SIEMENS

SIMATIC

Point-to-Point Communication CP 441 Installation and Parameter Assignment

Manual

This manual is part of the documentation package with the order number:

6ES7 441-2AA00-8BA0

04/2000
C79000-G7076-C441
Edition 05

Danger

indicates that death, severe personal injury or substantial property damage **will** result if proper precautions are not taken.

Warning

indicates that death, severe personal injury or substantial property damage **can** result if proper precautions are not taken.

Caution

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

Note

draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground, and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Note the following:

Warning

This device and its components may only be used for the applications described in the catalog or the technical descriptions, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.

Some of other designations used in these documents are also registered trademarks; the owner's rights may be violated if they are used by third parties for their own purposes.

Siemens AG
Bereich Automatisierungs- und Antriebstechnik
Geschaeftsgebiet Industrie-Automatisierungssysteme
Postfach 4848, D- 90327 Nuernberg

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Excellence in Automation & Drives: Siemens

Preface

Purpose

This manual explains how to establish and operate a point-to-point connection.

Contents of This Manual

The manual describes the hardware and software of the CP 441 communication processor and its integration in an S7-400 programmable controller. It is divided up into instruction-based chapters and a reference section (appendices).

The following subjects are covered:

- The basics of point-to-point communication with the CP 441
- Starting up the CP 441
- Mounting the CP 441
- Communication via the CP 441
- Debugging
- Application examples
- Properties and technical specifications

Scope of This Manual

The manual is relevant to the following CPs and interface submodules:

Product	Order Number	As of Release
CP 441-1	6ES7 441-1AA03-0AE0	01
CP 441-2	6ES7 441-2AA03-0AE0	01
RS 232C module	6ES7 963-1AA00-0AA0	01
20 mA TTY module	6ES7 963-2AA00-0AA0	01
X27 (RS 422/485) module	6ES7 963-3AA00-0AA0	01

Note

The descriptions of the CP 441 communication processor and the interface submodules in this manual were correct at the time of publication. We reserve the right to describe modifications to the functionality of the modules in a separate Product Information.

Changes Since the Previous Edition

Changes Since the Previous EditionSince edition 02 of this manual, *CP 441 Point-to-Point Communication, Installation and Parameter Assignment*, descriptions of the following functions of the CP 441 with order number 6ES7 441-_AA03-0AE0 have been added:

- Additional transfer rates of 57.6 kbaud and 115.2 kbaud for the CP 441-2 from STEP 7, V5.0 + Service Pack 2 onwards
- Dynamically changeable destinations in RK512 send message frames
- Firmware update
- Diagnostics alarm from STEP 7, V5.0 + Service Pack 2 onwards
- Fast switch-over for RS485 module in half-duplex mode
- RS422/485 level selection now for 3964(R), RK512 and printer
- Extended modes for reception with end-of-text codes
- Shorter character delay time at low baud rates
- ASCII protocol with fixed message frame length: sending in character delay time matrix can be deactivated

Conventions

This manual uses the generic term CP 441. The information in the manual applies to the CP 441-1 and CP 441-2 communication processors, unless otherwise specified.

Structure of This Manual

To help you to quickly find the information you require, this manual offers the following:

- A comprehensive list of contents.
- Following the appendices, a glossary defines important technical terms used in the manual.
- Finally, a comprehensive index allows quick access to information on specific subjects.

Other Manuals Required

The appendix F contains a list of further manuals and brochures on the S7-400 and programmable controllers.

Electronic Manuals

The entire set of SIMATIC S7 documentation is available on CD-ROM.

Standards, Certificates and Approvals

The CP 441 communication processor meets the requirements and criteria of IEC 1131, Part 2 and the requirements for CE marking. The CP 441 has CSA certification and UL recognition.

You will find more information on certification/recognition/approval and standards in Appendix A.2.

Recycling and Disposal

The CP 441 is an environment-friendly product. It is exceptional for the following:

- Housing plastic with halogen-free flame protection and is highly resistant to fire
- Laser inscriptions (i.e. no labels)
- Plastics identification in accordance with DIN 54840
- Fewer materials used due to size reduction; fewer parts due to integration in ASICs

The CP 441 is suitable for recycling on account of the low level of contaminants in its components.

For further information about environment-friendly recycling and the procedure for disposing of your old equipment, please contact:

Siemens Aktiengesellschaft
Anlagenbau und Technische Dienstleistungen
ATD ERC Essen Recycling/Remarketing
Fronthauser Str. 69
D-45127 Essen
Germany

Phone: + 49 201/816 1540 (Hotline)
Fax: + 49 201/816 1504

The people there will adapt their advice to suit your situation and provide a comprehensive and flexible recycling and disposal system at a fixed price. After disposal you will receive information giving you a breakdown of the relevant material fractions and the associated documents as evidence of the materials involved.

Additional Assistance

Please contact your local Siemens representative if you have any queries about the products described in this manual. A list of Siemens representatives worldwide is contained, for example, in the appendix entitled ”Siemens Worldwide” of the manual *S7-400/M7-400 Programmable Controller, Hardware and Installation*.

If you have any questions or suggestions concerning this manual, please fill out the form at the back and return it to the specified address. Please feel free to enter your personal assessment of the manual in the form provided.

We offer a range of courses to help get you started with the SIMATIC S7 programmable controller. Please contact your local training center or the central training center in Nuremberg, D-90027 Germany, Tel. +49 911 895 3200.

Up to the Minute Information

You can obtain the latest information on SIMATIC products from the following sources:

- on the **Internet** at *http://www.ad.siemens.de/*

SIMATIC Customer Support also provides you with the latest information and downloads that will help you use SIMATIC products:

- on the **Internet** at *http://www.ad.siemens.de/simatic-cs*
- From the SIMATIC Customer Support Mailbox at the number +49 (911) 895-7100

  To dial the mailbox, use a modem with up to V.34 (28.8 kbauds), with the following parameter settings: 8, N, 1, ANSI, or dial via ISDN (x.75, 64 Kbps).

SIMATIC Customer Support is available at the phone and fax numbers and at the E-mail addresses listed below. You can also contact us via Internet mail or mail at the mailbox mentioned above.

Worldwide (Nuremberg) **Technical Support** (FreeContact) Local time: Mon.-Fri. 7:00 to 17:00 Phone: +49 (180) 5050-222 Fax: +49 (180) 5050-223 E-Mail: techsupport@ad.siemens.de GMT: +1:00	**Worldwide** (Nuremberg) **Technical Support** (fee based, only with SIMATIC Card) Local time: Mon.-Fri. 0:00 to 24:00 Phone: +49 (911) 895-7777 Fax: +49 (911) 895-7001 GMT: +01:00	
Europe / Africa (Nuremberg) **Authorization** Local time: Mon.-Fri. 7:00 to 17:00 Phone: +49 (911) 895-7200 Fax: +49 (911) 895-7201 E-Mail: authorization@nbgm.siemens.de GMT: +1:00	**America** (Johnson City) **Technical Support and Authorization** Local time: Mon.-Fri. 8:00 to 19:00 Phone: +1 423 461-2522 Fax: +1 423 461-2289 E-Mail: simatic.hotline@sea.siemens.com GMT: –5:00	**Asia / Australia** (Singapore) **Technical Support and Authorization** Local time: Mon.-Fri. 8:30 to 17:30 Phone: +65 740-7000 Fax: +65 740-7001 E-Mail: simatic.hotline@sae.siemens.com.sg GMT: +8:00
The languages of the SIMATIC Hotlines are generally German and English, in addition, French, Italian and Spanish are spoken on the authorization hotline.		

Contents

Product Description

1

1.1 Uses of the CP 441

The CP 441 communication processor allows you to exchange data between programmable controllers or computers by means of a point-to-point connection.

Functionality of the CP 441

The CP 441 communication processor provides the following functionality:

- A choice of two models with either one (the CP 441-1) or two (CP 441-2) serial device interfaces, which can be adjusted to suit the properties of the communication partners by means of plug-in interface submodules. There are three interface submodules available:
  - RS 232C interface submodule
  - 20 mA TTY interface submodule
  - X27 (RS 422/485) interface submodule
- Transmission rate
  - CP 441-1:max. 38.4 kbaud
  - CP 441-2:max. 115.2 kbaud (total baud rate)
- Integration of the most important transmission protocols in the module firmware (see Table 1-1)
- Custom parameterization of the transmission protocols with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.
- Subsequent loading of additional drivers (transmission protocols) to the CP 441-2 (6ES7 441-2AA02-0AE0 and higher) with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

Integrated Transmission Protocols

The following transmission protocols are integrated in the module firmware of the CP 441:

Table 1-1 Transmission Protocols in the Module Firmware

Product	Order Number	Integrated drivers
CP 441-1	6ES7 441-1AA00-0AE0	3964(R) procedure
CP 441-2	6ES7 441-2AA00-0AE0	3964(R) procedure, RK 512 computer connection
CP 441-1	6ES7 441-1AA01-0AE0	3964(R) procedure, ASCII driver, printer driver
CP 441-2	6ES7 441-2AA01-0AE0	3964(R) procedure, ASCII driver, RK 512 computer connection, printer driver
CP 441-1	6ES7 441-1AA02-0AE0 6ES7 441-1AA03-0AE0*	3964(R) procedure, ASCII driver, printer driver

Table 1-1 Transmission Protocols in the Module Firmware, continued

Product	Order Number	Integrated drivers
CP 441-2	6ES7 441-2AA02-0AE0 6ES7 441-2AA03-0AE0*	3964(R) procedure, ASCII driver, RK 512 computer connection, printer driver

* Modules described in this manual

Uses of the CP 441

The CP 441 communication processor allows point-to-point communication with SIMATIC modules and with non-Siemens products. The SIMATIC modules that can be connected are listed in Appendix D.

Supported Interface Submodule Functions

Different driver functions can be used depending on the interface submodule used:

Table 1-2 Functions of the CP 441 Depending on the Interface Submodule Used

Function	RS 232C	20 mA TTY	X27 (RS 422/485)	
			RS 422*	RS 485*
3964(R) procedure	**yes**	**yes**	**yes**	**no**
RK 512 computer connection	**yes**	**yes**	**yes**	**no**
ASCII driver	**yes**	**yes**	**yes**	**yes**
• Use of RS 232C secondary signals	yes	no	no	no
• Controlling/reading of RS 232C secondary signals with FBs	yes	no	no	no
• RTS/CTS flow control	yes	no	no	no
• XON/XOFF flow control	yes	yes	yes	no
Printer driver	**yes**	**yes**	**yes**	**yes**
• RTS/CTS flow control	yes	no	no	no
• XON/XOFF flow control	yes	yes	yes	no

* The RS 422 and RS 485 are distinguished by means of parameterization.

1.2 Components Required for a Point-to-Point Connection with the CP 441

To establish a point-to-point connection between the CP 441 communication processor and a communication partner, you require certain hardware and software components.

Hardware Components

The following table lists the hardware components required for establishing a point-to-point connection with the CP 441.

Table 1-3 Hardware Components for a Point-to-Point Connection with the CP 441

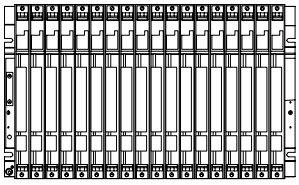
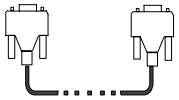
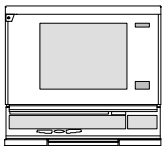
Components	Function	Diagram
Rack	... provides the mechanical and electrical connections of the S7-400.	
Power supply module (PS)	... converts the line voltage (120/230 VAC or 24 VDC) into the operating voltage of 24 V and 5 VDC required to supply the S7-400.	
CPU Accessories: memory card backup battery	... executes the user program; communicates via the MPI interface with other CPUs or with a programming device.	
CP 441 communication processor Interface submodules	... communicates via the interface with one or more communication partners. ... enable the CP 441 to be adapted to suit the communication partner.	
Standard connecting cable	... connects the CP 441 communication processor to the communication partner.	

Table 1-3 Hardware Components for a Point-to-Point Connection with the CP 441, continued

Components	Function	Diagram
Programming device cable	... connects a CPU to a programming device/PC.	
Programming device or PC	... communicates with the CPU of the S7-400.	

Software Components

The following table lists the software components required for establishing a point-to-point connection with the CP 441.

Table 1-4 Software Components for a Point-to-Point Connection with the CP 441

Components	Function	Diagram
STEP 7 software package	... configures, parameterizes, programs and tests the S7-400.	CD + Licence
The *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface	... parameterizes the interfaces of the CP 441.	CD
Function blocks	... for reading and controlling the RS 232C secondary signals.	
Programming example	... with user programs for printer output and data transfer by means of the RK 512 computer connection and the 3964(R) procedure.	
Loadable drivers	... with transmission protocols that can be loaded on the CP 441-2 in addition to the standard protocols in the module firmware.	CD + Dongle

1.3 Design of the CP 441

The CP 441-1 communication processor has one slot, the CP 441-2 two, for plug-in interface submodules. The operator control and display elements are in the same position on both the CP 441-1 and the CP 441-2, and the same elements have the same functions on both models.

Position of Operator Control and Display Elements

Fig. 1-1 shows the positions of the operator control and display elements on the front panel of the CP 441-1 and the CP 441-2 communication processors.

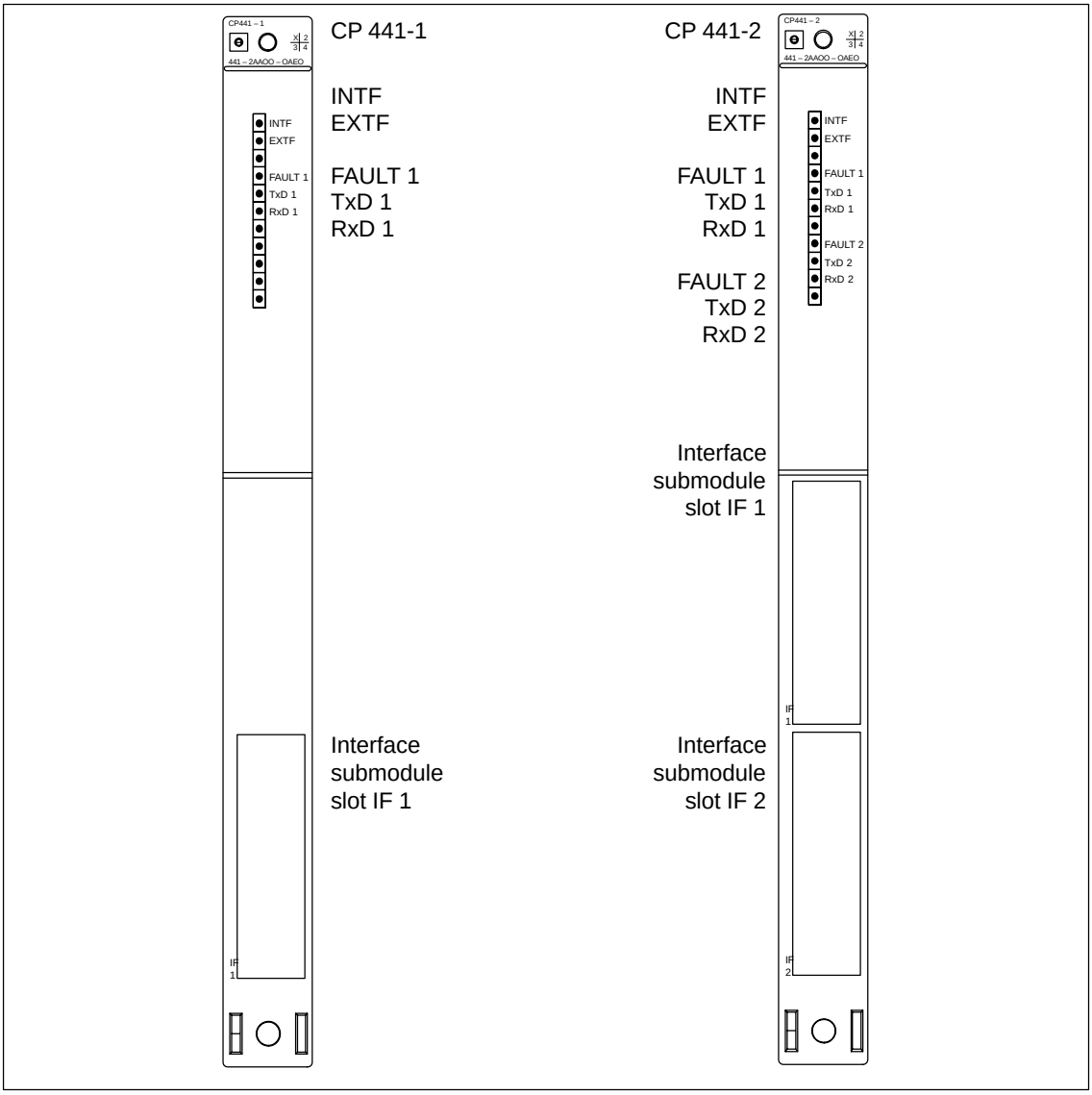


Figure 1-1 Position of Operator Control and Display Elements on CP 441-1 and CP 441-2 Communication Processors

LED Displays

The following LED displays are located on the front panel of the CP 441:

- **INTF** (red) CP 441 signals internal fault
- **EXFT** (red) CP 441 signals external fault
- **FAULT 1** (red) Fault LED for interface IF 1
- **TXD 1** (green) Interface IF 1 transmitting
- **RXD 1** (green) Interface IF 1 receiving
- **FAULT 2** (red) Fault LED for interface IF 2 (CP 441-2)
- **TXD 2** (green) Interface IF 2 transmitting (CP 441-2)
- **RXD 2** (green) Interface IF 2 receiving (CP 441-2)

The operating modes and errors indicated by these LEDs are described in Section 8.2. Section 5.7 contains information on LED displays which can occur when you load firmware updates.

Slot for interface modules

The CP 441-1 contains one slot, the CP 441-2 two, for plug-in interface submodules. By exchanging the interface submodules, you can adjust the CP 441 to suit the properties of the communication partners.

There are three types of interface submodule:

- RS 232C
- X27 (RS 422/485)
- 20 mA TTY

For a detailed description of these modules, see Section 1.4.

Base Connector for S7 Backplane Bus

On the back panel of the CP 441 you will find the base connector for the S7-400 backplane bus.

The S7-400 backplane bus is a serial data bus via which the CP 441 communicates with the modules of the programmable controller and is supplied with the necessary voltage.

1.4 Attributes of the Interface Submodules

By selecting different plug-in interface submodules, you can adjust the CP 441 to suit the properties of the communication partners.

The sections that follow describe the interface submodules of the CP 441.

1.4.1 Attributes of the RS 232C Interface Submodule

Definition

The RS 232C interface submodule is a voltage interface used for serial data transmission in compliance with the RS 232C standard.

Attributes

The RS 232C interface submodule has the following attributes:

- Type: voltage interface
- Front connector: 9-pin subminiature D male connector with a screw-type fitting (compatible with the 9-pin COM port (PC/PG))
- Max. transmission rate: 115.2 kbaud*
- Max. cable length: 10 m**
- Standard: DIN 66020, DIN 66259, EIA-RS 232C, CCITT V.24/V.28
- Degree of protection: IP00

* Please observe the maximum permitted transmission rates for the modules (see Section 2.3).
** See Appendix B for a list of the cables types you can use.

RS 232C Signals

Table 1-5 shows the meanings of the RS232C secondary signals.

Table 1-5 Signals of the RS 232C Interface

Signal	Designation	Meaning
TXD	**T**ransmitted **D**ata	Transmitted data; Transmission line is held by CP 441 on logic ”1” in idle state.
RXD	**R**eceived **D**ata	Received data; Receive line must be held on logic ”1” by communication partner.
RTS	**R**equest **T**o **S**end	RTS ”ON”: CP 441 clear to send RTS ”OFF”: CP 441 not sending
CTS	**C**lear **T**o **S**end	Communication partner can receive data from CP 441. The CP 441 expects the signal as response to RTS ”ON”.
DTR	**D**ata **T**erminal **R**eady	DTR ”ON”: CP 441 is active and ready for operation DTR ”OFF”: CP 441 is not active and not ready for operation
DSR	**D**ata **S**et **R**eady	DSR ”ON”: Communication partner is active and ready for operation DSR ”OFF”: Communication partner is not active and not ready for operation
RI	**R**ing **I**ndicator	Incoming call when connecting a modem
DCD	**D**ata **C**arrier **D**etect	Carrier signal when connecting a modem

1.4.2 Attributes of the 20 mA TTY Interface Submodule

Definition

The 20 mA TTY interface submodule is a current-loop interface used for serial data transmission.

Attributes

The 20 mA TTY interface submodule has the following attributes:

- Type: current-loop interface
- Front connector: 9-pin sub D female with screw fixing
- Max. transmission rate: 19.2 kbaud*
- Max. cable length: 1000 m at 9600 baud**
- Standard: DIN 66258 Part 1
- Degree of protection: IP00

* Please observe the maximum permitted transmission rates for the modules (see Section 2.3).
** See Appendix B for which cable types you may use.

1.4.3 Attributes of the X27 (RS 422/485) Interface Submodule

Definition

The X27 (RS 422/485) interface submodule is a differential voltage interface used for serial data transmission in compliance with the X27 standard.

Attributes

The X27 (RS 422/485) interface submodule has the following attributes:

- Type: differential voltage interface
- Front connector: 15-pin sub D female with screw fixing
- Max. transmission rate: 115.2 kbaud*
- Max. cable length: 1200 m at 19200 baud**
- Standard: DIN 66259 Parts 1 and 3,
  EIA-RS 422/485, CCITT V.11
- Degree of protection: IP00

Note

With the RK 512 and 3964(R) protocols, the X27 (RS 422/485) interface submodule can only be used in four-wire mode.

* Please observe the maximum permitted transmission rates for the modules (see Section 2.3).
** See Appendix B for which cable types you may use.

1.4.4 Cables for Connecting the CP 441 to a Communication Partner

Standard Connecting Cables

For point-to-point connection between the CP 441 and a communication partner, Siemens offers standard connecting cables in various lengths.

The lengths and order numbers of these cables are listed in Appendix E.

Constructing Your Own Connecting Cables

If you construct your own connecting cables, there are some points you must be aware of. These are described in Appendix B, along with wiring plans and the pin allocation for the sub D male connector.

1.5 Installation Guidelines

To be Observed

The general installation guidelines for S7-400 must be observed (see the *S7-400/M7-400 Programmable Controllers, Hardware and Installation* manual).

To meet the EMC (electromagnetic compatibility) values, the cable shield must be connected to a shield bus.

Basic Principles of Serial Data Transmission

2

2.1 Serial Transmission of a Character

For the exchange of data between two or more communication partners, various networking possibilities are available. The simplest form of data interchange is via a point-to-point connection between two communication partners.

Point-to-Point Connection

In a point-to-point connection the CP 441 communication processor forms the interface between a programmable controller and one or more communication partners. In a point-to-point connection with the CP 441, the data is transmitted serially.

Serial data transmission

In serial transmission, the individual bits of each byte of information are transmitted one after the other in a fixed order.

Uni/Bidirectional Data Traffic

The CP 441 itself handles data transmission with communication partners via the serial interface. The CP 441 is equipped with three different drivers for this purpose.

- **Unidirectional data traffic:**
  - Printer driver
- **Bidirectional data traffic:**
  - ASCII driver
  - 3964(R) procedure
  - RK 512 computer connection

The CP 441 handles data transmission via the serial interface in accordance with the interface type and the selected driver.

Unidirectional Data Traffic - Printer Output

In the case of printer output (printer driver), n bytes of user data are output to a printer. No characters are received. The only exception to this are data flow control characters (e.g. XON/XOFF).

Bidirectional Data Traffic – Operating Modes

The CP 441 has two operating modes for bidirectional data traffic:

- Half-duplex operation (3964(R) procedure, ASCII driver, RK 512)

  The data is exchanged between the communication partners in both directions alternately. In half-duplex operation, therefore, at any one time data is being either sent or received. The exception to this may be individual control characters for data flow control (e.g. XON/XOFF), which can also be sent during a receive operation or received during a send operation.

- Full-duplex operation (ASCII driver)

  The data is exchanged between the communication partners in both directions simultaneously. Each communication partner must be able to operate a send and a receive facility simultaneously.

With an RS 485 (2-wire) setting, the X27 (RS 422/485) interface submodule can only be run in half-duplex mode.

Asynchronous Data Transmission

With the CP 441, serial transmission occurs asynchronously. The so-called timebase synchronism (a fixed timing code used in the transmission of a fixed character string) is only upheld during transmission of a character. Each character to be sent is preceded by a synchronization impulse, or start bit. The end of the character transmission is signaled by the stop bit.

Declarations

As well as the start and stop bits, further declarations must be made between the two communication partners before serial transmission can take place. These include:

- Transmission speed (baud rate)
- Character and acknowledgment delay times
- Parity
- Number of data bits
- Number of stop bits

Sections 2.2 and 2.3 describe the importance of the declarations in the various transmission procedures, and how they are parameterized.

Character Frames

Data is transmitted between the communication processor and a communication partner via the serial interface in a character frame. Various data formats are available for the character frame. You can parameterize the format for data transmission with the parameterization interface *CP 441: Point-to-Point Communication, Parameter Assignment*.

The figure below shows examples of different data formats for a 10-bit character frame.

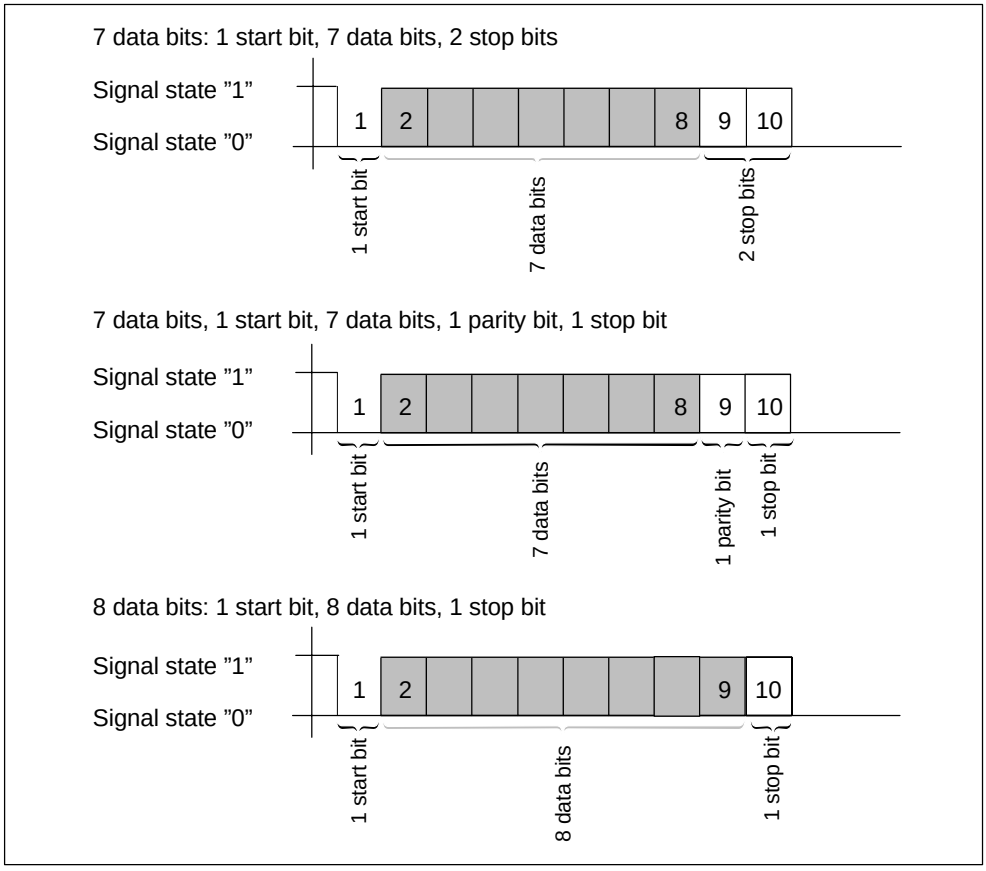


Figure 2-1 10-Bit Character Frame

Character Delay Time

The figure below shows the maximum time permitted between two characters received within a message frame. This is known as the character delay time.

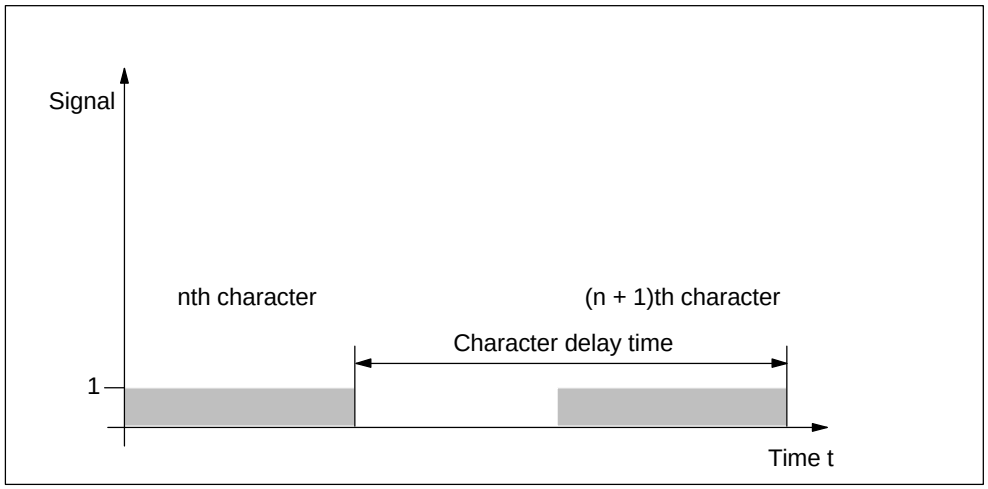


Figure 2-2 Character Delay Time

2.2 Transmission Procedure with a Point-to-Point Connection

When data is transmitted, all communication partners involved must follow fixed rules for handling and implementing the data traffic. The ISO has defined a 7-layer model, which is recognized as the basis for a worldwide standardization of transmission protocols for computer-to-computer communication.

2.2.1 ISO 7-Layer Reference Model for Data Transmission

Protocol

All communication partners involved in data transmission must follow fixed rules for handling and implementing the data traffic. Such rules are called protocols.

A protocol defines the following points:

- **Operating mode**

  Half-duplex or full-duplex mode

- **Initiative**

  Which communication partners can initiate the transmission and under what conditions

- **Control characters**

  Which control characters are to be used for data transmission

- **Character frame**

  Which character frames are to be used for data transmission

- **Data backup**

  The data backup procedure to be used

- **Character delay time**

  The time period within which an incoming character must be received

- **Transmission speed**

  The baud rate in bits/s

Procedure

This is the specific process according to which the data is transmitted.

ISO 7-Layer Reference Model

The reference model defines the external behavior of the communication partners. Each protocol layer, except for the lowest one, is embedded in the next one down.

The individual layers are as follows:

1. **Physical layer**
   - Physical conditions for communication, e.g. transmission medium, baud rate
2. **Data-connection layer**
   - Security procedure for the transmission
   - Access modes
3. **Network layer**
   - Network connections
   - Addressing for communication between two partners
4. **Transport layer**
   - Error-recognition procedure
   - Debugging
   - Handshaking
5. **Session layer**
   - Establishing communication
   - Data exchange management
   - Terminating communication
6. **Presentation layer**
   - Conversion of the standard form of data representation of the communication system into a device-specific form (data interpretation rules)
7. **Application layer**
   - Defining the communication task and the functions it requires

Processing the Protocols

The sending communication partner runs through the protocols from the highest layer (no. 7 - application layer) to the lowest (no. 1 - physical layer), while the receiving partner processes the protocols in the reverse order, i.e. starting with layer 1.

Not all protocols have to take all 7 layers into account. If the sending and receiving partners both use the same protocol, layer 6 can be omitted.

Transmission Integrity

Transmission integrity plays an important role in the transmission of data and in selection of the transmission procedure. Generally speaking, the more layers of the reference model are applied, the greater the transmission integrity.

Classifying the Supplied Protocols

The CP 441 governs the following protocols:

- 3964(R) procedure
- RK 512 computer connection
- ASCII driver
- Printer driver

The figure below illustrates how these protocols of the CP 441 fit into the ISO reference model:

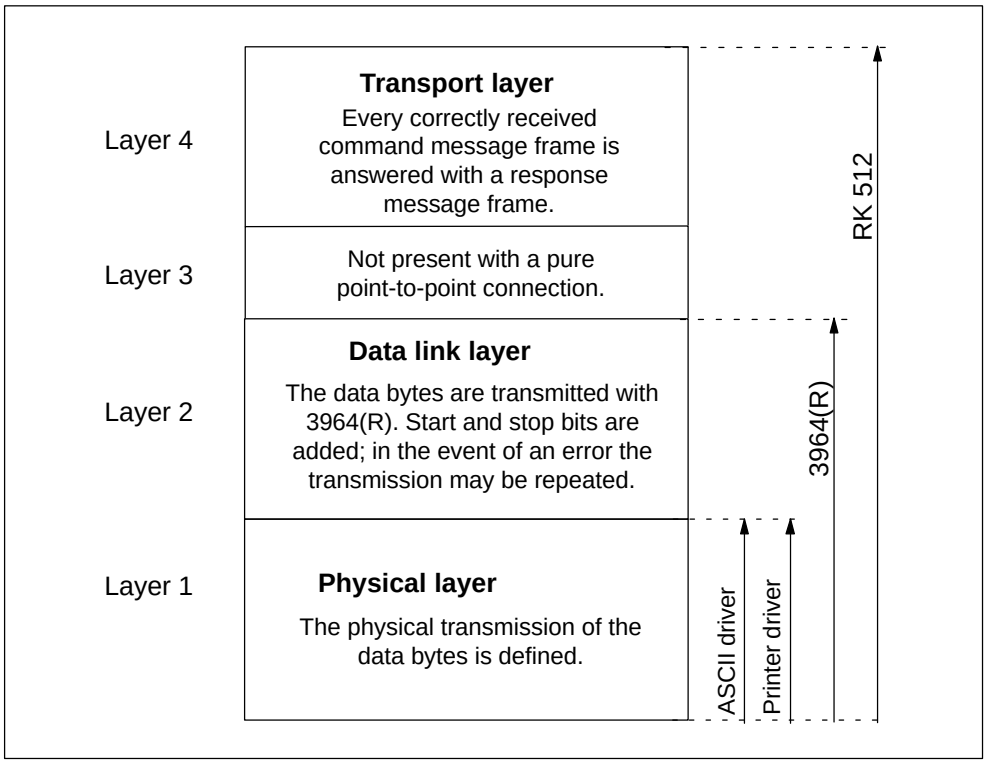


Figure 2-3 Position of the Supplied Protocols of the CP 441 in the ISO Reference Model

Transmission Integrity with the Printer Driver

Data Integrity When Using the Printer Driver:

- No data integrity precautions are taken for data transmission with the printer driver.
- To prevent data from being lost in the event of the printer receive buffer overflowing, you can work with data flow control (XON/XOFF, RTS/CTS).
- When data is output to the printer, the printer's BUSY signal is evaluated. The CP 441 receives the BUSY signal as a CTS signal and evaluates it in the same way (see ASCII driver). **Please note** that, in the case of CTS/RTS flow control, you must set the polarity of the BUSY signal to CTS = "OFF" on the printer (only in the case of the RS 232C interface; see Table 1-2).

Transmission Integrity with the ASCII Driver

Data Integrity When Using the ASCII Driver:

- When data is transmitted via the ASCII driver, there are no data integrity precautions other than the use of a parity bit (can also be canceled, depending on how the character frame is set). This means that, although this type of data transport has a very efficient throughput rate, security is not guaranteed.
- Using the parity bit ensures that the inversion of a bit in a character to be transmitted can be recognized. If two or more bits of a character are inverted, this error can no longer be detected.
- To increase transmission integrity, a checksum and length specification for a message frame can be employed. These measures must be implemented by the user.
- A further increase in data integrity can be achieved by means of acknowledgment message frames in response to send or receive message frames. This is the case with high-level protocols for data communication (see ISO 7-layer reference model).

Transmission Integrity with 3964

Enhanced Data Integrity with the 3964R Procedure:

- The Hamming distance with the 3964R is 3. This measures the integrity of data transmission.
- The 3964R procedure ensures high transmission integrity on the data line. This high integrity is achieved by means of a fixed message-frame setup and cleardown as well as the use of a block check character (BCC).

Two different procedures for data transmission can be used, either with or without a block check character:

- Data transmission without a block check character: **3964**
- Data transmission with block check character: **3964R**

In this manual, the designation **3964(R)** is used when descriptions and notes refer to both data transmission procedures.

Performance Limits with 3964(R)

Performance Limits of the 3964R Procedure:

- Further processing of the send/receive data by the PLC program in the communication partner is not guaranteed. You can only ensure this by using a programmable acknowledgment mechanism.
- The block check of the 3964R procedure (EXOR logic operation) cannot detect missing zeros (as a whole character) because a zero in the EXOR logic operation does not affect the result of the calculation.

  Although the loss of an entire character (this character has to be a zero!) is highly unlikely, it could possibly occur under very bad transmission conditions.

  You can protect a transmission against such errors by sending the length of the data message along with the data itself, and having the length checked at the other end.
- Such transmission errors are ruled out when the RK 512 computer connection is used for data transmission, because here (unlike the 3964(R) procedure) further processing is acknowledged via response message frames (e.g. stored in the destination data block) and the send data length is recorded in the message frame header. This enables the RK 512 to achieve a higher Hamming distance (of 4) than the 3964R.

Transmission Integrity with RK 512

Very High Data Integrity with the RK 512:

- The hamming distance with the RK 512 and 3964R is 4. This is a measure of the integrity of data transmission.
- Using the RK 512 computer connection guarantees high transmission integrity on the data line (because the RK 512 uses the 3964(R) procedure for data transport).
- Further processing in the communication partner is ensured (because the RK 512 interpreter checks the additional length specification in the header and, after storing the data in the destination data block of the communication partner, generates a message frame acknowledging the success or failure of the data transmission).
- The RK 512 driver guarantees the correct use of the 3964R procedure and the analysis/addition of the length specification as well as the independent generation of the response message frames. There is no user handling! All you need to do is evaluate the positive/negative final acknowledgment.

Performance Limits with RK 512

Performance Limits with RK 512

- The RK 512 computer connection provides a very high degree of data integrity. You can improve this still further by, for example, using other block check mechanisms (e.g. CRC checks).

2.2.2 Data Transmission with the 3964(R) Procedure

The 3964(R) procedure controls data transmission via a point-to-point connection between the CP 441 and a communication partner. As well as the physical layer (layer 1), this procedure also incorporates the data-connection layer (layer 2).

Control Characters

During data transmission, the 3964(R) procedure adds control characters to the information data (data-connection layer). These control characters allow the communication partner to check whether the data has arrived complete and without errors.

The 3964(R) procedure analyzes the following control codes:

- **STX** Start of Text; Start of character string for transfer
- **DLE** Data Link Escape; Data connection escape
- **ETX** End of Text; End of character string for transfer
- **BCC** Block check character (3964R only) Block check character
- **NAK** Negative Acknowledge; Negative acknowledgment

Note

If DLE is transmitted as an information string, it is sent twice so that it can be distinguished from the control code DLE during connection setup and release on the send line (DLE duplication). The receiver then reverses the DLE duplication.

Priority

With the 3964(R) procedure, one communication partner must be assigned a higher priority and the other partner a lower priority. If both partners begin connection setup at the same time, the partner with the lower priority will defer its send request.

Block Checksum

With the 3964R transmission protocol, data integrity is increased by the additional sending of a block check character (BCC).

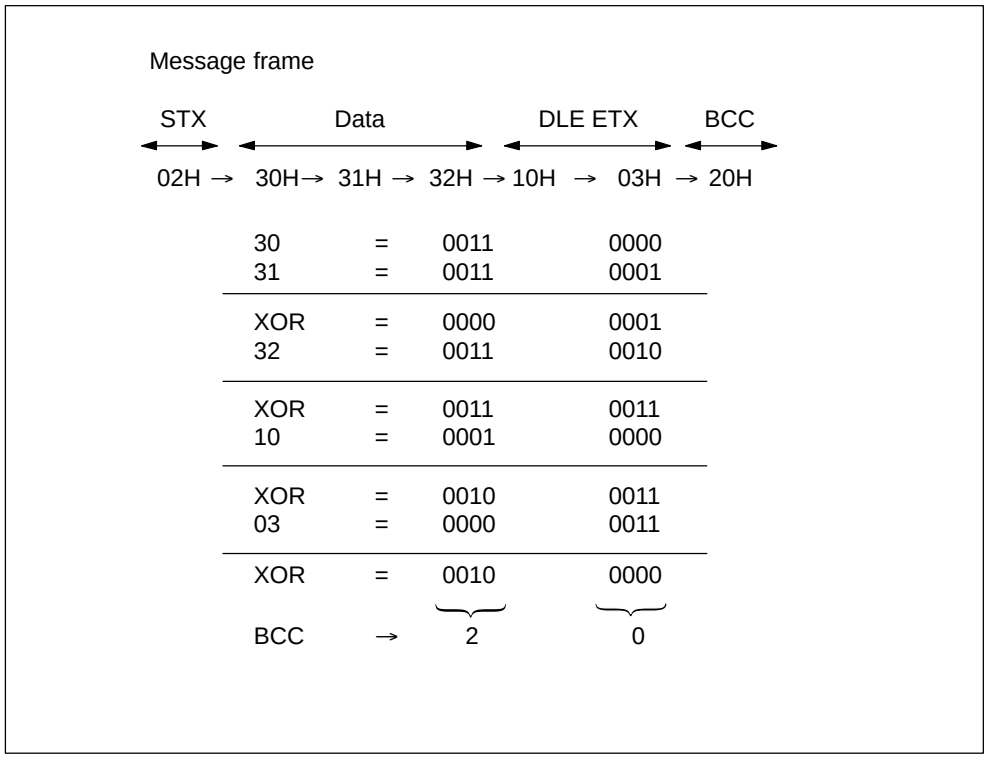


Figure 2-4 Block Checksum

The block checksum is the even longitudinal parity (EXOR logic operation of all data bytes) of a sent or received block. Its calculation begins with the first byte of user data (first byte of the message frame) after the connection setup, and ends after the DLE ETX code on connection release.

Note

If DLE duplication occurs, the DLE code is accounted for twice in the BCC calculation.

Sending Data with 3964(R)

The figure below illustrates the transmission sequence when data is sent with the 3964(R) procedure.

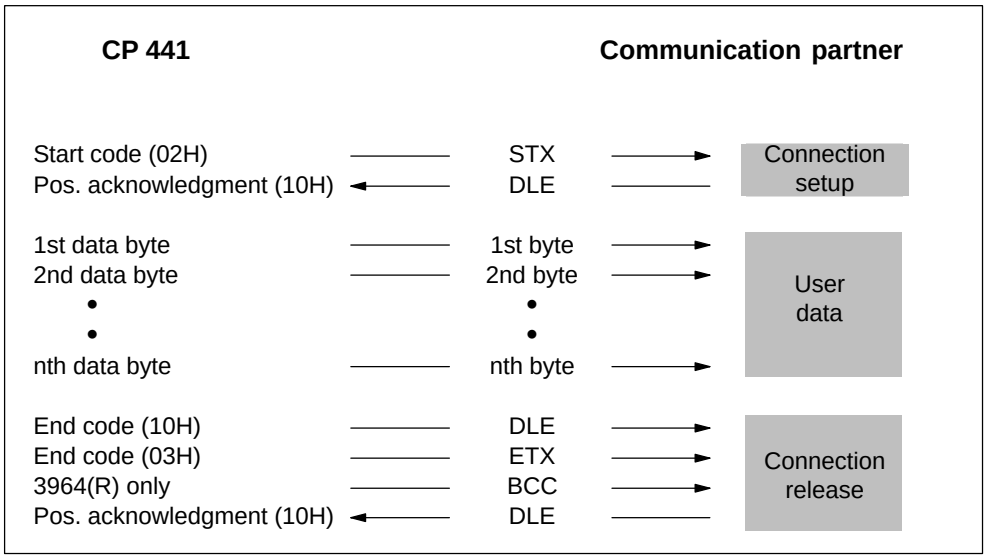


Figure 2-5 Data Traffic when Sending with the 3964(R) Procedure

Establishing a Send Connection

To establish the connection, the 3964(R) procedure sends the control code STX. If the communication partner responds with the DLE code before the acknowledgment delay time (ADT) expires, the procedure switches to send mode.

If the communication partner answers with NAK or with any other control code (except for DLE or STX), or the acknowledgment delay time expires without a response, the procedure repeats the connection setup. After the defined number of unsuccessful setup attempts, the procedure aborts the connection setup and sends the NAK code to the communication partner. The CP 441 enters an appropriate error number in its SYSTAT area.

Send data

If a connection is successfully established, the user data contained in the output buffer of the CP 441 is sent to the communication partner with the chosen transmission parameters. The partner monitors the times between incoming characters. The interval between two characters must not exceed the character delay time.

If the communication partner sends the NAK control code during an active send operation, the procedure aborts its transmission of the block and tries again as described above, beginning with connection setup. If a different code is sent, the procedure first waits for the character delay time to expire and then sends the NAK code to change the mode of the communication partner to idle. Then the procedure starts to send the data again with the connection setup STX.

Releasing a Send Connection

Once the contents of the buffer have been sent, the procedure adds the codes DLE, ETX and **with the 3964R only** the block checksum BCC as the end identifier, and waits for an acknowledgment code. If the communication partner sends the DLE code within the acknowledgment delay time, the data block has been received without errors. If the communication partner responds with NAK, any other code (except DLE), or a damaged code, or if the acknowledgment delay time expires without a response, the procedure starts to send the data again with the connection setup STX.

After the defined number of attempts to send the data block, the procedure stops trying and sends an NAK to the communication partner. The CP 441 reports the error in the SYSTAT error-signaling area.

Receiving Data with 3964(R)

The figure below illustrates the transmission sequence when data is received with the 3964R procedure.

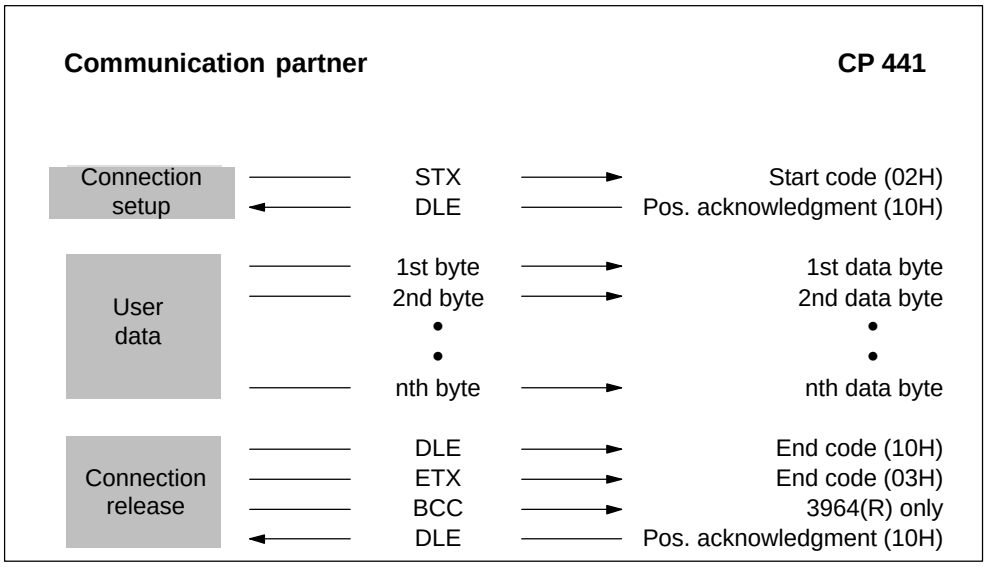


Figure 2-6 Data Traffic when Receiving with the 3964(R) Procedure

Note

As soon as it is ready, the 3964(R) procedure sends a single NAK to the communication partner to set the latter to idle.

Establishing a Receive Connection

In idle mode, when there is no send request to be processed, the procedure waits for the communication partner to establish the connection.

If no empty receive buffer is available during a connection setup with STX, a wait time of 400 ms is started. If there is still no empty receive buffer after this time has expired, the system program reports the error (error number in the SYSTAT), and the procedure sends an NAK and returns to idle mode. Otherwise, the procedure sends a DLE and receives the data as described above.

If the idle procedure receives any control code except for STX or NAK, it waits for the character delay time to expire, then sends the code NAK. The CP 441 reports the error in the SYSTAT error-signaling area.

Receive data

After a successful connection setup, the receive characters that are arrive are stored in the receive buffer. If two consecutive DLE codes are received, only one of these is stored in the receive buffer.

After each receive character, the procedure waits out the character delay time for the next character. If this period expires before another character is received, an NAK is sent to the communication partner. The CP 441 reports the error in the SYSTAT error-signaling area. The 3964(R) procedure does not initiate a repetition.

If transmission errors occur during receiving (lost character, frame error, parity error, etc.), the procedure continues to receive until the connection is released, then an NAK is sent to the communication partner. A repetition is then expected. If the undamaged block still cannot be received after the number of transmission attempts defined in the static parameter set, or if the communication partner does not start the repetition within a block wait time of 4 seconds, the procedure aborts the receive operation. The CP 441 reports the first errored transmission and the final abortion in the SYSTAT error-signaling area.

Releasing a Receive Connection

If the **3964** procedure recognizes the string DLE ETX, it stops receiving and sends to the communication partner a DLE if the block was received without errors. If the block is damaged it sends an NAK. A repetition is then expected.

If the **3964R** procedure recognizes the string DLE ETX BCC, it stops receiving and compares the received BCC with the internally calculated longitudinal parity. If the BCC is correct and no other receive errors have occurred, the 3964R procedure sends a DLE and returns to idle mode. If the BCC is errored or a different receive error occurs, an NAK is sent to the communication partner. A repetition is then expected.

Handling Errored Data

The figure below illustrates how errored data is handled with the 3964(R) procedure.

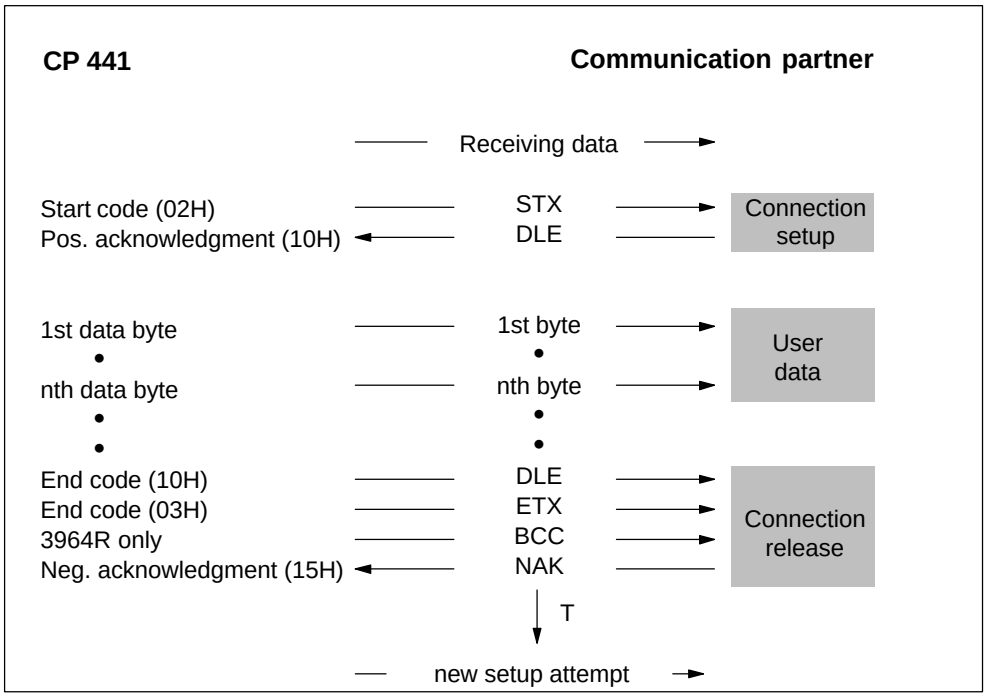


Figure 2-7 Data Traffic when Receiving Errored Data

When the string DLE ETX BCC is received, the CP 441 compares the BCC of the communication partner with its own internally calculated value. If the BCC is correct and no other receive errors occur, the CP 441 responds with DLE.

Otherwise, the CP 441 responds with an NAK and waits the block wait time (T) of 4 seconds for a new attempt. If after the defined number of transmission attempts the block cannot be received, or if no further attempt is made within the block wait time, the CP 441 aborts the receive operation.

Initialization Conflict

The figure below illustrates the transmission sequence during an initialization conflict.

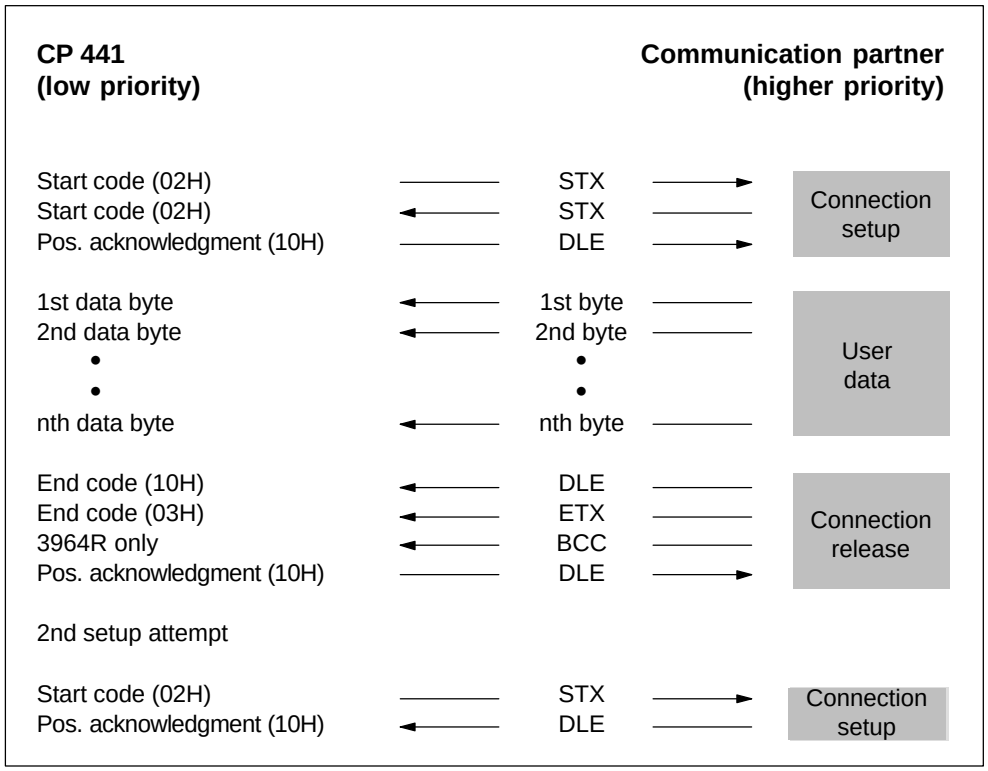


Figure 2-8 Data Traffic during an Initialization Conflict

If a device responds to the communication partner's send request (code STX) within the acknowledgment delay time (ADT) by sending the code STX instead of the acknowledgment DLE or NAK, an initialization conflict occurs. Both devices want to execute a send request. The device with the lower priority withdraws its send request and responds with the code DLE. The device with the higher priority sends its data in the manner described above. Once the connection has been released, the lower-priority device can execute its send request.

To be able to resolve initialization conflicts you must set different priorities for the communication partners.

Procedure Errors

The procedure recognizes both errors which are caused by the communication partner and errors caused by faults on the line.

In both cases, the procedure makes repeated attempts to send/receive the data block correctly. If this is not possible within the maximum number of transmission attempts set (or if a new error status occurs), the procedure aborts the send or receive process. It reports the error number of the first recognized error and returns to idle mode. The CP 441 reports the error in the SYSTAT error-signaling area.

If the CP 441 frequently reports the error number in the SYSTAT for send and receive repetitions, this implies occasional disturbances in the data traffic. The large number of transmission attempts compensates for this, however. In this case you are advised to check the transmission connection for possible sources of interference, because frequent repetitions reduce the user-data rate and integrity of the transmission. The disturbance could also be caused, however, by a malfunction on the part of the communication partner.

If the receive connection is interrupted, the system program reports a BREAK status (in SYSTAT) No repeat is started. The BREAK status in the SYSTAT is automatically reset as soon as the connection is restored on the line.

For every recognized transmission error (lost character, frame or parity error), a standard number is reported, regardless of whether the error was detected during sending or receiving of a data block. The error is only reported, however, following unsuccessful repetitions.

If the damaged character is received when the procedure is idle, the system program reports the error (error number in the SYSTAT) to inform you of major interference in the data transmission circuit.

Section 8.4 explains how you can carry out a diagnostics analysis via the error-signaling area SYSTAT.

3964(R) Procedure Start-Up

The figure below illustrates the start-up of the 3964(R).

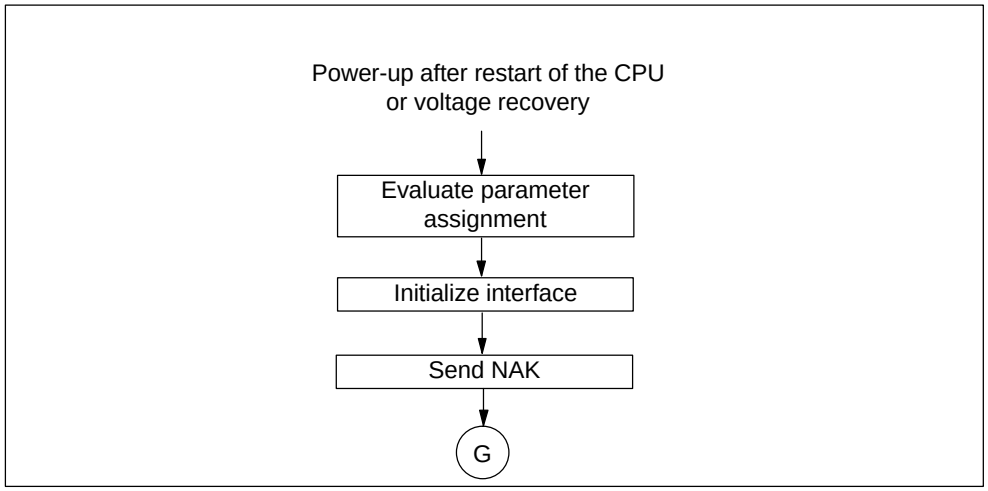


Figure 2-9 Flow Diagram of the Start-Up of the 3964(R) Procedure

Sending with the 3964(R) Procedure

The figure below illustrates sending with the 3964(R) procedure.

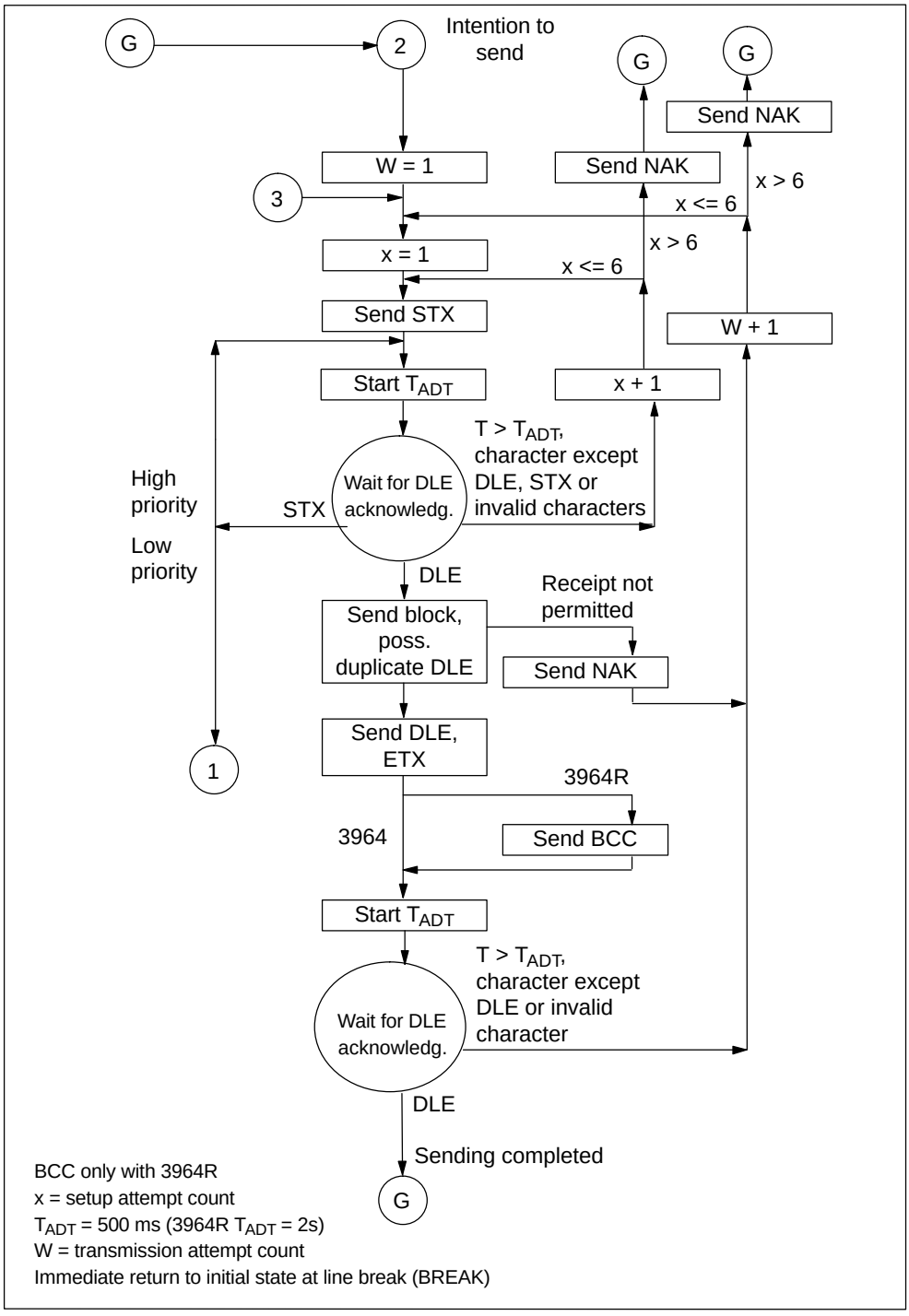


Figure 2-10 Flow Diagram of Sending with the 3964(R) Procedure

Receiving with the 3964(R) Procedure (Part 1)

The figure below illustrates receiving with the 3964(R) procedure.

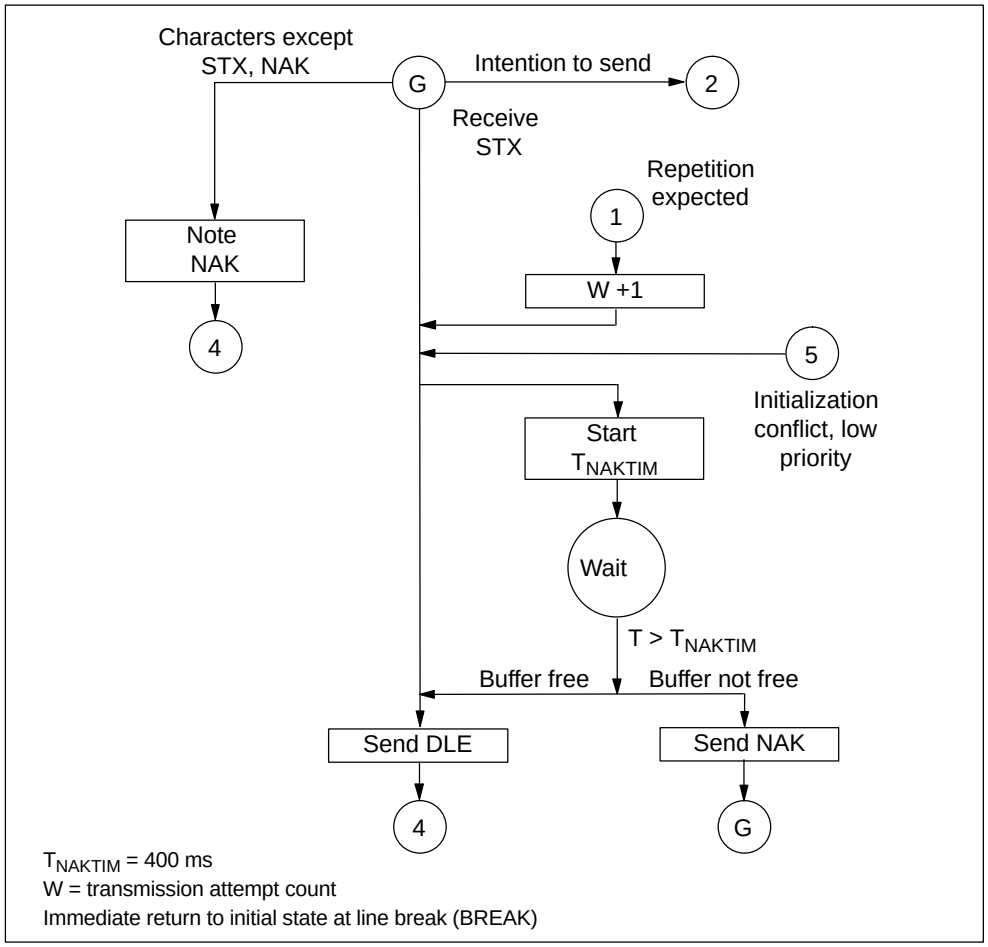


Figure 2-11 Flow Diagram of Receiving with the 3964(R) Procedure (Part 1)

Receiving with the 3964(R) Procedure (Part 2)

The figure below illustrates receiving with the 3964(R) procedure.

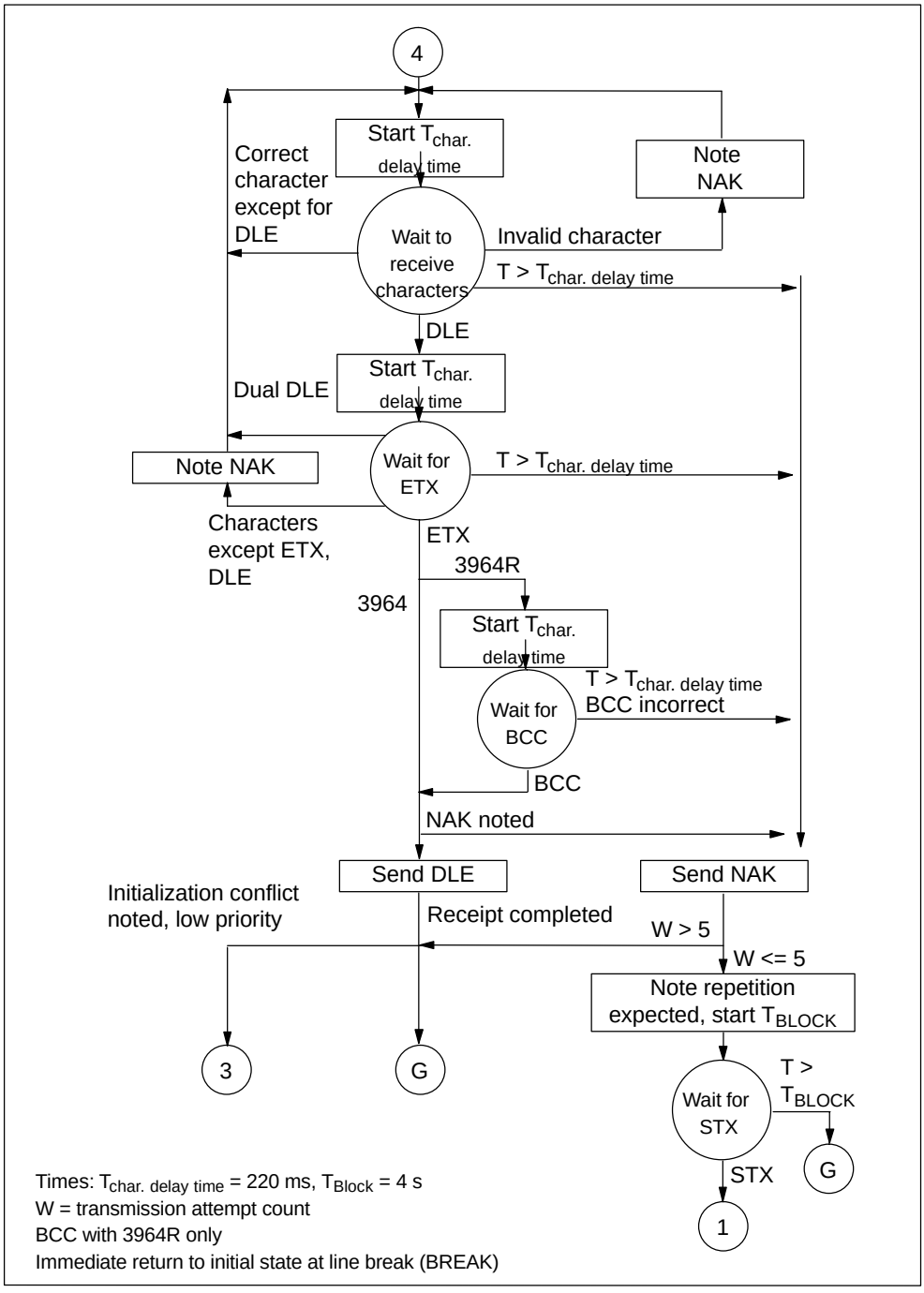


Figure 2-12 Flow Diagram of Receiving with the 3964(R) Procedure (Part 2)

2.2.3 Data Transmission with the RK 512 Computer Connection

The RK 512 computer connection controls data transmission via a point-to-point connection between the CP 441 and a communication partner.

Unlike the 3964(R) procedure, the RK 512 includes not only the physical layer (layer 1), and the data-link layer (layer 2), but also the transport layer (layer 4) of the ISO reference model. The RK 512 computer connection also offers higher data integrity and better addressing.

Response Message Frame

The RK 512 computer connection answers every command message frame it receives correctly with a response message frame to the CPU (transport layer). This allows senders to check whether their data has arrived undamaged at the CPU or whether the data they require is available on the CPU.

Command Message Frame

Command message frames are either SEND/PUT or GET message frames.

How to initiate a SEND/PUT or GET message frame is described in Chapter 6.

SEND/PUT Message Frame

A SEND/PUT message frame is created when the CP 441 sends a command message frame with user data, and the communication partner replies with a response message frame without user data.

GET Message Frame

A GET message frame is created when the CP 441 sends a command message frame without user data, and the communication partner replies with a response message frame with user data.

Continuation Message Frame

If the volume of data exceeds 128 bytes, SEND/PUT and GET message frames are automatically accompanied by continuation message frames.

Message frame header

Each message frame with the RK 512 begins with a message frame header. It can contain message frame IDs, information on the data destination and source and an error number.

Structure of the Message Frame Header

The table below indicates the structure of the header of the command message frame.

Table 2-1 Structure of Message Frame Header (RK 512)

Byte	Meaning
1	Message frame ID in command message frames (00H), in continuation command message frames (FFH)
2	Message frame ID (00H)
3	'A' (41H) – for SEND/PUT request with destination DB or 'O' (4FH) – for SEND request with destination DX or 'E' (45H) – for GET request
4	Data to be transmitted consists of:[1] 'D' (44H) = data block 'E' (45H) = input bytes 'A' (41H) =output bytes 'M' (4DH) = flag bytes 'C' (5AH) = counter cells 'T' (54H) =time cells
5 6	**Data destination** of SEND/PUT request or **data source** of GET request e.g. byte 5 = DB no., byte 6 = DW no.[2]
7 8	Length of high byte Length of data to be transmitted according to type in bytes or Length of low byte words
9	Byte number of the interprocessor communication flag[3]; FFH is displayed if you have not specified an interprocessor communication flag.
10	Bits 0 to 3: Bit number of the interprocessor communication flag[3]; if you have not specified an interprocessor communication flag, the protocol enters FH here. Bits 4 to 7: CPU number (digit from 1 to 4)[4]; 0H is displayed here if you have not specified a CPU No. but you have specified an interprocessor communication flag[3]; FH is displayed here if you have not specified a CPU No. or an interprocessor communication flag.

1 The entry for send requests with BSEND and PUT is always "D", irrespective of the areas from which the data come.

2 RK 512 addressing describes the data source and destination with word limits. Conversion to byte addresses in SIMATIC S7 is automatic.

3 You can specify interprocessor communication flags for send requests with "BSEND". In the block of the partner you cannot specify interprocessor communication flags, because the CP itself does not support them.

4 The CPU number 0 is supported as of STEP 7, Version 4.0.

The letters in bytes 3 and 4 are ASCII characters.

The header of the continuation command message frame consists of bytes 1 to 4 only.

Response Message Frame

Once the command message frame has been transmitted, the RK 512 waits for a response message frame from the communication partner within the monitoring time. The length of the monitoring time depends on the transmission speed (baud rate).

- 300 to 115200 baud 10 seconds

Structure and Contents of the Response Message Frame

The response message frame consists of 4 bytes and contains information on the progress of the request.

Byte	Meaning
1	Message frame ID in response message frames (00H), in continuation response message frames (FFH)
2	Message frame ID (00H)
3	Displays 00H
4	Error number of the communication partner (see Section 8.4) in the response message frame:* 00H if transmission was error-free > 00H error number

* The error number in the response message frame automatically causes an error number to be entered in the SYSTAT. The error numbers of the response message frame and of SYSTAT can be found in Chapter 8.

Sending Data with RK 512

The figure below shows the transmission process when sending data with a response message frame using the RK 512 computer connection.

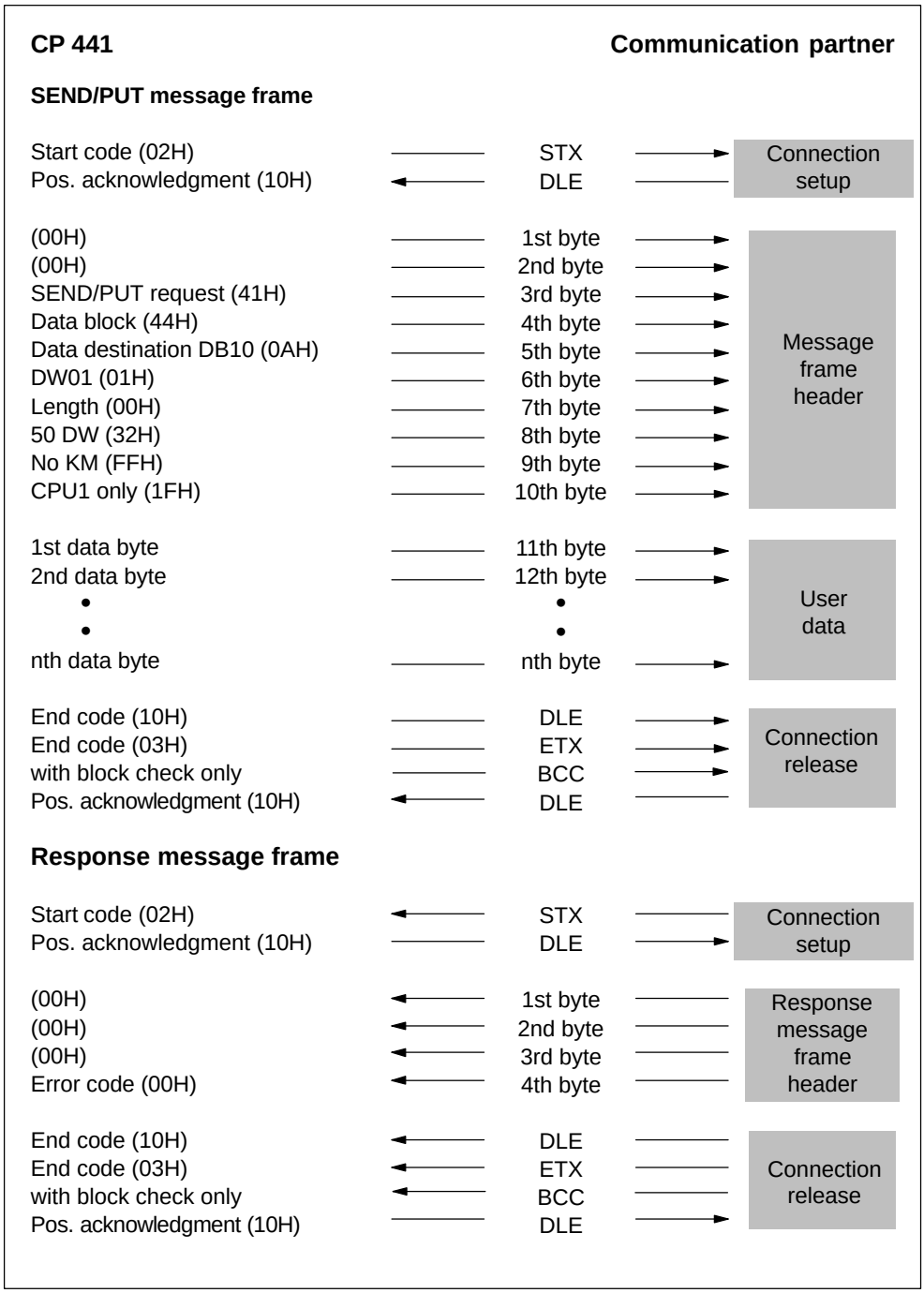


Figure 2-13 Data Traffic when Sending with a Response Message Frame

Send data

The SEND/PUT request is executed in the following sequence:

- **Active partner**

  sends a SEND/PUT message frame containing the message frame header and data.

- **Passive partner**

  receives the message frame, checks the header and the data, and acknowledges it with a response message frame after passing the data on to the CPU.

- **Active partner**

  receives the response message frame.

  Sends user data.

  If the volume of user data exceeds **128 bytes**, the active partner sends a continuation SEND/PUT message frame.

- **Passive partner**

  receives the continuation SEND/PUT message frame, checks the header and the data, and acknowledges it with a continuation response message frame after passing the data on to the CPU.

Note

If the CPU receives an errored SEND/PUT message frame or if an error has occurred in the message frame header, the communication partner enters an error number in the 4th byte of the response message frame. This does not apply in the case of protocol errors.

Continuation SEND/PUT Message Frame

A continuation SEND/PUT message frame is started if the volume of data exceeds **128 bytes**. The process is the same as for SEND/PUT message frames.

If more than 128 bytes are sent, the extra bytes are automatically transmitted in one or more continuation message frames.

The figure below shows the data transmission process when sending a continuation SEND/PUT message frame with a continuation response message frame.

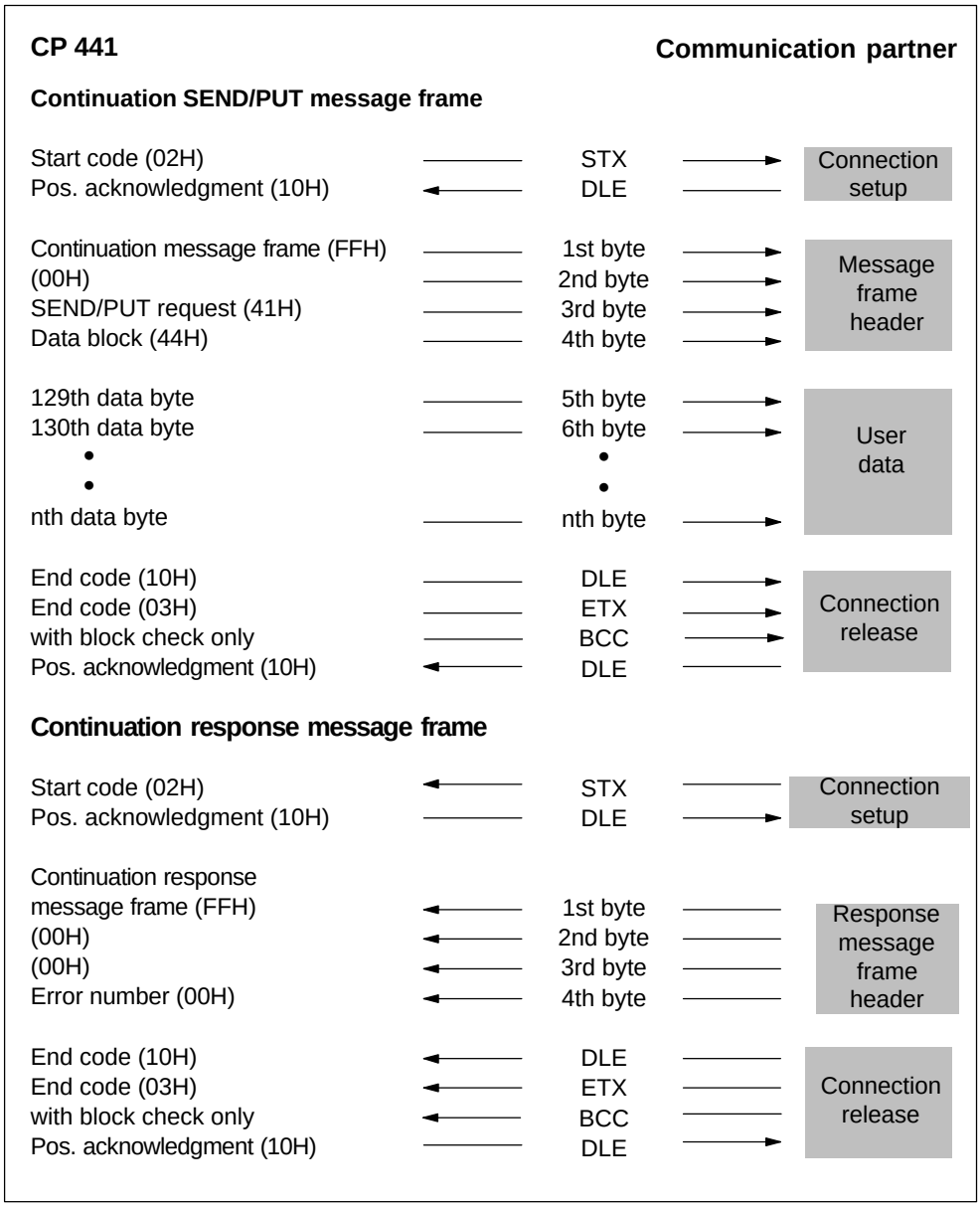


Figure 2-14 Sequence of a Continuation SEND Message Frame with a Continuation Response Message Frame

Fetching Data with RK 512

The figure below shows the transmission process when fetching data with a response message frame using the RK 512 computer connection.

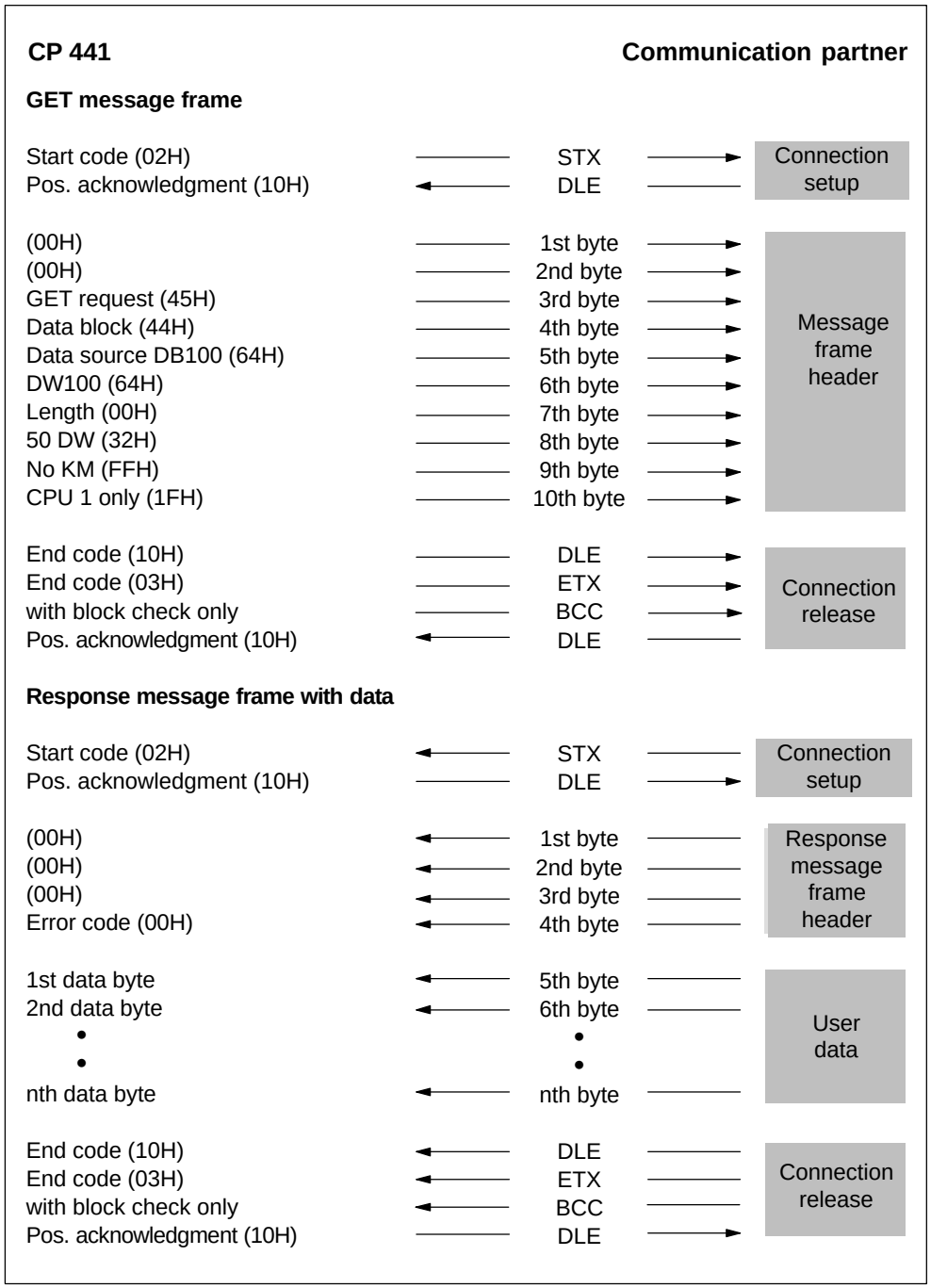


Figure 2-15 Data Traffic when Fetching with a Response Message Frame

Fetching Data

The GET request is executed in the following sequence:

- **Active partner**

  sends a GET message frame containing the header.

- **Passive partner**

  receives the message frame, checks the header, fetches the data from the CPU, and acknowledges this with a response message frame containing the data.

- **Active partner**

  receives the response message frame.

  If the volume of user data exceeds **128 bytes**, the active partner sends a continuation GET message frame containing bytes 1 to 4 of the header.

- **Passive partner**

  receives the continuation GET message frame, checks the header, fetches the data from the CPU, and acknowledges this with a continuation response message frame containing further data.

If there is an error number (not equal to 0) in the 4th byte, the response message frame does not contain any data.

If more than 128 bytes are requested, the extra bytes are automatically fetched in one or more continuation message frames.

Note

If the CPU receives an errored GET message frame or if an error has occurred in the message frame header, the communication partner enters an error number in the 4th byte of the response message frame. This does not apply in the case of protocol errors.

Continuation GET Message Frame

The figure below shows the transmission process when fetching data with a continuation response message frame.

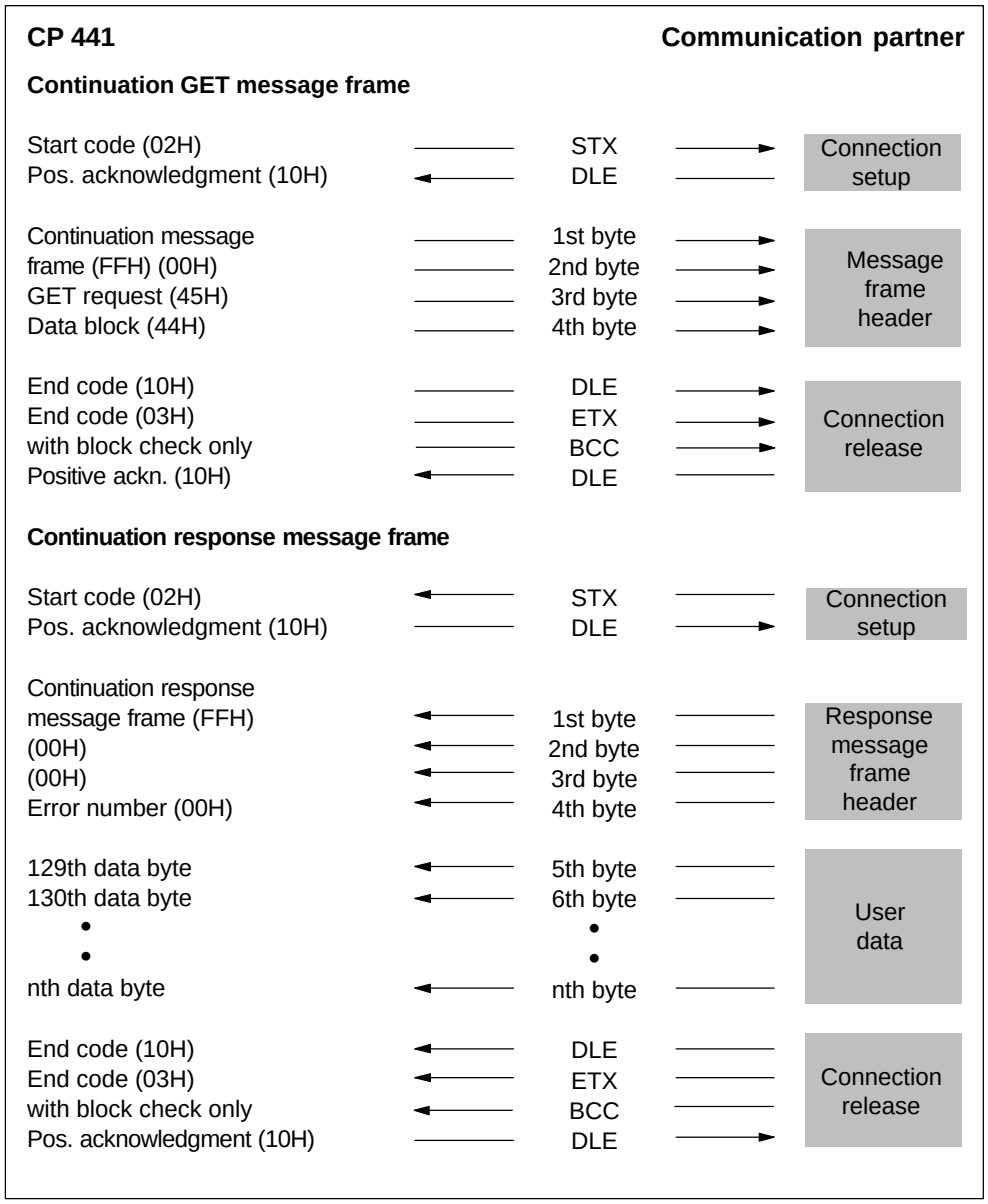


Figure 2-16 Sequence of a Continuation GET Message Frame with a Continuation Response Message Frame

Quasi-Full-Duplex Operation

Quasi full-duplex operation means: the partners can send command and response message frames at any time as long as the other partner is not sending. The maximum nesting depth for command and response message frames is “1”. The next command message frame, therefore, cannot be processed until the previous one has been answered with a response message frame.

It is possible under certain circumstances - if both partners want to send - to transmit a SEND/PUT message frame from the partner before the response message frame. For example, if a SEND/PUT message frame from the partner was entered in the output buffer of the CP 441 before the response message frame.

In the following figure the continuation response message frame to the first SEND/PUT message frame is not sent until after the **partner’s SEND/PUT message frame.**

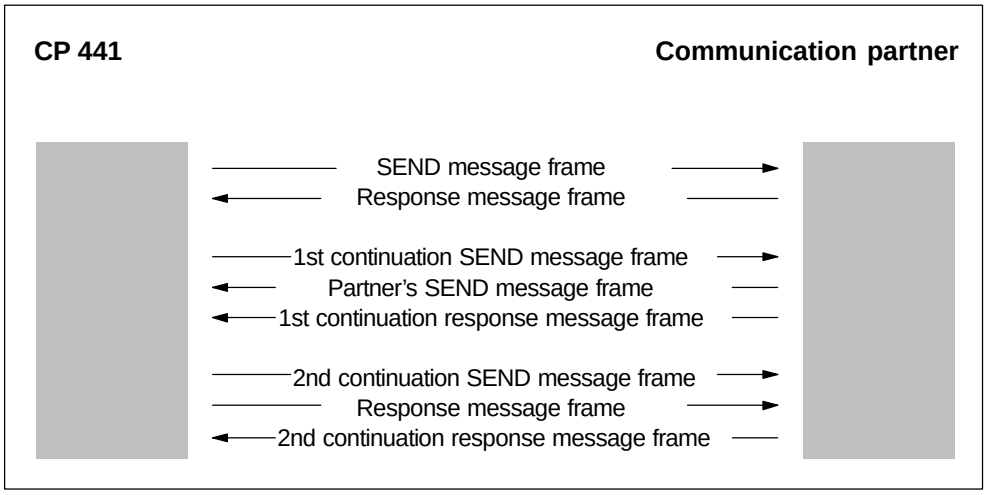


Figure 2-17 Quasi-Full-Duplex Operation

RK 512 CPU Requests

The figure below shows the processes involved in the RK 512 computer connection when CPU requests are made.

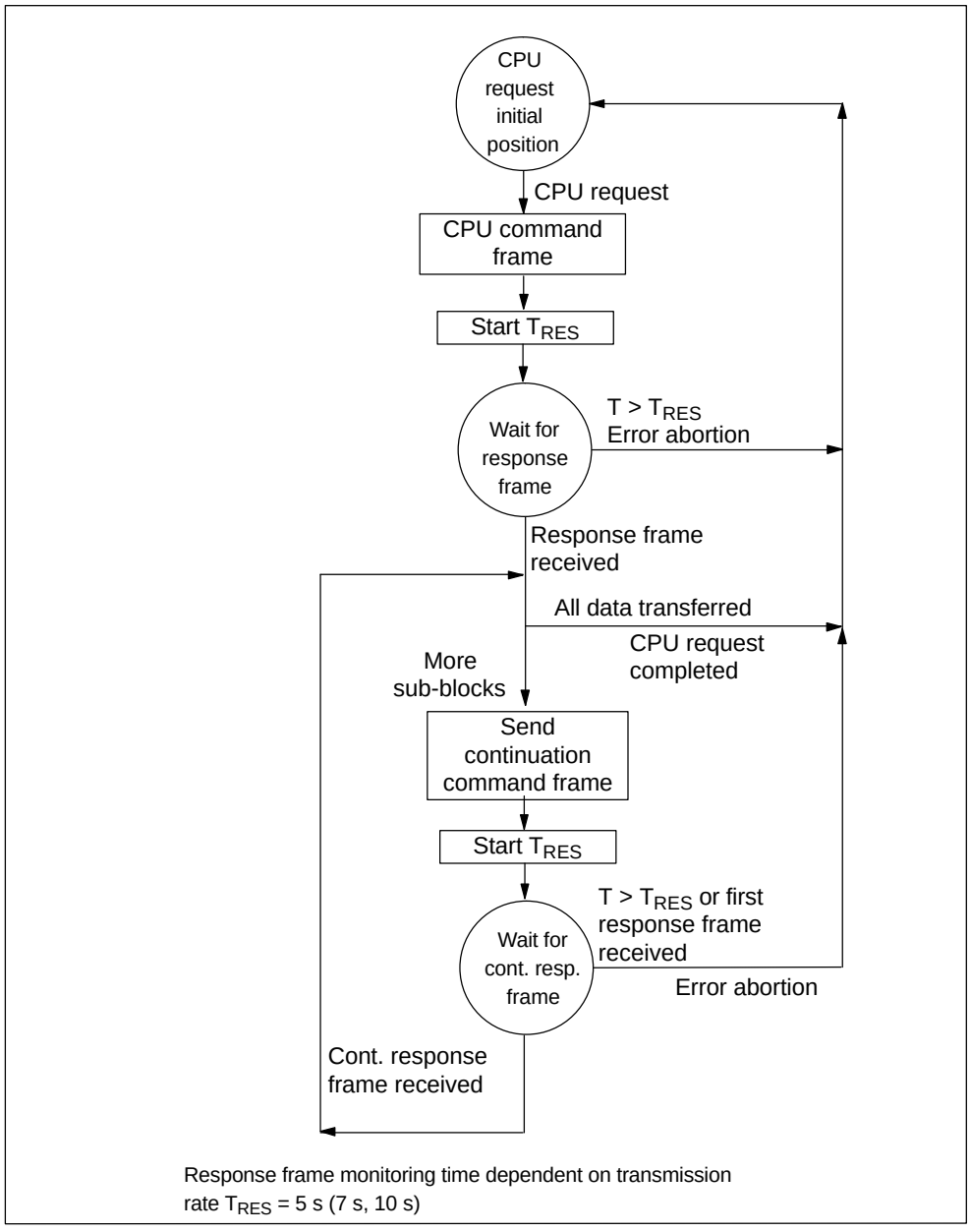


Figure 2-18 Flow Diagram of Data Transmission with the RK 512 When CPU Requests Are Made

RK 512 Partner Requests

The figure below shows the processes involved in the RK 512 computer connection when partner requests are made.

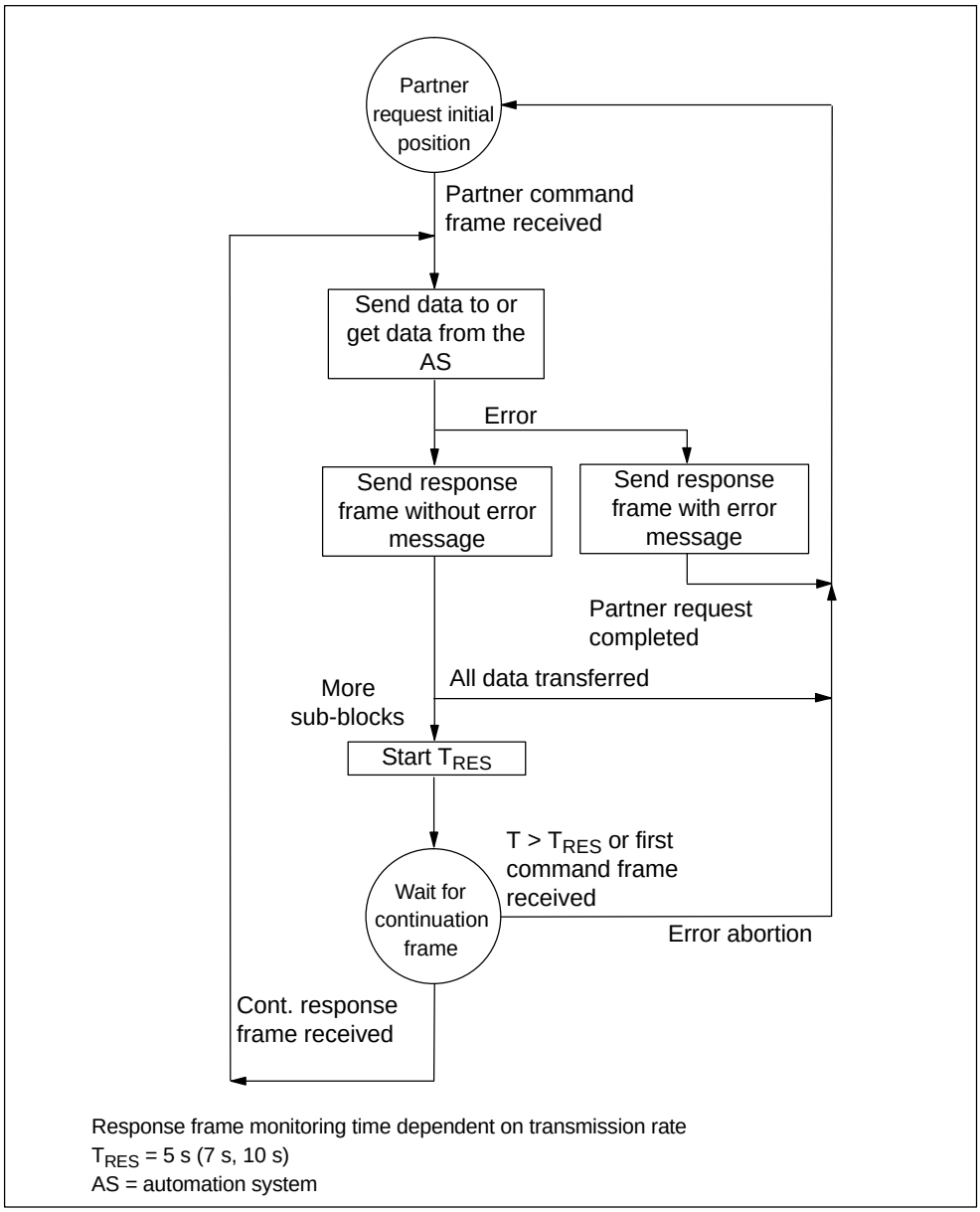


Figure 2-19 Flow Diagram of Data Transmission with the RK 512 When Partner Requests Are Made

2.2.4 Data Transmission with the ASCII Driver

The ASCII driver controls data transmission via a point-to-point connection between the CP 441 and a communication partner. This driver contains the physical layer (layer 1 of the ISO reference model.)

The structure of the message frames is left open through the S7 user passing on the complete send message frame to the CP 441. For the receive direction, the end criterion of a message must be parameterized. The structure of the send message frames may differ from that of the receive message frames.

The ASCII driver allows data of any structure (all printable ASCII characters as well as all other characters from 00 through FFH (with 8 data bit character frames) or from 00 through 7FH (with 7 data bit character frames) to be sent and received.

Sending Data with the ASCII Driver

When you send data, you specify the number of user data bytes to be transferred in the ”LEN” parameter of the call for the BSEND system function block.

When you work with the end criterion **”Character Delay Time”** when receiving data, the ASCII driver pauses between two message frames when sending. You can call the BSEND SFB at any time, but the ASCII driver does not begin with output until a time greater than the parameterized character delay time has passed since the last message frame was sent.

Note

When XON/XOFF flow control is parameterized, the user data must not contain any of the parameterized XON or XOFF codes. The default settings are DC1 = 11H for XON and DC3 = 13H for XOFF.

If you work with the **”End-of-Text Character”** criterion, you have a choice of three options:

- Send up to and including the end-of-text character

  The end-of-text character must be included in the data to be sent. Data is sent only up to and including the end-of-text character, even if the data length specified in the FB is longer.

- Send up to length parameterized at the FB

  Data is sent up to the length parameterized at the FB. The last character must be the end-of-text character.

- Send up to the length parameterized at the FB and automatically append the end-of-text character or characters

  Data is sent up to the length parameterized at the FB. The end-of-text character is automatically appended, in other words the end-of-text characters must not be included in the data to be sent. 1 or 2 characters more than the number specified at the FB are sent to the partner, depending on the number of end-of-text characters.

When you work with the end criterion **”Fixed Message Frame Length”**, the number of data bytes transferred in the send direction is as specified for the ”LEN” parameter of the BSEND. The number of data bytes transferred in the receive direction, i.e. in the receive DB, is as specified at the receiver using the ”fixed message frame length” parameter in the parameterization interface. The two parameter settings must be identical, in order to ensure correct data traffic. A pause equal to the length of the character delay time is inserted between two message frames when sending, to allow the partner to synchronize (recognize start of message frame).

If some other method of synchronization is used, the pause in sending can be deactivated by means of the parameter assignment interface.

Send data

The figure below illustrates a send operation.

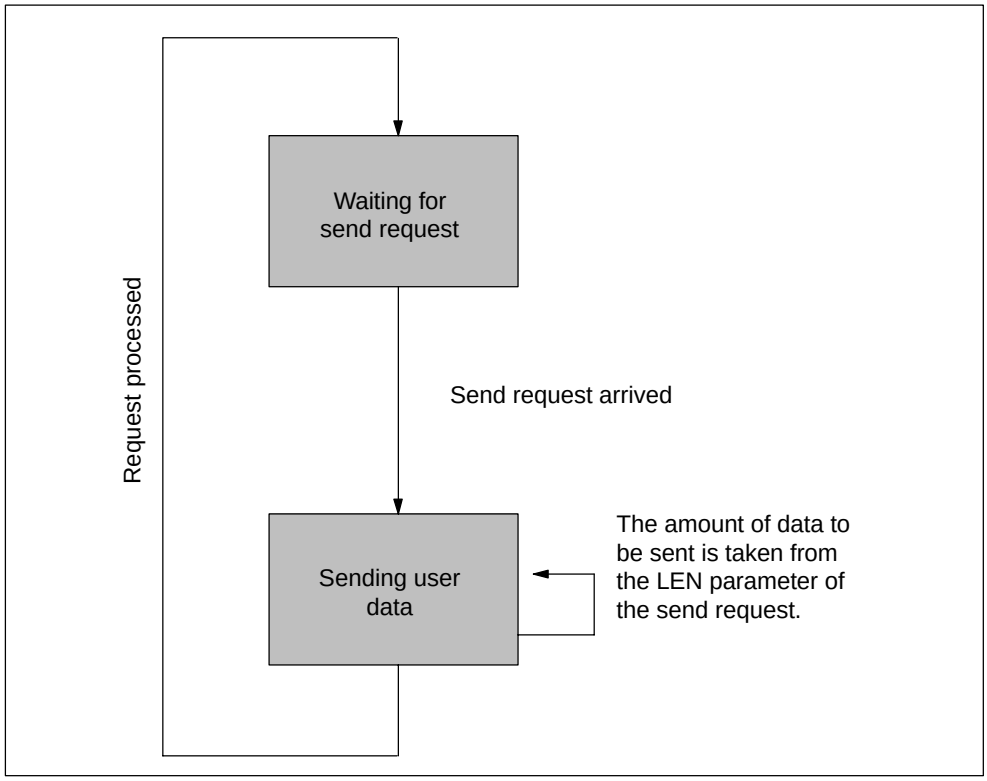


Figure 2-20 Sequence of a Send Operation

Receiving Data with the ASCII Driver

For data transmission using the ASCII driver you can choose between three different end criteria. The end criterion defines when a complete message frame is received. The possible end criteria are as follows:

- **On Expiry of Character Delay Time**

  The message frame has neither a fixed length nor a defined end-of-text character; the end of the message is defined by a pause on the line (expiry of character delay time).

- **On Receipt of End Character(s)**

  The end of the message frame is marked by one or two defined end-of-text characters.

- **On Receipt of Fixed Number of Characters**

  The length of the receive message frames is always identical.

Code Transparency

The code transparency of the procedure depends on the selection of the parameterized end criterion and the flow control:

- With one or two end-of-text characters
  - not code-transparent
- When end criterion is character delay time or fixed message frame length
  - code-transparent
- Code-transparent operation is not possible when XON/XOFF flow control is used.

Code-transparent means that any character combinations can occur in the user data without the end criterion being recognized.

End Criterion ”Expiry of Character Delay Time”

When data is received, the end of the message frame is recognized when the character delay time expires. The received data is accepted from the CPU.

In this case the character delay time must be set such that it easily expires between two consecutive message frames. But it should be long enough so that the end of the message frame is not falsely identified whenever the partner in the connection takes a send pause within a message frame.

The figure below illustrates a receive operation with the end criterion ”expiry of character delay time”.

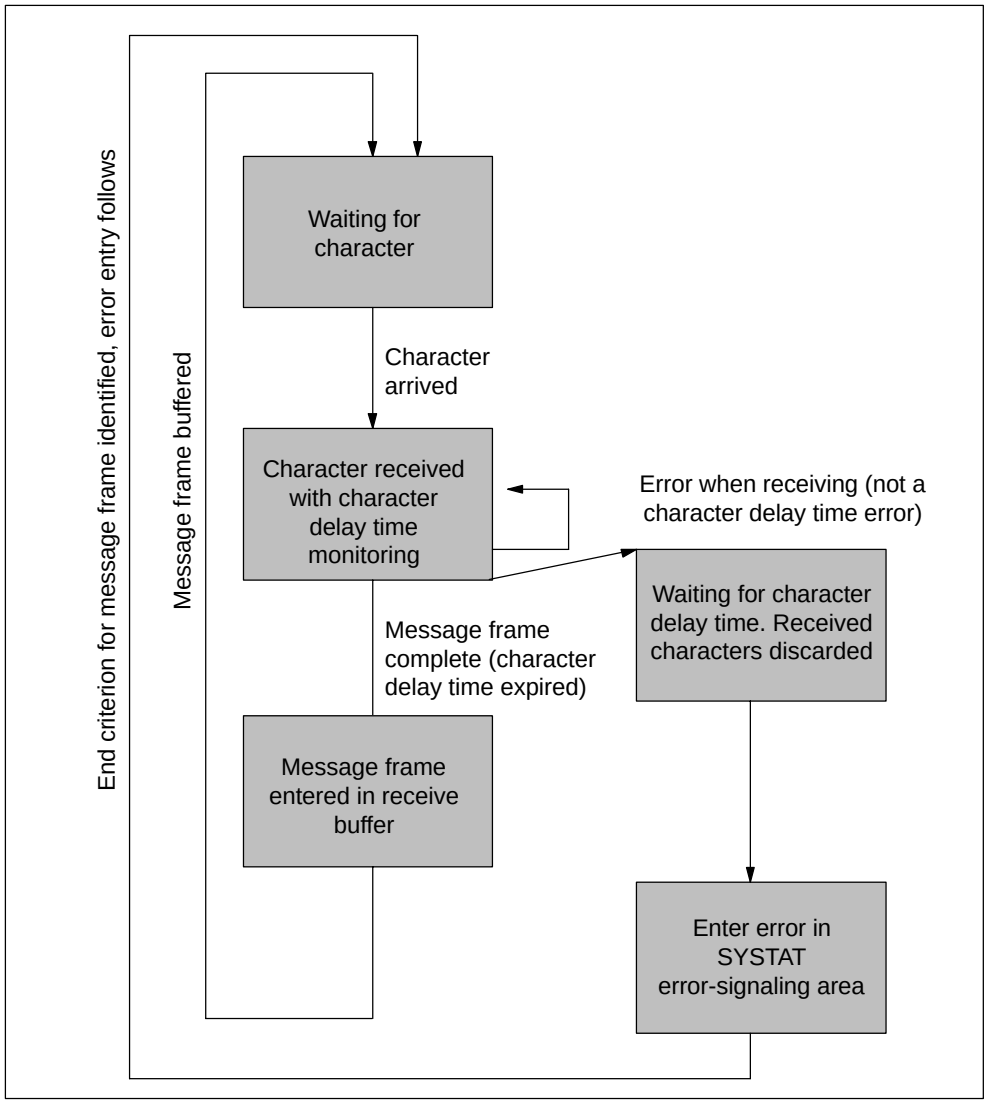


Figure 2-21 Sequence of Receive Operation with End Criterion ”Expiry of Character Delay Time”

End Criterion ”End-of-Text Character”

When data is received, the end of the message frame is recognized when the parameterized end-of-text character(s) arrive. The received data including the end-of-text character(s) is accepted from the CPU.

If the character delay time expires while the message frame is being received, the receive operation is terminated. An error message is issued and the message frame fragment is discarded.

If you are working with end-of-text characters, transmission is not code-transparent, and you must make sure that the end code(s) are not included in the user data of the user.

Note the following when the last character in the received message frame is not the end-of-text character.

- End-of-text character elsewhere in the message frame:

  All characters including the end-of-text character are entered in the receive DB. The characters following the end-of-text character

  – are discarded if the character delay time expires at the end of the message frame.

  – are blended with the next message frame if a new message frame is received before the character delay time expires.

- End-of-text character **not** included in message frame:

  The message frame

  – is discarded if the character delay time expires at the end of the message frame.

  – is blended with the next message frame if a new message frame is received before the character delay time expires.

The figure below illustrates a receive operation with the end criterion ”end-of-text character”.

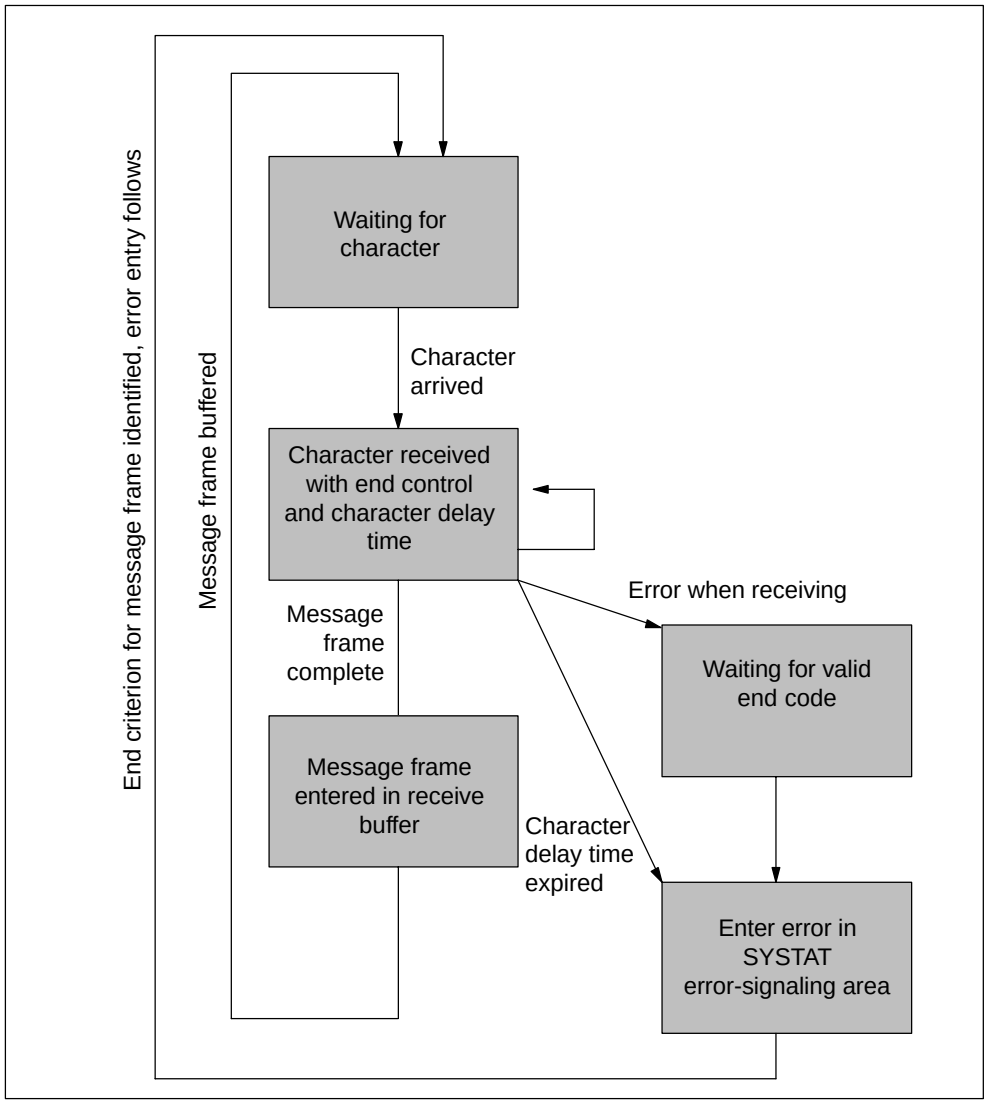


Figure 2-22 Sequence of Receive Operation with End Criterion ”End-of-Text Character”

End Criterion ”Fixed Message Frame Length”

When data is received, the end of the message frame is recognized when the parameterized number of characters has arrived. The received data is accepted from the CPU.

If the character delay time expires before the parameterized number of characters has been reached, the receive operation is terminated. An error message is issued and the message frame fragment is discarded.

Note the following if the message frame length of the received characters does not match the parameterized fixed message frame length:

- Message frame length of received characters greater than parameterized fixed message frame length:

  All characters received after the parameterized fixed message frame length is reached

  – are discarded if the character delay time expires at the end of the message frame.

  – are blended with the next message frame if a new message frame is received before the character delay time expires.

- Message frame length of received characters less than parameterized fixed message frame length:

  The message frame

  – is discarded if the character delay time expires at the end of the message frame.

  – is blended with the next message frame if a new message frame is received before the character delay time expires.

The figure below illustrates a receive operation with the end criterion ”fixed message frame length”.

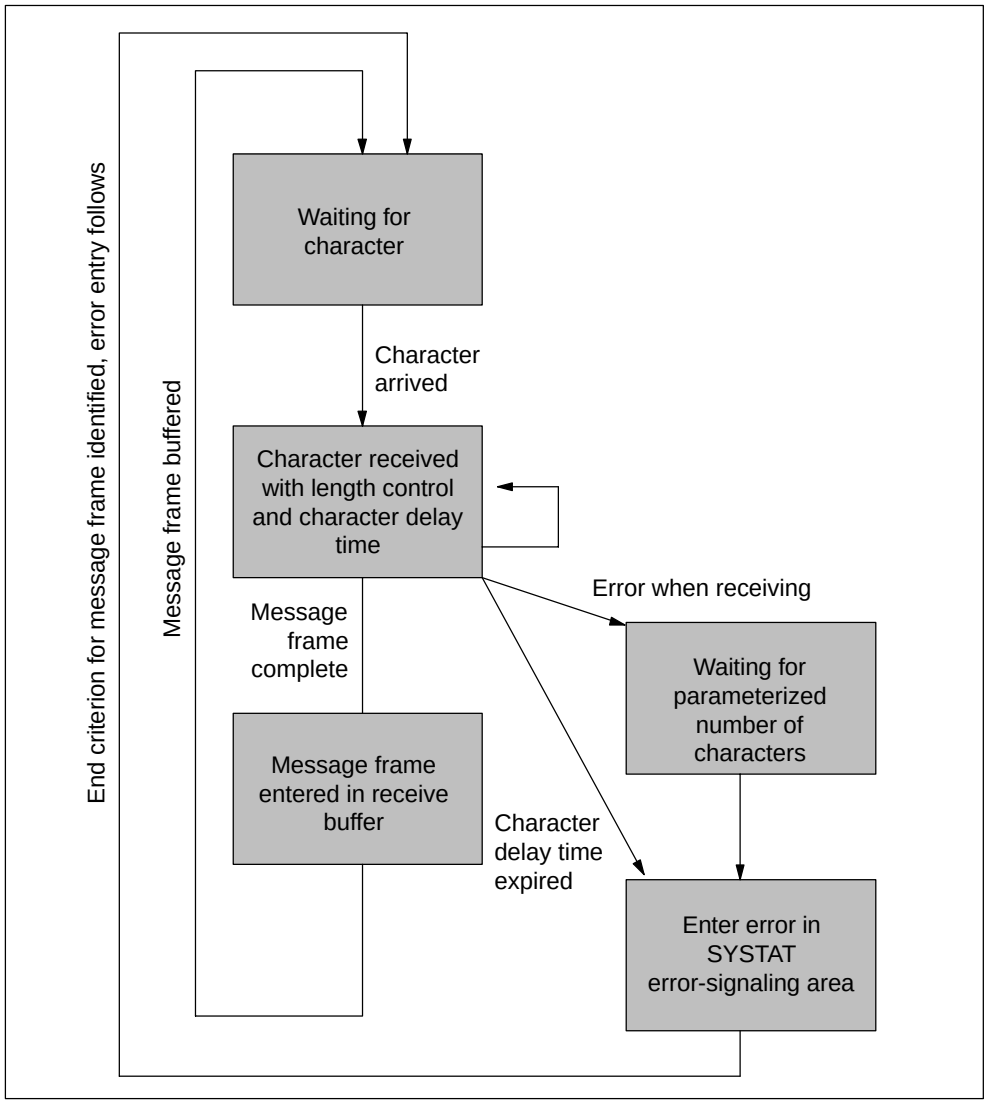


Figure 2-23 Sequence of Receive Operation with End Criterion ”Fixed Message Frame Length”

Receive Buffer on CP 441

The CP 441 receive buffer accommodates 4096 bytes. At parameterization you can specify whether overwriting of data in the receive buffer should be prevented. You can also specify the value range (1 to 250) for the number of buffered receive message frames.

The receive buffer on the CP 441 is a ring buffer:

- If two or more message frames are received into the receive buffer of the CP 441, the rule is that the oldest message frame is always transferred by the CP 441 to the CPU.
- If you only ever want to send the most recent message frame to the CPU, you must parameterize the value ”1” for the number of buffered message frames **and** deactivate the overwrite protection.

Note

If the constant reading of the receive data in the user program is interrupted for a while, you may find that when the receive data is requested again, the CP 441 transfers old message frames to the CPU before it transfers the latest one.

The old message frames were either on the way between the CP 441 and CPU or had already been received by the SFB.

RS 485 Mode

When you run the ASCII driver in RS 485 mode (half-duplex, two-wire mode), you must take steps in the user program to ensure that only one user sends data at any one time. If two users send data simultaneously, the message frame is corrupted.

RS 232C Secondary Signals

The following RS 232C secondary signals exist on the CP 441 when the RS 232C interface submodule is used (see also Appendix B):

- **DCD** (input) — Data carrier detect; Data carrier detected
- **DTR** (output) — Data terminal ready; CP 441 ready for operation
- **DSR** (input) — Data set ready; Communication partner ready for operation
- **RTS** (output) — Request to send; CP 441 ready to send
- **CTS** (input) — Clear to send; Communication partner can receive data from the CP 441 (response to RTS = ON of the CP 441)
- **RI** (input) — Ring indicator; Indication of an incoming call

When the CP 441 is switched on, the output signals are in the OFF state (inactive).

You can parameterize the way in which the DTR/DSR and RTS/CTS control signals are used with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface or control them by means of function calls (FBs) in the user program.

Using the RS 232C Secondary Signals

The RS 232C secondary signals can be used as follows:

- When the automatic use of all RS 232C secondary signals is parameterized
- When data flow control (RTS/CTS) is parameterized
- By means of the V24_STAT and V24_SET functions (FBs)

Note

When automatic use of the RS 232C secondary signals is parameterized, neither RTS/CTS data flow control nor RTS and DTR control by means of the V24_SET FB are possible.

When RTS/CTS data flow control is parameterized, RTS control by means of the V24_SET FB is not possible.

On the other hand, it is always possible to read all RS 232C secondary signals by means of the V24_STAT FB.

The sections that follow describe how the control and evaluation of the RS 232C secondary signals is handled.

Automatic Use of the Secondary Signals

The automatic use of the RS 232C secondary signals on the CP 441 is implemented as follows:

- As soon as the CP 441 is switched by means of parameterization to an operating mode with automatic use of the RS 232C secondary signals, it switches the RTS line to OFF and the DTR line to ON (CP 441 ready for use).

  Message frames cannot be sent and received until the DTR line is set to ON. As long as DTR remains set to OFF, no data is received via the RS 232C interface. If a send request is made, it is aborted with an error message.
- When a **send request** is made, RTS is set to ON and the parameterized data output waiting time starts. When the data output time elapses and CTS = ON, the data is sent via the RS 232C interface.
- If the CTS line is not set to ON within the data output time so that data can be sent, or if CTS changes to OFF during transmission, the send request is aborted and an error message generated.
- After the data is sent, the RTS line is set to OFF after the parameterized time to RTS OFF has elapsed. The CP340 does not wait for CTS to change to OFF.
- Data can be **received** via the RS 232C interface as soon as the DSR line is set to ON. If the receive buffer of the CP 441 threatens to overflow, the CP 441 does not respond.
- A send request or data receipt is aborted with an error message if DSR changes from ON to OFF. The message ”DSR = OFF (automatic use of V24 signals)” is entered in the diagnostics buffer of the CP 441.

Note

Automatic use of the RS 232C secondary signals is only possible in half-duplex mode.

When automatic use of the RS 232C secondary signals is parameterized, neither RTS/CTS data flow control nor RTS and DTR control by means of the V24_SET FB are possible.

Note

The “time to RTS OFF” must be set in the parameterization interface so that the communication partner can receive the last characters of the message frame in their entirety before RTS, and thus the send request, is taken away. The “data output waiting time” must be set so that the communication partner can be ready to receive before the time elapses.

Time Diagram

Figure 2-24 illustrates the chronological sequence of a send request.

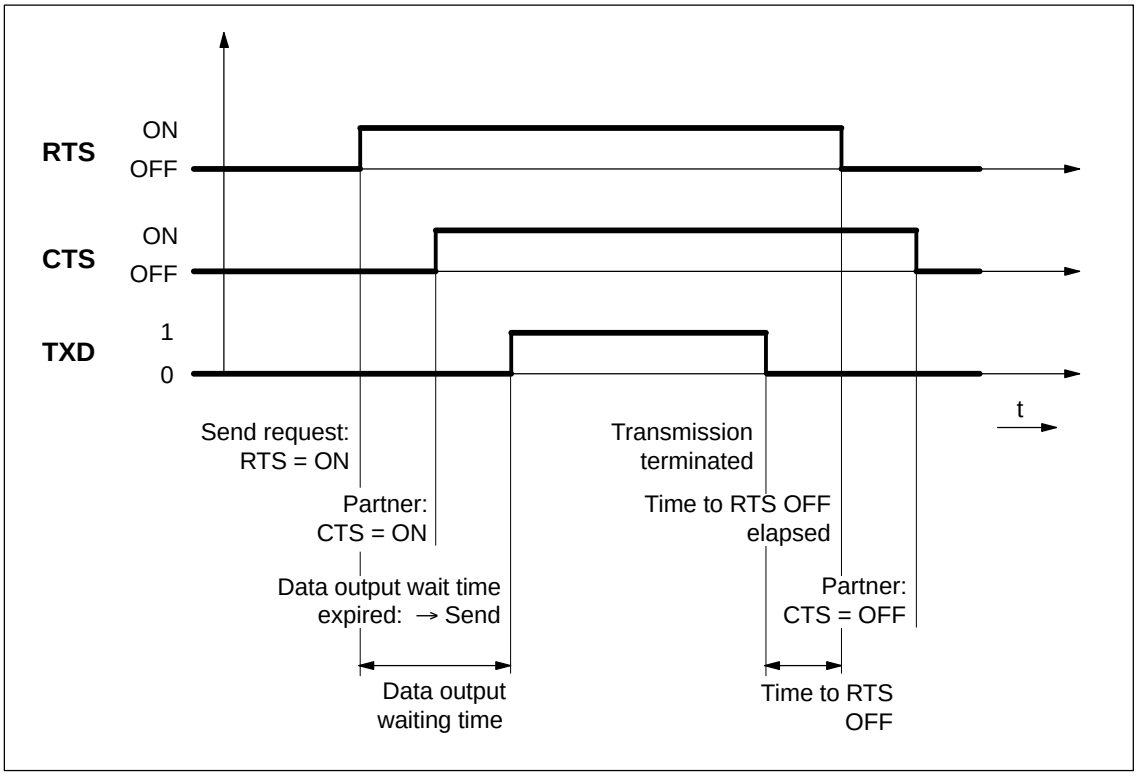


Figure 2-24 Time Diagram for Automatic Use of the RS 232C Secondary Signals

Data Flow Control/ Handshaking

Handshaking controls the data flow between two communication partners. Handshaking ensures that data is not lost in transmissions between devices that work at different speeds. There are essentially two types of handshaking:

- Software handshaking (e.g. XON/XOFF)
- Hardware handshaking (e.g. RTS/CTS)

Data flow control is implemented as follows on the CP 441:

- As soon as the CP 441 is switched by parameterization to an operating mode with flow control, it sends the XON character or sets the RTS line to ON.
- When the parameterized number of message frames is reached, or alternatively 50 characters before the receive buffer overflows (size of the receive buffer: 4096 bytes), the CP 441 sends the XOFF character or sets the RTS line to OFF. If the communication partner continues to send data regardless of this, the receive buffer overflows and an error message is generated. The data received in the last message frame is discarded.
- As soon as a message frame is fetched by the S7 CPU and the receive buffer is ready to receive, the CP 441 sends the XON character or sets the RTS line to ON.
- If the CP 441 receives the XOFF character, or the CTS control signal is set to OFF, the CP 441 interrupts the transmission. If neither an XON character is received nor CTS is set to ON before a parameterized time has elapsed, the transmission is aborted and an appropriate error message (0708H) is entered in the SYSTAT error-signaling area of the CP 441.

Note

When RTS/CTS data flow control is parameterized, you must fully wire the interface signals in the plug connection (see Appendix B).

When RTS/CTS data flow control is parameterized, RTS control by means of the V24_SET FB is not possible.

Reading/Control with the V24_STAT and V24_SET FBs

The FB V24_STAT function allows the status of each RS 232C secondary signal to be determined. The V24_SET function allows the DTR and RTS output signals to be controlled (see Section 6.6).

Switch-over Times for RS485 Module in Half-Duplex Mode

The maximum switch-over time between sending and receiving is 1 ms.

This value is applicable to modules as of MLFB number 6ES7 441-XAA03 / -0AE0

2.2.5 Data Transmission with the Printer Driver

The printer driver allows you to output message texts with the date and time to a printer. This enables you to monitor simple processes, print error or fault messages or issue instructions to the operating personnel, for example.

The printer driver contains the physical layer (layer 1).

Message Texts and Parameters for Printer Output

Use the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface to configure the message texts and set the parameters (page layout, character set, control characters) for printer output. Message texts and printer output parameters are transmitted to the CP 441 together with the module parameters when it starts up.

Message texts:

You can configure message texts with variables and control statements (e.g. for bold, condensed, expanded or italic type and underlining). Each message text is assigned a number during configuration. You print out a specific message text by specifying a reference (to the memory cell containing the message text number) for send parameters SD_1 to SD_4 of the PRINT system function block.

Page layout:

You can configure the margins, possible line breaks and headers and footers.

Character set:

The ANSI character set is converted to the printer character set by STEP 7 by means of a character conversion table. You can change a character conversion table suggested for a printer type in order to include special characters required for a particular language, for example.

Control characters:

By means of a control character table you can change the control statements in the message text for the printer emulation for switching on and off bold, condensed, expanded or italic type and underlining and to add control characters.

Printer Output

To output n bytes of user data to a printer, the format string and the variables of the message text must be specified as parameters when the PRINT system function block is called.

During output the data is edited for printing. The print job is processed in accordance with the parameter settings as set with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface (page layout, character set, control characters, etc.).

Characters are not received during printer output. The exception to this are any flow control characters that have been parameterized. Any characters received are not adopted.

Message Text Output

The figure below illustrates the sequence of operations at printer output.

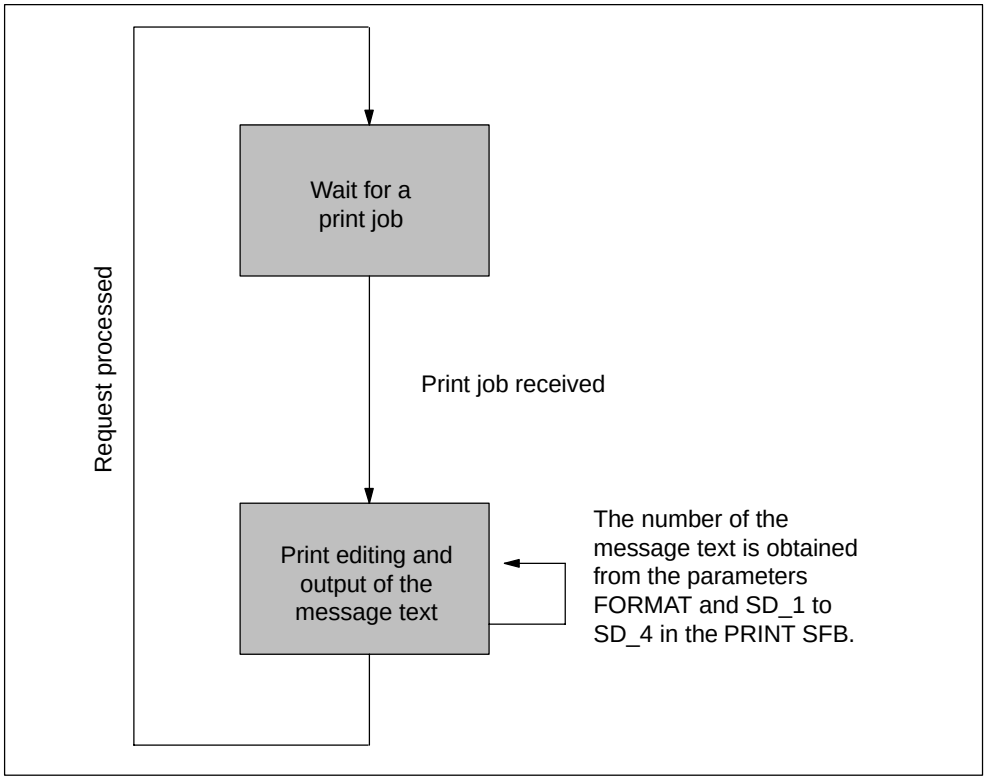


Figure 2-25 Flow Chart of Printer Output

2.3 Parameterization Data of the Protocols

By selecting different protocols, you can adjust your CP 441 communication processor to suit the communication partner.

The sections that follow describe the parameterization data of the 3964(R) procedure, RK 512 computer connection, ASCII driver and printer driver.

2.3.1 Parameterization Data of the 3964(R) Procedure

Using the parameterization data of the 3964(R) procedure, you can adjust the CP 441 to suit the properties of the communication partners.

Parameterization Data of the 3964(R) Procedure

Using the *CP 441 Point-to-Point Communication, Parameter Assignment* interface you specify the parameters for the physical layer (layer 1) and the data-connection layer (layer 2) of the 3964(R) procedure In the following you will find a detailed description of the parameters.

Chapter 5.2 describes how to enter parameterization data with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

X27 (RS422/485) interface submodule

Please note the following with reference to the X27 (RS 422/485) interface submodule:

Note

When the X27 (RS 422/485) interface submodule is used, the 3964(R) procedure can only be used in four-wire mode.

Protocol

The following table describes the 3964(R) protocol.

Table 2-2 3964(R) Protocol

Parameter	Description	Default Value
3964 with default values and no block check	• The protocol parameters are set to default values. • If the CP 441 recognizes the string DLE ETX, it stops receiving and sends a DLE to the communication partner if the block was received undamaged, or an NAK if it was damaged.	**3964R with standard values with block check**: char. delay time = 220 ms NAK = 2000 ms Setup attempts = 6 Transmission attempts = 6
3964R with default values and block check	• The protocol parameters are set to default values. • If the CP 441 recognizes the string DLE ETX BCC, it stops receiving and compares the received block check character with the longitudinal parity calculated internally. If the BCC is correct and no other receive errors have occurred, the CP 441 sends the code DLE to the communication partner. (In the event of an error, the NAK code is sent).	
3964 programmable without block check	• The protocol parameters are programmable. • If the CP 441 recognizes the string DLE ETX, it stops receiving and sends a DLE to the communication partner if the block was received undamaged, or an NAK if it was damaged.	
3964R programmable with block check	• The protocol parameters are programmable. • If the CP 441 recognizes the string DLE ETX BCC, it stops receiving and compares the received block check character with the longitudinal parity calculated internally. If the BCC is correct and no other receive errors have occurred, the CP 441 sends the code DLE to the communication partner. (In the event of an error, the NAK code is sent).	

Protocol Parameters

You can only set the protocol parameters if you have not set the default values in the protocol.

Table 2-3 Protocol Parameters (3964(R) Procedure)

Parameter	Description	Value Range	Default Value
Character delay time	The character delay time defines the maximum amount of time permitted between two incoming characters within a message frame.	20 ms to 655350 ms in 10 ms increments The shortest character delay time depends on the baud rate 300 bits/s 60 ms 600 bits/s 40 ms 1200 bits/s 30 ms 2400 to 115200 bits/s 20 ms	220 ms
Acknowledgment delay time (ADT)	The acknowledgment delay time defines the maximum amount of time permitted for the partner's acknowledgment to arrive during connection setup (time between STX and partner's DLE acknowledgment) or release (time between DLE ETX and partner's DLE acknowledgment).	20 ms to 655350 ms in 10 ms increments The shortest acknowledgment delay time (ADT) depends on the baud rate: 300 bits/s 60 ms 600 bits/s 40 ms 1200 bits/s 30 ms 2400 to 115200 bits/s 20 ms	2000 ms (550 ms with 3964 and no block check)
Setup attempts	This parameter defines the maximum number of attempts the CP 441 is allowed in order to establish a connection.	1 to 255	6
Transmission attempts	This parameter defines the maximum number of attempts permitted to transfer a message frame (including the first one) in the event of an error.	1 to 255	6

Baud Rate/Character Frame

The following table describes the baud rate/character frame.

Table 2-4 Baud Rate / Character Frame (3964(R) Procedure)

Parameter	Description	Value Range	Default Value
Baud rate	Speed of data transmission in bits/s **Note:** The maximum baud rate for the CP 441-1 is 38400 baud, and the total baud rate of the CP 441-2 is 115200 baud. This means that the combined baud rates of both interface submodules must not exceed 115200 baud. The maximum baud rate for the 20 mA TTY interface submodule is 19200.	• 300 • 600 • 1200 • 2400 • 4800 • 9600 • 19200 • 38400 • 57600 • 76800 • 115200	• 9600
Start bit	During transmission, a start bit is prefixed to each character to be sent.	• 1 (fixed value)	• 1
Data bits	Number of bits to which a character is mapped.	• 7 • 8	• 8
Stop bits	During transmission, a stop bit is appended to every character to be sent to signal the end of the character.	• 1 • 2	• 1
Parity	A sequence of information bits can be extended to include another bit, the parity bit. The addition of its value (0 or 1) brings the value of all the bits up to a defined status. Thus the data integrity is enhanced. A parity of ”none” means that no parity bit is transmitted.	• none • odd • even	• even
Priority	A partner has high priority if its send request takes precedence over the send request of the other partner. A partner has low priority if its send request must wait until the send request of the other partner has been dealt with. With the 3964(R) procedure, you **must** parameterize both communication partners with different priorities, i.e. one partner is assigned high priority, the other low.	• high • low	• low

Receive Buffer on CP

The table below describes the parameters for the CP receive buffer.

Table 2-5 Receive Buffer on CP (3964(R) procedure)

Parameter	Description	Value Range	Default Value
Delete CP receive buffer during start-up	The CP receive buffer of the CP 441 is not deleted at CPU start-up (STOP → RUN transition).	• yes (fixed)	• no
Use CPU receive mailbox	Here you can specify whether a receive mailbox is to be set up on the CPU. You **must** set up a receive mailbox if you have not programmed a BRCV system function block for the CP 441 in the user program of the CPU. If you have programmed a BRCV, you must deactivate this parameter, otherwise data will be stored in the receive mailbox defined here instead of being processed by the BRCV.	• yes • no	• no
DB number[1]	Number of the data block for the receive mailbox on the CPU.	1 to 65535 (depending on the CPU)	1

1 Only when “use receive mailbox on CPU” = “yes”.

X27 (RS 422) Interface

The table below contains descriptions of the parameters for the X27 (RS 422) interface. RS 485 operation is not possible in conjunction with the 3964(R) procedure.

Table 2-6 X27 (RS 422) Interface (3964(R) Procedure)

Parameter	Description	Value Range	Default Value
Initial state of the receive line	none: This setting only makes sense with bus-capable special drivers.	none	R(A)5V/R(B)0V
	R(A)5V/R(B)0V: Break detection is possible with this initial state.	R(A)5V/R(B)0V	
	R(A)0V/R(B)5V: Break detection is not possible with this initial state. (See also Figure 2-26)	R(A)0V/R(B)5V	

Initial state of the receive line

Figure 2-26 shows the wiring of the receiver at the X27 (RS 422) interface:

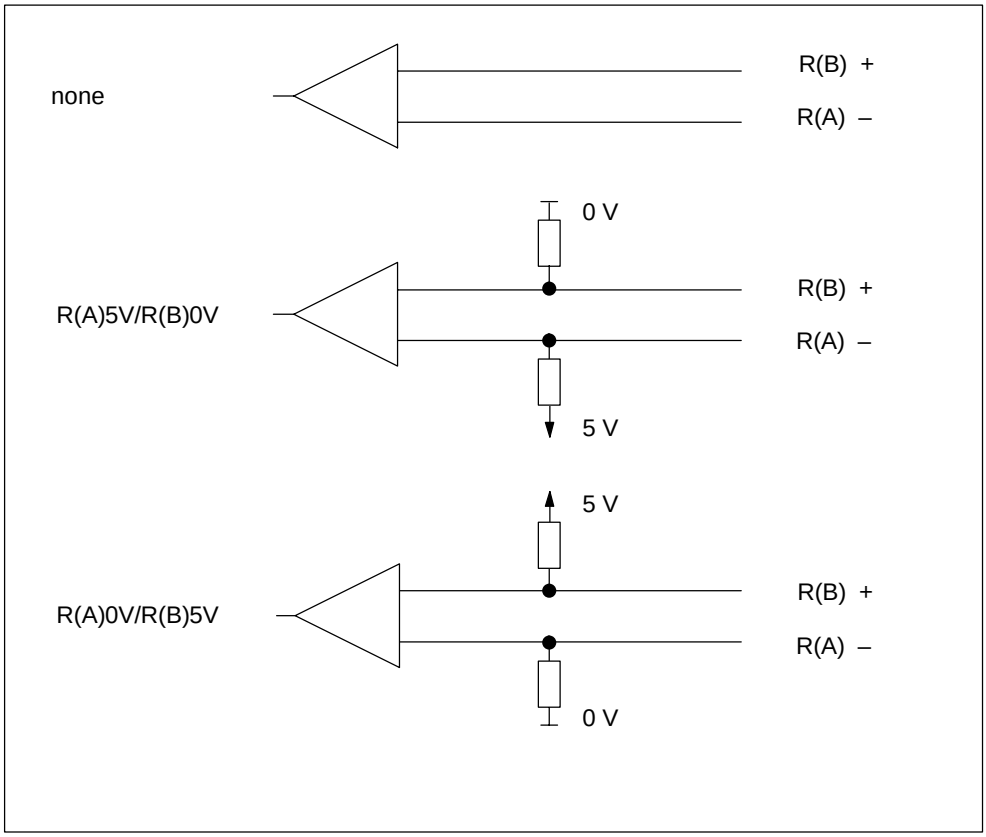


Figure 2-26 Wiring of the Recipient at the X27 (RS 422) Interface (3964(R) Driver)

2.3.2 Parameterization Data of the RK 512 Computer Connection

You can use the parameterization data of the RK 512 computer connection to adjust the CP 441 to suit the properties of the communication partner.

Parameterization Data of the RK 512 Computer Connection

The parameters are identical to those of the 3964(R) procedure because the 3964(R) procedure is a subset of the RK 512 computer connection in the ISO 7-layer reference model (see Section 2.3).

Note

Exception: The number of data bits per character is set permanently to **8** with the RK 512 computer connection.

The parameters of the transport layer (layer 4) must be specified in the system function blocks (SFB) used.

2.3.3 Parameterization Data of the ASCII Driver

Using the parameterization data of the ASCII driver, you can adjust the CP 441 to suit the properties of the communication partner.

Parameterization Data of the ASCII Driver

Using the *CP 441: Point-to-Point Communication, Parameter Assignment* interface you specify the parameters for the physical layer (layer 1) of the ASCII driver. In the following you will find a detailed description of the parameters.

Section 5.2 describes how to enter parameterization data with the parameterization interface *CP 441: Point-to-Point Communication, Parameter Assignment*.

X27 (RS422/485) interface submodule

Please note the following with reference to the X27 (RS 422/485) interface submodule:

Note

When the X27 (RS 422/485) interface submodule is used, the ASCII driver can be used in four-wire mode (RS 422) and two-wire mode (RS 485).

At parameterization, you specify the type of interface (RS 422 or RS 485).

Protocol Parameters

The table below describes the protocol parameters.

Table 2-7 Protocol Parameters (ASCII Driver)

Parameter	Description	Value Range	Default Value
Indicator for end of receive message frame	Defines which criterion signals the end of each message frame.	• On expiry of character delay time • On receipt of end-of-text character • On receipt of fixed number of characters	• On expiry of character delay time
Character delay time	The character delay time defines the maximum permitted time between 2 consecutively received characters.	2 to 65535 ms The shortest character delay time depends on the baud rate baud / Character delay time 300 130 600 65 1200 32 2400 16 4800 8 9600 4 19200 2 38400 2 57600 2 76800 2 115200 2	• 4 ms
End-of-text character 1[1]	First end code.	• 7 data bits[2]: 0 to 7FH (hex) • 8 data bits[2]: 0 to FFH (hex)	• 3 (03H = ETX)
End-of-text character 2[1]	Second end code, if specified.	• 7 data bits[2]: 0 to 7FH (hex) • 8 data bits[2]: 0 to FFH (hex)	• 0
Message frame length when received[3]	When the end criterion is ”fixed message frame length”, the number of bytes making up a message frame is defined.	1 to 1024 (bytes)	• 240

1 Can only be set if the end criterion is an end-of-text character.
2 Depending on the parameterization of the character frame (7 or 8 data bits) (see Table 2-8).
3 Can only be set if the end criterion is a fixed message frame length.

Baud Rate/Character Frame

The table below describes the parameters for the baud rate and character frame.

Table 2-8 Baud Rate / Character Frame (ASCII Driver)

Parameter	Description	Value Range	Default Value
Baud rate	Speed of data transmission in bits/s **Note:** The maximum baud rate of the CP 441-1 is 38400. The total baud rate of the CP 441-2 is 115200 baud. This means that the combined baud rates of both interface submodules must not exceed 115200 baud. The maximum baud rate for the 20 mA TTY interface submodule is 19200.	• 300 • 600 • 1200 • 2400 • 4800 • 9600 • 19200 • 38400 • 57600 • 76800 • 115200	• 9600
Start bit	During transmission, a start bit is prefixed to each character to be sent.	• 1 (fixed value)	• 1
Data bits	Number of bits to which a character is mapped.	• 7 • 8	• 8
Stop bits	During transmission, a stop bit is appended to every character to be sent to signal the end of the character.	• 1 • 2	• 1
Parity	A sequence of information bits can be extended to include another bit, the parity bit. The addition of its value (0 or 1) brings the value of all the bits up to a defined status. Thus the data integrity is enhanced. A parity of ”none” means that no parity bit is transmitted.	• none • odd • even	• even

Data Flow Control

In the following table the parameters for data flow control are described.

Data flow control is not possible with the RS 485 interface. Data flow control with ”RTS/CTS” and ”Automatic Use of V24 Signals” is only possible when the RS 232C interface submodule is used (see also Table 1-2).

Table 2-9 Data Flow Control (ASCII Driver)

Parameter	Description	Value Range	Default Value
Data flow control	Defines which type of data flow control is used.	• None • XON/XOFF • RTS/CTS • Automatic Use of V24 Signals	• None
XON character[1]	Code for XON character	• 7 data bits[2]: 0 to 7FH (hex) • 8 data bits[2]: 0 to FFH (hex)	• 11 (DC1)
XOFF character[1]	Code for XOFF character	• 7 data bits[2]: 0 to 7FH (hex) • 8 data bits[2]: 0 to FFH (hex)	• 13 (DC3)
Waiting for XON after XOFF (wait time for CTS=ON)[3]	Period of time for which the CP 441 shall wait for the XON code or for CTS=“ON” of the communication partner when sending.	20 to 655350 ms in 10 ms increments	• 20000 ms
Time to RTS OFF[4]	Time to elapse after the transmission before the CP 441 sets the RTS line to OFF.	0 to 655350 ms in 10 ms steps	• 10 ms
Data output waiting time[4]	Time that the CP 441 is to wait for the communication partner to set CTS to ON after setting the RTS line to ON and before starting the transmission.	0 to 655350 ms in 10 ms steps	• 10 ms

1 Only in the case of XON/XOFF data flow control.
2 Depending on the parameterization of the character frame (7 or 8 data bits) (see Table 2-8).
3 Only in the case of XON/XOFF or CTS/RTS flow control.
4 Only in the case of automatic use of the RS 232C secondary signals.

Further Information

You will find more information on data flow control with XON/XOFF and RTS/CTS and on automatic use of the RS 232C secondary signals in Section 2.2.4 as of the subsection entitled ”RS 232C Secondary Signals”.

Receive Buffer on CP

The table below describes the parameters for the CP receive buffer.

Table 2-10 Receive Buffer on CP (ASCII Driver)

Parameter	Description	Value Range	Default Value
Delete CP receive buffer during start-up	The CP receive buffer of the CP 441 is not deleted at CPU start-up (STOP → RUN transition).	• yes (fixed)	• no
Buffered receive message frames	Here you can specify the number of receive message frames to be buffered in the CP receive buffer. If you specify ”1” here and deactivate the following parameter ”prevent overwrite” **and** cyclically read the received data from the user program, a current message frame will always be sent to the CPU.	1 to 250	250
Prevent overwrite	You can deactivate this parameter if the parameter ”buffered receive message frames” is set to ”1”. This authorizes the buffered receive message frame to be overwritten.	• yes • no (only when ”buffered receive message frames”=”1”)	• yes
Use CPU receive mailbox	Here you can specify whether a receive mailbox is to be set up on the CPU. You **must** set up a receive mailbox if you have not programmed a BRCV system function block for the CP 441 in the user program of the CPU. If you have programmed a BRCV, you must deactivate this parameter, otherwise data will be stored in the receive mailbox defined here instead of being processed by the BRCV.	• yes • no	• no
DB number[1]	Number of the data block for the receive mailbox on the CPU.	1 to 65535 (depending on the CPU)	1

[1] Only in the case of ”Use CPU Receive Mailbox” = ”Yes”

Further Information

In Section 2.2.4 you can find further information on handling the receive buffer under “Receive Buffer on CP 441”.

X27 (RS422/485) interface

The table below contains descriptions of the parameters for the X27 (RS 422/485) interface submodule. RS485 operation is not possible in conjunction with the printer.

Table 2-11 X27 (RS 422/485) Interface Submodule (ASCII Driver)

Parameter	Description	Value Range	Default Value
Operating mode	Specifies whether the X27 (RS 422/485) interface is to be run in full-duplex mode (RS 422) or half-duplex mode (RS 485). (See also Section 2.1).	• Full-duplex (RS 422) four-wire mode • Half-duplex (RS 485) two-wire mode	• Full-duplex (RS 422) four-wire mode
Initial state of the receive line	none: This setting only makes sense with bus-capable special drivers. R(A)5V/R(B)0V: break detection is possible with this initial state in conjunction with ”Full Duplex (RS 422) Four-Wire Mode”. R(A)0V/R(B)5V: this initial state corresponds to idle (no senders active) in ”Half Duplex (RS 485) Two-Wire Mode”. Break detection is not possible with this initial state. (See also Figure 2-27)	• None: • R(A)5V/R(B)0V[1] • R(A)0V/R(B)5V	• R(A)5V / R(B)0V[2]

1 Only in the case of ”Full-Duplex (RS 422) Four-Wire Mode”

2 Only in the case of ”Full-Duplex (RS 422) Four-Wire Mode” in the case of ”Half Duplex (RS 485) Two-Wire Mode”, the default setting is R(A)0V/R(B)5V

Initial state of the receive line

Figure 2-27 shows the wiring of the recipient at the X27 (RS 422/ 485) interface:

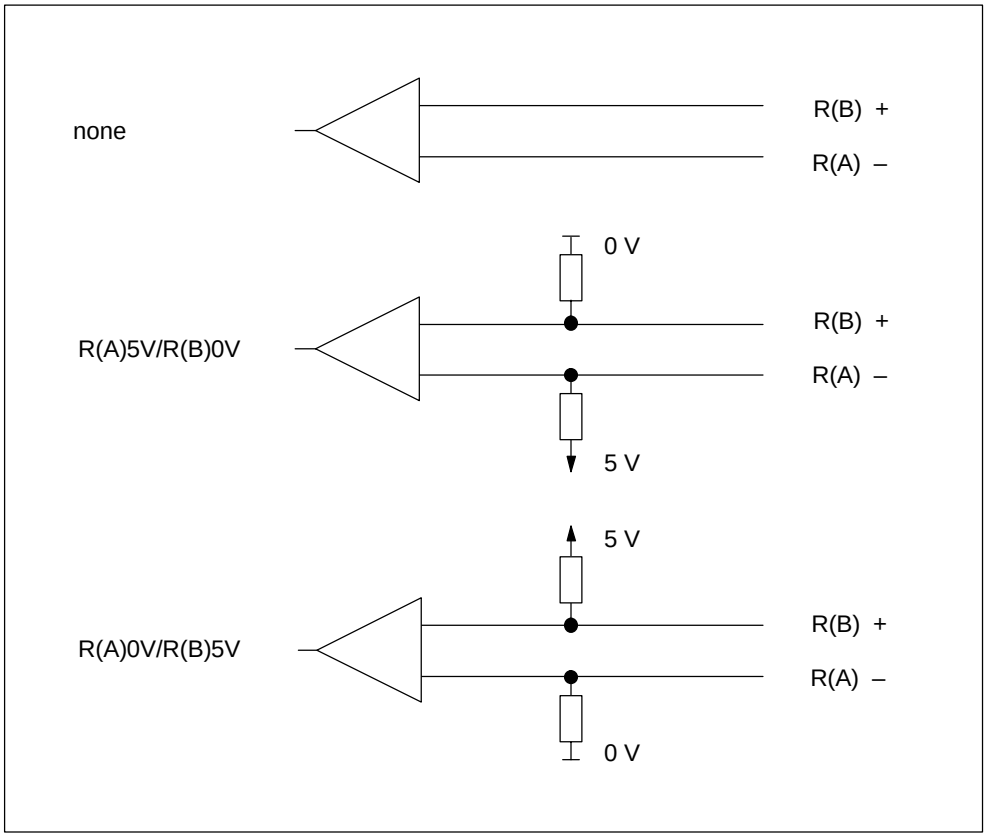


Figure 2-27 Wiring of the Recipient at the X27 (RS 422/485) Interface

2.3.4 Parameterization Data of the Printer Driver

You can use the parameterization data of the printer driver to configure the transmission-specific parameters and the message texts for printer output.

Parameterization Data of the Printer Driver

You set these parameters with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface:

- The parameters for the physical layer (layer 1) of the printer driver
- The message texts for printer output
- The page layout, character set and control characters for the message texts

In the following you will find a detailed description of the parameters. Section 5.2 describes how to enter parameterization data with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

Please Note

Please note the following:

- The size of the message text buffer depends on the module
  - 8 KB per interface for the CP 441, 6ES7 441-_AA00-0AE0
  - 8 KB per interface for the CP 441, 6ES7 441-_AA01-0AE0
  - 55 KB per interface for the CP 441, as of 6ES7 441-_AA02-0AE0
- The message texts are stored in the CPU's load memory together with the parameterization data and loaded automatically onto the CP 441 during the loading operation. You must therefore reserve the corresponding memory space in the load memory of the CPU for every interface for which you have created message texts.
- Before you transfer the message texts to the CP 441 you must increase the value of the parameter for transferring parameters to modules for the relevant CPU. You should plan in approximately 20 s per interface.

Baud Rate/Character Frame

The table below describes the parameters for the baud rate and character frame.

Table 2-12 Baud Rate/Character Frame (Printer Driver)

Parameter	Description	Value Range	Default Value
Baud rate	Speed of data transmission in bits/s **Note:** The maximum baud rate for the CP 441-1 is 38400 baud, and the total baud rate of the CP 441-2 is 115200 baud. This means that the combined baud rates of both interface submodules must not exceed 115200 baud. The maximum baud rate for the 20 mA TTY interface submodule is 19200.	• 300 • 600 • 1200 • 2400 • 4800 • 9600 • 19200 • 38400 • 57600 • 76800 • 115200	• 9600
Start bit	During transmission, a start bit is prefixed to each character to be sent.	• 1 (fixed value)	• 1
Data bits	Number of bits to which a character is mapped.	• 7 • 8	• 8
Stop bits	During transmission, a stop bit is appended to every character to be sent to signal the end of the character.	• 1 • 2	• 1
Parity	A sequence of information bits can be extended to include another bit, the parity bit. The addition of its value (0 or 1) brings the value of all the bits up to a defined status. Thus the data integrity is enhanced. A parity of ”none” means that no parity bit is sent.	• none • odd • even	• even

X27 (RS 422) Interface

The table below contains descriptions of the parameters for the X27 (RS 422) interface submodule. RS485 operation is not possible in conjunction with the printer.

Table 2-13 X27 (RS 422) Interface Submodule (Printer)

Parameter	Description	Value Range	Default Value
Initial state of the receive line	none: This setting only makes sense with bus-capable special drivers. R(A)5V/R(B)0V: Break detection is possible with this initial state. R(A)0V/R(B)5V: Break detection is not possible with this initial state.	• None: • R(A)5V/R(B)0V • R(A)0V/R(B)5V	• R(A)5V/ R(B)0V

Initial state of the receive line

Figure 2-28 shows the wiring of the receiver at the X27 (RS 422) interface:

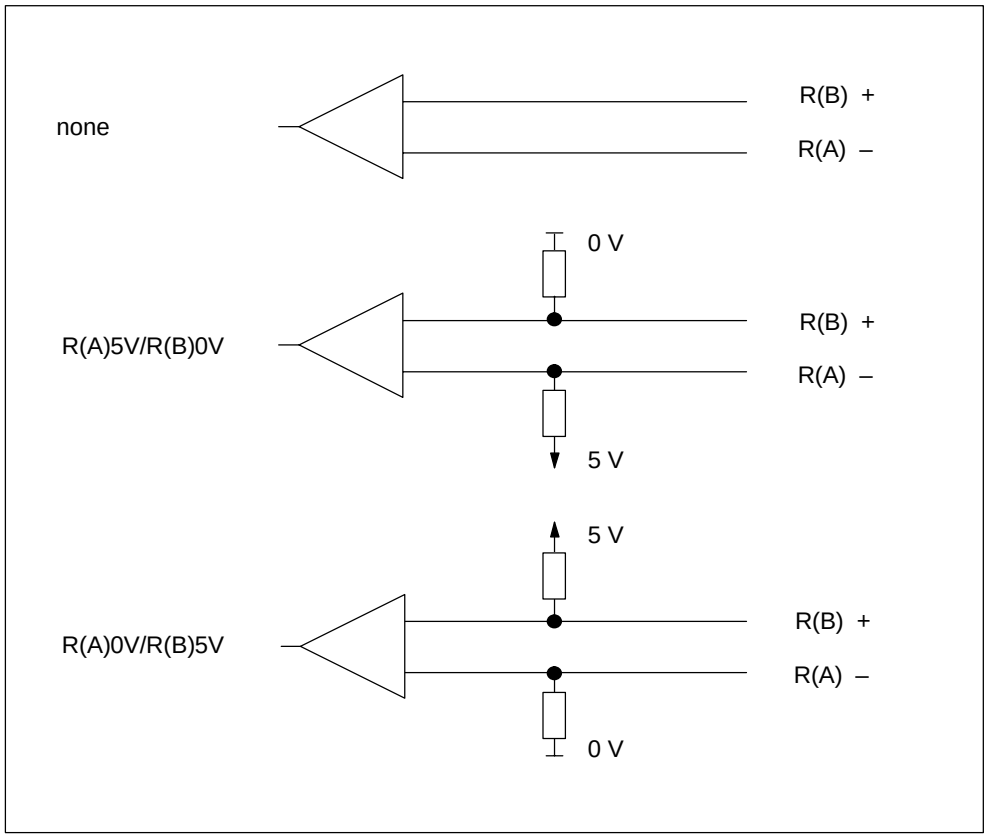


Figure 2-28 Wiring of the Recipient at the X27 (RS 422) Interface

Data Flow Control/Handshaking

Handshaking controls the data flow between two communication partners. Handshaking ensures that data is not lost in transmissions between devices that work at different speeds.

You can also send message texts with data flow control during printer output. There are essentially two types of handshaking:

- Software handshaking (e.g. XON/XOFF)
- Hardware handshaking (e.g. RTS/CTS)

Data flow control is implemented as follows on the CP 441 during printer output:

- As soon as the CP 441 is switched by parameterization to the operating mode with flow control, it sends the XON character or sets the RTS line to ON.
- If the CP 441 receives the XOFF character, or the CTS control signal is set to OFF, the CP 441 interrupts the output of characters. If neither an XON character is received nor CTS is set to ON before a parameterized time has elapsed, the printer output is aborted and an appropriate error message (0708H) is entered in the SYSTAT error-signaling area of the CP 441.

Note

RTS/CTS data flow control is only possible when the RS 232C interface submodule is used. To do this, you must fully wire the interface signals in the plug-in connection (see Appendix B.1).

BUSY Signal

The CP 441 evaluates the printer's "BUSY" control signal. The printer indicates to the CP 441 that it is ready to receive:

- CP 441 with 20-mA TTY interface submodule: current on RxD line
- CP 441 with RS 232C and X27 (RS 422/485) interface submodule: with the CTS = "ON" signal

Note

When you parameterize with RTS/CTS flow control, you must set the polarity of the BUSY signal on the printer as follows:

- BUSY signal: CTS = ”OFF”

Please note that some printers use the DTR signal to display the BUSY signal. In such cases you must wire the connecting cable to the CP 441 appropriately.

In the following table the parameters for data flow control are described.

Data flow control is not possible with the RS 485 interface. RTS/CTS data flow control is only possible when the RS 232C interface submodule is used (see also Table 1-2).

Table 2-14 Data Flow Control (Printer Driver)

Parameter	Description	Value Range	Default Value
Data flow control	Defines which type of data flow control is used.	• None • XON/XOFF • RTS/CTS	• None
XON character[1]	Code for XON character	• 7 data bits[2]: 0 to 7FH (hex) • 8 data bits[2]: 0 to FFH (hex)	• 11 (DC1)
XOFF character[1]	Code for XOFF character	• 7 data bits[2]: 0 to 7FH (hex) • 8 data bits[2]: 0 to FFH (hex)	• 13 (DC3)
Waiting for XON after XOFF (wait time for CTS=ON)[3]	Period of time for which the CP 441 shall wait for the XON code or for CTS=“ON” of the communication partner when sending.	20 to 655350 ms in 10 ms increments	• 2000 ms

1 Only in the case of XON/XOFF data flow control.
2 Depending on the parameterization of the character frame (7 or 8 data bits) (see Table 2-12).
3 Only in the case of XON/XOFF or CTS/RTS flow control.

Page Layout

The table below contains descriptions of the parameters for the page layout.

Table 2-15 Page Layout (Printer Driver)

Parameter	Description	Value Range	Default Value
Left margin (number of characters)	Number of spaces to precede each line in the body of the text, header or footer. You yourself must ensure that a line is not too long for the printer.	0 to 255	• 3
Lines per page (with header and footer)	Number of lines to be printed on each page. The number of printed lines is determined on the basis of the separators output, which means that all header and footer lines must be included in the count.	• 1 to 255 • 0 (continuous printing)	• 50
Separators/line end	Characters used to conclude each line in the body of the text, header or footer. The defined separator must be contained in the text, header and footer to be output. If the header does not contain a separator, the text begins right at the top of the page.	• CR (carriage return) • LF (line feed) • CR LF (carriage return andline feed) • LF CR (line feed andcarriage return)	• CR LF (carriage return and line feed)
Header lines	Text for up to max. 2 header and footer lines; a header or footer line is output when the entry field in the parameterization software contains a text or at least a blank. If a text is specified only for the 2nd header or footer line, the 1st header or footer line is automatically padded with a blank and printed. A blank line is output before and after the headers/footers.	• ASCII characters (text) • %P (conversion statement for outputting a page number) (max. 60 characters)	-
Footer lines			

Character Set

The table below contains descriptions of the parameters for the character set.

Table 2-16 Character Set (Printer Driver)

Parameter	Description	Value Range	Default Value
Printer character set	If you set ”IBM”, the system character set that is set in Windows is used (conversion of the ANSI character set to the printer character set). If you set ”User-Defined”, you can adapt the character set to include special characters for a particular language.	• IBM • User-Defined	• IBM

Control Characters

The table below contains a description of the parameters for control characters.

Table 2-17 Control Characters (Printer Driver)

Parameter	Description	Value Range	Default Value
Printer emulation	Sets the printer emulation (printer commands for the following control characters: bold, condensed, expanded and italic type and underlining). If you set ”User-Defined”, you can modify the printer emulation and include additional control characters of the printer. The characters A to Z and a to z are permissible as control characters.	• HP DeskJet • HP LaserJet • IBM Proprinter • User-Defined	• HP DeskJet

Message texts

You can configure message texts with variables and control statements (e.g. for bold, condensed, expanded or italic type and underlining). Each message text is assigned a number during configuration. You print out a specific message text by specifying a reference (to the memory cell containing the message text number) for send parameters SD_1 to SD_4 of the PRINT system function block.

Features

Conditions when configuring message texts:

- Max. size of the text SDB: 55 kbytes
- Max. length of a message text without variables: 150 characters
- Max. length of a message text with variables displayed: 4000 characters
- Max. number of variables per message text: 4 (3 + message-text number)

Variables

Up to 4 variables (3 + a message text number) can be displayed in a message text. The values which can be inserted as variables are: values calculated by the user program (e.g. levels), date and time, strings (string variables) or other message texts. The variables are parameterized as the send parameters SD_1 to SD_4 of the PRINT system function block.

A conversion statement must be specified in the configured message text or in the format string for each variable, and the meaning and output format of the variable value must be encoded in this statement.

Format String

The format string allows you to define the format and composition of a message text. The format string can consist of:

- Text (All printable characters are permitted, e.g.: The level ... l was reached at ... hours.).
- Conversion statements for variables (e.g. %N = expression of a message text stored on the CP CP 441; the desired message text number is parameterized by means of the reference (ANYPOINTER addressed to the memory cell in which the message text number is stored) in the send variables SD_1 to SD_4).

  For each variable there must be one conversion statement in the format string. The conversion statements are applied to the variables in the sequence in which they occur in the format string.
- Control statements with control characters for bold, condensed or italic type and underlining (e.g. \B = bold type on) or with additional control characters you have defined.

  You can use other control characters supported by your printer if you enter them in the control characters table of the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface and reparameterize the CP 441.

You will find all the conversion and control statements that are possible in the format string and in configured message texts in Section 2.3.5.

Please note that a line feed is carried out by default after each output.

The table below describes the parameters for configuring message texts (with the *CP 441: Point-to-Point Communication, Parameter Assignment*) parameterization interface).

Table 2-18 Message Texts (Printer Driver)

Parameter	Description	Value Range	Default Value
Name of text SDB/text file	The message texts for a CP 441 (serial interface) must be stored in a text SDB for parameterization. You can also store configured message texts in an external text file.	ASCII characters (max. 8 characters)	-
Version number	Version number of the text SDB/text file	0.1 to 99.9	-
Message texts	All the message texts in the text block are displayed here, complete with their message-text numbers; you can select a message-text line and modify it in the ”Edit message” parameter.	ASCII characters (unchangeable)	-

Table 2-18 Message Texts (Printer Driver), continued

Parameter	Description	Value Range	Default Value
Edit message	You can transfer message texts edited here to the ”Message Texts” list by clicking the ”Enter” button.	Message text number: 0 to 999 Message text (max. 150 characters): • ASCII characters (text) • Conversion statements (for variables) • Control characters (all those defined in the control character table)	-
Type style	You can easily assign control characters to text selected in the ”Edit Message” entry box by using buttons B to U.	• B (bold type) • C (condensed type) • E (expanded type) • I (italic type) • U (underlining)	-

Further Information

Section 2.3.5 contains detailed descriptions of the conversion statements and control characters that are possible in message texts and explains their purpose.

Examples

Here are some examples of message texts. The variables (SD_1, SD_2) in the examples must be parameterized at the PRINT SFB.

Example 1: The level ”200” l was reached at ”17.30” hours.

Format string	=	The level %i l was reached at %Z hours.
Variable (SD_1)	=	Time
Variable (SD_2)	=	Level

Example 2: The pressure in the chamber ”is falling”

Format string	=	%N %S
Variable (SD_1)	=	Reference to memory cell containing ”17” (text no. 17: The pressure in the chamber ...)
Variable (SD_2)	=	Reference to string (string variable: ... is falling)

The reference to the string is a symbolic address that specifies where the string is stored (DB).

Example 3: (Set page number to 10)

Format string	=	%P	
Variable (SD_1)	=	10	(page number: 10)

2.3.5 Conversion and Control Statements for Printer Output

The output of a message text with variables and control statements (e.g. for bold, condensed, expanded or italic type and underlining) is defined by means of a format string.

In the format string you can also define statements to execute other useful functions for printer output (e.g. to set a page number or begin a new page).

All the permissible characters and representation modes for the format string are described below. You can also configure all the control statements and conversion statements for variables (except for %P ”set page number”) in the message texts using the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

Format String

Figure 2-29 illustrates the structure of the format string schematically.

A format string can contain normal text and/or conversion statements for variables and/or control statements. Normal text, conversion statements and control statements can occur in any sequence in the format string.

There must be a conversion statement (and only one) for each variable in the format string or message text. The conversion statements are applied to the variables in the sequence in which they occur.

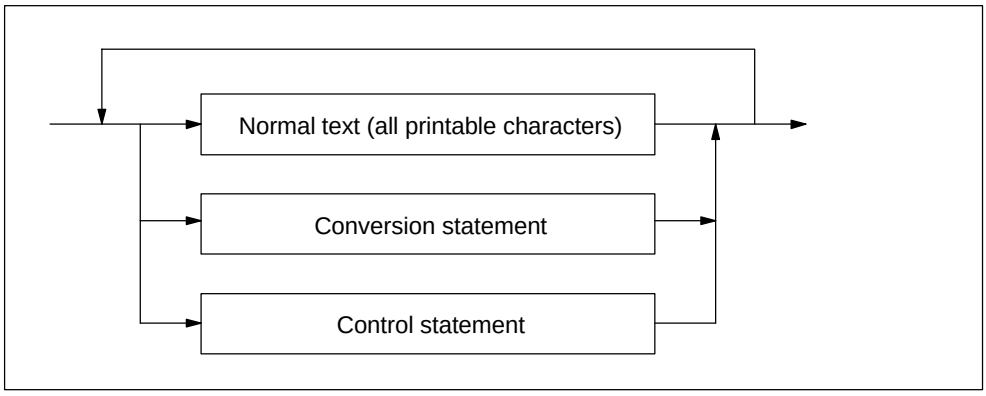


Figure 2-29 Schematic Structure of the Format String

Permissible Characters for Text

The following can be specified as normal text:

- All printable characters
- All characters preceded by $ at the language interface (ICE 1131-3). The language compilers convert these characters to the corresponding hex code.
  Exception: The character $N is not permissible.

  Example: Carriage return ODH = $R in the format string

Conversion Statement

Figure 2-30 illustrates the structure of a conversion statement schematically.

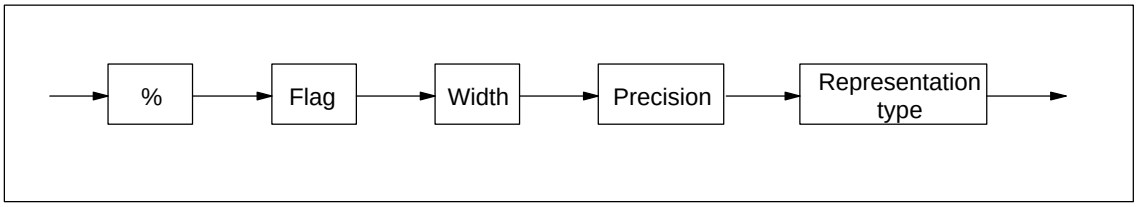


Figure 2-30 Schematic Structure of a Conversion Statement

Flag

Without	=	Right-justified output
-	=	Left-justified output

Width

Without	=	Standard output representation (see Table 2-19)
n	=	Exactly n characters are output (up to a maximum of 255 characters); blanks may be added before (right-justified output) or after (left-justified output)

Precision

Precision is only relevant to representation types A, D, F and R. In all other cases, it is ignored.

Without = Standard output representation (see Table 2-19)

.0 = No output of decimal point and places after the decimal point in the Real (R) and Floating point (F) representation types.

.n = Output of decimal point and n (1 to 99) significant places after the decimal point in the Real (R) and Floating point (F) representation types. In the case of dates (= representation types A and D), precision relates to the number of digits used for the year. Only 2 and 4 are permitted for dates.

Please note that the precision is always preceded by a period. The period serves to identify it and separate it from the width.

Representation Type

The following table describes the possible representation types of the variable values. Representation types N and P are exceptions. They are explained below the table.

Both upper- and lower-case characters are permitted for the representation type.

Table 2-19 Representation Types in the Conversion Statement

Repre-senta-tion type	Associated Data Type	Standard Repres.	Width of the Standard Repres.	Description
A	DATE, WORD	10.06.1992 (German)	10	German date format
C	CHAR, BYTE WORD DWORD ARRAY OF CHAR ARRAY OF BYTE	A B QB ABCD ABCDE ...	1 1 2 4 -	Alphanumeric characters
D	DATE, WORD	1996-06-10 (American)	10	ICE data format 1131-3
F	REAL, DWORD	0.123456	8	Floating point, without exponent
H	All data types incl. ARRAY OF BYTE	In accordance with the data type	In accordance with the data type	Hexadecimal format
I	INT, WORD DINT, DWORD	-32767 -2147483647	Max. 6 Max. 11	Integer range
N[1]	WORD (text number)	Message text output	-	Integer 0 to 999
P[2]	INT, WORD	Page number	5	-
R	REAL, DWORD	0.12E-04	8	Floating point, without exponent

Table 2-19 Representation Types in the Conversion Statement, continued

Repre-senta-tion type	Associated Data Type	Standard Repres.	Width of the Standard Repres.	Description
S	STRING	Text output	-	Text strings
T[1]	TIME, DWORD	2d_3h_10m_5s_250 ms	Max. 21	Duration
U	BYTE WORD DWORD	255 65535 4294967295	Max. 3 Max. 5 Max. 10	Integer range without plus/minus sign
X	BOOL BYTE WORD DWORD	1 11101100 11001... (16) 11001... (32)	1 8 16 32	Binary representation
Y[3]	DATE_AND_TIME_ OF_DAY, DT	10.06.1992 - 15:42:59.723	25	Date and time
C	TIME_OF_DAY DWORD	15:42:59.723	12	Time

1 If there is no message text number or system time in these representation types, 6 * characters appear in the printout instead (the CP 441 does not keep the time).

2 The P representation type is only permitted in the format string. P is not permitted in the configured message texts.

3 The current time and date must be read first by means of the ”READ_CLOCK” system function (SFC 1) and stored in the user memory (flag, data).

Output via Message Text Number (%N)

You use the N representation type to start printing message texts stored on the CP 441.

ANYPOINTER is the only data type permissible for the PRINT SFB send variables (SD_1 to SD_4). The variable thus points to the memory cell in which the desired message text number is entered. Please note that the message text number must be specified in the WORD data format.

The flag, width and precision do not affect the printer output in the case of the N representation type. The message text configured beforehand with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface is output in its entirety.

Example: The pressure in the chamber ”is falling”

Format string	=	%N %S
Variable (SD_1)	=	Reference to memory cell containing ”17” (text no. 17: The pressure in the chamber ...)
Variable (SD_2) (string variable: ... is falling)	=	Reference to string

The reference to the string is a symbolic address that specifies where the string is stored (DB).

Setting the Page Number (%P)

You use the P representation type to change the page number in the printout.

The CP 441 always begins a printout on page 1. This conversion statement allows you to set a page number of your choice. The conversion statement variable contains the number to be set.

Example: (Set page number to 10)

Format string	=	%P	
Variable (SD_1)	=	10	(page number: 10)

Note

In the case of the P representation type, there must be no further text, conversion or control statements in the format string.

The P representation type is not permitted in configured message texts.

Notes on the Conversion Statement

Please note the following in relation to conversion statements:

- Whenever a maximum length is specified for the standard representation, the actual output can also be shorter. Example: The output of the integer 10 consists of only 2 characters.
- The length of the data to be printed depends on the length of the variables. For example, in the case of the I representation type a maximum of 6 characters can be output for the INT data type and a maximum of 11 characters for the DINT data type.
- A width of ”0” is not permissible in conversion statements. This is printed out as ”******” with the valid conversion statement.
- If the specified width is too small, in the case of text-based output (representation types A, C, D, S, T, Y and Z), only the number of characters corresponding to the specified width are output (the output is truncated). In all other cases, * characters are output corresponding to the width.
- Undefined or invalid conversion statements are not executed. This is printed out as ”******” (e.g. representation type missing: %2.2).

  The rest of the conversion statement (e.g. everything after the character identified as incorrect) is output. This allows the exact cause of the error to be determined. If this is not possible, you can use the CP 441’s STATUS system function block to find out the cause of the error (see Chapter 6).
- Conversion statements without associated variables (send variables SD_1 to SD_4 for the PRINT SFB) are ignored. Variables for which there is no conversion statement are not output.
- Conversion statements that are not supported in a header or footer are not executed. Instead, they are forwarded to the printer transparently.
- You have to use control statements to specify formatting (line feed, tabs, etc.) in a message text or in the printer output of a long conversion statement.

Examples of Incorrect Conversion Statements

Here are some examples of incorrect conversion statements:

Example 1: ******.2R

Format string = %303.2R
Variable (SD_1)= 1.2345E6

Error: Invalid width in the R representation type. The maximum permissible value for all representation types is 255.

Example 2: ****

Format string = %4.1I
Variable (SD_1)= 12345 DEC

Error: The selected width was too small for the variable value to be output. The precision is not relevant to representation type I.

Example 3: 96-10-3

Format string = %7.2D
Variable (SD_1)= D#1996-10-31

Error: The format string is formally correct, but the selected width was too small to print the date out fully.

Example 4: **********

Format string = %.3A
Variable (SD_1)= D#1996-10-31

Error: The standard width of representation type A was selected but with invalid precision. The possible values here are 2 and 4.

Example 5: ******

Format string = %3.3
Variable (SD_1)= 12345 HEX

Error: A representation type was not specified.

Examples of Correct Conversion Statements

Here are some examples of correct conversion statements:

Example 1: 31.10.1996

Format string	=	%15.4A
Variable (SD_1)	=	D#1996-10-31

A width of 15 with a precision of 4 (width of the year) and right-justified formatting were selected.

Example 2: 12345.

Format string	=	%-6I
Variable (SD_1)	=	12345 DEC

The selected width was one character greater than the variable value to be output; left-justified formatting.

Example 3: 12d_0h_0m_23s_348ms

Format string	=	%T
Variable (SD_1)	=	T#12D23S348MS

The IEC time is in the standard format; unspecified time units are inserted with zeros.

Example 4: 1.234560E+02

Format string	=	%12.6R
Variable (SD_1)	=	1.23456E+002

A width of 10 is available to display the whole variable, with the precision (number of places after the decimal point) taking up 6 characters.

Example 5: TEST..

Format string	=	%-6C
Variable (SD_1)	=	TEST

Left-justified formatting of the text variable

Control Statements

Control statements are used to achieve specific results in the printout (e.g. underlining).

In addition to the standard control statements (for bold, condensed, expanded or italic type and underlining), you can also use other control characters if you enter them in the control character table of the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface (e.g. K for ”small caps” in Figure 2-31).

Figure 2-31 illustrates the structure of a control statement schematically.

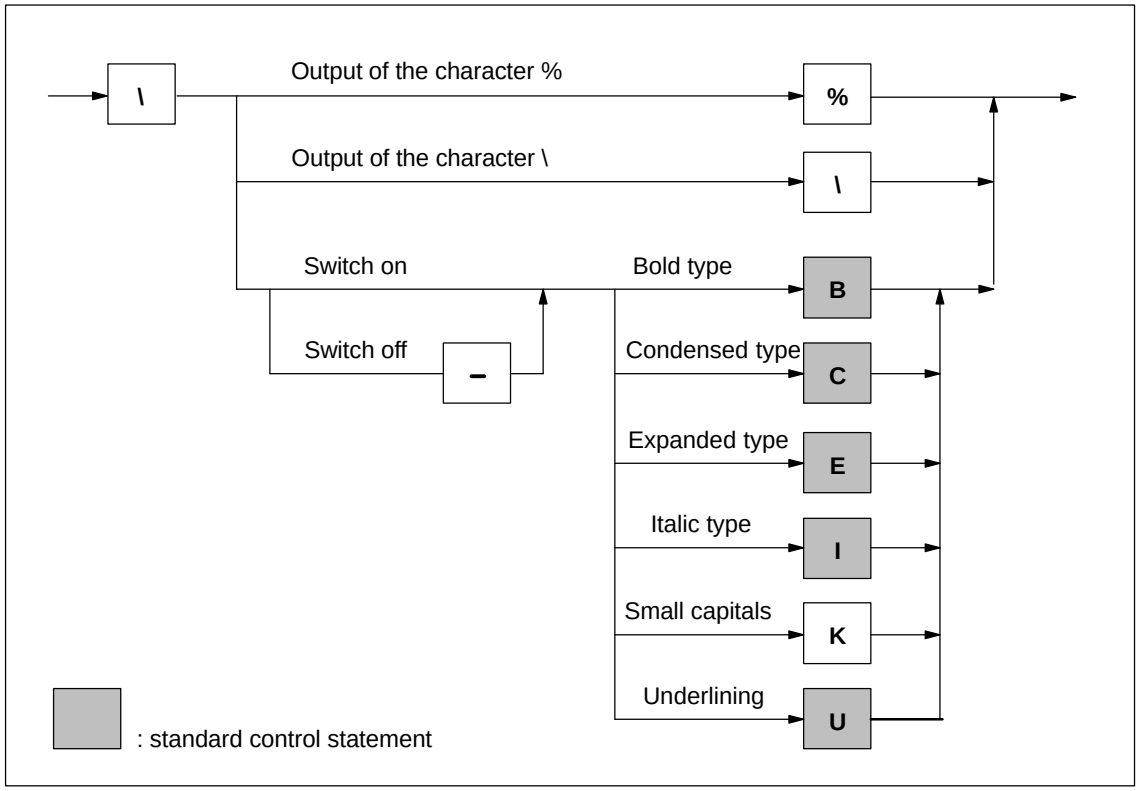


Figure 2-31 Schematic Structure of a Control Statement

Examples

Here are some examples with control statements:

Example 1:

To print the text ”**Bold type** and underlining are ways of highlighting a text”, you have to enter the following:

\BBold type\-B and \Uunderlining\-U are ways of highlighting a text

Example 2:

To output the format string with the conversion statement ”Message text no. %i of %8.2A” transparently on the printer, you have to enter the following:

’Message text no. \%i of \%8.2A’

Beginning a New Page (\F)

Given the parameterized page layout, i.e. the configured headers and footers and the number of lines per page, the \F control statement can be used to begin a new page. This differs from a pure form feed on the printer.

Example: (Beginning a new page)

Format string	=	\F

Note

In the case of the \F control statement, there must be no further text, conversion or control statements in the format string. The variables remain unassigned.

Printing Without a Line Break (\x)

The CP 441 normally appends the parameterized end-of-line character (CR, LF, CR LF, LF CR) when it sends a message text. The \x control statement cancels the line break after a message text. This means that you can print several messages in a single line in order, for example, to display more variables in a line. The \x control statement is appended at the end of the format string.

Example: The level ”200” l was reached at ”17.30” hours. ...

Format string	=	The level %i l was reached at %Z hours.\x
Variable SD_1	=	Time
Variable SD_2	=	Level

Note

Please note that when you use the \x control statement, the new line always begins without a left margin.

Notes on Control Statements

Please note the following in relation to control statements:

- If the deactivation of an effect is specified without it previously having been activated, or if the output device is incapable of producing the effect, the control statement is ignored.
- The % and \ characters required to define the format string can be printed by means of the control statement.
- Undefined or incorrect control statements are not executed.

Starting up the CP 441

3

Before starting up the CP 441 you will need to perform the following operations in the order given.

1. Mounting the CP 441
2. Configuring the CP 441
3. Parameterizing the CP 441
4. Configuring the connections for the CP 441
5. Creating a user program for the CP 441

Mounting the CP 441

Mounting the CP 441 involves inserting it into the mounting rack of your programmable controller and plugging in the interface submodules.

For a detailed description, see Chapter 4.

Configuring the CP 441

Configuring the CP 441 involves entering it in the configuration table. The CP 441 is configured using the STEP 7 software.

For a detailed description, see Section 5.1.

Parameterizing the CP 441

Parameterizing the CP 441 involves creating the specific parameters of the protocols and configuring message texts for printer output. You parameterize the CP 441 with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

For a detailed description, see Section 5.2.

Storing the parameterization data of the CP 441 involves saving the parameters, loading them in the CPU and transferring them to the CP 441. You use the STEP 7 to store the parameterization data.

For a detailed description, see Section 5.4.

Configuring the connections for the CP 441

Configuring the connections for the CP 441 means connecting the communication end points within a point-to-point network in the project of your programmable controller. Connections are configured using the STEP 7 software (connection configuration table).

For a detailed description, see Section 5.3.

Creating a user program for the CP 441

Programming the CP 441 involves configuring it for the associated CPU via the STEP 7 user program. The CP 441 is programmed using the language editors of the STEP 7 software.

For a detailed description, see chapter 6 of this manual and see the STEP 7 manual /1/. Chapter 9 contains a complete programming example.

/1/ *Programming with STEP 7 V5.1 Manual*

Mounting the CP 441

4

In Section	You Will Find	on Page
4.1	CP 441 Slots	4-2
4.2	Mounting and Dismounting the CP 441	4-2
4.3	Installing and Removing the Interface Submodules of the CP 441	4-3

4.1 CP 441 Slots

In the rack of the S7-400 programmable controller, there are no specific slots reserved for communication modules.

Position of the CP 441 in the Rack

The CP 441 can be plugged into any slot in the rack, with the following exception:

In all racks the power supply module occupies slots 1 to 3 depending on the width.

For further information on slots, see /2/

4.2 Mounting and Dismounting the CP 441

When mounting and dismounting the CP 441, you must observe certain rules.

Tool

For mounting and dismounting the CP 441 you require a 3.5 mm cylindrical screwdriver.

Mounting Sequence

To insert the CP 441 in a rack, proceed as follows:

1. Remove the filler panel from the slot you want to use by gripping it where marked and pulling it toward you.
2. Insert the CP 441 module and tilt it downward.
3. Screw the module down at the top and bottom with a torque of 0.8 ... 1.1 Nm.

Dismounting Sequence

To dismount the CP 441 from the rack, proceed as follows:

1. Undo the screws at the top and bottom of the module.
2. Tilt the module upward and remove it.
3. Replace the filler panel over the empty slot.

/2/ *S7-400/M7-400 Programmable Controllers, Hardware and Installation*, Manual

Note

The CP 441 can be hot-plugged and hot-pulled, in other words with voltage applied. This means that the CP 441 can be replaced while the programmable logic controller is in operation.

The CP 441 is parameterized automatically once it has been plugged in. The CP 441 then resumes operation.

4.3 Installing and Removing the Interface Submodules of the CP 441

When installing and removing the interface submodules of the CP 441, you must observe certain rules.

Caution

Before inserting the interface submodule, unplug the power supply module or dismount the CP 441 from the rack, otherwise the interface submodule could be permanently damaged.

Tool

To install the interface submodule you require a 3.5 mm cylindrical screwdriver.

Mounting Sequence

To install an interface submodule in the CP 441, proceed as follows:

1. First unplug the power supply module or dismount the CP 441 from the rack (see Section 4.2).
2. Insert the interface submodule carefully in the CP 441's slot. The printed circuit board of the interface submodule must be on the left of the slot.
3. Screw down the interface submodule at the top and bottom with a torque of 0.8 ... 1.1 Nm.

Dismounting Sequence

To remove an interface submodule from the CP 441, proceed as follows:

1. First unplug the power supply module and dismount the CP 441 from the rack (see Section 4.2).
2. Undo the screws at the top and bottom of the interface submodule.
3. Carefully remove the interface submodule from the module slot of the CP 441.

Note

To prevent interference, it is absolutely essential that the two screws used to attach the interface submodule are tightened properly and that the shield of the plug cable is connected to a shield bus.

Only then can the relevant EMC (electromagnetic compatibility) standards be complied with.

Configuring and Parameterizing the CP 441

5

In Section	You Will Find	on Page
5.1	Configuring the CP 441	5-2
5.2	Parameterizing the Communication Protocols	5-3
5.3	Connection Configuration	5-4
5.4	Managing the Parameter Data	5-16
5.5	Multiprocessor Communication	5-17
5.6	Subsequent Loading of Drivers (Transmission Protocols)	5-18
5.7	Subsequent Loading of Firmware Updates	5-20

Parameterization Options

You configure and parameterize the module variants of the CP 441 using STEP 7 or the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

Table 5-1 Configuration Options for the CP 441

Product	Order Number	Parameterized Using the Parameterization Interface	Under STEP 7
CP 441-1	6ES7 441-1AA00-0AE0	As of V1.0	As of V2.1
CP 441-2	6ES7 441-2AA00-0AE0		
CP 441-1	6ES7 441-1AA01-0AE0	As of V3.0	As of V3.0
CP 441-2	6ES7 441-2AA01-0AE0		
CP 441-1	6ES7 441-1AA02-0AE0	As of V4.0	As of V4.0
CP 441-2	6ES7 441-2AA02-0AE0		
CP 441-1	6ES7 441-1AA03-0AE0	As of V5.0	As of V4.02
CP 441-2	6ES7 441-2AA03-0AE0		

5.1 Configuring the CP 441

Once you have installed the CP 441, you have to make it known to the programmable controller. This process is known as ”configuration”.

Requirements

The *CP 441 Point-to-Point Communication, Parameter Assignment* parameterization interface must be installed on the programming device/PC under STEP 7 (see also Table 5-1).

Installation

The *CP 441: Point -to-Point Communication, Parameter Assignment* parameterization interface is supplied together with the programming example on CD. Proceed as follows to install the parameterization interface:

1. Insert the CD into the CD drive of your programming device/PC.
2. Under *Windows 95*, start the dialog for installing software by double-clicking on the “Software” icon in “Control panel”.
3. In the dialog box, select the CD drive and the **Setup.exe** file and start installation.
4. Now follow the step-by-step instructions of the installation program.

Configuration

In the sense used here, configuration means entering the CP 441 in the configuration table of the STEP 7 software. In the configuration table you enter the rack, the slot and the order number of the CP 441. STEP 7 then automatically assigns an address to the CP 441.

The CPU is now able to find the CP 441 in its slot in the rack by way of its address.

Requirements

Before you enter the CP 441 in the configuration table using STEP 7, you must use STEP 7 to create a project and a station.

Further Information

The procedure for configuring S7-400 modules is described in detail in the STEP 7 manual /3/.

In addition, STEP 7’s online help system will provide you with all the assistance you will need when configuring an S7-400 module.

/3/ *Configuring Hardware and Communication Connections STEP 7 V5.1* Manual

5.2 Parameterizing the Communication Protocols

Once you have entered the CP 441 in the configuration table, you have to supply the CP 441 communication processors and the interface submodules of the CP 441 with parameters. In the case of the printer driver, you can also configure message texts for printer output. This process is known as ”parameterization”.

Parameterization

The term ”parameterization” is used in the following to describe the setting of interface-specific parameters and the configuration of message texts. You set these parameters with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

You start the parameterization interface by double-clicking the order number (CP 441) in the configuration table or by selecting the CP 441 and choosing the **Edit > Object Properties** menu command.

You do not have to specify any settings for the CP 441 on the ”General” and ”Addresses” tabs.

Select the ”Basic parameters” tab and enter the interface number and the type of interface. Click on the ”Parameters” button to go to protocol selection. Set the protocol and double-click the icon for the transmission protocol (an envelope). This takes you to the dialog box for setting the protocol-specific parameters.

Further Information

The *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface is intuitive and easy to use; the procedure is the same for all communications processors. For this reason, the parameterization interface is not described in detail here. In addition, the online help system will provide you with all the assistance you will need for working with the parameterization interface.

5.3 Connection Configuration

The CP 441 represents the link between an S7 CPU and a communication partner linked by means of a point-to-point connection. The S7 connections are converted to the address mechanisms of the selected transmission protocol on the CP 441.

When you configure a connection, you specify the route the connection takes from the S7 CPU to the CP 441 via the serial link to the communication partner.

As a result of connection configuration, you obtain the connection ID that you have to specify as the parameter ”ID” in your user program when you call a system function block in order to exchange data with the corresponding communication partner.

Simplified Connection Configuration

STEP 7, version 4.0 and later, also allows you to carry out simplified connection configuration. In this case, you do not need to create any PTP subnets or network the interface. To carry out simplified connection configuration, you need to do two things:

1. Enter the connection in the connection table.
2. Set the object properties of the connection.

Complete Connection Configuration

You configure a point-to-point connection between your CP 441 and the communication partner using STEP 7. Proceed as follows:

1. Insert a point-to-point subnet.
2. Attach the CP 441 to the subnet.
3. Select or insert the connection partner, and attach the partner to the subnet.
4. Enter the connection in the connection table.
5. Set the object properties of the connection.

There are differences depending on whether the connection partner is a CP 441 **or** a CP 340, an S5 CP, a printer or a non-Siemens station or device **and** on the protocol used for the connection.

Further Information

You will find general information on how to configure connections with STEP 7 in the STEP 7 Manual /3/.

In addition, STEP 7’s online help system will provide you with assistance in configuring a connection.

You will find specific application examples in Chapter 6, ”Programming Example for System Function Blocks”.

/3/ *Configuring Harware and Communication Connections STEP 7 V5.1* Manual

5.3.1 Simplified Connection Configuration

Entering a Connection in the Connection Table

Proceed as follows:

1. In SIMATIC Manager, display the ”<Offline> (Project)” project window, and double-click the CPU in your SIMATIC 400 station.

   Result: The ”Connections” object (connection table) appears on the right.

2. Double-click this icon. The ”Configuring Connections” dialog box appears. Choose **Insert > Connection** to insert your connection in the connection table.

3. In the ”New Connection” dialog box, select ”Unspecified” as the communication partner and enter ”S7 PTP Connection” as the connection type. Then click ”OK” to exit the dialog box.

4. In the ”Object Properties” dialog box, set the specific properties of the connection:

   In the ”Object Properties” dialog box, change the name of the communication partner from ”Unspecified” to an appropriate name (the name is entered automatically in the connection table) and make selections in the “Via PTP CP” and “Interface” list boxes.

   No other settings are usually necessary. Read the following sections If you want to define more than one connection per interface (e.g. for multicomputing):

   – Section 5.3.3 ”Object Properties” Dialog Box for the ASCII Driver, Printer Drivers and 3964(R) Procedure

   – Section 5.3.4 ”Object Properties” Dialog Box for the RK 512 Computer Connection

5. Click ”OK” to return to the ”Configuring Connections” dialog box.

Result: The ”Configuring Connections” dialog box displays the connection that you have added and the ”Local ID (Hexadecimal)”. You have to specify this ID as the parameter ”ID” at the system function block in the user program of your CPU.

If your communication partner is a CP 441, please note the following:

Note

Whereas a homogeneous S7 connection ends directly at the two end points (CPUs) of the connection, a point-to-point connection consists of a ”partial connection” from the CPU to the CP 441 in one station and a ”partial connection” from the CPU to the CP 441 in the partner station. You therefore have to configure a connection on your partner station as well in order to enable a point-to-point connection between the two CPUs, and the local IDs may be different.

5.3.2 Complete Connection Configuration

To present the point-to-point connection graphically, proceed as follows:

Select Netpro

1. In SIMATIC Manager, display the ”<Offline> (Project)” project window, and double-click the CPU in your SIMATIC 400 station.

   Result: The ”Connections” object (connection table) appears on the right.

2. Double-click this icon. The ”Configuring Connections” dialog box appears.

Inserting a PTP Subnet

- Select **Insert > Network Object** to open a catalog. In the catalog, select “Subnets” and then select “PtP”.

  Result: The point-to-point network is displayed.

Selecting a Communication Partner

If your connection partner is another CP 441, the station should already be in the subnet. If your partner is an S5-CP PtP, a printer, a non-Siemens device or an S7-CP PtP, without communication-bus connection (CP 340, CP 341) enter “Other Station” or “SIMATIC S5” as a dummy value. You do so by selecting **Insert > Network Object**. In the open catalog, select “Stations” and then select “Other Station” or “SIMATIC S5”. You then have to identify the station as a PtP station. To do so:

Double-click station, select user list and click the ”New” button; select PtP station and link the station into the network by selecting the point-to-point network with PtP Network.

Attaching the CP 441 and Connection Partner to the PtP Network

Use the mouse to drag the PtP connection of the CP 441 to the PtP network in order to attach it.

Entering a Connection in the Connection Table

1. Select **Insert > Connection** to add a new connection to the connection table of the CPU you selected.
2. In the ”New Connection” dialog box, select SIMATIC 400 station(2) or ”Other Station” or ”SIMATIC S5” as communication partner and enter ”S7 PTP Connection” as the connection type. Then click ”OK” to exit the dialog box.
3. In the ”Object Properties” dialog box, set the specific properties of the connection:
   – See 5.3.3:”Object Properties” Dialog Box for the ASCII Driver, Printer Drivers and 3964(R) Procedure
   – See 5.3.4 ”Object Properties” Dialog Box for the RK 512 Computer Connection
4. Click ”OK” to return to the ”Configuring Connections” dialog box.

Result: The ”Configuring Connections” dialog box displays the ”Local ID (Hexadecimal)” of the connection that you have added. You have to specify this ID as the parameter ”ID” at the system function block in the user program of your CPU in the SIMATIC 400 station(1).

Note

If your connection partner is another SIMATIC 400 station with a CP 441, you have to configure a connection on your partner station as well in order to enable a point-to-point connection between the two CPUs, and the local IDs may be different.

Note

Please note that you cannot configure more than 8 connections for each interface of the CP 441.

5.3.3 ”Object Properties” Dialog Box for the ASCII Driver, Printer Driver and 3964(R) Procedure

”Object Properties” Dialog Box

In addition to making the entry in the connection table, you also have to set specific properties for each point-to-point connection.

If a point-to-point connection consists of two ”partial connections”, you have to set the object properties for each partial connection.

Below you will find a description of how to call and set the parameters of the ”Object Properties” dialog box for the ASCII driver, the printer driver and the 3964(R) procedure.

Calling the Dialog Box

The ”Object Properties” dialog box appears automatically when you insert a new connection in the connection table. You can also call this dialog box for a connection subsequently:

1. Select the connection from the connection table.
2. Choose **Edit > Object Properties**.

Result: The ”Object Properties” dialog box appears.

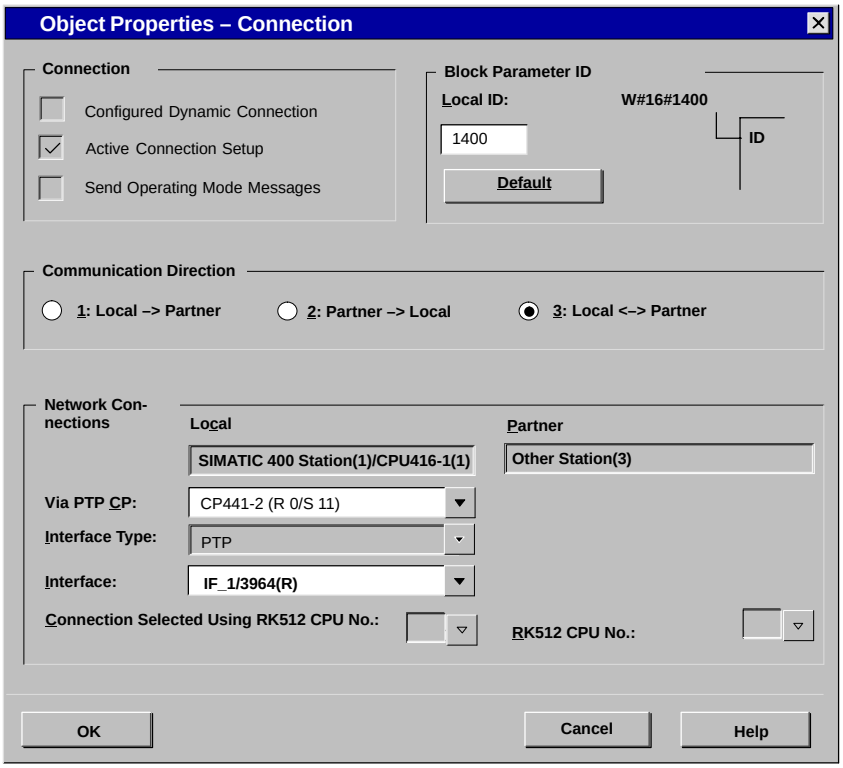


Figure 5-1 Object Properties for an S7 Point-to-Point Connection (1)

Settings

You set the properties of a connection in the ”Object Properties” dialog box:

Table 5-2 Settings in the ”Object Properties” Dialog Box for the ASCII Driver, Printer Driver and 3964(R) Procedure

Parameter	Description
Configured Dynamic Connection	The check box is grayed and not selected: The connection is set up automatically during startup and is sustained until shutdown.
Active Connection Setup	The check box is grayed and selected: The connection is always set up by the local station.
Send Operating Mode Messages	The check box is grayed and not selected: Operating status messages cannot be sent.
Local ID	Local ID (hexadecimal) which you have to specify as the parameter ”ID” at the system function block in the user program of your CPU. You can change the suggested ID if you have programmed the SFBs with certain IDs.
Communication direction	Specify the direction in which communication is to take place by selecting the appropriate option (see also the sections entitled ”One Connection Configured” and ”Several Connections Configured”)
Interface	Interface The CP 441-2 has two channels (the interfaces IF1 and IF2) via which the point-to-point connections can be set up. Select the channel used for the configured connection. /Protocol Various protocols can be used to send data via point-to-point connections. You specified the protocol when you configured the module.
Connection Selected Using RK512 CPU No.	These fields are relevant only to the RK 512 Computer Connection (see Section 5.3.4). These fields are grayed out.
RK512 CPU no.	

Note

If your communication partner is a CP 441, you also have to set the object properties for the partial connection in the partner station.

One Connection Configured

If you have only configured one connection via an interface, you do not have to specify any settings in the ”Object Properties” dialog box.

Several Connections Configured

Up to eight connections can go via a single interface. You can send data via all eight connections (active requests: BSEND). You can only receive data passive requests: (BRCV) via only one connection, however, since the ASCII driver and the 3964(R) procedure do not send any address information with the data.

In the ”Communication Direction” area of the ”Object Properties” dialog box, you have to specify whether you want to send and/or receive data via the selected connection:

1: Local → Partner

For the connections via which you send data. No other settings are necessary.

2: Partner → Local

For the connection via which you receive data. No other settings are necessary.

3: Local ↔ Partner

For the connection via which you send and receive data. No other settings are necessary.

Note

Data can only be received via one connection for each interface. If you have set ”2: Partner → Local” or ”3: Local ↔ Partner” as the communication direction for a connection via one interface, you can select ”1: Local → Partner” for the other connections via the interface in question.

5.3.4 ”Object Properties” Dialog Box for the RK 512 Computer Connection

”Object Properties” Dialog Box

In addition to making the entry in the connection table, you also have to set specific properties for each point-to-point connection.

If a point-to-point connection consists of two ”partial connections”, you have to set the object properties for each partial connection.

Below you will find a description of how to call and set the parameters of the ”Object Properties” dialog box for the RK 512.

Calling the Dialog Box

The ”Object Properties” dialog box appears automatically when you insert a new connection in the connection table. You can also call this dialog box for a connection subsequently:

1. Select the connection from the connection table.
2. Choose **Edit > Object Properties**.

Result: The ”Object Properties” dialog box appears.

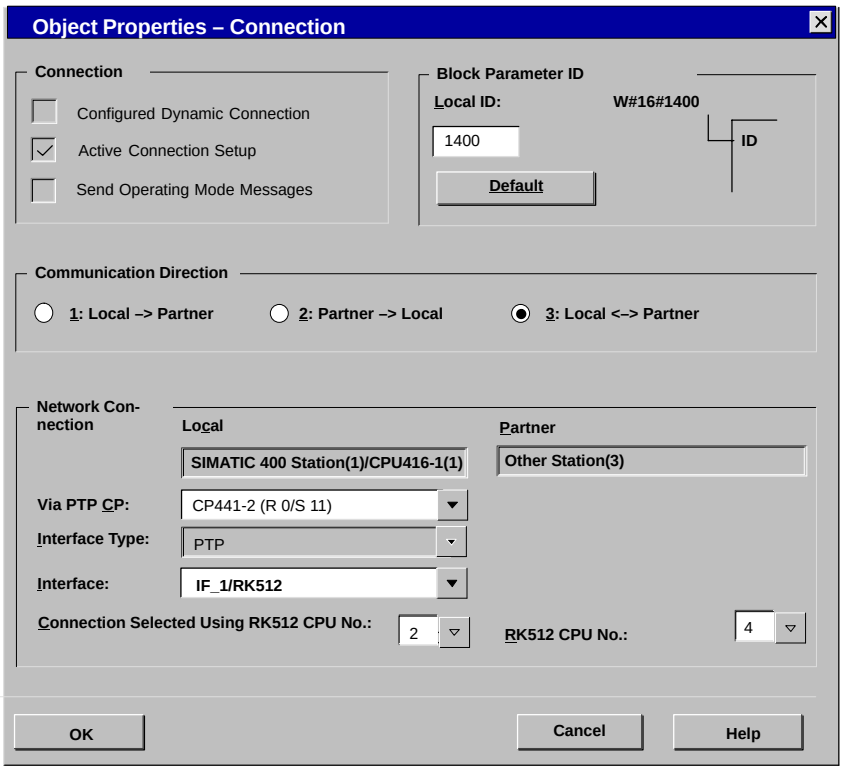


Figure 5-2 Object Properties for an S7 Point-to-Point Connection (2)

Settings

You set the properties of a connection in the ”Object Properties” dialog box:

Table 5-3 Settings in the ”Object Properties” Dialog Box for the RK 512

Parameter	Description
Configured Dynamic Connection	The check box is grayed and not selected: The connection is set up automatically during startup and is sustained until shutdown.
Active Connection Setup	The check box is grayed and selected: The connection is always set up by the local station.
Send Operating Mode Messages	The check box is grayed and not selected: Operating status messages cannot be sent.
Local ID	Local ID (hexadecimal) which you have to specify as the parameter ”ID” at the system function block in the user program of your CPU. You can change the suggested ID if you have programmed the SFBs with certain IDs.
Communication direction	Specify the direction in which communication is to take place by selecting the appropriate option (see also the sections entitled ”One Connection Configured” and ”Several Connections Configured”)
Interface	Interface The CP 441-2 has two channels (the interfaces IF1 and IF2) via which the point-to-point connections can be set up. Select the channel used for the configured connection. /Protocol Various protocols can be used to send data via point-to-point connections. You specified the protocol when you configured the module.
Connection Selected Using RK512 CPU No.	If you have selected **Partner** → **Local** or **Local** ↔ **Partner** as the communication direction, enter the CPU number (1 to 4) by means of which your partner can address this connection.
RK512 CPU no.	If you have selected **Local** → **Partner** or **Local** ↔ **Partner** as the communication direction, enter the CPU number (1 to 4) to which the connection goes.

Note

If your partner is a CP 441, you also have to set the object properties for the partial connection in the partner station.

One Connection Configured

If you need only one connection for your interface, you do not need to specify any settings in the ”Object Properties” dialog box.

Several Connections Configured

In the ”Communication Direction” area of the ”Object Properties” dialog box, you have to specify whether you want to send (active requests: BSEND, PUT, GET) and/or receive (passive requests: BRCV) message frames via the selected connection **and**, for ”Partner” and ”Local”, you have to specify the CPUs via which the connection will be routed. If you want to send and receive message frames, you must make an entry for both ”Partner” and ”Local”.

1: Local → Partner

For the connections via which you send message frames (BSEND, PUT, GET).

2: Partner → Local

For the connections via which you receive message frames (BRCV).

3: Local ↔ Partner

For the connections viawhich yousend and receive message frames.

Local, Connection Selected Using RK512 CPU No.

The RK512 protocol allows you to specify a CPU No. in the range 1-4 in the message frame header in order to address up to four 4 CPUs (connections).

To receive (BRCV) message frames, you enter a CPU number from 1 to 4. The CP 441 compares the CPU number you set here with the CPU number of the received RK 512 message frame entered in the message frame header, byte 10. If they are the same, the CP 441 forwards the data received via this connection.

Note

The number of connections per interface via which you can **send** message frames is limited to eight.

The fact that the CPU numbers in the message frame header are limited to 1 to 4 limits the number of connections via which you can **receive** message frames at a single interface to a maximum of four.

In the ”Local” box, a CPU number that has already been assigned cannot be assigned again for a different connection at the same interface.

Note

If you have set”2: Partner → Local” or ”3: Local ↔ Partner” as the communication direction for a connection via one interface, you must enter another CPU No. in the ”Local” field for another connection for receiving data or, if you only want to send data, you must select ”1: Local → Partner” as the communication direction.

Partner, RK 512 CPU No.

To send (BSEND, PUT, GET) message frames, you enter a CPU number from 1 to 4. The CP 441 enters the CPU number you set here in the message frame header, byte 10, of the RK 512 message frame to be sent. This makes it possible via this connection to address one of four different recipients at the partner.

Examples

Example 1:

Task: You want data to be sent (or fetched) by means of RK 512 from your S7-400 system. The data is to be stored on the partner’s CPU 3 (or fetched by CPU 3).

Parameterization: You must specify Local → Partner as the communication direction and enter the CPU number 3 in the ”Partner, RK512 CPU No.” field.

Example 2:

Task: You want by means of RK 512 to receive from the partner data identified by the CPU number 2 in the RK 512 message frame.

Parameterization: You must specify Partner → Local as the communication direction and enter the CPU number 2 in the ”Local, Connection Selected Using RK 512 CPU No.” field.

Example 3:

Task: You want data to be sent from your S7-400 system to the partner (or fetched) by means of RK 512. The data is to be stored on the partner’s CPU 3 (or fetched by CPU 3). At the same time, you want to use the connection to receive from the partner data identified by the CPU number 2 in the RK 512 message frame.

Parameterization: You must specify Local ↔ Partner as the communication direction and enter the CPU number 3 in the ”Partner, RK512 CPU No.” field and the CPU number 2 in the ”Local, Connection Selected Using RK 512 CPU No.” field.

5.4 Managing the Parameter Data

The CP 441's configuration and parameterization data (including the message texts) is stored in the current project (on the hard disk of the programming device/PC).

Data Storage

When you leave the configuration table (see Section 5.1) and the connection configuration table (see Section 5.3), the configuration and parameterization data (including the module parameters) is saved and stored automatically in the project/user file you have created.

Loading the Configuration and Parameterization Data

You can load the configuration and parameterization data onto the CPU online from the programming device (by choosing **PLC > Download**). The CPU puts the parameters into effect as soon as they are loaded.

The module parameters of the CP 441 are transferred to the CP 441 automatically at start-up of the CPU as soon as the CP 441 is accessible via the S7-400 backplane bus. Default settings apply if parameters are not changed.

Reading Back Parameters

From STEP 7, V5.0 + Service Pack 2 onwards, you can view the parameters of the CP 441 modules online via HW Config. From this view, you cannot change the parameters and they are therefor dsiplayed in gray.

Further Information

The Manual for STEP 7 /3/ describes in detail how to:

- save the configuration and parameters.
- load the configuration and parameters onto the CPU.
- read, modify, copy and print the configuration and parameters.

/3/ *Configuring Hardware and Communication Connections STEP 7 V5.1* Manual

5.5 Multiprocessor Communication

The CP 441 (6ES7 441-_AA02-0AE0 and higher) enables communication to be set up involving up to 4 CPUs in an automation system.

Requirements

STEP 7, Version 4.02 or higher

Please Note

Please observe the following rules of multiprocessor communication:

- Data can be sent from any CPU.
- In the case of the ASCII driver and the 3964(R) procedure, data can only be received via one CPU, since these protocols do not send any address information with the data.
- In the case of the RK 512 computer connection, data can be received on 4 CPUs. Addressing is by means of the CPU numbers 1 to 4 in the header of the RK 512 message frame.

  The CPU numbers automatically assigned in the configuration table of STEP 7 are entered by default during connection configuration in the ”Connection Selected Using RK512 CPU No.” field of the ”Object Properties” dialog box (see Section 5.3.4).

5.6 Subsequent Loading of Drivers (Transmission Protocols)

To extend the functionality of the CP 441 and adapt it to the communication partner, you can load other transmission protocols on the CP 441-2 (loadable drivers) in addition to the standard protocols in the module firmware (ASCII, 3964(R), RK 512, printer).

The loadable drivers are not shipped with the CP 441 or the parameterization interface as standard. You have to order them separately (see the chapter entitled ”Loadable Drivers” in the ST 70 catalog).

To find out how to install and parameterize a loadable driver and load it onto the CP 441-2, consult the separate documentation for the loadable driver. Only the requirements and the fundamentals are described below.

Requirements

The prerequisites for loading the drivers are:

- STEP 7 V4.02 or higher
- *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface, V5.0 or higher
- The CP 441-2 (order number 6ES7 441-2AA02- 0AE0 or higher)
- The dongle provided with the driver must be installed on the CP 441-2.

Entry via Parameterization Interface

You select the loadable driver for parameterization in the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

After you have successfully installed the parameterization interface and loadable drivers, you select the driver you want and set the protocol-specific parameters in the same way as you do for the standard protocols. See section 5.2 for more information on installing the parameterization interface and selecting a transmission protocol.

To find out what is parameterized and how to load the drivers onto the CP 441, consult the separate documentation for the loadable driver.

Please Note

Please note the following:

- The loadable driver is stored in the CPU's load memory together with the parameterization data and loaded automatically onto the CP 441-2 during the loading operation. You must therefore reserve the requisite memory space in the load memory of the CPU for every interface on which you want to load the driver.
- Loadable drivers are loaded only once to the CP 441. The driver download (2 SDBs) is interrupted if parameterization is attempted. This is indicated by the INTF LED on the CPU and the corresponding entry (error loading SDB) in the CPU's diagnostic buffer. In this case the entry is of no significance.
- Before you transfer the loadable driver to the CP 441-2, you must increase the value of the parameter for transferring parameters to modules for the relevant CPU. Allow approximately 15 seconds for each loadable driver.

5.7 Subsequent Loading of Firmware Updates

Firmware updates can be uploaded to the operating-system memory of the CP 441 as patches.

Subsequent loading of firmware updates with the parameterization interface *CP 441: Point-to-Point Communication, Parameter Assignment*.

Basic Firmware

The CP 441 is shipped with basic firmware preinstalled.

Requirements

The prerequisites for loading firmware updates are:

- *STEP 7*, V4.02 or higher
- *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface, V5.0 or higher
- You must create a valid project under the hardware configuration and upload it to the CPU before you can update the firmware of the communications processor with the parameterization interface.
- The instructions accompanying the firmware update always detail the destination directories for the files.

The **..\CP441.**nnn path always identifies the firmware version.

Loading Firmware

You upload the firmware update to the CP 441 with the aid of the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

Proceed as follows:

1. Switch the CPU to STOP mode.
2. Start the parameterization interface:

   In SIMATIC Manager: **File > Open Object > Project >** Open Hardware Config **>** double-click on CP 441 **>** select the “Parameters” button.
3. Select the menu command **Options > Firmware Update**.

   Result:

   If a connection can be established to the CP 441, the current module firmware status is displayed.

   If there is no firmware loaded on the CP 441, the display shows “ – – – – “. This can occur, for example, if a firmware update was canceled. The original firmware is deleted prior to the cancellation. You have to upload firmware to the module before it can be restarted.
4. Click on the ”Find File ...” button to select the firmware to be loaded (*.UPD).

   Note:

   The basic firmware consists of three files each with a *.UPD extension. Select only the file called **HEADER.UPD** for the basic firmware.

   Result:

   The version of the firmware you select is displayed under ”Status of selected firmware:”.
5. Click on the ”Load Firmware” button to start uploading to the CP 441. You are prompted for confirmation. The upload procedure is canceled immediately if you click on the ”Cancel” button.

   Note:

   Before the basic firmware is deleted from the module, the CP 441 checks the MLFB No. of the firmware to be downloaded in order to ensure that the firmware is suitable for the CP 441.

   Result:

   The new firmware is loaded into the operating-system memory of the CP 441. ”Done” shows progress in bar-graph form and as a percentage.

LED Indicators

LED indicators for firmware update:

Table 5-4 LED Indicators for Firmware Update

Status	INTF/ EXTF	FAULT	TXD	RXD	Comment	Remedy
Firmware update in progress	on	on	on	on	–	–
Firmware update completed	on	off	off	off	–	–
CP 441 without module firmware	flashing (2Hz)	on	off	off	Module firmware deleted, firmware update was canceled, firmware update still possible	Reload the firmware
Hardware fault during firmware update	flashing (2Hz)	off	flashing (2Hz)	flashing (2Hz)	Delete/write failed	Switch power supply to module off and then on again and reload the firmware. Check module for defects.

Viewing the Hardware and Firmware Version

You can view the current hardware and firmware version of the CP 441 in *STEP 7* in the “Module Status” dialog box. To access this dialog box:

In SIMATIC Manager: **File > Open Object > Project >** Open HW Config **> Station > Open Online >** and double-click on CP 441.

Communication via System Function Blocks

6

6.1 Communication via System Function Blocks

Communication between the CPU, the CP 441 and a communication partner takes place via the system function blocks of the CPU and the protocols of the CP 441.

Communication CPU and CP 441

The system function blocks form the software interface between the CPU and the CP 441. They are called from the user program.

Communication CP 441 and Communication Partner

The transmission protocols are applied on the CP 441. By means of the protocol, the interface of the CP 441 is adapted to the interface of the communication partner.

This enables you to link an S7 automation system with any communication partner that can handle the modern, standard protocols available in SIMATIC S5 (ASCII driver, 3964(R) procedure or RK 512 computer connection).

6.2 Overview of the System Function Blocks

The S7-400 programmable controller provides you with a number of system function blocks which initiate and control communication between the CPU and the CP 441 communication processor in the user program. The system function blocks are stored permanently in the CPU memory.

S7-400 System Function Blocks

The following table shows the system function blocks of the S7-400 programmable controller which you can use for communication between the CPU and the CP 441.

Table 6-1 System Function Blocks of the S7-400 Programmable Controller

SFB	Meaning
BSEND (SFB 12)	The BSEND system function block allows you to send data from an S7 data area to a communication partner with fixed destination.
BRCV (SFB 13)	The BRCV system function block allows you to receive data from a communication partner and transfer it to an S7 data area.
GET (SFB 14)	**RK 512 only:** The GET system function block enables you to fetch data from a communication partner.
PUT (SFB 15)	**RK 512 only:** The PUT system function block enables you to send data to a communication partner with dynamically changeable destination.
PRINT (SFB 16)	The PRINT system function block allows you to output a message text containing up to 4 variables to a printer.
STATUS (SFB 22)	The STATUS system function block allows you to query the device status of a communication partner. This system function block is described in Section 8.4 ”Diagnosis via the Error-Signaling Area SYSTAT”.

For a detailed description of the system function blocks and their parameters, see the reference manual on S7-300 and S7-400 system software /4/.

/4/ *System Software for S7-300 and S7-400, System and Standard Functions* Reference Manual

6.3 Using the System Function Blocks

The following sections describe what you must take into account when supplying parameters for the system function blocks in your own programmable controller (S7-400).

The communication between two CP 441s is described.

For points to note with regard to other communication partners of the CP 441, see the relevant *SIMATIC S5 literature* or *non-Siemens literature*.

Description of the SFB Parameters

The parameters of the SFBs can be subdivided in terms by function into the following five classes (classification):

- Control parameters (for activating communication)
- Addressing parameters (for addressing the remote communication partner)
- Send parameters (which point to the data areas to be sent to the remote partner)
- Receive parameters (which point to the data areas in which the data received from the remote partner is entered)
- Status parameters (for monitoring whether the block has completed a task without errors and for analyzing the errors that occur)

Control Parameters

Data transfer is only activated if the associated control parameters have a defined value when the SFB is called or if the value has changed in a defined way since the last SFB call. We therefore refer to them as level- or edge-triggered control parameters.

Table 6-2 SFB Control Parameters

Parameter	Meaning	Sender/ Recipient	Function Activated at	Description
REQ	Request	Sender of the request	Positive edge (compared to last SFB call). In other words, before you call the SFB with ”1”, it must have run through once with ”0”.	Activates data transfer (provided certain conditions are fulfilled)
R	Reset	Sender of the request	Positive edge (compared to last SFB call). In other words, before you call the SFB with ”1”, it must have run through once with ”0”.	Activates cancellation of an active data transfer
EN_R	Enabled to receive	Recipient of the request	Level 1	Indicates readiness to receive

Addressing Parameters

The addressing parameters of the SFBs are listed below:

Note

The addressing parameters ID and R_ID are only evaluated the **first time the block is called** (the actual parameters or the predefined values from the instance). The communication relationship (connection) to the remote partner is thus defined at the first call and remains so until the next restart of the CPU.

Table 6-3 SFB Addressing Parameters

Parameter	Description	Please Note
ID	At the SFBs you specify as the ”ID” the ”local ID” (a hexadecimal value between 1000 and 1400) of the connection via which the system function block is to go. In order to do this, you must first have configured the connection using STEP 7 (see Section 5.3). You get the value of the ”Local ID” from the ”Configuring Connections” dialog box in STEP 7.	• ID must be specified in the form W#16#wxyz.
R_ID	The meaning of the R_ID parameter is given in the subsequent descriptions of the transmission protocols.	• R_ID must be specified in the form W#16#wxyz.

Status Parameters

The status parameters allow you to monitor whether the block has completed its task successfully or is still doing it. They also display errors that occur.

Note

The status parameters are only valid for a single cycle – from the first command following the SFB call to the next SFB call. Consequently, you have to evaluate these parameters after **every** block cycle.

Table 6-4 SFB Status Parameters

Parameter	Data Type	Sender/ Recipient	Description
DONE	BOOL	Sender	0: The request has not yet been started or is still being executed. 1: The request has been completed without error. In other words: – With ASCII driver: Request was sent to the communication partner. This does not necessarily mean that the data was received by the communication partner. – With 3964(R) procedure: Request was sent to the communication partner and positive acknowledgement was returned. This does not necessarily mean that the data was forwarded to the partner CPU. – With RK 512 computer connection: Request was sent to the communication partner, which forwarded it without error to the partner CPU.
NDR	BOOL	Receiver	0: The request has not yet been started or is still running. 1: The request has been completed successfully.

Table 6-4 SFB Status Parameters

<table>
<tr><th>Parameter</th><th>Data Type</th><th>Sender/ Recipient</th><th colspan="3">Description</th></tr>
<tr><td rowspan="6">ERROR
STATUS</td><td rowspan="6">BOOL
WORD</td><td rowspan="6">Sender and recipient</td><td colspan="3">Error display:</td></tr>
<tr><td>ERROR</td><td>STATUS</td><td>Meaning</td></tr>
<tr><td>0</td><td>0</td><td>Neither a warning nor an error</td></tr>
<tr><td>0</td><td>≠0</td><td>Warning. STATUS provides detailed information.</td></tr>
<tr><td>1</td><td>≠0</td><td>There is an error. STATUS provides detailed information on the error.</td></tr>
<tr><td colspan="3">What the ”STATUS” error information means for each system function block is described in Section 8.3. You can obtain point-to-point error information that goes beyond this (described in Section 8.4) by calling the STATUS system function block (SFB 22).

Note: The ERROR bit and STATUS remain there only until the next SFB call. To display the STATUS, you should therefore copy it to a free data area.</td></tr>
</table>

Note

The data consistency is specified by the receiving CPU (CPU 412/413: 16 bytes, CPU 414/416: 32 bytes). For further information on data consistency, refer to the reference manual to the system and standard funcitons /4/.

To guarantee further data consistency, please observe the following:

- Sender: Only access the send DB when all data have been completely transferred (DONE = 1).
- Receiver: Only access the receive DB when all data are received (NDR = 1). Then you must inhibit the receive DB (EN_R = 0) until you have processed the data.

Send and Receive Parameters

The SD_i send parameters and the RD_i receive parameters are of the ANY data type, but no bit fields can be used.

See the CP441 ANY demo project for instructions on how to change the send and receive parameters of the ANY data type at runtime. The demo project is in the ”Examples” STEP7 catalog under CP441.

If you do not use all the send and receive parameters with an SFB, the first unused parameter must be a NIL pointer, and the used parameters must come one after the other without any gaps.

At the first call, the connection and the maximum amount of data that can be transferred via it per job is fixed. The system creates a communication buffer to ensure data consistency.

At subsequent calls you can send/receive any amount of data as long as it does not exceed that of the first call.

The BSEND and BRCV SFBs represent and exception to this rule. You can transfer up to 64 KB per request using them.

The following applies to the BSEND/BRCV SFBs:

- The number of SD_i and RD_i parameters used at the sending and receiving ends must match.
- The data types of SD_i and RD_i parameters at the sending and receiving ends that belong together must match.
- The amount of data to be sent by means of the SD_i parameter must not be greater than the area made available by the associated RD_i parameter.

If you break these rules, this is indicated to you by means of ERROR = 1 and STATUS = 4.

Examples of Send and Receive Parameters

Access to data blocks, bytes 10–109

P#DB20.DBX10.0 byte 100

Access to memory markers 10–12

P#M10.0 BYTE 3

Access to inputs 20–24

P#E20.0 BYTE 5

Access to outputs

P#A20.0 BYTE 5

Access to times 1–5

L#1 TIMER 5

Access to counters 1–10

L#1 COUNTER 10

Parallel Processing of Requests

The number of requests (BSEND and GET) which can be processed simultaneously depends on the data volume transmitted with the individual requests.

The requests are buffered on the CP 441 in data blocks of 450 bytes. Up to 40 data blocks can be buffered per interface.

If no further data blocks can be buffered, the request is terminated with an error (STATUS 02). The message 050FH is entered in the error-signaling area.

Example:

If all requests are 2000 bytes long, for example, 8 requests can be buffered.

Number of transmittable data sets

If a PLC has more than one CP. the number of data sets that can be transmitted depends largely on CPU performance. A CPU 416, for example, can handle approximately 80-100 message frames of 240 bytes per second (order number of the CPU: 6ES7 416-1XJ02-0AB0.

If the communication load is increased please note the following:

Behavior	Remedy
Transfer between CP and CPU receives negative acknowledgment (0407 or 0408 in the CP's diagnostic buffer).	• Increase the value of the "Cyclic load due to communication" parameter in the CPU screen form and • Call BRCV in the time OB or call BRCV more frequently in the cycle.
Contents of the diagnostics buffer on the CP cannot be read with a programming device.	Increase the value of the "Minimum cycle time" parameter in the CPU screen form "Cycle"
A newly inserted CP is not parameterized.	Increase the value of the "Transfer parameters to modules" parameter in the "Startup" CPU screen form.

6.4 Using the System Function Blocks with the 3964(R) Procedure

If you are using the 3964(R) procedure as your transmission procedure, you can transmit data from your S7-400 programmable controller to a communication partner.

Data Transmission to a Communication Partner Using 3964(R)

The figure below illustrates how data is sent to a communication partner.

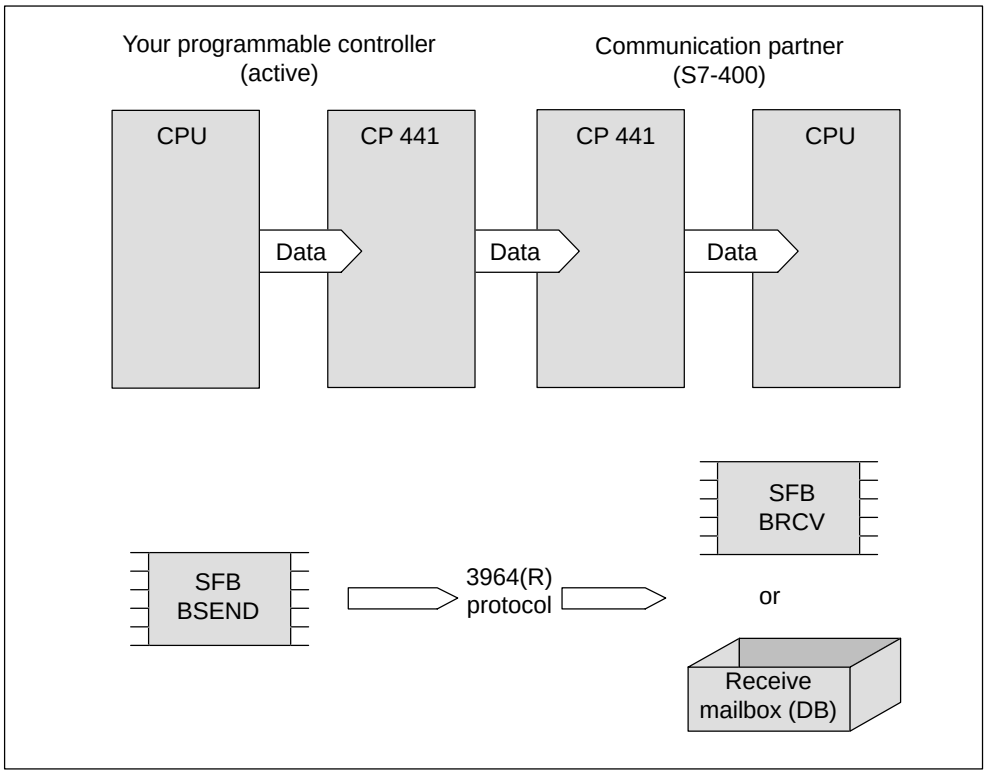


Figure 6-1 Sending Data to a Communication Partner with the 3964(R) Procedure

Data Transmission to the Communication Partner with 3964(R). Options:

To transmit data via the 3964(R) procedure, you have the following options:

- You can send the data with the system function block BSEND and receive the data at the communication partner with the system function block BRCV (see Section 6.4.1).

  This type of data transmission has the advantage that, using the BRCV, you can interpret the NDR parameter to establish when the complete data was received, and the EN_R parameter to prevent unprocessed data from being overwritten at the receiver.

- You can send the data with the BSEND system function block and use the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface to define a receive mailbox (DB) on the communication partner for the CP 441; the incoming data will be stored in this mailbox on the communication partner's CPU (see Section 6.4.2).

  If you use this type of data transmission, you do not need to do any programming in the user program of the communication partner. Note, however, that at the receiver you cannot tell when a transmission is taking place. Therefore, the receiving CP 441 cannot prevent unprocessed data from being overwritten at the receiver.

Note

Please note that destination information is not transferred when transmitting data using the 3964(R) protocol. The data can therefore be sent from more than one source (BSENDs), but stored on one destination only per serial interface (BSEND or receive mailbox).

6.4.1 Data Transmission with 3964(R) Using BSEND and BRCV

What To Do

This type of data transmission has the advantage that, using the BRCV, you can interpret the NDR parameter to establish when the complete data was received, and the EN_R parameter to prevent unprocessed data from being overwritten at the receiver.

On Your Programmable Controller

For each communication request you must program a BSEND (SFB12) system function block in the S7 user program of the CPU.

The R_ID parameter takes any value. When programming more than one BSEND you must use different R_IDs.

For the SD_1 parameter (data type ANY), specify which data (source) is to be passed on.

Example: `p#DB10.DBX5.0 WORD 1`

The length is not evaluated with data type ANY, since the length of the data to be sent is defined in the LEN parameter.

Note that the length of the transmittable data is restricted to 4 KB.

At the CP 441 Communication Partner

In the S7 user program of the CPU you must program the BRCV system function block (SFB 13).

Note

So that no destination information can be transferred to the protocol by this means, the data of all BSENDs with different R_IDs must be received by means of a BRCV.

No more than **one** BRCV system function block can therefore be created for a serial interface.

The **value ”0”** must be specified for the R_ID parameter.

For the RD_1 parameter (data type ANY), specify where the data is to be stored (destination). The length defines the maximum length of the block to be received.

Example: `p#DB20.DBX10.0 WORD 2048`

To prevent unprocessed data from being overwritten, you must call the BRCV with the value 0 at the control input EN_R.

Note that you might have to use the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface to disable (delete) a receive mailbox defined on the CP 441, as otherwise the data will be placed in the data block specified in the receive mailbox instead of being forwarded to BRCV.

Example

Examples of calls for SFB12 (BSEND) and SFB13 (BRCV):

```
L       50                          Calls for SFB 12 and SFB 13
T       DB60.DBW806
CALL SFB 12, DB62                   Following a rising edge at the REQ
REQ     :=DB60.DBX812.0             parameter, the data for a length of 50
R       :=DB60.DBX812.1             bytes starting at data byte 5 in DB10
ID      := W#16#1000                is sent to the communication partner
R_ID    := W#16#5                   with the BSEND request.
DONE    :=DB60.DBX812.2
ERROR   :=DB60.DBX812.3
STATUS  :=DB60.DBW802
SD_1    :=p#DB10.DBX5.0 WORD 1
LEN     :=DB60.DBW806
```

```
SET                                 The data is received with the BRCV
=       DB60.DBX812.4               request and stored in DB20, starting at
CALL SFB 13, DB63                   data byte 10. The LEN parameter shows
ID      := W#16#1001                the length of the received data (50
R_ID    := W#16#0                   bytes).
NDR     :=DB60.DBX812.5             Note that this protocol requires 0 as
ERROR   :=DB60.DBX812.6             the R_ID of the BRCV system function
STATUS  :=DB60.DBW800               block.
RD_1    :=p#DB20.DBX10.0 WORD 2048
LEN     :=DB60.DBW804
```

Request Table

The following table lists the data types which can be transmitted.

Table 6-5 Request Table for Sending Data with the 3964(R) Using BSEND and BRCV

Source, BSEND from S7	To Destination, Communication Partner	Parameterization in the BSEND SFB, Parameter SD_1 (Source)			Parameterization in BSEND SFB, Parameter LEN (Source)	Parameterization in BRCV SFB, Parameter RD_1 (Destination)		
		S-TYPE	S-DBNO	S-Offset (byte)	Length (max. 4096 byte)	D-TYPE	D-DBNO	D-Offset (byte)
Data block	Data block	DB	1 - *	0 - *	1 - *	DB	1 - *	0 - *
Memory Markers	Data block	MB	irrelevant	0 - *	1 - *	DB	1 - *	0 - *
Inputs	Data block	IB	irrelevant	0 - *	1 - *	DB	1 - *	0 - *
Outputs	Data block	QB	irrelevant	0 - *	1 - *	DB	1 - *	0 - *
Counters	Data block	C	irrelevant	0 - *	1 - *	DB	1 - *	0 - *
Times	Data block	T	irrelevant	0 - *	1 - *	DB	1 - *	0 - *

* This value is dictated by the CPU that you use

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

6.4.2 Data Transmission with 3964(R) Using BSEND and a Receive Mailbox

What To Do

This type of data transmission has the advantage that no programming is necessary in the user program of the communication partner.

Note that at the receiver you cannot tell when a transmission is taking place. Therefore, the receiving CP 441 cannot prevent unprocessed data from being overwritten at the receiver. The data is transferred even when the receiving S7 CPU is in STOP mode. The receiving CPU determines data consistency (CPU 412/413: 16 bytes, CPU 414/416: 32 bytes).

You will find more information on data consistency in the section entitled “Exchange of Data via Communication Function Blocks” in the reference manual on system functions and standard functions /4/.

On Your Programmable Controller

For each communication request you must program a BSEND (SFB12) system function block in the S7 user program of the CPU.

The R_ID parameter takes any value. When programming more than one BSEND you must use different R_IDs.

For the SD_1 parameter (data type ANY), specify which data (source) is to be passed on.

Example: `p#DB10.DBX5.0 WORD 1`

The length is not evaluated with data type ANY, since the length of the data to be sent is defined in the LEN parameter.

Please note that the length of the data that can be transferred when a receive mail box is used on the CP 441 is limited to 450 bytes depending on the CPU of the communication partner.

If you use a different communication partner, you can transfer up to 4 kilobytes.

At the CP 441 Communication Partner

You must specify a receive mailbox with its data block (DB) on the CP 441 with the aid of the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface.

In the data block the data arriving via the CP 441 is stored in the CPU. The data block must have been created previously on the CPU. The data block must be 2 bytes longer than the transmittable data, because the receiving CP enters the length of the transmitted data in the first two bytes.

/4/ *System Software for S7-300 and S7-400, System and Standard Functions* Reference Manual

Request Table

The following table lists the data types which can be transmitted.

Table 6-6 Request Table for Sending Data with 3964(R) Using BSEND and a Receive Mailbox

Source, BSEND from S7	To Destination, Communication Partner	Parameterization in the BSEND SFB, Parameter SD_1 (Source)			Parameterization in the BSEND SFB, Parameter LEN (Source)	Specification of DB in Mailbox
		S-TYPE	S-DBNO	S-Offset (byte)	Length max. 450 bytes**	D-DB
Data block	Data block	DB	1 - *	0 - *	1 - *	1 - *
Memory Markers	Data block	MB	irrelevant	0 - *	1 - *	1 - *
Inputs	Data block	IB	irrelevant	0 - *	1 - *	1 - *
Outputs	Data block	QB	irrelevant	0 - *	1 - *	1 - *
Counters	Data block	C	irrelevant	0 - *	1 - *	1 - *
Times	Data block	T	irrelevant	0 - *	1 - *	1 - *

* This value is dictated by the CPU that you use.
** Depending on the CPU of the communication partner, 450 bytes

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-DB	Destination data block

6.5 Using the System Function Blocks with the RK 512 Computer Connection

If you are using the RK 512 computer connection as your transmission procedure, you can

- Send data from your S7-400 programmable controller to a communication partner with a static destination definition (see sections 6.5.1 to 6.5.4),
- Send data from your S7-400 programmable controller to a communication partner with a dynamic destination definition (see section 6.5.5),
- Fetch data from a communication partner (see section ”Fetching Data from a Communication Partner with RK 512”).

6.5.1 Send data with a static destination definition with RK 512

The figure below illustrates how data is sent to a communication partner with static destination definition using RK 512.

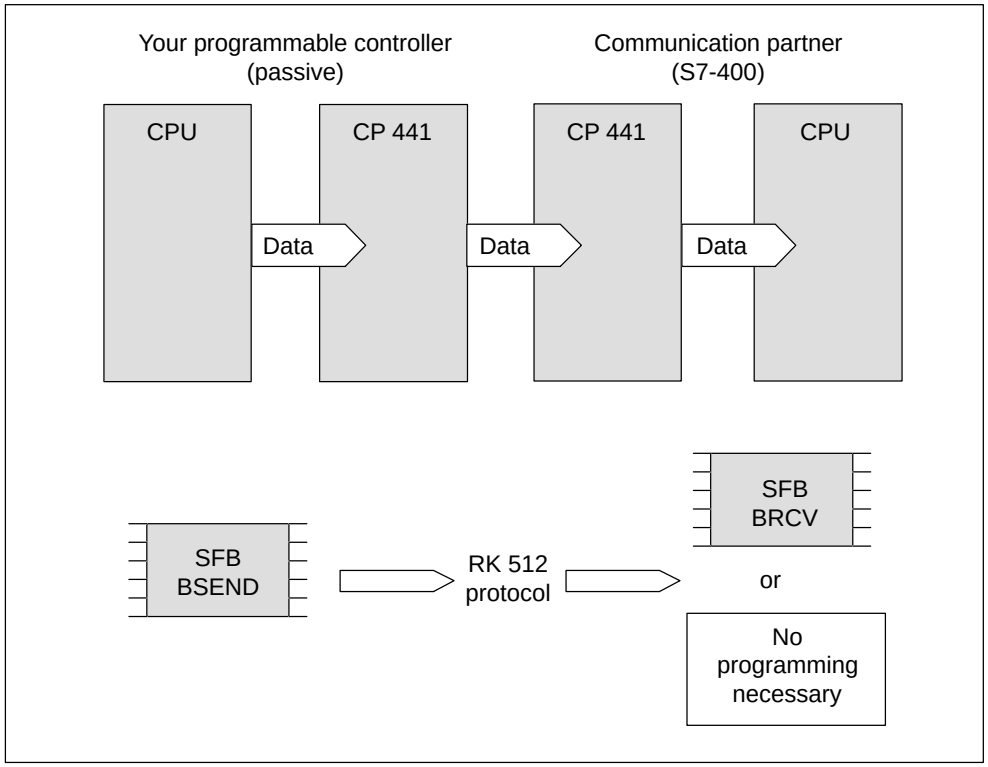


Figure 6-2 Sending Data to a Communication Partner with Static Destination Definition with the RK 512 Computer Link

Note

If you are transmitting data via the RK 512 computer connection, you must distinguish between sending data to another CP 441 (see sections 6.5.2 and 6.5.3) or linking up to an S5 module or non-Siemens device (see section 6.5.4).

Sending an Odd Number of Data

Please note the following when sending an odd number of data:

Note

The RK 512 protocol only allows an even number of data to be sent. If you specify an odd number of data, an additional filler byte with a value of ”0” is sent at the end.

Data Transmission with RK 512 to the CP 441 Communication Partner

If your partner in the link is a CP 441, you have the following options:

- To transmit data you can program a BSEND at the sender and a BRCV at the receiver, indicated by the data type DX in the RK 512 message frame header (see section 6.5.2).

  This type of data transmission has the advantage that, using the BRCV, you can interpret the NDR parameter to establish when the complete data was received, and the EN_R parameter to prevent unprocessed data from being overwritten at the receiver.

- To transmit data you can program a BSEND at the sender only, with no programming at the receiver, indicated by the data type DB in the RK 512 message frame header (see section 6.5.3).

  If you use this type of data transmission, you do not need to do any programming in the user program of the communication partner. Note, however, that at the receiver you cannot tell when a transmission is taking place. Therefore, the receiving CP 441 cannot prevent unprocessed data from being overwritten at the receiver.

6.5.2 Sending Data with RK 512 to the Communication Partner CP 441 with Static Destination Definition, Use of BSEND and BRCV

What To Do

This type of data transmission has the advantage that, using the BRCV, you can interpret the NDR parameter to establish when the complete data was received, and the EN_R parameter to prevent unprocessed data from being overwritten at the receiver.

On Your Programmable Controller

With this programming option, the data source is specified at the sender and the data destination is specified at the receiver.

In the S7 user program of the CPU you must program the BSEND system function block (SFB 12).

For the R_ID parameter you can specify a value from 0 to 255 (decimal). **The value is accepted once during CPU startup and cannot subsequently be changed.** No other values may be specified. The CP 441 transmits the R_ID value 0-255 in the RK 512 message frame header as DX 0-255 (extended data block) to the CP 441 of the communication partner.

R_ID 0-255 (decimal) → DX 0-255 (decimal)

In the CP 441 of the communication partner, this becomes the R_ID value 0-255 again, with which the corresponding BRCV on the partner CPU can be addressed.

For the SD_1 parameter (data type ANY), specify which data (source) is to be passed on.

Example: `p#DB10.DBX5.0 WORD 1`

The length is not evaluated with data type ANY, since the length of the data to be sent is defined in the LEN parameter.

Note that the length of the transmittable data is restricted to 4 KB.

Note

If you send data to a CP 441 by means of an S5 CP or a non-Siemens device, in this mode you must specify DX as the destination data block with the corresponding number in the request block (S5 CP).

The start address is not evaluated. Interprocessor communication flags are not evaluated either. The connection via which the data is forwarded from the CP 441 to the S7 CPU is selected by means of the CPU number (see Section 5.3.4).

Up to 4 KB of data can be transferred.

At the CP 441 Communication Partner

The CP 441 of the communication partner recognizes from the data type DX in the RK 512 message frame header the type of data transmission you have selected. In this case the specifications in the RK 512 message frame header are not the destination parameters but represent the reference to the R_ID of a BRCV (SFB 13) which you must call in the S7 user program of the CPU.

The following applies:

DX 0-255 (decimal) → R_ID=0-255 (decimal)

The actual destination parameters must be specified in the BRCV system function block with the RD_1 parameter (data type ANY). The length defines the maximum length of the block to be received.

Example: `p#DB20.DBX10.0 WORD 2048`

The IPC flag byte and bit from the RK 512 message header (see Section 2.2.3) are not interpreted.

To prevent unprocessed data from being overwritten, you must call the BRCV with the value 0 at the control input EN_R.

Example

Examples of calls for SFB12 (BSEND) and SFB13 (BRCV):

```
L       50                          Calls for SFB 12 and SFB 13
T       DB60.DBW806
CALL SFB 12, DB62                   Following a rising edge at the REQ
REQ     :=DB60.DBX812.0             parameter, the data for a length of 50
R       :=DB60.DBX812.1             bytes starting at data byte 5 in DB10
ID      := W#16#1000                is sent to the communication partner
R_ID    := W#16#5                   with the BSEND request.
DONE    :=DB60.DBX812.2
ERROR   :=DB60.DBX812.3
STATUS  :=DB60.DBW802
SD_1    :=p#DB10.DBX5.0 WORD 1
LEN     :=DB60.DBW806
```

```
SET                                 The data is received with the BRCV
=       DB60.DBX812.4               request and stored in DB20, starting at
CALL SFB 13, DB63                   data byte 10. The LEN parameter shows
EN_R    DB60.DBX812.4               the length of the received data (50
ID      := W#16#1001                bytes).
R_ID    := W#16#5                   Note that the R_ID of the BRCV must be
NDR     :=DB60.DBX812.5             identical to the R_ID of the BSEND.
ERROR   :=DB60.DBX812.6
STATUS  :=DB60.DBW800
RD_1    :=p#DB20.DBX10.0 WORD 2048
LEN     :=DB60.DBW804
```

Request Table

The following table lists the data types which can be transmitted.

Table 6-7 Request Table for Sending Data with RK 512 to the CP 441 Communication Partner, Using BSEND and BRCV"

Source, BSEND from S7	To Destination, S7 Communication Partner	Parameterization in the BSEND SFB, Parameter SD_1 (Source)			Parameterization in BSEND SFB, Parameter LEN (Source)	Parameterization in the BSEND/ BRCV SFB, Parameter R_ID	Parameterization in BRCV SFB, Parameter RD_1 (Destination)		
		S-TYPE	S-DBNO	S-Offset (byte)	Length (max. 4096 byte)	No.	D-TYPE	D-DBNO	D-Offset (byte)
Data block	Data block	DB	1 - *	0 - *	1 - *	0-255	DB	1 - *	0 - *
Memory Markers	Data block	MB	irrelevant	0 - *	1 - *	0-255	DB	1 - *	0 - *
Inputs	Data block	IB	irrelevant	0 - *	1 - *	0-255	DB	1 - *	0 - *
Outputs	Data block	QB	irrelevant	0 - *	1 - *	0-255	DB	1 - *	0 - *
Counters	Data block	C	irrelevant	0 - *	1 - *	0-255	DB	1 - *	0 - *
Times	Data block	T	irrelevant	0 - *	1 - *	0-255	DB	1 - *	0 - *

* This value is dictated by the CPU that you use

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

Specifications in the Message Frame Header of the RK 512 Protocol

The following table shows the specifications in the RK 512 message frame header.

Table 6-8 Specifications in Message Frame Header of RK 512 Protocol, “Sending Data to Communication Partner CP 441 with RK 512, Using BSEND and BRCV”

Source, BSEND from S7	**To Destination, S7 Communication Partner (BRCV)**	**Message Frame Header, Bytes**		
		3/4 Command* Type	**5/6 D-DXNO/D-Offset**	**7/8 Number in**
Data block	Data block	OD	DX/DW	words
Memory Markers	Data block	OD	DX/DW	bytes
Inputs	Data block	OD	DX/DW	bytes
Outputs	Data block	OD	DX/DW	bytes
Counters	Data block	OD	DX/DW	words
Times	Data block	OD	DX/DW	words

* The source information is not transferred to the CP, so the CP always enters the ID for DB (OD) at this point.

Abbreviations:

D-DXNO	Extended destination data block number
D-Offset	Destination start address
DW	Offset in words

6.5.3 Sending Data with RK 512 to the Communication Partner CP 441 with Static Destination Definition, Use of BSEND

What To Do

This type of data transmission has the advantage that no programming is necessary in the user program of the communication partner.

Note that at the receiver you cannot tell when a transmission is taking place. Therefore, the receiving CP 441 cannot prevent unprocessed data from being overwritten at the receiver. The data is transferred even when the receiving S7 CPU is in STOP mode. The receiving CPU determines data consistency (CPU 412/413: 16 bytes, CPU 414/416: 32 bytes).

You will find more information on data consistency in the section entitled “Exchange of Data via Communication Function Blocks” in the reference manual on system functions and standard functions /4/.

/4/ *System Software for S7-300 and S7-400, System and Standard Functions* Reference Manual

On Your Programmable Controller

With this type of programming, the source data and the destination data are specified at the sender.

In the S7 user program of the CPU you must program the BSEND system function block (SFB 12).

For the SD_1 parameter (data type ANY), specify which data (source) is to be passed on.

Example: `p#DB10.DBX5.0 WORD 1`

The length is not evaluated with data type ANY, since the length of the data to be sent is defined in the LEN parameter.

For the destination data area you must specify a data block (DB) between 1 and 255 (decimal). Specify the destination data area of the communication partner in the R_ID parameter. **The value is accepted once during CPU startup and cannot subsequently be changed.** The R_ID parameter (DWORD) is structured as follows:

Byte 1	=	Identifier for data type DB: 1 (hexadecimal)
Byte 2	=	Not relevant (any value)
Byte 3	=	Offset: 0-255 (decimal, in words)
Byte 4	=	DB no.: 1-255 (decimal)

The parameters of the destination data area are transmitted to the communication partner in the RK 512 message frame header.

Please note that the length of the data that can be transferred is limited to 450 bytes depending on the CPU of the communication partner.

Note also that the parameter limits of the RK 512 protocol at the partner only allow you to access data blocks 1 through 255 and to specify as the offset a maximum of 255.

Note

If you send data to a CP 441 by means of an S5 CP or a non-Siemens device, in this mode you must specify DB as the destination data block with the corresponding number and start address (offset) in the request block.

Interprocessor communication flags are not evaluated. The connection via which the data is forwarded from the CP 441 to the S7 CPU is selected by means of the CPU number (see Section 5.3.4).

The length of the transferable data is 450 bytes.

At the CP 441 Communication Partner

The CP 441 of the communication partner recognizes from the data type DB in the RK 512 message frame header the type of data transmission you have selected.

No programming is necessary in the S7 user program of the CPU.

Example

Example of call for SFB12 (BSEND):

```
L       50                          Length = 50 bytes
T       DB60.DBW806
L       B#16#1                      Data type DB
T       DB60.DBB820
L       0                           Not relevant
T       DB60.DBB821
L       20                          starting at data word 20 (offset)
T       DB60.DBB822
L       71                          DB No. 71
T       DB60.DBB823
CALL SFB 12, DB62
REQ     :=DB60.DBX812.0             Following a rising edge at the REQ
R       :=DB60.DBX812.1             parameter, the data for a length of 50
ID      := W#16#1000                bytes starting at data byte 5 in DB 10
R_ID    :=DB60.DBD820               is sent to the communication partner.
DONE    :=DB60.DBX812.2             The partner stores the data in DB 71,
ERROR   :=DB60.DBX812.3             starting at data word 20. The
STATUS  :=DB60.DBW802               destination information in the R_ID is
SD_1    :=p#DB10.DBX5.0 WORD 1      accepted once during CPU startup and
LEN     :=DB60.DBW806               cannot subsequently be changed.
```

Request Table

The following table lists the data types which can be transmitted.

Table 6-9 Request Table for “Sending Data with RK 512 to the Communication Partner CP 441, Using BSEND”

Source, BSEND from S7	To Destination, Communication Partner	Parameterization in the BSEND SFB, Parameter SD_1 (Source)			Parameterization in BSEND SFB, Parameter LEN (Source)	Parameterization in the BSEND SFB, Parameter R_ID (Destination)		
		S-TYPE	S-DBNO	S-Offset (byte)	Leng max. 450 bytes**	D-TYPE	D-DBNO	D-Offset (words)
Data block	Data block	DB	1 - *	0 - *	1 - *	DB	1-255	0-255
Memory Markers	Data block	MB	irrelevant	0 - *	1 - *	DB	1-255	0-255
Inputs	Data block	IB	irrelevant	0 - *	1 - *	DB	1-255	0-255
Outputs	Data block	QB	irrelevant	0 - *	1 - *	DB	1-255	0-255
Counters	Data block	C	irrelevant	0 - *	1 - *	DB	1-255	0-255
Times	Data block	T	irrelevant	0 - *	1 - *	DB	1-255	0-255

* This value is dictated by the CPU that you use

** Depending on the CPU of the communication partner, 450 bytes

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

Specifications in the Message Frame Header of the RK 512 Protocol

The following table shows the specifications in the RK 512 message frame header.

Table 6-10 Specifications in Message Frame Header of RK 512 Protocol, Sending Data to Communication partner CP 441 with RK 512, Using BSEND

Source, BSEND from S7	To Destination, S7 Communication Partner (BRCV)	Message Frame Header, Bytes		
		3/4 Command Type*	5/6 D-DBNO/ D-Offset	7/8 Number in
Data block	Data block	AD	DB/DW	words
Memory Markers	Data block	AD	DB/DW	bytes
Inputs	Data block	AD	DB/DW	bytes
Outputs	Data block	AD	DB/DW	bytes
Counters	Data block	AD	DB/DW	words
Times	Data block	AD	DB/DW	words

* The source information is not transferred to the CP, so the CP always enters the ID for DB (AD) at this point.

Abbreviations:

D-DBNO	Destination data block number
D-Offset	Destination start address
DW	Offset in words

6.5.4 Sending Data with RK 512 to the S5 Communication Partner or Non-Siemens Device with Static Destination Definition

What To Do

If your partner in the link is an S5 CP or a non-Siemens device, proceed as follows:

On Your Programmable Controller

In the S7 user program of the CPU you must program the BSEND system function block (SFB 12).

For the SD_1 parameter (data type ANY), specify which data (source) is to be passed on.

Example: `p#DB10.DBX5.0 WORD 1`

The length is not evaluated with data type ANY, since the length of the data to be sent is defined in the LEN parameter.

Specify the destination data area of the communication partner in the R_ID parameter. **The value is accepted once during CPU startup and cannot subsequently be changed.** The R_ID parameter (DWORD) is structured as follows:

Byte 1, bits 0,1,2,3	=	Identifier for data type DX: 0 (hexadecimal) DB: 1 (hexadecimal)
Byte 1, bits 4,5,6,7	=	IPC flag bits 0–7 (hexadecimal); if you are not using an IPC flag, the protocol enters FH in the message frame header.
Byte 2	=	Byte for IPC flag 1-233 (decimal), or, if you are not using an IPC flag, 255 (decimal)
Byte 3	=	Offset: 0-255 (decimal, in words)
Byte 4	=	DB no.: 3-255 (decimal)

The parameters of the destination data area are transmitted to the communication partner in the RK 512 message frame header.

Note that the length of the transmittable data is restricted to 4 KB.

At the S5 Communication Partner or Non-Siemens Device

For the relevant information you should **read the notes** in the appropriate *S5 manual*, or in the relevant *literature* if you are using a non-Siemens device.

Example

Example of call for SFB12 (BSEND):

```
L       50                          Length = 50 bytes
T       DB60.DBW806
L       B#16#31                     IPC flag bit 3 / data type DB
T       DB60.DBB820
L       30                          Interprocessor communication flag byte
T       DB60.DBB821                 30
L       20                          starting at data word 20 (offset)
T       DB60.DBB822
L       71                          DB No. 71
T       DB60.DBB823
CALL SFB 12, DB62
REQ     :=DB60.DBX812.0             Following a rising edge at the REQ
R       :=DB60.DBX812.1             parameter, the data for a length of 50
ID      := W#16#1000                bytes starting at data byte 5 in DB 10
R_ID    :=DB60.DBD820               is sent to the communication partner.
DONE    :=DB60.DBX812.2             The partner stores the data in DB 71,
ERROR   :=DB60.DBX812.3             starting at data word 20.
STATUS  :=DB60.DBW802               Interprocessor communication flag byte
SD_1    :=p#DB10.DBX5.0 WORD 1      and IPC bit are also transferred. The
LEN     :=DB60.DBW806               destination information in the R_ID is
                                    accepted once during CPU startup and
                                    cannot subsequently be changed.
```

Request Table

The following table lists the data types which can be transmitted.

Data destination DB:

Table 6-11 Request Table for “Sending Data to an S5 Communication Partner or Non-Siemens Device with RK 512, Data Destination DB”

Source, BSEND from S7	To Destination, S5 Communication Partner or Non-Siemens Device	Parameterization in the BSEND SFB, Parameter SD_1 (Source)			Parameterization in the BSEND SFB, Parameter LEN (Source)	Parameterization in the BSEND SFB, Parameter R_ID (Destination)		
		S-TYPE	S-DBNO	S-Offset (byte)	Length (max. 4096 byte)	D-TYPE	D-DBNO	D-Offset (words)
Data block	Data block	DB	1 - *	0 - *	1 - *	DB	3-255	0-255
Memory Markers	Data block	MB	irrelevant	0 - *	1 - *	DB	3-255	0-255
Inputs	Data block	IB	irrelevant	0 - *	1 - *	DB	3-255	0-255
Outputs	Data block	QB	irrelevant	0 - *	1 - *	DB	3-255	0-255
Counters	Data block	C	irrelevant	0 - *	1 - *	DB	3-255	0-255
Times	Data block	T	irrelevant	0 - *	1 - *	DB	3-255	0-255

* This value is dictated by the CPU that you use

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

Specifications in the Message Frame Header of the RK 512 Protocol

The following table shows the specifications in the RK 512 message frame header.

Data destination DB:

Table 6-12 Specifications in Message Frame Header of RK 512 Protocol, “Sending Data to an S5 Communication Partner or Non-Siemens Device with RK 512, Data Destination DB”

Source, BSEND from S7	To Destination, S5 Communication Partner or Non-Siemens Device	Message Frame Header, Bytes		
		3/4 Command Type*	5/6 D-DBNO/D-Offset	7/8 Number in
Data block	Data block	AD	DB/DW	words
Memory Markers	Data block	AD	DB/DW	bytes
Inputs	Data block	AD	DB/DW	bytes
Outputs	Data block	AD	DB/DW	bytes
Counters	Data block	AD	DB/DW	words
Times	Data block	AD	DB/DW	words

* The source information is not transferred to the CP, so the CP always enters the ID for DB (AD) at this point.

Abbreviations:

D-DBNO	Destination data block number
D-Offset	Destination start address
DW	Offset in words

Request Table

The following table lists the data types which can be transmitted.

Data destination DX:

Table 6-13 Request Table for “Sending Data to an S5 Communication Partner or Non-Siemens Device with RK 512, Data Destination DX”

Source, BSEND from S7	To Destination, S5 Communication Partner or Non-Siemens Device	Parameterization in the BSEND SFB, Parameter SD_1 (Source)			Parameterization in BSEND SFB, Parameter LEN (Source)	Parameterization in the BSEND SFB, Parameter R_ID (Destination)		
		S-TYPE	S-DBNO	S-Offset (byte)	Length (max. 4096 byte)	D-TYPE	D-DBNO	D-Offset (words)
Data block	Extended data block	DB	1 - *	0 - *	1 - *	DX	3-255	0-255
Memory Markers	Extended data block	MB	irrelevant	0 - *	1 - *	DX	3-255	0-255
Inputs	Extended data block	IB	irrelevant	0 - *	1 - *	DX	3-255	0-255
Outputs	Extended data block	QB	irrelevant	0 - *	1 - *	DX	3-255	0-255
Counters	Extended data block	C	irrelevant	0 - *	1 - *	DX	3-255	0-255
Times	Extended data block	T	irrelevant	0 - *	1 - *	DX	3-255	0-255

* This value is dictated by the CPU that you use

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

Specifications in the Message Frame Header of the RK 512 Protocol

The following table shows the specifications in the RK 512 message frame header.

Data destination DX:

Table 6-14 Specifications in Message Frame Header of RK 512 Protocol, “Sending Data to an S5 Communication Partner or Non-Siemens Device with RK 512, Data Destination DX”

Source, BSEND from S7	**To Destination, S5 Communication Partner or Non-Siemens Device**	**Message Frame Header, Bytes**		
		3/4 Command Type*	**5/6 D-DXNO/D-Offset**	**7/8 Number in**
Data block	Extended Data block	OD	DX/DW	words
Memory Markers	Extended Data block	OD	DX/DW	bytes
Inputs	Extended Data block	OD	DX/DW	bytes
Outputs	Extended Data block	OD	DX/DW	bytes
Counters	Extended Data block	OD	DX/DW	words
Times	Extended Data block	OD	DX/DW	words

* The source information is not transferred to the CP, so the CP always enters the ID for DB (OD) at this point.

Abbreviations:

D-DXNO	Extended destination data block number
D-Offset	Destination start address
DW	Offset in words

6.5.5 Sending Data to a Communication Partner with Dynamic Destination Definition with the RK 512 Computer Link

The figure below illustrates how data is sent to a communication partner with dynamically modifiable destination definition using RK 512.

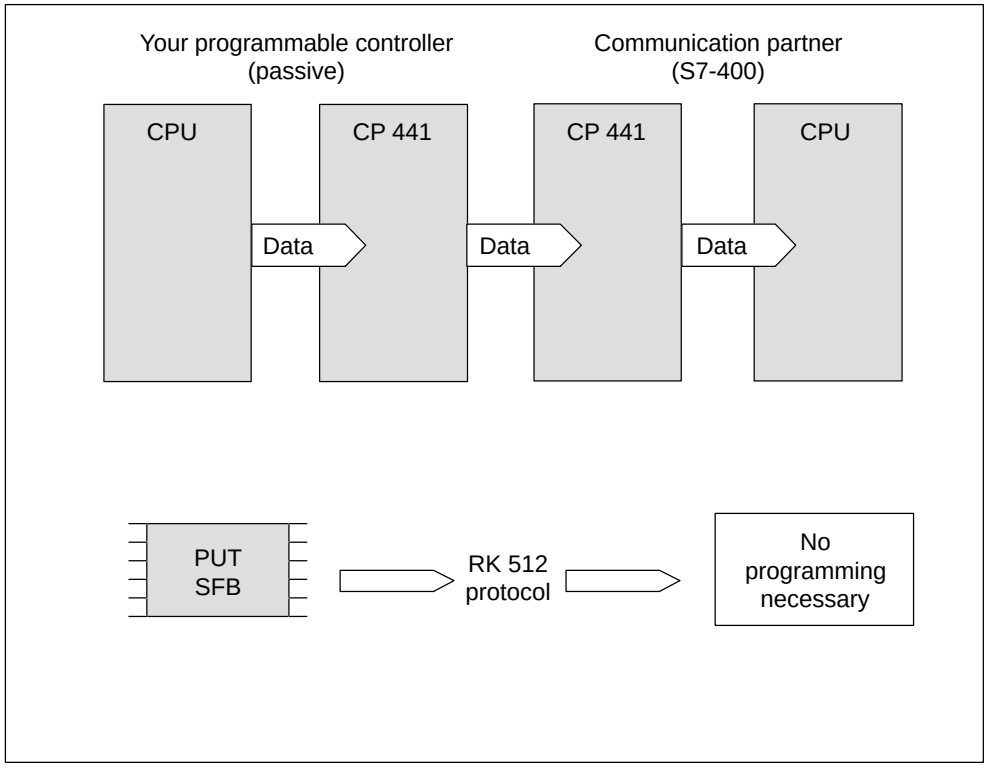


Figure 6-3 Sending Data to a Communication Partner with Dynamic Destination Definition with the RK 512 Computer Link

Data Transmission to a Communication Partner with RK 512. What To Do

To send data to a communication partner, proceed as follows:

On Your Programmable Controller

In the S7 user program of the CPU you must program the PUT system function block (SFB 15).

For the SD parameter (ANY data type) you specify the data you want to send to the partner (destination). Only even-numbered values (a maximum of 510 bytes) can be specified as the offset for the data types DB, Z and T. In the case of the data types MB, EB and AB, the maximum offset is 255 bytes.

Example: `p#DB10.DBX6.0 WORD 10`

For the ADDR parameter (ANY data type) you specify where the data is to be stored on your partner (destination). The length specified must agree with the value specified for the ADDR parameter. The parameter limits of the RK 512 protocol mean that only data blocks 1 to 255 of the partner can be reached. For the transfer of times and counters, the ”CHAR” data type must be specified for the data block in which the times are stored.

Please note that the length of the data that can be transferred is limited to 400 bytes depending on your CPU and the CPU of the communication partner.

Note that at the receiver you cannot tell when a transmission is taking place. Therefore, the receiving CP 441 cannot prevent unprocessed data from being overwritten at the receiver. The data is transferred even when the receiving S7 CPU is in STOP mode. Interprocessor communication flags are not supported for interconnection involving S5 CPs. The sending and receiving CPUs determine data consistency (CPU 412/413: 16 bytes, CPU 414/416: 32 bytes). The weaker partner in terms of consistency determines the resulting consistency length for data transfer.

You will find more information on data consistency in the reference manual on system functions and standard functions /4/.

/4/ *System Software for S7-300 and S7-400, System and Standard Functions* Reference Manual

Note

If you send data to a CP 441 by means of an S5 CP or a non-Siemens device, in this mode you must specify DB as the destination data block with the corresponding number and start address (offset) in the request block.

Interprocessor communication flags are not evaluated. The connection via which the data is forwarded from the CP 441 to the S7 CPU is selected by means of the CPU number (see Section 5.3.5).

The length of the transferable data is 400 bytes.

At the CP 441 Communication Partner

At the communication partner no programming is necessary in the S7 user program of the CPU.

Example

Example for calling SFB 15 (PUT):

STL	Explanation
CALL SFB 15, DB52 REQ := DB400.DBX0.0 ID := W#16#1000 DONE = DB400.DBX0.4 ERROR := DB400.DBX0.5 STATUS := DB400.DBW12 ADDR_1 := P#DB30.DBX 0.0 WORD 10 ADDR_2 := ADDR_3 := ADDR_4 := SD_1 := P#DB10.DBX 0.0 WORD 10 SD_2 := SD_3 := SD_4 :=	When this SFB is called, at a positive edge at bit DBX0.0, data is sent to the communication partner, where it is placed in DB30. If several data areas are sent at the same time, additional SD_i and ADDR_i pairs can be parameterized.

See the CP441 ANY demo project for instructions on how to change the send and receive parameters of the ANY data type at runtime. The demo project is in the ”Examples” STEP 7 catalog under CP441.

Request Table

The following table lists the data types which can be transmitted.

Table 6-15 Request Table for “Sending Data with RK 512 to the Communication Partner CP 441, Using PUT”

Source, PUT from S7	To Destination, Communication Partner	Parameterization in the PUT SFB, Parameter SD_1 (Source)			Parameterization in PUT SFB, Parameter LEN (Source)	Parameterization in PUT SFB, Parameter ADDR (Destination)		
		S-TYPE	S-DBNO	S-Offset (byte)	Length max. 400 bytes**	D-TYPE	D-DBNO	D-Offset (words)
Data block	Data block	DB	1 - *	0 - *	1 - *	DB	1-255	0-255
Memory Markers	Data block	MB	irrelevant	0 - *	1 - *	DB	1-255	0-255
Inputs	Data block	IB	irrelevant	0 - *	1 - *	DB	1-255	0-255
Outputs	Data block	QB	irrelevant	0 - *	1 - *	DB	1-255	0-255
Counters	Data block	C	irrelevant	0 - *	1 - *	DB	1-255	0-255
Times	Data block	T	irrelevant	0 - *	1 - *	DB	1-255	0-255

* This value is dictated by the CPU that you use
** Depending on the CPU of the communication partner, 400 bytes

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

Specifications in the Message Frame Header of the RK 512 Protocol

The following table shows the specifications in the RK 512 message frame header.

Table 6-16 Specifications in Message Frame Header of RK 512 Protocol, “Sending Data to Communication partner CP 441 with RK 512, Using PUT”

Source, PUT from S7	**To Destination, S7 Communication Partner (PUT)**	**Message Frame Header, Bytes**		
		3/4 Command Type*	**5/6 D-DBNO/D-Offset**	**7/8 Number in**
Data block	Data block	AD	DB/DW	words
Memory Markers	Data block	AD	DB/DW	bytes
Inputs	Data block	AD	DB/DW	bytes
Outputs	Data block	AD	DB/DW	bytes
Counters	Data block	AD	DB/DW	words
Times	Data block	AD	DB/DW	words

* The source information is not transferred to the CP, so the CP always enters the ID for DB (AD) at this point.

Abbreviations:

D-DBNO	Destination data block number
D-Offset	Destination start address
DW	Offset in words

6.5.6 Fetching Data from a Communication Partner with RK 512

Fetching data from a communication partner with RK 512:

The figure below illustrates how data is fetched from a communication partner.

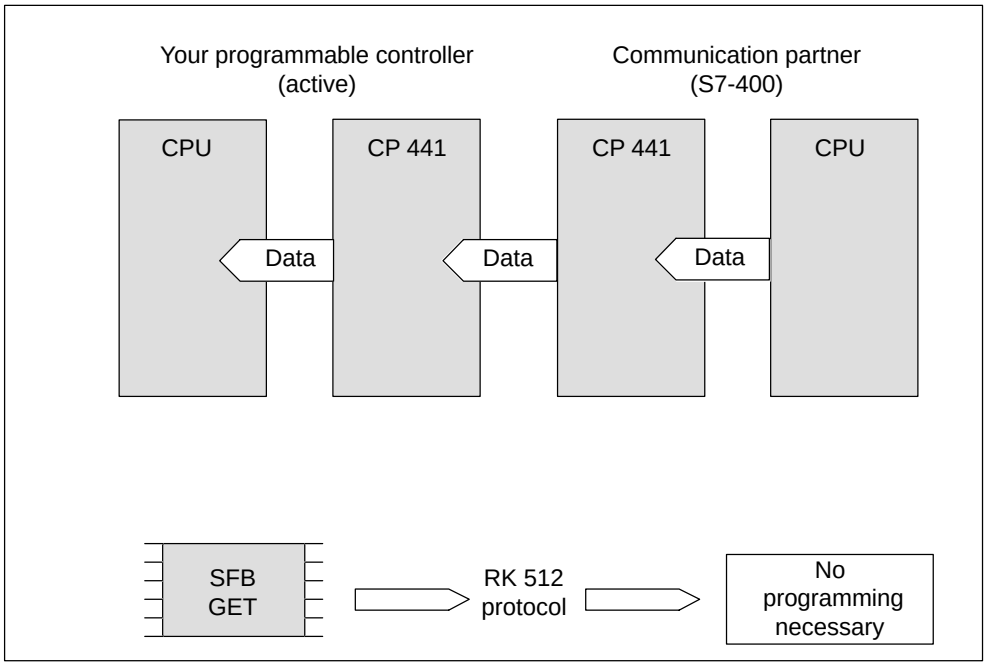


Figure 6-4 Fetching Data from a Communication Partner with the RK 512 Computer Connection

Fetching Data from a Communication Partner with RK 512. What To Do

To fetch data from a communication partner, proceed as follows:

On Your Programmable Controller

In the S7 user program of the CPU you must program the GET system function block (SFB 14).

For the ADDR parameter (ANY data type) you specify the data you want to fetch from the partner (source).The parameter limits of the RK 512 protocol mean that only data blocks 1 to 255 of the partner can be reached. Only even-numbered values (a maximum of 510 bytes) can be specified as the offset for the data types DB, Z and T. In the case of the data types MB, EB and AB, the maximum offset is 255 bytes.

Example: `p#DB10.DBX6.0 WORD 10`

For the RD parameter (ANY data type) you specify where the data is stored on your module (destination). The length specified must agree with the value specified for the ADDR parameter. For the transfer of times and counters, the ”CHAR” data type must be specified for the data block in which the times are stored.

Please note that the length of the data that can be transferred is limited to 400 bytes depending on your CPU and the CPU of the communication partner.

Please note that the partner cannot recognize when you fetch data. It is therefore not possible to prevent as yet unprocessed data being fetched from the partner. The data is transferred even when the receiving S7 CPU is in STOP mode. Interprocessor communication flags are not supported for interconnection involving S5 CPs. The sending and receiving CPUs determine data consistency (CPU 412/413: 16 bytes, CPU 414/416: 32 bytes). The weaker partner in terms of consistency determines the resulting consistency length for data transfer.

You will find more information on data consistency in the reference manual on system functions and standard functions /4/.

Note

If you fetch data from a CP 441 by means of an S5 CP or a non-Siemens device, you must specify the source data type in the request block (S5 CP). In the case of the source data type DB, you must specify the corresponding number and the start address (offset).

Interprocessor communication flags are not evaluated. The connection via which the data is fetched from the S7 CPU is selected by means of the CPU number (see Section 5.3.4).

The length of the transferable data is 400 bytes.

/4/ *System Software for S7-300 and S7-400, System and Standard Functions* Reference Manual

At the CP 441 Communication Partner

At the communication partner no programming is necessary in the S7 user program of the CPU.

Example

Example for calling SFB 14 (GET):

STL	Explanation
CALL SFB 14, DB14 REQ := DB10.DBX10.0 ID := W#16#1000 NDR := DB10.DBX10.2 ERROR := DB10.DBX10.3 STATUS := DB10.DBW20 ADDR_1 := P#DB10.DBX 6.0 WORD 10 ADDR_2 := ADDR_3 := ADDR_4 := RD_1 := P#DB100.DBX 0.0 WORD 10 RD_2 := RD_3 := RD_4 :=	When this SFB is called at a positive edge at bit DBX10.0, data is fetched from the communication partner specified under connection 1000. The data source is specified at ADDR_1: DB 10, 10 words starting at byte 6. This data is placed in DB 100, starting at byte 0. The same data length must be specified. If several data areas are fetched at the same time, additional ADDR_i and RD_i pairs can be parameterized.

Request Table

The following table lists the data types which can be transmitted.

Table 6-17 Request Table for “Fetching Data with RK 512 from Communication Partner”

Source, Fetch (GET) from Communica. Partner	**To Destination, Your S7 PLC**	**Parameterization in the GET SFB, Parameter ADDR (Source)**			**Parameterization in the GET SFB, Parameter RD (Destination)**			
		S-TYPE	**S-DBNO**	**S-Offset (byte)**	**Length max. 400 bytes**	**D-TYPE**	**D-DBNO**	**D-Off-set (byte)**
Data block	Data block	DB	* - 255	0 - 510***	1 - *	DB	1 - *	0 - *
Memory Markers	Data block	MB	irrelevant	0 - 255*	1 - *	DB	1 - *	0 - *
Inputs	Data block	IB	irrelevant	0 - 255*	1 - *	DB	1 - *	0 - *
Outputs	Data block	QB	irrelevant	0 - 255*	1 - *	DB	1 - *	0 - *
Counters	Data block	C	irrelevant	0 - 510***	1 - *	DB	1 - *	0 - *
Times	Data block	T	irrelevant	0 - 510***	1 - *	DB	1 - *	0 - *

* The maximum value is determined by the partner CPU that you use.
** Depending on your CPU and the CPU of the communication partner, 400 bytes
*** Only even-numbered values are permitted for these data types. The maximum value is determined by the partner CPU.

Abbreviations:

S-TYPE	Source type
S-DBNO	Source data block number
S-Offset	Source start address
Length	Source length
D-TYPE	Destination type
D-DBNO	Destination data block number
D-Offset	Destination start address

Specifications in the Message Frame Header of the RK 512 Protocol

The following table shows the specifications in the RK 512 message frame header.

Table 6-18 Specifications in Message Frame Header of RK 512 Protocol, “Fetching Data from the Communication Partner with RK 512”

Source, Fetch (GET) from Communication Partner	**To Destination, your S7 PLC**	**Message Frame Header, Bytes**		
		3/4 Command Type	**5/6 D-DBNO/D-Offset**	**7/8 Number in**
Data block	Data block	ED	DB/DW	words
Memory Markers	Data block	EM	DB/DW	bytes
Inputs	Data block	EI	DB/DW	bytes
Outputs	Data block	EQ	DB/DW	bytes
Counters	Data block	EC	DB/DW	words
Times	Data block	ET	DB/DW	words

Abbreviations:

S-DBNO	Source data block number
S-Offset	Source start address
DW	Offset in words

6.6 Using the System Function Blocks with the ASCII Driver

The same functions can be used for data transmission with the ASCII as with the 3964(R) procedure. In other words, the information in Section 6.4 on the 3964(R) procedure also applies to the ASCII driver.

In addition, when the ASCII driver is used with the RS 232C interface submodule, you can read and control the RS 232C secondary signals. The following describes only what you have to do to use these additional functions.

RS 232C Secondary Signals

Function blocks are available to you for reading and controlling the RS 232C secondary signals. The table below contains the function blocks of the CP 441 and describes their purpose.

Table 6-19 Function Blocks / Functions of the CP 441

FB	Meaning
FB 5 V24_STAT	The V24_STAT function allows you to read the signal states at the RS 232C interface of the CP 441.
FB 6 V24_SET	The V24_SET function allows you to set/reset the outputs at the RS 232C interface of the CP 441.

Scope of Supply and Installation

The function blocks of the CP 441, together with the parameterization interface and the programming example, are supplied on CD which comes with this manual.

The function blocks are installed together with the parameterization interface. Installation is described in Section 5.2. After installation, the function blocks are stored in the following library:
CP441

You open the library in STEP 7 SIMATIC Manager by choosing **File > Open > Library**.

For working with the function blocks, you only need to copy the required function block in your project.

Reading the RS 232C Secondary Signals

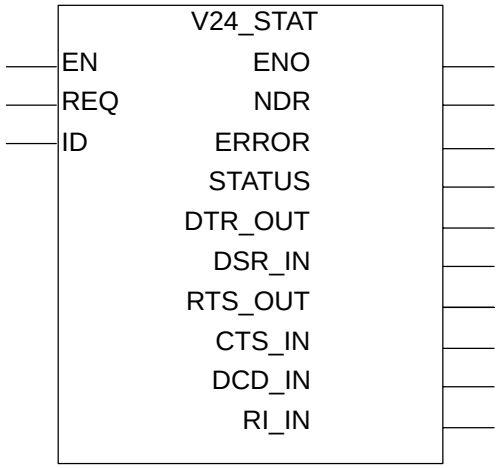
The V24_STAT FB reads the RS 232C secondary signals from an interface of the CP 441 and makes them available to the user at the module parameters. The V24_STAT FB is called statically (without conditions) in the cycle or alternatively in a time-controlled program.

The RS 232C secondary signals are updated each time the function is called (cyclic polling). You select the interface by specifying at the V24_STAT FB the local ID of one of the connections that uses this interface.

The binary result BR is not affected.

What To Do

Block call

STL representation		LAD representation
CALL	V24_STAT	
REQ:	=	
ID:	=	
NDR:	=	
ERROR:	=	
STATUS:	=	
DTR_OUT:	=	
DSR_IN:	=	
RTS_OUT:	=	
CTS_IN:	=	
DCD_IN:	=	
RI_IN:	=	

Note

The parameters EN and ENO are only present in the graphical representation (LAD or FBD). To process these parameters, the compiler uses the binary result BR.

The binary result is set to signal state ”1” if the block was terminated without errors. If there was an error, the BR is set to ”0”.

V24_STAT FB 5 Parameters

The following table lists the parameters of the 5 V24_STAT FB:

Table 6-20 FB 5 V24_STAT Parameters

Name	Type	Data Type	Comment	Permitted Values, Comment
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge	-
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner	-
NDR	VAR_OUTPUT	BOOL	Positive edge indicates that new receive data is available to the user program	-
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error	-
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning	-
DTR_OUT	OUTPUT	BOOL	**D**ata **t**erminal **r**eady, CP 441 ready for operation	(CP 441 output)
DSR_IN	OUTPUT	BOOL	**D**ata **s**et **r**eady, communication partner ready for operation	(CP 441 input)
RTS_OUT	OUTPUT	BOOL	**R**equest **t**o **s**end, CP 441 ready to send[1]	(CP 441 output)
CTS_IN	OUTPUT	BOOL	**C**lear **t**o **s**end, communication partner can receive data from the CP 441 (response to RTS = ON of the CP 441)[1]	(CP 441 input)
DCD_IN	OUTPUT	BOOL	**D**ata **C**arrier **D**etect, receive signal level	(CP 441 input)
RI_IN	OUTPUT	BOOL	**R**ing **I**ndicator, pole signal	(CP 441 input)

[1] For further information on this RS 232C secondary signal, see Section 2.2.4.

Example

Example for calling V24_STAT (FB 5):

```
STL
       CALL FB 5, DB55
           REQ      := DB30.DBX10.0
           ID       := W#16#1000
           NDR      := DB30.DBX10.1
           ERROR    := DB30.DBX10.2
           STATUS   := DB30.DBW20
           DTR_OUT  := DB30.DBX30.0
           DSR-IN   := DB30.DBX30.1
           RTS_OUT  := DB30.DBX30.2
           CTS_IN   := DB30.DBX30.3
           DCD_IN   := DB30.DBX30.4
           RI_IN    := DB30.DBX30.5
```

Controlling the RS 232CSecondary Signals

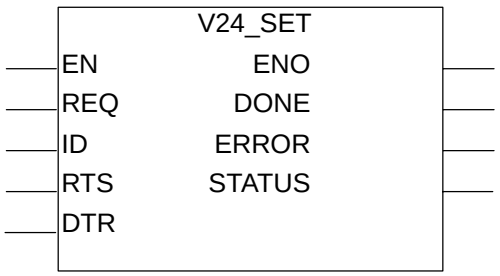
The user can use the parameter inputs of the V24_SET FB to set or reset the corresponding interface outputs of an interface of the CP 441. The V24_STAT FB is called statically (without conditions) in the cycle or alternatively in a time-controlled program.

You select the interface by specifying at the V24_SET FB the local ID of one of the connections that uses this interface.

The binary result BR is not affected.

What To Do

Block call

STL representation		LAD representation
CALL	V24_SET	
REQ:	=	
ID:	=	
DONE:	=	
ERROR:	=	
STATUS:	=	
RTS:	=	
DTR:	=	

Note

The parameters EN and ENO are only present in the graphical representation (LAD or FBD). To process these parameters, the compiler uses the binary result BR.

The binary result is set to signal state ”1” if the block was terminated without errors. If there was an error, the BR is set to ”0”.

V24_SET FB 6 Parameters

The following table lists the parameters of FB 6, V24_SET:

Table 6-21 FB 6 V24_SET Parameters

Name	Type	Data Type	Comment	Permitted Values, Comment
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge	-
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner	-
DONE	VAR_OUTPUT	BOOL	Indicates at a positive edge the error-free completion of a request	-
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error	-
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning	-
RTS	INPUT	BOOL	**R**equest **t**o **s**end, CP 441 ready to send[1]	(Control CP 441 output)
DTR	INPUT	BOOL	**D**ata **t**erminal **r**eady, CP 441 ready for operation[1]	(Control CP 441 output)

[1] For further information on the RS 232C secondary signals, see Section 2.2.4.

Example

Example for calling the 6 V24_SET FB:

```
STL
       CALL FB 6, DB56
           REQ      := DB40.DBX10.0
           ID       := W#16#1000
           DONE     := DB40.DBX10.1
           ERROR    := DB40.DBX10.2
           STATUS   := DB40.DBW20
           RTS      := DB40.DBX30.2
           DTR      := DB40.DBX30.0
```

6.7 Using the System Function Blocks with the Printer Driver

The P_PRINT system function block (SFB) is available to you for outputting message texts to a printer. PRINT transfers a process message to the CP 441, for example. The CP 441 logs the process message on the connected printer.

Outputting Message Texts

The PRINT SFB transfers a message text with up to four variables to the CP 441. The PRINT SFB is called statically (without conditions) for data transfer cyclically or in a time-controlled program.

The transmission of the message text is initiated by a positive edge at the REQ input. The format string of the message text is sent first, Variables 1 to 4 (SD_1 to SD_4) are then transferred.

The DONE output shows ”request completed without errors”. ERROR indicates whether an error has occurred. In STATUS the error number is displayed in the event of an error. If there were no errors, STATUS has the value ”0”.

What To Do

In the S7 user program of the CPU you must program the PRINT system function block (SFB 16).

You specify as the ID the connection via which PRINT is to be executed (see also Section 5.3).

The PRN_NR parameter (printer number) has no significance for the CP 441, since only one printer can be addressed via a serial interface.

You specify the format string in the FORMAT parameter. Proceed as follows:

1. You must store the format string in a separate data block. If this block does not exist, you must define it.

   Make an entry in the declaration field of the DB to output a message text as in the following example:

   ```
   -- Name: Anna
   -- Type: STRING
   -- Start value: 'Content of message text: %N'
   ```

2. The DB for the format string can only be specified symbolically.

   After the DB has been saved, you should therefore set symbolic representation under "View" in the STL compiler and then enter a symbol for the data block in the symbol table under "Options" (e.g. print_db). As the address and data type, you enter in the symbol table the DB in which the format strings are stored.

3. Switch the view of your module in which you call "PRINT" to symbolic representation as well, and enter "print_db".Anna as the FORMAT parameter at the system function block.

Up to four variables can be specified at SD_1 to SD_4 for printer output (the number of the message text to be output is to be specified in the example).

Example

```
CALL SFB 16, DB116                  PRINT SFB call in an FB
REQ     :=DB60.DBX100.0
ID      :=W#16#1000
DONE    :=DB60.DBX100.1
ERROR   :=DB60.DBX100.2
STATUS  :=DB60.DBW110
PRN_NR  :=DB60.DBB120
FORMAT  :="print_db".Anna           Byte 200 in DB 60 contains the message
SD_1    :=p#DB60.DBX200.0 Byte 1    text number
SD_2    :=
SD_3    :=
SD_4    :=
```

Important Notes

Note that the volume of data consisting of format string and variables that can be transferred is limited to 400 bytes.

Note that if changes are made subsequently to the format string, you cannot enter them in the string under ”Initial value”; instead, you have to select the data view under ”View” and enter the changed format string under ”Initial value”.

Note that no string can be transferred in the case of the C (character) representation type. Use the S representation type for strings.

Note that you cannot specify the string directly at the SD_i parameter in the case of the S (string) representation type. As in the case of the format string, you have to store the string in a data block and address it symbolically at the SD_i parameter.

6.8 Summary

The tables below summarize the following information on the protocols:

- The possible communication types
- The system function blocks used
- Whether or not overwrite protection is possible and
- The maximum length of transmittable data

ASCII Driver and 3964(R)

The following applies for the 3964(R) procedure:

Communication Type	CP 441 Sending Data to CP 441 Communication Partner	
	Programming at Sender and Receiver	Programming at Sender Only
System function block at CP 441	BSEND	BSEND
System function block at CP 441 communication partner	BRCV	none (receive mailbox)
Overwrite protection	yes	no
Maximum length of transmittable data	4 KB	450 bytes*, receiving CPU

* The receiving CP 441 restricts the maximum length of transmittable data. If you use a different communication partner, you can transfer up to 4 kilobytes.

RK 512, Sending Data

The following applies when sending data via the RK 512 computer connection:

Communication Type	CP 441 Sending Data to CP 441 Communication Partner		
	Programming at Sender and Receiver/ (Type DX)	Programming at Sender Only (Type DB)	Programming at Sender Only
System function block at CP 441	BSEND	BSEND	PUT
System function block at CP 441 communication partner	BRCV	none	none
Overwrite protection	yes	no	no
Maximum length of transmittable data	4 KB	450 bytes*, receiving CPU	400 bytes
Dynamically changeable destination definition	no	no	yes

* The receiving CP 441 restricts the maximum length of transmittable data. If you use a different communication partner, you can transfer up to 4 kilobytes.

RK 512, Fetching Data

The following applies when fetching data via the RK 512 computer connection:

Communication Type	CP 441 Sending Data to CP 441 Communication Partner
System function block at CP 441	GET
System function block at CP 441 communication partner	none
Overwrite protection	no
Maximum length of transmittable data	400 bytes
Dynamically changeable source definition	yes

Printer Output

The following applies to the output of message texts on a printer:

Communication Type	CP 441 Sending Data to the Printer
System function block at CP 441	PRINT
Maximum length of transmittable data (format string and variables)	400 bytes

Start-up Characteristics and Operating Mode Transitions of the CP 441

7

7.1 Start-up Characteristics of the CP 441

The CP 441 start-up is divided into two phases:

- Initialization (CP 441 in POWER ON mode)
- Parameterization

Initialization

As soon as the CP 441 is connected to the power supply, the firmware on the CP 441 is prepared for operation after a hardware test program has been executed.

Parameterization

During parameterization the CP 441 receives the module parameters which have been assigned to the current slot.

The CP 441 is now ready for operation.

Please Note

Please note the following for the start-up behavior of the CP 441:

Note

After power on, the CP 441 requires several seconds for initialization and hardware and memory testing before it is ready for operation. The parameterization attempts made by the CPU during this phase are aborted and an error is entered in the diagnostic buffer:

”SDB processing error, error class 1”, and the SDB is identified, and ”Parameterization error on module parameter assignment”, and the SDB is identified.

As soon as the module test has been completed, parameterization is performed by the CPU without errors.

SFB calls in the user program will result in an error as long as the CP 441 has not been parameterized.

7.2 Operating Mode Transitions of the CP 441

Once the CP 441 has been started up, all data is exchanged between the CPU and the CP 441 by means of the system function blocks. The operating mode transition behavior of the CP 441 depends on the operating mode of the CPU.

CPU STOP

Communication direction CPU > CP:

Communication between the CPU and CP 441 still takes place even when the CPU is in STOP mode. When the CPU is in STOP mode, the initiated SFB requests (e.g. BSEND) on the CPU are fully executed and the data is transferred in its entirety to the CP 441 and forwarded to the communication partner.

Communication direction CP > CPU:

The message frames are received by the CP 441. Any attempt to send the data on to the CPU is aborted with an error message.

CPU RUN

The CP 441 does not recognize how the CPU switches to RUN mode (cold restart or restart following CPU STOP). The CP 441 behaves identically in both cases.

Cold Restart of the CPU

When the CPU is restarted cold, the SFB requests on the CPU are reset, that is all current requests between the CPU and the CP are automatically aborted. The requests on the CP are deleted.

Restart of the CPU

When the CPU is restarted, the SFB requests continue to be processed.

Diagnostics with the CP 441

8

8.1 Diagnostics Functions of the CP 441

The diagnostics functions of the CP 441 enable you to quickly localize any errors which occur. The following diagnostics options are available:

- Diagnosis via the display elements of the CP 441
- Diagnosis via the STATUS output of the system function blocks
- Diagnosis via the error-signaling area SYSTAT
- Diagnosis via the error numbers in the response message frame
- Diagnosis via the diagnostic buffer of the CP 441
- Diagnostics alarm

Display Elements (LED)

The display elements show the operating mode or possible error states of the CP 441. The display elements give you an initial overview of any internal or external errors as well as interface-specific errors (see Chapter 8.2).

Section 5.7 contains information on LED displays which can occur when you load firmware updates.

STATUS Output of the SFBs

Every system function block has a STATUS output for error diagnostics. Reading the STATUS output of the system function blocks gives you general information on errors which have occurred during communication between the CP 441 and the assigned CPU. You can analyze the STATUS parameter in the user program (see Chapter 8.3).

SYSTAT Error Message Area

The programming of the STATUS system function block in the user program allows you to obtain the status of an interface. By reading SYSTAT you obtain detailed information on errors/events that have occurred during communication between the CP 441, the assigned CPU and the communication partner connected at this interface (see Section 8.4).

Error Numbers in the Response Message Frame

If you are working with the RK 512 computer connection and there is a SEND or GET message frame error at the communication partner, the communication partner sends a response message frame with an error number in the 4th byte (see Section 8.5).

Diagnostic Buffer of the CP 441

All the errors/events in the SYSTAT error-signaling area of the CP 441 are also entered in the diagnostic buffer of the CP 441.

In the same way as with the diagnostic buffer of the CPU, you can also use the STEP 7 information functions on the programming device to display the information in the CP diagnostic buffer (see Section 8.6).

Diagnostics alarm

The CP 441 can trigger a diagnostics alarm on the CPU assigned to it. The CP 441 provides 4 bytes of diagnostics information on the S7-400 backplane bus. This information is analyzed via the user program (OB 82) or using a programming device to read from the CPU diagnostics buffer.

The CP 441 also enters diagnostic events that trigger a diagnostics alarm in its diagnostic buffer.

When a diagnostics alarm event occurs, the EXTF LED (red) lights up.

8.2 Diagnosis via the Display Elements of the CP 441

The display elements of the CP 441 provide information on the CP 441. The following display functions are distinguished:

- **Special LEDs**
  - TXD — Sending active; lights up when the CP is sending user data via the interface.
  - RXD — Receiving active; lights up when the CP is sending user data via the interface.
- **Group fault LEDs**
  - INTF — Internal fault
  - EXTF — External fault
- **Interface fault LED**
  - FAULT — Interface error

Note

Section 5.7 contains information on LED displays which can occur when you load firmware updates.

Error Messages of the Display Elements

The table below describes the error messages of the display elements.

Table 8-1 Error Messages of the CP 441 Display Elements

Error LED	Error Description	Remedy
INTF comes on	CP 441 signals internal fault, e.g. hardware fault or software error.	Program the STATUS SFB for detailed information or read the diagnostic buffer of the CP 441.
EXTF comes on	CP 441 signals external fault, e.g. break on the line.	Program the STATUS SFB for detailed information or read the diagnostic buffer of the CP 441.
FAULT off	Interface ready for operation or interface submodule not plugged in.	–
FAULT flashing slowly	Interface initialized and ready for operation but communication via S7-400 backplane bus not possible.	Check configuration and connection configuration for incorrect entries (e.g. slot, ID no., etc.).
FAULT flashing fast	Invalid parameter(s), or wrong or faulty interface submodule inserted (module and interface parameters not compatible).	Check the parameter settings in the *CP441: Point-to-Point Communication, Parameter Assignment* parameterization interface and/or the interface submodule.
FAULT lit up	No interface parameters or serious fault in submodule (hardware).	Perform parameterization with the *CP441: Point-to-Point Communication, Parameter Assignment* parameterization interface or check the interface submodule.

8.3 Diagnostics Messages of the System Function Blocks

Every system function block has a STATUS parameter for error diagnostics. The STATUS message numbers always have the same meaning, irrespective of which system function block is used.

The tables below are copied from the STEP 7 manual and represent only the current status. Refer to the original tables if you discover discrepancies.

Displaying and Evaluating the STATUS Output

You can display and evaluate the STATUS output of the system function blocks using the STEP 7 variable table.

For further information on using the variable table, see the STEP 7 manual /3/.

Messages in the STATUS Output of the SFBs

The tables below list the messages of the STATUS parameter.

Error Information for SFB 12

Table 8-2 contains all the SFB 12 specific error information that can be output in the ERROR and STATUS parameters.

Table 8-2 Error Information for SFB 12 ”BSEND”

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New request ineffective because previous request not yet completed
1	1	Communication problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (e.g. cable, CPU off)
1	2	Negative acknowledgment from partner SFB. The function cannot be executed.
1	3	R_ID is not known on the communication link identified by ID, or receive block not yet called.
1	4	Error in send area pointer SD_1 regarding data length or data type, or the LEN parameter received the value 0 or error in receive area pointer RD_1 of SFB 13 ”BRCV”
1	5	The reset request has been completed.
1	6	The status of the partner SFB is DISABLED (value of EN_R is 0)
1	7	Status of partner SFB is not correct (receive block not called since last data transfer).
1	8	Access to remote object in user memory denied.

/3/ *Configuring Hardware and Communication Connections STEP 7 V5.1*, Manual

Table 8-2 Error Information for SFB 12 ”BSEND”, continued

ERROR	STATUS (decimal)	Explanation
1	10	Access to local user memory not possible (for example access to deleted DB)
1	12	The SFB call • specified an Instance DB which does not belong to SFB 12 • specified a global DB instead of an instance DB. • did not have an Instance DB (programming device: load a new instance DB)
1	18	R_ID already exists in the link.
1	20	Insufficient main memory

Error Information for SFB 13

Table 8-3 contains all the SFB 13 specific error information that can be output in the ERROR and STATUS parameters.

Table 8-3 Error Information for SFB 13 ”BRCV”

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New request ineffective because previous request not yet completed
0	17	Warning: block receiving asynchronous data.
1	1	Communication problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (e.g. cable, CPU off, CP stopped)
1	2	The function cannot be executed.
1	4	Error in receive area pointer RD_1 regarding data length or data type (data block sent is longer than receive area).
1	5	Reset request received, incomplete transfer.
1	8	Access to remote object in user memory denied.
1	10	Access to local user memory not possible (for example access to deleted DB)
1	12	The SFB call • specified an Instance DB which does not belong to SFB 13 • specified a global DB instead of an instance DB. • did not have an Instance DB (programming device: load a new instance DB)
1	18	R_ID already exists in the link.
1	20	Insufficient main memory

Error Information for SFB 14

Table 8-4 contains all the SFB 14 specific error information that can be output in the ERROR and STATUS parameters.

Table 8-4 Error Information for SFB 14 ”GET”

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New request ineffective because previous request not yet completed
1	1	Communication problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (e.g. cable, CPU off)
1	2	Negative acknowledgment from partner device. The function cannot be executed.
1	4	Error in receive area pointers RD_i regarding data length or data type
1	8	Access error at the partner CPU
1	10	Access to local user memory not possible (for example access to deleted DB)
1	12	The SFB call • specified an Instance DB which does not belong to SFB 14 • specified a global DB instead of an instance DB. • did not have an Instance DB (programming device: load a new instance DB)
1	20	Insufficient main memory

Error Information for SFB 15

Table 8-5 contains all the SFB 15 specific error information that can be output in the ERROR and STATUS parameters.

Table 8-5 Error Information for SFB 15 ”PUT”

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New request ineffective because previous request not yet completed
1	1	Communication problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (e.g. cable, CPU off)
1	2	Negative acknowledgment from partner device. The function cannot be executed.
1	4	Error in send area pointers SD_i regarding data length or data type
1	8	Access error at the partner CPU
1	10	Access to local user memory not possible (for example access to deleted DB)
1	12	The SFB call • specified an Instance DB which does not belong to SFB 15 • specified a global DB instead of an instance DB. • did not have an Instance DB (programming device: load a new instance DB)
1	20	Insufficient main memory

Error Information for SFB 16

Table 8-6 contains all the SFB 16 specific error information that can be output in the ERROR and STATUS parameters.

Table 8-6 Error Information for SFB 16 ”PRINT”

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New request ineffective because previous request not yet completed
1	1	Communication problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (e.g. cable, CPU off)
1	2	Negative acknowledgment from printer. The function cannot be executed.
1	3	PRN_NR is not known on the communication link identified by ID.
1	4	Error in in/put parameter FORMAT or in send area pointers SD_i regarding data length or data type.

Table 8-6 Error Information for SFB 16 ”PRINT”, continued

ERROR	STATUS (decimal)	Explanation
1	6	The status of the remote printer is OFFLINE.
1	7	The status of the remote printer is incorrect (e.g. ”Paper out”)
1	10	Access to local user memory not possible (for example access to deleted DB)
1	13	Error in the FORMAT in/out parameter
1	20	Insufficient main memory

Error Information for SFB 22

Table 8-7 contains all the SFB 22 specific error information that can be output in the ERROR and STATUS parameters.

Table 8-7 Error Information for SFB 22 ”STATUS”

ERROR	STATUS (decimal)	Explanation
0	11	Warning: New request ineffective because previous request not yet completed
1	1	Communication problems, for example: • Connection description not loaded (local or remote) • Connection interrupted (e.g. cable, CPU off)
1	2	Negative acknowledgment from partner device. The function cannot be executed.
1	4	Error in PHYS, LOG or LOCAL regarding data length or data type
1	8	Access to remote object denied.
1	10	Access to local user memory not possible (for example access to deleted DB)
1	12	The SFB call • specified an Instance DB which does not belong to SFB 22 • specified a global DB instead of an instance DB. • did not have an Instance DB (programming device: load a new instance DB)
1	20	Insufficient main memory

8.4 Diagnosis via the Error-Signaling Area SYSTAT

The SYSTAT error message area is a data area on the CP 441 by means of which you can query the device status of an interface using the STATUS system function block.

Error-Signaling Area SYSTAT

The error-signaling area SYSTAT is an error area which is available for every interface (ID number). The SYSTAT records all errors/events which can occur during data transmission on an interface.

Note

Because the STATUS request is executed asynchronously to the rest of the requests running at an interface, an SFB with a specific R_ID cannot be assigned to the error messages. This means that although SYSTAT can display which errors have occurred at an interface, it cannot show which SFB call (R_ID number) triggered the error.

Errors/Events

The SYSTAT messages are entered in bytes 2 to 15 of the LOCAL parameter when the STATUS SFB is called. In addition to the error byte (byte 2), the first six errors/events are displayed. Error event 1 is the oldest.

If other error events occur, these cannot be reported until the ”old” entries are deleted. The error-signaling area must therefore be deleted in good time. This is done when the STATUS SFB is called.

The errors/events are stored as follows:

- Byte 0 Operating mode of CP (02H for RUN, 05H for faulty)
- Byte 1 Reserved
- Byte 2 Bit 0 -F Error entered in SYSTAT
Bit 1 -U Error overflow
Bit 2 -B Break
- Byte 3 Reserved
- Bytes 4/5 Event 1
- Bytes 6/7 Event 2
- Bytes 8/9 Event 3
- Bytes 10/11 Event 4
- Bytes 12/13 Event 5
- Bytes 14/15 Event 6

Example for calling SFB 22 (STATUS)

```
CALL SFB 22, DB22                    STATUS SFB call in an FB
REQ     :=DB450.DBX0.0
ID      :=W#16#1000                  The errors/events are entered in bytes
NDR     :=DB450.DBX0.4               20–35 of DB450 following a rising edge
ERROR   :=DB450.DBX0.5               of the REQ parameter. SFB22 should not
STATUS  :=DB450.DBW12                be run unless an error occurs in data
PHYS    :=p#DB450.DBX16.0 Byte 2     transfer. For example, the error bit of
LOG     :=p#DB450.DBX18.0 Byte 2     a BSEND (ERROR parameter) can be used
LOCAL   :=p#DB450.DBX20.0 Byte 16    as the request bit for the STATUS
                                     (REQ). Calling SFB22 automatically
                                     clears the error-signaling area of
                                     SYSTAT.
```

Numbering Scheme

The numbering scheme for the events in the error-signaling area SYSTAT has the following structure:

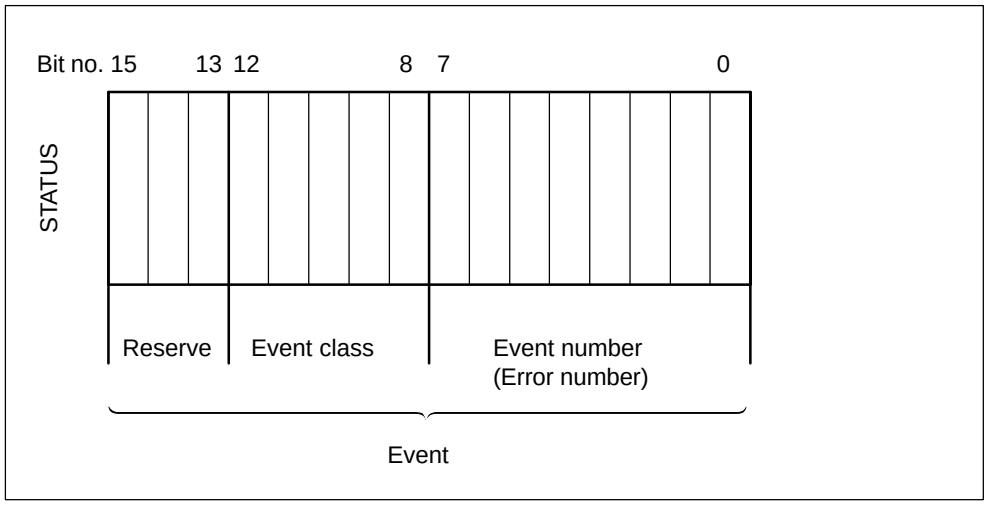


Figure 8-1 Numbering Scheme for Events in Error-Signaling Area SYSTAT

Event Classes

The table below describes the various event classes and numbers. Event classes and event numbers are listed with hexadecimal notation.

Table 8-8 Event Classes and Event Numbers

Event Class 1 (01H): "Hardware fault on CP"		
Event No.	**Event Text**	**Remedy**
(01)01H	Fault while testing operating system EPROM of CP	CP defective; replace CP.
(01)02H	RAM test of CP errored	
(01)03H	Request interface of CP defective	
(01)04H	No interface submodule inserted	Insert suitable interface submodule for CP.
(01)05H	• Parameter memory defective • Interface submodule unplugged after parameterization	Exchange CP or insert suitable interface submodule for CP.
(01)10H	Fault in CP firmware	Switch module off and on again. If necessary, replace module.
Event Class 2 (02H): "Initialization error"		
Event No.	**Event Text**	**Remedy**
(02)01H	No parameters Parameter memory empty or has unknown contents	Load interface parameters.
(02)08H	Parameterization and interface submodule incompatible	Check parameters set for interface submodule.
(02)0FH	Invalid parameterization detected at start of parameterized communication. Interface could not be parameterized	Correct invalid parameterization and restart.
(02)10H	Total baud rate exceeded	Reduce the baud rates of the two interfaces so that the total baud rate is not exceeded.
(02)11H	Total baud rate back in range	Total baud rate was out of range. It is now back in range after the excursion.

Table 8-8 Event Classes and Event Numbers, continued

Event Class 3(03H): ”Error in parameterization of CFB”		
Event No.	**Event Text**	**Remedy**
(03)01H	Invalid or no source/destination data type Invalid area (start address, length) Invalid or no DB (e.g. DB 0), or Other data type invalid or missing	Check parameterization on CPU and CP, and correct if necessary. **RK 512 only:** Partner returns invalid parameters in message frame header. Check parameterization on CPU and CP; possibly create block. See request tables for valid data types. **RK 512 only:** Partner returns incorrect parameters in message frame header.
(03)02H	Area too short	Check parameterization on CPU and CP; possibly check block/area. **RK 512 only:** Partner returns incorrect parameters in message frame header.
(03)03H	Area cannot be accessed	Check parameterization on CPU and CP. See request tables for valid start addresses and lengths. **RK 512 only:** Partner returns incorrect parameters in message frame header.
Event Class 4 (04H): ”CP detected error in data traffic CP – CPU”		
Event No.	**Event Text**	**Remedy**
(04)01H	CP cannot accept requests (overload)	In your user program, reduce number of requests called concurrently for CP.
(04)02H	CP cannot process request type	Check whether system function blocks you have called in user program are valid for CP.
(04)03H	Incorrect, unknown or illegal data type	Check program, e.g. for incorrect parameterization of SFB.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(04)07H	Error during data transmission between CPU and CP	If fault indication persists, check whether SFBs you have called in your user program are parameterized correctly. If error is indicated immediately after POWER ON, no connection has yet been set up to the CPU. In the case of the ASCII driver and the 3964(R) procedure, the receiving CP 441 re-attempts data transfer until the data is transmitted to the CPU. In the case of RK 512 the request receives a negative acknowledgment and must be repeated in the user program. If fault indication is sporadic in the course of data transfer, the CPU is temporarily unable to accept data. In the case of the ASCII driver and the 3964(R) procedure, the receiving CP 441 re-attempts data transfer until the data is transmitted to the CPU. In the case of RK 512 the request receives a negative acknowledgment and must be repeated in the user program. You can rectify the situation by calling the BRCV SFB more frequently in your user program.
(04)08H	Error during data transmission between CPU and CP (reception) • CPU is temporarily overloaded, request queued for repetition • CPU data areas temporarily unavailable for access, for example because receive block is called too infrequently. • CPU data areas temporarily unavailable for access, for example because receive block is temporarily locked (EN=false).	• Reduce number of communication calls • Call the receive block more frequently • Check whether the receive block is disabled for too long
(04)09H	Data cannot be received. Error during data transmission between CPU and CP (reception) Data cannot be received. Request is canceled in 10 seconds following multiple attempts, because • Receive block is not called • Receive block is disabled • access to CPU data areas denied • CPU data area too short	• Check whether your application runs the receive block. • Check whether the receive block is disabled. • Check that the data area to which the data is to be transferred is available. • Check the length of the data area.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(04)0BH	Error during data transmission between CPU and CP, because • no connection has been configured • no receipt possible via configured connection	 • Configure the connection in “NetPro” • Enter in “NetPro” under “Object Properties Connection) as communication direction: 2: Partner —> Local or 3: Local <—> Partner
Event Class 5 (05H): ”Error while processing CPU request”		
Event No.	**Event Text**	**Remedy**
(05)01H	Current request aborted as a result of CP restart.	No remedy is possible at POWER ON. When reparameterizing the CP from the programming device, before writing an interface you should ensure there are no more requests running from the CPU.
(05)02H	Request not permitted in this operating mode of CP (e.g. device interface not parameterized).	Parameterize the device interface.
(05)03H	Wrong time, incorrect format	Check the time parameters.
(05)05H	**With printer driver only:** System data block with message texts not available on the CP	Use the parameterization software to configure the message text, and then carry out a restart.
(05)06H	**With printer driver only:** Message text not available	Use the parameterization software to configure the message texts, and then carry out a restart.
(05)07H	**With printer driver only:** Message text too long	Change the message text so that it has no more than 150 characters (or no more than 250 characters if it contains variables).
(05)08H	**With printer driver only:** Too many conversion statements	You have configured more conversion statements than variables. The conversion statements without associated variables are ignored.
(05)09H	**With printer driver only:** Too many variables	You have configured more variables than conversion statements. Variables for which there is no conversion statement are not output.
(05)0AH	**With printer driver only:** Unknown conversion statement	Check the conversion statement. Undefined or unsupported conversion statements are replaced in the printout with ******.
(05)0BH	**With printer driver only:** Unknown control statement	Check the control statement. Undefined or incorrect control statements are not executed. The control statement is not output as text either.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(05)0CH	**With printer driver only:** Conversion statement not executable	Check the conversion statement. Conversion statements that cannot be executed are output in the expression in accordance with the defined width and the valid remainder of the conversion statement or the standard representation with * characters.
(05)0DH	**With printer driver only:** Width in conversion statement too small or too great	Correct the specified width of the variable in the conversion statement on the basis of the variable's maximum number of characters in text-based representation types (A, C, D, S, T, Y, Z). Only as many characters as will fit in the specified width appear in the printout; the text is truncated to this width. In all other cases, * characters are output corresponding to the width.
(05)0EH	**With 3964(R) and ASCII driver only:** Invalid message frame length	The message frame is > 4096 bytes in length. Select a smaller message frame length.
(05)0FH	Number of requests that can be processed simultaneously too great	Change your STEP 7 program so that fewer requests can run simultaneously.
(05)10H	Area occupied (resource)	Repeat the request.
(05)11H	Length not permissible for this request type	Divide up the data to be transmitted into several requests.
(05)12H	**RK 512 only:** Mismatch between SFB's source and destination parameters.	Obtain the permissible values from the request tables.
(05)13H	Data type error (DB ...): Unknown or impermissible data type (e.g. DE) **RK 512 only:** Mismatch between SFB's source and destination data types.	Obtain the permissible data types and their combinations from the request tables.
(05)14H	Specified start addresses too high for desired data type, or start address or DB/DX number too low.	Obtain from the request tables the permissible start addresses and DB/DX numbers that can be specified in the program.
(05)15H	**RK 512 only:** Wrong bit number specified for coordination flag.	Permissible bit numbers: 0 to 7
(05)16H	**RK 512 only:** Specified CPU too high.	Permissible CPU numbers: none, 1, 2, 3 or 4
(05)17H	Transmission length > 4 KB too great for CP or too short for interface parameters.	Split the request up into several shorter requests.
(05)18H	Transmission length at sending too great (> 4 KB)	**RK 512 only:** Obtain the permissible lengths from the request tables. Split the request up into several shorter requests.
(05)19H	CP in wrong mode for PLC request	Check whether the addressed interface is parameterized.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(05)1AH	**RK 512 only:** Error sending a command message frame An associated procedure number has just been entered in STATUS.	See the remedy for the previous error number.
(05)1BH	**With printer driver only:** Precision invalid	Correct the specified precision in the conversion statement. The precision is always preceded by a period to identify it and separate it from the width (e.g. ".2" to output a decimal a point and 2 places after the decimal point). Precision is only relevant to representation types F, R A, and D. In all other cases, it is ignored.
(05)1CH	**With printer driver only:** Variable invalid (Variable length incorrect/incorrect type)	Correct the specified send variable. Table 2-19 indicates the data types possible for each representation type.
Event Class 6 (06H): **"Error processing a partner request" only with RK512**		
Event No.	**Event Text**	**Remedy**
(06)01H	Error in 1st command byte (not 00 or FFH)	Header layout error at partner. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)02H	Error in 3rd command byte (not A, 0 or FFH)	Header layout error at partner. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)03H	Error in 3rd command byte in the case of continuation message frames (command not as for 1st message frame)	Header layout error at partner. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)04H	Error in 4th command byte (command letter incorrect)	Header layout error at partner or a command combination has been requested that is not permitted at the CP. Check the permissible commands. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)05H	Error in 4th command byte in the case of continuation message frames (command not as for 1st message frame)	Header layout error at partner. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)06H	Error in 5th command byte (DB number not permissible)	Obtain from the request tables the permissible DB numbers, start addresses or lengths.
(06)07H	Error in 5th or 6th command byte (start address too high)	Obtain from the request tables the permissible DB numbers, start addresses or lengths.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(06)08H	Error in 7th or 8th command byte (impermissible length)	Obtain from the request tables the permissible DB/DX numbers, start addresses or lengths.
(06)09H	Error in 9th and 10th command byte (coordination flag for this data type impermissible or bit number too high)	Header layout error at partner. Find out from the request tables when a coordination flag is permitted.
(06)0AH	Error in 10th command byte (CPU number not permitted)	Header layout error at partner.
(06)0BH	SEND message frame was longer/shorter than expected (more/less data received than announced in message frame header).	Correction required at the partner
(06)0CH	GET command message frame received with user data	Correction required at the partner
(06)0DH	CP received message frame during impermissible mode. • Receive connection between CPU and CP not set up or not yet correctly set up • CP startup not fully completed • Interface is currently being reparameterized	• Check whether the addressed connection is (correctly) parameterized. • This error message can occur only during CP startup. Repeat the request. • This is a temporary error. Repeat the request.
(06)0EH	Synchronous fault of partner • New (continuation) command message frame received before response message frame sent. • 1st command message frame expected and continuation message frame came. • Continuation command message frame expected and 1st message frame came	This error may be reported after your own programming device is restarted in the case of long message frames or when the partner is restarted. These cases represent normal system start-up behavior. The error can also occur during operation as a consequence of error statuses only recognized by the partner. Otherwise, you have to assume an error on the part of the partner device. The error may not occur in the case of requests < 128 bytes.
(06)0FH	DB locked by coordination function	In local program: After processing of the last transmission data, enable the last receive block with ”EN”. In partner program: Repeat the request
(06)10H	Message frame received too short (length < 4 bytes in the case of continuation or response message frames or < 10 bytes in the case of command message frames)	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)11H	Message frame length and length specified in message frame header are not the same.	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(06)12H	Error sending the (continuation) response message frame. An associated procedure error number has been entered in STATUS immediately beforehand.	See remedy for the error number entered immediately beforehand in STATUS.

Table 8-8 Event Classes and Event Numbers, continued

Event Class 7 (07H): "Send error"		
Event No.	**Event Text**	**Remedy**
(07)01H	Transmission of the first repetition: • An error was detected during transmission of the message frame, or • The partner requested a repetition by means of a negative acknowledgment code (NAK).	A repetition is not an error, but it can be an indication that there are disturbances on the transmission line or that the partner device is behaving incorrectly. If the message frame still has not been transmitted after the maximum number of repetitions, an error number describing the first error that occurred is output.
(07)02H	Error during connection setup: • After STX was sent, NAK or another code (except DLE or STX) was received, or • The response came too early, or • An initialization conflict occurred	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(07)03H	Acknowledgment delay time exceeded: After STX was sent, no response came from partner within acknowledgment delay time.	The partner device is too slow or not ready to receive, or there is a break in the transmission line, for example. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(07)04H	Termination by partner: One or more codes were received from the partner during sending	Check whether the partner is also showing an error, possibly because not all transmission data has arrived (e.g. due to a break in the send line) or due to serious faults or because the partner device has malfunctioned. If necessary, use an interface test device switched into the transmission line for this purpose.
(07)05H	Negative acknowledgment during sending	Check whether the partner is also showing an error, possibly because not all transmission data has arrived (e.g. due to a break in the send line) or due to serious faults or because the partner device has malfunctioned. If necessary, use an interface test device switched into the transmission line for this purpose.
(07)06H	Error at end of connection: • Partner rejected message frame at end of connection with NAK or a random string (except for DLE), or • Acknowledgment code (DLE) received too early.	Check whether the partner is also showing an error, possibly because not all transmission data has arrived (e.g. due to a break in the send line) or due to serious faults or because the partner device has malfunctioned. If necessary, use an interface test device switched into the transmission line for this purpose.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(07)07H	Acknowledgment delay time exceeded at end of connection or response monitoring time exceeded after send message frame: After connection release with DLE ETX, no response received from partner within acknowledgment delay time.	Partner device faulty or too slow. If necessary, use an interface test device switched into the transmission line to check.
(07)08H	**ASCII Driver and printer driver only:** Waiting time for XON or CTS = ON has expired.	The communication partner has a fault, is too slow or is switched off-line. Check the communication partner or, if necessary, change the parameterization.
(07)09H	Connection setup not possible. Number of permitted setup attempts exceeded.	Check the interface cable or the transmission parameters.
(07)0AH	The data could not be transmitted. The permitted number of transfer attempts was exceeded.	Check the interface cable or the transmission parameters.
Event Class 8 (08H): ”Receive error”		
Event No.	**Event Text**	**Remedy**
(08)01H	Expectation of the first repetition: An error was detected on receipt of a message frame, and the CP requests a repetition by means of negative acknowledgment (NAK) at the partner.	A repetition is not an error, but it can be an indication that there are disturbances on the transmission line or that the partner device is behaving incorrectly. If the message frame still has not been transmitted after the maximum number of repetitions, an error number describing the first error that occurred is output.
(08)02H	Error during connection setup: • In idle mode, one or more random codes (other than NAK or STX) were received, or • after an STX was received, partner sent more codes without waiting for response DLE. After POWER ON of the partner: • While partner is being activated, CP receives an undefined code.	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(08)05H	Logical error during receiving: After DLE was received, a further random code (other than DLE or ETX) was received.	Check whether partner DLE in message frame header and in data string is always in duplicate or the connection is released with DLE ETX. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(08)06H	Character delay time exceeded: • Two successive characters were not received within character delay time, or • 1st character after sending of DLE during connection setup was not received within character delay time.	Partner device faulty or too slow. Use an interface test device switched into the transmission line to check.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(08)08H	Block check code (BCC) error **(only in the case of RK 512 with the 3964R procedure and the 3964R procedure)** Internally calculated value of BCC does not match BCC received by partner at end of connection.	Check whether there is a serious problem with the connection. In this case, error codes of the event class 8/event number 12 sometimes occur. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(08)0AH	There is no free receive buffer available: After receipt of STX, there was no empty receive buffer available to the procedure at connection setup and after an additional waiting time	The function block for receiving must be called more frequently in the user program.
(08)0CH	Transmission error: Transmission error (parity error, stop bit error, overflow error) detected. If faulty character is received in idle mode, the error is reported immediately so that disturbances on the transmission line can be detected early. **Only in the case of RK 512 and 3964(R):** If this occurs during send or receive operation, repetitions are initiated.	Disturbances on the transmission line cause message frame repetitions, thus lowering user data throughput. Danger of an undetected error increases. Correct fault by changing system setup or line installation. Check that the settings for baud rate, parity and number of stop bits are the same on both devices.
(08)0DH	BREAK The connection line (receive line) to the partner device is interrupted	Set up the connection between the devices or switch the partner device on. In the case of TTY, check whether there is a current loop in the idle state.
(08)12H	**With ASCII driver only:** More characters were received after the CP had sent XOFF or set CTS to OFF	Reparameterize communication partner or read data from CP more quickly.
(08)15H	Discrepancy between settings for transfer attempts at CP and communication partner.	Parameterize same number of transfer attempts at communications partner as at CP. Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(08)16H	• The length of a received message frame was longer than the length agrred upon or • the length of the parameterized receive buffer (with CP 441 only) is too short.	• A correction is neccesary at the partner or • the length of the receive buffer (with CP 441 only) must be enlargened
(08)18H	**With ASCII driver only:** DSR = OFF or CTS = OFF	The partner has switched the DSR or CTS signal to ”OFF” before or during a transmission. Check the partner’s control of the RS 232C secondary signals.

Table 8-8 Event Classes and Event Numbers, continued

Event Class 9 (09H): "Response message frame received from interconnection partner with error or error message frame"		
Event No.	**Event Text**	**Remedy**
(09)02H	**RK 512 only:** Memory access error at partner (memory does not exist) When SIMATIC S5 is the partner: • Incorrect area at status word, or • Data area does not exist (except DB/DX), or • Data area too short (except DB/DX)	Check that the partner has the desired data area and that it is big enough, or check the parameters of the called system function block. Check the specified length at the system function block.
(09)03H	**RK 512 only** DB/DX access error at the partner (DB/DX does not exist or is too short) When SIMATIC S5 is the partner: • DB/DX does not exist, or • DB/DX too short, or • DB/DX number impermissible Permissible source area for GET request exceeded	Check that the partner has the desired data area and that it is big enough, or check the parameters of the called system function block. Check the specified length at the system function block.
(09)04H	**RK 512 only:** Partner returns "Request type not permitted".	Partner malfunction, because a system command is never issued from the CP 441.
(09)05H	**RK 512 only:** Error at partner or at SIMATIC S5 as partner: • Source/destination type not permissible, or • Memory error in partner programmable controller, or • Error notifying CP/CPU at the partner, or • Partner programmable controller is in STOP state	Check whether the partner can transmit the desired data type. Check the structure of the hardware at the partner. Set the partner programmable controller to RUN.
(09)08H	**RK 512 only:** Partner detecting synchronization error: Message frame sequence error.	This error occurs at restart of your own programmable controller or of the partner. This represents normal system start-up behavior. You do not need to correct anything. The error is also conceivable during operation as a consequence of previous errors. Otherwise, you can assume an error on the part of the partner device.
(09)09H	**RK 512 only:** DB/DX disabled at partner by coordination flag	In partner program: After processing of the last transmission data, reset the coordination flag. In own program: Repeat the request.
(09)0AH	**RK 512 only:** Error detected by partner in message frame header: 3rd command byte in header is incorrect	Check whether the error is the result of disturbances or of a malfunction at the partner. Use an interface test device switched into the transmission line to check.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(09)0BH	**RK 512 only:** Error in message frame header: 1st or 4th command byte in header is incorrect	Check whether the error is the result of disturbances or of a malfunction at the partner. Use an interface test device switched into the transmission line to check.
(09)0CH	**RK 512 only:** Partner detects incorrect message frame length (total length).	Check whether the error is the result of disturbances or of a malfunction at the partner. Use an interface test device switched into the transmission line to check.
(09)0DH	**RK 512 only:** Partner has not yet restarted.	Restart the partner programmable controller or set the mode selector on the CP or CPU to RUN.
(09)0EH	**RK 512 only:** Unknown error number received in response message frame.	Check whether the error is the result of disturbances or of a malfunction at the partner. Use an interface test device switched into the transmission line to check.
Event class 10 (0AH): **"Errors in response message frame of the partner detected by the CP"**		
Event No.	**Event Text**	**Remedy**
(0A)01H	**RK 512 only:** Synchronization error of partner, because: • Response message frame without request • Response message frame received before continuation message frame sent • Continuation response message frame received after an initial message frame was sent • A first response message frame was received after a continuation message frame was sent	This error is reported after your own programming device is restarted in the case of long message frames or when the partner is restarted. This represents normal system start-up behavior. You do not have to correct anything. The error can also occur during operation as a consequence of error statuses only recognized by the partner. Otherwise, you can assume an error on the part of the partner device. The error may not occur in the case of requests < 128 bytes.
(0A)02H	**RK 512 only:** Error in the structure of the received response message frame (1st byte not 00 or FF)	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(0A)03H	**RK 512 only:** Received response message frame has too many data or not enough data.	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.
(0A)04H	**RK 512 only:** Response message frame for SEND request arrived with data.	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.

Table 8-8 Event Classes and Event Numbers, continued

Event No.	Event Text	Remedy
(0A)05H	**RK 512 only:** No response message frame from partner within monitoring time.	Is the partner a slow device? This error is also often displayed as a consequence of a previous error. For example, procedure receive errors (event class 8) can be displayed after a GET message frame was sent. Reason: As a result of disturbances, the response message frame could not be received, and the monitoring time elapsed. This error can also occur if a restart was carried out at the partner before it could respond to the most recently received GET message frame.
(0A)06H	**RK 512 only:** Received response message frame after GET request has not enough data.	Check for malfunction at partner device, possibly by using interface test device switched into the transmission line.

8.5 Error Numbers in the Response Message Frame

If you are working with the RK 512 computer connection and an error occurs at the communication partner in a SEND/PUT or GET message frame, the communication partner sends a response message frame with an error number in the 4th byte.

Error Numbers in the Response Message Frame

The table below shows how the error numbers in the response message frame (REATEL) are assigned to the event classes/numbers in the STATUS output of the communication partner. The error numbers in the response message frame are output as hexadecimal values.

Table 8-9 Error Messages in the Response Message Frame with RK 512

REATEL	SYSTAT Error Messages
	Event Class/Event Number
0AH	0301H 0303H 0407H 0905H
0CH	0301H 0302H 0607H 0609H 060AH 0902H
10H	0601H 0604H 0605H 090BH
12H	0904H
14H	0301H 0302H 0606H 0903H
16H	0602H 0603H 090AH
2AH	060DH 090DH
32H	060FH 0909H

Table 8-9 Error Messages in the Response Message Frame with RK 512, continued

REATEL	SYSTAT Error Messages	
	Event class	**Event Number**
34H	0608H 060BH 060CH 0611H 090CH	
36H	060EH 0908H	

8.6 Diagnosis by Means of the Diagnostic Buffer of the CP 441

Diagnostic Buffer on the CP 441

The CP 441 (6ES7 441-_AA02-0AE0 and later) has its own diagnostic buffer in which all the diagnostic events of the CP 441 are entered in the order in which they occur.

The following are displayed in the diagnostic buffer of the CP 441:

- Operating status of the CP441
- Hardware/firmware errors on the CP 441
- Initialization and parameterization errors
- Errors during execution of a CPU request
- Data transmission errors (send and receive errors)

The diagnostic buffer allows the causes of errors in point-to-point communication to be evaluated subsequently in order, for example, to determine the causes of a STOP of the CP 441 or to trace the occurrence of individual diagnostic events.

Note

The diagnostic buffer is a ring buffer for a maximum of 64 diagnostic entries. When the diagnostic buffer is full, the oldest entry is deleted when a new entry is made in it. The most recent entry always comes first.

When the power of the CP 441 is switched off, the contents of the diagnostic buffer are lost.

Reading the Diagnostic Buffer at the Programming Device

The contents of the diagnostic buffer of the CP 441 can be read by means of the STEP 7 information functions.

Note

Diagnostic events in the diagnostic buffer of the CP 441 can be read using STEP 7 **as of Version 4.0**.

All the user-relevant information in the CP diagnostic buffer is displayed to you on the ”Diagnostic Buffer” in the ”Module Information” dialog box. You can call the ”Module Information” dialog box under STEP 7 from SIMATIC Manager.

Precondition: In order to obtain the status of the module, there must be an on-line connection from the programming device to the programmable controller (on-line view in the project window).

Proceed as follows:

1. Open the relevant SIMATIC 400 station (by double-clicking it or by choosing the **Edit > Open**) menu command.
2. Open the ”Hardware” object contained in it (again by double-clicking it or by choosing the **Edit > Open**) menu command.

   Result: The window containing the configuration table appears.
3. Select the CP 441 in the configuration table.
4. Choose the **PLC > Module menu command**.

   Result: The ”Module Information” dialog box appears for the CP 441. The ”General” tab is displayed by default the first time you call it.
5. Select the ”Diagnostic Buffer” tab.

   Result: The ”Diagnostic Buffer” tab displays the most recent diagnostic events of the CP 441. Any additional information on the cause of the problem appears in the ”Details of the event” part of the tab.

The event’s numeric code is displayed in the “Event ID” field. The 16#F1C8 leader for interface 1 and the 16#F9C8 leader for interface 2 are non-variables. The rest of the ID code corresponds to event class and event number of the events described in Section 8.4. By clicking the ”Help on Event” button you can display the help text on the event text as described in Section 8.4.

If you click the ”Update” button, the current data is read from the CP 441. By clicking the ”Help on Event” button you can display a help text on the selected diagnostic event with information on error correction.

8.7 Diagnostic Alarm

The CP 441 can trigger a diagnostics alarm on the assigned CPU, thus indicating a malfunction of the CP 441. From STEP 7, V5.0 + Service Pack 2 onwards, you can specify at parameterization whether the CP 441 is to trigger a diagnostics alarm or not in the event of serious errors.

”Diagnostics alarm = NO” is the default.

Diagnostics alarm

In the event of an error, the CP 441 provides diagnostics data on the S7-400 backplane bus. In response to a diagnostics alarm, the CPU reads the system-specific diagnostics data and enters it in its diagnostics buffer. You can read the contents of the diagnostics buffer on the CPU by means of an attached programming device.

When a diagnostics alarm event occurs, the INTF LED (red) lights up. In addition, the OB 82 is called with this diagnostics data as start information.

Organization Block OB 82

You have the option of programming error responses in the user program in OB 82.

If no OB 82 is programmed, the CPU automatically enters STOP mode in the event of a diagnostics alarm.

Diagnostics Information (as Bit Pattern)

The CP 441 provides 4 bytes of diagnostics information. To display the error that has occurred, these 4 bytes are occupied as follows:

2nd byte:

The 2nd byte of diagnostics data contains the class ID of the CP 441 in bits 0 to 3.

2nd byte							
7	6	5	4	3	2	1	0
0	0	0	1	1	1	0	0

1st, 3rd and 4th bytes:

The 1st, 3rd and 4th bytes of the diagnostics data represent the error which has occurred.

Bit 0 in the 1st byte is the group error display (INTF). Bit 0 is always set to ”1” if at least one bit from bits 1 to 7 is set to ”1”, i.e. if at least one error is entered in the diagnostics data.

Event	1st byte								3rd byte								4th byte							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Wire break	0	0	1	0	1	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Incorrect parameter	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Diagnostics Information (hexadecimal)

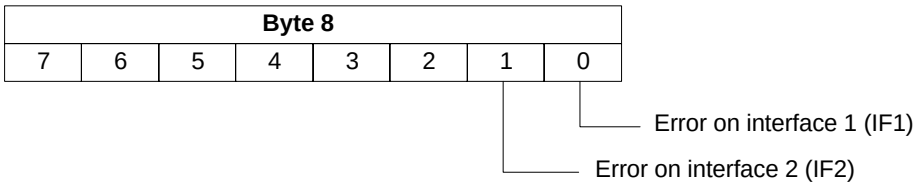
The table below shows the 4th byte in diagnostics data of the CP 441 in hexadecimal notation.

Event	1st byte	2nd byte	3rd byte	4th byte
Wire break	2DH	1CH	02H	00H
Incorrect parameter	8BH	1CH	00H	00H

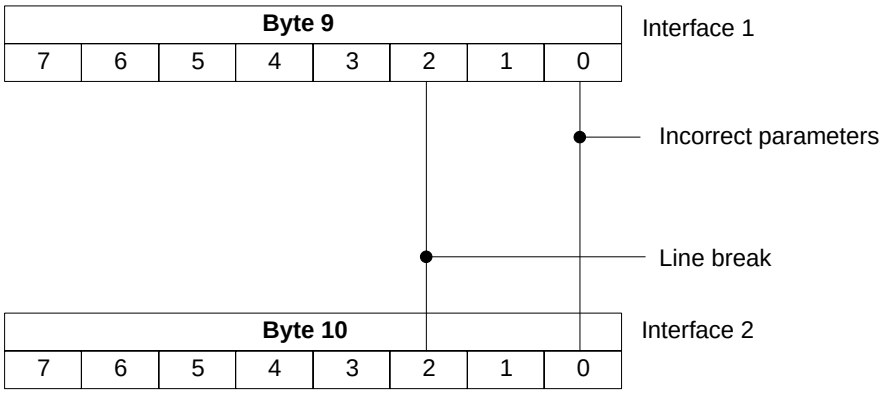
Diagnostic Information, CP 441 with Two Interfaces

DS 1 contains the information as to which interface of your CP 441-2 (6ES7 441-2AA03-0AE0) was errored. You can read DS 1 by calling SFC 59 ”RD_REC”.

A bit is set in byte 7 for each interface on which an error occurs.



The error is defined in byte 8 for interface 1 or byte 9 for interface 2.



Dependency of Diagnostics Alarm on CPU Operating Mode

A diagnostics alarm is generated via the I/O bus when fault events (rising edge) and back-to-normal events (falling edge) occur.

When the CPU switches from STOP mode to RUN mode, the following happens:

- Events (both fault and back-to-normal) which occurred when the CPU was in STOP mode are not stored,
- Events that are still present when the CPU is back to RUN mode are signaled via the diagnostics alarm.

Programming Example for System Function Blocks

9

9.1 General

The programming example in this chapter describes how to create a project and, by means of a simple data transfer operation, the basic usage of the system function blocks for operating the CP 441 communication processor.

The individual steps described for configuring and programming should make it easier for you to create a project.

Because only the general procedure is presented here and the sequence of the individual steps may vary over the course of time in the individual STEP 7 packages, you should also consult the current documentation for these packages.

At the end of the chapter you will find an example of how to program the output of message texts on a printer.

Objectives

The programming example

- aims to show the most important functions,
- is clear and easy to understand,
- can be extended for your own purposes without difficulty.

The example shows how a connection to a communication partner can be configured using the system function blocks BSEND and BRCV (for sending and receiving data respectively).

The CP 441 modules are parameterized by the CPU at CPU start-up (system service).

Requirements

The example can be executed with the minimum hardware equipment.

Program Example

The examples are on the CD which contains the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface. They are in compiled form.

These programs are installed together with the parameterization interface. The installation procedure is described in Section 5.2.

Once installation has completed, the examples are in the STEP 7 “Examples” catalog under CP 441.

9.2 Device Configuration

Using the Example

The following are some of the devices that could be used to try out the program example:

- An S7-400 PLC (rack, power supply, CPU 414 or CPU 416)
- A CP 441
- A programming device (e.g. PG 740)

Data transmission is from interface 1 to interface 2 of the CP 441. If you use a CP 441-1 you can ignore the settings for interface 2; your communication partner receives the data.

Device Configuration

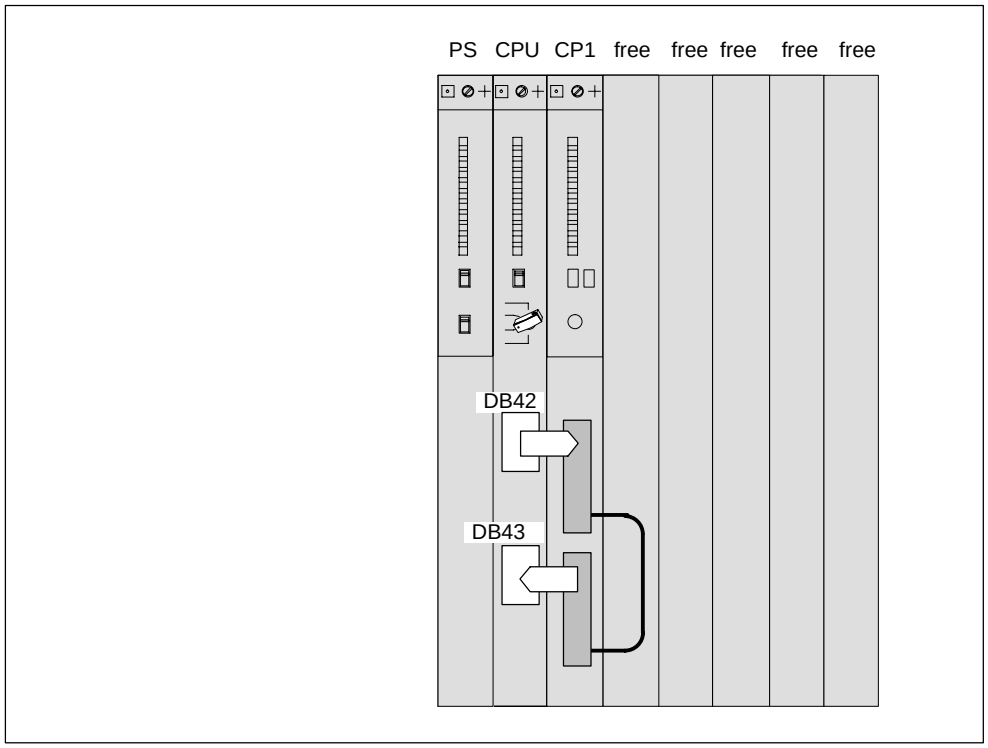


Figure 9-1 Data Flow: Device Configuration with a CP 441-2

9.3 Configuring the Controller Setup

Configuration

An S7-400 station is configured using ”HW Config” in STEP 7. Place the modules in the configuration table in accordance with your hardware configuration.

In the configuration table you must configure the controller setup as follows:

- Slots 1 and 2: power supply (e.g. PS 10A)*
- Slot 4: CPU
- Slot 10: CP 441

* The number of occupied slots depends on the type of power supply used.

9.4 Parameterizing the CP 441

Parameterization

Once you have arranged the modules in your mounting rack, you can double-click the CP 441 (in the configuration table) to display the ”Properties” dialog box:

1. Under ”Basic Parameters”, specify in the ”Interface” entry field (1 or 2) the type of the interface submodule installed here.
2. Choose the ”Parameters” button in the ”Properties” dialog box.

   Result: The *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface for entering the protocol parameters of the CP441 is opened.
3. Select the desired transmission protocol (”RK 512”, ”3964(R)”, ”ASCII” or ”Printer”).

   Result: The parameterization interface that is available in accordance with the protocol is displayed. The gray buttons allow you to open additional parameterization dialogs.
4. Accept the default settings, and return to the configuration table.
5. If necessary, carry out steps 1. to 4. in accordance with your configuration for the second interface of the CP 441.

9.5 Configuring the Connection to the Communication Partner

Configuring a Connection

You configure a point-to-point connection between your CP 441 and the communication partner using ”NETPRO”. Proceed as follows:

1. Enter the connection in the connection table.
2. Set the object properties of the connection.

Entering the connection for interface 1 in the connection table

Proceed as follows:

1. Return to the ”<Offline> (Project)” project window, and double-click the CPU in the SIMATIC 400 station.

   Result: The ”Connections” object (connection table) appears on the right.

2. Double-click this icon. The ”Configuring Connections” dialog box appears. Choose **Insert > Connection** to insert your connection in the connection table.
3. In the ”New Connection” dialog box, select ”Unspecified” as the communication partner and enter ”PTP Connection” as the connection type. Then click ”OK” to exit the dialog box.

Setting the object properties of the connection

In the ”Object Properties” dialog box, set the specific properties of the connection for interface 1:

In the ”Object Properties” dialog box, change the name of the communication partner from ”Unspecified” to an appropriate name and select the “PTP– CP Rack/Slot” and “IF_1” interface. No other settings are necessary.

4. Click ”OK” to return to the ”Configuring Network” dialog box.

Result: The ”Configuring Network” dialog box displays the connection that you have added and the ”Local ID (Hexadecimal)”. You have to specify this ID as the parameter ”ID” at the BSEND system function block in the user program of your CPU.

Entering the connection for interface 2 in the connection table

Proceed as follows:

1. After returning to the ”Configuring Network” dialog box, insert the connection in the connection table by choosing **Insert > Connection**.
2. In the ”New Connection” dialog box, select ”Unspecified” as the communication partner and enter ”S7 PTP Connection” as the connection type. Then click ”OK” to exit the dialog box.

Setting the object properties of the connection

In the ”Object Properties” dialog box, set the specific properties of the connection for interface 2:

In the ”Object Properties” dialog box, change the name of the communication partner from ”Unspecified” to an appropriate name and select the “PTP– CP Rack/Slot” and “IF_2” interface. No other settings are necessary.

3. Click ”OK” to return to the ”Configuring Network” dialog box.

Result: The ”Configuring Network” dialog box displays the connection that you have added and the ”Local ID (Hexadecimal)”. You have to specify this ID as the parameter ”ID” at the BRCV system function block in the user program of your CPU.

9.6 Programming an RK 512 User Program

9.6.1 Program CP441 RK 512 Send/Recv

General

In OB 1, the blocks FC21 and FC 23 are called cyclically, FC21 for sending data (SFB BSEND) and FC 23 for receiving data (SFB BRCV).

In the example, the system function blocks BSEND and BRCV work with the data blocks DB 12 and DB 13 as instance DBs, and with DB 42 and DB 43 as send and receive DBs respectively.

In the example the system function blocks are parameterized partly via constants and partly via symbolically addressed actual operands.

Connection_ID 1000 (hexadecimal) is entered for the BSEND and the associated STATUS. Connection_ID 1001 (hexadecimal) is entered for the BRCV and the associated STATUS. If you are using a CP 441-1, you must enter connection ID 1000 (hexadecimal) for the BRCV and the associated STATUS. You can then receive data from your communication partner via interface 1.

For the data transmission the block pair BSEND and BRCV are used. The same R_ID is used for both blocks, as explained in Section 6.5.

The values for R_ID are accepted once during startup and cannot subsequently be changed.

This ensures that the SFBs BSEND and STATUS are run through once at the beginning with REQ = ”0” (so the edge rises from ”0” to ”1” at the REQ input), the REQ parameter in OB 100 is set to ”0” once after a complete restart.

Description of FC 21 (SEND)

The “Generate edge P_SEND_REQ” program section:

BSEND is run through once at the start with P_BSEND_REQ =0. BSEND _REQ is then set to 1. The BSEND request is started when a signal state change from ”0” to ”1” is detected at the BSEND_REQ control parameter.

When BSEND_DONE = 1 or BSEND_ERROR = 1, BSEND_REQ is reset to “0”.

The “BSEND_DONE = 1” program section

If the transfer is successful, BSEND_DONE at the parameter output of the BSEND is set to ”1”.

To distinguish between consecutive transfers, ”BSEND_COUNTER_OK” send counter is included in data word 0 of the source block DB 42.

The “BSEND_ERROR = 1” program section

If BSEND runs through with BSEND_ERROR=1, the BSEND_COUNTER_ERR error counter in data word 2 increments. The BSEND_STATUS is copied, because it will be overwritten with 0 in the next run and could not subsequently be read.

In addition, in the event of an error, the STATUS system function block is activated so that detailed fault messages (LOCAL parameter) can be read (see Section 8.4).

Description of FC 23 (RECEIVE)

The “Enable Receive Data“ program section:

For data to be received, the receive enable (control parameter BRCV_EN_R for the BRCV block) must have the signal ”1”.

The “BRCV_NDR=1” program section:

When BRCV_NDR is set, new data has been received and the BRCV_COUNTER_OK receive counter increments.

The “BRCV_ERROR = 1” program section:

If the startup is unsuccessful, i.e. if the ERROR bit is set at the parameter output of BRCV, the BRCV_COUNTER_ERR error counter increments. The BSEND_STATUS is copied, because it will be overwritten with 0 in the next run and could not subsequently be read.

In addition, in the event of an error, the STATUS system function block is activated so that detailed fault messages (LOCAL parameter) can be read (see Section 8.4).

All relevant values can be observed for test purposes in the variable table.

Special Features of the CP 441-1:

If you want to receive data from your communication partner, the partner must specify DX 33 (21 hexadecimal) as the destination address. In this way BRCV is referenced with R_ID 21 (hexadecimal) in FC 23.

9.6.2 Blocks Used in the Example Program

Blocks Used

The table below lists the blocks used for the example program.

Precondition: All symbolic designations have already been declared in the symbol table.

Table 9-1 Blocks Used in the Sample Program

Block	Symbol	Comment
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Start-up OB for restart
FC 21	SEND	FC with call and analysis of BSEND SFB
FC 23	RECEIVE	FC with call and analysis of BRCV SFB
DB 12	SEND IDB	Instance DB for BSEND SFB
DB 13	RCV IDB	Instance DB for BRCV SFB
DB 22	STATUS IDB BSEND	Instance DB for STATUS SFB
DB 23	STATUS IDB RECEIVE	Instance DB for STATUS SFB
DB 42	SEND SRC DB	Send data block (source)
DB 43	RCV DST DB	Receive data block (destination)
DB 40	SEND WORK DB	Work DB for BSEND
DB 41	RCV WORK DB	Work DB for BRCV
DB 45	STATUS WORK DB BSEND	Work DB for STATUS
DB 46	STATUS WORK DB BRCV	Work DB for STATUS

9.7 Programming an ASCII/3964(R) User Program

If you intend to transfer data using the ASCII/3964(R) protocol (program example CP441 ASCII Send/Recv), you need only make the following changes:

- When parameterizing, you must use the parameter dialogs for the ASCII/3964(R) protocol.
- For data transmission, the block pair BSEND and BRCV is also used in the user program. As described in Section 6.4, subsection ”Data Transmission with 3964(R) Using BSEND and BRCV”, any R_ID can be used for the BSEND, whereas for the BRCV you must use the R_ID ”0”.

9.8 Programming a Printer User Program

The following sections describe an example of how to send data to a printer. The example program indicates the procedure for the data editing and parameterization of the PRINT SFB.

Requirements

Message texts configured beforehand with the *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface (menu command: **Hardware > Properties of CP 441 > Parameter > Messages**) . The messages have been transferred together with the other parameterization data on the CP 441.

Message text examples:

1:\B%C\-B F220340 Share of component no. \B%I\-B reached \B%10.2R Kg\-B
2:\B%S\-B H244312 Bypass fitting closed
3:\B%S\-B H620125 Failure of station input hydraulic mechanism
4:\B%S\-B P215055 Tank gauge pressure %12.4R bar

The result on the printer for message text No. 1, for example, is: ”**W** F220340 proportion component No. **6** attained **1.45E+02 Kg**”

9.8.1 Cyclic Program

General

The organization block OB 1 contains the cyclic program.

In the example the PRINT system function block works with the DB 16 data blocks, and the STATUS system function block works with the DB 22 data block as the instance DB (clipboard).

The same connection ID is to be entered at the appropriate input parameters for the PRINT SFB and the associated STATUS SFB in the program, since the STATUS SFB works on a connection-related basis.

The PRINT request is supplied with data from DB 146, DB 160 and DB 165.

In FB 50, the PRINT SFB and the STATUS SFB are initialized by means of a one-off call with the signal ”0” at the REQ input parameter.

Description of the ”Printer Output” Program

The print request is sent to the printer when a signal state change from ”0” to ”1” is detected at the REQ control parameter.

In the event of successful completion of the request, the DONE output parameter is set to the signal ”1” at the PRINT SFB. In the event of an error, the ERROR output parameter is set to the signal ”1” instead. In the program, the REQ control input is reset to ”0” if one of these signals is positive.

In the next cycle, the DONE parameter is set to ”0”. The REQ input thus becomes ”1” and the requested signal state change from ”0” to ”1” results in data transfer to the printer.

In addition, in the event of an error, the STATUS system function block (SFB 22) is called with DB 22 as the instance DB in order to obtain detailed information on the cause of the error. The ERROR output parameter of the PRINT SFB serves as the trigger for the STATUS SFB.

In the event of a signal state change from ”0” to ”1”, the REQ input of the STATUS SFB is activated. The 16 bytes addressed at the LOCAL parameter receive the current error status of the connection until the next STATUS SFB call.

The status of the memory words or data words is monitored in the variable table. Other test options are available to you if you insert load commands in the program (of any variables) and then monitor them by means of the ”Block Status” function in online operation.

You can select other options with the variable table by means of CONTROL VARIABLES.

The ENABLE_JOB_1, ENABLE_JOB_2 and ENABLE_JOB_3 variables offer you a choice of three different jobs.

The first job returns a printout of a message text. The second and third jobs return a simple printout of a single variable or more than one variable, respectively.

9.8.2 Blocks Used in the Example Program

Blocks Used

The table below lists the blocks used for the example program.

Precondition: All symbolic designations have already been declared in the symbol table.

Table 9-2 Blocks Used in the Sample Program

Block	Symbol	Comment
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Start-up OB for restart
FB 50	PRINT A	“PRINT A”
FB 51	PRINT B	“PRINT B”
FB 52	PRINT C	“PRINT C”
DB 16	PRINT IDB	Instance DB for the PRINT SFB
DB 22	STATUS IDB	Instance DB for STATUS SFB
DB 146	DB_with_Convers _ Statem	DB with conversion statement for representation type N
DB 160	Process_Values	DB transfer of process values
DB 165	ME_WA_AL	Message type

9.9 Installation, Error Messages

Installation

The hardware for the example is fully set up and the programming device is connected.

Once the CPU has been reset (operating mode STOP), transfer the example corresponding to your hardware configuration fully into the user memory. Then use the operating mode switch to change from STOP to RUN_P **(start-up characteristic CRST)**.

Malfunction

If an error occurs during start-up, the block calls to be processed cyclically will not be executed and the error LED INTF or EXTF on the CPU will be set. Detailed information on the cause of the error can be found in the diagnostics buffer.

Technical Specifications

A

In Section	You Will Find	on Page
A.1	Technical Specifications of the CP 441 and the Interface Submodules	A-2
A.2	Certification and Areas of Application	A-6

A.1 Technical Specifications of the CP 441 and the Interface Submodules

Technical Specifications of the CP 441

The following table contains the technical specifications of the CP 441.

Table A-1 Technical Specifications of the CP 441

Technical Specifications	
Power supply	max. 0.7 A at 5 V
Operating temperature	0° C to +60° C
Storage temperature	≃ 40° C to +70° C
Power loss	3.5 W
Degree of protection	IP20
EMC	EN 50082
Dimensions W × H × D	25 × 290 × 210 mm
Weight	approx. 0.3 kg
Indicators	LEDs for transmit (TXD), receive (RXD) and interface fault (FAULT) Group alarm LEDs for internal fault (INTF) and external fault (EXTF)
Protocol drivers CP 441-1	ASCII driver 3964 (R) procedure printer
Protocol drivers CP 441-2	ASCII driver 3964 (R) procedure RK 512 computer connection printer loadable drivers
Alarms • Diagnostics alarm	 parameterizable
Diagnostic functions • Indicators for internal and external faults • Diagnostics information dump	 yes, 2 red LEDs yes

Technical Specifications of the Interface Submodules

The following table contains the technical specifications of the plug-in interface submodules of the CP 441.

Table A-2 Technical Specifications of the Interface Submodules

Technical Specifications	RS 232C	20 mA TTY	X27 (RS 422/485)
Power supply	max. 0.1 A at 5 V	max. 0.1 A at 5 V max. 0.045 A at 24 V	max. 0.25 A at 5 V
Power loss	0.5 W	1.5 W	1.25 W
Operating temperature	0° C to +60° C	0° C to +60° C	0° C to +60° C
Storage temperature	≃ 40° C to +70° C	≃ 40° C to +70° C	≃ 40° C to +70° C
Degree of protection	IP00	IP00	IP00
EMC	EN 50082	EN 50082	EN 50082
Isolation	no	yes	yes
Dimensions W × H × D	approx. 95 × 70 × 20 mm	approx. 95 × 70 × 20 mm	approx. 95 × 70 × 20 mm
Weight	0.08 kg	0.08 kg	0.08 kg
Baud rate	max. 115.2 kbaud min. 300 baud	max. 19.2 kbaud min. 300 baud	max. 115.2 kbaud min. 300 baud
Line length	max. 10 m	max. 1000 m at 9600 baud	max. 1200 m at 19200 baud
Front connector	9-pin sub D male with screw fixing	9-pin sub D female with screw fixing	15-pin sub D female with screw fixing

Transmission Times

The tables below indicate the transmission times required depending on the transmission protocol selected.

Two S7-400s each with a CPU 416 (6ES7 416-1XJ00-0AB0) and a CP 441-2 were used to measure the times. A BSEND system function block was programmed in the user program of the active CPU, and a BRCV system function block was programmed in the user program of the passive CPU. The second interface of the CP 441-2 was not parameterized. The time that elapsed between the initiation and completion of the request was measured.

If you are using a CPU 412, 413 or 441, you should add around 20% to the times indicated here.

You will find the runtimes of the communication blocks in the relevant CPU manual.

ASCII driver

Transmission times with the ASCII driver:

Table A-3 Transmission Times with the ASCII Driver

Baud rate (bd) / User data	115200	76800	57600	38400	19200	9600	4800	2400	1200	600	300
1 byte	0.022s	0.022s	0.022s	0.022s	0.023s	0.024s	0.025s	0.027s	0.045s	0.090s	0.185s
10 bytes	0.023s	0.023s	0.024s	0.024s	0.027s	0.031s	0.042s	0.063s	0.125s	0.267s	0.500s
20 bytes	0.024s	0.024s	0.025s	0.027s	0.034s	0.045s	0.068s	0.127s	0.217s	0.428s	0.882s
50 bytes	0.026s	0.029s	0.030s	0.035s	0.052s	0.094s	0.156s	0.267s	0.500s	1.000s	2.000s
100 bytes	0.030s	0.035s	0.038s	0.048s	0.098s	0.157s	0.258s	0.517s	1.000s	2.000s	4.000s
200 bytes	0.039s	0.049s	0.074s	0.098s	0.159s	0.277s	0.517s	1.000s	2.000s	4.000s	8.000s
500 bytes	0.121s	0.141s	0.163s	0.217s	0.384s	0.652s	1.250s	2.400s	4.800s	9.600s	18.800s
1000 bytes	0.215s	0.238s	0.302s	0.394s	0.681s	1.400s	2.400s	4.800s	10.600s	21.200s	37.600s
2000 bytes	0.417s	0.469s	0.586s	0.750s	1.360s	2.500s	5.000s	9.600s	21.200s	42.400s	65.200s
4000 bytes	0.831s	0.943s	1.166s	1.500s	2.700s	5.000s	10.000s	19.200s	42.400s	84.800s	146.600s

3964(R) procedure

Transmission times with the 3964(R) procedure:

Table A-4 Transmission Times with the 3964(R) Procedure

Baud rate (bd) / User data	115200	76800	57600	38400	19200	9600	4800	2400	1200	600	300
1 byte	0.022s	0.023s	0.022s	0.025s	0.028s	0.032s	0.041s	0.062s	0.096s	0.166s	0.294s
10 bytes	0.023s	0.029s	0.024s	0.031s	0.037s	0.046s	0.063s	0.101s	0.174s	0.326s	0.600s
20 bytes	0.024s	0.031s	0.025s	0.033s	0.043s	0.058s	0.084s	0.148s	0.267s	0.500s	1.000s
50 bytes	0.026s	0.036s	0.030s	0.043s	0.061s	0.094s	0.157s	0.283s	0.555s	1.000s	2.000s
100 bytes	0.031s	0.047s	0.040s	0.057s	0.097s	0.148s	0.277s	0.535s	1.100s	2.100s	4.000s
200 bytes	0.041s	0.058s	0.077s	0.089s	0.151s	0.272s	0.500s	1.000s	2.000s	4.200s	8.000s
500 bytes	0.119s	0.123s	0.159s	0.202s	0.365s	0.600s	1.250s	2.400s	4.800s	9.600s	18.800s
1000 bytes	0.211s	0.230s	0.298s	0.384s	0.714s	1.250s	2.500s	4.800s	10.600s	21.200s	37.600s
2000 bytes	0.412s	0.454s	0.575s	0.714s	1.363s	2.500s	5.000s	9.600s	21.200s	42.400s	65.200s
4000 bytes	0.820s	0.882s	1.161s	1.500s	2.726s	5.000s	10.000s	19.200s	42.400s	84.800s	146.600s

RK 512 computer connection

Transmission times with the RK 512 computer connection:

Table A-5 Transmission Times with the RK 512 Computer Connection

Baud rate (bd) / User data	115200	76800	57600	38400	19200	9600	4800	2400	1200	600	300
1 byte	0.046s	0.046s	0.051s	0.054s	0.067s	0.079s	0.108s	0.172s	0.312s	0.555s	1.153s
10 bytes	0.050s	0.049s	0.053s	0.055s	0.068s	0.093s	0.132s	0.214s	0.375s	0.714s	1.500s
20 bytes	0.052s	0.054s	0.057s	0.060s	0.076s	0.105s	0.150s	0.258s	0.468s	0.937s	1.764s
50 bytes	0.053s	0.056s	0.062s	0.067s	0.091s	0.138s	0.217s	0.384s	0.750s	1.363s	2.900s
100 bytes	0.058s	0.065s	0.074s	0.084s	0.150s	0.197s	0.326s	0.625s	1.250s	2.400s	4.700s
200 bytes	0.098s	0.122s	0.132s	0.147s	0.208s	0.357s	0.652s	1.200s	2.400s	4.800s	9.400s
500 bytes	0.182s	0.241s	0.282s	0.340s	0.500s	0.833s	1.666s	3.000s	4.800s	11.000s	22.000s
1000 bytes	0.341s	0.416s	0.530s	0.625s	0.937s	1.700s	3.000s	6.000s	11.000s	22.000s	44.000s
2000 bytes	0.675s	0.833s	1.045s	1.200s	1.800s	3.300s	6.000s	12.000s	22.000s	44.000s	88.000s
4000 bytes	1.320s	1.600s	2.050s	2.500s	3.500s	6.600s	12.000s	24.000s	44.000s	88.000s	176.000s

A.2 Certification and Areas of Application

This section contains information for the CP 441 on:

- The most important standards with which the CP 441 complies
- Certificates and approvals of the CP 441

IEC 1131

The CP 441 communication processor meets the requirements and criteria of IEC 1131, Part 2.

CE Marking

Our products fulfill the requirements and safety objectives of the following EC Directives and comply with the harmonized European standards (EN) published for programmable logic controllers in the official journals of the European Communities:

- 89/336/EEC Electromagnetic Compatibility Directive (EMC Directive)
- 73/23/EEC Low Voltage Directive (for electrical equipment)

The EC Declarations of Conformity are available to the relevant authorities at the following address:

Siemens Aktiengesellschaft
Bereich Automatisierungs- und Antriebstechnik
A&D AS 48
Postfach 1963
D-92209 Amberg
Germany

Area of Application

In accordance with this CE marking, the CP 441 has the following area of application:

Area of Application	Requirements	
	Emitted Interference	Noise Immunity
Industry	EN 50081-2 : 1993	EN 50082-2 : 1995

UL Recognition

UL Recognition Mark
Underwriters Laboratories (UL) to
Standard UL 508, Report E 85972

CSA Certification

CSA Certification Mark
Canadian Standard Association (CSA) to
Standard C22.2 No. 142, Report No. LR 63533

Connecting Cables

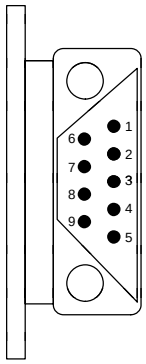
B

B.1 Interface Submodule RS 232C

Pin Allocation

The table below shows the pin allocation for the 9-pin sub D male connector in the front panel of the RS 232C interface submodule.

Table B-1 Pin Allocation for the 9-Pin Sub D Male Connector of the RS 232C Interface Submodule

Male Connector on Module*	Pin	Designation		Input/Output	Meaning
	1	DCD	Received Detector	Input	Received signal level
	2	RXD	Received Data	Input	Received data
	3	TXD	Transmitted Data	Output	Transmitted data
	4	DTR	Data Terminal Ready	Output	Data terminal is ready
	5	GND	Ground	–	Signal ground (GND_{int})
	6	DSR	Data Set Ready	Input	Data set ready
	7	RTS	Request To Send	Output	Request to send
	8	CTS	Clear To Send	Input	Clear to send
	9	RI	Ring Indicator	Input	Ring indicator

* Front view

Connecting Cables

If you construct your own connecting cables you must remember that unconnected inputs at the communication partner may have to be connected to open-circuit potential.

Note that you must only use shielded connector casings. A large surface area of both sides of the cable shield must be in contact with the connector casing and the shield contact.

Caution

Never connect the cable shield with the GND, as this could destroy the interface submodules.

GND (pin 5) must always be connected on both sides, otherwise the interface submodules could again be destroyed.

The following pages contain examples of connecting cables for a point-to-point connection between the CP 441 and S7 modules or SIMATIC S5.

RS 232C Connecting Cables (S7/M7 (CP 441) – S7/M7 CP 441/CP 340)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 441/CP 340.

For the connecting cables you require the following female connectors:

- At CP 441: 9-pin sub D female with screw fixing
- at the communication partner: 9-pin sub D female with screw fixing

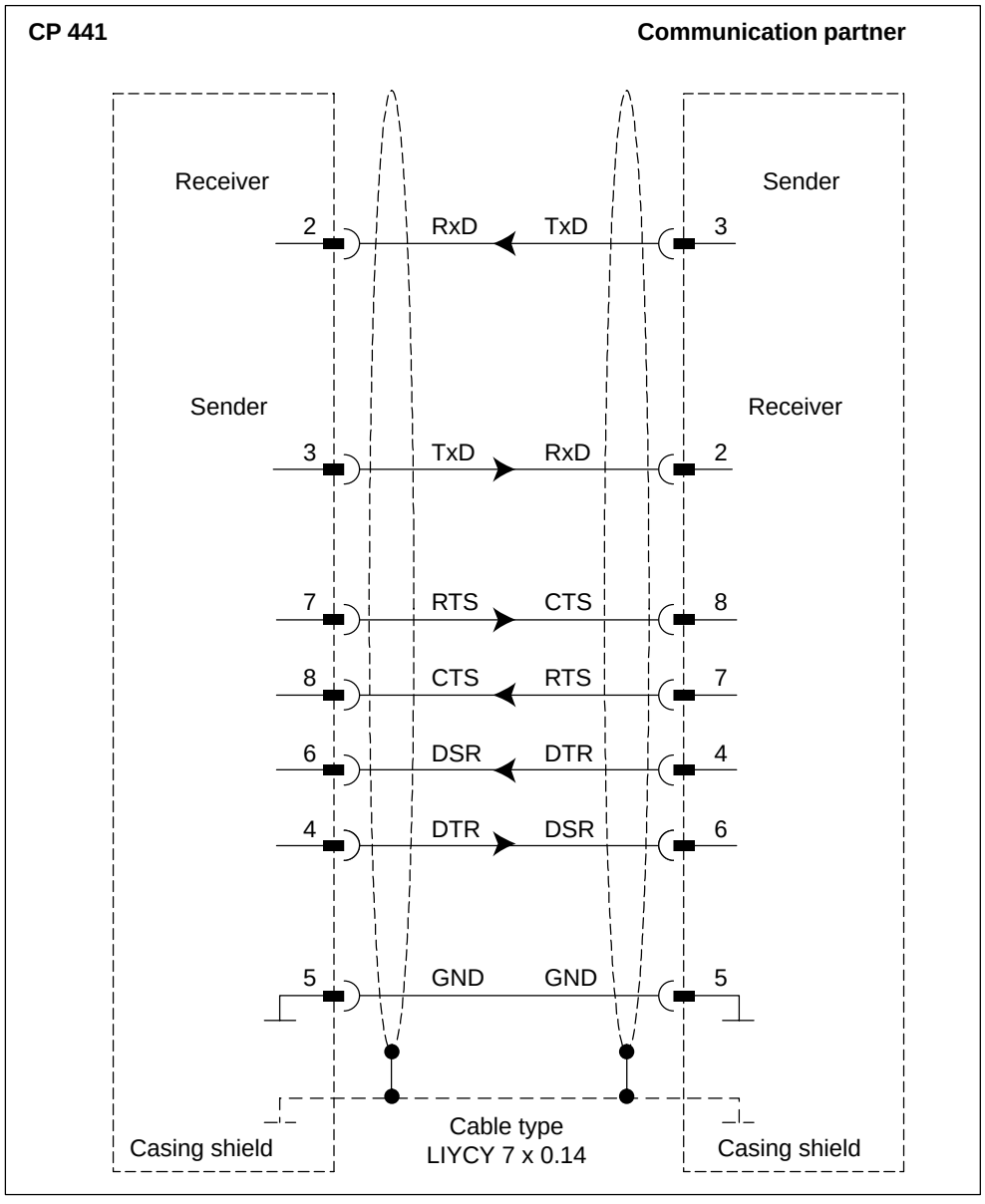


Figure B-1 RS 232C Connecting Cable CP 441 – CP 441/CP 340

The connecting cable is available using the order number (6ES7 902-1...) specified in Appendix E.

RS 232C Connecting Cables (S7/M7 (CP 441) – CP 544, CP 524, CPU 928B, CPU 945, CPU 948)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 544, CP 524, CPU 928B, CPU 945 or CPU 948.

For the connecting cables you require the following female/male connectors:

- At CP 441: 9-pin sub D female with screw fixing
- at the communication partner: 25-pin sub D male with clip fixing

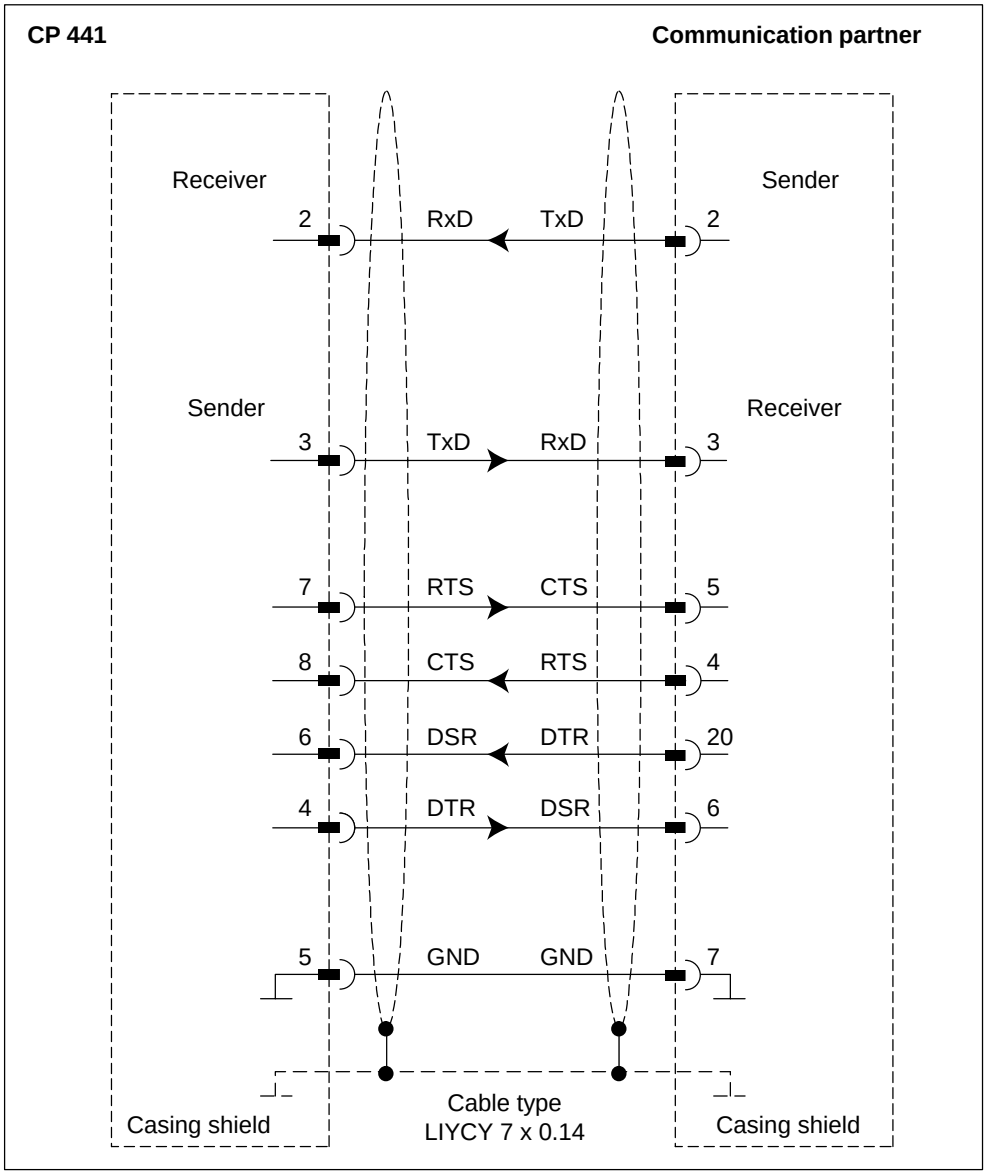


Figure B-2 RS 232C Connecting Cable CP 441 – CP 544, CP 524, CPU 928B, CPU 945, CPU 948

RS 232C Connecting Cables (S7/M7 (CP 441) – CP 521 SI/CP 521 BASIC)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 521 SI/CP 521 BASIC.

For the connecting cables you require the following female/male connectors:

- At CP 441: 9-pin sub D female with screw fixing
- at the communication partner: 25-pin sub D male with screw fixing

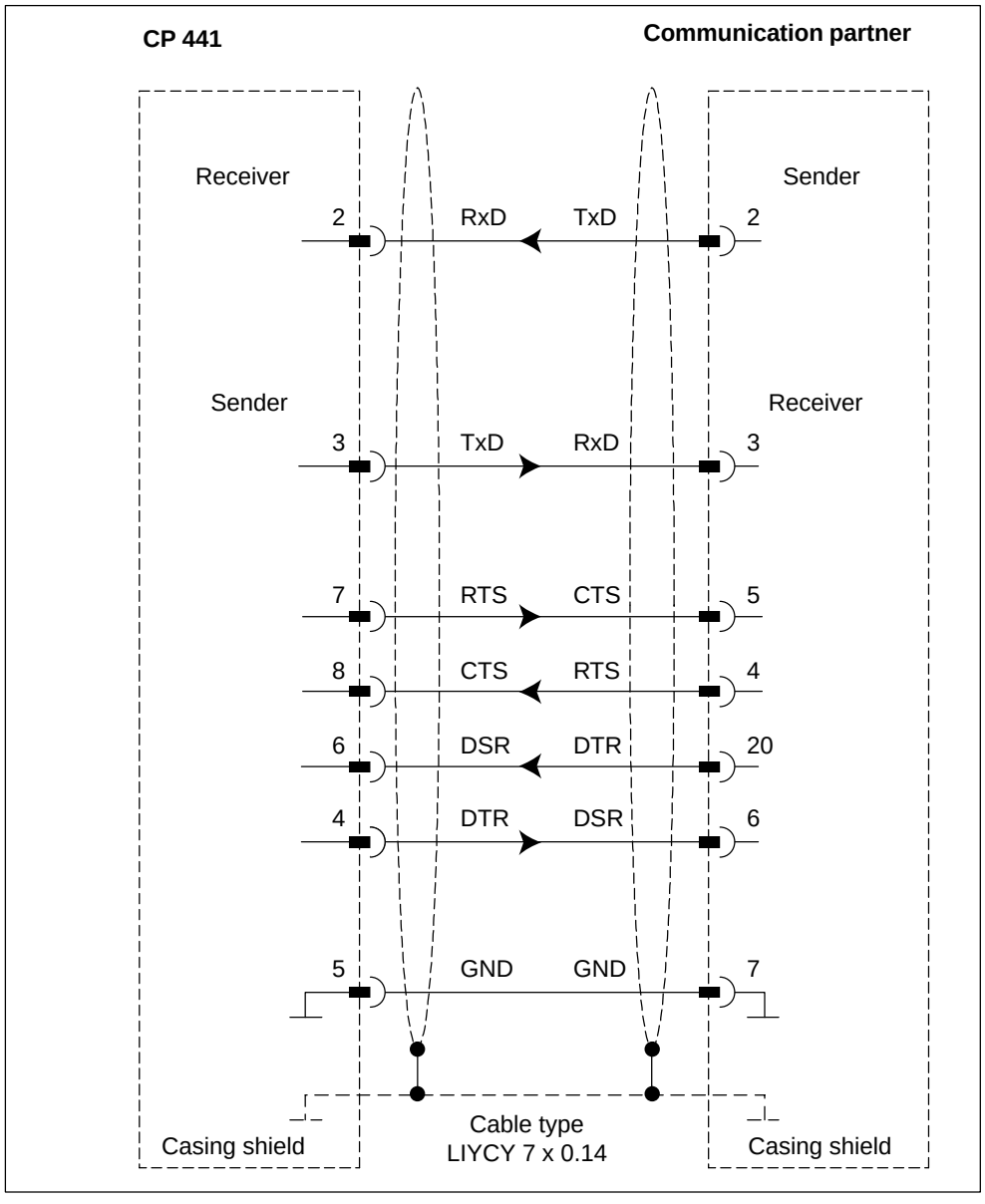


Figure B-3 RS 232C Connecting Cable CP 441 – CP 521SI/CP 521BASIC

RS 232C Connecting Cables (S7/M7 (CP 441) – CP 523)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 523.

For the connecting cables you require the following female/male connectors:

- At CP 441: 9-pin sub D female with screw fixing
- at the communication partner: 25-pin sub D male with screw fixing

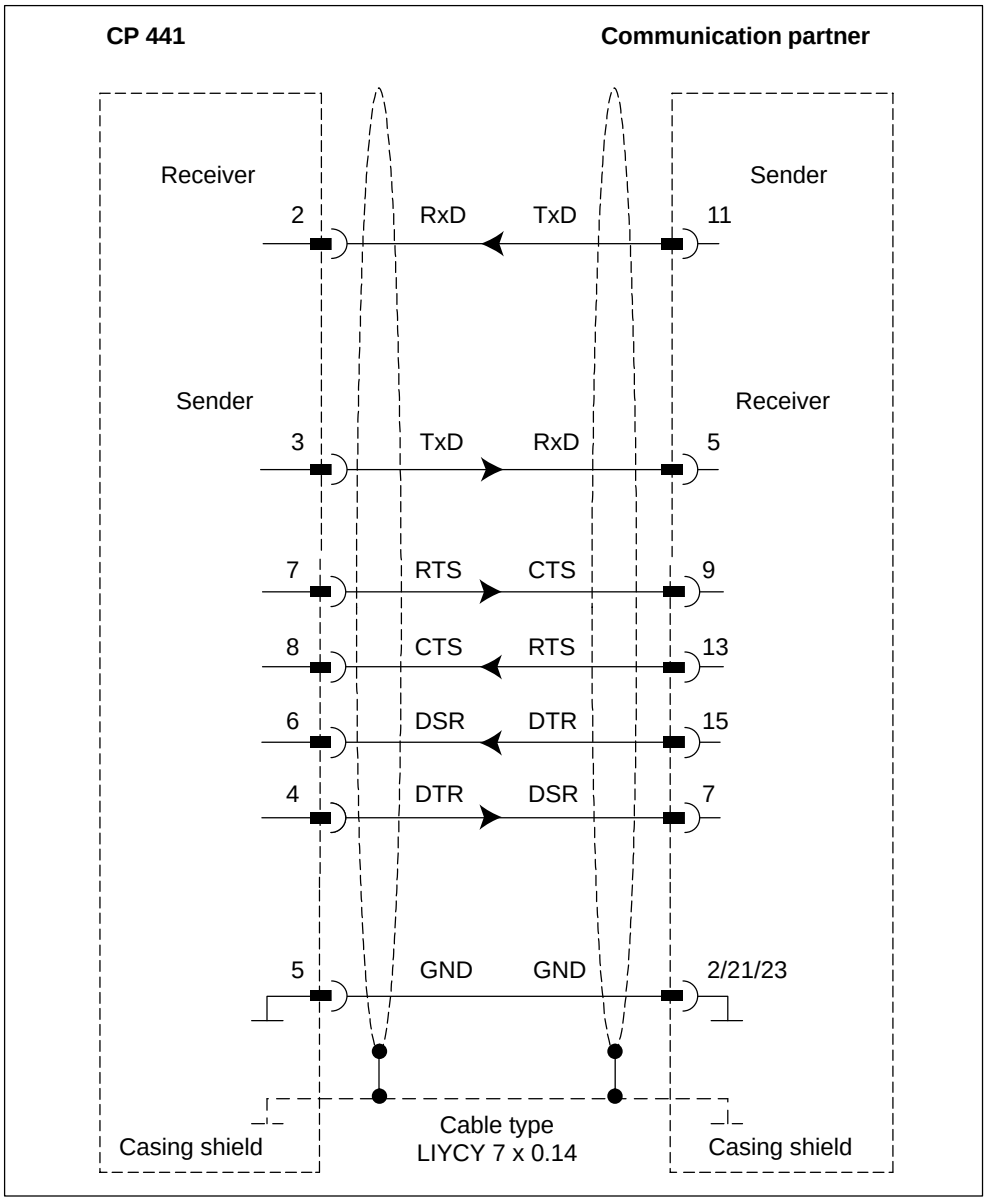


Figure B-4 RS 232C Connecting Cable CP 441 – CP 523

RS 232C Connecting Cable (S7/M7 (CP 441) – IBM Proprinter (PT 88), DR 230)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and an IBM Proprinter with a serial interface (PT 88 or IBM-compatible printer).

For the connecting cable you require the following female/male connectors:

- At CP 441: 9-pin sub D female
- At IBM Proprinter: 25-pin sub D male

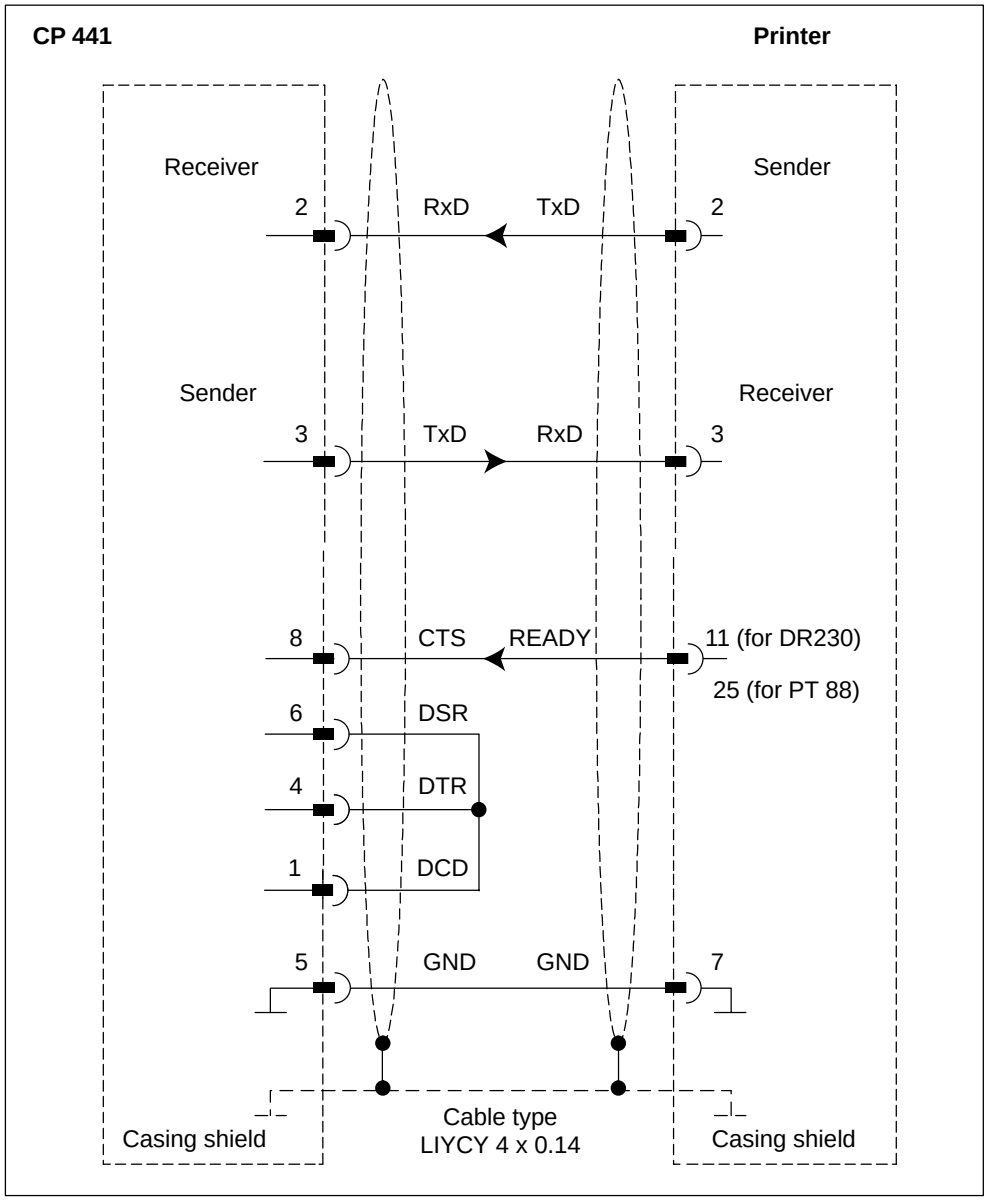


Figure B-5 RS 232C Connecting Cable CP 441 – IBM Proprinter

RS 232C Connecting Cable (S7/M7 (CP 441) – Laser Printer)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a laser printer with a serial interface (PT 10 or LaserJet Series II).

For the connecting cable you require the following female/male connectors:

- At CP 441: 9-pin sub D female
- At IBM Proprinter: 25-pin sub D male

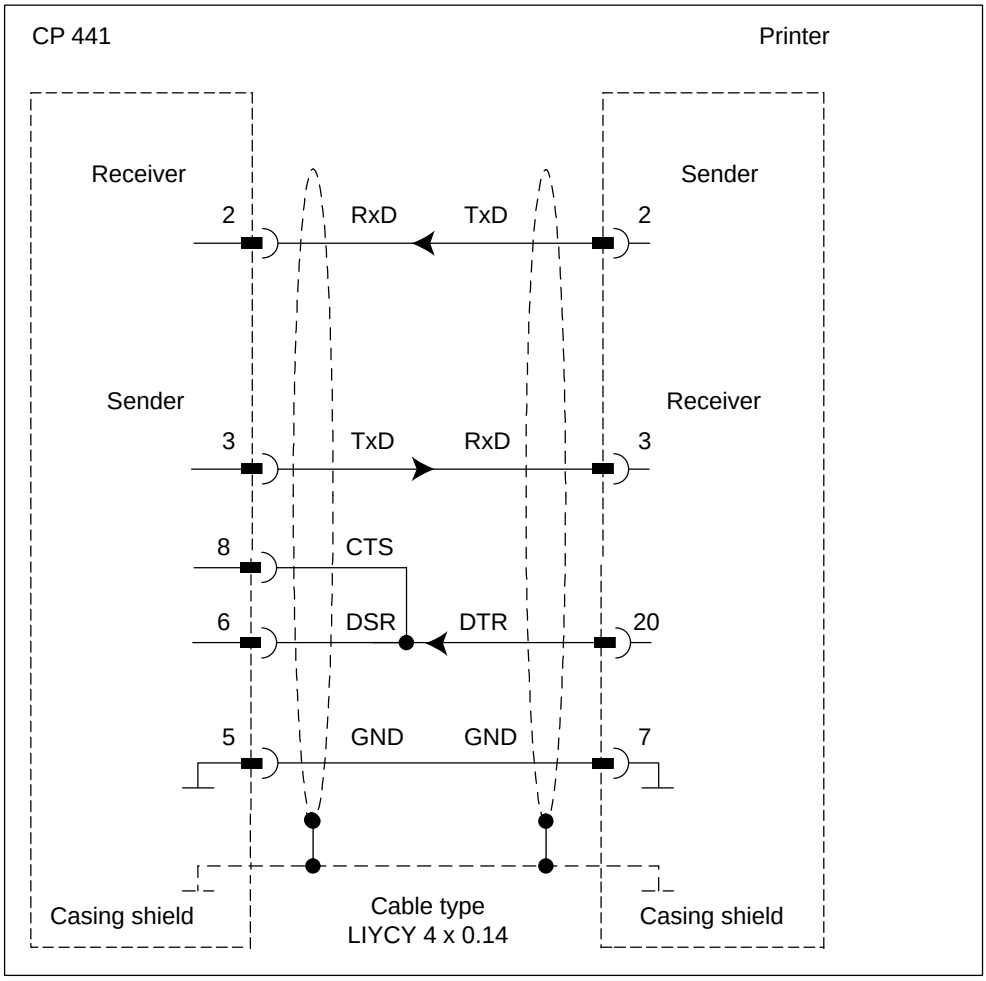


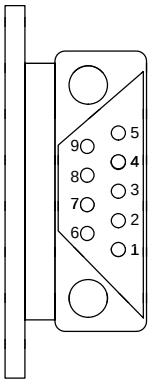
Figure B-6 RS 232C Connecting Cable CP 441 – Laser Printer

B.2 Interface Submodule 20 mA TTY

Pin Allocation

The table below shows the pin allocation for the 9-pin sub D female connector in the front panel of the 20 mA TTY interface submodule.

Table B-2 Pin Allocation for the 9-Pin Sub D Female Connector of the 20 mA TTY Interface Submodule

Female Connector on Module*	Pin	Designation	Input/Output	Meaning
	1	TxD –	Output	Transmitted data
	2	20 mA –	Input	Ground 24 V
	3	20 mA + (I_1)	Output	20 mA current generator 1
	4	20 mA + (I_2)	Output	20 mA current generator 2
	5	RxD +	Input	Received data +
	6	–		
	7	–		
	8	RxD	Output	Received data –
	9	TxD +	Input	Transmitted data +

* Front view

Block Diagram

The figure below shows the block diagram of a 20 mA TTY interface IF963-TTY.

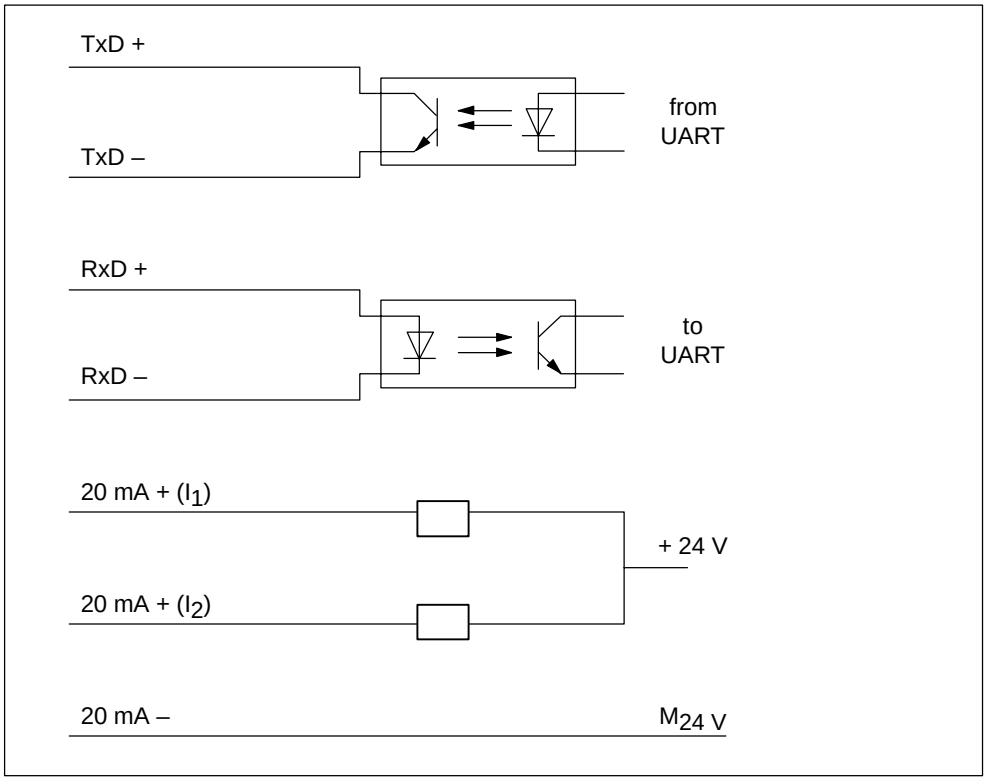


Figure B-7 Block Diagram of the 20 mA TTY Interface IF963-TTY

Connecting Cables

If you construct your own connecting cables you must remember that unconnected inputs at the communication partner may have to be connected to open-circuit potential.

Note that you must only use shielded connector casings. A large surface area of both sides of the cable shield must be in contact with the connector casing and the shield contact.

Caution

Never connect the cable shield with the GND, as this could destroy the interface submodules.

The following pages contain examples of connecting cables for a point-to-point connection between the CP 441 and S7 modules or SIMATIC S5.

20 mA TTY Connecting Cables (S7/M7 (CP 441) – S7/M7 (CP 441/CP 340)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 441/CP 340.

For the connecting cables you require the following male connectors:

- At CP 441: 9-pin sub D male with screw fixing
- at the communication partner: 9-pin sub D male with screw fixing

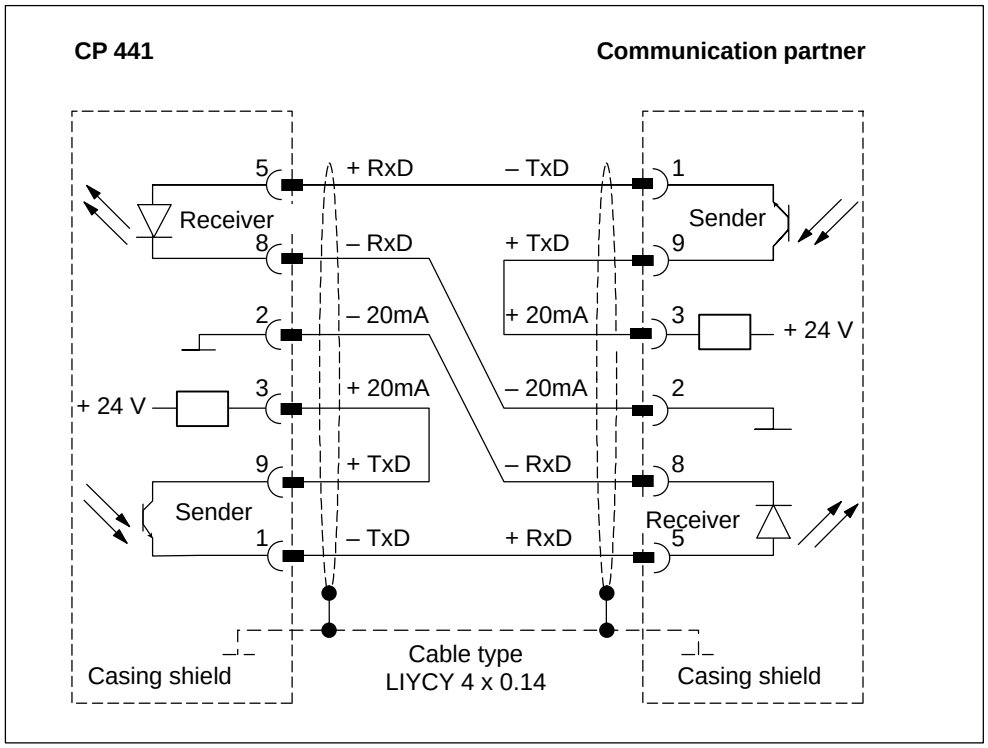


Figure B-8 20 mA TTY Connecting Cable CP 441 – CP 441/CP 340

The connecting cable is available under the order number (6ES7 902-2...) specified in Appendix E.

Note

This cable type (LIYCY 4 x 0.14) can be used in the following lengths for the CP 441 as communication partner:

- Max. 1000 m at 9600 baud
- Max. 500 m at 19.2 kbaud

20 mA TTY Connecting Cable (S7/M7 (CP 441) – CP 544, CP 524, CPU 928B, CPU 945, CPU 948)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 544, CP 524, CPU 928B, CPU 945 or CPU 948.

For the connecting cables you require the following male connectors:

- At CP 441: 9-pin sub D male with screw fixing
- at the communication partner: 25-pin sub D male with clip fixing

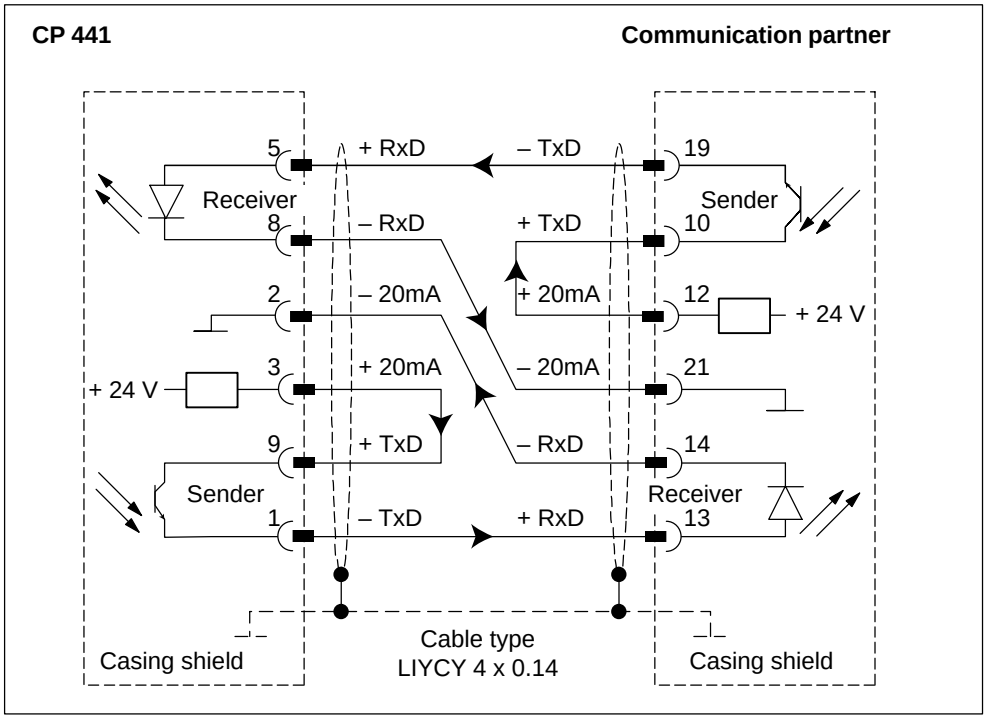


Figure B-9 20 mA TTY Connecting Cable CP 441 – CP 544, CP 524, CPU 928B, CPU 945, CPU 948

20 mA TTY Connecting Cables (S7/M7 (CP 441) – CP 523)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 523.

For the connecting cables you require the following male connectors:

- At CP 441: 9-pin sub D male with screw fixing
- at the communication partner: 25-pin sub D male with screw fixing

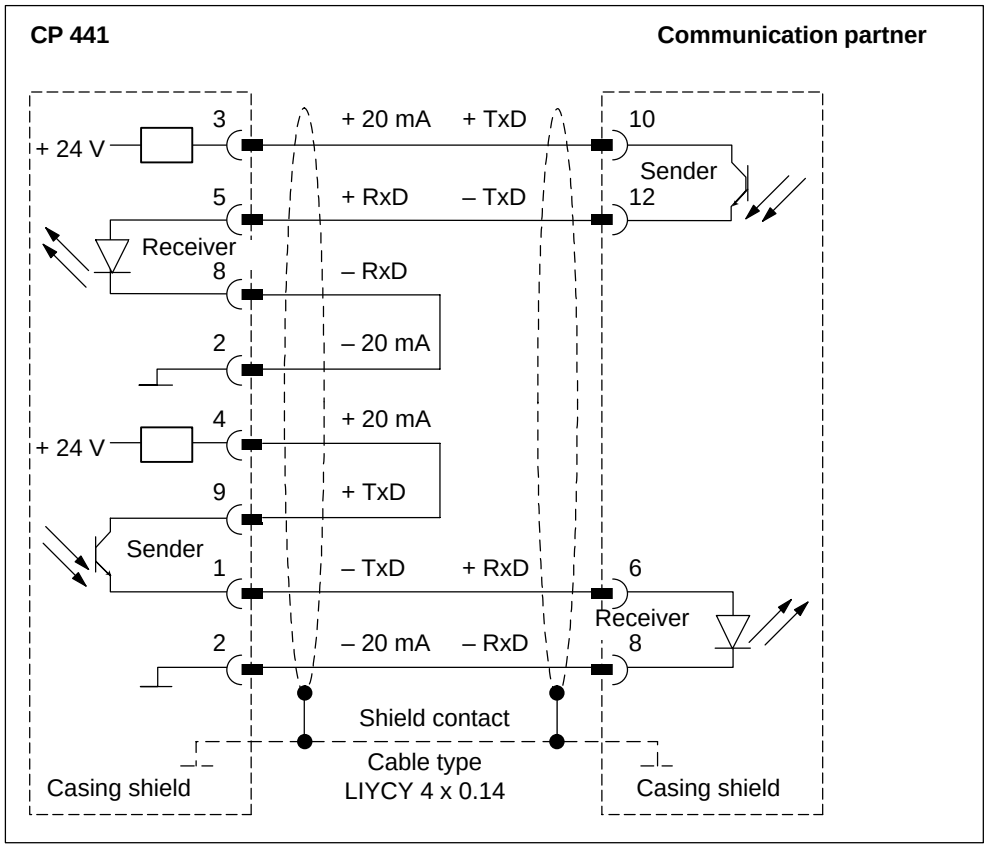


Figure B-10 20 mA TTY Connecting Cable CP 441 – CP 523

20 mA TTY Connecting Cable (S7/M7 (CP 441) – CP 521 SI/CP 521 BASIC/ IBM-Compatible Printer)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 521 SI/CP 521 BASIC.

For the connecting cables you require the following male connectors:

- At CP 441: 9-pin sub D male with screw fixing
- at the communication partner: 25-pin sub D male with screw fixing

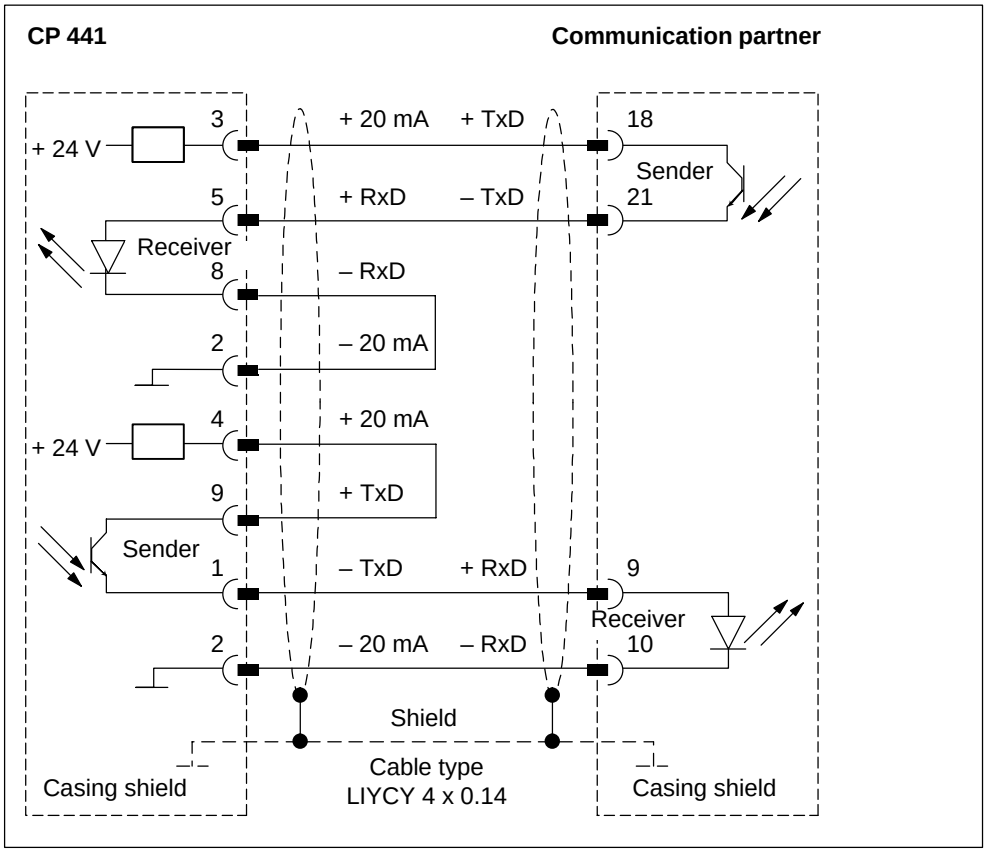


Figure B-11 20 mA TTY Connecting Cable CP 441 – CP 521SI/CP 521BASIC

20 mA TTY Connecting Cables (S7/M7 (CP 441) – CPU 944/AG 95)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CPU 944/AG 95.

For the connecting cables you require the following male connectors:

- At CP 441: 9-pin sub D male with screw fixing
- at the communication partner: 15-pin sub D male with clip fixing

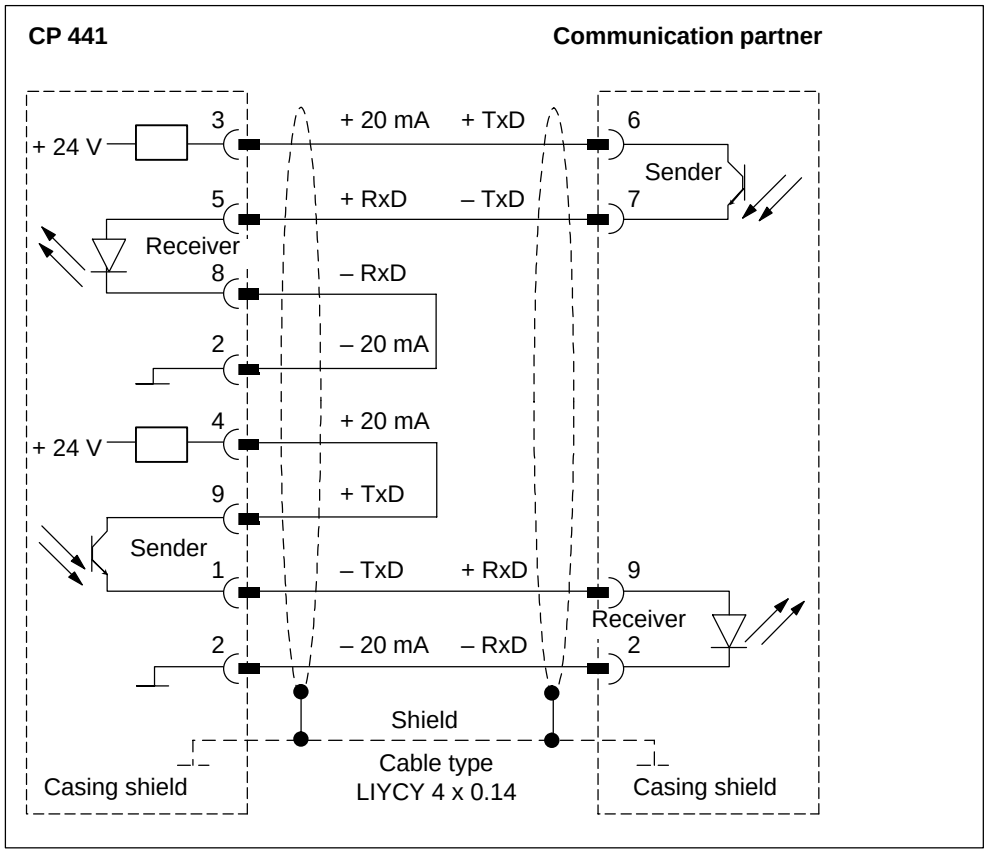


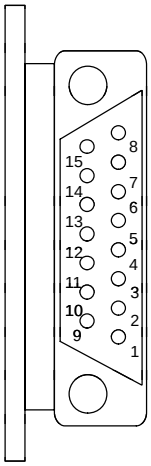
Figure B-12 20 mA TTY Connecting Cable CP 441 – CPU 944/AG 95

B.3 Interface Submodule X27 (RS 422/485)

Pin Allocation

The table below shows the pin allocation for the 15-pin sub D female connector in the front panel of the X27 interface submodule.

Table B-3 Pin Allocation for the 15-Pin sub D Female Connector of the X27 Interface Submodule

Female Connector on Module*	Pin	Designation	Input/Output	Meaning
	1	–	–	–
	2	T (A)–	Output	Transmitted data (four-wire operation)
	3	–	–	–
	4	R (A)/T (A)–	Input Input/Output	Received data (four-wire operation) Received/transmitted data (two-wire operation)
	5	–	–	–
	6	–	–	–
	7	–	–	–
	8	GND	–	Functional ground (floating)
	9	T (B)+	Output	Transmitted data (four-wire operation)
	10	–	–	–
	11	R (B)/T (B)+	Input Input/Output	Received data (four-wire operation) Received/transmitted data (two-wire operation)
	12	–	–	–
	13	–	–	–
	14	–	–	–
	15	–	–	–

* Front view

Connecting Cables

If you construct your own connecting cables you must remember that unconnected inputs at the communication partner may have to be connected to open-circuit potential.

Note that you must only use shielded connector casings. A large surface area of both sides of the cable shield must be in contact with the connector casing and the shield contact.

Caution

Never connect the cable shield with the GND, as this could destroy the interface submodules.

GND (pin 8) must always be connected on both sides, otherwise the interface submodules could be destroyed.

The following pages contain examples of connecting cables for a point-to-point connection between the CP 441 and S7 modules or SIMATIC S5.

X 27 Connecting Cable (S7/M7 (CP 441) – CP 441/CP 340)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 441/CP 340 for RS 422 operation.

For the connecting cables you require the following male connectors:

- At CP 441: 15-pin sub D male with screw fixing
- at the communication partner: 15-pin sub D male with screw fixing

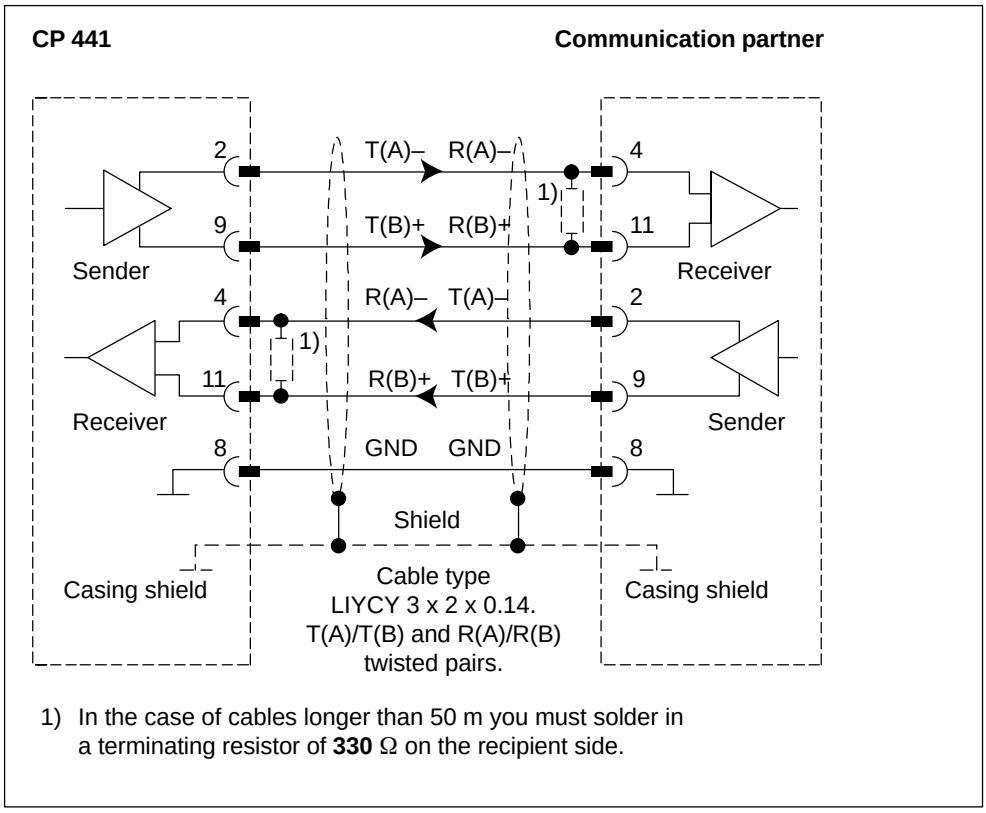


Figure B-13 X27 Connecting Cable CP 441 – CP 441/CP 340 for RS 422 Operation (Four-Wire Mode)

The connecting cable is available using the order number (6ES7 902-3...) specified in Appendix E.

Note

This cable type can be used in the following lengths for the CP 441 as communication partner:

- Max. 1200 m at 19 200 baud
- Max. 500 m at 38 400 baud
- Max. 250 m at 76 800 baud
- Max. 200 m at 115 200 baud

X 27 Connecting Cable (S7/M7 (CP 441) – CP 441/CP 340)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 441/CP 340 for RS 485 operation.

For the connecting cables you require the following male connectors:

- At CP 441: 15-pin sub D male with screw fixing
- at the communication partner: 15-pin sub D male with screw fixing

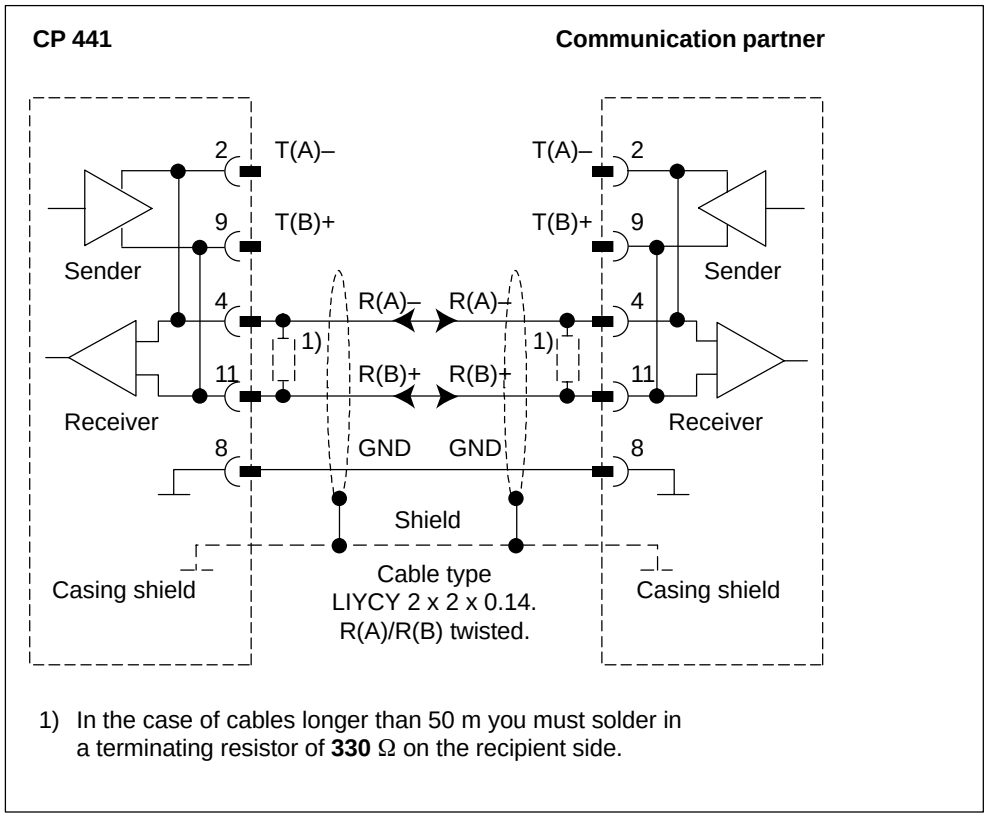


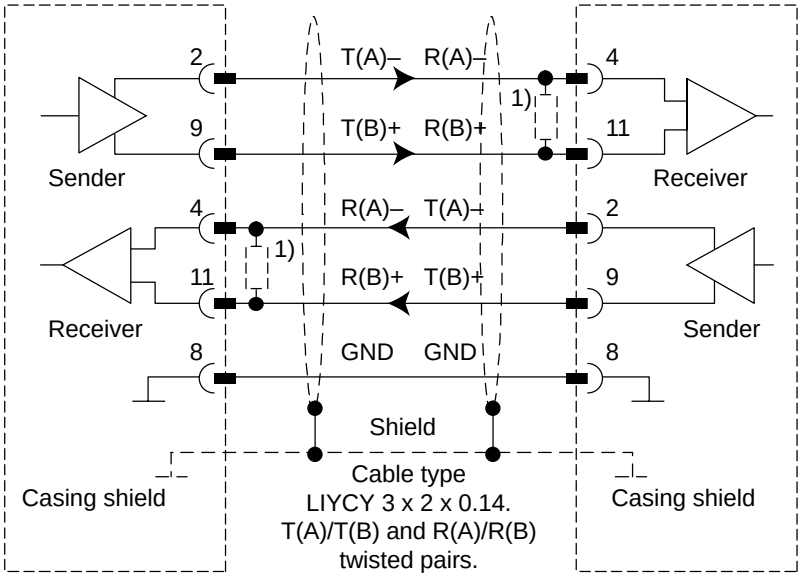
Figure B-14 X27 Connecting Cable CP 441 – CP 340/CP 441 for RS 485 Operation (Two-Wire Mode)

X 27 Connecting Cable (S7/M7 (CP 441) – CP 544, CP 524, CPU 928B, CPU 945, CPU 948)

The figure below illustrates the connecting cable for a point-to-point connection between a CP 441 and a CP 544, CP 524, CPU 928B, CPU 945, CPU 948 for RS 422 operation.

For the connecting cables you require the following male connectors:

- At CP 441: 15-pin sub D male with screw fixing
- at the communication partner: 15-pin sub D male with clip fixing



1) In the case of cables longer than 50 m you must solder in a terminating resistor of **330** Ω on the recipient side.

Figure B-15 X27 Connecting Cable CP 441 – CP 544, CP 524, CPU 928B, CPU 945, CPU 948 for RS 422 Operation (Four-Wire Mode)

SFB Parameters

C

For a detailed description of the system function blocks, see the reference manual *System Software for S7-300 and S7-400, System and Standard Functions*.

Error Messages

Each system function block has a STATUS parameter for error diagnostics. The STATUS message numbers always have the same meaning, irrespective of which system function block is used. The possible STATUS message numbers are listed in Section 8.3.

SFB Parameters

The following tables provides a brief description of the parameters of the system function blocks.

BSEND System Function Block (SFB 12)			
Parameter	**Type**	**Type**	**Meaning**
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge
R	VAR_INPUT	BOOL	Activates resetting of BSEND to initial state with positive edge
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner
R_ID	VAR_INPUT	DWORD	Unique block relationship in a communication connection
SD_1	VAR_IN_OUT	ANY	Data to be sent
LEN	VAR_IN_OUT	WORD	Length of data block to be transmitted
DONE	VAR_OUTPUT	BOOL	Signals successful completion of BSEND request with positive edge
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning

BRCV System Function Block (SFB 13)			
Parameter	**Type**	**Type**	**Meaning**
EN_R	VAR_INPUT	BOOL	Positive edge signals that remote communication partner is ready to receive
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner
R_ID	VAR_INPUT	DWORD	Unique block relationship in a communication connection
RD_1	VAR_IN_OUT	ANY	Data to be received
LEN	VAR_IN_OUT	WORD	Length of data block to be transmitted
NDR	VAR_OUTPUT	BOOL	Positive edge indicates that new receive data is available to the user program
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning

GET System Function Block (SFB 14)			
Parameter	**Type**	**Type**	**Meaning**
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner
ADDR_1 ... ADDR_4	VAR_IN_OUT	ANY	Pointer to the data areas in the partner CPU to be fetched
RD_1 ... RD_4	VAR_IN_OUT	ANY	Pointer to the data areas of the local CPU in which the fetched data is placed.
NDR	VAR_OUTPUT	BOOL	Positive edge indicates that new receive data is available to the user program
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning

PUT System Function Block (SFB 15)			
Parameter	**Type**	**Type**	**Meaning**
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner
DONE	VAR_OUTPUT	BOOL	Signals successful completion of PUT request with positive edge
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning
ADDR_1 ... ADDR_4	VAR_IN_OUT	ANY	Pointer to the data areas in the partner CPU into which data will be written.
SD_1 ... SD_4	VAR_IN_OUT	ANY	Pointer to the data areas of the local CPU which contain the data to be sent.

PRINT System Function Block (SFB 16)			
Parameter	**Type**	**Type**	**Meaning**
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner
PRN_NR	VAR_IN_OUT	BYTE	Selects a specific printer when several are connected
FORMAT	VAR_IN_OUT	STRING	Format string for the message text, including the conversion and control statements for variables SD_1 to SD_4
SD_1 ... SD_4	VAR_IN_OUT	ANY	Variables in the message text in order, for example, to display computed values of the user program or dates and times
DONE	VAR_OUTPUT	BOOL	Indicates at a rising edge the error-free completion of the PRINT request
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning

STATUS System Function Block (SFB 22)			
REQ	VAR_INPUT	BOOL	Activates a transfer at positive edge
ID	VAR_INPUT	WORD	Unique communication connection to a communication partner
NDR	VAR_OUTPUT	BOOL	Positive edge indicates that new receive data is available to the user program
ERROR	VAR_OUTPUT	BOOL	Positive edge indicates error
STATUS	VAR_OUTPUT	WORD	Contains detailed error message or warning
PHYS	VAR_IN_OUT	ANY	Logical device status

Communication Matrix of the Protocols

D

The CP 441 communication processor can communicate with the following CPs and CPUs of the SIMATIC S5 and SIMATIC S7 programmable controllers.

Communication Matrix 3964(R)

The diagram below shows the communication matrix of the 3964(R) procedure.

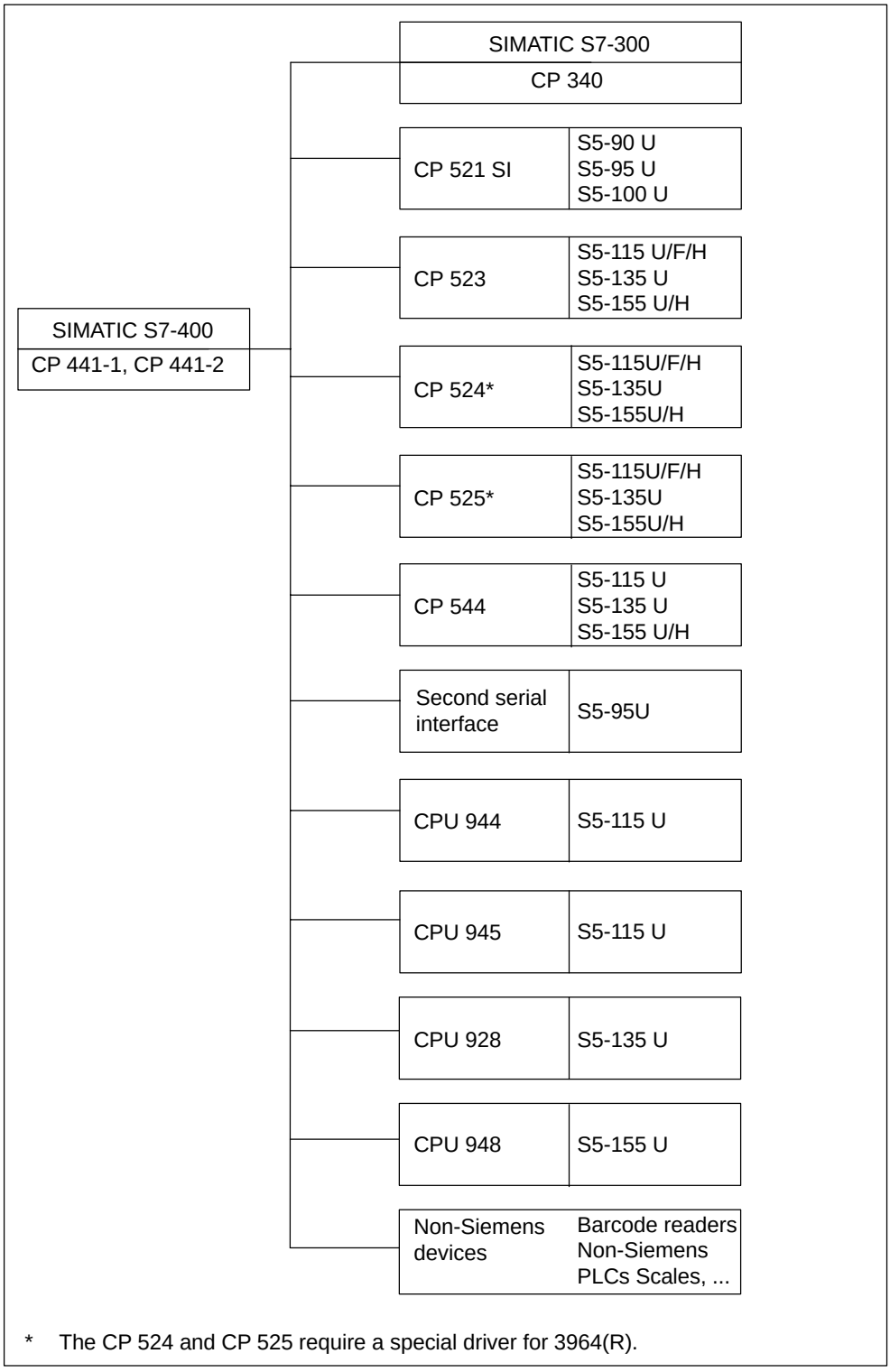


Figure D-1 Communication Matrix of the 3964(R) Procedure

Communication Matrix RK 512

The diagram below shows the communication matrix of the RK 512 computer connection.

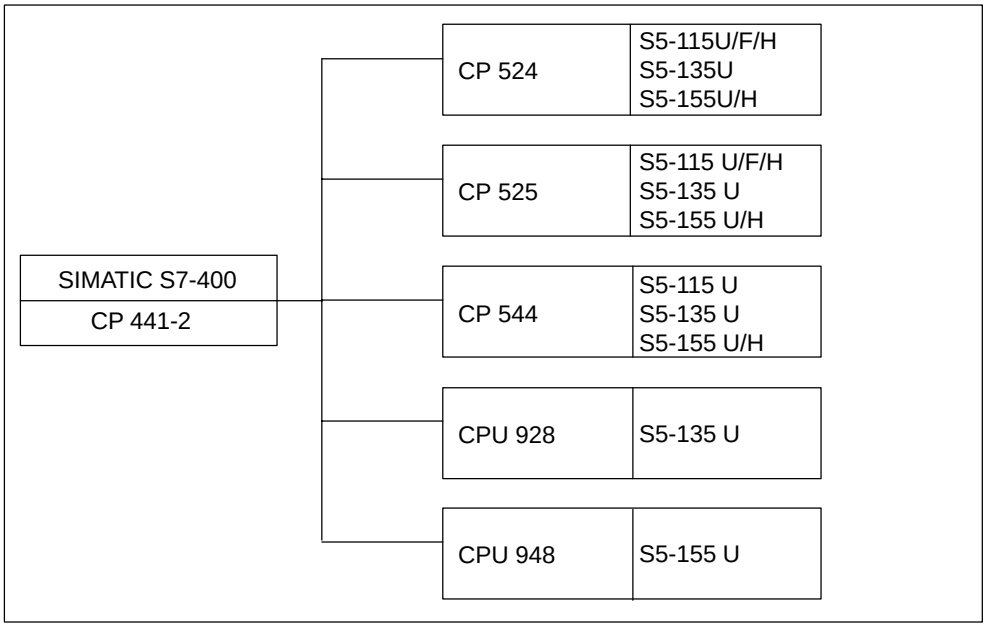


Figure D-2 Communication Matrix of the RK 512 Computer Connection.

Communication Matrix ASCII Driver

The figure below shows the communication matrix of the ASCII driver.

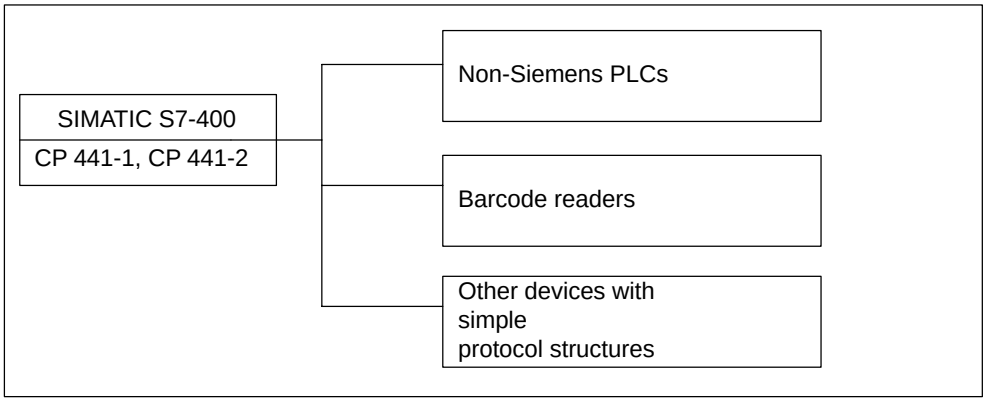


Figure D-3 Communication Matrix of the ASCII Driver

Communication Matrix Printer Driver

The figure below shows the communication matrix of the printer driver.

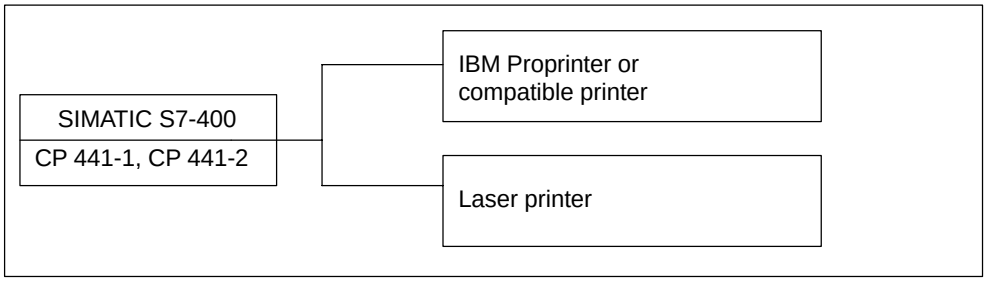


Figure D-4 Communication Matrix of the Printer Driver

Accessories and Order Numbers

E

Accessories and Order Numbers

The following is an overview of the accessories for the CP 441:

Product	Order Number
CP 441-1	6ES7 441-1AA03-0AE0
CP 441-2	6ES7 441-2AA03-0AE0
Interface submodule:	
• RS 232C module	6ES7 963-1AA00-0AA0
• 20 mA TTY module	6ES7 963-2AA00-0AA0
• X27 (RS 422/485) module	6ES7 963-3AA00-0AA0
Connecting cable (CP 441 – CP 441/CP 340) as in Figure B-1	
• RS 232C, 5 m	6ES7 902-1AB00-0AA0
• RS 232C, 10 m	6ES7 902-1AC00-0AA0
As in Figure B-8:	
• 20 mA TTY, 5 m	6ES7 902-2AB00-0AA0
• 20 mA TTY, 10 m	6ES7 902-2AC00-0AA0
• 20 mA TTY, 50 m	6ES7 902-2AG00-0AA0
As in Figure B-13:	
• X27 (RS 422), 5 m	6ES7 902-3AB00-0AA0
• X27 (RS 422), 10 m	6ES7 902-3AC00-0AA0
• X27 (RS 422), 50 m	6ES7 902-3AG00-0AA0

SIMATIC S7 Reference Literature

F

Literature Referenced in This Manual

/1/ *Programming with STEP 7 V5.1*
Manual

/2/ *S7-400/M7-400 Programmable Controllers, Hardware and Installation* Manual

/3/ *Configuring Hardware and Communication Connections STEP 7 V5.1* Manual

/4/ *System Software for S7-300 and S7-400, System and Standard Functions* Reference Manual

Literature on SIMATIC S7

On the following pages, you will find a comprehensive overview of:

- manuals that you require for configuring and programming the S7-400,
- manuals which describe the components of a PROFIBUS DP network,
- technical overviews which provide you with an overview of the SIMATIC S7 and *STEP 7*.

Manuals for Configuring and Starting Up

An extensive user documentation is available to assist you in configuring and programming the S7-400. You can select and use this documentation as required. Table F-1 lists also documentation for *STEP 7*.

Table F-1 Manuals for Configuring and Programming the S7-400

Title	Contents
Programming with STEP 7 V5.1 *Manual*	The programming manual offers basic information on the design of the operating system and a user program of an S7 CPU. For novice users of an S7-300/400 it provides an overview of the programming principles on which the design of user programs is based.
Configuring Hardware and Communication Connections STEP 7 V5.1 *Manual*	The STEP 7 user manual explains the principles for using the STEP 7 automation software and its functions. Novice users of STEP 7 as well as experienced users of STEP 5 are provided with an overview of the configuring, programming and start-up procedures for an S7-300/400. When working with the software, an on-line help assists you if you require detailed information on the software.
Statement List (STL) for S7-300 and S7-400 *ReferenceManual*	The manuals for the STL, LAD, FDB and SCL packages each comprise the user manual and the language description. For programming an S7-300/400 you need only one of the languages, but, if required, you can switch between the language to be used in a project. If it is the first time that you use one of the languages, the manuals will help you in getting familiar with the programming principles. When working with the software, you can use the on-line help, which provides you with detailed information on editors and compilers.
Ladder Logic (LAD) for S7-300 and S7-400 *Reference Manual*	
Function Block Diagram (FDB) for S7-300 and S7-400 *Reference Manual*	
Structured Control Language (SCL)[1] for S7-300 and S7-400 *Reference Manual*	
S7-GRAPH[1] for S7-300 and S7-400 Programming Sequential Control Systems *Manual*	With the GRAPH, HiGraph, CFC languages, you can implement sequential function charts, state diagrams or graphic interconnections of blocks. Each of the manuals comprises a user manual and a language description. If it is the first time that you use one of these languages, the manual will help you in getting familiar with the programming principles. When working with the software, you can also use the on-line help (not for HiGraph), which provides you with detailed information on editors and compilers.
S7-HiGraph[1] for S7-300 and S7-400 Programming State Graphs *Manual*	
Continuous Function Charts[1] for S7 and M7 *Manual*	
System Software for S7-300 and S7-400 *Systems and Standard Functions* *Reference Manual*	The S7 CPU's offer systems and standard functions which are integrated in the operating system. You can use these functions when writing programs in one of the languages, that is STL, LAD and SCL. The manual provides an overview of the functions available with S7 and, for reference purposes, detailed interface descriptions which you require in your user program.

[1] Optional packages for S7-300/400 system software

Manuals for PROFIBUS-DP

For the configuration and startup of a PROFIBUS-DP network, you will need the descriptions of the other nodes and network components integrated in the network. For this purpose, you can order the manuals listed in Table F-2.

Table F-2 Manuals for PROFIBUS-DP

Manual
ET 200M Distributed I/O Station
SINEC L2-DP Interface of the S5-95U Programmable Controller
ET 200B Distributed I/O Station
ET 200C Distributed I/O Station
ET 200U Distributed I/O Station
ET 200 Handheld Unit
SINEC L2/L2FO-Network Components

Glossary

A

Address

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

B

Block

Blocks are elements of the user program which are defined by their function, structure, or purpose. With STEP 7 there are

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

Block Call

A block call occurs when program processing branches to the called block.

Block Parameter

Block parameters are wildcards within multiple-use blocks, which are replaced with current values when the relevant block is called.

C

Communications Processor

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

Configuration

The configuration is the setup of individual modules of the PLC in the configuration table.

CPU

Central processing unit of the S7 programmable controller with control and arithmetic unit, memory, operating system, and interfaces to I/O modules.

Cycle Time

The cycle time is the time the CPU needs to scan the user program once.

Cyclic Program Processing

In cyclic program processing, the user program is executed in a constantly repeating program loop, called a cycle.

D

Data Block (DB)

These are blocks containing data and parameters with which the user program works. Unlike all other blocks, data blocks do not contain instructions. They are subdivided into global data blocks and instance data blocks. The data held in the data blocks can be accessed absolutely or symbolically. Complex data can be stored in structured form.

Data Type

Data types allow users to define how the value of a variable or constant is to be used in the user program. They are subdivided into elementary and structured data types.

Default Setting

The default setting is a practical basic setting which is always used if no other value is specified.

Diagnostic Events

Diagnostic events are, for example, errors on a module or system errors in the CPU, which are caused by, say, a program error or by operating mode transitions.

Diagnostic Buffer

Every CPU has a diagnostic buffer, in which detailed information on diagnostic events is stored in the order in which they occur.

The CP 340 has its own diagnostic buffer, in which all the diagnostic events of the CP 340 are entered (hardware/firmware errors, initialization/parameterization errors, send and receive errors).

Diagnostics Functions

The diagnostics functions cover the entire system diagnosis and include detection, analysis and reporting of errors within the PLC.

Download

Downloading means loading load objects (e.g. code blocks) from the programming device into the load memory of the CPU.

F

Function Block (FB)

Function blocks are components of the user program and, in accordance with the IEC standard, are ”blocks with memory”. The memory for the function block is an assigned data block of the ”instance data block”. Function blocks can be parameterized but can also be used without parameters.

H

Hardware

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

I

Instance Data Block

An instance data block is a block assigned to a function block and contains data for this special function block.

Interrupt

An interrupt occurs when program processing in the processor of a PLC is interrupted by an external alarm.

M

Module

Modules are pluggable printed circuit boards for programmable controllers.

Module Parameter

Module parameters are used to set the module reactions. A distinction is made between static and dynamic module parameters.

O

On-line/Off-line

On-line means that a data circuit exists between PLC and programming device. Off-line means that no such data circuit exists.

On-line Help

STEP 7 allows you to display contextual help texts on the screen while working with the programming software.

Operand

An operand is part of a STEP 7 instruction and states with what the processor is to do something. It can be both absolutely and symbolically addressed.

Operating Mode

The SIMATIC S7 programmable controllers have three different operating modes: STOP, RESTART and RUN. The functionality of the CPUs varies in the individual operating modes.

Operating System of the CPU

The operating system of the CPU organizes all functions and operations of the CPU which are not connected to a specific control task.

P

Parameter

Parameters are values that can be assigned. A distinction is made between block parameters and module parameters.

Parameterization

Parameterization means setting the behavior of a module.

Parameterization Interface *CP 441: Point-to-Point Communication, Parameter Assignment*

The *CP 441: Point-to-Point Communication, Parameter Assignment* parameterization interface is used to parameterize the submodules of the communications processor and configure the message texts for printer output.

Point-to-Point Connection

In a point-to-point connection the communications processor forms the interface between a PLC and a communications partner.

Procedure

The execution of a data interchange operation according to a specific protocol is called a procedure.

Process Image

This is a special memory area in the PLC. At the beginning of the cyclic program, the signal states of the input modules are transferred to the process image input table. At the end of the cyclic program, the process image output table is transferred to the output modules as signal state.

Programmable Controller

Programmable controllers (PLCs) are electronic control devices consisting of at least one central processing unit, various input/output modules, and operator control and monitoring devices.

Protocol

The communications partners involved in a data interchange must abide by fixed rules for handling and implementing the data traffic. These rules are called protocols.

R

Rack

A rack is the rail containing slots for mounting modules.

RESTART

On transition from the STOP to the RUN mode, the PLC goes through the RESTART mode.

S

S7-300 Backplane Bus

The S7-300 backplane bus is a serial data bus via which the modules communicate with each other and are supplied with the necessary voltage.

Software

Software is the term given to all programs used on a computer system. These include the operating system and the user programs.

STEP 7

This is the programming software for SIMATIC S7 programmable controllers.

System Block

System blocks differ from the other blocks in that they are already integrated into the S7-300 system and are available for already defined system functions. They are subdivided into system data blocks, system functions, and system function blocks.

System Function (SFC)

System functions are modules without memory which are already integrated into the operating system of the CPU and can be called up by the user as required.

System Function Block (SFB)

System function blocks are modules with memory which are already integrated into the operating system of the CPU and can be called up by the user as required.

U

Upload

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

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

V

Variable

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

W

Work Memory

The work memory is a RAM on the CPU which the processor accesses while processing the user program.

Index

D

E

F

G

H

I

L

M

O

P

R

S

T

U

Siemens AG
A&D AS E 81

Oestliche Rheinbrueckenstr. 50
D-76181 Karlsruhe
Federal Republic of Germany

From:
Your Name: _
Your Title: _
Company Name: _
Street: _
City, Zip Code_ _
Country: _
Phone: _

Please check any industry that applies to you:

- ❒ Automotive
- ❒ Chemical
- ❒ Electrical Machinery
- ❒ Food
- ❒ Instrument and Control
- ❒ Nonelectrical Machinery
- ❒ Petrochemical
- ❒ Pharmaceutical
- ❒ Plastic
- ❒ Pulp and Paper
- ❒ Textiles
- ❒ Transportation
- ❒ Other _ _ _ _ _ _ _ _ _ _ _

Remarks Form

Your comments and recommendations will help us to improve the quality and usefulness of our publications. Please take the first available opportunity to fill out this questionnaire and return it to Siemens.

Please give each of the following questions your own personal mark within the range from 1 (very good) to 5 (poor).

1. Do the contents meet your requirements? ☐
2. Is the information you need easy to find? ☐
3. Is the text easy to understand? ☐
4. Does the level of technical detail meet your requirements? ☐
5. Please rate the quality of the graphics/tables: ☐

Additional comments:

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _