communication types?):

Click (online) or open (offline) the chapter (1).

Result:

The table in this chapter shows all communication types possible via PN/IE in comparison.

The following figure shows a section from the table. (Table 19-58).

Figure 3-6

	SIMATIC-specific	Open standard	
	S7 communication	Open communication	
		Send/Receive blocks	T-blocks
Protocols	ISO, IoT	ISO, IoT, TCP, UDP	IoT, TCP, UDP
Interfaces	ISO: CP IoT: CPU, CP	CP	CPU, CP
Communication blocks (max. data)	BSEND (≤ 64 Kbytes) USEND,USEND_E (≥ 160 bytes) PUT, PUT_E, GET, GET_E (≥ 160 bytes)	AG_xSEND (ISO, IoT, TCP ≤ 8 Kbytes) (UDP ≤ 2 Kbytes) ------------------------------------ Server for FETCH, WRITE (not for UDP)	TSEND, TUSEND, ... (IoT ≤ 32 Kbytes) (TCP ≤ 64 Kbytes) (UDP = 1472 bytes)
remote confirmation	BSEND: application USEND: transport PUT, GET: application	ISO, IoT, TCP: transport UDP: none	IoT, TCP: transport UDP: none
Routing-capable?	ISO: no IoT: yes	ISO: no sonst: yes	yes
Connections?	yes	UDP: no sonst: yes	UDP: no sonst: yes

Details on a specific communication type

Procedure on answering question 4 (what do user interfaces (communication blocks) specifically look like?):

Click (online) or open (offline) the chapter (2).

<u>Result:</u>

The table in this chapter shows all important properties of the S7 communication.

The following figure shows a section from the table (Table 29-1).

Figure 3-7

Communication type:		S7 communication		
Protocol:		S7 protocol (*1)		
General information				
Media		MPI, PB, PN/IE, backplane bus (*3)		
Interfaces		CPU, CP		
Connection	SIMATIC S5	no		
	third-party (open standards)	no		
User interface				
Communication blocks		**BSEND / BRCV**	**USEND / URCV USEND_E / URCV_E (*4)**	
max data volume (*2)		<= 64 Kbytes	>= 160 bytes	
Number of variables when calling the communication block		1	S7-300:	USEND/URCV: 1
				USEND_E/URCV_E: 1 to 4
			S7-400: 1 to 4	
dynamic addressing of data		S7-300: yes	S7-300: yes	
		S7-400: no	S7-400: no	
remote addressing		application	transport	
Model		client / client	client / client	

If further information on the communication blocks is required, then these can be read up in the chapter on S7 communication.

Example: parameters of the communication block BSEND (Table 29-12):

Figure 3-8

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the send job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running /job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Send area (*2)
LEN	WORD	Length of the data to be sent

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas:

- S7-300: M, D
- S7-400: E, A, M, D, T, Z

PART 1: Introduction

Part 1 serves as an introduction into the topic of CPU-CPU communication

Explanation of terms and correlations necessary for understanding the document.

PART 1: Structure and content

Table 3-1

Chapter	Structure		Content
4	Functional model CPU-CPU communication		Overview of functional models
5	Connections for SIMATIC		The most important information on connections
6	Data consistency with SIMATIC		Explanations on data consistency
7	SIMATIC Controller		Overview of SIMATIC Controller and families
8	Media for SIMATIC communication		Overview of all media for SIMATIC communication
9		PROFINET/Industrial Ethernet (PN/IE)	The following is described per medium: • Characteristics • ISO/OSI reference model
10		PROFIBUS (PB)	
11		MPI	
12		SIMATIC backplane bus	
13		Serial interface (PtP)	
14		Comparison of the media	Table comparison of the media
15	Interfaces of the SIMATIC families		Overview of all SIMATIC interfaces
16	Information		Notes regarding further information

4 Models on CPU-CPU Communication

4.1 Definition of controller

The following definitions are used in the document:

A controller is a central or decentralized automation station (station) with the components CPU, CP (optional) and distributed I/O. Within the station the components are connected via the backplane bus.

Central station:

- contains a distributed I/O
- communicates with distributed stations via PROFINET IO or PROFIBUS DP

Decentralized station:

- contains a distributed I/O
- communicates with central station via PROFINET IO or PROFIBUS DP

4.2 Definition CPU-CPU communication

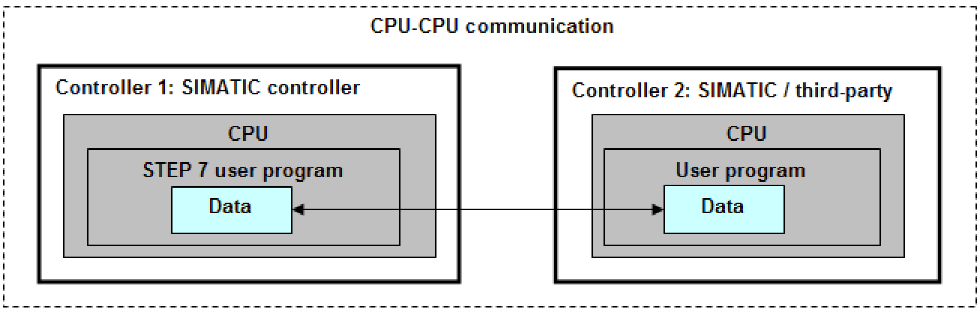
For CPU-CPU communication data is exchanged between two controllers:

- Controller 1: SIMATIC controller
- Controller 2: SIMATIC controller or third-party controller

Source or target of the data is the user data area of the CPU of the controller:

- data block, flag, inputs, outputs, ...

Figure 4-1



For the CPU-CPU communication the following cases are differentiated:

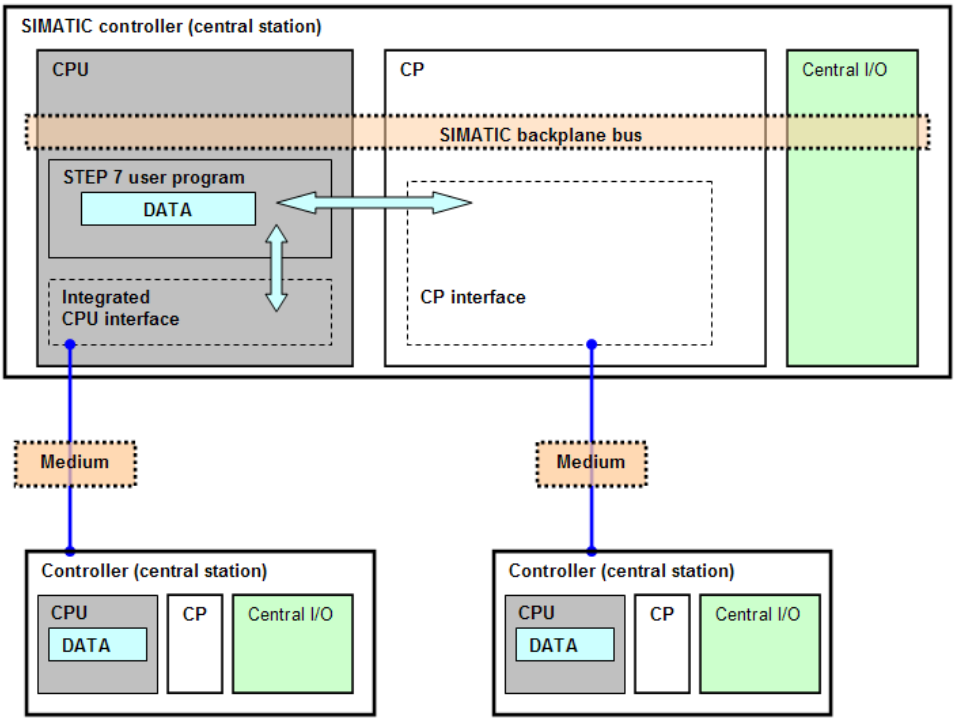
- CPUs in different central stations
- CPUs in central and decentralized station (*1)
- CPUs within a central SIMATIC station

Regarding (*1): a decentralized station with CPU is also referred to as I-slave (for PROFIBUS) or I-device (for PROFINET).

4.3 CPUs in different central stations

The figure shows the functional model for the CPU-CPU communication between distributed stations.

Figure 4-2



Interfaces for communication:

- Interface on CPU (integrated interface)
- Interface on CP (external interface)

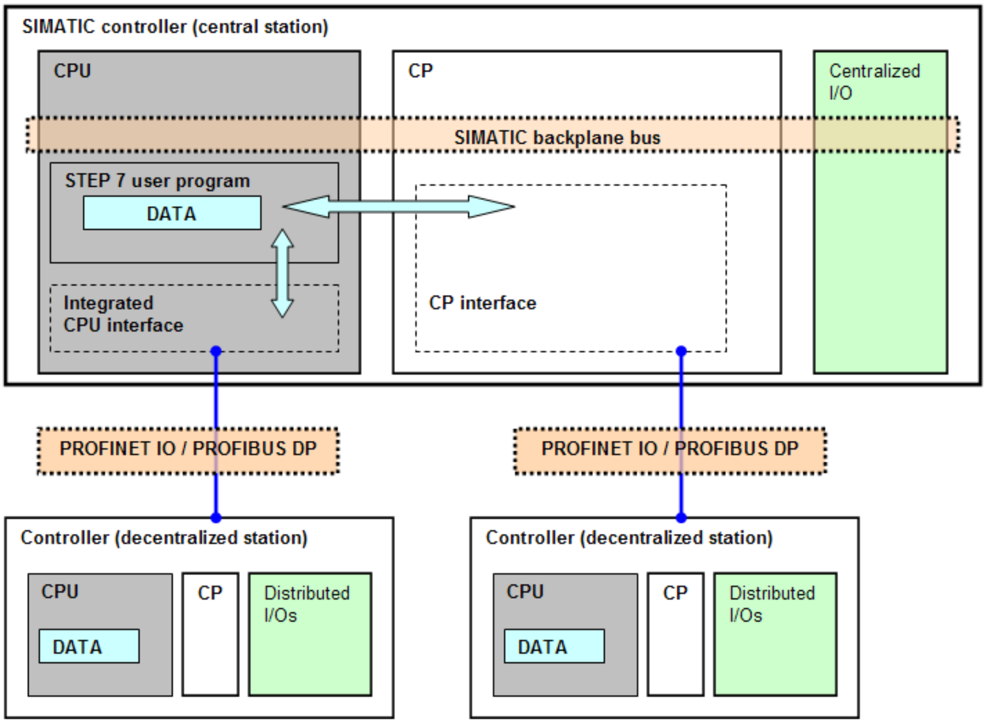
Media for communication:

- Network (PROFINET/Industrial Ethernet, PROFIBUS, MPI)
- Serial interface (ASCII, 3964(R), RK 512, …)

4.4 CPUs in central and decentralized station

The figure shows the functional model for the CPU-CPU communication between central and decentralized station.

Figure 4-3



Interfaces for communication:

- Interface on CPU (integrated interface)
- Interface on CP (external interface)

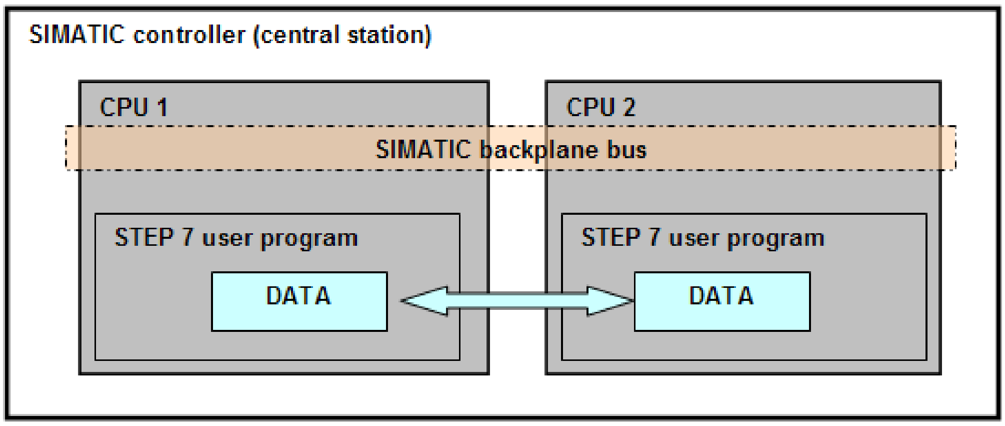
Media for communication:

- PROFINET/Industrial Ethernet (PROFINET IO)
- PROFIBUS (PROFIBUS DP)

4.5 CPUs within a central station

The figure shows the functional model for the CPU-CPU communication between CPUs within a central SIMATIC station.

Figure 4-4



Medium for communication:

- SIMATIC backplane bus

Note

This is only possible for S7-400, and is referred to as “multi-computing” there. Up to 4 S7-CPUs in a central SIMATIC station can be operated simultaneously.

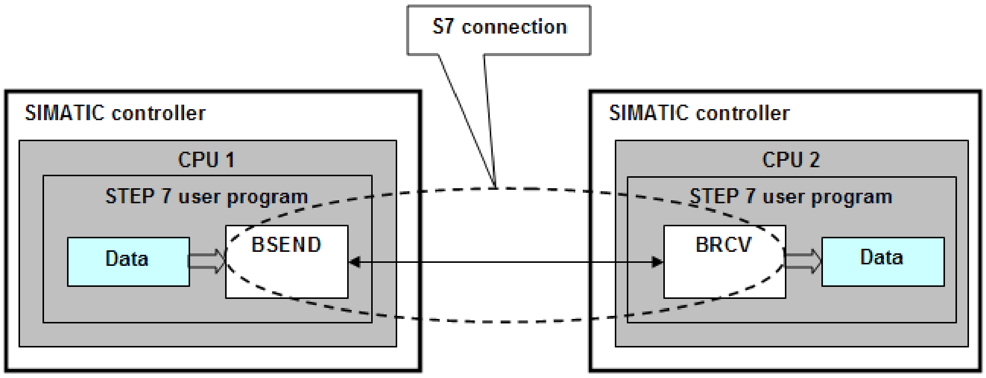
5 Connections for SIMATIC

5.1 Connections

Introduction

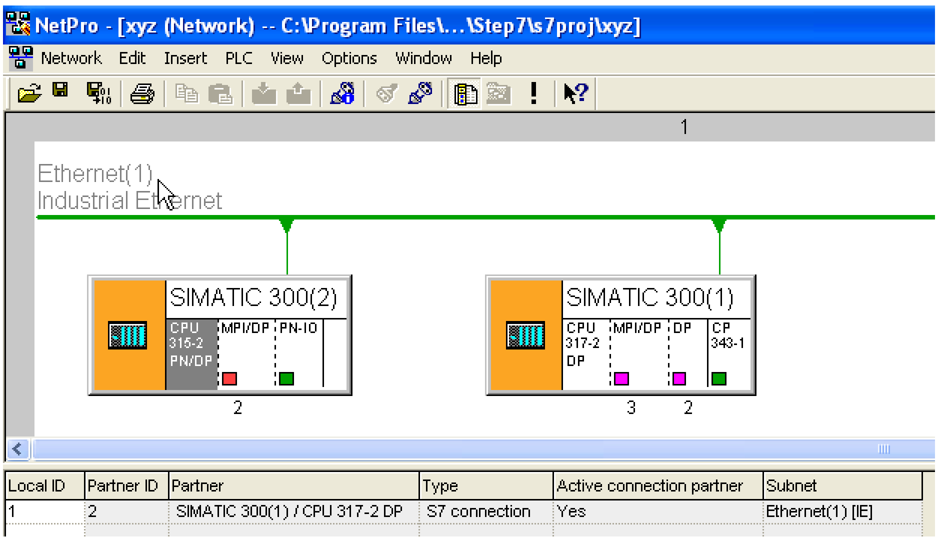
For the most SIMATIC communication types CPU-CPU communication occurs via connections. A connection defines which communication partners (CPUs) communiate with each other. The figure below shows an example of a connection: CPU 1 communicates with CPU 2 via an S7 connection.

Figure 5-1



Connections must be established (chapter 5.4). This is possible for example via configuration. The following figure shows an S7 connection configured in STEP 7 (NetPro).

Figure 5-2



Properties of a connection

A connections is defined by the following properties (examples):

- involved CPUs (CPU 1, CPU 2)
- used protocol (TCP, ISO on TCP, ...)
- behavior after the data transmission:
  connection remains established is cancelled

Function of a connection

During the connection process both communication partners (integrated interfaces of the CPU or interface of the CP) negotiate the connection parameters (maximal frame length, ...).

The active communication partner suggests a value. The other communication partner confirms or suggests something else. The negotiated value then applies for the duration of the connection.

During an established connection the following tasks are performed by the operating system of the CPU or the CP:

- Flow control (avoiding overload in the partners, …)
- Monitoring of the connection (partner can still be reached, ...)
- Exchange of acknowledgements (data arrived without error, …)

5.2 Protocols

A protocol defines the rules of the communication between two communication partners. An ISO/OSI reference model is often used to describe (chapter 50.1).

Two classes are described for the protocols.

Connection-oriented protocols

These protocols establish a connection between both communication partners.

Examples: TCP, ISO on TCP

The protocols are used if the priority lies on a reliable data transmission

Protocols without connection

These protocols do not establish a connection between both communication partners.

Example: UDP

The protocols are used if the priority lies on a fast data transmission

5.3 Connection resources

Description

Connections use resources on the CPU or the CP (e.g. memory areas of the operating system). In SIMATIC these resources are also referred to as "connection resources".

The number of maximal possible connections per CPU or CP is limited. It depends on the available connection resources of the CPU or the CP. The technical data in the manuals for CPUs and CPs therefore states how many connections are possible per communication type.

The following figure shows an example from the S7-300 manual.

Figure 5-3

Technical data of CPU 31x

8.5 CPU 315-2 PN/DP

Technical data	
Diagnostics buffer	Yes
• Number of entries (not configurable)	max. 500
• POWER OFF / POWER ON	The last 100 entries are retentive
Communication functions	
Open IE communication	
Number of connections / access points, total	8
TCP/IP	Yes (via integrated PROFINET interface and loadable FBs)
• Maximum number of connections	8
• Data length for connection type 01_H, max.	1460 bytes
• Data length for connection type 11_H, max.	8192 bytes
ISO on TCP	Yes (via integrated PROFINET interface and loadable FBs)
• Number of connections, max.	8
• Data length, max.	8192 bytes
UDP	Yes (via integrated PROFINET interface and loadable FBs)
• Number of connections, max.	8
• Data length, max.	1472 bytes

Assignment and release

Connection resources are assigned during the connection process and are released upon disconnection.

If connections are configured in STEP 7 (NetPro), then STEP 7 monitors meeting the technical data regarding the maximal possible connections.

If connections are established or cancelled in the STEP 7 user program the user must manage the connection resources himself. This means that the user must ensure that the number of maximal possible connections of the modules involved in the communication is met.

5.4 Establishing connections

Connections must be established.

The following cases must be distinguished:

- configured connection
- non-configured connection

In the following chapters both scenarios are described.

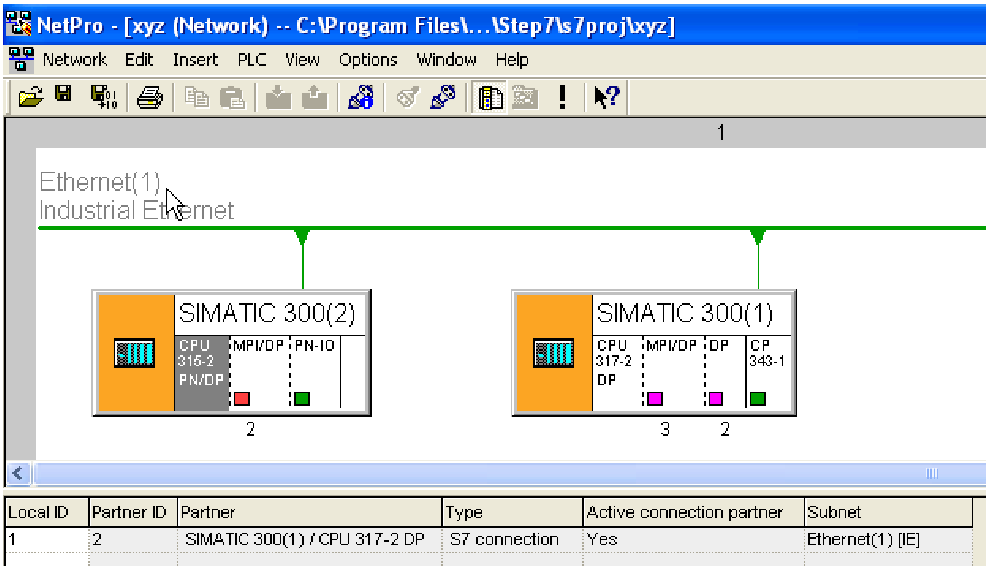
Table 5-1

	Chapter
Configured connection	5.5
Non-configured connection	5.6

5.5 Configured connection

Configured connections are connections established with STEP 7 (NetPro). The figure below shows an example.

Figure 5-4



Configured connections are, for example, used for the following communication types:

- S7 communication
- Open communication with send/receive blocks

Realizing a communication

To realize a communication the following steps must be performed:

- Configuration with STEP 7:
  connection configuration with NetPro
- Programming in STEP 7:
  calling communication blocks (for data transmission)

For the connection configuration a unique local ID per connection is assigned, the "Local ID". This local ID is required during configuration of the communication blocks.

Connecting and disconnecting process

Connecting

Automatic connection during startup of the communication-capable modules (CPU, CP). During establishing the connection the required connection resources are assigned by the operating system of the modules.

Disconnecting

The connection is cancelled after the data transfer, i.e. the connection resources remain permanently assigned.

5.6 Non-configured connection

Non-configured connections are connections not configured with STEP 7 (NetPro).

Non-configured connections are used for the following communication types (examples):

- S7 basic communication
- open communication with T blocks

Two cases must be distinguished:

- automatic establishing of a connection (example: S7 basic communication)
- programmed establishing of a connection (example: open communication with T blocks)

5.6.1 Automatic establishing of a connection

Realizing a communication

To realize a communication the following steps must be performed:

- Programming in STEP 7:
  calling communication blocks (for data transmission)

Connecting and disconnecting process

Connecting

At the first call of the communication block the operating system of the communication-capable modules (CPU, CP) establishes a connection. During establishing the connection the required connection resources are assigned by the operating system of the modules.

Disconnecting

The configuration at the communication block defines whether the connection remains after completing the data transmission or whether the connection is cancelled. Canceling an existing connection releases previously assigned connection resources.

If the connection is not cancelled after transmitting the data, the processing time of the communication block for a renewed transmission becomes shorter, since the connection already exists and needs not be established first.

5.6.2 Programmed establishing of a connection

Realizing a communication

To realize a communication the following steps must be performed:

- Programming in STEP 7:
  calling the connection blocks (for establishing connections)
- Programming in STEP 7:
  calling communication blocks (for data transmission)

Connecting and disconnecting process

Connecting

When calling the connection block the operating system of the communication-capable modules (CPU, CP) establishes a connection.

Disconnecting

The connections can be cancelled by calling a connection block. This can release connection resources.

6 Data Consistency with SIMATIC

In this chapter the topic of data consistency is discussed from the point of view of CPU-CPU communication with SIMATIC controllers.

6.1 Definitions

Data area

A data area is a connected area of data in the user memory of a SIMATIC CPU (for example: MW100 to MW200).

Consistent data

A data area which cannot be changed simultaneously by competing processes (user programs, operating system programs, ...) is referred to as a consistent (related) data area.

This data area contains consistent data. In the document, the size of this data area is referred to as “volume of consistent data”.

Inconsistent data

A data area larger than the consistent data area can be falsified. The data area can at one given time consist of new and partially of old consistent data areas.

<u>Example</u>

Inconsistent data can occur if in the STEP 7 user program a running communication block is interrupted by a process alarm OB with higher priority. If the user program then changes the data in this OB, which partially have already been processed by the communication block, then inconsistent data may occur.

The data is then inconsistent (unrelated), because:

- part of the data origins from the time before process alarm processing (“old data area”)
- part of the data origins from the time after process alarm processing (“new data area”)

Data consistency

For data consistency two cases are distinguished:

- system-related data consistency
- additional data consistency

<u>Properties of the system-related data consistency:</u>

Point (b) applies under boundary condition (a):

(a): data area with “volume of data” ≤ “volume of consistent data”

(b): data consistency is guaranteed <u>without</u> additional measures in the STEP 7 user program

Properties of the additional data consistency:

Point (b) applies under boundary condition (a):

(a): data area with “volume of data” ≤ “volume of consistent data”

(b): data consistency is not guaranteed. Data consistency is only guaranteed with additional measures in the STEP 7 user program (example: block process alarm OB for the duration of the data transmission)

6.2 System-related data consistency

The operating system of the SIMATIC controller guarantees a system-related data consistency. This system-related data consistency depends on:

- type CPU (in case of communication via CPU)
- type CP and type CPU (in case of communication via CP)
- communication block type

A number of statements on the various SIMATIC families are given below.

6.2.1 S7-300

For S7-300 the data is copied consistently from the operating system to the STEP 7 user memory in blocks of x bytes (see below). Copying occurs in the cycle control point of the operating system. For larger data areas no data consistency is guaranteed by the system.

If a certain data consistency is required the data in the user program must not exceed these x bytes.

Table 6-1

Case discrimination	Volume of consistent data
Communication via integrated interface of the CPU	64 bytes up to 240 bytes (*1)
Communication via CP	32 bytes

(*1): The specific values on the “volume of consistent data” is available in the manuals of the CPUs or CPs.

6.2.2 S7-400

For S7-400, as opposed to the S7-300, the data are not processed in the cycle control point of the operating system but in fixed time slices during the cycle (OB1).

The data consistency of a variable (byte, word or double-word) is guaranteed. The maximal data consistency is 32 bytes.

6.2.3 S7-1200

The CPU guarantees data consistency for all elementary data types (e.g. Word or DWord) and all system-defined structures (e.g. IEC_TIMERS or DTL).

6.3 Additional data consistency

6.3.1 Additional measures

In order to guarantee additional data consistency, additional measures must be taken in the STEP 7 user program of sender and receiver:

Additional measures in the sender

Access to send area (data block, flag, …) only if the data has been completely transferred. This can be read from the control parameters of the communication blocks (example: DONE = 1).

Additional measures in the receiver

Access to receive area (data block, flag, …) only if the data has been completely received. This can be read from the control parameters of the communication blocks (example: NDR = 1).

Then blocking the receive area until the data has been processed. This can be read from the control parameters of the communication blocks (example: EN_R = 0).

6.3.2 Case discrimination

Two cases must be distinguished:

- client-client communication
- client-server communication

Client-client communication

Examples for communication blocks: BSEND / BRCV

If additional data consistency is to be guaranteed the data must not be modified during the transmission (see “Additional measures” above).

Client-server communication

Examples for communication blocks: PUT, GET

In the STEP 7 user program of the server no communication block exists. Therefore the access to the data in the user program cannot be coordinated.

During programming or configuration the system-related size of the consistent data areas (system-related data consistency) must be considered.

7 SIMATIC Controller

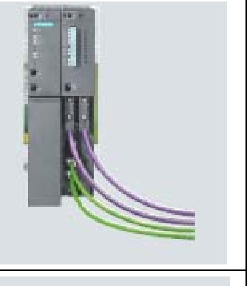
An overview of the SIMATIC families (/1/) discussed in this document follows below. The abbreviations for the SIMATIC families used in the document come in brackets.

Note

The headers (SIMATIC Modular Controller, ...) on the SIMATIC families (ET 200 CPU, S7-300, ...) correspond to the current navigation in the product support of the Service & Support Portal (as of September 2010, (/0/).

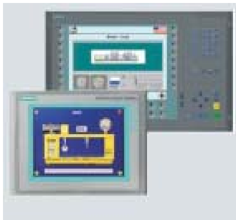
SIMATIC Modular Controller

Table 7-1

SIMATIC family	
SIMATIC ET 200 (ET 200 CPU) Modular, distributed I/O system with local intelligence	
SIMATIC S7-300 (S7-300) The modular controller for system solutions in the manufacturing industry	
SIMATIC S7-400 (S7-400) The power controller for system solutions in the manufacturing and process industry	
SIMATIC S7-1200 (S7-1200) Modular, compact controller for discrete and stand-alone automation solutions	

SIMATIC Embedded Automation

Table 7-2

SIMATIC family	
SIMATIC S7-modular Embedded Controller (S7-mEC) Embedded controller in S7-300 design with Software Controller and Runtime visualization software	
SIMATIC Embedded Box PC-Bundles (Box PC) Ready to switch on top hat rail PC with Software Controller and Runtime visualization software	
SIMATIC Embedded Panel PC-Bundles (Panel PC) Ready to switch on Panel PC with software controller and Runtime visualization software	
SIMATIC WinAC MP for multi panels (WinAC MP) Software controller for multi panels	

SIMATIC PC-based controller

Table 7-3

SIMATIC family	
SIMATIC WinAC RTX (WinAC RTX) Software controller – open, flexible and reliable	

8 Media on SIMATIC communication

For SIMATIC there are various options to realize a CPU-CPU communication. Data can be transmitted via different media. The following table shows the available media.

Table 8-1

Medium		Communication partner	
		SIMATIC S7 Controller	Third-party controller
Net	PROFINET/Industrial Ethernet (PN/IE)	x	x
	PROFIBUS (PB)	x	x
	MPI	x	---
SIMATIC backplane bus (*1)		x	---
Serial interface (PtP)		x	x

(*1): only possible for SIMATIC S7-400 (multi-computing, chapter 4.5)

In the following chapters, the media are briefly characterized.

Details on the media can be researched in the extensive literature available (chapter 16).

Overview of the following chapters:

Table 8-2

Media on SIMATIC communication		Chapter
PROFINET/Industrial Ethernet (PN/IE)		9
	Preliminary remarks	9.1
	Ethernet	9.2
	Industrial Ethernet (IE)	9.3
	PROFINET (PN)	9.4
PROFIBUS (PB)		10
MPI		11
SIMATIC Backplane Bus		12
Serial Interface (PtP)		13
Comparison of the Media		14

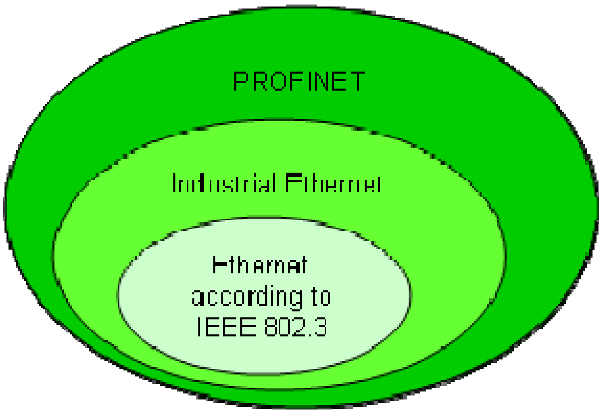
9 PROFINET/Industrial Ethernet (PN/IE)

9.1 Preliminary remarks

PROFINET, Industrial Ethernet and Ethernet are related terms:

- PROFINET is based on Industrial Ethernet
- Industrial Ethernet is based on Ethernet.

Figure 9-1



Ethernet, Industrial Ethernet and PROFINET are briefly characterized below.

9.2 Ethernet

Introduction

Ethernet is the standard for networks (LAN) in the office communication sector.

Characteristics

- International standard: IEEE 802.3
- Worldwide distribution
- Simple and standardized cabling
- Basis higher-level protocols (TCP/IP, UDP, …)
- Failsafe networks due to redundancy
- Simple connection to wireless networks
  (Industrial Wireless LAN, according to IEEE 802.11)
- Scalable performance due to switched Ethernet (*1)

(*1): Switched Ethernet

Switched Ethernet divides the network into subnets connected by switches.

The following functionality can be realized this way:

- Several pairs of nodes are interconnected simultaneously. Each connection has full data throughput.
- Local data traffic remains local. Only data of another subnet are passed on by the switches.

Advantage of switched Ethernet:

- Increased data throughput due to structured data traffic

For SIMATIC a switch can be realized in different ways:

- as independent component (SCALANCE X)
- integrated in SIMATIC components (S7-CPU, S7-CP with PN/IE connection)

ISO/OSI reference model

Ethernet comprises layer 1 and layer 2 of the ISO/OSI reference model:

- Layer 2: Access control and addressing (MAC addresses)
- Layer 1: transmission technology (physics)

Figure 9-2

Layer 7	Application Layer		
Layer 6	Presentation Layer		
Layer 5	Session Layer		
Layer 4	Transport Layer		
Layer 3	Network Layer		
Layer 2	Data Link Layer	IEEE 802.2 Logical Link Control (LLC) IEEE 802.1 Media Access Control (MAC)	**Ethernet**
Layer 1	Physical Layer	IEEE 802.3 Ethernet	

9.3 Industrial Ethernet (IE)

Introduction

IE is the industry-capable version of Ethernet.

Characteristics

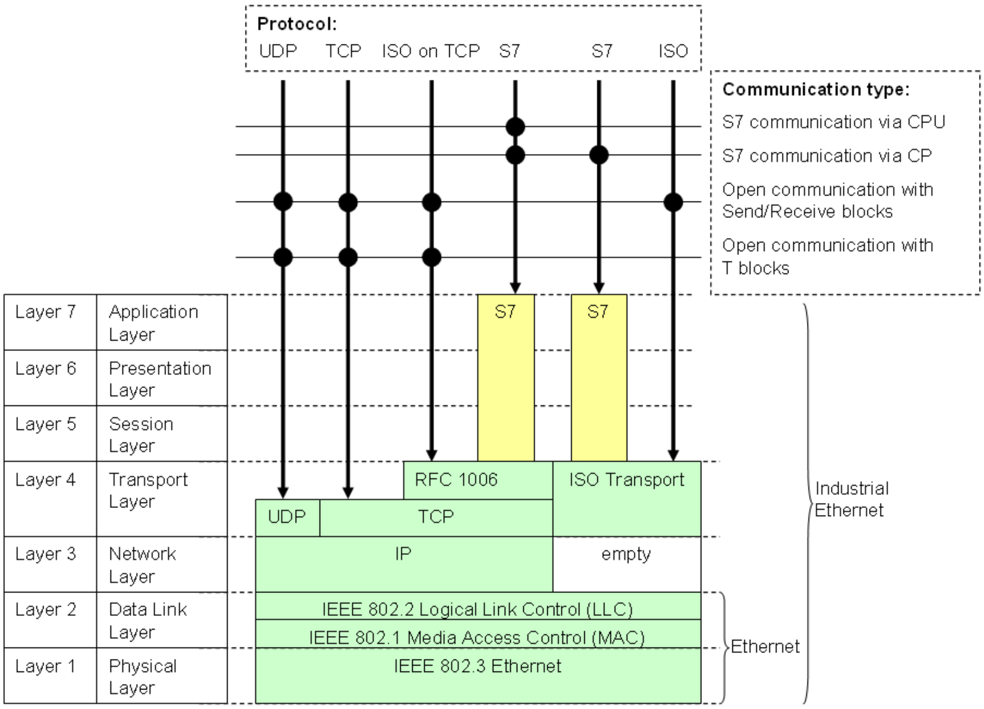
In addition to the characteristics for Ethernet (chapter 9.2), there are the following characteristics for Industrial Ethernet:

- Combines different application areas: office and production
- Uses the possibilities of IT standards (known from the office communication sector) in automation technology (browser, e-mail, …).
- Optimized communication between automation components and simultaneous communication according to TCP/IP (open standard)
- Network components for application in harsh industrial environment (dust, humidity, vibrations, …)
- Simple connection technology on site

ISO/OSI reference model

Ethernet comprises layer 1 to layer 7 of the ISO/OSI reference model. The following figure shows all protocols and communication types supported by the SIMATIC controllers.

Figure 9-3



9.4 PROFINET (PN)

Introduction

PN is the open Industrial Ethernet standard for automation. PN is based on Industrial Ethernet.

<u>Versions</u>

Within PROFINET there are two versions:

- PROFINET IO
- PROFINET CBA

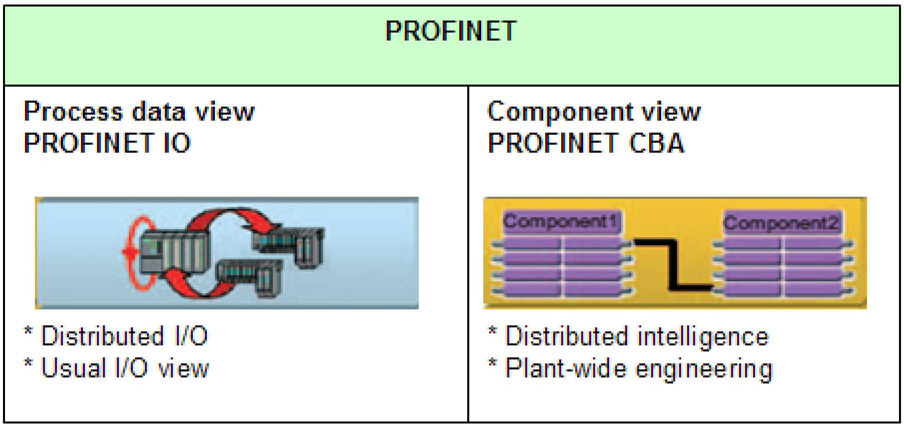
PROFINET IO is an automation concept for realizing modular applications through integrating distributed I/O with real-time communication.

PROFINET CBA is a component model for automation solutions based on distributed components and partial functions.

<u>Views</u>

PROFINET IO and PROFINET CBA are two different views on automation devices at the Industrial Ethernet.

Figure 9-4



PROFINET IO provides a picture of the automation plant very similar to the PROFIBUS DP view. The individual automation devices are configured and programmed.

PROFINET CBA divides a complete automation plant into various functions. These functions are configured and programmed.

Characteristics

PROFINET

- International standard: IEC 61158, IEC 61784
- Consistent communication via fieldbus and Ethernet
- Integration of existent fieldbus systems (PROFIBUS, ASi)
- Using the TCP/IP protocol
- Communication in real-time
- Clock-synchronized drive regulation for motion control applications

PROFINET IO

- Communication of field devices (IO device) with controllers (IO controller)
- IO view such as for PROFIBUS DP

PROFINET CBA

- Communication between CBA components
- Communication is configured, not programmed (with the Engineering Tool iMap)

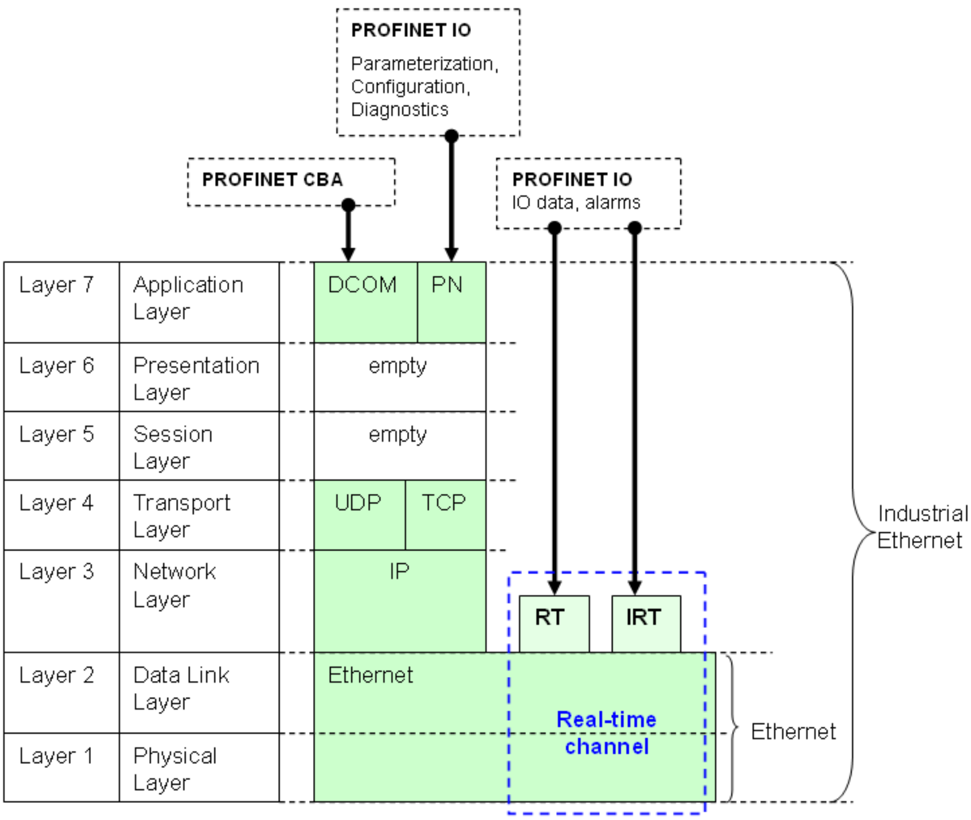
ISO/OSI reference model

The PROFINET communication is based on the Ethernet communication. It differentiates three communication channels, or three performance classes:

Table 9-1

Communication channel		Application	Examples
TCP/IP		for not time-critical communication	• acyclic reading and writing of data records • Settings • Configuration • Diagnosis
Real-time	Real Time (RT)	for time-critical communication	• cyclic process image of the field devices (IO data) • Alarms
	Isochrones Real Time (IRT)	for high-performant deterministic and clock-synchronized communication	• Process data in the area of motion control

Figure 9-5



10 PROFIBUS (PB)

Introduction

PROFIBUS is a simple, internationally standardized electrical field bus system.

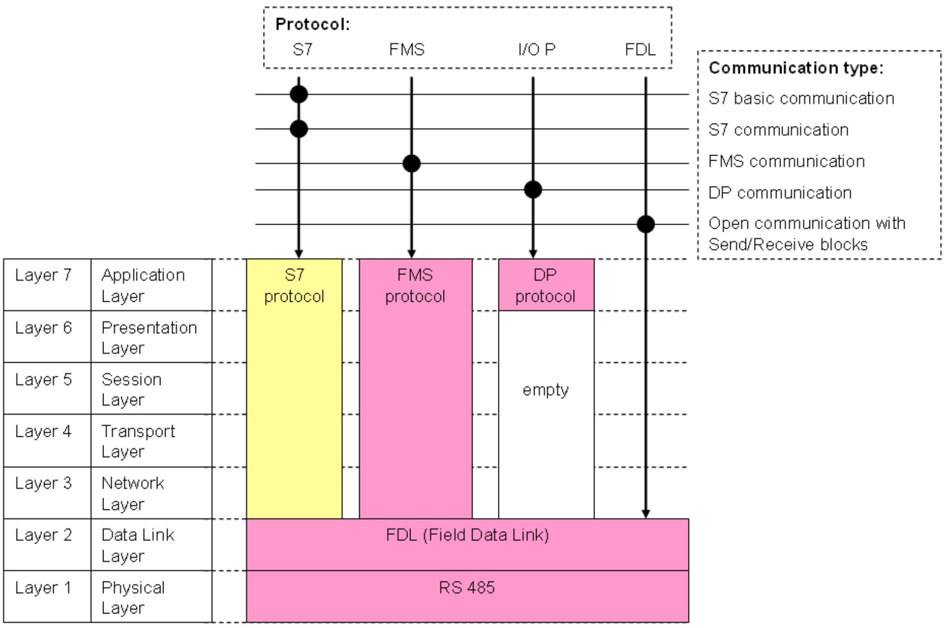
Characteristics

- International standard: IEC61158, IEC61784
- Mechanism communication between controllers:
  Token Umlauf
- Mechanism communication between field devices and controllers:
  Master/slave principle

ISO/OSI reference model

The following figure shows all protocols and communication types supported by the SIMATIC controllers.

Figure 10-1



11 MPI

Introduction

MPI is the SIMATIC network for communication with PG/OP and CPU-CPU communication

An MPI interface is integrated on almost all CPUs (*1) of the modular SIMATIC controllers.

(*1): Exception: S7-1200

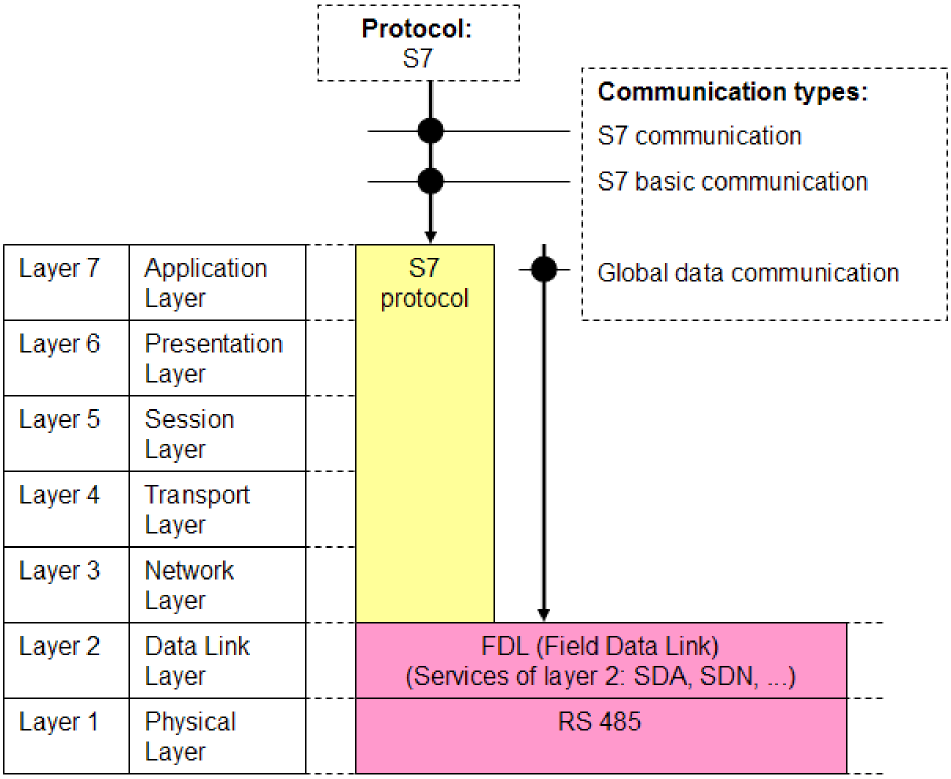
Characteristics

- Manufacturer-specific interface (no open standard)
- Network with low expansion and low number of nodes
- MPI is based on PROFIBUS

ISO/OSI reference model

The following figure shows all protocols and communication types supported by the SIMATIC controllers.

Figure 11-1



12 SIMATIC Backplane Bus

Introduction

CPU-CPU communication is possible within one SIMATIC station.

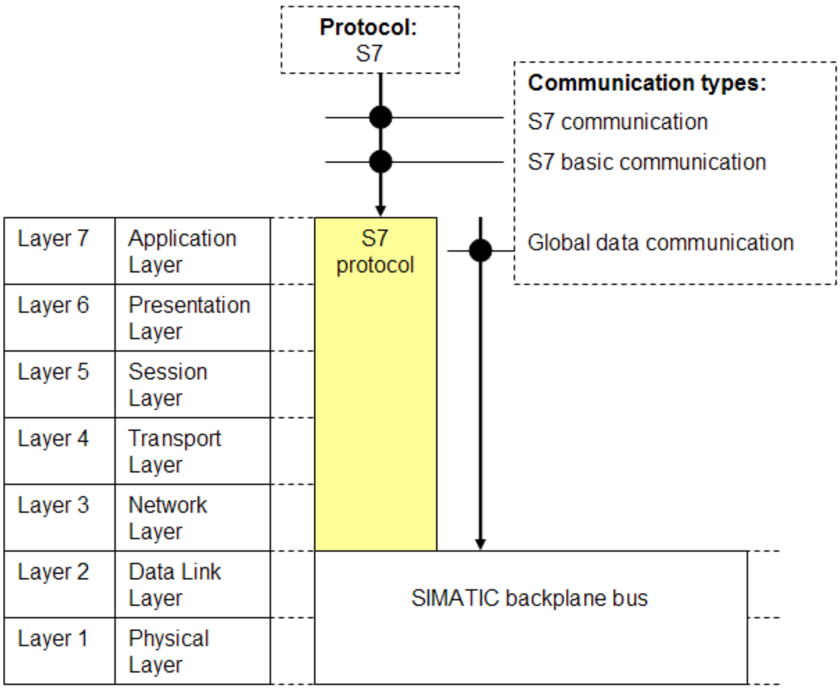
Characteristics

- Manufacturer-specific interface
- only possible for SIMATIC S7-400 (multi-computing, chapter 4.5)
- for SIMATIC-specific communication types:
  Global data communication, S7 communication

ISO/OSI reference model

The following figure shows all protocols and communication types supported by the SIMATIC controllers.

Figure 12-1



13 Serial Interface (PtP)

Introduction

The communication via a “serial interface” provides a simple option to exchange data between two communication partners.

SIMATIC controllers can communicate with different partners via the “serial interface”:

- simple devices, such as printer, barcode reader
- Drives (USS protocol, ...)
- SIMATIC controller / third-party controller

<u>Number of nodes</u>

Generally, precisely two communication partners in total are involved in the communication (point to point connection).

However, for RS 422/485 more than two communication partners are also possible (point to multipoint connection).

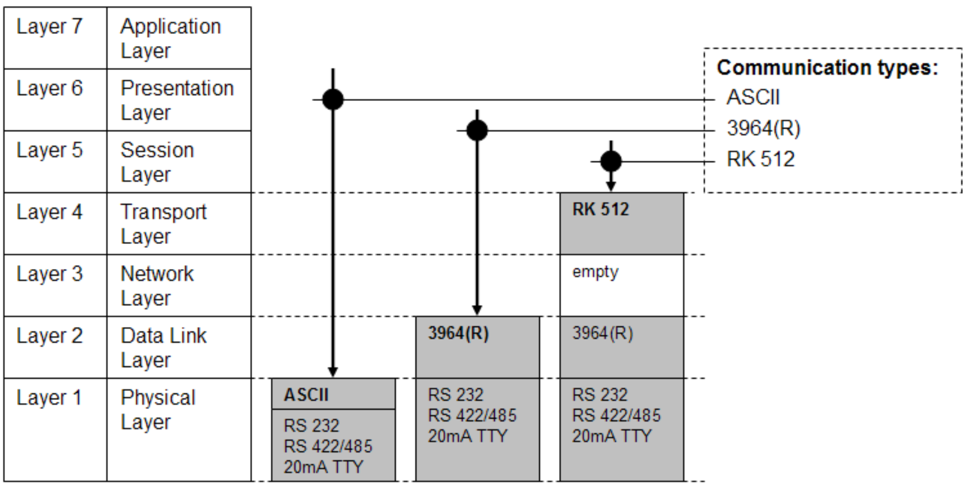
Characteristics

- Communication is mostly open
- Communication via standardized interfaces (physics):
  RS 232C, RS 422/485, 20mA TTY

ISO/OSI reference model

The following figure shows all protocols and communication types supported by the SIMATIC controllers.

Figure 13-1



14 Comparison of the Media

The following table shows how the media principally differ from each other. The comparison is a simplification. In individual cases there may be deviations from the entered values. In concrete application cases the respective manuals must therefore be consulted.

Table 14-1

Medium		Transmission rate (from … to)	maximal number of partners	maximal distance of two partners	maximal data volume per job	Routing possible?	Redundant structure possible?
Net	PROFINET/ Industrial Ethernet	PN: 10 to 100 MBit/s IE: 1 GBit/s	over 1000 (*1)	electrical: 100 m (*2) optical: 26 km (*2) Radio waves: 100 m (*2)	64 KByte	yes (not PN IO)	yes
	PROFIBUS	9.6 Kbit/s to 12 Mbit/s	126	electrical: 1 km (without repeater) or 10 km (with repeater) optical: 1875 km (with OLM)	64 Kbytes	no	yes
	MPI	187.5 Kbit/s to 12 Mbit/s	126	electrical: 50 m	64 Kbytes	no	no
SIMATIC backplane bus (only for S7-400)		---	4 CPUs in the SIMATIC station	---	64 Kbytes	---	---
Serial interface (ASCII, 3964(R), RK 512)		110 bit/s up to 115.2 Kbit/s (20mA-TTY: up to 19.2 Kbit/s)	Point-to-Point: 2 Multipoint: 32	RS232C: 15 m RS422/485: 1200 m 20mA-TTY: 1000 m	4 Kbytes	---	---

<u>Explanations for the table:</u>

(*1): the maximal number of partners in a PROFINET IO network is restricted to 512.

(*2): maximal segment length

15 Interfaces of the SIMATIC Families

The table gives an overview of which SIMATIC families can communicate via which media (CPU-CPU communication). For PROFIBUS and PROFINET/Industrial (PN/IE) it is additionally specified which functionality the interface can take.

Table 15-1

SIMATIC Controller	**SIMATIC Familie**	**MPI**	**PROFIBUS**			**PN/IE**				**SIMATIC backplane bus**	**Serial interface**
				Functionality			Functionality				
				DP master	DP slave		PROFINET IO Controller	PROFINET IO Device	PROFINET CBA		
Modular controllers	ET 200 CPU	X	X	X	X	X	X	X	X	---	X
	S7-300	X	X	X	X	X	X	X	X	---	X
	S7-400	X	X	X	X	X	X	X	X	X	X
	S7-1200	---	---	---	---	X	---	---	---	---	X
Embedded automation	S7-mEC (with WinAC RTX)	---	X	X	---	X	X	---	X	---	X
	Box PC (with WinAC RTX)	---	X	X	---	X	X	---	X	---	X
	Panel PC (with WinAC RTX)	---	X	X	---	X	X	---	X	---	X
	WinAC MP (with WinAC MP)	---	X	X	---	X	---	---	---	---	X
PC-based controller	WinAC RTX (installed on PC)	---	X	X	---	X	X	---	X	---	X

Meaning of entries into the table

“x”: interface / functionality exists

“---” interface / functionality does not exist

Note

“x” does not mean that all interfaces of a SIMATIC family have this functionality.
“x” means that there is at least one interface with this functionality.

16 Information Part 1

The table contains references to information on the topics in Part 1.

All references /x/ are stored centrally in chapter 47. There the respective links to the internet are also available.

Table 16-1

Reference	Title / content	Information on
---	STEP 7 online help: “Configuration of connections and data exchange”	Connections Connection resources Communication types Data consistency
/0/	Service & Support Portal: FAQs, Manuals (SIMATIC CPU, CP)	
/3/	SIMATIC / Communication with SIMATIC System manual	
/6/	SIMATIC system software and standard functions for S7-300/400 – reference manual	
/15/	SIMATIC / Configuring hardware and connections with STEP 7, manual	
/1/	SIMATIC Controller / The innovative solution for all automation tasks. Brochure	SIMATIC controllers
/4/	Catalog ST 70 / Products for Totally Integrated Automation and Micro Automation	
/2/	SIMATIC NET / Industrial communication Brochure	Media
/3/	SIMATIC / Communication with SIMATIC System manual	
/5/	Catalog IK PI / Industrial communication	

PART 2: Selection aids

Part 2 is the central part of the document:

Clear representation of all options for a CPU-CPU communication with SIMATIC controllers

PART 2: Structure and content

Table 16-2

Chapter	Structure	Content
17	Preliminary remarks	Explanations on the used tables
18	Jump distributor	Per medium there is one page with cross-references to the central chapters of the documentation. This makes it possible to select information quickly.
19	Selection aid PN/IE	The following is described per medium: • All interfaces (CPU, CP) and communication types per SIMATIC family (Interfaces table). • All possibilities how SIMATIC families can communicate with each other (Combinations table) • Comparison of all communication types (Communication types – compact table)
20	Selection aid PB	
21	Selection aid MPI	
22	Selection aid SIMATIC backplane bus	
23	Selection aid Serial interface	The following is described per SIMATIC family: • All interfaces (CPU, CP) and communication types (ASCII, 3964(R), …). • Properties of the communication types
24	Information	Notes regarding further information

Explanations for the table:

Medium:

- Network: PN/IE, PB, MPI
- SIMATIC backplane bus

SIMATIC family:

- Modular controller: ET 200 CPU, S7-300, S7-400, S7-1200
- Embedded Automation: S7-mEC, Box PC, Panel PC, WinAC MP
- PC-based Controller: WinAC RTX

Communication types:

- SIMATIC specific
- Open standard

17 Preliminary Remarks

The structure of the tables used in Part 2 is explained below:

- Interfaces table
- Combinations table
- Communication types – compact table

First of all it is described according to which system the numerous combination options of the SIMATIC families is represented in the document.

17.1 Overview of all combinations

17.1.1 Terms

<u>Definitions</u>

A pair:

- SIMATIC family x, SIMATIC family y

A combination:

- Controller 1 / Controller 2

One pair makes two combinations:

- Controller 1 / Controller 2: SIMATIC family x / SIMATIC family y
- Controller 1 / Controller 2: SIMATIC family y / SIMATIC family x

<u>Example</u>

Pair:

- ET 200 CPU, S7-300

Combinations:

- ET 200 CPU / S7-300
- S7-300 / ET 200 CPU

17.1.2 Symmetry of the combinations

The following table shows all combinations of SIMATIC families (independent of the medium). Each entry in the table (designated as “x.y”) makes a defined combination of two SIMATIC families.

Table 17-1

Combination 1.2

Controller 2		Controller 1								
		Modular				Embedded				PC-based
		ET 200 CPU	S7-300	S7-400	S7-1200	S7-mEC	Box PC	Panel PC	WinAC MP	WinAC RTX
Modular	ET 200 CPU	1.1	2.1	3.1	4.1	5.1	6.1	7.1	8.1	9.1
	S7-300	1.2	2.2	3.2	4.2	5.2	6.2	7.2	8.2	9.2
	S7-400	1.3	2.3	3.3	4.3	5.3	6.3	7.3	8.3	9.3
	S7-1200	1.4	2.4	3.4	4.4	5.4	6.4	7.4	8.4	9.4
Embedded	S7-mEC	1.5	2.5	3.5	4.5	5.5	6.5	7.5	8.5	9.5
	Box PC	1.6	2.6	3.6	4.6	5.6	6.6	7.6	8.6	9.6
	Panel PC	1.7	2.7	3.7	4.7	5.7	6.7	7.7	8.7	9.7
	WinAC MP	1.8	2.8	3.8	4.8	5.8	6.8	7.8	8.8	9.8
PC-based	WinAC RTX	1.9	2.9	3.9	4.9	5.9	6.9	7.9	8.9	9.9

For each pair (SIMATIC family x, SIMATIC family y) there are two combinations (entries) in the table:

- in the orange area of the table
- in the gray area of the table

<u>Example</u>

Pair:

- ET 200 CPU, S7-300

Combinations 1.2:

- ET 200 CPU / S7-300

Combinations 2.1:

- S7-300 / ET 200 CPU

The following chapter shows how this symmetry is considered in the document.

17.1.3 Realization in the document

For each pair of two SIMATIC families (x,y) there are two chapters in the document:

- one chapter for the combination x/y (chapter x/y)
- one chapter for the combination y/x (chapter y/x)

The description of the possible communication types for one pair is of course only given in one chapter (chapter x/y). The other chapter (chapter y/x) only contains a reference to this chapter (chapter x/y).

This procedure effects the structure of the document and the representation of the combinations (jump distributor).

Structure of the document (table of contents)

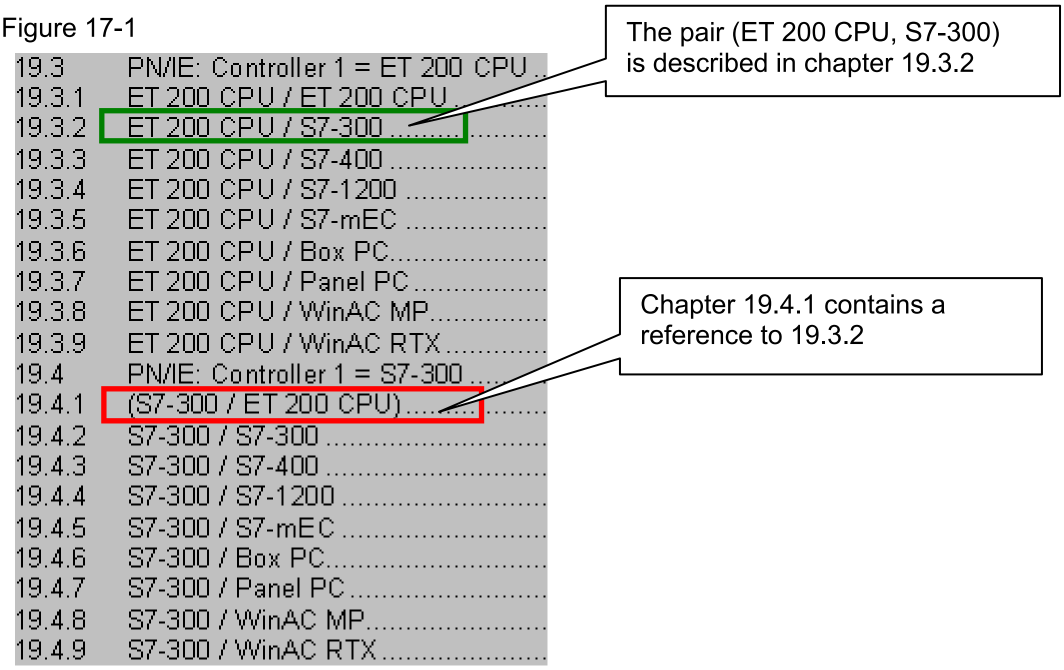
If one chapter (chapter y/x) only contains a reference to another chapter (chapter x/y), then the header of this chapter (chapter y/x) is given in brackets.

<u>Example</u>

Discussing the pair (ET 200 CPU, S7-300) for medium PN/IE.

The following figure shows an extract from the table of contents.

Figure 17-1


19.3 PN/IE: Controller 1 = ET 200 CPU ..
19.3.1 ET 200 CPU / ET 200 CPU ....
19.3.2 ET 200 CPU / S7-300 ....
19.3.3 ET 200 CPU / S7-400 ....
19.3.4 ET 200 CPU / S7-1200 ....
19.3.5 ET 200 CPU / S7-mEC ....
19.3.6 ET 200 CPU / Box PC...................
19.3.7 ET 200 CPU / Panel PC ....
19.3.8 ET 200 CPU / WinAC MP..............
19.3.9 ET 200 CPU / WinAC RTX............
19.4 PN/IE: Controller 1 = S7-300 ....
19.4.1 (S7-300 / ET 200 CPU)....................
19.4.2 S7-300 / S7-300 ....
19.4.3 S7-300 / S7-400 ....
19.4.4 S7-300 / S7-1200 ....
19.4.5 S7-300 / S7-mEC ....
19.4.6 S7-300 / Box PC..........................
19.4.7 S7-300 / Panel PC ....
19.4.8 S7-300 / WinAC MP......................
19.4.9 S7-300 / WinAC RTX ....

Combinations (jump distributor)

In the document the combinations are represented in tables (jump distributor). For each combination the chapter is entered in which this combination is described specifically.

Example

Discussing the pair (ET 200 CPU, S7-300) for medium PN/IE.

The figure below shows the jump distributor for PN/IE.

Figure 17-2

Controller 2		Controller 1								
		Modular				Embedded				PC-based
		ET 200 CPU	S7-300	S7-400	S7-1200	S7-mEC	Box PC	Panel PC	WinAC MP	WinAC RTX
Modular	ET 200 CPU	19.3.1	19.3.2	19.3.3	19.3.4	19.3.5	19.3.6	19.3.7	19.3.8	19.3.9
	S7-300	19.3.2	19.4.2	19.4.3	19.4.4	19.4.5	19.4.6	19.4.7	19.4.8	19.4.9
	S7-400	19.3.3	19.4.3	19.5.3	19.5.4	19.5.5	19.5.6	19.5.7	19.5.8	19.5.9
	S7-1200	19.3.4	19.4.4	19.5.4	19.6.4	19.6.5	19.6.6	19.6.7	19.6.8	19.6.9
Embedded	S7-mEC	19.3.5	19.4.5	19.5.5	19.6.5	19.7.5	19.7.6	19.7.7	19.7.8	19.7.9
	Box PC	19.3.6	19.4.6	19.5.6	19.6.6	19.7.6	19.8.6	19.8.7	19.8.8	19.8.9
	Panel PC	19.3.7	19.4.7	19.5.7	19.6.7	19.7.7	19.8.7	19.9.7	19.9.8	19.9.9
	WinAC MP	19.3.8	19.4.8	19.5.8	19.6.8	19.7.8	19.8.8	19.9.8	19.10.8	19.10.9
PC-based	WinAC RTX	19.3.9	19.4.9	19.5.9	19.6.9	19.7.9	19.8.9	19.9.9	19.10.9	19.11.9

For the pair there are two entries (Controller 1 / Controller 2) in table:

- ET 200 CPU / S7-300 (green frame)
- S7-300 / ET 200 CPU (red frame)

Both entries contain identical chapter numbers (19.3.2).

17.2 Interfaces table

17.2.1 Purpose of this table

The Interfaces table contains a compact representation of the interfaces provided by the SIMATIC families. For each medium and each SIMATIC family there is one table. This table is the basis for the Combinations tables (chapter 17.3).

The table answers the following questions:

- Which tables (CPU, CP) are available?
- Which communication types are possible?
- What are the characteristic features?

17.2.2 Structure of the table

The structure of the tables is described in greater detail below.

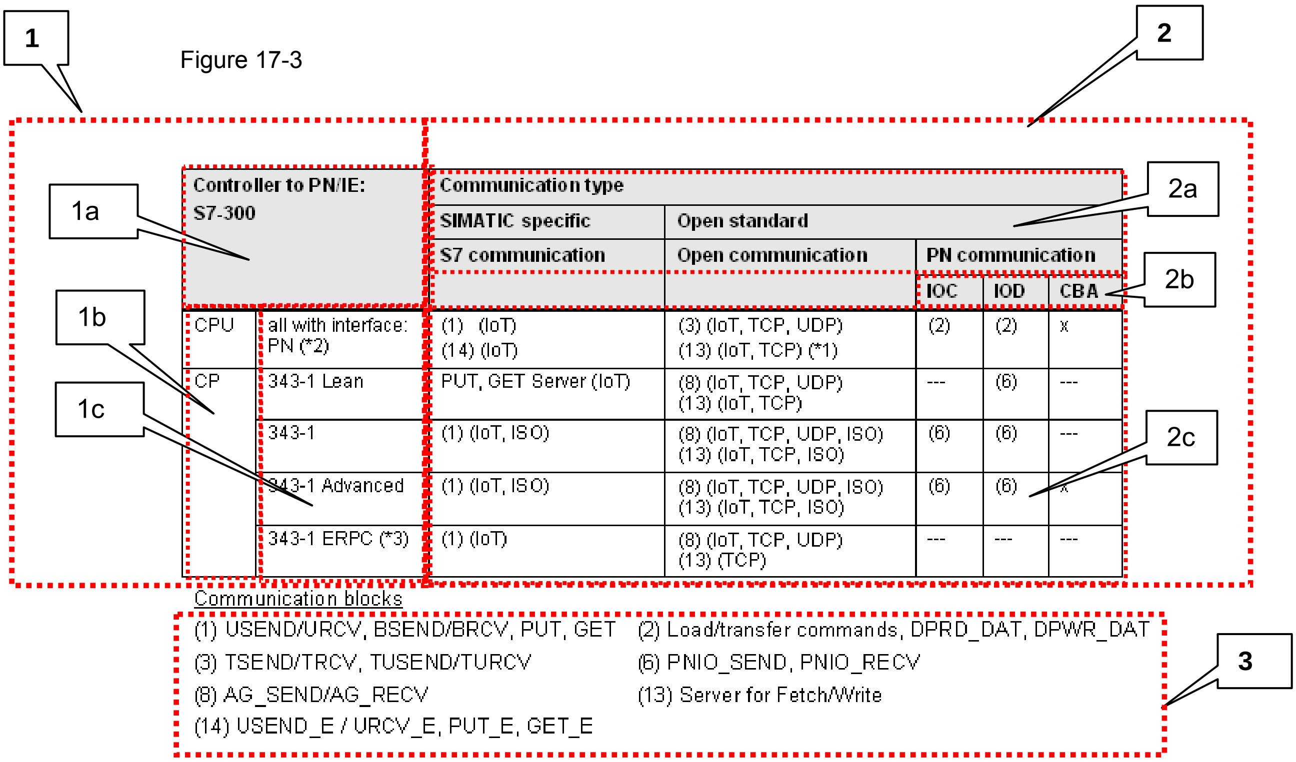
Medium PN/IE

The structure is explained by means of a direct example:

- S7-300 an PN/IE

The figure shows the respective Interfaces table (Table 19-2).

Figure 17-3



Controller to PN/IE: S7-300		Communication type				
		SIMATIC specific	Open standard			
		S7 communication	Open communication	PN communication		
				IOC	IOD	CBA
CPU	all with interface: PN (*2)	(1) (IoT) (14) (IoT)	(3) (IoT, TCP, UDP) (13) (IoT, TCP) (*1)	(2)	(2)	x
CP	343-1 Lean	PUT, GET Server (IoT)	(8) (IoT, TCP, UDP) (13) (IoT, TCP)	---	(6)	---
	343-1	(1) (IoT, ISO)	(8) (IoT, TCP, UDP, ISO) (13) (IoT, TCP, ISO)	(6)	(6)	---
	343-1 Advanced	(1) (IoT, ISO)	(8) (IoT, TCP, UDP, ISO) (13) (IoT, TCP, ISO)	(6)	(6)	x
	343-1 ERPC (*3)	(1) (IoT)	(8) (IoT, TCP, UDP) (13) (TCP)	---	---	---

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(6) PNIO_SEND, PNIO_RECV
(8) AG_SEND/AG_RECV
(13) Server for Fetch/Write
(14) USEND_E / URCV_E, PUT_E, GET_E

The table consists of several areas, which are explained below.

Area 1: controller interfaces

Here the properties of the SIMATIC family are described:

Table 17-2

	Description	Example
1a	Designation of the medium and the SIMATIC family	S7-300 at PN/IE
1b	Type of interface: Integrated interface of a CPU or interface of a CP	There are both variants: CPU and CP
1c	Designation of the modules	CPU: see figure CP: see figure

Area 2: communication types

Here the communication options are described.

Table 17-3

	Description	Example
2a	Designation of the communication types, divided into the classes “SIMATIC-specific” and “Open standard”.	S7 communication Open communication PN communication
2b	Functionality of the interface	PROFINET IO Controller, PROFINET IO Device, PROFINET CBA
2c	Here you enter the communication options for the modules. The possible communication blocks are either entered directly, or represented via an index. The index is explained in area 3.	direct entry: • PUT, GET Server (IoT) Index: • (1), (2), ...
	The possible protocols are given in brackets. If a controller can only be used as server, then this is also entered.	CP 343-1 Lean can communicate via PUT and GET. The module is the server. The possible protocols are IoT and ISO:
	Meaning of the entries: --- communication is not possible x communication is possible	

Area 3: communication blocks for the index (x)

Here the K-blocks possible for an index are listed.

Medium PB

The structure is explained by means of a direct example:

- S7-300 at PB

The figure shows an extract from the Interfaces table (Table 20-2).

2b

Figure 17-4

Controller to PB: S7-300		Functionality of the interface		Communication type	
				SIMATIC-specific	
				S7 basic communication	S7 communication
CPU	all with interface: DP, MPI/DP (*2)	DP master		I_PUT, I_GET, **Client** (S7)	PUT, GET, Server (S7)
		DP slave (*3)	active	I_PUT, I_GET, Server (S7)	PUT, GET, Server (S7)
			passive	I_PUT, I_GET, Server (S7)	---
CP	342-5 (*1)	no DP operation		---	(1) (S7)
		DP master		---	(1) (S7)
		DP slave (*3)	active	---	PUT, GET, Server (S7)
			passive	---	---
	343-5	no DP operation		---	PUT, GET, Server (S7)

The structure of the table is the same as for medium PN/IE (see above), with one exception:

- Functionality of the interface (area 2b)

<u>Area 4: functionality of the interface</u>

The functionalities of the interface are given here. There are various communication types depending on the functionality.

Area 2b of the table for PB corresponds to the area 2b of the table for PN/IE.

17.2.3 Abbreviations and indices

An overview of the used abbreviations and indices is given in chapter 17.5.

17.3 Combinations table

17.3.1 Purpose of this table

The Combinations table contains a compact representation of how SIMATIC controllers can communicate with each other. For each medium and each combination of two SIMATIC families there is one table in the document.

The table answers the following questions:

- Which tables (CPU, CP) are available?
- Which communication types are possible?
- What characteristic features are there regarding the communication?

17.3.2 Structure of the table

The structure of the table is described by means of a simplified (not real) example.

Boundary conditions:

- Controller 1 is a SIMATIC family with a CPU interface (CPU11) and two CP interfaces (CP11, CP12)
- Controller 2 is a SIMATIC family with a CPU interface (CPU21) and two CP interfaces (CP21, CP22)
- It is assumed that two communication types are possible:
  S7 communication (S7) and open communication (OC)

Under the above boundary conditions the Combinations table looks like this:

Area 2
Column 1
Area 1
Row 1
Row 4
Area 3
Row 1
Column 1
Area 4
Column 3
Area 5

Table 17-4

Controller 2: S7-400			**Controller 1: S7-300 an PN/IE**					
			CPU		CP			
			CPU11		CP11		CP12	
			IOC, IOD		IOD		IOD	
			S7	**OC**	**S7**	**OC**	**S7**	**OC**
CPU	CPU21	IOC	(1)	(3)	(21)	(8) / (3)	(1)	(8) / (3)
CP	CP21	IOD	(1)	(3) / (8)	(21)	(8)	(1)	(8)
	CP22	IOD	(1)	(3) / (8)	(21)	(8)	(1)	(8)

Communication blocks
(1) USEND/URCV, BSEND/BRCV, PUT, GET
(3) TSEND/TRCV, TUSEND/TURCV
(8) AG_SEND/AG_RECV
(21) Controller 1 is server (for PUT, GET)

An explanation of the individual areas of the table is given below.

Overview

The table shows all combinations of the modules of controller 1 (area 1) and controller 2 (area 2):

- CPU11 with CPU21, CP21, CP22
- CP11 with CPU21, CP21, CP22
- CP12 with CPU21, CP21, CP22

For each combination it is specified in area 4 which communication types (area 3) are possible.

The required communication blocks are listed in area 5.

Area 1 (yellow)

Area 1 shows all communication-capable modules of controller 1.

One column represents one module. Modules with the same behavior are combined.

The modules are grouped according to the type of the interface (row 2):

- CPU interfaces: CPU11 (row 3)
- CP interfaces: CP11, CP12 (row 3)

The functionality of the interface has been entered in row 4:

- CPU11: IOC, IOD (PROFINET IO Controller, PROFINET IO Device)
- CP11: IOD (PROFINET IO Device)

Area 2 (blue)

Area 2 shows all communication-capable modules of controller 2.

One row represents one module. Modules with the same behavior are combined.

The modules are grouped according to the type of the interface (column 1):

- CPU interfaces: CPU21 (column 2)
- CP interfaces: CP21, CP22 (column 2)

The functionality of the interface has been entered in column 3:

- CPU21: IOC (PROFINET IO Controller)
- CP21: IOD (PROFINET IO Device)

Area 3 (gray)

Area 3 lists the communication types principally possible for both controllers. They are repeated for each module (column) of controller 1 (area 1):

- S7 (S7 communication)
- OC (open communication)

Area 4 (white)

Each line in area 4 represents a combination of two modules. Simultaneously each cell is assigned to a communication type (area 3).

Area 4 provides the following information per combination:

- possible communication types (area 3)
- respective communication blocks, represented via an index ((x))

The entries in the cells have the following meaning:

Table 17-5

Entry in cell	Communication type according to area 3 possible?	Which communication blocks can be used?
(1)	yes	See area 5: Index (1)
(1)+(2)	yes	See area 5: (1) and (2)
(1) / (2)	yes	Controller 1 with (1) / Controller 2 with (2)
(1) / (2)+(3)	yes	Two cases are possible: • Controller 1 with (1) / Controller 2 with (2) • Controller 1 with (1) / Controller 2 with (3)
x	yes	Applies only for CBA (there no communication blocks are required).
---	no	N/A

Area 5 (communication blocks)

In area 5 the indices (x) from area 4 are explained. It is specified for each index which communication blocks are possible and which characteristics must be observed.

Summary

Each cell in area 4 provides the following information:

- Module x (area 1) can communicate with module y (area 2) via the communication type z (area 3).
- The possible communication blocks are described in area 5.

Example

The red cells in Table 17-4 mean:

Combination:

- CPU11 / CP21

Communication type:

- Open communication

Communication blocks:

- For controller 1: TSEND/TRCV, TUSEND/TURCV (T-blocks)
- For controller 2: AG_SEND/AG_RECV (Send/Receive blocks)

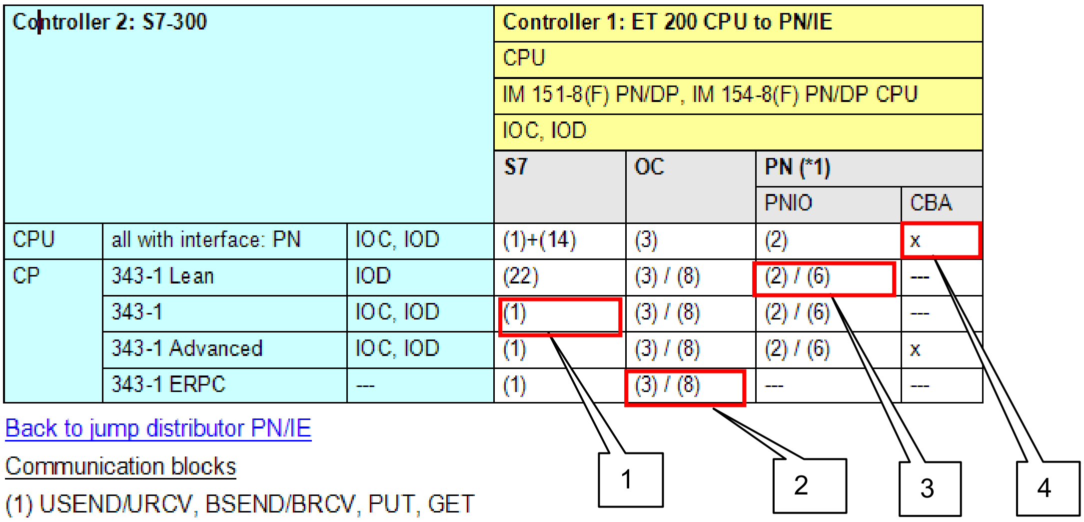
17.3.3 Real example

The use of the table is now explained by means of a real example:

- Communication via PN/IE
- Controller 1: ET 200 CPU
- Controller 2: S7-300

The figure shows the respective Combinations table (Table 19-14).

Figure 17-5



Controller 2: S7-300			Controller 1: ET 200 CPU to PN/IE			
			CPU			
			IM 151-8(F) PN/DP, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	OC	PN (*1)	
					PNIO	CBA
CPU	all with interface: PN	IOC, IOD	(1)+(14)	(3)	(2)	x
CP	343-1 Lean	IOD	(22)	(3) / (8)	(2) / (6)	---
	343-1	IOC, IOD	(1)	(3) / (8)	(2) / (6)	---
	343-1 Advanced	IOC, IOD	(1)	(3) / (8)	(2) / (6)	x
	343-1 ERPC	---	(1)	(3) / (8)	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(6) PNIO_SEND, PNIO_RECV
(8) AG_SEND/AG_RECV
(14) USEND_E/URCV_E, PUT_E, GET_E
(22) Controller 2 is server (for PUT, GET)

Examples (red frames in the above table):

Table 17-6

	Controller 1		Controller 2		Communication type	Communication blocks (Index)
	Interface	Functionality	Interface	Functionality		
1	IM151-8, IM154-8	IOC, IOD	343- 1	IOC, IOD	S7	Both controllers: (1)
2	IM151-8, IM154-8	IOC, IOD	343-1 ERPC	---	OC	Controller 1: (3) Controller 2: (8)
3	IM151-8, IM154-8	IOC, IOD	343-1 Lean	IOD	PN: PNIO	Controller 1: (2) Controller 2: (6)
4	IM151-8, IM154-8	IOC, IOD	S7-300 CPU PN	IOC, IOD	PN: CBA	---

17.3.4 Abbreviations and indices

An overview of abbreviations and indices is given in chapter 17.5.

17.4 Communication types – compact table

17.4.1 Purpose of this table

The Communication types – compact table answers the following questions for each medium (PN/IE, PB, MPI):

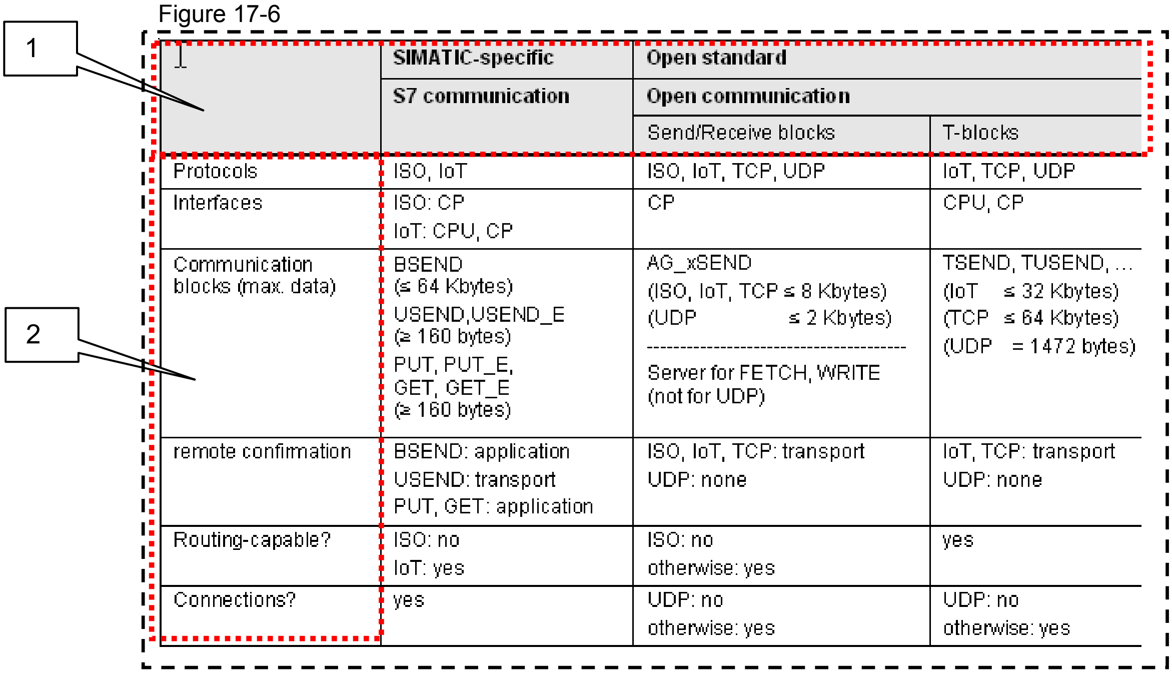
- Which communication types exist per medium?
- What are the main differences?

The Communication types – compact table is a summary (compact version) of the Communication types – detailed tables. Not each SIMATIC family or module fulfills all values given in the Communication types – compact table. The values must be considered as supersets.

17.4.2 Structure of the table

Example: extract from the table for PN/IE (Table 19-58).

Figure 17-6



	SIMATIC-specific	Open standard	
	S7 communication	Open communication	
		Send/Receive blocks	T-blocks
Protocols	ISO, IoT	ISO, IoT, TCP, UDP	IoT, TCP, UDP
Interfaces	ISO: CP IoT: CPU, CP	CP	CPU, CP
Communication blocks (max. data)	BSEND (≤ 64 Kbytes) USEND,USEND_E (≥ 160 bytes) PUT, PUT_E, GET, GET_E (≥ 160 bytes)	AG_xSEND (ISO, IoT, TCP ≤ 8 Kbytes) (UDP ≤ 2 Kbytes) -------------------------------- Server for FETCH, WRITE (not for UDP)	TSEND, TUSEND, ... (IoT ≤ 32 Kbytes) (TCP ≤ 64 Kbytes) (UDP = 1472 bytes)
remote confirmation	BSEND: application USEND: transport PUT, GET: application	ISO, IoT, TCP: transport UDP: none	IoT, TCP: transport UDP: none
Routing-capable?	ISO: no IoT: yes	ISO: no otherwise: yes	yes
Connections?	yes	UDP: no otherwise: yes	UDP: no otherwise: yes

The table consists of several areas, which are explained below.

Area 1: communication types

Designation of the communication types, divided into the classes “SIMATIC-specific” and “Open standard”.

Area 2: criteria

The criteria are described in chapter 25.2.3.

<u>Note</u>

The criteria of the Communication types – compact table are a subset of the Communication types – detailed tables.

17.5 Overview of abbreviations and indices

17.5.1 Abbreviations

The following abbreviations are used in the Interfaces table and the Combinations table:

Table 17-7

Abbreviation	Description
CBA	PROFINET CBA (Component Based Automation)
Cx	Controller x
Cx Server	Controller x is the server
DP	DP communication
GD	Global data communication
IOC	IO controller
IOD	IO device
Master	DP master
OC	Open communication “Open communication” stands for both communication types (*1): • open communication with T blocks • open communication with send/receive blocks
PN	PN communication
PNIO	PROFINET IO
S/R	Send/Receive
S7	S7 communication
S7 Basis S7 B	S7 basic communication
Slave	DP slave

(*1): comparison

Table 17-8

	Open communication	
	Open communication with T blocks	**Open communication with send/receive blocks**
Medium	PN/IE	PN/IE, PB
Communication blocks	TSEND / TRCV TUSEND / TURCV	AG_SEND / AG_RECV AG_SSEND / AG_SRECV AG_LSEND / AG_LRECV
Interface	CPU, CP	CP
Communication with third-party controllers	yes	yes

17.5.2 Overview of the indices

The following indices are used in the Interfaces table and the Combinations table:

Table 17-9

(x)	Available communication blocks
(1)	USEND/URCV, BSEND/BRCV, PUT, GET
(2)	Load/transfer commands, DPRD_DAT, DPWR_DAT
(3)	TSEND/TRCV, TUSEND/TURCV
(4)	TSEND/TRCV
(5)	TSEND_C/TRCV_C
(6)	PNIO_SEND, PNIO_RECV
(7)	DP_SEND, DP_RECV
(8)	AG_SEND/AG_RECV
(9)	AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV
(10)	READ, WRITE, REPORT
(11)	X_SEND/X_RCV, X_PUT, X_GET
(12)	GD_SND/GD_RCV
(13)	Server for FETCH/WRITE
(14)	USEND_E/URCV_E, PUT_E, GET_E

The following indices are only used in the Combinations table:

Table 17-10

(x)	Available communication blocks
S7 communication	
(21)	Controller 1 is the server (for PUT, GET)
(22)	Controller 2 is the server (for PUT, GET)
(31)	Controller 1 is the server (for PUT_E, GET_E)
(32)	Controller 2 is the server (for PUT_E, GET_E)
S7 basic communication	
(61)	Controller 1 is the server (for I_PUT, I_GET)
(62)	Controller 2 is the server (for I_PUT, I_GET)

18 ********** Jump Distributor **********

18.1 Network PN/IE

Overview of interfaces and communication types

Table 18-1

Family		Chapter
Modular controllers	ET 200 CPU	19.2.1
	S7-300	19.2.2
	S7-400	19.2.3
	S7-1200	19.2.4
Embedded controllers	S7-mEC	19.2.5
	Box PC	19.2.6
	Panel PC	19.2.7
	WinAC MP	19.2.8
PC-based controller	WinAC RTX	19.2.9

Combination controller 1 / controller 2

Table 18-2

Controller 2		Controller 1								
		Modular				Embedded				PC-based
		ET 200 CPU	S7-300	S7-400	S7-1200	S7-mEC	Box PC	Panel PC	WinAC MP	WinAC RTX
Modular	ET 200 CPU	19.3.1	19.3.2	19.3.3	19.3.4	19.3.5	19.3.6	19.3.7	19.3.8	19.3.9
	S7-300	19.3.2	19.4.2	19.4.3	19.4.4	19.4.5	19.4.6	19.4.7	19.4.8	19.4.9
	S7-400	19.3.3	19.4.3	19.5.3	19.5.4	19.5.5	19.5.6	19.5.7	19.5.8	19.5.9
	S7-1200	19.3.4	19.4.4	19.5.4	19.6.4	19.6.5	19.6.6	19.6.7	19.6.8	19.6.9
Embedded	S7-mEC	19.3.5	19.4.5	19.5.5	19.6.5	19.7.5	19.7.6	19.7.7	19.7.8	19.7.9
	Box PC	19.3.6	19.4.6	19.5.6	19.6.6	19.7.6	19.8.6	19.8.7	19.8.8	19.8.9
	Panel PC	19.3.7	19.4.7	19.5.7	19.6.7	19.7.7	19.8.7	19.9.7	19.9.8	19.9.9
	WinAC MP	19.3.8	19.4.8	19.5.8	19.6.8	19.7.8	19.8.8	19.9.8	19.10.8	19.10.9
PC-based	WinAC RTX	19.3.9	19.4.9	19.5.9	19.6.9	19.7.9	19.8.9	19.9.9	19.10.9	19.11.9

Communication types

Table 18-3

Communication type		Chapter
Communication types (compact table)		19.12
Table with details	S7 communication	29.2
	Open communication with send/receive blocks	31.2
	Open communication with T blocks	32.2
	PNIO	34.2
Modbus/TCP (SIMATIC / third-party controller)		44

18.2 Network PB

Overview of interfaces and communication types

Table 18-4

Family		Chapter
Modular controllers	ET 200 CPU	20.2.1
	S7-300	20.2.2
	S7-400	20.2.3
	S7-1200	20.2.4
Embedded controllers	S7-mEC	20.2.5
	Box PC	20.2.6
	Panel PC	20.2.7
	WinAC MP	20.2.8
PC-based controller	WinAC RTX	20.2.9

Combination controller 1 / controller 2

Table 18-5

Controller 2		Controller 1								
		Modular				Embedded				PC-based
		ET 200 CPU	S7-300	S7-400	S7-1200	S7-mEC	Box PC	Panel PC	WinAC MP	WinAC RTX
Modular	ET 200 CPU	20.3.1	20.3.2	20.3.3	---	20.3.4	20.3.5	20.3.6	20.3.7	20.3.8
	S7-300	20.3.2	20.4.2	20.4.3	---	20.4.4	20.4.5	20.4.6	20.4.7	20.4.8
	S7-400	20.3.3	20.4.3	20.5.3	---	20.5.4	20.5.5	20.5.6	20.5.7	20.5.8
	S7-1200	---	---	---	---	---	---	---	---	---
Embedded	S7-mEC	20.3.4	20.4.4	20.5.4	---	20.6.4	20.6.5	20.6.6	20.6.7	20.6.8
	Box PC	20.3.5	20.4.5	20.5.5	---	20.6.5	20.7.5	20.7.6	20.7.7	20.7.8
	Panel PC	20.3.6	20.4.6	20.5.6	---	20.6.6	20.7.6	20.8.6	20.8.7	20.8.8
	WinAC MP	20.3.7	20.4.7	20.5.7	---	20.6.7	20.7.7	20.8.7	20.9.7	20.9.8
PC-based	WinAC RTX	20.3.8	20.4.8	20.5.8	---	20.6.8	20.7.8	20.8.8	20.9.8	20.10.8

Communication types

Table 18-6

Communication type		Chapter
Communication types (compact table)		20.11
Table with details	S7 basic communication	28.2
	S7 communication	29.2
	Open communication with send/receive blocks	36.2
	FMS communication	37.2
	DP communication	38.2

18.3 Network MPI

Overview of interfaces and communication types

Table 18-7

Family		Chapter
Modular controllers	ET 200 CPU	21.2.1
	S7-300	21.2.2
	S7-400	21.2.3

Combination controller 1 / controller 2

Table 18-8

Controller 2:		Controller 1								
		Modular				Embedded				PC-based
		ET 200 CPU	S7-300	S7-400	S7-1200	S7-mEC	Box PC	Panel PC	WinAC MP	WinAC RTX
Modular	ET 200 CPU	21.3.1	21.3.2	21.3.3	---	---	---	---	---	---
	S7-300	21.3.2	21.4.2	21.4.3	---	---	---	---	---	---
	S7-400	21.3.3	21.4.3	21.5.3	---	---	---	---	---	---
	S7-1200	---	---	---	---	---	---	---	---	---
Embedded	S7-mEC	---	---	---	---	---	---	---	---	---
	Box PC	---	---	---	---	---	---	---	---	---
	Panel PC	---	---	---	---	---	---	---	---	---
	WinAC MP	---	---	---	---	---	---	---	---	---
PC-based	WinAC RTX	---	---	---	---	---	---	---	---	---

Communication types

Table 18-9

Communication type		Chapter
Communication types (compact table)		21.6
Table with details	Global data	27.2
	S7 basic communication	28.2
	S7 communication	29.2

18.4 SIMATIC backplane bus

Overview of interfaces and communication types

Table 18-10

Family		Chapter
Modular controllers	S7-400	22.2

Combination controller 1 / controller 2

Table 18-11

Controller	Controller		
	Modular	Embedded	PC-based
Modular	22.3	---	---
Embedded	---	---	---
PC-based	---	---	---

Communication types

Table 18-12

Communication type		Chapter
Communication types (compact table)		22.4
Table with details	Global data	27.2
	S7 basic communication	28.2
	S7 communication	29.2

18.5 Serial interfaces

Overview of interfaces and communication types

Table 18-13

Family		Chapter
Modular controllers	ET 200 CPU	23.2
	S7-300	23.3
	S7-400	23.4
	S7-1200	23.5
Embedded controllers	S7-mEC	23.6
	Box PC	23.7
	Panel PC	23.8
	WinAC MP	23.9
PC-based controller	WinAC RTX	23.10

Communication types

Table 18-14

Communication type	Chapter
3964(R), ASCII, RK 512	40
User-defined protocol	41
Modbus serial (RTU format)	45

19 Selection Aid: PROFINET/Industrial Ethernet (PN/IE)

19.1 PN/IE: Content of the chapter

The following is described for the PN/IE medium:

- which interfaces (modules) and communication types are available?
  (-> Interfaces table)
- which partners can communicate via which communication types?
  (-> Combinations table)
- overview of all available communication types
  (-> Communication types – compact table)

19.2 PN/IE: Interfaces and communication types

19.2.1 ET 200 CPU to PN/IE

Table 19-1

Controller to PN/IE: ET 200 CPU			**Communication types**				
			SIMATIC specific	**Open standard**			
			S7 communication	**Open communication**	**PN communication**		
					IOC	**IOD**	**CBA**
ET 200S	CPU	IM 151-8(F) PN/DP CPU	(1) (IoT) (14) (IoT)	(3) (IoT, TCP, UDP)	(2)	(2)	x
ET 200Pro	CPU	IM 154-8(F) PN/DP CPU	(1) (IoT) (14) (IoT)	(3) (IoT, TCP, UDP)	(2)	(2)	x

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(14) USEND_E/URCV_E, PUT_E, GET_E

19.2.2 S7-300 to PN/IE

Table 19-21

Controller to PN/IE: S7-300		**Communication type**				
		SIMATIC specific	**Open standard**			
		S7 communication	**Open communication**	**PN communication**		
				IOC	**IOD**	**CBA**
CPU	all with interface: PN (*2)	(1) (IoT) (14) (IoT)	(3) (IoT, TCP, UDP) (13) (IoT, TCP) (*1)	(2)	(2) (*4)	x
CP	343-1 Lean	PUT, GET Server (IoT)	(8) (IoT, TCP, UDP) (13) (IoT, TCP)	---	(6)	---
	343-1	(1) (IoT, ISO)	(8) (IoT, TCP, UDP, ISO) (13) (IoT, TCP, ISO)	(6)	(6)	---
	343-1 Advanced	(1) (IoT, ISO)	(8) (IoT, TCP, UDP, ISO) (13) (IoT, TCP, ISO)	(6)	(6)	x
	343-1 ERPC (*3)	(1) (IoT)	(8) (IoT, TCP, UDP) (13) (TCP)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(6) PNIO_SEND, PNIO_RECV
(8) AG_SEND/AG_RECV
(13) Server for Fetch/Write
(14) USEND_E / URCV_E, PUT_E, GET_E

Explanations for the table:

(*1): Communication blocks: FW_TCP and FW_IOT (use the T-blocks internally). Contained in STEP 7 (standard library).

(*2): CPU 315(F)-2 PN/DP, CPU 317(F)-2 PN/DP, CPU 319(F)-3 PN/DP

(*3): The CP supports the ERPC communication. Since this is not subject of the documentation no entry is made in the table.

(*4) IO controller as I-device

19.2.3 S7-400 to PN/IE

Table 19-3

Controller to PN/IE: S7-400		Communication type				
		SIMATIC specific	Open standard			
		S7 communication	Open communication	PN communication		
				IOC	IOD	CBA
CPU	all with interface: PN (*2)	(1) (IoT)	(3) (IoT, TCP, UDP) (13) (IoT, TCP) (*1)	(2)	(2)	x
CP	443-1	(1) (IoT, ISO)	(4) (IoT) (9) (ISO, IoT, TCP, UDP) (13) (ISO, IoT, TCP)	(2)	(2) (*3)	---
	443-1 Adv.	(1) (IoT, ISO)	(4) (IoT) (9) (ISO, IoT, TCP, UDP) (13) (ISO, IoT, TCP)	(2)	(2) (*3)	x

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

(13) Server for Fetch/Write

Explanations on the table

(*1): Communication blocks: FW_TCP and FW_IOT (use the T-blocks internally)

(*2): CPU 412-2 PN, CPU 414(F)-3 PN/DP, CPU 416(F)-3 PN/DP

(*3) IO controller as I-device

19.2.4 S7-1200 to PN/IE

Table 19-4

Controller to PN/IE: S7-1200		**Communication type**				
		SIMATIC-specific	**Open standard**			
		S7 communication	**Open communication**	**PN communication**		
				IOC	**IOD**	**CBA**
CPU	1211C 1212C 1214C	PUT, GET Server (IoT)	(4) (IoT, TCP) (5) (IoT, TCP)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

19.2.5 S7-mEC to PN/IE

Table 19-5

Controller to PN/IE: S7-mEC			**Communication type**				
			SIMATIC-specific	**Open standard**			
			S7 communication	**Open communication**	**PN communication**		
					IOC	**IOD**	**CBA**
CPU	EC31 (*1)	under the control of RTX (submodules) (*2): X1	(1) (IoT) (14) (IoT)	(3) (TCP, UDP)	(2)	---	x
		under the control of Windows (*3): X2	(1) (IoT) (14) (IoT)	---	---	---	---
CP	EM PC	under the control of Windows (*3): X1	(1) (IoT)	---	---	---	---
	EM PCI-104 (*4)	---	---	---	---	---	---
	CPs aus S7-300	---	---	---	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(14) USEND_E / URCV_E, PUT_E, GET_E

Explanations on the table

(*1): with preinstalled WinAC RTX (F) 2009

(*2): Interface, under the control of WinAC RTX. (F). The interface must be configured as “submodule”:

(*3): Interfaces under the control of Windows. Both interfaces cannot be operated at the same time (EC31 EXOR EM PC). The interface at the EM PC is a “Gigabit Ethernet interface”.

(*4): a maximum of 3 cards of the PCI-104 standard can be plugged

19.2.6 Box PC to PN/IE

Table 19-6

Controller to PN/IE: Box PC (*1) IPC427C bundles with RTX		**Communication type**				
		SIMATIC-specific	**Open standard**			
		S7 communication	**Open communication**	**PN communication**		
				IOC	**IOD**	**CBA**
CPU / CP (*2)	under the control of RTX (submodules) (*3): • “CP1616-CP1604”	(1) (IoT)	(3) (TCP, UDP)	(2)	---	x
	under the control of Windows (*4): • “IE General”	(1) (IoT)	---	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

<u>Explanations on the table</u>

(*1): with preinstalled WinAC RTX (F) 2009 (optional)

(*2): meaning of the interface CPU or CP: CPU = onboard interface of the PC, CP = plug-in PC card

The designations of the interfaces in the table are collective names. This is explained in the following table.

Table 19-7

Collective name	PC card (corresponds to the CP interface)	Onboard interface of the PC (corresponds to the CPU interface)
“CP1616-CP1604”	CP 1616, CP1604	Onboard PN interface of the SIMATIC PC: CP 1616 integrated
“IE General”	Intel PRO/1000: 82574L, 82571EB, 82573L, 82541PI (non shared IRQ required)	Onboard Ethernet interface of the SIMATIC PC: 4x7B, 6x7B, 8x7B, 4x7C

(*3): Interfaces, under the control of WinAC RTX. The interfaces must be configured as “submodules”:

(*4): Interfaces under the control of Windows.

19.2.7 Panel PC to PN/IE

Table 19-8

Controller to PN/IE: Box PC (*1) HMI IPC477C bundles with RTX		**Communication type**				
		SIMATIC-specific	**Open standard**			
		S7 communication	**Open communication**	**PN communication**		
				IOC	**IOD**	**CBA**
CPU / CP (*2)	under the control of RTX (submodules) (*3): • “CP1616-CP1604”	(1) (IoT)	(3) (TCP, UDP)	(2)	---	x
	under the control of Windows (*4): • “IE General”	(1) (IoT)	---	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

Explanations on the table

(*1): with preinstalled WinAC RTX (F) 2009 (optional)

(*2): meaning of the interface CPU or CP: CPU = onboard interface of the PC, CP = plug-in PC card

The designations of the interfaces in the table are collective names. This is explained in the following table.

Table 19-9

Collective name	PC card (corresponds to the CP interface)	Onboard interface of the PC (corresponds to the CPU interface)
“CP1616-CP1604”	CP 1616, CP1604	Onboard PN interface of the SIMATIC PC: CP 1616 integrated
“IE General”	Intel PRO/1000: 82574L, 82571EB, 82573L, 82541PI (non shared IRQ required)	Onboard Ethernet interface of the SIMATIC PC: 4x7B, 6x7B, 8x7B, 4x7C

(*3): Interfaces, under the control of WinAC RTX. The interfaces must be configured as “submodules”:

(*4): Interfaces under the control of Windows.

19.2.8 WinAC MP to PN/IE

Table 19-10

Controller to PN/IE: WinAC MP for Multipanel (*1) MP177, MP277, MP377		**Communication type**			
		SIMATIC-specific	**Open standard**		
		S7 communication	**PN communication**		
			IOC	**IOD**	**CBA**
CPU	under the control of WinAC MP: integrated IE interface	(1) (IoT)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

Explanations on the table

(*1): with WinAC MP 2008

19.2.9 WinAC RTX to PN/IE

Table 19-11

Controller to PN/IE: WinAC RTX (*1)		**Communication type**				
		SIMATIC-specific	**Open standard**			
		S7 communication	**Open communication**	**PN communication**		
				IOC	**IOD**	**CBA**
CPU / CP (*2)	under the control of RTX (submodules) (*3): • “CP1616-CP1604” • “IE General”	(1) (IoT)	(3) (TCP, UDP)	(2)	---	x
	under the control of Windows (*4): • “CP1616-CP1604” • “IE General”	(1) (IoT)	---	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

Explanations on the table

(*1): WinAC RTX (F) 2009 on PC

(*2): meaning of the interface CPU or CP: CPU = onboard interface of the PC, CP = plug-in PC card

The designations of the interfaces in the table are collective names. This is explained in the following table.

Table 19-12

Collective name	PC card (corresponds to the CP interface)	Onboard interface of the PC (corresponds to the CPU interface)
“CP1616-CP1604”	CP 1616, CP1604	Onboard PN interface of the SIMATIC PC: CP 1616 integrated
“IE General”	Intel PRO/1000: 82574L, 82571EB, 82573L, 82541PI (non shared IRQ required)	Onboard Ethernet interface of the SIMATIC PC: 4x7B, 6x7B, 8x7B, 4x7C

(*3): Interfaces, under the control of WinAC RTX. The interfaces must be configured as “submodules”:

(*4): Interfaces under the control of Windows.

19.3 PN/IE: Controller 1 = ET 200 CPU

19.3.1 ET 200 CPU / ET 200 CPU

Table 19-13

Controller 2: ET 200 CPU			**Controller 1: ET 200 CPU to PN/IE**			
			CPU			
			IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	**OC**	**PN (*1)**	
					PNIO	CBA
CPU	IM 151-8(F) PN/DP CPU IM 154-8(F) PN/DP CPU	IOC, IOD	(1)+(14)	(3)	(2)	x

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(14) USEND_E/URCV_E, PUT_E, GET_E

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.3.2 ET 200 CPU / S7-300

Table 19-14

Controller 2: S7-300			**Controller 1: ET 200 CPU to PN/IE**			
			CPU			
			IM 151-8(F) PN/DP, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	**OC**	**PN (*1)**	
					PNIO	CBA
CPU	all with interface: PN	IOC, IOD	(1)+(14)	(3)	(2)	x
CP	343-1 Lean	IOD	(22)	(3) / (8)	(2) / (6)	---
	343-1	IOC, IOD	(1)	(3) / (8)	(2) / (6)	---
	343-1 Advanced	IOC, IOD	(1)	(3) / (8)	(2) / (6)	x
	343-1 ERPC	---	(1)	(3) / (8)	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(6) PNIO_SEND, PNIO_RECV

(8) AG_SEND/AG_RECV

(14) USEND_E/URCV_E, PUT_E, GET_E

(22) Controller 2 is server (for PUT, GET)

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.3.3 ET 200 CPU / S7-400

Table 19-15

Controller 2: S7-400			**Controller 1: ET 200 CPU to PN/IE**			
			CPU			
			IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	**OC**	**PN (*1)**	
					PNIO	CBA
CPU	all with interface: PN	IOC, IOD	(1)+(14)	(3)	(2)	x
CP	443-1	IOC, IOD	(1)	(3) / (4)+(9)	(2)	---
	443-1 Advanced	IOC, IOD	(1)	(3) / (4)+(9)	(2)	x

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

(14) USEND_E/URCV_E, PUT_E, GET_E

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.3.4 ET 200 CPU / S7-1200

Table 19-16

Controller 2: S7-1200			**Controller 1: ET 200 CPU to PN/IE**			
			CPU			
			IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	**OC**	**PN**	
					PNIO	CBA
CPU	1211C 1212C 1214C	---	(22)+(32)	(4) / (4)+(5)	---	---

Back to jump distributor PN/IE

Communication blocks

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

(22) Controller 2 is server (for PUT, GET)

(32) Controller 2 is server (for PUT_E, GET_E)

19.3.5 ET 200 CPU / S7-mEC

Table 19-17

Controller 2: S7-mEC				**Controller 1: ET 200 CPU to PN/IE**			
				CPU			
				IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU			
				IOC, IOD			
				S7	**OC**	**PN (*2)**	
						PNIO	CBA
CPU	EC31	Control of RTX (submodules): X1	IOC	(1)+(14)	(3)	(2)	x
		Control of Windows (*1): X2	---	(1)+(14)	---	---	---
CP	EM PC	Control of Windows (*1): X1	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(14) USEND_E/URCV_E, PUT_E, GET_E

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

(*2): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.3.6 ET 200 CPU / Box PC

Table 19-18

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: ET 200 CPU to PN/IE**			
			CPU			
			IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	**OC**	**PN (*1)**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)+(14)	(3)	(2)	x
	Control of Windows: “IE General”	---	(1)+(14)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(14) USEND_E/URCV_E, PUT_E, GET_E

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.3.7 ET 200 CPU / Panel PC

Table 19-19

Controller 2: Panel PC HMI IPC477C bundles with RTX			**Controller 1: ET 200 CPU to PN/IE**			
			CPU			
			IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU			
			IOC, IOD			
			S7	**OC**	**PN (*1)**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)+(14)	(3)	(2)	x
	Control of Windows: “IE General”	---	(1)+(14)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(14) USEND_E/URCV_E, PUT_E, GET_E

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.3.8 ET 200 CPU / WinAC MP

Table 19-20

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC MP on Multipanel
MP177, MP277, MP377</td><td colspan="4">Controller 1: ET 200 CPU to PN/IE</td></tr>
<tr><td colspan="4">CPU</td></tr>
<tr><td colspan="4">IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU</td></tr>
<tr><td colspan="4">IOC, IOD</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td>CPU</td><td>Control of WinAC MP: integrated IE interface</td><td>---</td><td>(1)+(14)</td><td>---</td><td>---</td><td>---</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(14) USEND_E/URCV_E, PUT_E, GET_E

19.3.9 ET 200 CPU / WinAC RTX

Table 19-21

Controller 2: WinAC RTX on PC WinAC RTX 2009			**Controller 1: ET 200 CPU to PN/IE** CPU IM 151-8(F) PN/DP CPU, IM 154-8(F) PN/DP CPU IOC, IOD			
			S7	**OC**	**PN (*1)**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): • “CP1616-CP1604” • “IE General”	IOC	(1)+(14)	(3)	(2)	x
	Control of Windows: • “CP1616-CP1604” • “IE General”	---	(1)+(14)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(14) USEND_E/URCV_E, PUT_E, GET_E

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.4 PN/IE: Controller 1 = S7-300

19.4.1 (S7-300 / ET 200 CPU)

See ET 200 CPU / S7-300: 19.3.2

19.4.2 S7-300 / S7-300

Table 19-22

Controller 2: S7-300			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU	all with interface: PN	IOC, IOD	(1)+(14)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	x	(1)	(8)/(3)
CP	343-1 Lean	IOD	(22)	(3)/(8)	(2)/(6)	---	---	(8)	---	---	(22)	(8)	(6)	---	(22)	(8)	(6)	---	(22)	(8)
	343-1	IOC, IOD	(1)	(3)/(8)	(2)/(6)	---	(21)	(8)	(6)	---	(1)	(8)	(6)	---	(1)	(8)	(6)	---	(1)	(8)
	343-1 Advanced	IOC, IOD	(1)	(3 /(8)	(2)/(6)	x	(21)	(8)	(6)	---	(1)	(8)	(6)	---	(1)	(8)	(6)	x	(1)	(8)
	343-1 ERPC	---	(1)	(3)/(8)	---	---	(21)	(8)	---	---	(1)	(8)	---	---	(1)	(8)	---	---	(1)	(8)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(6) PNIO_SEND, PNIO_RECV
(8) AG_SEND/AG_RECV
(14) USEND_E/URCV_E, PUT_E, GET_E
(21) Controller 1 is server (for PUT, GET)
(22) Controller 2 is server (for PUT, GET)

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.4.3 S7-300 / S7-400

Table 19-23

Controller 2: S7-400			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU	all with interface: PN	IOC, IOD	(1)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	x	(1)	(8)/(3)
CP	443-1	IOC, IOD	(1)	(3)/(4)+(9)	(2)	---	(21)	(8)/(4)+(9)	(6)/(2)	---	(1)	(8)/(4)+(9)	(6)/(2)	---	(1)	(8)/(4)+(9)	(6)/(2)	---	(1)	(8)/(4)+(9)
	443-1 Advanced	IOC, IOD	(1)	(3)/(4)+(9)	(2)	x	(21)	(8)/(4)+(9)	(6)/(2)	---	(1)	(8)/(4)+(9)	(6)/(2)	---	(1)	(8)/(4)+(9)	(6)/(2)	x	(1)	(8)/(4)+(9)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(4) TSEND/TRCV
(6) PNIO_SEND, PNIO_RECV
(21) Controller 1 is server (for PUT, GET)
(8) AG_SEND/AG_RECV
(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.4.4 S7-300 / S7-1200

Table 19-24

Controller 2: S7-1200			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU	1211C 1212C 1214C	---	(22)	(3)/(4)+(5)	---	---	---	(8)/(4)+(5)	---	---	(22)	(8)/(4)+(5)	---	---	(22)	(8)/(4)+(5)	---	---	(22)	(8)/(4)+(5)

Back to jump distributor PN/IE

Communication blocks

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

(8) AG_SEND/AG_RECV

(22) Controller 2 is server (for PUT, GET)

19.4.5 S7-300 / S7-mEC

Table 19-25

Controller 2: S7-mEC				Controller 1: S7-300 to PN/IE																	
				CPU				CP													
				all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
				IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
				S7	**OC**	**PN (*2)**		**S7**	**OC**	**PN (*2)**		**S7**	**OC**	**PN (*2)**		**S7**	**OC**	**PN**		**S7**	**OC**
						PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU	EC31	Control of RTX (submodules): X1	IOC	(1)+(14)/(1)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	x	(1)	(8)/(3)
		Control of Windows (*1): X2	---	(1)+(14)/(1)	---	---	---	(21)	---	---	---	(1)	---	---	---	(1)	---	---	---	(1)	---
CP	EM PC	Control of Windows (*1): X1	---	(1)	---	---	---	(21)	---	---	---	(1)	---	---	---	(1)	---	---	---	(1)	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(6) PNIO_SEND, PNIO_RECV

(8) AG_SEND/AG_RECV

(14) USEND_E/URCV_E, PUT_E, GET_E

(21) Controller 1 is server (for PUT, GET)

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

(*2): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.4.6 S7-300 / Box PC

Table 19-26

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	x	(1)	(8)/(3)
	Control of Windows: “IE General”	---	(1)	---	---	---	(21)	---	---	---	(1)	---	---	---	(1)	---	---	---	(1)	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

(8) AG_SEND/AG_RECV

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(6) PNIO_SEND, PNIO_RECV

(21) Controller 1 is server (for PUT, GET)

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.4.7 S7-300 / Panel PC

Table 19-27

Controller 2: Panel PC HMI IPC477C bundles with RTX			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	x	(1)	(8)/(3)
	Control of Windows: “IE General”	---	(1)	---	---	---	(21)	---	---	---	(1)	---	---	---	(1)	---	---	---	(1)	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

(8) AG_SEND/AG_RECV

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(6) PNIO_SEND, PNIO_RECV

(21) Controller 1 is server (for PUT, GET)

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.4.8 S7-300 / WinAC MP

Table 19-28

Controller 2: WinAC MP MP177, MP277, MP377			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU	Control of WinAC MP: integrated IE interface	---	(1)	---	---	---	(21)	---	---	---	(1)	---	---	---	(1)	---	---	---	(1)	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(21) Controller 1 is server (for PUT, GET)

19.4.9 S7-300 / WinAC RTX

Table 19-29

Controller 2: WinAC RTX auf PC WinAC RTX 2009			**Controller 1: S7-300 to PN/IE**																	
			CPU				CP													
			all with interface: PN				343-1 Lean				343-1				343-1 Advanced				343-1 ERPC	
			IOC, IOD				IOD				IOC, IOD				IOC, IOD				---	
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**
					PNIO	CBA			PNIO	CBA			PNIO	CBA			PNIO	CBA		
CPU/CP	Control of RTX (submodules) (IOC): • "CP1616-CP1604" • "IE General"	IOC	(1)	(3)	(2)	x	(21)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	---	(1)	(8)/(3)	(6)/(2)	x	(1)	(8)/(3)
	Control of Windows (---): • "CP1616-CP1604" • "IE General"	---	(1)	---	---	---	(21)	---	---	---	(1)	---	---	---	(1)	---	---	---	(1)	

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(6) PNIO_SEND, PNIO_RECV

(8) AG_SEND/AG_RECV

(21) Controller 1 is server (for PUT, GET)

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.5 PN/IE: Controller 1 = S7-400

19.5.1 (S7-400 / ET 200 CPU)

See ET 200 CPU / S7-400: 19.3.3

19.5.2 (S7-400 / S7-300)

See S7-300 / S7-400: 19.4.3

19.5.3 S7-400 / S7-400

Table 19-30

Controller 2: S7-400			**Controller 1: S7-400 to PN/IE**											
			CPU				CP							
			all with interface: PN				443-1				443-1 Advanced			
			IOC, IOD				IOC, IOD				IOC, IOD			
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**	
					PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU	all with interface: PN	IOC, IOD	(1)	(3)	(2)	x	(1)	(4)+(9)/(3)	(2)	---	(1)	(4)+(9)/(3)	(2)	x
CP	443-1	IOC, IOD	(1)	(3)/(4)+(9)	(2)	---	(1)	(4)+(9)	(2)	---	(1)	(4)+(9)	(2)	---
	443-1 Advanced	IOC, IOD	(1)	(3)/(4)+(9)	(2)	x	(1)	(4)+(9)	(2)	---	(1)	(4)+(9)	(2)	x

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.5.4 S7-400 / S7-1200

Table 19-31

Controller 2: S7-1200			**Controller 1: S7-400 to PN/IE**											
			CPU				CP							
			all with interface: PN				443-1				443-1 Advanced			
			IOC, IOD				IOC, IOD				IOC, IOD			
			S7	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**	**PN**	
					PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU	1211C 1212C 1214C	---	(22)	(3)/(4)+(5)	---	---	(22)	(4)+(9)/(4)+(5)	---	---	(22)	(4)+(9)/(4)+(5)	---	---

Back to jump distributor PN/IE

Communication blocks

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

(22) Controller 2 is server (for PUT, GET)

19.5.5 S7-400 / S7-mEC

Table 19-32

Controller 2: S7-mEC				**Controller 1: S7-400 to PN/IE**											
				CPU				CP							
				all with interface: PN				443-1				443-1 Advanced			
				IOC, IOD				IOC, IOD				IOC, IOD			
				S7	**OC**	**PN (*2)**		**S7**	**OC**	**PN (*2)**		**S7**	**OC**	**PN (*2)**	
						PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU	EC31	Control of RTX (submodules): X1	IOC	(1)	(3)	(2)	x	(1)	(4)+(9)/(3)	(2)	---	(1)	(4)+(9)/(3)	(2)	x
		Control of Windows (*1): X2	---	(1)	---	---	---	(1)	---	---	---	(1)	---	---	---
CP	EM PC	Control of Windows (*1): X1	---	(1)	---	---	---	(1)	---	---	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

(*2): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.5.6 S7-400 / Box PC

Table 19-33

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: S7-400 to PN/IE**											
			CPU				CP							
			all with interface: PN				443-1				443-1 Advanced			
			IOC, IOD				IOC, IOD				IOC, IOD			
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**	
					PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)	(3)	(2)	x	(1)	(4)+(9)/(3)	(2)	---	(1)	(4)+(9)/(3)	(2)	x
	Control of Windows: “IE General”	---	(1)	---	---	---	(1)	---	---	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET (2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV (4) TSEND/TRCV

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.5.7 S7-400 / Panel PC

Table 19-34

Controller 2: Panel PC HMI IPC477C bundles with RTX			**Controller 1: S7-400 to PN/IE**											
			CPU				CP							
			all with interface: PN				443-1				443-1 Advanced			
			IOC, IOD				IOC, IOD				IOC, IOD			
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**	
					PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)	(3)	(2)	x	(1)	(4)+(9)/(3)	(2)	---	(1)	(4)+(9)/(3)	(2)	x
	Control of Windows: “IE General”	---	(1)	---	---	---	(1)	---	---	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.5.8 S7-400 / WinAC MP

Table 19-35

Controller 2: WinAC MP MP177, MP277, MP377			**Controller 1: S7-400 to PN/IE**											
			CPU				CP							
			all with interface: PN				443-1				443-1 Advanced			
			IOC, IOD				IOC, IOD				IOC, IOD			
			S7	**OC**	**PN**		**S7**	**OC**	**PN**		**S7**	**OC**	**PN**	
					PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU	Control of WinAC MP: integrated IE interface	---	(1)	---	---	---	(1)	---	---	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

19.5.9 S7-400 / WinAC RTX

Table 19-36

Controller 2: WinAC RTX on PC WinAC RTX 2009			**Controller 1: S7-400 to PN/IE**											
			CPU				CP							
			all with interface: PN				443-1				443-1 Advanced			
			IOC, IOD				IOC, IOD				IOC, IOD			
			S7	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**		**S7**	**OC**	**PN (*1)**	
					PNIO	CBA			PNIO	CBA			PNIO	CBA
CPU/CP	Control of RTX (submodules): • “CP1616-CP1604” • “IE General”	IOC	(1)	(3)	(2)	x	(1)	(4)+(9)/(3)	(2)	---	(1)	(4)+(9)/(3)	(2)	x
	Control of Windows: • “CP1616-CP1604” • “IE General”	---	(1)	---	---	---	(1)	---	---	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(3) TSEND/TRCV, TUSEND/TURCV
(4) TSEND/TRCV
(9) AG_SEND/AG_RECV, AG_LSEND/AG_LRECV, AG_SSEND/AG_SRECV

Explanations on the table

(*1): Communication via PNIO between PROFINET IO controller and PROFINET IO device.

19.6 PN/IE: Controller 1 = S7-1200

19.6.1 (S7-1200 / ET 200 CPU)

See ET 200 CPU / S7-1200: 19.3.4

19.6.2 (S7-1200 / S7-300)

See S7-300 / S7-1200: 19.4.4

19.6.3 (S7-1200 / S7-400)

See S7-400 / S7-1200: 19.5.4

19.6.4 S7-1200 / S7-1200

Table 19-37

<table>
<tr><td colspan="3" rowspan="6">Controller 2: S7-1200</td><td colspan="4">Controller 1: S7-1200 to PN/IE</td></tr>
<tr><td colspan="4">CPU</td></tr>
<tr><td colspan="4">1211C, 1212C, 1214C</td></tr>
<tr><td colspan="4">---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td>CPU</td><td>1211C
1212C
1214C</td><td>---</td><td>---</td><td>(4)+(5)</td><td>---</td><td>---</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

19.6.5 S7-1200 / S7-mEC

Table 19-38

Controller 2: S7-mEC				**Controller 1: S7-1200 to PN/IE**			
				CPU			
				1211C, 1212C, 1214C			

				S7	**OC**	**PN**	
						PNIO	CBA
CPU	EC31	Control of RTX (submodules): X1	IOC	(21)	(4)+(5)/(3)	---	---
		Control of Windows (*1): X2	---	(21)	---	---	---
CP	EM PC	Control of Windows (*1): X1	---	(21)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

(21) Controller 1 is server (for PUT, GET)

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

19.6.6 S7-1200 / Box PC

Table 19-39

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: S7-1200 to PN/IE**			
			CPU			
			1211C, 1212C, 1214C			

			S7	**OC**	**PN**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(21)	(4)+(5)/(3)	---	---
	Control of Windows: “IE General”	---	(21)	---	---	---

[Back to jump distributor PN/IE]

Communication blocks

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C (only for S7-1200)

(21) Controller 1 is server (for PUT, GET)

19.6.7 S7-1200 / Panel PC

Table 19-40

Controller 2: Panel PC HMI IPC477C bundles with RTX			**Controller 1: S7-1200 to PN/IE**			
			CPU			
			1211C, 1212C, 1214C			

			S7	**OC**	**PN**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(21)	(4)+(5)/(3)	---	---
	Control of Windows: “IE General”	---	(21)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C

(21) Controller 1 is server (for PUT, GET)

19.6.8 S7-1200 / WinAC MP

Table 19-41

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC MP
MP177, MP277, MP377</td><td colspan="4">Controller 1: S7-1200 to PN/IE</td></tr>
<tr><td colspan="4">CPU</td></tr>
<tr><td colspan="4">1211C, 1212C, 1214C</td></tr>
<tr><td colspan="4">---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td>CPU</td><td>Control of WinAC MP:
integrated IE interface</td><td>---</td><td>(21)</td><td>---</td><td>---</td><td>---</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(21) Controller 1 is server (for PUT, GET)

19.6.9 S7-1200 / WinAC RTX

Table 19-42

Controller 2: WinAC RTX on PC WinAC RTX 2009			**Controller 1: S7-1200 to PN/IE**			
			CPU			
			1211C, 1212C, 1214C			

			S7	**OC**	**PN**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): • “CP1616-CP1604” • “IE General”	IOC	(21)	(4)+(5)/(3)	---	---
	Control of Windows: • “CP1616-CP1604” • “IE General”	---	(21)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(3) TSEND/TRCV, TUSEND/TURCV

(4) TSEND/TRCV

(5) TSEND_C/TRCV_C (only for S7-1200)

(21) PUT, GET, Controller 1 is server

19.7 PN/IE: Controller 1 = S7-mEC

19.7.1 (S7-mEC / ET 200 CPU)

See ET 200 CPU / S7-mEC: 19.3.5

19.7.2 (S7-mEC / S7-300)

See S7-300 / S7-mEC: 19.4.5

19.7.3 (S7-mEC / S7-400)

See S7-400 / S7-mEC: 19.5.5

19.7.4 (S7-mEC / S7-1200)

See S7-1200 / S7-mEC: 19.6.5

19.7.5 S7-mEC / S7-mEC

Table 19-43

Controller 2: S7-mEC				Controller 1: S7-mEC to PN/IE					
				CPU					CP
				EC31					EM PC
				Submodule: X1				Windows (*1): X2	Windows (*1): X1
				IOC				---	---
				S7	**OC**	**PN**		**S7**	**S7**
						PNIO	CBA		
CPU	EC31	Control of RTX (submodules): X1	IOC	(1)	(3)	---	x	(1)	(1)
		Control of Windows (*1): X2	---	(1)	---	---	---	(1)	(1)
CP	EM PC	Control of Windows (*1): X1	---	(1)	---	---	---	(1)	(1)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

19.7.6 S7-mEC / Box PC

Table 19-44

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: S7-mEC to PN/IE**					
			CPU					CP
			EC31					EM PC
			Submodule: X1				Windows (*1): X2	Windows (*1): X1
			IOC				---	---
			S7	**OC**	**PN**		**S7**	**S7**
					PNIO	CBA		
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)	(3)	---	x	(1)	(1)
	Control of Windows: “IE General”	---	(1)	---	---	---	(1)	(1)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

19.7.7 S7-mEC / Panel PC

Table 19-45

<table>
<tr><td colspan="3" rowspan="6">Controller 2: Panel PC
IPC477C bundles with RTX</td><td colspan="5">Controller 1: S7-mEC to PN/IE</td></tr>
<tr><td colspan="5">CPU</td><td>CP</td></tr>
<tr><td colspan="5">EC31</td><td>EM PC</td></tr>
<tr><td colspan="4">Submodule:
X1</td><td>Windows (*1):
X2</td><td>Windows (*1):
X1</td></tr>
<tr><td colspan="4">IOC</td><td>---</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td rowspan="2">CPU/CP</td><td>Control of RTX (submodules):
“CP1616-CP1604”</td><td>IOC</td><td>(1)</td><td>(3)</td><td>---</td><td>x</td><td>(1)</td><td>(1)</td></tr>
<tr><td>Control of Windows:
“IE General”</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(1)</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

19.7.8 S7-mEC / WinAC MP

Table 19-46

Controller 2: WinAC MP MP177, MP277, MP377			**Controller 1: S7-mEC to PN/IE**					
			CPU					CP
			EC31					EM PC
			Submodule: X1				Windows (*1): X2	Windows (*1): X1
			IOC				---	---
			S7	**OC**	**PN**		**S7**	**S7**
					PNIO	CBA		
CPU	Control of WinAC MP: integrated IE interface	---	(1)	---	---	---	(1)	(1)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

19.7.9 S7-mEC / WinAC RTX

Table 19-47

<table>
<tr><td colspan="3" rowspan="7">Controller 2: WinAC RTX on PC
WinAC RTX 2009</td><td colspan="6">Controller 1: S7-mEC to PN/IE</td></tr>
<tr><td colspan="5">CPU</td><td>CP</td></tr>
<tr><td colspan="5">EC31</td><td>EM PC</td></tr>
<tr><td colspan="4">Submodule:
X1</td><td>Windows (*1):
X2</td><td>Windows (*1):
X1</td></tr>
<tr><td colspan="4">IOC</td><td>---</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td rowspan="2">CPU/CP</td><td>Control of RTX (submodules):
• “CP1616-CP1604”
• “IE General”</td><td>IOC</td><td>(1)</td><td>(3)</td><td>---</td><td>x</td><td>(1)</td><td>(1)</td></tr>
<tr><td>Control of Windows:
• “CP1616-CP1604”
• “IE General”</td><td>---</td><td>(1)</td><td></td><td>---</td><td>---</td><td>(1)</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

Explanations on the table

(*1): Both interfaces cannot be operated at the same time (EC31 EXOR EM PC).

19.8 PN/IE: Controller 1 = Box PC

19.8.1 (Box PC / ET 200 CPU)

See ET 200 CPU / Box PC: 19.3.6

19.8.2 (Box PC / S7-300)

See S7-300 / Box PC: 19.4.6

19.8.3 (Box PC / S7-400)

See S7-400 / Box PC: 19.5.6

19.8.4 (Box PC / S7-1200)

See S7-1200 / Box PC: 19.6.6

19.8.5 (Box PC / S7-mEC)

See S7-mEC / Box PC: 19.7.6

19.8.6 Box PC / Box PC

Table 19-48

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: Box PC to PN/IE**				
			IPC427C bundles with RTX				
			CPU/CP				
			Control of RTX (submodules): “CP1616-CP1604”				Control of Windows: “IE General”
			IOC				---
			S7	**OC**	**PN**		**S7**
					PNIO	CBA	
CPU/CP	Control of RTX (submodules): “CP1616-CP1604”	IOC	(1)	(3)	---	x	(1)
	Control of Windows: “IE General”	---	(1)	---	---	---	(1)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

19.8.7 Box PC / Panel PC

Table 19-49

<table>
<tr><td colspan="3" rowspan="7">Controller 2: Panel PC
IPC427C bundles with RTX</td><td colspan="5">Controller 1: Box PC to PN/IE</td></tr>
<tr><td colspan="5">IPC427C bundles with RTX</td></tr>
<tr><td colspan="5">CPU/CP</td></tr>
<tr><td colspan="4">Control of RTX (submodules):
“CP1616-CP1604”</td><td>Control of Windows:
“IE General”</td></tr>
<tr><td colspan="4">IOC</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td rowspan="2">CPU/CP</td><td>Control of RTX (submodules):
“CP1616-CP1604”</td><td>IOC</td><td>(1)</td><td>(3)</td><td>---</td><td>x</td><td>(1)</td></tr>
<tr><td>Control of Windows:
“IE General”</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

19.8.8 Box PC / WinAC MP

Table 19-50

<table>
<tr><td colspan="3" rowspan="7">Controller 2: WinAC MP
MP177, MP277, MP377</td><td colspan="5">Controller 1: Box PC to PN/IE</td></tr>
<tr><td colspan="5">IPC427C bundles with RTX</td></tr>
<tr><td colspan="5">CPU/CP</td></tr>
<tr><td colspan="4">Control of RTX (submodules):
“CP1616-CP1604”</td><td>Control of Windows:
“IE General”</td></tr>
<tr><td colspan="4">IOC</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td>CPU</td><td>Control of WinAC MP:
integrated IE interface</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

19.8.9 Box PC / WinAC RTX

Table 19-51

<table>
<tr><td colspan="3" rowspan="7">Controller 2: WinAC RTX on PC
WinAC RTX 2009</td><td colspan="5">Controller 1: Box PC to PN/IE</td></tr>
<tr><td colspan="5">IPC427C bundles with RTX</td></tr>
<tr><td colspan="5">CPU/CP</td></tr>
<tr><td colspan="4">Control of RTX (submodules):
“CP1616-CP1604”</td><td>Control of Windows:
“IE General”</td></tr>
<tr><td colspan="4">IOC</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td rowspan="2">CPU/CP</td><td>Control of RTX (submodules):
• “CP1616-CP1604”
• “IE General”</td><td>IOC</td><td>(1)</td><td>(3)</td><td>---</td><td>x</td><td>(1)</td></tr>
<tr><td>Control of Windows:
• “CP1616-CP1604”
• “IE General”</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

19.9 PN/IE: Controller 1 = Panel PC

19.9.1 (Panel PC / ET 200 CPU)

See ET 200 CPU / Panel PC: 19.3.7

19.9.2 (Panel PC / S7-300)

See S7-300 / Panel PC: 19.4.7

19.9.3 (Panel PC / S7-400)

See S7-400 / Panel PC: 19.5.7

19.9.4 (Panel PC / S7-1200)

See S7-1200 / Panel PC: 19.6.7

19.9.5 (Panel PC / S7-mEC)

See S7-mEC / Panel PC: 19.7.7

19.9.6 (Panel PC / Box PC)

See Box PC / Panel PC: 19.8.7

19.9.7 Panel PC / Panel PC

Table 19-52

<table>
<tr><td colspan="3" rowspan="7">Controller 2: Panel PC
IPC4277C bundles with RTX</td><td colspan="5">Controller 1: Panel PC to PN/IE</td></tr>
<tr><td colspan="5">IPC477C bundles with RTX</td></tr>
<tr><td colspan="5">CPU/CP</td></tr>
<tr><td colspan="4">Control of RTX (submodules):
“CP1616-CP1604”</td><td>Control of Windows:
“IE General”</td></tr>
<tr><td colspan="4">IOC</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td rowspan="2">CPU/CP</td><td>Control of RTX (submodules):
“CP1616-CP1604”</td><td>IOC</td><td>(1)</td><td>(3)</td><td>---</td><td>x</td><td>(1)</td></tr>
<tr><td>Control of Windows:
“IE General”</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

19.9.8 Panel PC / WinAC MP

Table 19-53

Controller 2: WinAC MP MP177, MP277, MP377			**Controller 1: Panel PC to PN/IE**				
			IPC477C bundles with RTX				
			CPU/CP				
			Control of RTX (submodules): “CP1616-CP1604”				Control of Windows: “IE General”
			IOC				---
			S7	**OC**	**PN**		**S7**
					PNIO	CBA	
CPU	Control of WinAC MP: integrated IE interface	---	(1)	---	---	---	(1)

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

19.9.9 Panel PC / WinAC RTX

Table 19-54

<table>
<tr><td colspan="3" rowspan="7">Controller 2: WinAC RTX on PC
WinAC RTX 2009</td><td colspan="5">Controller 1: Panel PC to PN/IE</td></tr>
<tr><td colspan="5">IPC477C bundles with RTX</td></tr>
<tr><td colspan="5">CPU/CP</td></tr>
<tr><td colspan="4">Control of RTX (submodules):
“CP1616-CP1604”</td><td>Control of Windows:
“IE General”</td></tr>
<tr><td colspan="4">IOC</td><td>---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td><td rowspan="2">S7</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td rowspan="2">CPU/CP</td><td>Control of RTX (submodules):
• “CP1616-CP1604”
• “IE General”</td><td>IOC</td><td>(1)</td><td>(3)</td><td>---</td><td>x</td><td>(1)</td></tr>
<tr><td>Control of Windows:
• “CP1616-CP1604”
• “IE General”</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(1)</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

19.10 PN/IE: Controller 1 = WinAC MP

19.10.1 (WinAC MP / ET 200 CPU)

See ET 200 CPU / WinAC MP: 19.3.8

19.10.2 (WinAC MP / S7-300)

See S7-300 / WinAC MP: 19.4.8

19.10.3 (WinAC MP / S7-400)

See S7-400 / WinAC MP: 19.5.8

19.10.4 (WinAC MP / S7-1200)

See S7-1200 / WinAC MP: 19.6.8

19.10.5 (WinAC MP / S7-mEC)

See S7-mEC / WinAC MP: 19.7.8

19.10.6 (WinAC MP / Box PC)

See Box PC / WinAC MP: 19.8.8

19.10.7 (WinAC MP / Panel PC)

See Panel PC / WinAC MP: 19.9.8

19.10.8 WinAC MP / WinAC MP

Table 19-55

<table>
<tr><td colspan="3" rowspan="7">Controller 2: WinAC MP
MP177, MP277, MP377</td><td colspan="4">Controller 1: WinAC MP to PN/IE</td></tr>
<tr><td colspan="4">MP177, MP277, MP377</td></tr>
<tr><td colspan="4">CPU</td></tr>
<tr><td colspan="4">Control of WinAC MP:
integrated IE interface</td></tr>
<tr><td colspan="4">---</td></tr>
<tr><td rowspan="2">S7</td><td rowspan="2">OC</td><td colspan="2">PN</td></tr>
<tr><td>PNIO</td><td>CBA</td></tr>
<tr><td>CPU</td><td>Control of WinAC MP:
integrated IE interface</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td></tr>
</table>

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

19.10.9 WinAC MP / WinAC RTX

Table 19-56

Controller 2: WinAC RTX on PC WinAC RTX 2009			**Controller 1: WinAC MP to PN/IE** MP177, MP277, MP377 CPU Control of WinAC MP: integrated IE interface ---			
			S7	**OC**	**PN**	
					PNIO	CBA
CPU/CP	Control of RTX (submodules): • “CP1616-CP1604” • “IE General”	IOC	(1)	---	---	---
	Control of Windows: • “CP1616-CP1604” • “IE General”	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

19.11 PN/IE: Controller 1 = WinAC RTX

19.11.1 (WinAC RTX / ET 200 CPU)

See ET 200 CPU / WinAC RTX: 19.3.9

19.11.2 (WinAC RTX / S7-300)

See S7-300 / WinAC RTX: 19.4.9

19.11.3 (WinAC RTX / S7-400)

See S7-400 / WinAC RTX: 19.5.9

19.11.4 (WinAC RTX / S7-1200)

See S7-1200 / WinAC RTX: 19.6.9

19.11.5 (WinAC RTX / S7-mEC)

See S7-mEC / WinAC RTX: 19.7.9

19.11.6 (WinAC RTX / Box PC)

See Box PC / WinAC RTX: 19.8.9

19.11.7 (WinAC RTX / Panel PC)

See Panel PC / WinAC RTX: 19.9.9

19.11.8 (WinAC RTX / WinAC MP)

See WinAC MP / WinAC RTX: 19.10.9

19.11.9 WinAC RTX / WinAC RTX

Table 19-57

Controller 2: WinAC RTX on PC WinAC RTX 2009			**Controller 1: WinAC RTX to PN/IE**							
			WinAC RTX 2009							
			CPU/CP							
			Control of RTX (submodules): • "CP1616-CP1604" • "IE General"				Control of Windows: • "CP1616-CP1604" • "IE General"			
			IOC				---			
			S7	**OC**	**PN**		**S7**	**OC**	**PN**	
					PNIO	CBA			PNIO	CBA
CPU/CP	Control of RTX (submodules): • "CP1616-CP1604" • "IE General"	IOC	(1)	(3)	---	x	(1)	---	---	---
	Control of Windows: • "CP1616-CP1604" • "IE General"	---	(1)	---	---	---	(1)	---	---	---

Back to jump distributor PN/IE

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(3) TSEND/TRCV, TUSEND/TURCV

19.12 PN/IE: Overview of communication types

Comparison of all communication types with PN/IE.

The data is taken from the Communication types - details table:

- S7 communication (Table 29-1)
- Open communication with send/receive blocks (Table 31-3)
- Open communication with T blocks (Table 32-3)
- PN communication (Table 34-1)

Communication types – compact table:

Table 19-58

	SIMATIC-specific	Open standard			
	S7 communication	**Open communication**			**PN communication**
		Send/Receive blocks	T-blocks	T-compact blocks	**PNIO**
Protocols	ISO, IoT	ISO, IoT, TCP, UDP	IoT, TCP, UDP	IoT, TCP	PN
Interfaces	ISO: CP IoT: CPU, CP	CP	CPU, CP	CPU	CPU, CP
Communication blocks (max. data)	BSEND (≤ 64 Kbytes) USEND,USEND_E (≥ 160 bytes) PUT, PUT_E, GET, GET_E (≥ 160 bytes)	AG_xSEND (ISO, IoT, TCP ≤ 8 Kbytes) (UDP ≤ 2 Kbytes) -- Server for FETCH, WRITE (not for UDP)	TSEND, TUSEND, … (IoT ≤ 32 Kbytes) (TCP ≤ 64 Kbytes) (UDP = 1472 bytes)	TSEND_C (≤ 8192 bytes)	Load commands/ transfer commands (1, 2, 4 bytes) -- DPR_DAT, DPWR_DAT PNIO_SEND, PNIO_RECV (IOC ≤ 8192 bytes) (IOD ≤ 1440 bytes)
remote confirmation	BSEND: application USEND: transport PUT, GET: application	ISO, IoT, TCP: transport UDP: none	IoT, TCP: transport UDP: none	Transport	Application
Routing-capable?	ISO: no IoT: yes	ISO: no otherwise: yes	yes	yes	no
Connections?	yes	UDP: no otherwise: yes	UDP: no otherwise: yes	yes	no

Back to jump distributor PN/IE

20 Selection Aid: PROFIBUS (PB)

20.1 PB: Content of the chapter

The following is described for the PB medium:

- which interfaces (modules) and communication types are available?
(-> Interfaces table)
- which partners can communicate via which communication types?
(-> Combinations table)
- overview of all available communication types
(-> Communication types – compact table)

20.2 PB: Interfaces and communication types

20.2.1 ET 200 CPU to PB

Table 20-1

Controller to PB: ET 200 CPU			**Functionality of the interface**		**Communication type**		
					SIMATIC-specific		**Open standard**
					S7 basic communication	**S7 communication**	**DP communication**
ET 200 S	CPU	IM151-7 CPU	DP slave (*1)	active	I_PUT, I_GET, Server (S7)	PUT, GET, Server (S7)	(2)
				passive	I_PUT, I_GET, Server (S7)	---	(2)
	CP	DP master module	DP master		I_PUT, I_GET, **Client** (S7)	PUT, GET, Server (S7)	(2)
ET 200 Pro	CPU	IM154-8(F) PN/DP CPU	DP master		I_PUT, I_GET, **Client** (S7)	PUT, GET, Server (S7)	(2)
			DP-Slave (*1)	active	I_PUT, I_GET, Server (S7)	PUT, GET, Server (S7)	(2)
				passive	I_PUT, I_GET, Server (S7)	---	(2)
	CP	DP master module	DP master		I_PUT, I_GET, **Client** (S7)	PUT, GET, Server (S7)	(2)

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

Explanations on the table

(*1): A DP slave can be active or passive. An active DP slave receives the token and is therefore also an active node at PROFIBUS. A passive DP slave receives no token. The settings are made by configuring the operating mode of the DB interface in STEP 7 HW Configuration.

20.2.2 S7-300 to PB

Table 20-2

Controller to PB: S7-300		**Functionality of the interface**		**Communication type**				
				SIMATIC-specific		**Open standard**		
				S7 basic communication	**S7 communication**	**Open communication**	**FMS communication**	**DP communication**
CPU	all with interface: DP, MPI/DP (*2)	DP master		I_PUT, I_GET, **Client** (S7)	PUT, GET, Server (S7)	---	---	(2)
		DP slave (*3)	active	I_PUT, I_GET, Server (S7)	PUT, GET, Server (S7)	---	---	(2)
			passive	I_PUT, I_GET, Server (S7)	---	---	---	(2)
CP	342-5 (*1)	no DP operation		---	(1) (S7)	(8) (FDL)	---	---
		DP master		---	(1) (S7)	(8) (FDL)	---	(7)
		DP slave (*3)	active	---	PUT, GET, Server (S7)	(8) (FDL)	---	(7)
			passive	---	---	---	---	(7)
	343-5	no DP operation		---	PUT, GET, Server (S7)	(8) (FDL)	(10) (FMS)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(7) DP_SEND, DP_RECV

(8) AG_SEND/AG_RECV

(10) READ, WRITE, REPORT

Explanations on the table

(*1): 342-5 stands for the variants: CP 342-5, CP 342-5 FO

(*2): MPI/DP interface in DP operating mode

(*3): A DP slave can be active or passive. An active DP slave receives the token and is therefore also an active node at PROFIBUS. A passive DP slave receives no token. The settings are made by configuring the operating mode of the DB interface in STEP 7 HW Configuration.

20.2.3 S7-400 to PB

Table 20-3

Controller to PB: S7-400		Functionality of the interface		Communication type				
				SIMATIC-specific		Open standard		
				S7 basic communication	S7 communication	Open communication	FMS communication	DP communication
CPU	all with interface: DP (*3), MPI/DP (*2)	DP master		I_PUT, I_GET, **Client** (S7)	(1) (S7)	---	---	(2)
		DP slave (*4)	active	I_PUT, I_GET, Server (S7)	PUT, GET, Server, (S7)	---	---	(2)
			passive	I_PUT, I_GET, Server (S7)	---	---	---	(2)
CP	443-5 Basic	no DP operation		---	(1) (S7)	(8) (FDL)	(10) (FMS)	---
	443-5 Extented	no DP operation		---	(1) (S7)	(8) (FDL)	---	---
		DP master		---	(1) (S7)	(8) (FDL)	---	(2)

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(8) AG_SEND/AG_RECV

(10) READ, WRITE, REPORT

Explanations on the table

(*2): MPI/DP interface in DP operating mode

(*3): integrated interface (X1, ...) or plug-in interface module (IF1, ...)

(*4): a DP slave can be active or passive. An active DP slave receives the token and is therefore also an active node at PROFIBUS. A passive DP slave receives no token. The settings are made by configuring the operating mode of the DB interface in STEP 7 HW Configuration.

20.2.4 S7-1200 to PB

S7-1200 has no interface to PROFIBUS!

Back to jump distributor PB

20.2.5 S7-mEC to PB

Table 20-4

Controller to PB: S7-mEC (*1)			**Functionality of the interface**	**Communication type**	
				SIMATIC-specific	**Open standard**
				S7 communication	**DP communication**
CP	EM PCI-104 (*3)	under control of RTX (submodules) (*2): CP 5603	DP master	(1) (S7)	(2)

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

Explanations on the table

(*1): with preinstalled WinAC RTX (F) 2009

(*2): Interfaces, under the control of WinAC RTX. (F). The interfaces must be configured as “submodules”:

(*3): a maximum of 3 cards of the PCI-104 standard can be plugged in

20.2.6 Box PC to PB

Table 20-5

Controller to PB: Box PC (*1) IPC427C bundles with RTX		**Functionality of the interface**	**Communication type**	
			SIMATIC-specific	**Open standard**
			S7 communication	**DP communication**
CPU/CP (*2)	under control of RTX (submodules) (*3): • „CP5611-CP5621" • "CP5613-CP5603"	DP master	(1) (S7)	(2)

Back to jump distributor PB

Communication blocks and protocols

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

Explanations on the table

(*1): with WinAC RTX (F) 2009

(*2): meaning of the interface CPU or CP: CPU = onboard interface of the PC, CP = plug-in PC card

The designations of the interfaces in the table are collective names. This is explained in the following table.

Table 20-6

Interface	CP (plug-in to PC)	CPU (Onboard interface of the PC)
"CP5611-CP5621"	CP 5611-A2, CP5621	Onboard PB interface of the SIMATIC PC: CP 5611 integrated
"CP5613-CP5603"	CP 5613, CP 5613-A2, CP 5603	---

(*3): Interfaces, under the control of WinAC RTX. The interfaces must be configured as "submodules":

20.2.7 Panel PC to PB

Table 20-7

Controller to PB: Box PC (*1) HMI IPC477C bundles with RTX		**Functionality of the interface**	**Communication type**	
			SIMATIC-specific	**Open standard**
			S7 communication	**DP communication**
CPU/CP (*2)	under control of RTX (submodules) (*3): • "CP5611-CP5621" • "CP5613-CP5603"	DP master	(1) (S7)	(2)

Back to jump distributor PB

Communication blocks and protocols

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

Explanations on the table

(*1): with WinAC RTX (F) 2009

(*2): meaning of the interface CPU or CP: CPU = onboard interface of the PC, CP = plug-in PC card

The designations of the interfaces in the table are collective names. This is explained in the following table.

Table 20-8

Interface	CP (plug-in to PC)	CPU (Onboard interface of the PC)
"CP5611-CP5621"	CP 5611-A2, CP5621	Onboard PB interface of the SIMATIC PC: CP 5611 integrated
"CP5613-CP5603"	CP 5613, CP 5613-A2, CP 5603	---

(*3): Interfaces, under the control of WinAC RTX. The interfaces must be configured as "submodules":

20.2.8 WinAC MP to PB

Table 20-9

Controller to PB: WinAC MP for Multipanel (*1) MP177, MP277, MP377		**Functionality of the interface**	**Communication type**	
			SIMATIC-specific	**Open standard**
			S7 communication	**DP communication**
CPU	Under control of WinAC MP: integrated PB interface	DP master	(1) (S7)	(2)

Back to jump distributor PB

Communication blocks and protocols

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

Explanations on the table

(*1): with WinAC MP 2008

20.2.9 WinAC RTX to PB

Table 20-10

Controller to PB: WinAC RTX (*1)		Functionality of the interface	Communication type	
			SIMATIC-specific	Open standard
			S7 communication	DP communication
CPU/CP (*2)	under control of RTX (submodules) (*3): • "CP5611-CP5621" • "CP5613-CP5603"	DP master	(1) (S7)	(2)

Back to jump distributor PB

Communication blocks and protocols

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

Explanations on the table

(*1): WinAC RTX (F) 2009 on PC

(*2): meaning of the interface CPU or CP: CPU = onboard interface of the PC, CP = plug-in PC card

The designations of the interfaces in the table are collective names. This is explained in the following table.

Table 20-11

Interface	CP (plug-in to PC)	CPU (Onboard interface of the PC)
"CP5611-CP5621"	CP 5611-A2, CP5621	Onboard PB interface of the SIMATIC PC: CP 5611 integrated
"CP5613-CP5603"	CP 5613, CP 5613-A2, CP 5603	---

(*3): Interfaces, under the control of WinAC RTX. The interfaces must be configured as "submodules":

"CP5611-CP5621"

20.3 PB: Controller 1 = ET 200 CPU

20.3.1 ET 200 CPU / ET 200 CPU

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-12

Controller 2: ET 200 CPU				**Controller 1: ET 200 CPU to PB**														
				ET200 S						ET 200 Pro								
				CPU			CP			CPU						CP		
				IM151-7 CPU			DP master module			IM154-8 (F) PN/DP CPU						DP master module		
				Slave passive			Master			Master			Slave passive			Master		
				S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7-B**	**S7**	**DP**	**S7 B**	**S7**	**DP**
ET200 S	CPU	IM151-7 CPU	Slave passive	---	---	---	(62)	---	(2)	(62)	---	(2)	---	---	---	(62)	---	(2)
	CP	DP master module	Master	(61)	---	(2)	---	---	---	---	---	---	(61)	---	(2)	---	---	---
ET200 Pro	CPU	IM154-8(F) PN/DP CPU	Master	(61)	---	(2)	---	---	---	---	---	---	(61)	---	(2)	---	---	---
			Slave passive	---	---	---	(62)		(2)	(62)	---	(2)	---	---	---	(62)	---	(2)
	CP	DP master module	Master	(61)	---	(2)	---	---	---	---	---	---	(61)	---	(2)	---	---	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(61) Controller 1 is server (for I_PUT, I_GET)

(62) Controller 2 is server (for I_PUT, I_GET)

20.3.2 ET 200 CPU / S7-300

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-13

Controller 2: S7-300			**Controller 1: ET 200 CPU to PB**															
			ET200 S						ET 200 Pro								
			CPU			CP			CPU						CP		
			IM151-7 CPU			DP master module (*1)			IM154-8 PN/DP CPU						DP master module (*1)		
			Slave passive			Master			Master			Slave passive			Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7-B**	**S7**	**DP**	**S7-B**	**S7**	**DP**
CPU	all with interface: DP, MPI/DP	Master	(61)	---	(2)	---	---	---	---	---	---	(61)	---	(2)	---	---	---
		Slave passive	---	---	---	(62)	---	(2)	(62)	---	(2)	---	---	---	(62)	---	(2)
CP	342-5	no DP	---	---	---	---	(21)	---	---	(21)	---	---	---	---	---	(21)	---
		Master	---	---	(2) / (7)	---	(21)	---	---	(21)	---	---	---	(2) / (7)	---	(21)	---
		Slave passive	---	---	---	---	---	(2) / (7)	---	---	(2) / (7)	---	---	---	---	---	(2) / (7)
	343-5	no DP	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(7) DP_SEND, DP_RECV

(21) Controller 1 is server (for PUT, GET)

(61) Controller 1 is server (for I_PUT, I_GET)

(62) Controller 2 is server (for I_PUT, I_GET)

Explanations on the table

(*1): The DP master module behaves like a CPU interface: Load/transfer commands, DPRD_DAT, DPWR_DAT

20.3.3 ET 200 CPU / S7-400

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-14

Controller 2: S7-400			**Controller 1: ET 200 CPU to PB**														
			ET200 S						ET 200 Pro								
			CPU			CP			CPU						CP		
			IM151-7 CPU			DP master module			IM154-8 PN/DP CPU						DP master module		
			Slave passive			Master			Master			Slave passive			Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7-B**	**S7**	**DP**
CPU	all with interface: DP, MPI/DP	Master	(61)	---	(2)	---	(21)	---	---	(21)	---	(61)	---	(2)	---	(21)	---
		Slave passive	---	---	---	(62)	---	(2)	(62)	---	(2)	---	---	---	(62)	---	(2)
CP	443-5 Basic	no DP	---	---	---	---	(21)	---	---	(21)	---	---	---	---	---	(21)	---
	443-5 Ext.	no DP	---	---	---	---	(21)	---	---	(21)	---	---	---	---	---	(21)	---
		Master	---	---	(2)	---	(21)	---	---	(21)	---	---	---	(2)	---	(21)	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21)Controller 1 is server (for PUT, GET)

(61) Controller 1 is server (for I_PUT, I_GET)

(62) Controller 2 is server (for I_PUT, I_GET)

20.3.4 ET 200 CPU / S7-mEC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-15

Controller 2: S7-mEC				**Controller 1: ET 200 CPU to PB**														
				ET200 S						ET 200 Pro								
				CPU			CP			CPU						CP		
				IM151-7 CPU			DP master module			IM154-8 PN/DP CPU						DP master module		
				Slave passive			Master			Master			Slave passive			Master		
				S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**
CP	EM PCI-104	Control of RTX (submodules): CP5603	Master	---	---	(2)	---	(21)	---	---	(21)	---	---	---	(2)	---	(21)	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.3.5 ET 200 CPU / Box PC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-16

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: ET 200 CPU to PB**														
			ET200 S						ET 200 Pro								
			CPU			CP			CPU						CP		
			IM151-7 CPU			DP master module			IM154-8 PN/DP CPU						DP master module		
			Slave passive			Master			Master			Slave passive			Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**
CPU/CP	Control of RTX (submodules): • „CP5611-CP5621” • “CP5613-CP5603”	Master	---	---	(2)	---	(21)	---	---	(21)	---	---	---	(2)	---	(21)	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.3.6 ET 200 CPU / Panel PC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-17

Controller 2: Panel PC HMI IPC477C bundles with RTX			**Controller 1: ET 200 CPU to PB**														
			ET200 S						ET 200 Pro								
			CPU			CP			CPU						CP		
			IM151-7 CPU			DP master module			IM154-8 PN/DP CPU						DP master module		
			Slave passive			Master			Master			Slave passive			Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	---	---	(2)	---	(21)	---	---	(21)	---	---	---	(2)	---	(21)	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.3.7 ET 200 CPU / WinAC MP

In the following table it is assumed that the DP slave interface is "passive". If the DP slave interface is operated as "active", then there are additional communication options (see chapter 20.2).

Table 20-18

Controller 2: **WinAC MP for Multipanel** MP177, MP277, MP377			**Controller 1: ET 200 CPU to PB**														
			ET200 S						ET 200 Pro								
			CPU			CP			CPU						DP		
			IM151-7 CPU			DP master module			IM154-8 PN/DP CPU						DP master module		
			Slave passive			Master			Master			Slave passive			Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7 B**	**S7**	**DP**
CPU	Control of WinAC MP: integrated PB interface	Master	---	---	(2)	---	(21)	---	---	(21)	---	---	---	(2)	---	(21)	---

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.3.8 ET 200 CPU / WinAC RTX

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-19

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC RTX on PC
WinAC RTX 2009</td><td colspan="15">Controller 1: ET 200 CPU to PB</td></tr>
<tr><td colspan="6">ET200 S</td><td colspan="9">ET 200 Pro</td></tr>
<tr><td colspan="3">CPU</td><td colspan="3">CP</td><td colspan="6">CPU</td><td colspan="3">CP</td></tr>
<tr><td colspan="3">IM151-7 CPU</td><td colspan="3">DP master module</td><td colspan="6">IM154-8 PN/DP CPU</td><td colspan="3">DP master module</td></tr>
<tr><td colspan="3">Slave passive</td><td colspan="3">Master</td><td colspan="3">Master</td><td colspan="3">Slave passive</td><td colspan="3">Master</td></tr>
<tr><td>S7 B</td><td>S7</td><td>DP</td><td>S7 B</td><td>S7</td><td>DP</td><td>S7 B</td><td>S7</td><td>DP</td><td>S7 B</td><td>S7</td><td>DP</td><td>S7 B</td><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>---</td><td>---</td><td>(2)</td><td>---</td><td>(21)</td><td>---</td><td>---</td><td>(21)</td><td>---</td><td>---</td><td>---</td><td>(2)</td><td>---</td><td>(21)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.4 PB: Controller 1 = S7-300

20.4.1 (S7-300 / ET 200 CPU)

See ET 200 CPU / S7-300: 20.3.2

20.4.2 S7-300 / S7-300

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-20

Controller 2: S7-300			**Controller 1: S7-300 to PB**																
			CPU						CP										
			all with interface: DP, MPI/DP						342-5								343-5		
			Master			Slave passive			no DP		Master			Slave passive			no DP		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CPU	all with interface: DP, MPI/DP	Master	---	---	---	(61)	---	(2)	(22)	---	(22)	---	---	---	---	(7) / (2)	---	---	---
		Slave passive	(62)	---	(2)	---	---	---	---	---	---	---	(7) / (2)	---	---	---	---	---	---
CP	342-5	no DP	---	(21)	---	---	---	---	(1)	(8)	(1)	(8)	---	---	---	---	(21)	(8)	---
		Master	---	(21)	---	---	---	(2) / (7)	(1)	(8)	(1)	(8)	---	---	---	(7)	(21)	(8)	---
		Slave passive	---	---	(2) / (7)	---	---	---	---	---	---	---	(7)	---	---	---	---	---	---
	343-5	no DP	---	---	---	---	---	---	(22)	(8)	(22)	(8)	---	---	---	---	---	(8)	(10)

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET
(2) Load/transfer commands, DPRD_DAT, DPWR_DAT
(7) DP_SEND, DP_RECV
(8) AG_SEND/AG_RECV
(10) READ, WRITE, REPORT
(21) Controller 1 is server (for PUT, GET)
(22) Controller 2 is server (for PUT, GET)
(61) Controller 1 is server (for I_PUT, I_GET)
(62) Controller 2 is server (for I_PUT, I_GET)

20.4.3 S7-300 / S7-400

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-21

Controller 2: S7-400			**Controller 1: S7-300 to PB**																
			CPU						CP										
			all with interface: DP, MPI/DP						342-5								343-5		
			Master			Slave passive			no DP		Master			Slave passive			no DP		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CPU	all with interface: DP, MPI/DP	Master	---	(21)	---	(61)	---	(2)	(1)	---	(1)	---	---	---	---	(7)/(2)	(21)	---	---
		Slave passive	(62)	---	(2)	---	---	---	---	---	---	---	(7)/(2)	---	---	---	---	---	---
CP	443-5 Basic	no DP	---	(21)	---	---	---	---	(1)	(8)	(1)	(8)	---	---	---	---	(21)	(8)	(10)
	443-5 Ext.	no DP	---	(21)	---	---	---	---	(1)	(8)	(1)	(8)	---	---	---	---	(21)	(8)	---
		Master	---	(21)	---	---	---	(2)	(1)	(8)	(1)	(8)	---	---	---	(7)/(2)	(21)	(8)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(7) DP_SEND, DP_RECV

(8) AG_SEND/AG_RECV

(10) READ, WRITE, REPORT

(21) Controller 1 is server (for PUT, GET)

(61) Controller 1 is server (for I_PUT, I_GET)

(62) Controller 2 is server (for I_PUT, I_GET)

20.4.4 S7-300 / S7-mEC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-22

Controller 2: S7-400				**Controller 1: S7-300 to PB**																
				CPU						CP										
				all with interface: DP, MPI/DP						342-5								343-5		
				Master			Slave passive			no DP		Master			Slave passive			no DP		
				S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CP	EM PCI-104	Control of RTX (submodules): CP5603	Master	---	(21)	---	---	---	(2)	(1)	---	(1)	---	---	---	---	(2)	(21)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.4.5 S7-300 / Box PC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-23

Controller 2: S7-400 IPC427C bundles with RTX			**Controller 1: S7-300 to PB**																
			CPU						CP										
			all with interface: DP, MPI/DP						342-5								343-5		
			Master			Slave passive			no DP		Master			Slave passive			no DP		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	---	(21)	---	---	---	(2)	(1)	---	(1)	---	---	---	---	(2)	(21)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.4.6 S7-300 / Panel PC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-24

Controller 2: S7-400 HMI IPC477C bundles with RTX			**Controller 1: S7-300 to PB**																
			CPU						CP										
			all with interface: DP, MPI/DP						342-5								343-5		
			Master			Slave passive			no DP		Master			Slave passive			no DP		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	---	(21)	---	---	---	(2)	(1)	---	(1)	---	---	---	---	(2)	(21)	---	---

[Back to jump distributor PB](#)

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.4.7 S7-300 / WinAC MP

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-25

Controller 2: S7-400 MP177, MP277, MP377			**Controller 1: S7-300 to PB**																
			CPU						CP										
			all with interface: DP, MPI/DP						342-5								343-5		
			Master			Slave passive			no DP		Master			Slave passive			no DP		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CPU	Control of WinAC MP: integrated PB interface	Master	---	(21)	---	---	---	(2)	(1)	---	(1)	---	---	---	---	(2)	(21)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.4.8 S7-300 / WinAC RTX

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-26

Controller 2: S7-400 WinAC RTX 2009			**Controller 1: S7-300 to PB**																
			CPU						CP										
			all with interface: DP, MPI/DP						342-5								343-5		
			Master			Slave passive			no DP		Master			Slave passive			no DP		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**S7**	**OC**	**DP**	**S7**	**OC**	**DP**	**S7**	**OC**	**FMS**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	---	(21)	---	---	---	(2)	(1)	---	(1)	---	---	---	---	(2)	(21)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(21) Controller 1 is server (for PUT, GET)

20.5 PB: Controller 1 = S7-400

20.5.1 (S7-400 / ET 200 CPU)

See ET 200 CPU / S7-400: 20.3.3

20.5.2 (S7-400 / S7-300)

See S7-300 / S7-400: 20.4.3

20.5.3 S7-400 / S7-400

In the following table it is assumed that the DP slave interface is "passive". If the DP slave interface is operated as "active", then there are additional communication options (see chapter 20.2).

Table 20-27

Controller 2: S7-400			**Controller 1: S7-400 to PB**													
			CPU						CP							
			all with interface: DP, MPI/DP						443-5 Basic			443-5 Ext.				
			Master			Slave passive			no DP			no DP		Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**FMS**	**S7**	**OC**	**S7**	**OC**	**DP**
CPU	all with interface: DP, MPI/DP	Master	---	(1)	---	(21)	---	(2)	(1)	---	---	(1)	---	(1)	---	---
		Slave passive	(22)	---	(2)	---	---	---	---	---	---	---	---	---	---	(2)
CP	443-5 Basic	no DP	---	(1)	---	---	---	---	(1)	(8)	(10)	(1)	(8)	(1)	(8)	---
	443-5 Ext.	no DP	---	(1)	---	---	---	---	(1)	(8)	---	(1)	(8)	(1)	(8)	---
		Master	---	(1)	---	---	---	(2)	(1)	(8)	---	(1)	(8)	(1)	(8)	---

[Back to jump distributor PB]

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

(7) DP_SEND, DP_RECV

(8) AG_SEND/AG_RECV

(10) READ, WRITE, REPORT

(21) Controller 1 is server (for PUT, GET)

(22) Controller 2 is server (for PUT, GET)

20.5.4 S7-400 / S7-mEC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-28

Controller 2: S7-400				**Controller 1: S7-400 to PB**													
				CPU						CP							
				all with interface: DP, MPI/DP						443-5 Basic			443-5 Ext.				
				Master			Slave passive			no DP			no DP		Master		
				S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**FMS**	**S7**	**OC**	**S7**	**OC**	**DP**
CP	EM PCI-104	Control of RTX (submodules): CP5603	Master	---	(1)	---	---	---	(2)	(1)	---	---	(1)	---	(1)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

20.5.5 S7-400 / Box PC

In the following table it is assumed that the DP slave interface is "passive". If the DP slave interface is operated as "active", then there are additional communication options (see chapter 20.2).

Table 20-29

Controller 2: S7-400 IPC427C bundles with RTX			**Controller 1: S7-400 to PB**													
			CPU						CP							
			all with interface: DP, MPI/DP						443-5 Basic			443-5 Ext.				
			Master			Slave passive			no DP			no DP		Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**FMS**	**S7**	**OC**	**S7**	**OC**	**DP**
CPU/CP	Control of RTX (submodules): • "CP5611-CP5621" • "CP5613-CP5603"	Master	---	(1)	---	---	---	(2)	(1)	---	---	(1)	---	(1)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

20.5.6 S7-400 / Panel PC

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-30

<table>
<tr><td colspan="3" rowspan="5">Controller 2: S7-400
HMI IPC477C bundles with RTX</td><td colspan="14">Controller 1: S7-400 to PB</td></tr>
<tr><td colspan="6">CPU</td><td colspan="8">CP</td></tr>
<tr><td colspan="6">all with interface: DP, MPI/DP</td><td colspan="3">443-5 Basic</td><td colspan="5">443-5 Ext.</td></tr>
<tr><td colspan="3">Master</td><td colspan="3">Slave passive</td><td colspan="3">no DP</td><td colspan="2">no DP</td><td colspan="3">Master</td></tr>
<tr><td>S7 B</td><td>S7</td><td>DP</td><td>S7 B</td><td>S7</td><td>DP</td><td>S7</td><td>OC</td><td>FMS</td><td>S7</td><td>OC</td><td>S7</td><td>OC</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>---</td><td>(1)</td><td>---</td><td>---</td><td>---</td><td>(2)</td><td>(1)</td><td>---</td><td>---</td><td>(1)</td><td>---</td><td>(1)</td><td>---</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

20.5.7 S7-400 / WinAC MP

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-31

Controller 2: S7-400 MP177, MP277, MP377			**Controller 1: S7-400 to PB**														
			CPU						CP							
			all with interface: DP, MPI/DP						443-5 Basic			443-5 Ext.				
			Master			Slave passive			no DP			no DP		Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**FMS**	**S7**	**OC**	**S7**	**OC**	**DP**
CPU	Control of WinAC MP: integrated PB interface	Master	---	(1)	---	---	---	(2)	(1)	---	---	(1)	---	(1)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

20.5.8 S7-400 / WinAC RTX

In the following table it is assumed that the DP slave interface is “passive”. If the DP slave interface is operated as “active”, then there are additional communication options (see chapter 20.2).

Table 20-32

Controller 2: S7-400 WinAC RTX 2009			**Controller 1: S7-400 to PB**													
			CPU						CP							
			all with interface: DP, MPI/DP						443-5 Basic			443-5 Ext.				
			Master			Slave passive			no DP			no DP		Master		
			S7 B	**S7**	**DP**	**S7 B**	**S7**	**DP**	**S7**	**OC**	**FMS**	**S7**	**OC**	**S7**	**OC**	**DP**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	---	(1)	---	---	---	(2)	(1)	---	---	(1)	---	(1)	---	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(2) Load/transfer commands, DPRD_DAT, DPWR_DAT

20.6 PB: Controller 1 = S7-mEC

20.6.1 (S7-mEC / ET 200 CPU)

See ET 200 CPU / S7-mEC: 20.3.4

20.6.2 (S7-mEC / S7-300)

See S7-300 / S7-mEC: 20.4.4

20.6.3 (S7-mEC / S7-400)

See S7-400 / S7-mEC: 20.5.4

20.6.4 S7-mEC / S7-mEC

Table 20-33

Controller 2: S7-mEC				**Controller 1: S7-mEC to PB**	
				CP	
				EM PCI-104	
				Control of RTX (submodules): CP5603	
				Master	
				S7	**DP**
CP	EM PCI-104	Control of RTX (submodules): CP5603	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.6.5 S7-mEC / Box PC

Table 20-34

<table>
<tr><td colspan="3" rowspan="6">Controller 2: Box PC
IPC427C bundles with RTX</td><td colspan="2">Controller 1: S7-mEC to PB</td></tr>
<tr><td colspan="2">CP</td></tr>
<tr><td colspan="2">EM PCI-104</td></tr>
<tr><td colspan="2">Control of RTX (submodules): CP5603</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.6.6 S7-mEC / Panel PC

Table 20-35

<table>
<tr><td colspan="3" rowspan="6">Controller 2: Panel PC
HMI IPC477C bundles with RTX</td><td colspan="2">Controller 1: S7-mEC to PB</td></tr>
<tr><td colspan="2">CP</td></tr>
<tr><td colspan="2">EM PCI-104</td></tr>
<tr><td colspan="2">Control of RTX (submodules): CP5603</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.6.7 S7-mEC / WinAC MP

Table 20-36

Controller 2: WinAC MP for Multipanel MP177, MP277, MP377			Controller 1: S7-mEC to PB	
			CP	
			EM PCI-104	
			Control of RTX (submodules): CP5603	
			Master	
			S7	**DP**
CPU	Control of WinAC MP: integrated PB interface	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.6.8 S7-mEC / WinAC RTX

Table 20-37

Controller 2: WinAC RTX WinAC RTX 2009			**Controller 1: S7-mEC to PB**	
			EM PCI-104	
			Control of RTX (submodules): CP5603	
			EM PCI-104	
			Master	
			S7	**DP**
CPU/CP	Control of RTX (submodules): • "CP5611-CP5621" • "CP5613-CP5603"	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.7 PB: Controller 1 = Box PC

20.7.1 (Box PC / ET 200 CPU)

See ET 200 CPU / Box PC: 20.3.5

20.7.2 (Box PC / S7-300)

See S7-300 / Box PC: 20.4.5

20.7.3 (Box PC / S7-400)

See S7-400 / Box PC: 20.5.5

20.7.4 (Box PC / S7-mEC)

See S7-mEC / Box PC: 20.6.5

20.7.5 Box PC / Box PC

Table 20-38

Controller 2: Box PC IPC427C bundles with RTX			**Controller 1: Box PC to PB** IPC427C bundles with RTX CPU/CP Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603” Master	
			S7	**DP**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • CP5613-CP5603“	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.7.6 Box PC / Panel PC

Table 20-39

<table>
<tr><td colspan="3" rowspan="6">Controller 2: Panel PC
HMI IPC477C bundles with RTX</td><td colspan="2">Controller 1: Box PC to PB</td></tr>
<tr><td colspan="2">IPC427C bundles with RTX</td></tr>
<tr><td colspan="2">CPU/CP</td></tr>
<tr><td colspan="2">Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.7.7 Box PC / WinAC MP

Table 20-40

Controller 2: WinAC MP MP177, MP277, MP377			**Controller 1: Box PC to PB** IPC427C bundles with RTX CPU/CP Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603” Master	
			S7	**DP**
CPU	Control of WinAC MP: integrated PB interface	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.7.8 Box PC / WinAC RTX

Table 20-41

Controller 2: WinAC RTX WinAC RTX 2009			**Controller 1: Box PC to PB**	
			IPC427C bundles with RTX	
			CPU/CP	
			Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	
			Master	
			S7	**DP**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.8 PB: Controller 1 = Panel PC

20.8.1 (Panel PC / ET 200 CPU)

See ET 200 CPU / Panel PC: 20.3.6

20.8.2 (Panel PC / S7-300)

See S7-300 / Panel PC: 20.4.6

20.8.3 (Panel PC / S7-400)

See S7-400 / Panel PC: 20.5.6

20.8.4 (Panel PC / S7-mEC)

See S7-mEC / Panel PC: 20.6.6

20.8.5 (Panel PC / Box PC)

See Box PC / Panel PC: 20.7.6

20.8.6 Panel PC / Panel PC

Table 20-42

<table>
<tr><td colspan="3" rowspan="6">Controller 2: Panel PC
HMI IPC477C bundles with RTX</td><td colspan="2">Controller 1: Panel PC to PB</td></tr>
<tr><td colspan="2">HMI IPC477C bundles with RTX</td></tr>
<tr><td colspan="2">CPU/CP</td></tr>
<tr><td colspan="2">Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.8.7 Panel PC / WinAC MP

Table 20-43

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC MP
MP177, MP277, MP377</td><td colspan="2">Controller 1: Panel PC to PB</td></tr>
<tr><td colspan="2">HMI IPC477C bundles with RTX</td></tr>
<tr><td colspan="2">CPU/CP</td></tr>
<tr><td colspan="2">Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU</td><td>Control of WinAC MP:
integrated PB interface</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.8.8 Panel PC / WinAC RTX

Table 20-44

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC RTX
WinAC RTX 2009</td><td colspan="2">Controller 1: Panel PC to PB</td></tr>
<tr><td colspan="2">HMI IPC477C bundles with RTX</td></tr>
<tr><td colspan="2">CPU/CP</td></tr>
<tr><td colspan="2">Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.9 PB: Controller 1 = WinAC MP

20.9.1 (WinAC MP / ET 200 CPU)

See ET 200 CPU / WinAC MP: 20.3.7

20.9.2 (WinAC MP / S7-300)

See S7-300 / WinAC MP: 20.4.7

20.9.3 (WinAC MP / S7-400)

See S7-400 / WinAC MP: 20.5.7

20.9.4 (WinAC MP / S7-mEC)

See S7-mEC / WinAC MP: 20.6.7

20.9.5 (WinAC MP / Box PC)

See Box PC / WinAC MP: 20.7.7

20.9.6 (WinAC MP / Panel PC)

See Panel PC / WinAC MP: 20.8.7

20.9.7 WinAC MP / WinAC MP

Table 20-45

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC MP
MP177, MP277, MP377</td><td colspan="2">Controller 1: WinAC MP to PB</td></tr>
<tr><td colspan="2">MP177, MP277, MP377</td></tr>
<tr><td colspan="2">CPU</td></tr>
<tr><td colspan="2">Control of WinAC MP:
integrated PB interface</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU</td><td>Control of WinAC MP:
integrated PB interface</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.9.8 WinAC MP / WinAC RTX

Table 20-46

Controller 2: WinAC RTX WinAC RTX 2009			**Controller 1: WinAC MP to PB** MP177, MP277, MP377 CPU Control of WinAC MP: integrated PB interface Master	
			S7	**DP**
CPU/CP	Control of RTX (submodules): • “CP5611-CP5621” • “CP5613-CP5603”	Master	(1)	---

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.10 PB: Controller 1 = WinAC RTX

20.10.1 (WinAC RTX / ET 200 CPU)

See ET 200 CPU / WinAC RTX: 20.3.8

20.10.2 (WinAC RTX / S7-300)

See S7-300 / WinAC RTX: 20.4.8

20.10.3 (WinAC RTX / S7-400)

See S7-400 / WinAC RTX: 20.5.8

20.10.4 (WinAC RTX / S7-mEC)

See S7-mEC / WinAC RTX: 20.6.8

20.10.5 (WinAC RTX / Box PC)

See Box PC / WinAC RTX: 20.7.8

20.10.6 (WinAC RTX / Panel PC)

See Panel PC / WinAC RTX: 20.8.8

20.10.7 (WinAC RTX / WinAC MP)

See WinAC MP / WinAC RTX: 20.9.8

20.10.8 WinAC RTX / WinAC RTX

Table 20-47

<table>
<tr><td colspan="3" rowspan="6">Controller 2: WinAC RTX
WinAC RTX 2009</td><td colspan="2">Controller 1: WinAC RTX to PB</td></tr>
<tr><td colspan="2">WinAC RTX 2009</td></tr>
<tr><td colspan="2">CPU/CP</td></tr>
<tr><td colspan="2">Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td></tr>
<tr><td colspan="2">Master</td></tr>
<tr><td>S7</td><td>DP</td></tr>
<tr><td>CPU/CP</td><td>Control of RTX (submodules):
• “CP5611-CP5621”
• “CP5613-CP5603”</td><td>Master</td><td>(1)</td><td>---</td></tr>
</table>

Back to jump distributor PB

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

20.11 PB: Overview of communication types

Comparison of all communication types with PB.

The data is taken from the Communication types - details table:

- S7 basic communication (Table 28-1)
- S7 communication (Table 29-1)
- Open communication with send/receive blocks (Table 36-1)
- FMS communication (Table 37-1)
- DP communication (Table 38-1)

Communication types – compact table:

Table 20-48

	SIMATIC-specific		Open standard		DP communication
	S7 basic communication	**S7 communication**	**Open communication**	**FMS communication**	
Protocols	S7 (DP)	S7 (FDL)	FDL	FMS	DP
Interfaces	CPU	CPU, CP	CP	CP	CPU, CP
Communication blocks (max. data)	I_PUT (= 84 bytes) I_GET (= 94 bytes)	BSEND (<= 64 Kbytes) USEND (>= 160 bytes) PUT, GET (>= 160 bytes)	AG_SEND (=240 bytes) AG_LSEND (=240 bytes)	READ (<= 237 bytes) WRITE (<= 233 bytes) REPORT (<= 233 bytes)	Load commands/ transfer commands (1, 2, 4 bytes) DPR_DAT, DPWR_DAT (<= 64 words) DP_SEND, DP_RECV (<= 244 bytes)
Remote confirmation	Application	BSEND: application USEND: transport PUT, GET: application	Transport	READ: application WRITE: application REPORT: no	Application
Routing-capable?	no	no	no	no	no
Connections?	yes	yes	yes	yes	no

Back to jump distributor PB

21 Selection Aid: MPI (MPI)

21.1 MPI: Content of the chapter

The following is described for the MPI medium:

- which interfaces (modules) and communication types are available?
  (-> Interfaces table)
- which partners can communicate via which communication types?
  (-> Combinations table)
- overview of all available communication types
  (-> Communication types – compact table)

21.2 MPI: Interfaces and communication types

21.2.1 ET 200 CPU to MPI

Table 21-1

Controller to MPI: ET 200 CPU			**Communication type**		
			SIMATIC-specific		
			Global data	**S7 basic communication**	**S7 communication**
ET 200 S	CPU	IM151-7 CPU	cyclic	(11)	PUT, GET, Server, (S7)
ET 200 Pro	CPU	IM154-8(F) PN/DP CPU	cyclic	(11)	PUT, GET, Server, (S7)

Back to jump distributor MPI

Communication blocks

(11) X_SEND/X_RCV, X_PUT, X_GET

21.2.2 S7-300 to MPI

Table 21-2

Controller to MPI: S7-300		Communication type		
		SIMATIC-specific		
		Global data	S7 basic communication	S7 communication
CPU	all with interface: MPI, MPI/DP (*2)	cyclic	(11)	PUT, GET, Server, (S7)

Back to jump distributor MPI

Communication blocks

(11) X_SEND/X_RCV, X_PUT, X_GET

Explanations on the table

(*2): MPI/DP interface in MPI operating mode

21.2.3 S7-400 to MPI

Table 21-3

Controller to MPI: S7-400		**Communication type**		
		SIMATIC-specific		
		Global data	**S7 basic communication**	**S7 communication**
CPU	all with interface: MPI/DP (*2)	cyclic acyclic (12)	(11)	(1)

Back to jump distributor MPI

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(11) X_SEND/X_RCV, X_PUT, X_GET

(12) GD_SND/GD_RCV

Explanations on the table

(*2): MPI/DP interface in MPI operating mode

21.3 MPI: Controller 1 = ET 200 CPU

21.3.1 ET 200 CPU / ET 200 CPU

Table 21-4

Controller 2: ET 200 CPU		**Controller 1: ET 200 CPU to MPI**		
		CPU		
		IM151-7 CPU, IM154-8(F) PN/DP CPU		
		GD	**S7 Basis**	**S7**
CPU	151-7, 154-8	cyclic	(11)	---

Back to jump distributor MPI

Communication blocks

(11) X_SEND/X_RCV, X_PUT, X_GET

21.3.2 ET 200 CPU / S7-300

Table 21-5

Controller 2: S7-300		**Controller 1: ET 200 CPU to MPI**		
		CPU		
		IM151-7 CPU, IM154-8(F) PN/DP CPU		
		GD	**S7 Basis**	**S7**
CPU	all with interface: MPI, MPI/DP	cyclic	(11)	---

Back to jump distributor MPI

Communication blocks

(11) X_SEND/X_RCV, X_PUT, X_GET

21.3.3 ET 200 CPU / S7-400

Table 21-6

Controller 2: S7-400		**Controller 1: ET 200 CPU to MPI**		
		CPU		
		IM151-7 CPU, IM154-8(F) PN/DP CPU		
		GD	**S7 Basis**	**S7**
CPU	all with interface: MPI/DP	cyclic + C2 acyclic (12)	(11)	(21)

Back to jump distributor MPI

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(11) X_SEND/X_RCV, X_PUT, X_GET

(12) GD_SND/GD_RCV

(21) Controller 1 Server (for PUT, GET)

21.4 MPI: Controller 1 = S7-300

21.4.1 (S7-300 / ET 200 CPU)

See ET 200 CPU / S7-300: 21.3.2

21.4.2 S7-300 / S7-300

Table 21-7

Controller 2: S7-300		**Controller 1: S7-300 to MPI**		
		CPU		
		all with interface: MPI, MPI/DP		
		GD	**S7 Basis**	**S7**
CPU	all with interface: MPI, MPI/DP	cyclic	(11)	---

Back to jump distributor MPI

Communication blocks

(11) X_SEND/X_RCV, X_PUT, X_GET

21.4.3 S7-300 / S7-400

Table 21-8

Controller 2: S7-400		**Controller 1: S7-300 to MPI**		
		CPU		
		all with interface: MPI, MPI/DP		
		GD	**S7 Basis**	**S7**
CPU	all with interface: MPI/DP	cyclic + C2 acyclic (12)	(11)	(21)

Back to jump distributor MPI

Communication blocks

(11) X_SEND/X_RCV, X_PUT, X_GET

(12) GD_SND/GD_RCV

(21) Controller 1 Server (for PUT, GET)

21.5 MPI: Controller 1 = S7-400

21.5.1 (S7-400 / ET 200 CPU)

See ET 200 CPU / S7-400: 21.3.3

21.5.2 (S7-400 / S7-300)

See S7-300 / S7-400: 21.4.3

21.5.3 S7-400 / S7-400

Table 21-9

Controller 2: S7-400		**Controller 1: S7-400 to MPI**		
		CPU		
		all with interface: MPI/DP		
		GD	**S7 Basis**	**S7**
CPU	all with interface: MPI/DP	cyclic + acyclic (12)	(11)	(1)

Back to jump distributor MPI

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(11) X_SEND/X_RCV, X_PUT, X_GET

(12) GD_SND/GD_RCV

21.6 MPI: Overview of communication types

Comparison of all communication types with MPI.

The data is taken from the Communication types - details table:

- Global data communication (Table 27-1)
- S7 basic communication (Table 28-1)
- S7 communication (Table 29-1)

Communication types – compact table:

Table 21-10

	SIMATIC-specific		
	Global data communication	**S7 basic communication**	**S7 communication**
Protocols	S7	S7	S7 (FDL)
Interfaces	CPU	CPU	CPU, CP
Communication blocks (max. data)	cyclic: no communication blocks -- acyclic: GD_SND, GD_RCV (22 bytes per GD package)	X_PUT (= 76 bytes) X_GET (= 76 bytes)	BSEND (<= 64 Kbytes) USEND (>= 160 bytes) PUT, GET (>= 160 bytes)
Remote confirmation	no	Application	BSEND: application USEND: transport PUT, GET: application
Routing-capable?	no	no	no
Connections?	no	yes	yes

Back to jump distributor MPI

22 Selection Aid: SIMATIC Backplane Bus

22.1 Content of the chapter

The following is described for the SIMATIC backplane bus medium:

- which interfaces (modules) are available?
- which partners can communicate via which communication types? (*1)
- overview of all available communication types

(*1) Discussed combinations controller 1 / controller 2:

For S7-400, up to 4 CPUs can be operated in one single controller (multi-computing, chapter 4.5). I.e. in the following tables there is no differentiation between controller 1 and controller 2. The CPUs are in the same controller!

22.2 Interfaces and communication types

Table 22-1

CPU at backplane bus: S7-400		**Communication type**	
		SIMATIC	
		Global data communication	**S7 communication**
CPU	all	+ acyclic	(1)

Back to jump distributor Backplane bus

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

22.3 Controller 1 = S7-400 / Controller 2 = S7-400

Table 22-2

Controller: S7-400		**Controller: S7-400**	
		CPU	
		all	
		GD	**S7**
CPU	all	cyclic, acyclic (12)	(1)

Back to jump distributor Backplane bus

Communication blocks

(1) USEND/URCV, BSEND/BRCV, PUT, GET

(12) GD_SND/GD_RCV

22.4 Overview of communication types

Comparison of all communication types available via the SIMATIC backplane bus.

The data is taken from the Communication types - details table:

- Global data communication (Table 27-1)
- S7 basic communication (Table 28-1)

Table 22-3

Communication type	SIMATIC	
	Global data communication	**S7 communication**
Protocols	S7 protocol	S7 protocol
Interfaces	CPU	CPU
Communication blocks (max. data)	Cyclic transmission: no communication blocks necessary acyclic transmission GD_SND, GD_RCV (54 bytes per GD package)	BSEND (<= 64 Kbytes) USEND (>= 160 bytes) PUT, GET (>= 160 bytes)
Remote confirmation:	none	BSEND: application USEND: transport PUT, GET: application
Routing-capable?	---	---
Connections?	no	yes

Back to jump distributor Backplane bus

23 Selection Aid: Serial Interface (PtP)

23.1 Content of the chapter

This chapter describes:

- what options the SIMATIC families offer for communication via a “serial interface”?
- the combination of SIMATIC families (such as for PN/IE, PB and MPI networks) is not discussed here.

Discussed interfaces:

- Modular controllers: ET 200 CPU, S7-300, S7-400, S7-1200
- Embedded controllers: S7-mEC
- Decentralized station

23.2 ET 200 CPU to PtP

Table 23-1

Controller to PtP: ET 200 CPU			Protocol		maximal data volume	Physics (max. distance communication partner)	Transmission rate
			Type	integrated on CP / loadable			
ET 200S	CP	Modbus/USS module	Modbus serial (RTU format), Modbus master and Modbus slave USS protocol (*1)	integrated	224 bytes	RS 232C (15 m) RS 422/485 (1200 m)	110 Bit/s up to 115.2 Kbit/s
		1SI module	3964(R) ASCII	integrated			

Additional option: in decentralized ET 200 stations serial interface modules can be used. See 23.11

Back to jump distributor SERIAL

Explanations on the table

(*1): not further discussed in the document since no CPU-CPU communication can be realized with it.

23.3 S7-300 to PtP

Table 23-2

<table>
<tr><th colspan="2" rowspan="2">Controller to PtP:
S7-300</th><th colspan="2">Protocol</th><th rowspan="2">maximum volume of data</th><th rowspan="2">Physics (max. distance communication partner)</th><th rowspan="2">Transmission rate</th></tr>
<tr><th>Type</th><th>integrated on CP/ loadable</th></tr>
<tr><td rowspan="2">CPU</td><td>313C-2 PtP</td><td>3964(R), ASCII</td><td rowspan="2">integrated</td><td rowspan="2">1024 bytes</td><td rowspan="2">RS 422/485 (1200 m)</td><td rowspan="2">300 Bits/s
up to 38.4 Kbit/s</td></tr>
<tr><td>314C-2 PtP</td><td>3964(R), ASCII, RK512</td></tr>
<tr><td rowspan="3">CP</td><td>340 (*2)</td><td>3964(R), ASCII
printer (*3)</td><td>integrated</td><td>1024 bytes</td><td>RS 422/485 (1200 m)
RS 232C (15 m)
20mA TTY (active: 100 m, passive 1000 m)</td><td>2.4 Kbit/s
up to 19.2 Kbit/s</td></tr>
<tr><td rowspan="2">341 (*2)</td><td>3964R, ASCII, RK512
printer (*3)</td><td>integrated</td><td>4096 bytes</td><td rowspan="2">RS 422/485 (1200m)
RS 232C (15m)
20mA TTY (active and passive 1000m)</td><td rowspan="2">300 Bit/s
up to 115.2 Kbit/s

at 20mA:
up to 19.2 Kbit/s</td></tr>
<tr><td>Modbus serial (RTU format), Modbus master and Modbus slave</td><td>loadable</td><td>dependent on function code (*1)</td></tr>
</table>

Additional option: in decentralized ET 200 stations serial interface modules can be used. See 23.11

Back to jump distributor SERIAL

Explanations on the table

(*1): see respective manuals.

(*2): select the physics of the interface via the module variant

(*3): not further discussed in the document since no CPU-CPU communication can be realized with it.

23.4 S7-400 to PtP

Table 23-3

<table>
<tr><th colspan="2" rowspan="2">Controller to PtP:
S7-400</th><th colspan="2">Protocols</th><th rowspan="2">max. data volume</th><th rowspan="2">Physics (max. distance communication partner)</th><th rowspan="2">Transmission rate</th></tr>
<tr><th>Type</th><th>loadable to CP / integrated</th></tr>
<tr><td rowspan="4">CP</td><td>440</td><td>3964, ASCII</td><td>integrated</td><td>400 bytes</td><td>RS 422/485 (1200m)</td><td>300 Bit/s
up to 115.2 kBit/s</td></tr>
<tr><td>441-1 (*1)</td><td>3964, ASCII
printer (*2)</td><td>integrated</td><td rowspan="2">ASCCII, 3964: 4096 bytes

RK 512, send: 4096 bytes
RK 512, fetch: 450 bytes</td><td rowspan="3">RS 232C (10m)
RS 422/485 (1200m)
20mA-TTY (1000m)</td><td rowspan="3">300 Bit/s
up to 115.2 Kbit/s

for 20mA-TTY:
up to 19.2 Kbit/s</td></tr>
<tr><td rowspan="2">441-2 (*1)</td><td>3964, ASCII, RK512
printer (*2)</td><td>integrated</td></tr>
<tr><td>Modbus serial (RTU format),
Modbus master and Modbus slave</td><td>loadable</td><td>dependent on function code (*3)</td></tr>
</table>

Additional option: in decentralized ET 200 stations serial interface modules can be used. See 23.11

Back to jump distributor SERIAL

Explanations on the table

(*1): selecting the physics of the interface via the plug-in IF module

(*2): not further discussed in the document since no CPU-CPU communication can be realized with it.

(*3): see respective manuals.

23.5 S7-1200 to PtP

Table 23-4

Controller to PtP: S7-1200		Protocol Type	integrated on CP/ loadable	max. data volume	Physics (max. distance communication partner)	Transmission rate
CP	CM 1241	USS drive protocol (*2) User-defined protocol Modbus serial (RTU format), Modbus Master and Slave	integrated	1024 bytes	RS 485 (1000 m) RS 232 (10 m) (*1)	300 Bit/s up to 57.6 Kbit/s

Back to jump distributor SERIAL

Explanations on the table

(*1): selecting the physics: various module variants

(*2): not further discussed in the document since no CPU-CPU communication can be realized with it.

23.6 S7-mEC to PtP

Table 23-5

Controller to PtP: S7-mEC		**Protocol**		**max. data volume**	**Physics (max. distance communication partner)**	**Transmission rate**
		Type	**integrated on CP/ loadable**			
CP	CP 340	(*1)	(*1)	(*1)	(*1)	(*1)

Additional option: in decentralized ET 200 stations serial interface modules can be used. See 23.11.

Back to jump distributor SERIAL

Explanations on the table

(*1): See S7-300 / CP 340: chapter 23.3.

23.7 Box PC to PtP

In decentralized ET 200 stations serial interface modules can be used. See 23.11.

Back to jump distributor SERIAL

23.8 Panel PC to PtP

In decentralized ET 200 stations serial interface modules can be used. See 23.11.

Back to jump distributor SERIAL

23.9 WinAC MP to PtP

In decentralized ET 200 stations serial interface modules can be used. See 23.11.

Back to jump distributor SERIAL

23.10 WinAC RTX to PtP

In decentralized ET 200 stations serial interface modules can be used. See 23.11.

Back to jump distributor SERIAL

23.11 Decentralized station

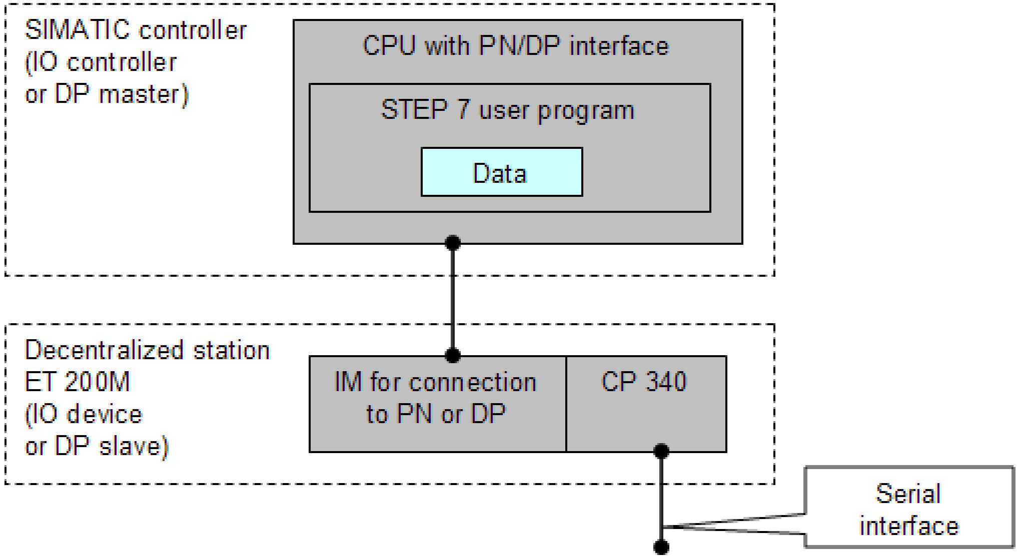
In decentralized ET 200 stations serial interface modules can be used.

Table 23-6

Serial interface module	Decentralized station	Medium
1-SI Modul	ET 200S	PROFINET, PROFIBUS
CP340, CP341	ET 200M	PROFINET, PROFIBUS

This results in further options for a CPU-CPU communication via serial interface. All controllers with a PN or DP interface can use this option as IO controller or DP master. The figure shows one example.

Figure 23-1



[Back to jump distributor SERIAL](#)

24 Information Part 2

The tables contain references to information on the topics in Part 2.

All references /x/ are stored centrally in chapter 47. There the respective links to the internet are also available.

24.1 Communication types

Table 24-1

Reference	Title / content	Information on
/7/	CPU 31xC and CPU 31 x technical data Device manual	Communication via: MPI, PB, IE
/8/	Automation system S7-400 CPU data Device manual	
/9/	Configuring and commissioning S7-CPs for Industrial Ethernet	
/10/	Configuring and commissioning S7-CPs for PROFIBUS	
/19/	Establishing and parameterizing point-to-point connection CP 340	Communication via: serial interface
	Establishing and parameterizing point-to-point connection CP 341	
/20/	Establishing and parameterizing point-to-point connection CP 440	
	Establishing and parameterizing point-to-point connection CP 441	
/21/	S7-300 CPU 31xC technological functions (CPU 312C, CPU 313C, CPU 314C)	
/14/	Function blocks, examples and manuals of the serial interface ET200S 1SI	
/28/	Manual ET 200S serial interface modules	
/31/	Configuration software “PtP-Param”	
/23/	SIMATIC S7-300/S7-400 Loadable driver for point-to-point CPs: Modbus protocol, RTU format, S7 is slave Operating instructions	
/24/	SIMATIC S7-300/S7-400 Loadable driver for point-to-point CPs: Modbus protocol, RTU format, S7 is master Operating instructions	

24.2 Specific application examples

Table 24-2

/x/	Title / content
/200/	Applications on the topic of communication
/25/	SIMATC Net CD

24.3 Performance data

Table 24-3

/x/	Title / content
/18/	Performance data on the communication

In the planning phase of a plant it is important to know the transmission time for CPU-CPU communication in a PROFIBUS, PROFINET IO or Industrial Ethernet network.

To make reliable statements on practice relevant plants with different topologies, we have setup and measured around 10,000 configurations.

Entry /18/ contains various tools with easy-to-operate user interfaces. This enables optimal plant design regarding the communication behavior as well as comparing different plant configurations.

Examples:

- Transmission time for typical configurations on Industrial Ethernet
- PN response time for typical configurations in the PROFITNET IO
- DP response time, HMI update time and HMI operating time for typical configurations on the non-clocked PROFIBUS DP

PART 3: Communication types

Part 3 provides in-depth information

Detailed information on all communication types

PART 3: Structure and content

Table 24-4

Chapter	Structure			Content
26	SIMATIC S7-specific communication			Overview
27		Global data communication		Description of the communication types
28		S7 basic communication		
29		S7 communication		
30	PROFINET/Industrial Ethernet			Overview
31		Open communication with send/receive blocks		Description of the communication types
32		Open communication with T blocks		
---		PN communication		
33			CBA	
34			PNIO	
35	PROFIBUS			Overview
36		Open communication with send/receive blocks		Description of the communication types
37		FMS communication		
38		DP communication		
39	Serial interface			Overview
40		ASCII, 3964(R, RK 512		Description of the communication types
41		User-defined protocol		
42	Information			Device manuals, FAQs, applications

Connecting to controllers with Modbus interface is described in Part 4:

Table 24-5

Chapter	Structure	Content
44	Modbus/TCP	Communication via PN/IE network
45	Modbus serial (RTU format)	Communication via serial interface

25 Preliminary Remarks

For each communication type there are the following chapters:

- Characteristics
- Properties (Communication types – detailed table)
- Application
- Overview user interfaces
- User interfaces (description and overview of the parameters)

The contents of these chapters is described below.

25.1 Characteristics

This chapter gives a brief characterization of the communication type.

No details are given here, only keywords (highlights).

25.2 Properties (Communication types – detailed table)

This chapter contains a table named Communication types – detailed. The most important properties of a communication type are described in this table.

25.2.1 Purpose of this table

The table answers the following questions:

- What are the properties of this communication type?
- What are the properties of the user interfaces (communication blocks) of this communication type?

The table is the basis for a further table named Communication types – compact (17.4). In the Communication types – compact table all communication types of a medium (PN/IE, ...) are compared.

25.2.2 Structure of the table

The structure is explained by means of an example:

- PN/IE medium
- Communication type open communication with T blocks.

The figure shows an extract of the table (Table 32-3).

Figure 25-1

1

2

Communication type:		Open communication with T blocks		
Protocol:		ISO on TCP		TCP
General information				
Media		PN/IE		PN/IE
Interfaces		CPU, CP		CPU, CP
Connection	SIMATIC S5	yes		yes
	third-party (open standards)	yes		yes
User interface				
Communication blocks		**TSEND / TRCV**	**TSEND_C / TRCV_C**	**TSEND / TRCV**
max data volume (*1)		<= 32 Kbytes	= 8192 bytes	<= 64 Kbytes
Number of variables when calling the communication block		1	1	1
dynamic addressing of data		yes	yes	yes
remote addressing		transport	transport	transport
Model		client / client	client / client	client / client

The table consists of several areas, which are explained below.

Area 1: communication type / protocol

Designation of the communication type and the possible protocols.

Area 2: criteria

The criteria are summarized under the following headings:

- General information
- User interface
- Protocol
- User activity

The criteria are described in following chapter (chapter 25.2.3).

25.2.3 Explanation of the criteria

A subset of these criteria appears in the Communication types - compact table. These are underlined (example: Interfaces).

Table 25-1

Criterion		Description	Value range (*1)
General information			
Media		Here the media are entered via which both communication partners exchange data.	PN/IE, PB, MPI, backplane bus, Serial interface
Interfaces		The module the media are connected to is entered here. CPU: integrated interface CP: external interface	CPU, CP
Connection	SIMATIC S5	Communication with SIMATIC S5 possible?	yes / no
	third-party	Is it possible to communicate with third-party controllers via open standards ?	yes / no
User interface			
Communication blocks		All function blocks (FB, SFB, FC, SFC) available for data transmission are listed here (communication blocks). Function blocks which possibly might be required for the connecting and disconnecting process are not mentioned here (connection blocks). Meaning of the notation: a / b: data transmission requires both blocks (example: BSEND / BRCV) a, b: each individual block can perform a data transmission (example: PUT, GET)	BSEND / BRCV AG_SEND / AG_RECV PUT, GET etc.
maximal data volume		The maximum data volume that can be transferred at once by a communication block is entered here: From “trigger job”, to “job finished” Meaning of the notation: = x bytes: always x bytes (under all circumstances) <= x bytes: x bytes at the most, however it can also be less (depending on CPU, CP, ...) >= x bytes: x bytes at least, however it can also be more (depending on CPU, CP, ...)	= x bytes <= x bytes >= x bytes

Criterion	Description	Value range (*1)
Number of variables when calling the communication block	It is specified here how many different data areas (variables) can be configured at the communication block by the user. (transfer area).	1 1 to 4
Dynamic addressing of data	Is it possible to change the addressing of data areas during runtime (in RUN of the CPU)?	yes, no
remote confirmation:	Here it is specified whether a data transmission of remote communication partners (CPU, CP) is confirmed and what the confirmation means. Case discrimination: no confirmation from remote (“no”) • These data were sent and have left the local partner. • The sender is not notified whether the data has arrived in the remote user area (in the remote application of the CPU). transport confirmation from remote (“transport”) • Data was sent and received by the remote partner (CPU, CP). • The sender is not notified whethere the data has arrived in the remote user area (in the remote application of the CPU). application confirmation from remote (“application”) • Data was sent and the data has arrived in the remote user area (in the remote application of the CPU).	no, transport, application
Model	Here it is specified which communication model the data transmission is based on. Description of the models: See chapter 50.2.	client / client client / server, S7 only server, master / slave, consumer / provider
Protocol		
dynamic data length	Can the data length be changed at the communication block during runtime (RUN of the CPU)? This is equal to: can the protocol detect beginning and end of the transferred data?	yes, no
multicast / broadcast	Here it is entered whether multicast or broadcast are possible. Multicast: simultaneous sending to several communication partners Broadcast: simultaneous sending to all communication partners	multicast / broadcast

Criterion		Description	Value range (*1)
Connections	to the remote partner?	Details on the connections: see chapter 5. Case discrimination: no: a connection to the remote partner is not established for the communication yes: a connection to the remote partner is established for the communication	no, yes
	dynamic / static?	Case discrimination: dynamic: the connection is cancelled after data transmission. static: the connection remains after data transmission. dynamic + static: both above cases are possible.	dynamic, static, dynamic + static
routing-capable		Can CPU-CPU communication occur beyond network boundaries (via routers)? This criterion is only relevant for PN/IE.	yes / no
User activity			
Communication relationship	establish with	How (using which tool) can a communication relationship be established? In the sense of: which CPU communicates with which CPU?	GD editor, Hardware config, NetPro, iMAP
Connection	Configuration with NetPro	Is the connection configured with NetPro?	yes, no, --- (connection not required)
	Programming in STEP 7?	Are connection blocks necessary for establishing a connection?	yes, no, --- (connection not required)
Data transmission	Communication block in STEP 7?	Are communication blocks necessary for data transmission?	yes / no

Explanations for the table:

(*1: Generally, the following applies for the value range:

"---" means: not relevant / left out / not significant here / not applicable

25.3 Application

In this chapter, the most important steps are shown which the user must perform in order to realize a CPU-CPU communication. The principle procedure is shown here.

Detailed information is available in the following sources:

- STEP 7 online-help
- STEP 7 manuals
- Manuals on SIMATIC controllers
- Manuals on SIMATIC CPs

25.4 Overview of user interfaces

This chapter gives an overview of all communication blocks of the respective communication type. The following questions are answered:

- For which family and interface (CPU, CP) are the communication blocks suitable?
- Where in STEP 7 (library) can these communication blocks be found?

25.5 User interface

The user interfaces of the respective communication types are briefly described:

- Functionality of the communication blocks
- Parameters of the communication blocks

Detailed information is available in:

- STEP 7 online-help
- Manual system and standard functions for S7-300/400 (/6/)

In the document uniform terms are used for the description. The following can be distinguished in both cases:

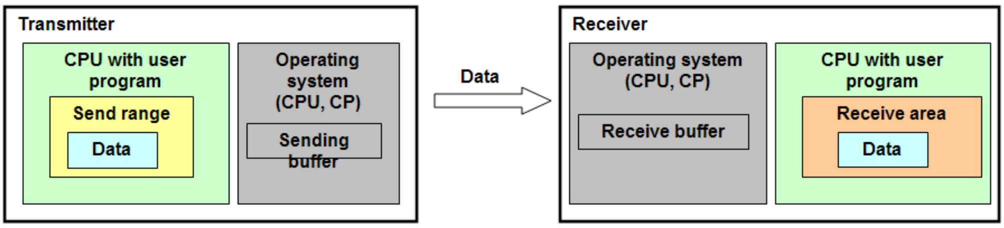
- the partner is not given any address information of the data
- the partner is given the address information of the data

The following figures show the terms used in this context.

No address information of the data

Sending and receiving data

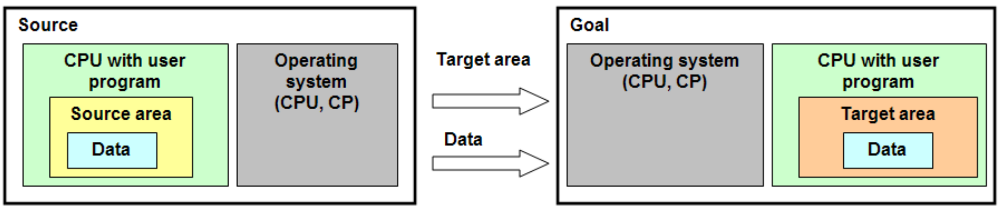
Figure 25-2



Address information of the data

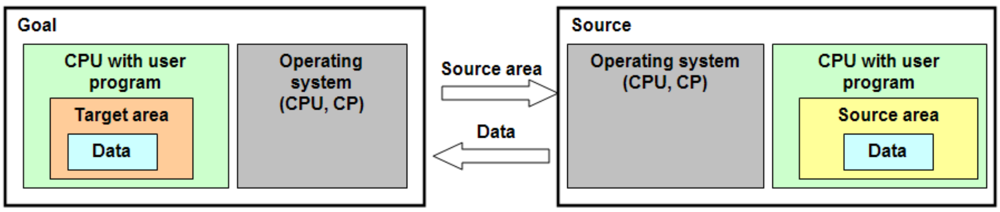
Write data

Figure 25-3



Read data, fetch

Figure 25-4



26 SIMATIC S7-specific communication

26.1 Characteristics

The SIMATIC-specific communication is characterized by the following characteristics:

- Optimized communication between SIMATIC S7 controllers
- Manufacturer-specific communication (no open standard)

26.2 Overview

Two following communication types and media are available:

Table 26-1

Communication type	Media				
	Network			Backplane bus (*1)	Serial interface
	PN/IE	PB	MPI		
Global data communication	---	---	x	x	---
S7 basic communication	---	x (DP)	x	---	---
S7 communication	x	x	x	x	---

(*1): only possible for SIMATIC S7-400 (multi-computing, chapter 4.5)

In the following chapters the communication types are described:

Table 26-2

	Chapter
Global data communication	27
S7 basic communication	28
S7 communication	29

27 Global Data Communication

27.1 Characteristics

Global data

Data exchanged in this form of communication are referred to as global data (GD).

Global data can be:

- Inputs, outputs (process image)
- Memory bits
- Areas of data blocks
- Times, counter

Data exchange occurs cyclic during updating the process image of the inputs and outputs. For S7-400 event-controlled data exchange is possible.

Global data are transferred in GD packages. A GD package is a frame sent from a SIMATIC CPU to one or several other SIMATIC CPUs.

Characteristics

The GD communication is characterized by the following characteristics:

- very simple application
- data volume: <= 54 bytes

27.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 27-1

	Communication type:	GD communication
	Protocol:	**S7 protocol**
General information		
Media		MPI, backplane bus (*1)
Interfaces		CPU
Connection	SIMATIC S5	no
	third-party (open standards)	no
User interface		
Communication blocks		cyclic transmission: no communication blocks necessary acyclic transmission (only S7-400): **GD_SND, GD_RCV**
max data volume		S7-300: data volume per GD package: = 22 bytes
		S7-400: data volume per GD package: = 54 bytes
Number of variables when calling the communication block		---
dynamic addressing of data		no
remote addressing		no
Model		---

Communication type:		GD communication
Protocol:		S7 protocol
Protocol		
dynamic data length		no
Multicast / broadcast		Multicast
Connections	to remote partner?	no
	dynamic / static	---
routing-capable		no
User activity		
Communication relationship.	establish with	GD-Editor
Connections	conf. with NetPro?	---
	prog. in STEP 7?	---
Data transmission	Communication block in STEP 7?	cyclic transmission: no acyclic transmission (only S7-400): yes

Back to jump distributor MPI
Back to jump distributor Backplane Bus

Explanations for the table:

(*1): only possible for SIMATIC S7-400 (Multicomputing, chapter 4.5)

27.3 Application

An overview of the most important activities for realizing a CPU-CPU communication is given below.

Cyclic transmission

Table 27-2

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Configuration of global data groups	STEP 7, GD editor

Acyclic transmission

As above, and additionally:

Table 27-3

Activity	Engineering Tool
Calling communication block in STEP 7 user program	STEP 7, language editor

27.4 Overview of user interfaces

Cyclic transmission

No communication blocks are required.

Acyclic transmission

Overview of communication blocks:

Table 27-4

Communication blocks	S7-400
	CPU
GD_SND	SFC 60
GD_RCV	SFC 61

Communication blocks in STEP 7:

Table 27-5

Interface	STEP 7 library
CPU	Standard Library / System Function Blocks

27.5 User interface GD_SND, GD_RCV

27.5.1 Description

The communication blocks enable an acyclic GD communication between the CPUs of S7-400.

GD_SND

Programmed sending of a GD package

GD_RCV

Programmed receiving of a GD package

27.5.2 Parameters for GD_SND

Table 27-6

INPUT	Type	Remarks
CIRCLE_ID	BYTE	Number of GD group in which the GD package to be send is located.
BLOCK_ID	BYTE	Number of GD packet to be sent in the selected GD circle.
OUTPUT	**Type**	**Remark**
RET_VAL	INT	Error information

27.5.3 Parameters for GD_RCV

Table 27-7

INPUT	Type	Remarks
CIRCLE_ID	BYTE	Number of GD group to which the arrived GD package is to be entered.
BLOCK_ID	BYTE	Number of GD package to which the arrived data is to be entered.
OUTPUT	**Type**	**Remark**
RET_VAL	INT	Error information

28 S7 Basic Communication

28.1 Characteristics

Communication partners

Possible communication partners for a CPU-CPU communication using the S7 basic communication:

- CPU outside the own station (*1)
- CPU within the own station (*1):
  - CPU in the central device or expansion unit
  - CPU decentralized

(*1): Definition “own station”

“Own station” refers to the following:

- central station with CPU, CP, central I/O, and
- decentralized station.

In a decentralized station a CPU can be plugged in. In this case this decentralized station is also referred to as “intelligent slave (I-slave)”.

CPU outside the own station

The respective communication blocks are referred to as “X blocks”.

CPU within the own station

The respective communication blocks are referred to as “I-blocks”.

Data is exchanged between DP master and DP slave:

- With I-block the DB master reads data from the DP slave
- With I-block the DB master writes data to the DP slave

No communication blocks are required in the DP slave.

Characteristics

The S7 basic communication is characterized by the following characteristics:

- Communication only via the integrated interface of the CPU
- Communication via connections. The connections are established by the communication block
  (Configuration is not necessary in STEP 7).
- Data volume per communication job: < 94 bytes
- The user program in the sender contains an item of information if the data in the user data area of the receiver have arrived (confirmation by the remote application).

28.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 28-1

Communication type:		S7 basic communication		
Protocol:		**S7 protocol**		
General information				
Media		PB (DP)	MPI	MPI
Interfaces		CPU	CPU	CPU
Connection	SIMATIC S5	no	no	no
	third-party (open standards)	no	no	no
User interface				
Communication blocks		**I_PUT, I_GET**	**X_PUT, X_GET**	**X_SEND / X_RCV**
max data volume		I_PUT: = 84 bytes I_GET: = 94 bytes	= 76 bytes	= 76 bytes
Number of variables when calling the communication block		1	1	1
dynamic addressing of data		yes	yes	yes
remote addressing		application	application	application
Model		client / server	client / server	client / client

<table>
<tr><th colspan="2">Communication type:</th><th>S7 basic communication</th></tr>
<tr><th colspan="2">Protocol:</th><th>S7 protocol</th></tr>
<tr><td colspan="2">Protocol</td><td></td></tr>
<tr><td colspan="2">dynamic data length</td><td>yes</td></tr>
<tr><td colspan="2">Multicast / broadcast</td><td>no</td></tr>
<tr><td rowspan="2">Connections</td><td>to remote partner?</td><td>yes</td></tr>
<tr><td>dynamic / static</td><td>dynamic + static</td></tr>
<tr><td colspan="2">routing-capable</td><td>no</td></tr>
<tr><td colspan="2">User activity</td><td></td></tr>
<tr><td>Communication relationship</td><td>establish with</td><td>Hardware config</td></tr>
<tr><td rowspan="2">Connections</td><td>conf. with NetPro?</td><td>no</td></tr>
<tr><td>prog. in STEP 7?</td><td>no</td></tr>
<tr><td>Data transmission</td><td>communication block required?</td><td>client: yes
server: no</td></tr>
</table>

Back to jump distributor MPI
Back to jump distributor Backplane Bus
Back to jump distributor PB

28.3 Application

Below is an overview of the most important activities for realizing a CPU-CPU communication. Case discrimination:

Table 28-2

Case	Communication blocks	Communication
X-blocks	X_PUT, X_GET X_SEND/X_RCV	Communication outside the own station
I-blocks	I_PUT, I_GET	Communication within the own station

28.3.1 X-blocks

Client / Client communication

On both sides of the communication relationship:

Table 28-3

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Programming data exchange: calling communication blocks in STEP 7 user program (X_SEND, X_RCV)	STEP 7, language editor

Client / Server communication

On the client side of the communication relationship:

Table 28-4

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Programming data exchange: calling communication blocks in STEP 7 user program (X_PUT, X_GET)	STEP 7, language editor

On the server side of the communication relationship:
As in the above table, however, without “Programming data exchange”.

28.3.2 I-blocks

Client / Server communication

On the client side of the communication relationship:

Table 28-5

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Configuring I-slave (DP-slave with CPU) Configuring DP master, configuring DP slave	
Programming data exchange: calling communication blocks in STEP 7 user program (I_PUT, I_GET)	STEP 7, language editor

On the server side of the communication relationship:
as in the above table, however, without “Programming data exchange”.

28.4 Overview of user interfaces

Overview of communication blocks:

Table 28-6

Communication block		S7-300	S7-400
		CPU	CPU
X-blocks	X_SEND / X_RCV	SFC 65 / SFC 66	SFC 65 / SFC 66
	X_PUT	SFC 68	SFC 68
	X_GET	SFC 67	SFC 67
I-blocks	I_PUT	SFC 73	SFC 73
	I_GET	SFC 72	SFC 72

Communication blocks in STEP 7:

Table 28-7

Interface	STEP 7 library
CPU	Standard Library / System Function Blocks

28.5 User interface X_SEND/ X_RCV

28.5.1 Description

With the communication blocks a CPU communicates with another CPU outside the own station (client / client communication).

X_SEND

Send data to CPU outside the own station

X_RCV

Receive data from CPU outside the own station.

28.5.2 Parameters for X_SEND

Table 28-8

INPUT	Type	Remark
REQ	BOOL	Trigger send job
CONT	BOOL	Connection after completing the job: hold / disconnect
DEST_ID	WORD	MPI address of communication partner
REQ_ID	DWORD	Job ID for receiver
SD	ANY	Send area (*2)
OUTPUT	**Type**	**Remark**
RET_VAL	INT	Error information
BUSY	BOOL	Job running / job finished (*1)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

28.5.3 Parameters for X_RCV

Table 28-9

INPUT	Type	Remarks
EN_DT	BOOL	Trigger receive job
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Error information
REQ_ID	DWORD	Job ID of sender
NDA	BOOL	Data in receive buffer: yes / no
INOUT	**Type**	**Remarks**
RD	ANY	Receive area (*1)

(*1): SIMATIC S7 storage areas: I, Q, M, D

28.6 User interface X_PUT/ X_GET

28.6.1 Description

With the communication blocks a CPU communicates with another CPU outside the own station (client / server communication).

X_PUT

Writing data to CPU outside the own station.

X_GET

Reading data from CPU outside the own station.

28.6.2 Parameters for X_PUT

Table 28-10

INPUT	Type	Remarks
REQ	BOOL	Triggering write job
CONT	BOOL	Connection after completing the job: hold / disconnect
DEST_ID	WORD	MPI address of the communication partner
VAR_ADDR	DWORD	Destination area (*2)
SD	ANY	Source area (*2)
OUTPUT	**Type**	**Remarks**
RET_VAL	RET_VAL	Error information
BUSY	BUSY	Job running /job finished (*1)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

28.6.3 Parameters for X_GET

Table 28-11

INPUT	Data type	Remarks
REQ	BOOL	Triggering read job
CONT	BOOL	Connection after completing the job: hold / disconnect
DEST_ID	WORD	MPI address of the communication partner
VAR_ADDR	DWORD	Source area (*2)
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Error information
BUSY	BOOL	Job running /job finished (*1)
INOUT	**Type**	**Remarks**
RD	ANY	Destination area (I, Q, M, D)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

28.7 User interface I_PUT, I_GET

28.7.1 Description

With the communication blocks a CPU communicates with another CPU within the own station (client / server communication).

I_PUT

Writing data to CPU within the own station.

I_GET

Reading data from CPU within the own station.

28.7.2 Parameters for I_PUT

Table 28-12

INPUT	Type	Remarks
REQ	BOOL	Triggering write job
CONT	BOOL	Connection after completing the job: hold / disconnect
IOID	BYTE	Address area of the partner module (PE, PA)
LADDR	WORD	Logic address of the partner module
VAR_ADDR	ANY	Destination area (*2)
SD	ANY	Source area (*2)
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Error information
BUSY	BOOL	Job running /job finished (*1)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

28.7.3 Parameters for I_GET

Table 28-13

INPUT	Type	Remarks
REQ	BOOL	Triggering read job
CONT	BOOL	Connection after completing the job: hold / disconnect
IOID	BYTE	Address area of the partner module (PE, PA)
LADDR	WORD	Logic address of the partner module
VAR_ADDR	ANY	Source area (*2)
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Error information
BUSY	BOOL	Job running /job finished (*1)
RD	ANY	Destination area (*2)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

29 S7 Communication

29.1 Characteristics

The S7 communication is characterized by the following characteristics:

- Network-independent user interface:
  identical handling for PN/IE, PB and MPI
- Communication via integrated interface of the CPU or via CP
- Communication via connections.
  The connections are configured in STEP 7 (NetPro).
- Data volume per communication job: <= 64 Kbytes
- The user program in the sender contains an item of information if the data in the user data area of the receiver have arrived (confirmation by the remote application).

29.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 29-1

<table>
<tr><th colspan="2">Communication type:</th><th colspan="5">S7 communication</th></tr>
<tr><th colspan="2">Protocol:</th><th colspan="5">S7 protocol (*1)</th></tr>
<tr><td colspan="2">General information</td><td colspan="5"></td></tr>
<tr><td colspan="2">Media</td><td colspan="5">MPI, PB, PN/IE,
backplane bus (*3)</td></tr>
<tr><td colspan="2">Interfaces</td><td colspan="5">CPU, CP</td></tr>
<tr><td rowspan="2">Connection</td><td>SIMATIC S5</td><td colspan="5">no</td></tr>
<tr><td>third-party (open standards)</td><td colspan="5">no</td></tr>
<tr><td colspan="2">User interface</td><td></td><td colspan="2"></td><td colspan="2"></td></tr>
<tr><td colspan="2">Communication blocks</td><td>BSEND / BRCV</td><td colspan="2">USEND / URCV
USEND_E / URCV_E (*4)</td><td colspan="2">PUT, GET
PUT_E, GET_E (*4)</td></tr>
<tr><td colspan="2">max data volume (*2)</td><td><= 64 Kbytes</td><td colspan="2">>= 160 bytes</td><td colspan="2">>= 160 bytes</td></tr>
<tr><td colspan="2" rowspan="3">Number of variables when calling the communication block</td><td rowspan="3">1</td><td rowspan="2">S7-300:</td><td>USEND/URCV: 1</td><td rowspan="2">S7-300:</td><td>PUT, GET: 1</td></tr>
<tr><td>USEND_E/URCV_E: 1 to 4</td><td>PUT_E, GET_E: 1 to 4</td></tr>
<tr><td colspan="2">S7-400: 1 to 4</td><td colspan="2">S7-400: 1 to 4</td></tr>
<tr><td colspan="2" rowspan="2">dynamic addressing of data</td><td>S7-300: yes</td><td colspan="2">S7-300: yes</td><td colspan="2">S7-300: yes</td></tr>
<tr><td>S7-400: no</td><td colspan="2">S7-400: no</td><td colspan="2">S7-400: no</td></tr>
<tr><td colspan="2">remote addressing</td><td>application</td><td colspan="2">transport</td><td colspan="2">application</td></tr>
<tr><td colspan="2">Model</td><td>client / client</td><td colspan="2">client / client</td><td colspan="2">client / server</td></tr>
</table>

Communication type:		S7 communication
Protocol:		**S7 protocol (*1)**
Protocol		
dynamic data length		yes
Multicast / broadcast		no
Connections	to remote partner?	yes
	dynamic / static	static
routing-capable		only for network PN/IE and protocol ISO on TCP
User activity		
Communication relationship	establish with	NetPro
Connections	conf. with NetPro?	yes (server for PUT/GET: no)
	prog. in STEP 7?	no
Data transmission	Communication block necessary?	client: yes server: no

[Back to jump distributor PN/IE]
[Back to jump distributor PB]
[Back to jump distributor MPI]
[Back to jump distributor Backplane Bus]

Explanations for the table:

(*1): The S7 protocol is based on the following protocols:

Table 29-2

Net	Communication via CPU	Communication via CP
PN/IE	ISO on TCP	ISO, ISO on TCP
MPI, PB	FDL	FDL

(*2): the maximal data volume per communication job depends on:

- Network (MPI, PB, PN/IE)
- Interface (communication via CPU or CP)
- Communication partner (S7-300, S7-400, ...)
- Communication block (BSEND, PUT, …)

For some communication blocks it is specified how much data can transferred at a **minimum** (≥ x). If this minimum number is insufficient, the exact maximal number can be determined via /6/ (chapter S7 Communication).

Table 29-3

Communication block	S7-300		S7-400
	CPU	CP	CPU, CP
BSEND / BRCV	= 64 Kbytes	≤ 32 Kbytes	= 64 Kbytes
USEND / URCV	≥ 160 bytes	≤ 160 bytes	≥ 440 bytes
PUT, GET	≥ 160 bytes	≤ 160 bytes	≥ 400 bytes
USEND_E	≥ 160 bytes	not existent	not existent
PUT_E, GET_E	≥ 160 bytes	not existent	not existent

(*3): only possible for SIMATIC S7-400 (Multicomputing, chapter 4.5)

(*4): the communication blocks currently only support connections via the integrated PN interface of the CPU.

29.3 Application

Below is an overview of the most important activities for realizing a CPU-CPU communication.

Client / Client communication

On both sides of the communication relationship:

Table 29-4

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Configuring the connections: • select the modules which are to communicate • select the connection type • configure the connection	STEP 7, NetPro
Programming data exchange: calling communication blocks in STEP 7 user program (BSEND/BRCV, USEND/URCV)	STEP 7, language editor

Client / Server communication

On the client side of the communication relationship:

Table 29-5

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Configuring the connections: • select the modules which are to communicate • select the connection type • configure the connection	STEP 7, NetPro
Programming data exchange: calling communication blocks in STEP 7 user program (PUT, GET)	STEP 7, language editor

On the server side of the communication relationship:

As in the above table, however, without:

- Configuring the connections
- Programming the data exchange

29.4 Overview of user interfaces

Overview of communication blocks:

Table 29-6

Communication block	S7-300, ET 200 CPU		S7-400
	CPU (*1)	CP (*2)	CPU, CP (*3)
USEND / URCV	FB 8 / FB 9	FB 8 / FB 9	SFB 8 / SFB 9
BSEND / BRCV	FB 12 / FB 13	FB 12 / FB 13	SFB 12 / SFB 13
PUT	FB 15	FB 15	SFB 15
GET	FB 14	FB 14	SFB 14
USEND_E / URCV_E	FB 28 / FB 29 (*1)	---	---
PUT_E	FB 35 (*1)	---	---
GET_E	FB 34 (*1)	---	---

(*1): The communication blocks currently only support connections via the PN interface of the S7-300 CPU.

Communication blocks in STEP 7:

Table 29-7

Interface		From the STEP 7 library
S7-300	CPU	Standard Library / Communication Blocks
	CP	SIMATIC_NET_CP / CP300
S7-400	CPU, CP	Standard Library / System Function Blocks

29.5 User interface: USEND / URCV

29.5.1 Description

The communication block USEND sends data to a communication block URCV. URCV receives data from USEND, and copies it to the configured receive areas.

USEND and URCV are not coordinated with each other:

- A send job is complete if data has arrived in the receive buffer. The data is then generally not yet in the receive area. For a completed send job it is therefore not ensured that the data has been copied from the receive buffer to the receive area with URCV.
- A new send job (USEND) can be started, even if the previously sent data has not been copied to the receive area yet by URCV. Data can therefore be overwritten in the receive buffer unnoticed by the sender. In this case the URCV, in the receiver, turns out an Overrun warning.

USEND

Uncoordinated sending of data

URCV

Uncoordinated receiving of data

29.5.2 Parameters for USEND

Table 29-8

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running /job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_i	ANY	Send area (i=1,2,3,4) (*2)

(*1): remote addressing: transport

(*2): Table 29-9

	S7-300	S7-400
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of send areas	1 send area	up to 4 send areas

29.5.3 Parameters for URCV

Table 29-10

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in receive buffer: yes / no
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_i	ANY	Receive area (i=1,2,3,4) (*1)

(*1):Table 29-11

	S7-300	S7-400
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of receive areas	1 receive area	up to 4 receive areas

29.6 User interface: BSEND / BRCV

29.6.1 Description

The communication block BSEND sends data to a communication block BRCV. BRCV receives data from BSEND, and copies it to the configured receive areas.

BSEND and BRCV are coordinated with each other:

- A send job is complete if the data has arrived in the receive area.
- A new send job (BSEND) can only be started if the previously sent data has not been entered to the receive area by BRCV.

<u>Notes on the internal mode of operation</u>

The data to be sent is divided into data blocks. Each data block is sent to the communication partner individually (block-oriented sending). After each received data block a confirmation is sent to BSEND, and parameter LEN is updated (block-oriented receiving).

BSEND

Block-oriented sending of data

BRCV

Block-oriented receiving of data

29.6.2 Parameters for BSEND

Table 29-12

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the send job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running /job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Send area (*2)
LEN	WORD	Length of the data to be sent

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas:

- S7-300: M, D
- S7-400: I, Q, M, D, T, C

29.6.3 Parameters for BRCV

Table 29-13

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in the receive area: yes / no
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_1	ANY	Receive area (*1)
LEN	WORD	Length of the received data

(*1): SIMATIC S7 storage areas

- S7-400: I, Q, M, D, T, C
- S7-300: M, D

29.7 User interface PUT, GET

29.7.1 Description

With the communication blocks a CPU communicates with another CPU (client / server communication).

PUT

Writing data to CPU

GET

Reading data from CPU

29.7.2 Parameters for PUT

Table 29-14

INPUT	Type	Remarks
REQ	BOOL	Triggering write job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running /job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
ADDR_i	ANY	Destination area (i=1,2,3,4) (*2)
SD_i	ANY	Source area (i=1,2,3,4) (*2)

(*1): remote addressing: application

(*2): Table 29-15

	S7-300	S7-400
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of areas	1 area	up to 4 areas

29.7.3 Parameters for GET

Table 29-16

INPUT	Type	Remarks
REQ	BOOL	Triggering read job
ID	WORD	Reference to the respective connection (from the configured connection in STEP 7)
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running /job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
ADDR_i	ANY	Destination area (i=1,2,3,4) (*2) (*3)
RD_i	ANY	Source area (i=1,2,3,4) (*2) (*3)

(*1): remote addressing: application

(*2): Table 29-17

	S7-300	S7-400
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of areas	1 area	up to 4 areas

29.8 User interface: USEND_E / URCV_E

Note
The communication blocks currently only support connections via the PN interface.

29.8.1 Description

See chapter 29.5.1.

29.8.2 Parameters for USEND_E

See chapter 29.5.2.
Differences of the communication blocks:
Table 29-18

	S7-300 CPUs, ET 200 CPUs	
	USEND	USEND_E
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of send areas	1 send area	up to 4 send areas

29.8.3 Parameters for URCV_E

See chapter 29.5.3.
Differences of the communication blocks:
Table 29-19

	S7-300 CPUs, ET 200 CPUs	
	URCV	URCV_E
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of receive areas	1 receive area	up to 4 receive areas

29.9 User interface PUT_E, GET_E

Note
The communication blocks currently only support connections via the PN interface.

29.9.1 Description

See chapter 29.7.1.

29.9.2 Parameters for PUT_E

See chapter 29.7.2.
Differences of the communication blocks:
Table 29-20

	S7-300 CPUs, ET 200 CPUs	
	PUT	**PUT_E**
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of areas	1 area	up to 4 areas

29.9.3 Parameters for GET_E

See chapter 29.7.3
Differences of the communication blocks:
Table 29-21

	S7-300 CPUs, ET 200 CPUs	
	GET	**GET_E**
SIMATIC S7 storage areas	M, D	I, Q, M, D, T, C
Number of areas	1 area	up to 4 areas

30 PROFINET/Industrial Ethernet (PN/IE)

30.1 Characteristics

The communication via PN/IE is characterized by the following characteristics:

- Communication between SIMATIC controllers, and
- Communication with third-party controllers

30.2 Overview

Via PN/IE the following communication types are possible:

Table 30-1

	Communication type		**Chapter**
SIMATIC S7 spezifisch	S7 communication		29
Open standard	Open communication with send/receive blocks		31
	Open communication with T blocks		32
	PN communication	CBA	33
		PNIO	34

The communication types for the “open standard” are discussed below.

Note

The communication with MODICON devices via the PN/IE network is described in Part 4 of the documentation.

Table 30-2

	Communication type	**Chapter**
Coupling via Modbus	Modbus/TCP	44

31 PN/IE: open communication with send/receive blocks

31.1 Characteristics

The open communication with send/receive blocks is characterized by the following characteristics:

- Open standard (communication with third-party controllers possible)
- Communication always via CP
- Communication via different protocols: TCP, ISO on TCP, UDP
- Communication via connections (not with protocol UDP)
- Connections are configured
- Data volume per communication job: <= 8 Kbytes
- Conformation from the remote transport system (not for UDP protocol).
- No confirmation from the remote application (all products)

Configuration of Connections

Connections are configured in STEP 7 (NetPro). Connecting and disconnecting occurs via the operating system of the CP. Due to the configuration of the connections the connection parameters cannot be modified during runtime.

Exceptions for protocol UDP

An “UDP connection” must be configured in STEP 7 (NetPro). However, an “UDP connection” is not a connection in the sense of the document (chapter 5). A connection with the remote communication partner is not established.

Connection types

Table 31-1

Protocol	Connection types	
TCP	B#16#01, B#16#11	to RFC 793
ISO on TCP	B#16#12	to RFC 1006
UDP	B#16#13	to RFC 768

Exceptions for protocol UDP

An “UDP connection” must be configured in STEP 7 (NetPro). However, an “UDP connection” is not a connection in the sense of the document (chapter 5). A connection with the remote communication partner is not established.

Characteristics of protocols

Table 31-2

Protocol	Connecting to the remote partner	Transferring length and end of the data
TCP	yes	no (*1)
ISO on TCP	yes	yes
UDP	no	yes

(*1): This is not a problem during sending, since the sender knows how much data it can send. However, the receiver has no option detecting where the data ends.

31.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 31-3

Communication type:		open communication with send/receive blocks						
Protocol:		**ISO**		**ISO on TCP**		**TCP**		**UDP**
General information								
Media		PN/IE		PN/IE		PN/IE		PN/IE
Interfaces		CP		CP		CP		CP
Connection	SIMATIC S5	yes		yes		yes		yes
	third-party (open standards)	no		yes		yes		yes
User interface								
Communication blocks **(*2)**		AG_xSEND / AG_xREC	FETCH, WRITE	AG_xSEND / AG_xREC	FETCH, WRITE	AG_xSEND / AG_xREC	FETCH, WRITE	AG_xSEND / AG_xREC
max data volume **(*2)**		<= 8 Kbytes		<= 8 Kbytes		<= 8 Kbytes		<= 2 Kbytes
Number of variables when calling the communication block		1	---	1	---	1	---	1
dynamic addressing of data		yes	---	yes	---	yes	---	yes
remote addressing		transport	---	transport	---	transport	---	no
Model		client / client	S7 only server	client / client	S7 only server	client / client	S7 only server	client / client

Communication type:		open communication with send/receive blocks			
Protocol:		**ISO**	**ISO on TCP**	**TCP**	**UDP**
Protocol					
dynamic data length		yes	yes	no	yes
Multicast / broadcast		no	no	no	yes / yes
Connections	to remote partner?	yes	yes	yes	no
	dynamic / static	static	static	static	static
routing-capable		no	yes	yes	yes
User activity					
Communica tion relation-ship	establish with	NetPro	NetPro	NetPro	NetPro
Connections	conf. with NetPro?	yes	yes	yes	yes **(*1)**
	prog. in STEP 7?	no	no	no	no
Data trans-mission	Communication block in STEP 7?	client: yes server: no	client: yes server: no	client: yes server: no	client: yes server: no

Back to jump distributor PN/IE

<u>Explanations for the table:</u>

(*1): An “UDP connection” must be configured in STEP 7 (NetPro). However, an “UDP connection” is not a connection in the sense of the document (chapter 5). A connection with the remote communication partner is not established.

(*2): the maximal data volume depends on:

- Communication partner (S7-300, S7-400, ...)
- Communication block (AG_SEND, AG_LSEND, AG_SSEND)
- Protocol (ISO, ISO on TCP, ...)

Overview:

Table 31-4

Communication block	Protocol	S7-300	S7-400	Remarks
		CP	CP	
AG_SEND / AG_RECV	ISO, TCP, ISO on TCP	8 Kbytes	240 bytes	----
	UDP	2 Kbytes	240 bytes	including IP header and UDP header
AG_LSEND / AG_LRECV	ISO, TCP, ISO on TCP	---	8 Kbytes	---
	UDP	---	2 Kbytes	including IP header and UDP header
AG_SSEND / AG_SRECV	ISO, TCP, ISO on TCP	---	1452 bytes	---
	UDP	---	1452 bytes	including IP header and UDP header

31.3 Application

Below is an overview of the most important activities for realizing a CPU-CPU communication.

Client / Client communication

On both sides of the communication relationship:

Table 31-5

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Configuring the connection: • select the modules which are to communicate • select the connection type • configure the connection	STEP 7, NetPro
Programming data exchange: calling communication blocks in the STEP 7 user program	STEP 7, language editor

Client / Server communication

On the client side of the communication relationship:

SIMATIC S7 controllers are only servers.

On the server side of the communication relationship:

As in the above table, however, without:

- Programming the data exchange

31.4 Overview of user interfaces

Send/Receive blocks

Overview of communication blocks:

Table 31-6

Communication block	S7-300	S7-400
	CP	CP
AG_SEND / AG_RECV	FC 5 / FC 6	FC 5 / FC 6
AG_LSEND / AG_LREC	---	FC 50 / FC 60
AG_SSEND / AG_SREC	---	FC 53 / FC 63

Depending on the family (S7-300, S7-400) different communication blocks must be used. The communication blocks are stored in STEP 7 under various libraries.

Communication blocks in STEP 7:

Table 31-7

Interface	STEP 7 library
S7-300, CP	SIMATIC_NET_CP / CP300
S7-400, CP	SIMATIC_NET_CP / CP400

Server for Fetch/Write

A SIMATIC S7-CP can be server for FETCH/WRITE jobs of another controller (third-party controller, SIMATIC S5).

In the S7-CPU of the server no communication blocks are required for the data exchange.

31.5 User interface AG_xSEND, AG_xRECV

For the names of the communication blocks the following abbreviations are used:

- AG_**x**SEND stands for: AG_SEND, AG_LSEND, AG_SSEND
- AG_**x**RECV stands for: AG_RECV, AG_LRECV, AG_SSRECV

Meaning of x = L

The communication blocks are optimized for transferring extensive data (L stands for “long”).

Meaning of x = s

The communication blocks are time-optimized (S stands for “speed”):

- Optimized communication between CPU and CP in the station
- Without effect on the communication via the network.

31.5.1 Description

The communication block AG_xRECV sends data to the communication block AG_xRECV.

The mode of operation of the communication blocks depends on the used CP (/13/).

AG_xSEND

The communication block transfers data to the CP which are sent via a configured connection.

AG_xRECV

The communication block receives data from the CP which were received via a configured connection.

31.5.2 Parameters for AG_SEND, AG_LSEND, AG_SSEND

Table 31-8

INPUT	Type	Explanation
ACT	BOOL	Triggering send job
ID	INT	Reference to the respective connection (from the configured connection in STEP 7)
LADDR	WORD	Address of the module (from the hardware configuration in STEP 7)
SEND	ANY	Send area (*2)
LEN	INT	Length of the data to be sent
OUTPUT	**Type**	**Explanation**
DONE	BOOL	Job running /job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “job finished”:

Different meaning depending on the used protocol:

- data lie on the remote CP: ISO, ISO on TCP, TCP
- data were sent by the local CP: UDP

(*2): SIMATIC S7 storage areas: M, D

31.5.3 Parameters for AG_RECV, AG_LRECV, AG_SSRECV

Table 31-9

INPUT	Type	Explanation
ID	INT	Reference to the respective connection (from the configured connection in STEP 7)
LADDR	WORD	Address of the module (from the hardware configuration in STEP 7)
RECV	ANY	Receive area (*2)
OUTPUT	**Type**	**Explanation**
NDR	BOOL	Data in the receive area: yes (NDR=1) / no (NDR=0) (*1)
LEN	INT	Length of the received data
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “yes”: data were copied from the receive buffer (CP) into the receive area (CPU).

Exception TCP connection:

Here the length specified in the RECV parameter rules.

A receive job is completed with the NDR=1 display as soon as a data volume equal to the specified length has been written to the receive area. This means, NDR is only set if the receive area has been filled up completely. LEN always shows the length of the receive area.

(*2): SIMATIC S7 storage areas: M, D

31.6 User interface FETCH, WRITE (server)

In the S7-CPU of the server no communication blocks are required for the data exchange.

FETCH (fetch data)

The connection partner (third-party controller, SIMATIC S5) has read access to the data in the S7-CPU (server).

WRITE (write data)

The connection partner (third-party controller, SIMATIC S5) has write access to the data in the SIMATIC S7.

Data

The following data can be accessed in the S7-CPU (server):

- data blocks
- memory bits
- process image of inputs and outputs
- distributed I/O
- counter, times

Connection types

For access with FETCH or WRITE a connection in the S7-CPU (server) must be configured to FETCH passive or WRITE passive mode. The following connection types area possible:

- ISO
- ISO on TCP
- TCP

Access coordination via the STEP 7 user program

The blocks (FC) AG_LOCK and AG_UNLOCK are available for the access coordination. With these blocks the access to data can be coordinated by blocking or enabling the connections.

32 PN/IE: open communication with T blocks

32.1 Characteristics

The open communication with T-blocks is characterized by the following characteristics:

- Open standard (communication with third-party controllers possible)
- Communication via CPU or CP
- Communication via different protocols: TCP, ISO on TCP, UDP
- Communication via connections (not with protocol UDP)
- Connections are programmed
- Data volume per communication job: <= 64 Kbytes
- Conformation from the remote transport system (not for UDP protocol).
- No confirmation from the remote application. (applies for all protocols)

Programming of connections

The connecting and disconnecting process is programmed in the STEP 7 user program. This requires calling particular blocks (connection blocks). Due to the programming of the connections the connection parameters can be modified during runtime (in RUN of the CPU).

Note on S7-1200

There the additional T-compact blocks are available. The connection blocks are integrated in these communication blocks:

Exceptions for protocol UDP:

Here the local communication access point is parameterized with the connection blocks. A connection with the remote communication partner is not established.

Connection types

Table 32-1

Protocol	Connection types	
TCP	B#16#01, B#16#11	to RFC 793
ISO on TCP	B#16#12	to RFC 1006
UDP	B#16#13	to RFC 768

Exceptions for protocol UDP:

Here the local communication access point is parameterized with the connection blocks. A connection with the remote communication partner is not established.

Characteristics of protocols

Table 32-2

Protocol	Connecting to the remote partner	Transferring length and end of the data
TCP	yes	no (*1)
ISO on TCP	yes	yes
UDP	no	yes

(*1): This is not a problem during sending, since the sender knows how much data it can send. However, the receiver has no option detecting where the data ends

32.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 32-3

Communication type:		**Open communication with T blocks**				
Protocol:		**ISO on TCP**		**TCP**		**UDP**
General information						
Media		PN/IE		PN/IE		PN/IE
Interfaces		CPU, CP		CPU, CP		CPU, CP
Connection	SIMATIC S5	yes		yes		no
	third-party (open standards)	yes		yes		yes
User interface						
Communication blocks		**TSEND / TRCV**	**TSEND_C / TRCV_C**	**TSEND / TRCV**	**TSEND_C / TRCV_C**	**TUSEND / TURCV**
max data volume **(*1)**		<= 32 Kbytes	= 8192 bytes	<= 64 Kbytes	= 8192 bytes	= 1472 bytes
Number of variables when calling the communication block		1	1	1	1	1
dynamic addressing of data		yes	yes	yes	yes	yes
remote addressing		transport	transport	transport	transport	no
Model		client / client	client / client	client / client	client / client	client / client

Communication type:		Open communication with T blocks		
Protocol:		**ISO on TCP**	**TCP**	**UDP**
Protocol				
dynamic data length		yes	no	yes
Multicast / broadcast		no	no	no
Connections	to remote partner?	yes	yes	no
	dynamic / static	TSEND/TRCV: dynamic + static	TSEND/TRCV: dynamic + static	dynamic + static
		TSEND_C/TRCV_C: dynamisch	TSEND_C/TRCV_C: dynamisch	
routing-capable		yes	yes	yes
User activity				
Communica-tion relation-ship	establish with	HW Config or devices & networks (S7-1200)	HW Config or devices & networks (S7-1200)	HW Config or devices & networks (S7-1200)
Connections	configure?	no	no	no
	prog. in STEP 7?	yes (connection blocks)	yes (connection blocks)	yes (connection blocks) **(*2)**
Data transmission	Communication block in STEP 7?	yes (communication blocks)	yes (communication blocks)	yes (communication blocks)

Back to jump distributor PN/IE

<u>Explanations for the table:</u>

(*1): the maximal data volume depends on:

- Communication partner (S7-300, S7-400, ...) and used interface (CPU, CP)
- Communication block (TSEND, ...)
- Protocol (TCP, …) and connection type

Overview:

Table 32-4

Communication block	**Protocol**	**Connection type**	**S7-1200**	**ET 200 CPU**	**S7-300**	**S7-400**		**S7-mEC**	**WinAC RTX 2009**
			CPU	**CPU**	**CPU**	**CPU**	**CP**	**CPU**	**CPU / CP**
				151-8, 154-8				**EC31**	**(Submodul)**
TSEND/TRCV	TCP	B#16#01	1460 bytes	1460 bytes	1460 bytes	---	---	---	---
		B#16#11	8 Kbytes	32 Kbytes	32 Kbytes	32 Kbytes	---	8 Kbytes	64 Kbytes
	ISO on TCP	B#16#12	8 Kbytes	32 Kbytes	32 Kbytes	32 Kbytes	1452 bytes	---	---
TUSEND/TURCV	UDP	B#16#13	1472 bytes	1472 bytes	1472 bytes	1472 bytes	---	1472 bytes	1472 bytes

(*2): Here the local communication access point is parameterized with the connection blocks. A connection with the remote communication partner is not established.

32.3 Application

Below is an overview of the most important activities for realizing a CPU-CPU communication.

Case discrimination:

- TSEND/TRCV and TUSEND/TURCV
- TSEND_C/TRCV_C

In all cases this is a client / client communication.

TSEND/TRCV and TUSEND/TURCV

Table 32-5

Activity	Engineering Tool	
	SIMATIC (ohne S7-1200)	**S7-1200**
Networking and address assignment	STEP 7, HW Config	STEP 7 Basic, devices & networks
Programming of connections: • Calling connection blocks in the STEP 7 user program • Storing parameters in a data block for each connection (*1)	STEP 7, language editor	STEP 7 Basic
Programming data exchange: calling communication blocks in the STEP 7 user program	STEP 7, language editor	STEP 7 Basic

(*1): the user is supported by wizards:

- SIMATIC (without S7-1200): Open Communication Wizard /29/
- S7-1200: Wizard in STEP 7 Basic

TSEND_C/TRCV_C

Table 32-6

Activity	Engineering Tool
	S7-1200
Networking and address assignment	STEP 7 Basic, devices & networks
Programming: • calling communication blocks in the STEP 7 user program. the blocks realize the data exchange and the established connection (connection blocks are integrated in the communication blocks). • storing parameters for connections and data transfer in a data block (*1)	STEP 7 Basic

(*1): the user is supported by a wizard in STEP 7 Basic.

32.4 Overview of user interfaces

T-blocks

Overview of communication blocks

Table 32-7

Communication block	S7-300	S7-400	S7-1200
	CPU	CPU, CP	CPU
TSEND / TRCV	FB 63 / FB 64	FB 63 / FB 64	(*1)
TSEND_C / TRCV_C	---	---	(*1)
TUSEND / TURCV	FB 67 / FB 68	FB 67 / FB 68	(*1)

(*1): the blocks are part of the language range of the S7-1200: programming instructions / communication operations / open Ethernet communication

Overview: connection blocks (only for communication via CPU).

Table 32-8

Connection block	S7-300	S7-400	S7-1200
	CPU	CPU	CPU
TCON	FB 65	FB 65	(*1)
TDISCON	FB 66	FB 66	(*1)

(*1): the blocks are part of the language range of the S7-1200: programming instructions / communication operations / open Ethernet communication

Communication blocks in STEP 7:

Table 32-9

Interfaces	STEP 7 library
CPU, CP	Standard Library / Communication Blocks

Server for Fetch/Write

Overview: communication blocks

Table 32-10

Communication block	S7-300	S7-400
	CPU	CPU
FW_TCP	FB 210	FB 210
FW_IOT	FB 220	FB 220

Communication blocks in STEP 7:

Table 32-11

Interfaces	STEP 7 library
CPU, CP	Standard Library / Communication Blocks

<u>Note</u>

A SIMATIC S7-CPU can be server for FETCH/WRITE jobs of another controller (third-party controller, SIMATIC S5).

The communication blocks of the server internally use the T-blocks:

- TSEND / TRCV (sending and receiving of data)
- TCON, TDISCON (connecting and disconnecting)

32.5 User interface TSEND / TRCV

32.5.1 Description

The communication block TSEND sends data to the communication block TRCV.

The data is transferred via a connection which is programmed with connection blocks.

TSEND

Sending data

TRCV

Receiving data

32.5.2 Parameters for TSEND

Table 32-12

INPUT	Type	Explanation
REQ	BOOL	Triggering send job
ID	WORD	Reference to the respective connection
LEN	INT	Length of the data to be sent
OUTPUT	**Type**	**Explanation**
DONE	BOOL	Job running /job finished (*1)
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Explanation**
DATA	ANY	Send area (*2)

(*1): remote addressing: transport

(*2): SIMATIC S7 storage areas: I, Q, M, D

32.5.3 Parameters for TRCV

Table 32-13

INPUT	Type	Remarks
EN_R	BOOL	Trigger: receive job
ID	WORD	Reference to the respective connection
LEN	INT	Length of the receive area
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in the receive area: yes (NDR=1) / no (NDR=0) (*1)
BUSY	BOOL	
ERROR	BOOL	Error display
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
DATA	ANY	Receive area (*2)

(*1): “yes”: case discrimination:

Table 32-14

Protocol	LEN	Specifying the receive area	Receiving data
TCP	= 0	Address: DATA Length: DATA	Directly after receiving “DATA” data, it is copied to the receive area and NDR is set to 1. A maximum of 1472 bytes is received.
	<> 0	Address: DATA Length: LEN	Directly after receiving “LEN” data, it is copied to the receive area and NDR is set to 1.
ISO on TCP	---	Address: DATA Length: DATA	As soon as all data of a job has been received, they are copied to the receive area and NDR is set to 1.

(*2): SIMATIC S7 storage areas: I, Q, M, D

32.6 User interface TUSEND / TURCV

32.6.1 Description

The communication block TUSEND sends data to the communication block TURCV.

The data is transferred via a connection which is programmed with connection blocks.

TUSEND

Sending data

TURCV

Receiving data

32.6.2 Parameters for TUSEND

Table 32-15

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
ID	WORD	Reference to the respective connection
LEN	INT	Length of the data to be sent
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running /job finished (*1)
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
DATA	ANY	Send area (*2)
ADDR	ANY	Address of the receiver

(*1): remote addressing: none

(*2): SIMATIC S7 storage areas: I, Q, M, D

32.6.3 Parameters for TURCV

Table 32-16

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
ID	WORD	Reference to the respective connection
LEN	INT	Length of the receive area
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in the receive area: yes (NDR=1) / no (NDR=0) (*1)
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	
RCVD_LEN	INT	Volume of actually received data.
IN_OUT	**Type**	**Remarks**
DATA	ANY	Receive area (*2)
ADDR	ANY	Address of the sender

(*1): remote addressing: none

(*2): SIMATIC S7 storage areas: I, Q, M, D

32.7 User interface for connection blocks

32.7.1 Description

With connection block TCON a connection is established, with connection block TDISCON a connection is cancelled.

Exceptions for protocol UDP

For UDP the local communication access point is parameterized with the connection blocks. A connection with the remote communication partner is not established.

TCON

Establishing a connection

TDISCON

Terminating a connection

32.7.2 Parameters for TCON

Table 32-17

INPUT	Type	Remarks
REQ	BOOL	Triggering the connecting process
ID	WORD	Reference to the connection
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
CONNECT	ANY	Connection description (UDT 65)

32.7.3 Parameters for TDISCON

Table 32-18

INPUT	Type	Remarks
REQ	BOOL	Triggering the disconnecting process
ID	WORD	Reference to the connection
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	

32.8 User interface TSEND_C / TRCV_C

To distinguish them from the other T-blocks these blocks are referred to as T-compact blocks below.

32.8.1 Description

The communication block TSEND_C sends data to the communication block TRCV_C.

The data is transferred via a connection which is established and cancelled by these communication blocks.

The communication blocks also perform the following function:

- Connecting and disconnecting process
- Sending and receiving data

TSEND_C

Establishes a connection with the partner, sends data and can also terminate the connection again.

TSEND_C connects the functions of TCON, TDISCON and TSEND.

TRCV_C

Establishes a connection with the partner, receives data and can also terminate the connection again.

TRCV_C connects the functions of TCON, TDISCON and TRCV.

32.8.2 Parameters for TSEND_C

Table 32-19

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
CONT	BOOL	Connecting / disconnecting process
LEN	INT	Length of the byte to be sent
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running /job finished (*1)
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
DATA	Variant	Send area (*2)
CONNECT	TCON Parameter	Connection description
COM_RST	BOOL	New start of the block

(*1): remote addressing: transport

(*2): SIMATIC S7 storage areas: I, Q, M, D

32.8.3 Parameters for TRCV_C

Table 32-20

INPUT	Type	Remarks
EN_R	BOOL	Trigger: receive job
CONT	BOOL	Connecting / disconnecting process
LEN	INT	Length of the receive area
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Data in the receive area: yes (DONE=1) / no (DONE=0) (*1)
BUSY	BOOL	
RCVD_LEN	INT	Volume of received data
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
DATA	Variant	Receive area (*2)
CONNECT	TCON Parameter	Connection description
COM_RST	BOOL	New start of the block

(*1): “yes”: case discrimination:

Table 32-21

Protocol	LEN	Specifying the receive area	Receiving data
TCP	= 0	Address: DATA Length: DATA	Directly after receiving “DATA” data, it is copied to the receive area and DONE is set to 1. A maximum of 1472 bytes is received.
	<> 0	Address: DATA Length: LEN	Directly after receiving “LEN” data, it is copied to the receive area and DONE is set to 1.
ISO on TCP	---	Address: DATA Length: DATA	As soon as all data of a job has been received, they are copied to the receive area and DONE is set to 1.

(*2): SIMATIC S7 storage areas: I, Q, M, D

32.9 User interface FETCH, WRITE (Server)

32.9.1 Description

A SIMATIC S7-CPU can be server for FETCH/WRITE jobs of another controller (third-party controller, SIMATIC S5).

The communication blocks of the server internally use the T-blocks:

- TSEND / TRCV (sending and receiving of data)
- TCON, TDISCON (connecting and disconnecting)

FW_TCP

Communication block for server, used protocol is TCP

FW_IOT

Communication block for server, used protocol is ISO on TCP

32.9.2 Parameters for FW_TCP

Table 32-22

INPUT	Type	Remarks
ENABLE	BOOL	Connecting and disconnecting process
CONNECT	ANY	Description of TCP connection
ADDRMODE	INT	Addressing mode S5 or S7
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data of the WRITE job were adopted
ERROR	BOOL	Error display
MODE	BYTE	Performing FETCH or WRITE job
STATUS	WORD	Status display

32.9.3 Parameters for FW_IOT

Table 32-23

INPUT	Type	Remarks
ENABLE	BOOL	Connecting and disconnecting process
CONNECT	ANY	Description of IoT connection
ADDRMODE	INT	Addressing mode S5 or S7
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data of the WRITE job were adopted
ERROR	BOOL	Error display
MODE	BYTE	Performing FETCH or WRITE job
STATUS	WORD	Status display

33 PN/IE: CBA

The following abbreviations are used in the document:

- CBA for PROFINET CBA
- PNIO for PROFINET IO

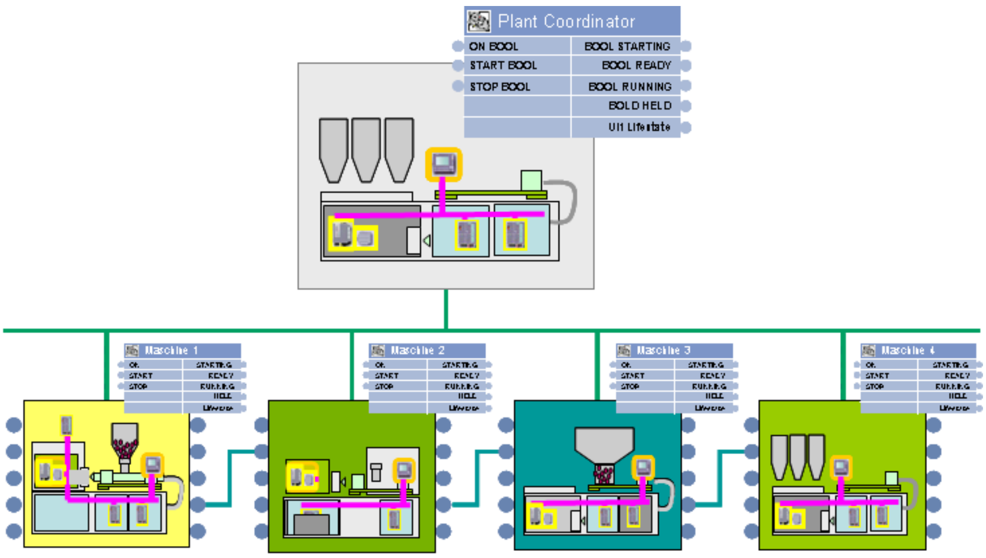
33.1 Characteristics

CBA

CBA (Component Based Automation) is an automation concept for realizing modular applications on the basis of the open PROFINET standard:

- Simple modularization of plants and production lines through decentralized intelligence. Modularization occurs with PROFINET components.
- Machine-machine communication along the production line
- Graphic configuration of the communication

Figure 33-1

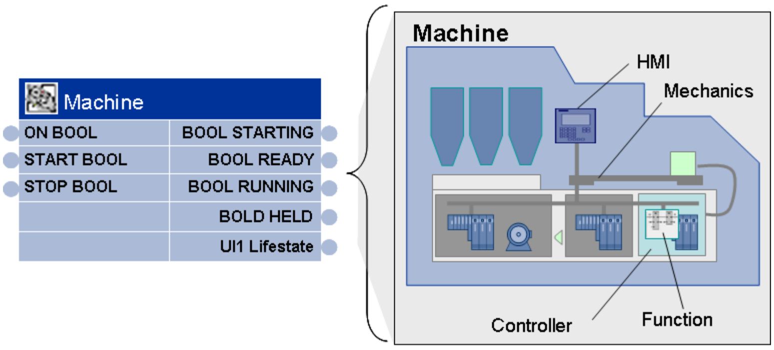


PROFINET component

A PROFINET component is a reusable functional unit:

- encapsulation of the automation functionality in a software program
- unique component interfaces for the data exchange with other components.

Figure 33-2



Characteristics

- Graphic configuration of the communication (configuring instead of programming)
- The performance of the communication can be calculated offline.
- Cross-vendor communication

CBA and PNIO

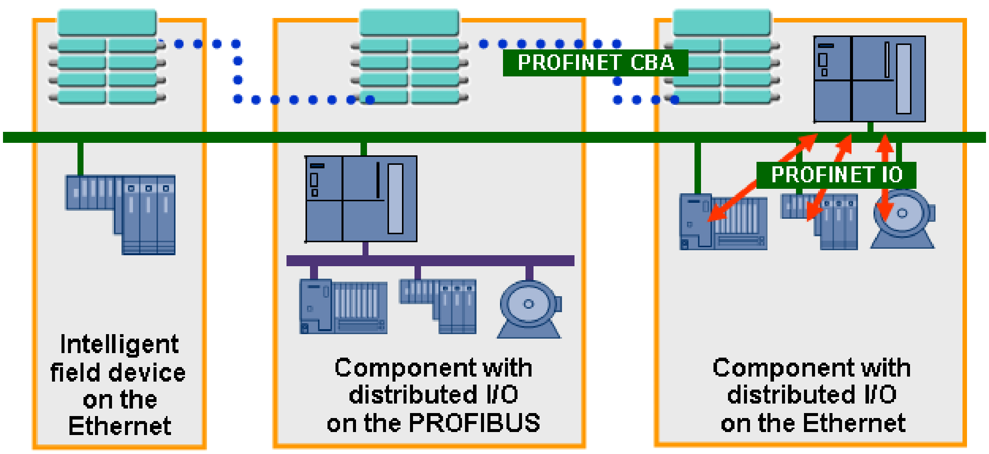
CBA (PROFINET CBA):

- cyclic and acyclic data exchange between controllers

PNIO (PROFINET IO):

- cyclic data exchange between a PN controller and the respective PN devices.

Figure 33-3



33.2 Application

CPU-CPU communication

Below is an overview of the most important activities for realizing a CPU-CPU communication.

Table 33-1

Activity		Engineering Tool
Creating components in STEP 7	Specifying the devices and their functions Creating the hardware configuration	STEP 7, HW Config
	Creating PROFINET interfaces Creating a component	STEP 7, SIMATIC Manager
Importing and using a component in iMAP Graphic interconnection of the components.		SIMATIC iMAP

SIMATIC iMAP

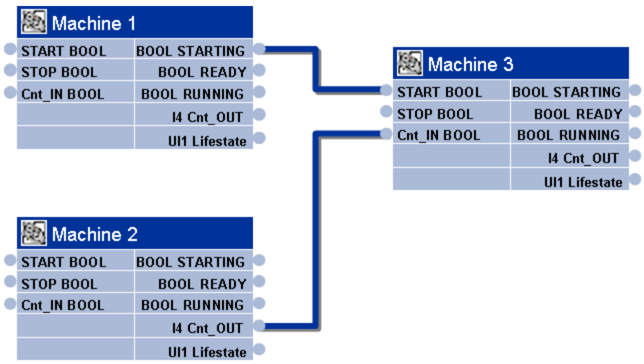
PROFINET components are created and interconnected with SIMATIC iMAP.

PROFINET components do not depend on the manufacturer due to their standardized interface description. The SIMATIC iMap engineering tool therefore also enables importing and managing third-party components.

33.3 User interfaces

The communication between PROFINET components occurs via graphic interconnection. Therefore it is not required to program anything in the STEP 7 user program for the communication, which makes user interfaces unnecessary.

Figure 33-4



<u>Exception: communication via S7-CP.</u>

In this case so-called copy blocks (FB88, FB90) must be called in the STEP 7 user program.

After the installation of SIMATIC iMap in the STEP 7 library "PROFINET System Library" the blocks are available in the folders "CP 300" or "CP 400".

34 PN/IE: PNIO

The following abbreviations are used in the document:

- CBA for PROFINET CBA
- PNIO for PROFINET IO

34.1 Characteristics

Preliminary remarks

The communication with PNIO is a special case of CPU-CPU communication.

Here the communication mechanisms of the “distributed I/O” are used for a CPU-CPU communication:

- a CPU is plugged in at the central station
- the other CPU is plugged in at the decentralized station

Please refer to the functional model: 4.4

Characteristics

The communication with PNIO is characterized by the following characteristics:

- cyclic exchange of data between IO controller
  and IO device via the PROFINET IO protocol:
  - IO controller sends data to IO device (output area)
  - IO device sends data to IO controller (input area)
- The data exchange occurs consistent via the entire length
  (system-related data consistency)

34.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 34-1

<table>
<tr><th colspan="2">Communication type:</th><th colspan="2">PNIO</th></tr>
<tr><th colspan="2">Protocol:</th><th colspan="2">PN</th></tr>
<tr><td colspan="2">General information</td><td colspan="2"></td></tr>
<tr><td colspan="2">Media</td><td colspan="2">PN</td></tr>
<tr><td colspan="2">Interfaces</td><td colspan="2">CPU, CP</td></tr>
<tr><td rowspan="2">Connection</td><td>SIMATICS5</td><td colspan="2">yes</td></tr>
<tr><td>third-party (open standards)</td><td colspan="2">yes</td></tr>
<tr><td colspan="2">User interface</td><td colspan="2"></td></tr>
<tr><td colspan="2" rowspan="3">Communication blocks</td><td rowspan="2">S7-CPU and S7-400 CP:</td><td>Load commands / transfer commands</td></tr>
<tr><td>DPRD_DAT, DPWR_DAT</td></tr>
<tr><td>S7-300 CP:</td><td>PNIO_SEND, PNIO_RECV</td></tr>
<tr><td colspan="2" rowspan="3">max data volume</td><td>Load command / transfer command:</td><td>1, 2, 4 bytes</td></tr>
<tr><td>DPRD_DAT, DPWR_DAT:</td><td rowspan="2">IO controller <= 8192 bytes
IO device <= 1440 bytes</td></tr>
<tr><td>PNIO_SEND, PNIO_REC:</td></tr>
<tr><td colspan="2">Number of variables when calling the communication block</td><td colspan="2">1 variable</td></tr>
<tr><td colspan="2">dynamic addressing of data</td><td colspan="2">no</td></tr>
<tr><td colspan="2">remote addressing</td><td colspan="2">application</td></tr>
<tr><td colspan="2">Model</td><td colspan="2">consumer / provider</td></tr>
</table>

Communication type:		PNIO
Protocol:		PN
Protocol		
dynamic data length		no
Multicast / broadcast		no
Connections	to remote partner?	no
	dynamic / static	---
routing-capable		no
User activity		
Communication relationship	establish with	HW configuration
Connections	conf. with NetPro?	---
	prog. in STEP 7?	---
Data transmission	Communication block in STEP 7?	yes (*1)

Back to jump distributor PN/IE

Explanations for the table:

(*1): for a CPU-CPU communication the communication blocks are only required in one CPU (not in both CPUs)

34.3 Application

Below is an overview of the most important activities for realizing a CPU-CPU communication.

Table 34-2

Activity	Engineering Tool
Hardware configuration of the station: IO controller	STEP 7, HW Config
Hardware configuration of the station: IO device	
Programming the data exchange	STEP 7, language editor

34.4 Overview of user interfaces

Case discrimination

S7-CPUs and S7-400 CPs:

- Load commands and transfer commands
- Communication blocks DPRD_DAT, DPWR_DAT

Only for S7-300 CP:

- Communication blocks PNIO_SEND, PNIO_RECV

Communication blocks

Overview of communication blocks:

Table 34-3

Communication block	S7-300		S7-400
	CPU	CP	CPU, CP
PNIO_SEND	---	FC 11	---
PNIO_RECV	---	FC 12	---
DPRD_DAT	SFC 14	---	SFC 14
DPWR_DAT	SFC 15	---	SFC 15

Communication blocks in STEP 7:

Table 34-4

Interface		STEP 7 library
S7-300	CPU	Standard Library / System Function Blocks
S7-400	CPU, CP	
S7-300	CP	SIMATIC_NET_CP / CP300

34.5 User interface PNIO_SEND, PNIO_RECV

34.5.1 Description

The communication blocks are used for communication between local CPU and local CP. Der CP ist IO Controller oder IO Device

PNIO_SEND

The communication block transfers data to the CP.

Case discrimination:

CP is IO controller:

The data transferred to the CP with PNIO_SEND,
are sent to the IO devices by the CP.

CP is IO device:

The data transferred to the CP with PNIO_SEND,
are sent to an IO controller.

PNIO_RECV

The communication block receives on data from the CP.

Case discrimination:

CP is IO controller:

The data received by the CP with PNIO_RECV,
were sent to the CP by the IO devices.

CP is IO device:

The data received by the CP with PNIO_RECV,
were sent to the CP by an IO controller.

34.5.2 Parameters for PNIO_SEND

Table 34-5

INPUT	Type	Explanation
CPLADDR	WORD	Configured start address of the CP
MODE	BYTE	CP is IO controller or IO device
LEN	INT	Length of the data to be sent
OUTPUT	**Type**	**Explanation**
CHECK_IOCS	BOOL	IOCS status area
IOCS	ANY	
ERROR	BOOL	Error information
STATUS	WORD	
DONE	BOOL	Display whether job was executed without errors (*1).
IN_OUT	**Type**	**Explanation**
SEND	ANY	Send area (*2)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: M, D

34.5.3 Parameters for PNIO_RECV

Table 34-6

INPUT	Type	Explanation
CPLADDR	WORD	Configured start address of the CP
MODE	BYTE	CP is IO controller or IO device
LEN	INT	Length of data to be received
OUTPUT	**Type**	**Explanation**
CHECK_IOPS	BOOL	IOCS status area
IOPS	ANY	
NDR	BOOL	Display whether job was executed without errors (*1).
ERROR	BOOL	Error information
STATUS	WORD	
ADD_INFO	WORD	
IN_OUT	**Type**	**Explanation**
RECV	ANY	Receive area (*2)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: M, D

34.6 User interface DPRD_DAT, DPWR_DAT

34.6.1 Description

With the communication blocks an IO controller has access to the data of an IO device.

DPRD_DAT

Reading consistent data of an IO device

DPWR_DAT

Writing consistent data to IO device

34.6.2 Parameters for DPRD_DAT

Table 34-7

INPUT	Type	Remarks
LADDR	WORD	Configured start address in the receive area to be read from
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Display whether job was executed without errors (*1).
RECORD	ANY	Destination area (*2)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

34.6.3 Parameters for DPWR_DAT

Table 34-8

INPUT	Type	Remarks
LADDR	WORD	Configured start address in the output area to be written to
RECORD	ANY	Source area (*2)
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Display whether job was executed without errors (*1).

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

35 PROFIBUS (PB)

35.1 Characteristics

The communication via PROFIBUS is characterized by the following characteristics:

- Communication between SIMATIC controllers, and
- Communication with third-party controllers

35.2 Overview

Via Pb the following communication types are possible:

Table 35-1

	Communication type	Chapter
SIMATIC S7-specific communication	S7 basic communication	28
	S7 communication	29
Open standard	Open communication with send/receive blocks	36
	FMS communication	37
	DP communication	38

The communication types for the “open standard” are discussed below.

36 PB: open communication with send/receive blocks

36.1 Characteristics

The open communication with send/receive blocks is characterized by the following characteristics:

- open standard (communication with third-party controllers possible)
- communication always via CP
- communication via connections (FDL connections)
- connections are configured
- data volume per communication job: <= 240 bytes
- confirmation of remote transport system
- no confirmation from the remote application

FDL connections

During configuration the following properties can be assigned to an FDL connection:

- unspecified
- specified
- multicast / broadcast

Unspecified

An unspecified FDL connection can be used in 2 ways:

- connection with a station in another STEP 7 project
- open Layer 2 access

Open Layer 2 access

The remote communication partner is not established during configuration but in the STEP 7 user program.

This requires installing a job header (4 bytes) in the data. The job header specifies the address of the destination station, and the service (SDA or SDN).

Specified

For a specified FDL connection the remote communication partner is defined during the configuration.

The data has no job header.

Multicast / broadcast

The data contains a job header.

The job header has no function.

36.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 36-1

Communication type:		Open communication with send/receive blocks
Protocol:		FDL
General information		
Media		PB
Interfaces		CP
Connection	SIMATIC S5	yes
	third-party (open standards)	yes
User interface		
Communication blocks		AG_SEND / AG_REC, AG_LSEND / AG_LREC
maximum data volume (*1)		= 240 bytes
Number of variables when calling the communication block		1 variable
dynamic addressing of data		yes
remote addressing		transport
Model		client / client

Communication type:		Open communication with send/receive blocks
Protocol:		**FDL**
Protocol		
dynamic data length		yes
Multicast / broadcast		yes / yes
Connections	to remote partner?	yes
	dynamic / static	static
routing-capable		no
User activity		
Communica-tion relation-ship	establish with	NetPro
Connections	conf. with NetPro?	specified FDL connection: yes unspecified FDL connection: no
	prog. in STEP 7?	specified FDL connection: no unspecified FDL connection: yes
Data transmission	Communication block in STEP 7?	yes

Back to jump distributor PB

Explanations for the table:

(*1): includes the possibly contained job header (4 bytes).

36.3 Application

An overview of the most important activities for realizing a CPU-CPU communication is given below.

Client / Client communication

On both sides of the communication relationship:

Table 36-2

Activity	Engineering Tool
Networking and address assignment	STEP 7, HW Config
Configuring the connections: • select the modules which are to communicate • select the connection type • connection configuration and service	STEP 7, NetPro
Programming data exchange: calling communication blocks in STEP 7 user program	STEP 7, language editor

36.4 Overview of user interfaces

Overview of communication blocks:

Table 36-3

Communication block	S7-300	S7-400
	CP	CP
AG_SEND / AG_RECV	FC 5 / FC 6	FC 5 / FC 6
AG_LSEND / AG_LREC (*3)	------	FC 50 / FC 60

(*3): AG_LSEND / AG_LREC can be used, however, it has no other function like AG_SEND / AG_RECV

Depending on the family (S7-300, S7-400) different communication blocks must be used. The communication blocks are stored in STEP 7 under various libraries.

Communication blocks in STEP 7:

Table 36-4

Interface	STEP 7 library
S7-300, CP	SIMATIC_NET_CP / CP300
S7-400, CP	SIMATIC_NET_CP / CP400

36.5 User interface: AG_xSEND, AG_xRECV

For the names of the communication blocks the following abbreviations are used:

- AG_**x**SEND stands for: AG_SEND, AG_LSEND
- AG_**x**RECV stands for: AG_RECV, AG_LRECV

Meaning of x = L

The communication blocks are optimized for transferring extensive data (L stands for “long”).

36.5.1 Description

The communication block AG_xRECV sends data to the communication block AG_xRECV.

The mode of operation of the communication blocks depends on the used CP (/13/).

AG_xSEND

The communication block transfers data to the CP which are sent via a configured connection.

AG_xRECV

The communication block receives data from the CP which were received via a configured connection.

36.5.2 Parameters for AG_SEND, AG_LSEND

Table 36-5

INPUT	Type	Explanation
ACT	BOOL	Triggering send job
ID	INT	Reference to the respective connection (from configured connection in STEP 7)
LADDR	WORD	Address of the module (from the hardware configuration inSTEP 7)
SEND	ANY	Send area (*2)
LEN	INT	Length of data to be sent
OUTPUT	**Type**	**Explanation**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “job finished”:

different meaning, depending on the FDL connection type:

- data lies on the remote CP: specified
- data was sent by the local CP:
  - broadcast / multicast
  - unspecified and SDN service

(*2): SIMATIC S7 storage areas: I, Q, M, D

36.5.3 Parameters for AG_RECV, AG_LRECV

Table 36-6

INPUT	Type	Explanation
ID	INT	Reference to the respective connection (from configured connection in STEP 7)
LADDR	WORD	Address of the module (from the hardware configuration inSTEP 7)
RECV	ANY	Receive area (*2)
OUTPUT	**Type**	**Explanation**
NDR	BOOL	Data in receive area: yes (NDR=1) / no (NDR=0) (*1)
LEN	INT	Length of received data
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “yes”: data was copied from the receive buffer (CP) into the receive area (CPU).

(*2): SIMATIC S7 storage areas: I, Q, M, D

37 PB: FMS Communication

37.1 Characteristics

The FMS communication is characterized by the following characteristics:

- Open standard (communication with third-party controllers possible)
- Data is transferred in a device-neutral form (FMS variable). Conversion of FMS variables into device-specific form, and vice versa, occurs in the communication partners.
- Additional configuration workload for defining the FMS variables
- Communication via CP
- Communication via connections
- Connections are configured

37.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 37-1

	Communication type:	**FMS communication**	
	Protocol:	**FMS**	
General information			
Media		PB	
Interfaces		CP	
Connection	SIMATIC S5	yes	
	third-party (open standards)	yes	
User interface			
Communication blocks		READ, WRITE	REPORT
maximum data volume (*1)		READ <= 237 bytes WRITE <= 233 bytes	<= 233 bytes
Number of variables when calling the communication block		1 variable	1 variable
dynamic addressing of data		yes	yes
remote addressing		application	no
Model		client / server	client / server

Communication type:		**FMS communication**
Protocol:		**FMS**
Protocol		
dynamic data length		yes
Multicast / broadcast		no / yes
Connections	to remote partner?	yes
	dynamic / static	static
routing-capable		no
User activity (*3)		
Communication relationship	establish with	NetPro
Connections	conf. with NetPro?	yes
	prog. in STEP 7?	no
Data transmission	Communication block in STEP 7?	yes

Back to jump distributor PB

Explanations for the table:

(*1): contains information which describe the data (FMS variable). Further details: see /12/

(*3): in addition: configuration of FMS variables

37.3 Application

An overview of the most important activities for realizing a CPU-CPU communication is given below.

Table 37-2

Activity		Engineering Tool
Networking and address assignment		STEP 7, HW Config
Configuring FMS variable	Creating the communication objects	STEP 7, DB editor
	Specifying the communication objects	STEP 7, symbolic editor
Configuring the connections: • select the modules which are to communicate • select the connection type • connection configuration and service		STEP 7, NetPro
Programming data exchange: calling communication blocks in the STEP 7 user program		STEP 7, language editor

37.4 Overview of user interfaces

Overview: communication blocks

Table 37-3

Communication block	S7-300	S7-400
	CP	CP
READ	FB 3	FB 3
WRITE	FB 6	FB 6
REPORT	FB 4	FB 4

Depending on the family (S7-300, S7-400) different communication blocks must be used. The communication blocks are stored in STEP 7 under various libraries.

Communication blocks in STEP 7:

Table 37-4

Interface	STEP 7 library
S7-300, CP	SIMATIC_NET_CP / CP300
S7-400, CP	SIMATIC_NET_CP / CP400

37.5 User interface: READ

37.5.1 Description

With the communication block, data is read from the remote communication partner.

The structure description of the FMS variable lies in the remote communication partner (FMS server). When establishing the FMS connection, the local communication partner reads the structure description from the remote communication partner, which the local communication partner uses to convert the data accordingly.

READ

Reading the variable

37.5.2 Parameters

Table 37-5

INPUT	Type	Remarks
REQ	BOOL	Triggering read job
ID	DWORD	Reference to the respective connection (from configured connection in STEP 7)
VAR_1	ANY	Variable to be read remotely
RD_1	ANY	Destination area (*2)
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in the destination area (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

37.6 User interface WRITE

37.6.1 Description

With the communication block data is written to the remote communication partner.

The structure description of the FMS variable lies in the remote communication partner (FMS server). When establishing the FMS connection, the local communication partner reads the structure description from the remote communication partner, which the local communication partner uses to convert the data accordingly.

WRITE

Writing the variable

37.6.2 Parameter

Table 37-6

INPUT	Type	Remarks
REQ	BOOL	Triggering write job
ID	DWORD	Reference to the respective connection (from configured connection in STEP 7)
VAR_1	ANY	Variable to be written remotely
SD_1	ANY	Source area (*2)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

37.7 User interface REPORT

37.7.1 Description

The communication block enables unconfirmed transmission of variables to an FMS client. The communication block is also used for transferring the broadcast to FMS connections.

REPORT

Reporting the variable

37.7.2 Parameters

Table 37-7

INPUT	Data type	Remarks
REQ	BOOL	Triggering report job
ID	DWORD	Reference to the respective connection (from configured connection in STEP 7)
SD_1	ANY	Source area local (*2)
VAR_1	ANY	Variable to be written remotely (*2)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing: none

(*2): SIMATIC S7 storage areas: I, Q, M, D

38 PB: DP Communication

38.1 Characteristics

Preliminary remarks

DP communication is a special case of CPU-CPU communication.

Here the communication mechanisms of the “distributed I/O” are used for a CPU-CPU communication:

- a CPU is plugged in at the central station
- the other CPU is plugged in at the decentralized station

Please refer to the functional model: 4.4

Characteristics

The DP communication is characterized by the following characteristics:

- Cyclic exchange of data between IO master and IO slave via the PROFIBUS IO protocol:
  - the DB master reads data from the DP slaves (input area)
  - the DB master writes data to the DP slaves (output area)
- The data exchange occurs consistent via the entire length (system-related data consistency)

38.2 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 38-1

Communication type:		DP communication	
Protocol:		**DP**	
General information			
Media		PB	
Interfaces		CPU, CP	
Connection	SIMATIC S5	yes	
	third-party (open standards)	yes	
User interface			
Communication blocks		Load commands / transfer commands	S7-CPU and S7-400 CP
		DPRD_DAT, DPWR_DAT	
		DP_SEND, DP_RECV	S7-300 CP
maximum data volume **(*1)**		Load command / transfer command:	1, 2, 4 bytes
		DPRD_DAT, DPWR_DAT:	<= 128 bytes
		DP_SEND, DP_REC:	DP master: <= 4 Kbytes DP slave: <=244 bytes
Number of variables when calling the communication block		1 variable	
dynamic addressing of data		no	
remote addressing		application	
Model		Master / Slave	

	Communication type:	DP communication
	Protocol:	**DP**
Protocol		
dynamic data length		no
Multicast / broadcast		no
Connections	to remote partner?	no
	dynamic / static	---
routing-capable		no
User activity		
Communication relationship	establish with	HW configuration
Connections	conf. with NetPro?	---
	prog. in STEP 7?	---
Data transmission	Communication block in STEP 7?	yes (*2)

Back to jump distributor PB

Explanations for the table:

(*1): maximum number of data may vary depending on CPU, CP type.

Per DP slave the following applies:

- input area: <= 244 bytes
- output area: <= 244 bytes

(*2): for a CPU-CPU communication the communication blocks are only required in one CPU (not in both CPUs)

38.3 Application

An overview of the most important activities for realizing a CPU-CPU communication is given below.

Table 38-2

Activity	Engineering Tool
Hardware configuration of the station: DP Master	STEP 7, HW Config
Hardware configuration of the station: DP Slave	
Programming the data exchange	STEP 7, language editor

38.4 Overview of user interfaces

Case discrimination

S7-CPUs and S7-400 CPs:

- Load commands and transfer commands
- Communication blocks DPRD_DAT, DPWR_DAT

Only for S7-300 CP:

- Communication blocks DP_SEND, DP_RECV

Communication blocks

Overview of communication blocks:

Table 38-3

Communication block	S7-300		S7-400
	CPU	CP	CPU, CP
DP_SEND	---	FC 1	---
DP_RECV	---	FC 2	---
DPRD_DAT	SFC 14	---	SFC 14
DPWR_DAT	SFC 15	---	SFC 15

Communication blocks in STEP 7:

Table 38-4

Interface		STEP 7 library
S7-300	CPU	Standard Library / System Function Blocks
S7-400	CPU, CP	
S7-300	CP	SIMATIC_NET_CP / CP300

38.5 User interface DP_SEND, DP_RECV

38.5.1 Description

The communication blocks are used for communication between local CPU and local CP (*1). The CP is DP master or DP slave.

DP_SEND

The communication block transfers data to the CP.

Case discrimination:

CP is DP master:

The data transferred to the CP with DP_SEND is written to the DP slaves cyclically.

CP is DP slave:

The data transferred to the CP with DP_SEND are read from the CP cyclically by a DP master.

DP_RECV

The communication block receives on data from the CP.

Case discrimination:

CP is DP master:

The data received by the CP with DP_RECV were read cyclically from the DP slaves by the CP.

CP is DP slave:

The data received by the CP with DP_RECV were written to the CP cyclically by a DP master.

38.5.2 Parameters for DP_SEND

Table 38-5

INPUT	Type	Remarks
CPLADDR	WORD	configured start address of the CP
SEND	ANY	Send area (*2)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Display whether job was executed without errors (*1).
ERROR	BOOL	Error display
STATUS	WORD	Status display

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

38.5.3 Parameters for DP_RECV

Table 38-6

INPUT	Type	Remarks
CPLADDR	WORD	configured start address of the CP
RECV	ANY	Receive area (*2)
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Display whether job was executed without errors (*1).
ERROR	BOOL	Error display
STATUS	WORD	Status display
DPSTATUS	BYTE	Status display

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

38.6 User interface DPRD_DAT, DPWR_DAT

38.6.1 Description

The communication blocks give the DP master access to the data of a DP slave.

DPRD_DAT

Reading consistent data of a DP standard slave

DPWR_DAT

Writing consistent data to DP standard slave

38.6.2 Parameters for DPRD_DAT

Table 38-7

INPUT	Type	Remarks
LADDR	WORD	Configured start address in the receive area to be read from
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Display whether job was executed without errors (*1).
RECORD	ANY	Destination area (*2)

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

38.6.3 Parameters for DPWR_DAT

Table 38-8

INPUT	Type	Remarks
LADDR	WORD	Configured start address in the output area to be written to
RECORD	ANY	Source area (*2)
OUTPUT	**Type**	**Remarks**
RET_VAL	INT	Display whether job was executed without errors (*1).

(*1): remote addressing: application

(*2): SIMATIC S7 storage areas: I, Q, M, D

39 Serial Interface

39.1 Characteristics

The communication via a serial interface is characterized by the following characteristics:

- simple option of a CPU-CPU coupling with nodes (point-to-point coupling)
- multipoint coupling is also possible (for RS 422/485)

39.2 Overview

The following communication types are possible via the serial interface:

Table 39-1

Communication type	Chapter
ASCII, 3964(R), RK 512	40
User-defined protocol	41

The communication types are described below.

Note

The communication with MODICON devices via the serial interface is described in Part 4 of this documentation:

Table 39-2

Communication type	Chapter
Modbus serial (RTU format)	**45**

40 ASCII and 3964(R) and RK 512

40.1 Characteristics

40.1.1 Classification

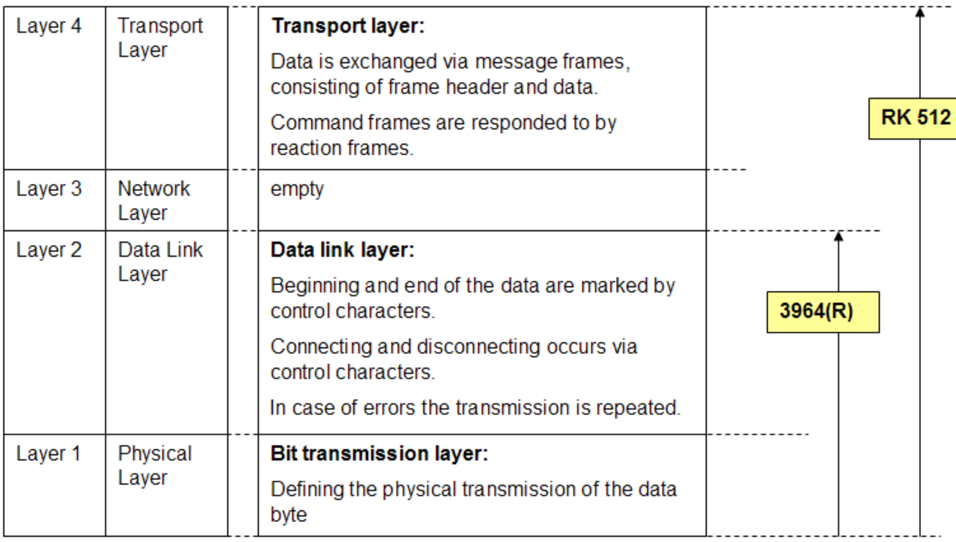
The communication types can be classified best by using the ISO/OSI reference model:

Table 40-1

Communication type	ISO/OSI reference model	Transmission security in comparison
ASCII	uses only layer 1	---
3964(R)	uses layer 1 and 2	higher as for ASCII
RK 512	uses layer 1, 2 and 4 layer 1 and 2 correspond to 3964(R)	higher as for 3964(R)

The picture shows the ISO/OSI reference model:

Figure 40-1



40.1.2 Characteristics ASCII

Functionality

The receiver detects the end of the data transmission ("end of the data") via a configurable end criterion (end of character delay time, receiving end characters, receiving fixed data volume).

Code transparency

A protocol is code transparent if any character (00H to FFH) can occur in the data to be transferred.

In the following cases ASCII is not code transparent:

- using the flow control
- using the end character as end criterion

Error recognition

Detected errors

- parity errors

Undetected errors:

- no storage of received data
- incomplete reception (missing data)

Remote confirmation / feedback message at the communication block

The user cannot detect whether the sent data has arrived in the user data area of the remote CPU (application) without error.

40.1.3 Characteristics 3964(R)

Functionality

- During the sending process control characters are added to the data (start character, end character, block check character).
- Connecting and disconnecting occurs via control characters.
- In case of errors the transmission is repeated.

Code transparency

A protocol is code transparent if any character (00H to FFH) can occur in the data to be transferred.

3964(R) is code transparent.

Error recognition

Detected errors:

- parity errors
- incomplete reception (missing data)
  exception: byte with “00H” (see below)

Undetected errors:

- byte with “00H” has been lost.

Remote confirmation / feedback message at the communication block

The user cannot detect whether the sent data has arrived in the user data area of the remote CPU (application) without error.

40.1.4 Characteristics RK 512

Functionality

The protocol works with message frames. The frames contain the data and an automatically added frame header. A command frame (SEND/PUT frame, GET frame) is followed by a reaction frame (with or without data).

Frame header of a command frame:

- Frame identifier (SEND/PUT frame, GET frame)
- Data destination for SEND/PUT job
- Data source for GET job
- Length of data to be transferred

Frame header of a reaction frame:

- Frame identifier (reaction frame, continuation frame)
- Error number

Sequence of sending/writing data:

- Communication partner sends command frame (SEND/PUT) with data
- Communication partner responds with reaction frame without data

Sequence for fetching data:

- Communication partner sends a command frame (GET) without data
- Communication partner responds with reaction frame with data

Error recognition

Detected errors:

- parity errors
- incomplete reception (missing data)

Remote confirmation / feedback message at the communication block

The user can detect whether the sent data has arrived in the user data area of the remote CPU (application) without error, or whether the data was fetched from the user data area.

40.2 Application

The serial interfaces are similar regarding their application.

The following activities must be performed by the user in STEP 7 to realize a CPU-CPU communication.

- Crating an S7 project
- Select the components (CPU, CP)
- Configure the interface (physics)
- Configuring protocols
- Create user program with communication blocks.

Additionally for S7-400:

- Configuring the connection
  (within a station: between S7-400 CPU and CP)

Additionally for loadable drivers:

- Installing the driver

Details on the application are available in:

- Manuals of the components
- STEP 7 online help

40.3 Properties

Explanations on setup and content of the table are available in chapter 25.2.

Communication types – detailed table:

Table 40-2

Communication type:		ASCII	3964(R)	RK 512
General information				
Media		Serial interface	Serial interface	Serial interface
Interfaces		CPU, CP	CPU, CP	CPU, CP
Connection	SIMATIC S5	yes	yes	yes
	third-party (open standards)	yes	yes	no
User interface				
Communication block		see 40.4	see 40.4	see 40.11
maximal data volume		<= 4096 bytes (see 40.4)	<= 4096 bytes (see 40.4)	<= 4096 bytes (see 40.11)
Number of variables when calling the communication block		1	1	1
dynamic addressing of data		yes, exception: S7-400 and SFB	yes, exception: S7-400 and SFB	yes, exception: S7-400 and SFB
remote addressing		no	transport	application
Model		Master / Master	Master / Master	Master / Master

<table>
<tr><th colspan="2">Communication type:</th><th>ASCII</th><th>3964(R)</th><th>RK 512</th></tr>
<tr><td colspan="2">Protocol</td><td colspan="3"></td></tr>
<tr><td colspan="2">dynamic data length</td><td colspan="3">yes</td></tr>
<tr><td colspan="2">Multicast / broadcast</td><td colspan="3">---</td></tr>
<tr><td rowspan="2">Connections</td><td>to remote partner?</td><td colspan="3">no (*2)</td></tr>
<tr><td>dynamic / static</td><td colspan="3">---</td></tr>
<tr><td colspan="2">routing-capable</td><td colspan="3">---</td></tr>
<tr><td colspan="2">User activity</td><td colspan="3"></td></tr>
<tr><td>Communication relationship:</td><td>establish with</td><td colspan="3">Hardware Config</td></tr>
<tr><td rowspan="2">Connections</td><td>proj. in STEP 7?</td><td colspan="3">only S7-400: NetPro (local connection)</td></tr>
<tr><td>prog. in STEP ?</td><td colspan="3">no</td></tr>
<tr><td>Data transmission</td><td>Communication block in STEP 7?</td><td colspan="3">yes (*1)</td></tr>
</table>

Back to jump distributor SERIAL

Explanations for the table:

(*1): exception: CP441: BSEND / receiving without BRCV. A communication block in the receiver is not necessary there.

(*2): For S7-400 a “PtP connection” is configured in NetPro. This is not a connection to the remote partner.

40.4 Overview of user interfaces ASCII and 3964(R)

Overview of communication blocks and maximal number of data (in brackets).

Table 40-3

Communication block	ET200S	S7-300			S7-400	
	1SI	CPU	CP 340	CP 341	CP 440	CP 441
S_SEND / S_RCV	FB3 / FB2 (224 bytes)	---	---	---	---	---
SEND_PTP / RCV_PTP	---	SFB60 /SFB61 (1024 bytes)	---	---	---	---
P_SEND / P_RCV	---	---	FB3 / FB2 (1024 bytes)	---	---	---
P_SND_RK / P_RCV_RK	---	---	---	FB8 / FB7 (1024 bytes)	---	---
SEND_440 / REC_440	---	---	---	---	FB10 / FB9 (400 bytes)	---
BSEND / BRCV	---	---	---	---	---	SFB12 / SFB13 (4096 bytes)
BSEND / Empfangsfach	---	---	---	---	---	SFB12 / --- (4096 bytes)

Communication blocks in STEP 7:

Table 40-4

Interface		STEP 7 library
ET200S	1SI	ET200sSI / ET200S Serial Interface **(*1)**
S7-300	CPU	Standard Library / System Function Blocks
	CP 340	CP PtP / CP 340
	CP 341	CP PtP / CP 341
S7-400	CP 440	CP PtP / CP 440
	CP 441	Standard Library / System Function Blocks

(*1): The blocks can be downloaded from the internet: /14/
after installation in given STEP 7 library

40.5 ASCII and 3964(R): user interface ET 200S

40.5.1 Description

The communication block S_SEND sends data to the communication block S_RCV.

S_SEND

Sending data

S_RCV

Receive data

40.5.2 Parameters for S_SEND

Table 40-5

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	INT	Base address ET 200S 1SI
DB_NO	INT	Send area (*2)
DBB_NO	INT	
LEN	INT	Length of data to be sent (*3)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
COM_RST	BOOL	New start of the FB

(*1): remote addressing:

- ASCII: none
- 3964(R): transport

(*2): SIMATIC S7 storage areas: D

(*3): maximal data volume: = 224 bytes

40.5.3 Parameters for S_RCV

Table 40-6

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
R	BOOL	Cancelling the job
LADDR	INT	Base address ET 200S 1SI
DB_NO	INT	Receive area (*2)
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
LEN	INT	Length of received data (*3)
NDR	BOOL	Data in receive area: yes / no (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
COM_RST	BOOL	New start of the FB

(*1): "yes": data was copied from the receive buffer into the receive area.

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 224 bytes

40.6 ASCII / 3964(R): user interface S7-300 CPU

40.6.1 Description

The communication block SEND_PTP sends data to the communication block RCV_PTP.

SEND_PTP

Sending data

RCV_PTP

Receiving data

40.6.2 Parameters for SEND_PTP

Table 40-7

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	WORD	I/O address of the submodule
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Send area (*2)
LEN	INT	Length of data to be sent (*3)

(*1): remote addressing:

- ASCII: none
- 3964(R): transport

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 1024 bytes

40.6.3 Parameters for RCV_PTP

Table 40-8

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
R	BOOL	Cancelling the job
LADDR	WORD	I/O address of the submodule
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in receive area: yes / no (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_1	ANY	Receive area (*2)
LEN	INT	Length of received data (*3)

(*1): “yes”: data was copied from the receive buffer into the receive area.

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 1024 bytes

40.7 ASCII / 3964(R): user interface CP 340

40.7.1 Description

The communication block P_SEND sends data to the communication block P_RCV.

P_SEND

Sending data

P_RCV

Receiving data

40.7.2 Parameters for P_SEND

Table 40-9

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 340
DB_NO	INT	Send area (*2)
DBB_NO	INT	
LEN	INT	Length of data to be sent (*3)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing:

- ASCII: none
- 3964(R): transport

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 1024 bytes

40.7.3 Parameters for P_RCV

Table 40-10

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 340
DB_NO	INT	Receive area (*2)
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
LEN	INT	Length of received data (*3)
NDR	BOOL	Data in receive area: yes / no (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “yes”: data was copied from the receive buffer into the receive area.

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 1024 bytes

40.8 ASCII / 3964(R): user interface CP 341

40.8.1 Description

The communication block P_SND_RK sends data to the communication block P_RCV_RK.

P_SND_RK

Sending data

P_RCV_RK

Receiving data

Note

The communication blocks (P_SND_RK, P_RCV_RK) are universally used for ASCII, 3964(R), and RK 512. Listed below are only those parameters which are relevant for ASCII and 3964(R).

40.8.2 Parameters for P_SND_RK

Table 40-11

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 341
DB_NO	INT	Send area (*2)
DBB_NO	INT	
LEN	INT	Length of data to be sent (*3)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing:

- ASCII: none
- 3964(R): transport

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 1024 bytes

40.8.3 Parameters for P_RCV_RK

Table 40-12

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 341
DB_NO	INT	Receive area (*2)
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
LEN	INT	Length of received data (*3)
NDR	BOOL	Data in receive area: yes / no (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “yes”: data was copied from the receive buffer into the receive area.

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 1024 bytes

40.9 ASCII / 3964(R): user interface CP 440

40.9.1 Description

The communication block SEND_440 sends data to the communication block REC_440.

SEND_440

Sending data

REC_440

Receiving data

40.9.2 Parameters for SEND_440

Table 40-13

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 440
DB_NO	INT	Send area (*2)
DBB_NO	INT	
LEN	INT	Length of data to be sent (*3)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing:

- ASCII: none
- 3964(R): transport

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 400 bytes

40.9.3 Parameters for REC_440

Table 40-14

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 440
DB_NO	INT	Receive area (*2)
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
LEN	INT	Length of received data (*3)
NDR	BOOL	Data in receive area: yes / no (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “yes”: data was copied from the receive buffer into the receive area.

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 400 bytes

40.10 ASCII / 3964(R): user interface CP 441

40.10.1 Description

Two options are available:

- sending with BSEND / receiving with BRCV
- sending with BSEND / receiving with receive buffer

In both cases the receive buffer is established in the receiver. The receive area is not transmitted during sending.

Sending with BSEND / receiving with BRCV

Communication blocks are required in sender and receiver.

Advantage of using BRCV in the receiver:

- The application (user program) recognizes the complete reception of the data
- Preventing overwriting of data in the receive buffer which have not yet been fetched by the application (user program).

Sending with BSEND / receiving with receive buffer

A communication block in the receiver is not necessary. The receive buffer (data block) is configured in the receiver.

Disadvantage of using a receive buffer in the receiver:

- The application (user program) cannot recognize when a data transfer takes place.

40.10.2 Parameters for BSEND

Table 40-15

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
R_ID	DWORD	Not applicable here
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Send area (*2)
LEN	WORD	Length of data to be sent (*3)

(*1): remote addressing:

- ASCII: none
- 3964(R): transport

(*2): SIMATIC S7 storage areas: I, Q, M, D, T, C

(*3): maximum data volume: = 4096 bytes

40.10.3 Parameters for BRCV

Table 40-16

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
ID	WORD	Reference to local connection description (given by configured connection in STEP 7)
R_ID	DWORD	Not applicable here
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in receive area: yes / no (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_1	ANY	Receive area (*2)
LEN	WORD	Length of received data (*3)

(*1): “yes”: data was copied from the receive buffer into the receive area.

(*2): SIMATIC S7 storage areas: D

(*3): maximum data volume: = 4096 bytes

40.11 Overview of user interfaces RK 512

Overview of communication blocks and maximal number of data (in brackets).

Table 40-17

Communication block	S7-300		S7-400
	CPU	CP 341	CP 441
SEND_RK / SERVE_RK	SFB 63 / SFB 65 (1024 bytes)	---	---
FETCH_RK / SERVE_RK	SFB 64 / SFB 65 (1024 bytes)	---	---
P_SND_RK / P_RCV_RK	---	FB 8 / FB 7 (4096 bytes)	---
BSEND / BRCV	---	---	SFB 12 / SFB 13 (4096 bytes)
BSEND / ---	---	---	SFB 12 / --- (4096 bytes)
PUT	---	---	SFB 15 / --- (<= 450 bytes (*1))
GET	---	---	SFB 14 / --- (<= 450 bytes (*1))

(*1): the maximal data volume depends on the used CPU:

The communication blocks are available:

Table 40-18

Interface		STEP 7 library
S7-300	CPU	Standard Library / System Function Blocks
	CP 341	CP PtP / CP 341
S7-400	CP 441	Standard Library / System Function Blocks

40.12 RK 512: user interface S7-300 CPU

40.12.1 Description

Case discrimination:

- Sending data with SEND_RK / SERVE_RK
- Fetching data with FETCH_RK / SERVE_RK

These two communication partners are referred to as:

- CPU_1
- CPU_2

40.12.2 Sending data

CPU_1 sends data to CPU_2.
CPU_1 determines where the data is stored in CPU_2.
Communication blocks:

- CPU_1: SEND_RK
- CPU_2: SERVE_RK

SEND_RK

Sending data, with specifying the receive area

SERVE_RK

Receiving data

40.12.3 Fetching data

CPU_1 fetches data from CPU_2.
CPU_1 determines which data is fetched from CPU_2.
Communication blocks:

- CPU_1: FETCH_RK
- CPU_2: SERVE_RK

FETCH_RK

Fetching data with specifying the source area

SERVE_RK

Data provision

40.12.4 Parameters for SEND_RK

Communication block in remote CPU: SERVE_RK

Table 40-19

INPUT	Type	Remarks
SYNC_DB	INT	Data block for synchronization
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	WORD	I/O address of the submodule
R_CPU	INT	Number of the remote CPU
R_TYPE	CHAR	Destination area in remote CPU (*2)
R_DBNO	INT	
R_OFFSET	INT	
R_CF_BYT	INT	Communication flag of the remote CPU
R_CF_BIT	INT	
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Source area in local CPU (*3)
LEN	INT	Length of data to be sent (*4)

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D, X)

(*3): SIMATIC storage area: data block (D)

(*4): maximum data volume: 1024 bytes

40.12.5 Parameters for SERVE_RK

Communication block in remote CPU: SEND_RK

Table 40-20

INPUT	Type	Remarks
SYNC_DB	INT	Data block for synchronization
EN_R	BOOL	Triggering receiving of data
R	BOOL	Cancelling the job
LADDR	WORD	I/O address of the submodule
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
L_TYPE	CHAR	Destination area on local CPU (*2)
L_DBNO	INT	
L_OFFSET	INT	
L_CF_BYT	INT	Communication flag
L_CF_BIT	INT	
IN_OUT	**Type**	**Remarks**
LEN	INT	Length of received data (*3)

(*1): “job finished”: data received

(*2): SIMATIC storage area: data block (D).

(*3): maximum data volume: = 1024 bytes

40.12.6 Parameters for FETCH_RK

Communication block in remote CPU: SERVE_RK

Table 40-21

INPUT	Type	Remarks
SYNC_DB	INT	Data block for synchronization
REQ	BOOL	Triggering fetching of data
R	BOOL	Cancelling the job
LADDR	EORD	I/O address of the submodule
R_CPU	INT	Number of the remote CPU
R_TYPE	CHAR	Source area in remote CPU (*2)
R_DBNO	INT	
R_OFFSET	INT	
R_CF_BYT	INT	Communication flag of the remote CPU
R_CF_BIT	INT	
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_1	ANY	Destination area in local CPU (*3)
LEN	INT	Length of data to be received (*4)

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D, X), M, I, Q, T, C

(*3): SIMATIC storage area: data block (D)

(*4): maximum data volume: = 1024 bytes

40.12.7 Parameters for SERVE_RK

Communication block in remote CPU: FETCH_RK

Table 40-22

INPUT	Type	Remarks
SYNC_DB	INT	Data block for synchronization
EN_R	BOOL	Triggering provision of data
R	BOOL	Cancelling the job
LADDR	WORD	I/O address of the submodule
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
L_TYPE	CHAR	Source area on local CPU (*2)
L_DBNO	INT	
L_OFFSET	INT	
L_CF_BYT	INT	Communication flag
L_CF_BIT	INT	
IN_OUT	**Type**	**Remarks**
LEN	INT	Length of provided data (*3)

(*1): “job finished”: data was fetched.

(*2): SIMATIC storage area: data block (D), M, I, Q, C, T

(*3): maximum data volume: = 1024 bytes

40.13 RK 512: user interface CP 341

40.13.1 Description

Case discrimination:

- Sending data with P_SND_RK / P_RCV_RK
- Fetching data with P_SND_RK / P_RCV_RK

These two communication partners are referred to as:

- CPU_1
- CPU_2

Note

The communication blocks (P_SND_RK, P_RCV_RK) are universally used for ASCII, 3964(R), and RK 512. Listed below are only those parameters which are relevant for RK 512.

40.13.2 Sending data

CPU_1 sends data to CPU_2.

CPU_1 determines where the data is stored in CPU_2.

Communication blocks:

- CPU_1: P_SND_RK
- CPU_2: P_RCV_RK

P_SND_RK

Sending data, with specifying the receive area

P_RCV_RK

Receiving data

40.13.3 Fetching data

CPU_1 fetches data from CPU_2.

CPU_1 determines which data is fetched from CPU_2.

Communication blocks:

- CPU_1: P_SND_RK
- CPU_2: P_RCV_RK

P_SND_RK

Fetching data with specifying the source area

P_RCV_RK

Data provision

40.13.4 Parameters for P_SND_RK

Communication block in remote CPU: P_RCV_RK

Table 40-23

INPUT	Type	Remarks
SF	**CHAR**	**SF = “S” (parameter for “sending data”)**
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
LADDR	INT	Base address des CP 341
DB_NO	INT	Source area in local CPU (*3)
DBB_NO	INT	
LEN	INT	Length of data to be sent (*4)
R_CPU_NO	INT	Number of the remote CPU
R_TYP	CHAR	Destination area in remote CPU (*2)
R_NO	INT	
R_OFFSET	INT	
R_CF_BYT	INT	Communication flag of the remote CPU
R_CF_BIT	INT	
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D, X)

(*3): SIMATIC storage area: data block (D)

(*4): maximum data volume: 4096 bytes

40.13.5 Parameters for P_RCV_RK

Communication block in remote CPU: P_SND_RK, with parameter SF = „S“

Table 40-24

INPUT	Type	Remarks
EN_R	BOOL	Triggering receiving of data
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 341
DB_NO	INT	Destination area on local CPU if data destination “DX” has been configured during sending (*2)
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
L_TYP	CHAR	Destination area on local CPU if data destination “DB” has been configured during sending (*2)
L_NO	INT	
L_OFFSET	INT	
LEN	INT	Length of received data (*3)
L_CF_BYT	INT	Communication flag
L_CF_BIT	INT	

(*1): “job finished”: data received

(*2): SIMATIC storage area: data block (D, X)

(*3): maximum data volume: = 4096 bytes

40.13.6 Parameters for P_SND_RK

Communication block in remote CPU: P_RCV_RK

Table 40-25

INPUT	Type	Remarks
SF	**CHAR**	**SF = “S” (parameter for “fetching data”)**
REQ	BOOL	Triggering fetching of data
R	BOOL	Cancelling the job
LADDR	INT	Base address des CP 341
DB_NO	INT	Destination area local CPU (*3)
DBB_NO	INT	
LEN	INT	Data length (*2)
R_CPU_NO	INT	Number of the remote CPU
R_TYP	CHAR	Source area remote CPU (*2)
R_NO	INT	
R_OFFSET	INT	
R_CF_BYT	INT	Communication flag of the remote CPU
R_CF_BIT	INT	
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D, X), M, I, Q, C, T

(*3): SIMATIC storage area: data block (D)

(*4): maximum data volume: 1024 bytes

40.13.7 Parameters for P_RCV_RK

Communication block in remote CPU: P_SND_RK, with parameter SF = “F”

Table 40-26

INPUT	Type	Remarks
EN_R	BOOL	Triggering provision of data
R	BOOL	Cancelling the job
LADDR	INT	Base address CP 341
DB_NO	INT	Not applicable
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
L_TYP	CHAR	Source area on local CPU (*2)
L_NO	INT	
L_OFFSET	INT	
LEN	INT	Data length (*3)
L_CF_BYT	INT	Communication flag
L_CF_BIT	INT	

(*1): “job finished”: data was fetched

(*2): SIMATIC storage area: data block (D), M, I, Q, C, T

(*3): maximum data volume: = 4096 bytes

40.14 RK 512: user interface CP 441

40.14.1 Description

Case discrimination

- Sending data with BSEND / BRCV
- Sending data with BSEND / ---
- Sending data with PUT
- Fetching data with GET

These two communication partners are referred to as:

- CPU_1
- CPU_2

40.14.2 Sending data

Sending data from CPU_1, and receiving in CPU_2.
CPU_1 determines where the data is stored.
Communication blocks:

- CPU_1: BSEND
- CPU_2: BRCV

BSEND

Sending data

BRCV

Receiving data, specifying the receive area

40.14.3 Sending data

CPU_1 sends data to CPU_2.
CPU_1 determines where the data is stored.
Communication blocks:

- CPU_1: BSEND
- CPU_2: ---

BSEND

Sending data, specifying the receive area

Note: CPU_2 cannot detect when a data transfer takes place.

Sending data PUT / ---

CPU_1 sends data to CPU_2.

Communication blocks:

- CPU_1: PUT
- CPU_2: ---

PUT: Sending data, specifying a maximum of four receive areas

Note: CPU_2 cannot detect when a data transfer takes place.

Fetching data GET / ---

CPU_1 fetches data from CPU_2.

Communication blocks:

- CPU_1: GET
- CPU_2: ---

GET: Fetching data, specifying a maximum of four source areas.

Note: CPU_2 cannot detect when a data transfer takes place.

40.14.4 Parameters for BSEND

Communication block in remote CPU: BRCV

Table 40-27

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Source area in local CPU (*2)
LEN	WORD	Length of data to be sent (*3)

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D), I, Q, M, T, C

(*3): maximal data volume of sent data: 4096 bytes

40.14.5 Parameters for BRCV

Communication block in remote CPU: BRCV

Table 40-28

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
R_ID	DWORD	Assigning the send SFB/FB and the receive SFB/FB. This enables communication of several SFB/FB pairs via the same logic connection.
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_1	ANY	Destination area in local CPU (*2)
LEN	WORD	Length of received data (*3)

(*1): “job finished”: data received

(*2): SIMATIC storage area: data block (D)

(*3): maximum data volume: 4096 bytes

40.14.6 Parameters for BSEND

Communication block in remote CPU: ---

Table 40-29

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
R_ID	DWORD	Destination area in remote CPU (*2)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error informationen
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Source area in local CPU (*3)
LEN	WORD	Length of data to be sent (*4)

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D)

(*3): SIMATIC storage area: data block (D), I, Q, M, T, C

(*4): maximum data volume: <= 450 bytes (depending on remote CPU)

40.14.7 Parameters for PUT

Communication block in remote CPU: ---

Table 40-30

INPUT	Type	Remarks
REQ	BOOL	Triggering write job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT		
ADDR_i	ANY	Destination areas in the remote CPU (i=1, 2, 3, 4) (*2) (*4)
SD_i	ANY	Source areas in the remote CPU (i=1, 2, 3, 4) (*3) (*4)

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D)

(*3): SIMATIC storage area: data block (D), I, Q, M, T, C

(*4): maximum data volume: <= 450 bytes (depending on remote CPU)

40.14.8 Parameters for GET

Communication block in remote CPU: ---

Table 40-31

INPUT	Type	Remarks
REQ	BOOL	Triggering read job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT		
ADDR_i	ANY	Source areas in the remote CPU (i=1, 2, 3, 4) (*2) (*4)
RD_i	ANY	Destination areas in the remote CPU (i=1, 2, 3, 4) (*3) (*4)

(*1): remote addressing: application

(*2): SIMATIC storage area: data block (D), I, Q, M, T, C

(*3): SIMATIC storage area: data block (D)

(*4): maximum data volume: <= 450 bytes (depending on remote CPU)

41 User-defined protocol

41.1 Characteristics

Note

The communication type is realized for S7-1200. Therefore “STEP 7 Basic” is mentioned as Engineering Tool below.

The project is realized in the STEP 7 Basic user program. This enables a high flexibility in the application.

STEP 7 Basic provides libraries with operations which can be used for programming the user-defined protocol.

41.2 Application

The following activities must be performed by the user in STEP 7 Basic to realize a CPU-CPU communication.

- Configuring devices
- Configuring interfaces of the communication modules (CM):
  - configuring in STEP 7 Basic
  - programming with special operations
- Configuring send parameters and receive parameters in STEP 7 Basic
- Programming the communication

Details on the application are available in:

- Manuals of the components
- STEP 7 online help

41.3 User interface S7-1200

41.3.1 Description

The communication block SEND_PTP sends data to the communication block RCV_PTP.

SEND_PTP

Sending data

RCV_PTP

Receiving data

41.3.2 Parameters for SEND_PTP

Table 41-1

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
PORT	PORT	Identifier of communication port
BUFFER	VARIANTE	Send area (*2)
LENGTH	UINT	Length of data to be sent (*3)
PTRCL	BOOL	---
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “job finished”: these data are transferred to the local CM

(*2): buffer of the local CPU

(*3): maximum data volume: =1024 bytes

41.3.3 Parameters for RCV_PTP

Table 41-2

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
PORT	PORT	Identifier of communication port
BUFFER	VARIANTE	Receive area (*2)
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
LENGTH	UINT	Length of received data (*3)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “job finished”: data was fetched from the local CM and written to the buffer of the CPU.

(*2): buffer of the local CPU

(*3): maximum data volume: =1024 bytes

42 Information Part 3

The tables contain references to information on the topics in Part 3.

All references /x/ are stored centrally in chapter 47. There the respective links to the internet are also available.

Table 42-1

Reference	Title / content	Information on
---	STEP 7 online help	Communication types, Communication blocks
/6/	System and standard functions for S7-300/400	Communication types, Communication blocks
/13/	SIMATIC NET functions (FC) and function blocks (FB) for SIMATIC NET S7-CPs	Communication types, Communication blocks
/100/	FAQs for S7-300 CPUs	Configuring and programming the communication: using communication blocks
/101/	FAQs for S7-400 CPUs	
/102/	FAQs for Industrial Ethernet S7-300/400CPs	
/103/	FAQs for PROFIBUS S7-300/400CPs	
/105/	FAQs for IE S7-300/400 CPs	Configuring and programming the communication: configuring the connections
/106/	FAQs for PROFIBUS S7-300/400 CPs	

PART 4: Communication with third-party controllers without using an open standard

Part 4 describes the communication with third-party controllers without using an open standard

PART 4: Structure and content

Table 42-2

Chapter	Structure	Content
44	Modbus/TCP	Coupling via PN/IE network: • Characteristics • User interfaces
45	Modbus serial (RTU format)	Coupling via serial interface: • Characteristics • User interfaces
46	Information	Device manuals, FAQs, applications, ...

43 Preliminary remarks

Open standards

A SIMATIC controller can communicate with third-party controllers via open standards (48), if the third-party controllers also have implemented open standards.

This is considered in Part 3 of the documentation in the Properties tables. There the criterion “third-party connection” exists. “Yes” means that the communication type is an open standard. This enables communication with third-party controllers which also have this implemented open standard.

Examples:

- Open communication via T blocks
- Open communication via send/receive blocks

Open protocols

Here, in Part 4 of the documentation the communication via open protocols is described.

Properties of an open protocol:

- The protocol has been opened by the manufacturer.
- The protocol is manufacturer-specific.
- Anybody can use the protocol.
- The protocol is not standardized (no international standard)

44 Modbus/TCP

44.1 Characteristics

Modbus is a worldwide distributed protocol which is open to all users. Modbus/TCP enables communication via TCP/IP networks.

44.2 Overview of user interfaces

44.2.1 General information

A SIMATIC controller can be:

- Modbus server
- Modbus client

SIMATIC without S7-1200

For the Modbus/TCP communication there are separate function blocks (Modbus blocks). The Modbus blocks are not contained in the delivery scope of the programming software STEP 7.

Information on functionality and ordering: /11/

Maximal data volume

Maximal transferable data with a Modbus TCP job:

Table 44-1

Job	Transfer bit by bit	Transfer word by word
Read job	250 bytes	250 bytes
Write job	100 bytes	200 bytes

Communication interface

SIMATIC controllers can communicate with Modbus controllers via CPU or CP.

44.2.2 Connection via SIMATIC CPU

Interface and user interface:

Table 44-2

Interface		User interface
S7-300	integrated PN/IE interface of the CPU	Modbus blocks for CPU
S7-400	integrated PN/IE interface of the CPU	
ET 200 CPU	integrated PN/IE interface of the CPU	
WinAC RTX	integrated PN/IE interface of the PC	
S7-1200	integrated PN/IE interface of the CPU	Global library operations in STEP 7 Basic

44.2.3 Connection via SIMATIC CP

Interface and user interface:

Table 44-3

Interface		User interface
S7-300	CP 343-1	Modbus blocks for CP
S7-400	CP 443-1	

44.2.4 Establishing the TCP connection

Two options are available:

- Programming the connection with T-blocks (TCON, TDISCON)
- Configuring the connection with “Modbus TCP Wizard” (/30/)

45 Modbus serial (RTU format)

45.1 Characteristics

Characteristics

Modbus is a worldwide distributed protocol which is open to all users. Modbus enables the communication via serial interfaces (RS232C, RS 422/485).

There are two versions for Modbus serial:

- RTU format: binary coding of the data
- ASCII format: ASCII coding of the data

The variant with RTU format is described below.

- Modbus serial (RTU format)

Modbus Master / Modbus Slave

Modbus serial works according to the master / slave principle.

A Modbus master can communicate with one or several Modbus slaves. Only the Modbus slave explicitly addressed by the Modbus master must send data back to the Modbus master.

The Modbus master can send jobs for reading and writing of operands to the Modbus slave:

Table 45-1

Job	Operand	
	Inputs, times, counter	Outputs, flags, data blocks
Read	x	x
Write	---	x

Addressing of Modbus slave

The address of a Modbus slave can be in the range of 1 to 255.

With the Modbus slave address zero the master addresses all slaves at the bus (broadcast).

Number of nodes

Overview:

Table 45-2

SIMATIC family	Physical Interface	Nodes
ET 200S S7-300 S7-400	RS 485	1 master, maximal 32 slaves
	RS 422	1 master, 1 slave
	RS 232C	1 master, 1 slave
S7-1200	RS 485	1 master, maximal 247 slaves
	RS 232C	1 master, 1 slave

45.2 Overview of user interfaces

45.2.1 General information

A SIMATIC controller can then be:

- Modbus master
- Modbus slave

45.2.2 Connection via SIMATIC CP

For communication via Modbus serial (RTU format) SIMATIC CPs and the respective drivers are required. The drivers are either integrated on the SIMATIC CP, or they can be downloaded. Downloadable drivers must be ordered separately and be installed in STEP 7 (/4/, chapter "Downloadable drivers").

SIMATIC CPs with integrated drivers for Modbus serial:

- ET 200S: 1SI module
- S7-1200: CM 1241

SIMATIC CPs with the option of downloading drivers for Modbus serial:

- S7-300: CP 341
- S7-400: CP 441-2

45.2.3 Modbus master

Overview of communication blocks:

Table 45-3

Communication block	ET200S	S7-300	S7-400	S7-1200
	1SI	CP 341	CP 441-2	CM 1241
S_SEND / S_RCV	FB 3 / FB 2	---	---	---
P_SND_RK / P_RCV_RK	---	FB 8 / FB 7	---	---
BSEND / BRCV	---	---	SFB 12 / SFB 13	---
MB_MASTER	---	---	---	Operation

The communication blocks are available in:

Table 45-4

Interface		STEP 7 library	Name extension
ET200S	1SI	ET200sSI / ET200S serial interface **(*1)**	ET200sSI
S7-300	CP 341	CP PtP / CP 341	CP341
S7-400	CP 441-2	Standard Library / System Function Blocks	COM_FUNC

Table 45-5

Interface		STEP 7 Basis library	Name extension
S7-1200	CM 1241	Global library operations	---

(*1): The blocks can be downloaded from the internet (/14/)

45.2.4 Modbus slave

Overview of communication blocks:

Table 45-6

Communication block	ET200S	S7-300	S7-400	S7-1200
	1SI	CP 341	CP 441-2	CM 1241
S_MODB	FB 81	---	---	---
S_SEND	FB 3	---	---	---
S_RCV	FB 2	---	---	---
	---	FB 80	---	---
P_SND_RK	---	FB 8	---	---
P_RCV_RK	---	FB 7	---	---
	---	---	FB 180	---
MB_SLAVE	---	---	---	Operation

The communication blocks are available in:

Table 45-7

Interface	Communica tion block	STEP 7 library	Name extension
ET200S, 1SI	FB 81	ET200sSI / ET200S serial interface **(*1)**	---
	FB 3 / FB 2	ET200sSI / ET200S serial Interface	ET200sSI
S7-300, CP 341	FB 80	Modbus **(*2)**	---
	FB 8 / FB 7	CP PtP / CP 341	CP341
S7-400, CP 441-2	FB 180	Modbus **(*2)**	---
S7-1200, CM 1241	---	STEP 7 Basis: global library operation	---

(*1): the blocks can be downloaded from the internet (/14/)

(*2): after the installation of the Modbus slave CD the FB is provided in the STEP 7 library “Modbus”.

45.3 Modbus master: user interface ET 200S

The user interface is identical with the user interface for ASCII and 3964(R): See chapter 40.5

45.4 Modbus master: user interface CP 341

45.4.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves.

P_SND_RK

Sending job to Modbus slave

P_RCV_RK

Receiving response frame by the Modbus slave

45.4.2 Parameter P_SND_RK

Table 45-8

INPUT	Type	Remark
SF	CHAR	SF = “S”
REQ	BOOL	Triggering the job
R	BOOL	Cancelling the job
LADDR	INT	Base address of the CP 341
DB_NO	INT	Send area in local CPU
DBB_NO	INT	
LEN	INT	Length of data to be sent
R_TYP	CHAR	Number of the remote CPU
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “job finished”:

For write function codes: after receiving the response frame

For read function codes: after receiving the response frame, and transferring the received data in the CPU

45.4.3 Parameter P_RCV_RK

Table 45-9

INPUT	Type	Remarks
EN_R	BOOL	Triggering receiving of data
R	BOOL	Cancelling the job
LADDR	INT	Base address of the CP 341
DB_NO	INT	Receive area on local CPU
DBB_NO	INT	
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in receive area: yes / no
LEN	INT	Length of received data
ERROR	BOOL	Error information
STATUS	WORD	

45.5 Modbus master: user interface CP 441-2

45.5.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves.

BSEND

Sending job to Modbus slave

BRCV

Receiving response frame by the Modbus slave

45.5.2 Parameter BSEND

Table 45-10

INPUT	Type	Remarks
REQ	BOOL	Triggering send job
R	BOOL	Cancelling the job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
R_ID	DORD	Parameter for addressing within a connection. Both communication partners must use the same value.
OUTPUT	**Type**	**Remarks**
DONE	BOOL	Job running / job finished (*1)
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
SD_1	ANY	Send range
LEN	WORD	Length of the data block to be sent

(*1): “job finished”:

For write function codes: after receiving the response frame

For read function codes: after receiving the response frame, and transferring the received data in the CPU

45.5.3 Parameter BRCV

Table 45-11

INPUT	Type	Remarks
EN_R	BOOL	Triggering receive job
ID	WORD	Reference to the respective connection (from configured connection in STEP 7)
R_ID	DWORD	Parameter for addressing within a connection. Both communication partners must use the same value.
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Data in receive area: yes / no
ERROR	BOOL	Error information
STATUS	WORD	
IN_OUT	**Type**	**Remarks**
RD_1	ANY	Receive area on local CPU
LEN	WORD	Length of received data

45.6 Modbus master: user interface CM 1241

45.6.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves.

MB_MASTER

Sending job to Modbus slave

Receiving response frame by the Modbus slave

45.6.2 Parameter MB_MASTER

Table 45-12

INPUT	Type	Remarks
REQ	BOOL	Triggering the job
MB_ADR	USINT	Modbus station address
MODE	USINT	Selecting the mode
DATA_ADDR	UDINT	Start address in the slave
DATA_LEN	UINT	The length of the read / write data
DATA_PTR	VARIANTE	Receive buffer (read) / send buffer (write)
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Job running / job finished (*1)
BUSY	BOOL	
ERROR	BOOL	Error information
STATUS	WORD	

(*1): “job finished”:

For write function codes: after receiving the response frame

For read function codes: after receiving the response frame, and transferring the received data in the CPU

45.7 Modbus slave: user interface ET 200S

45.7.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves. The Modbus slave performs the job and reacts with a response frame.

S_MODB

Executing a job from the Modbus master.

S_MODB calls internally: S_SEND, S_RCV (See chapter 40.5).

45.7.2 Parameter S_MODB

Table 45-13

INPUT	Type	Remarks
LADDR	INT	Base address ET 200S 1SI
START_TIMER	TIMER	Monitoring time
START_TIME	S5TIME	
DB_NO	INT	Modbus conversion table
OB_MASK	BOOL	Mask I/O access errors, delay alarms.
CP_START	BOOL	Initialization
CP_START_FM	BOOL	
OUTPUT	**Type**	**Remarks**
CP_NDR	BOOL	Modbus write job: running / cancelled
CP_START_OK	BOOL	Error information
CP_START_ERROR	BOOL	
ERROR_NR	WORD	Error information
ERROR_INFO	WORD	

45.8 Modbus slave: user interface CP 341

45.8.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves. The Modbus slave performs the job and reacts with a response frame.

FB80

Executing a job from the Modbus master.

FB80 calls internally: P_SND_RK, P_RCV_RK (see 40.8).

45.8.2 Parameter FB80

Table 45-14

INPUT	Type	Remarks
LADDR	INT	Base address of the CP 341
START_TIMER	TIMER	Monitoring time
START_TIME	S5TIME	
OB_MASK	BOOL	Mask I/O access errors, delay alarms.
CP_START	BOOL	Initialization
CP_START_FM	BOOL	
OUTPUT	**Type**	**Remarks**
CP_START_NDR	BOOL	Modbus write job: running / cancelled
CP_START_OK	BOOL	Error information
CP_START_ERROR	BOOL	
ERROR_NR	WORD	Error information
ERROR_INFO	WORD	

45.9 Modbus slave: user interface CP 441-2

45.9.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves. The Modbus slave performs the job and reacts with a response frame.

FB180

Executing a job from the Modbus master.

45.9.2 Parameter FB180

Table 45-15

INPUT	Type	Remarks
ID	INT	Reference to the respective connection (from configured connection in STEP 7)
START_TIMER	TIMER	Monitoring time
START_TIME	S5TIME	
STATUS_TIMER	TIMER	Monitoring time
STATUS_TIME	S5TIME	
OB_MASK	BOOL	Mask I/O access errors, delay alarms.
CP_START	BOOL	Initialization
CP_START_FM	BOOL	
OUTPUT	**Type**	**Remarks**
CS_START_NDR	BOOL	Modbus write job: running / cancelled
CP_START_OK	BOOL	Error information initialization
CP_START_ERROR	BOOL	
ERROR_NR	WORD	Error information job
ERROR_INFO	WORD	

45.10 Modbus slave: user interface CM 1241

45.10.1 Description

The Modbus master can access data in one or several Modbus slaves (write and read). The Modbus master sends jobs to Modbus slaves. The Modbus slave performs the job and reacts with a response frame.

MB_SLAVE

Executing a job from the Modbus master.

45.10.2 Parameter MB_SLAVE

Table 45-16

INPUT	Type	Remarks
MB_ADDR	USINT	Modbus station address
MB_HOLD_REG	VARIANT	Modbus holding register DB
OUTPUT	**Type**	**Remarks**
NDR	BOOL	Modbus write job: running / cancelled
DR	BOOL	Modbus read job: running / cancelled
ERROR	BOOL	Error information
STATUS	WORD	

46 Information Part 4

The tables contain references to information on the topics in Part 4.

All references /x/ are stored centrally in chapter 47. There the respective links to the internet are also available.

Table 46-1

/x/	Title / content	Information on
/17/	Communication between SIMATIC S7 and Modicon M340 via Modbus TCP	Infos on: Modbus TCP
/104/	How can I establish an OPEN Modbus / TCP communication from a SIMATIC S7 and where can I get further information?	
/107/	Which ports are enabled for Modbus/TCP communication and how many Modbus clients can communicate with a SIMATIC S7-CPU as Modbus server?	
/30/	Wizard for the communication via Modbus TCP	
/14/	Function blocks, examples and manuals of the serial interface ET200S 1SI	Infos on: Modbus RTU
/23/	SIMATIC S7-300/S7-400 Loadable driver for point-to-point CPs: Modbus protocol, RTU format, S7 is slave Operating instructions	
/24/	SIMATIC S7-300/S7-400 Loadable driver for point-to-point CPs: Modbus protocol, RTU format, S7 is master Operating instructions	
/26/	Loadable driver Modbus slave (RTU)	
/27/	Loadable driver Modbus master (RTU)	
/200/	Application for communication Task, soluton, STEP 7 project	Applications for Modbus

PART 5: Appendix

Part 5 is the appendix for the document:

PART 5: Structure and content

Table 46-2

Chapter	Structure	Content
47	Literature	References in the text: /x/
48	Central terms	Brief explanation. If a term must be described in greater detail, then see chapter Background Information.
49	Abbreviations	
50	Background Information	Description of important correlations
51	Discussed Components	Ordering data and versions
52	History	Changes / versions of the documentation

47 Literature

Content of the chapter

- Collection of helpful information on CPU-PU communication with SIMATIC controllers
- Sorted according to: information, FAQs and applications

Bibliographic references are labeled in the document with /x/.

47.1 Information

Table 47-1

/x/	Title / Content	Link
/0/	Service & Support portal Industry Automation and Drives Technologies Service & Support Portal	http://www.siemens.com/automation/service&support
/1/	SIMATIC Controller / The innovative solution for all automation tasks. Brochure April 2010 Overview of SIMATIC controllers	http://www.automation.siemens.com/salesmaterial-as/brochure/de/brochure_simatic-controller_en.pdf
/2/	SIMATIC NET / Industrial communication Brochure November 2009 Overview on industrial communication	http://www.automation.siemens.com/mcms/infocenter/dokumentencenter/sc/ic/Documentsu20Brochures/bs_k-schrift_en_1109.pdf
/3/	SIMATIC / Communication with SIMATIC System manual 09/2006 Basics on communication with SIMATIC	http://support.automation.siemens.com/WW/view/en/25074283
/4/	Catalog ST 70 2009 / Products for Totally Integrated Automation and Micro Automation Overview and ordering data for SIMATIC Controller	http://www.automation.siemens.com/salesmaterial-as/catalog/de/st7001_e.pdf
/5/	Catalog IK PI 2009 / Industrial communication Overview and ordering data for devices of industrial communication	http://www.automation.siemens.com/net/html_76/support/printkatalog.htm
/6/	SIMATIC system and standard functions for S7-300/400, reference manual, issue 05/2010 Detailed description of all functions	http://support.automation.siemens.com/WW/view/en/44240604
/7/	CPU 31xC and CPU 31 x technical data Device manual, issue 02/2009	http://support.automation.siemens.com/WW/view/en/12996906
/8/	Automation system S7-400 CPU data Device manual, issue 04/2009	http://support.automation.siemens.com/WW/view/en/23904550

/x/	Title Content	Link
/9/	Configuring and commissioning S7-CPs for Industrial Ethernet, issue 07/2009	http://support.automation.siemens.com/WW/view/en/30374198
/10/	Configuring and commissioning S7-CPs for PROFIBUS, issue 03/2009	http://support.automation.siemens.com/WW/view/en/1158693
/11/	S7 OpenModbus/TCP Product description, technical Data, ordering data, contact, downloads	http://www.industry.siemens.com/industrial-services/it/en/products/simatic_add_ons/s7_open_modbus_tcp.htm
/12/	SIMATIC NET NCM S7 for PROFIBUS / FMS band 2	http://support.automation.siemens.com/WW/view/en/1158418
/13/	SIMATIC NET functions (FC) and function blocks (FB) for SIMATIC NET S7-CPs, programming manual, issue 08/2009	http://support.automation.siemens.com/WW/view/en/30564821
/14/	Function blocks, examples and manuals of the serial interface ET200S 1SI	http://support.automation.siemens.com/WW/view/en/25358470
/15/	SIMATIC / Configuring hardware and connections with STEP 7, manual, issue 03/2006	http://support.automation.siemens.com/WW/view/en/18652631
/16/	SIMATIC PROFINET IO / From PROFIBUS DP to PROFINET IO / Programming manual, issue 10/2006	http://support.automation.siemens.com/WW/view/en/19289930
/17/	Communication between SIMATIC S7 and Modicon M340 via Modbus TCP	http://support.automation.siemens.com/WW/view/en/38586568
/18/	Performance data Results of measurements on CPU-CPU communication, in a PROFIBUS, PROFINET/Industrial Ethernet network, for different configurations.	http://support.automation.siemens.com/WW/view/en/25209605
/19/	Establishing and parameterizing point-to-point connection CP 340, issue 10/2007	http://support.automation.siemens.com/WW/view/en/1137332
	Establishing and parameterizing point-to-point connection CP 341, issue 09/2008	http://support.automation.siemens.com/WW/view/en/1117397
/20/	Establishing and parameterizing point-to-point connection CP 440, issue 09/2007	http://support.automation.siemens.com/WW/view/en/2042641
	Establishing and parameterizing point-to-point connection CP 441, issue 10/2005	http://support.automation.siemens.com/WW/view/en/1137419
/21/	S7-300 CPU 31xC technological functions (CPU 312C, CPU 313C, CPU 314C), issue 02/2007	http://support.automation.siemens.com/WW/view/en/12429336
/22/	--- not assigned ---	

/x/	Title Content	Link
/23/	SIMATIC S7-300/S7-400 Loadable driver for point-to-point CPs: Modbus protocol, RTU format, S7 is slave Operating instructions	http://support.automation.siemens.com/WW/view/en/1218007
/24/	SIMATIC S7-300/S7-400 Loadable driver for point-to-point CPs: Modbus protocol, RTU format, S7 is master Operating instructions	http://support.automation.siemens.com/WW/view/en/1220184
/25/	SIMATIC NET Quick Start Examples around the topic of communication Example programs and configurations	http://www.siemens.com/simatic-net/quickstart
/26/	Loadable driver Modbus slave (RTU)	http://support.automation.siemens.com/WW/view/de/27774276
/27/	Loadable driver Modbus master (RTU)	http://support.automation.siemens.com/WW/view/en/27774018
/28/	Manual ET 200S serial interface modules	http://support.automation.siemens.com/WW/view/en/9260793
/29/	Wizard for generating the connection data for the open TCP/IP communication	http://support.automation.siemens.com/WW/view/en/25209116
/30/	Wizard for the communication via Modbus TCP	http://support.automation.siemens.com/WW/view/en/31535566
/31/	Configuration software “PtP-Param”	http://support.automation.siemens.com/WW/view/en/27013524

47.2 FAQ

Table 47-2

/x/	Titel	Link
/100/	Configuring and programming the communication --- using communication blocks: S7-300 CPU31x	http://support.automation.siemens.com/WW/view/en/22866139
/101/	Configuring and programming the communication --- using communication blocks: S7-400 CPU41x	http://support.automation.siemens.com/WW/view/en/23522717
/102/	Configuring and programming the communication --- using communication blocks: IE S7-300/400 CPs	http://support.automation.siemens.com/WW/view/en/22548794
/103/	Configuring and programming the communication --- using communication blocks: PB S7-300/400 CPs	http://support.automation.siemens.com/WW/view/en/21629966
/104/	How can I establish an OPEN Modbus / TCP communication from a SIMATIC S7 and where can I get further information?	http://support.automation.siemens.com/WW/view/en/22660304
/105/	Configuring and programming the communication --- configuring the connections: IE S7-300/400 CPs	http://support.automation.siemens.com/WW/view/en/22387424
/106/	Configuring and programming the communication --- configuring the connections: PB S7-300/400 CPs	http://support.automation.siemens.com/WW/view/en/28526800
/107/	Which ports are enabled for Modbus/TCP communication and how many Modbus clients can communicate with a SIMATIC S7-CPU as Modbus server?	http://support.automation.siemens.com/WW/view/en/34010717

47.3 Applications from the Service & Support Portal

Table 47-3

/x/	Content	Link
/200/	Application for communication Task, solution, STEP 7 project	http://support.automation.siemens.com/WW/view/en/20229805/136000

48 Terms

This chapter contains an explanation of terms necessary for understanding the document.

Some terms are used as equivalents. These terms are marked with "=". Example: communication partner = partner

Some terms are described in greater detail elsewhere. In this case the following table contains a reference to the respective chapter ("Details" column).

Table 48-1

Term	Explanation	Details
Job = Communication job	A communication block executes a communication job. Example: "send x bytes"	
Client, Server Master, Slave Provider, Consumer	These terms are used for communication models. The terms describe properties of communication partners.	50.2
Controller	A controller is a central or decentralized automation station (station) with the components: CPU, CP and I/O. <u>Central station:</u> • station with centralized I/O. • communicates with distributed stations via PROFINET IO or PROFIBUS DP <u>Decentralized station:</u> • station with distributed I/O • communicates with central stations via PROFINET IO or PROFIBUS DP	4
CPU-CPU communication	CPU-CPU communication occurs between two CPUs: • SIMATIC CPU_1 <-> SIMATIC CPU_2 • SIMATIC CPU <-> CPU of a third-party controller	4
Data	Data refers to: net data, used data, user data, SIMATIC user data areas Examples: data block, flag, inputs, times	---
Communication via an open standard	The following applies for an "open standard": • protocols are open **and** internationally standardized. • anybody can replicate the protocols. • the protocols do not depend on the manufacturer:	---
Communication blocks	Function blocks (FB, SFB, FC, SFC), for integration into the STEP 7 user program. Implements the data transmission (send, receive).	---
Master	DP master	
Medium	• Networks: MPI, PB, PN/IE • Backplane bus • Serial interface	---
Networks	Here networks refers to industrial networks. These networks are used in the automation technology. A network can consist of one or several subnets.	1.1
Partner = Communication partner	Participants in the communication where data is exchanged	---

Term	Explanation	Details
Project	When creating an automation solution with STEP 7 the various automation tasks are solved by control programs. STEP 7 combines all control programs and the required data in one project. A project contains the following data (example): • configuration data via the Hardware setup. • Configuration data for the modules of the controller and for the distributed I/O. • Configuration data for the communication (PROFINET, ...) • Control program (LAD, FBD, ...)	---
Backplane bus	see controller	---
Interface = Communication interface	Controllers communicate via media (PN/IE, ...). The controllers are connected to the medium via interfaces. An interface may be an: • integrated interface: CPU • external interface: CP	---
Send/Receive blocks	Collective term for the following communication blocks: AG_SEND, AG_LSEND, AG_SSEND, AG_RECV, AG_LRECV, AG_SRECV	
Slave	DP slave	
Subnet	A subnet is located in the area of the LANs (Local Area Networks). It enables communication, for example, between CPUs of controllers, within a spatially restricted area. A subnet is closed in itself, it has its own address space. Several subnets form a network.	---
T-blocks	Collective term for the following communication blocks: TSEND, TUSEND, TRCV, TURCV	
Connection	Relationship between communication partners	5
Connection blocks	Function blocks (FB, SFB, FC, SFC), for integration into the STEP 7 user program. This realizes the connection (connect, disconnect).	---

49 Abbreviations

In the chapter the abbreviations are explained.

Table 49-1

Abbreviation	Explanation
CBA	PROFINET CBA (Component Based Automation)
CP	Communication Processor: module which deals with communication tasks, and connects a controller to a medium.
CPU	Central Procesing Unit Module on which a user program runs. In this user program data are sent or received.
DP	Decentralized Periphery (distributed I/O)
I, Q, M, D, T, C	SIMATIC S7 storage areas: process image inputs (I), process image output (Q), flag (M), data block (D), times (T), counter (C)
FMS	Fieldbus Message Specification
GD	Global data
HW Config	Hardware configuration: tool for configuration of Hardware in STEP 7
IOC	PROFINET IO Controller
IOD	PROFINET IO Device
IoT	ISO on TCP
MPI	Multi Point Interface
NetPro	Network configuration: Tool for configuration of connections in STEP 7
OP	Operator Panel
PB	PROFIBUS
PG	Programming device
PN/IE	PROFINET / Industrial Ethernet
PNIO	PROFINET IO
S/R-blocks	Send/Receive blocks
S7-CP	CP of SIMATIC S7
S7-CPU	CPU of SIMATIC S7.

50 Background Information

This chapter provides background information.

50.1 ISO/OSI reference model

The ISO/OSI reference model is a standardized model for describing open (manufacturer-independent) communication systems. The model describes the requirements for a communication system. The concrete implementation is not described. Most of the free to use protocols are based on this reference model (for example: TCP/IP).

The model consists of 7 layers with the following properties:

- Each layer has to fulfill fixed defined tasks
- The layers are independent of one another

Explanation of the layers

Table 50-1

Layer	Name	Task (examples)	Classification
Layer 7	Application layer	Interface with STEP 7 user program (confirmation on user level, ...).	application-oriented
Layer 6	Presentation layer	Interpretation of the data (converting the standardized representation of the communication system into a device-specific form.)	
Layer 5	Session layer	Organization of data transfer	
Layer 4	Transport layer	Creating a connection between two devices: • establishing, canceling, maintaining the connection Transferring data packages: • dividing the data into packages (segmentation) • preventing the status of packages • confirmation on transport layer	transport-oriented
Layer 3	Network layer	Transfer and delivery of data: • defining the communication paths • addressing the communication partners • routing in the network	
Layer 2	Data link layer	Monitoring and organizing the access to the transfer medium (flow control, ...) Correct transfer of data (checksum, ...)	
Layer 1	Physical layer	Defining the physical connection between two devices (transfer medium, baud rate, ...)	

50.2 Communication models

Communication models (short: models) describe the principle of a communication relation. They specify the role both communication partners play during data exchange.

50.2.1 Client and server

The terms client and server are used in the document as follows:

Field of application

Networks: PN/IE, PB, MPI

Client

Properties

A client can exchange data with a client or a server.

Providing the communication in the client

Provisions must be made in the STEP 7 user program:

- programming the communication blocks, and/or
- configuring/programming the connections

Server

Properties

A server can exchange data with a client.

The trigger for data exchange always comes from a client, i.e. a server cannot take initiative for a data exchange.

Providing the communication in the server

Two different cases must be distinguished here:

Case 1: the communication is provided by the operating system only, i.e. the communication is a system functionality.

Case 2: Provisions must be made in the STEP 7 user program:

- programming the communication blocks, and/or
- configuring/programming the connections

Client / Client communication

Both communication partners are clients.

One of both clients takes the initiative for the communication.

Client / Server communication

One communication partner is client, one communication partner is server.

Only the client can take the initiative for the communication.

50.2.2 Master and slave

The terms master and slave are used in the document as follows:

Field of application

Networks: PB (communication type DP communication)

Serial interface: Modbus serial, ...

Master

A master has the initiative during data exchange (behaves active):

- sends data to slave
- receives data from slave which he has requested from the slave beforehand

Slave

A slave has no initiative during data exchange (behaves passive):

- sends data to the master only if prompted by the master
- receives data from the master

Master / Slave communication

One communication partner is master, one communication partner is slave.

The master has the initiative.

Master / Master communication

Both communication partners are master.

Both communication partners can take the initiative to send

DP communication

Master / Master communication is possible, however, this is not discussed in the document. This would require additional hardware: DP/DP coupler.

Serial interface

Master / Master communication is not possible.

50.2.3 Consumer and provider

The terms consumer and provider used in the document as follows:

Field of application

Network: PN/IE (communication type PNIO)

Consumer

Receives data from the provider without request.

Provider

Sends data to the consumer without request.

Consumer / Provider communication

One communication partner is consumer, one communication partner is provider.
Consumer and provider are equal nodes in the network.

50.3 Confirmation

If data is transferred there are different feedback messages (confirmations) to the STEP 7 user program.

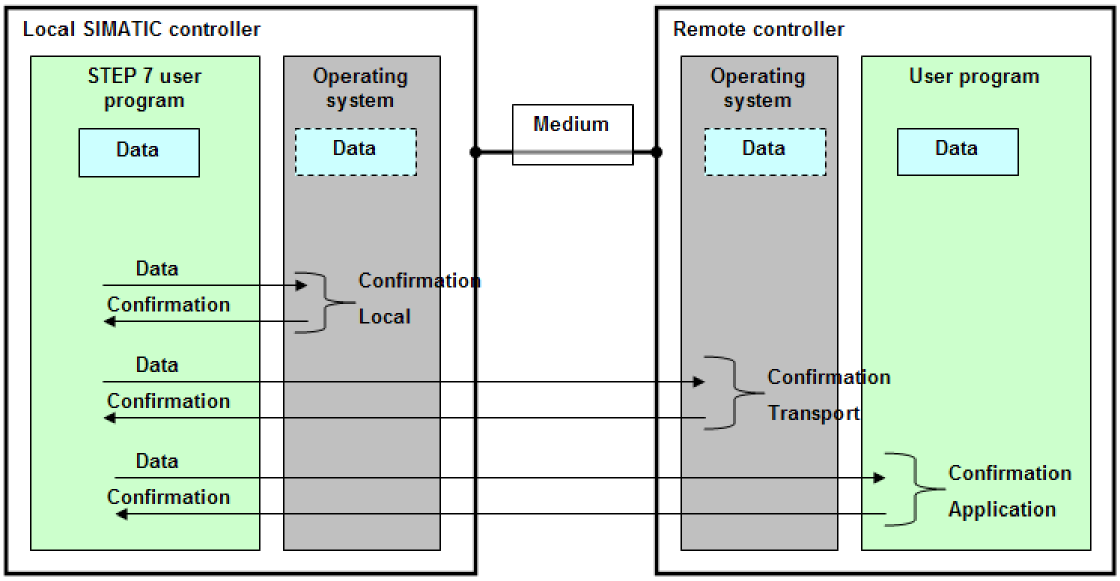
In the document the following confirmations are distinguished:

Table 50-2

Confirmation	Description	Explanations
local	The data lies in the operating system (CPU or CP) of the local controller	No statement on whether the data was transferred via the medium (PROFINET/IE, …).
Transport	The data lies in the operating system (CPU or CP) of the remote controller	The data was transferred via the medium (PROFINET/IE, …).
application	The data lies in the application of the remote controller.	

The following figure illustrates the relationships.

Figure 50-1



51 Discussed Components

This chapter lists the components for which the statements in the document are valid.

The following details are given for the components:

- Ordering code (MLFB)
- Issue / version

Component version:

- September 2010

51.1 Programming tools

SIMATIC family S7-1200

SIMATIC STEP 7 Basic, V10.5, SP2

All other SIMATIC families

SIMATIC STEP 7, V5.5

SIMATIC iMAP V3.0, SP1

SIMATIC Distributed Safety V5.4, SP5

51.2 SIMATIC CPU

Table 51-1

Controller	Family	CPU		MLFB	Version
modular	ET 200 CPU	ET 200S	IM151-8(F) PN/DP CPU	6ES7 151-8AB01-0AB0 (6ES7 151-8FB01-0AB0)	FW V3.2
		ET 200S	IM151-7(F) CPU	6ES7 151-7AA20-0AB0 (6ES7 151-7FA20-0AB0)	FW V2.6
		ET 200Pro	IM154-8(F) PN/DP CPU	6ES7 154-8AB01-0AB0 (6ES7 154-8FB01-0AB0)	FW V3.2
	S7-300	CPU 312		6ES7 312-1AE14-0AB0	FW V3.0
		CPU 314		6ES7 314-1AG14-0AB0	FW V3.0
		CPU 312C		6ES7 312-5BE03-0AB0	FW V2.6
		CPU 313C		6ES7 313-5BF03-0AB0	FW V2.6
		CPU 313C-2 DP		6ES7 313-6CF03-0AB0	FW V2.6
		CPU 314C-2 DP		6ES7 314-6CG03-0AB0	FW V2.6
		CPU 313C-2 PtP		6ES7 313-6BF03-0AB0	FW V2.6
		CPU 314C-2 PtP		6ES7 314-6BG03-0AB0	FW V2.6
		CPU 315(F)-2 DP		6ES7 315-2AH14-0AB0 (6ES7 315-6FF04-0AB0)	FW V3.0
		CPU 317(F)-2 DP		6ES7 317-2AJ10-0AB0 (6ES7 317-6FF03-0AB0)	FW V2.6
		CPU 315(F)-2 PN/DP		6ES7 315-2EH14-0AB0 (6ES7 315-2FJ14-0AB0)	FW V3.2
		CPU 317(F)-2 PN/DP		6ES7 317-2EK14-0AB0 (6ES7 317-2FK14-0AB0)	FW V3.2
		CPU 319(F)-3 PN/DP		6ES7 318-3EL01-0AB0 (6ES7 318-3FL01-0AB0)	FW V3.2
	S7-400	CPU 412-1		6ES7 412-1XJ05-0AB0	FW V5.3
		CPU 412-2		6ES7 412-2XJ05-0AB0	FW V5.3
		CPU 414-2		6ES7 414-2XK05-0AB0	FW V5.3
		CPU 414-3		6ES7 414-3XM05-0AB0	FW V5.3
		CPU 416(F)-2		6ES7 416-2XN05-0AB0 (6ES7 416-2FN05-0AB0)	FW V5.3
		CPU 416-3		6ES7 416-3XR05-0AB0	FW V5.3
		CPU 417-4		6ES7 417-4XT05-0AB0	FW V5.3
		CPU 412-2 PN		6ES7412-2EK06-0AB0	FW V6.0
		CPU 414(F)-3 PN/DP		6ES7 414-3EM06-0AB0 (6ES7414-3FM06-0AB0)	FW V6.0
		CPU 416(F)-3 PN/DP		6ES7 416-3ES06-0AB0 (6ES7 416-3FS06-0AB0)	FW V6.0
	S7-1200	CPU 1211C		6ES7211-xxx-0XB0	
		CPU 1212C		6ES7212-xxx-0XB0	
		CPU 1214C		6ES7214-xxx-0XB0	

Controller	Family	CPU	MLFB	Version
embedded	S7-mEC (*1)	EC31	6ES7677-1DD00-0BB0	
	Box PC (*1)	IPC427C bundles with RTX	6ES7 675-1D...	
	Panel PC (*1)	HMI IPC477C bundles with RTX	6AV7 884...	
	WinAC MP (*2)	MP177 with WinAC MP	6ES7 671-4EE00-0YA0	
		MP277 with WinAC MP	6ES7 671-5EF01-0YA0	
		MP377 with WinAC MP	6ES7 671-7EG01-0YA0	
PC-based	WinAC RTX	WinAC RTX (F) 2009 (Software)	6ES7 671-0RC07-0YA0 (6ES7 671-1RC07-0YA0)	V4.5

Explanations for the table:

(*1): with WinAC RTX 2009 as software controller

(*2): with WinAC MP 2008 as software controller

51.3 SIMATIC CP

Table 51-2

Controller	Family	CP			MLFB	Version
modular	ET 200 CPU	PB	ET 200S	DP master module	6ES7 138-4HA00-0AB0	FW V1.0
		PtP	ET 200S	1 SI 3964/ASCII	6ES7138-4DF01-0AB0	FW V1.4
		PtP	ET 200S	1 SI Modbus/USS	6ES7138-4DF11-0AB0	FW V1.4
	S7-300	PB	CP 342-5		6GK7 342-5DA02-0XE0	FW V5.0
		PB	CP 342-5 FO		6GK7342-5DF00-0XE0	FW V5.0
		PB	CP 343-5		6GK7 343-5FA01-0XE0	FW V4.0
		PN/IE	CP 343-1 Lean		6GK7 343-1CX10-0XE0	FW V2.3
		PN/IE	CP 343-1		6GK7 343-1EX30-0XE0	FW V2.3
		PN/IE	CP 343-1 Advanced		6GK7 343-1GX30-0XE0	FW V1.1
		PN/IE	CP 343-1 ERPC		6GK7343-1FX00-0XE0	FW V1.0
		PtP	CP 340		6ES7340-1xH02-0AE0	FW V1.0
		PtP	CP 341		6ES7341-1xH02-0AE0	FW V2.0
	S7-400	PB	CP 443-5 Basic		6GK7 443-5FX02-0XE0	FW V4.0
		PB	CP 443-5 Extended		6GK7 443-5DX04-0XE0	FW V6.4
		PN/IE	CP 443-1		6GK7 443-1EX20-0XE0	FW V2.1
		PN/IE	CP 443-1 Advanced		6GK7 443-1GX20-0XE0	FW V2.1
		PtP	CP 440		6ES7440-1CS00-0YE0	FW V1.0
		PtP	CP 441-1		6ES7441-1AA04-0AE0	FW V1.0
		PtP	CP 441-2		6ES7441-2AA04-0AE0	FW V1.0
	S7-1200	PtP	CM 1241 RS485		6ES7241-1CH30-0XB0	
		PtP	CM 1241 RS232		6ES7241-1AH30-0XB0	
embedded	S7-mEC	PB	EM PCI-104		6ES7677-1DD40-1AA0	
		PN/IE	EM PC		6ES7677-1DD50-2AA0	
		PtP	CP 340		6ES7340-1xH02-0AE0	V1.0
	Box PC	(*1)	---		---	---
	Panel PC	(*1)	---		---	---
	WinAC MP	---	---		---	---
PC-based	WinAC RTX	(*1)	---		---	---

(*1): CPs for Box PC, Panel PC and WinAC RTX

Principally the following modules can be employed:

Connection to PN/IE: CP 1616, CP 1604

Connection to PB: CP 56-11-A2, CP 5621, CP 5613, CP 5613-A2, CP 5603

Please refer to the catalog to find out which modules are possible in the concrete application case.

52 History

52.1 Versions

Table 52-1

Version	Date	
V1.0	04 / 2004	First issue
V2.0	11 / 2010	Complete revision
V2.01	01 / 2011	error correction

52.2 Main changes

Table 52-2

Version	Changes
V1.0 -> V2.0	• Update with new components • New structure of the doucument
V2.0 -> V2.01	page 368: paragraph deleted: SIMATIC S7-1200
	page 398: SIMATIC CPU supplemented: IM151-7 F CPU