

## SIMATIC

### S7-400

## Point-to-point connection CP 440 Installation and Parameter Assignment

Equipment Manual

### Preface

---

Product Description

1

Basic Principles of Serial  
Data Transmission

2

Commissioning the CP 440

3

Mounting the CP 440

4

Configuring and Parameter  
Assignment the CP 440

5

Communication via  
Function Blocks

6

Startup Characteristics and  
Operating Mode Transitions  
of the CP 440

7

Diagnostics with the CP 440

8

Programming Example for  
Standard Function Blocks

9

Technical specifications

10

Connecting cables

11

Accessories and order  
numbers

12

Literature

13

## Legal information

### Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

<b>⚠ DANGER</b>
indicates that death or severe personal injury <b>will</b> result if proper precautions are not taken.

<b>⚠ WARNING</b>
indicates that death or severe personal injury <b>may</b> result if proper precautions are not taken.

<b>⚠ CAUTION</b>
indicates that minor personal injury can result if proper precautions are not taken.

<b>NOTICE</b>
indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

### Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

### Proper use of Siemens products

Note the following:

<b>⚠ WARNING</b>
Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

### Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

### Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

# Preface

## Preface

### Purpose of This Manual

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

### Contents of the manual

The manual describes the hardware and software of the CP 440 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 440
- Commissioning the CP 440
- Mounting the CP 440
- Communication via the CP 440
- Troubleshooting
- Application examples
- Properties and technical specifications

### Scope of the manual

This manual is valid for the "CP 440 with X27 RS 422/485 interface", order number: "6ES7 440-1CS00-0YE0" as of version "01".

---

#### Note

The description of the CP 440 communication processor 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.

---

### Recycling and Disposal

The CP 440 is an environment-friendly product. It's characteristic features include:

- Halogen-free plastics used for housing, which nonetheless has a high degree of fire resistance
- Laser inscriptions (i.e. no labels)

- Plastics identification in accordance with DIN EN ISO 11469
- Less material used because the unit is smaller and with fewer components thanks to integration in ASICs

The CP 440 is suitable for recycling on account of the low level of contaminants in its components. Please contact a certified waste disposal company for eco-friendly recycling and to dispose of your old devices.

## Structure of This Manual

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

- At the beginning of the manual you can find a comprehensive list of contents.
- Finally, a comprehensive index allows quick access to information on specific subjects.

## Additional assistance

Please contact your local Siemens representative if you have any queries about the products described in this manual.

- Find your contact partner at:  
<http://www.siemens.de/automation/partner> (<http://www.siemens.com/automation/partner>)
- You will find the guide to the technical documentation for the individual SIMATIC products and systems at:  
<http://www.siemens.de/simatic-doku> (<http://www.siemens.com/simatic-doku>)
- The online catalog and the online ordering system are available at:  
<http://www.siemens.de/automation/mall> (<http://www.siemens.com/automation/mall>)

## Training Center

We offer a range of courses to help get you started with the S7 automation system. Please contact your local training center or the central training center in Nuremberg, D-90327 Germany.

- Internet: <http://www.siemens.com/sitrain> (<http://www.siemens.com/sitrain>)

## Technical support

Here you can access Technical Support for all A&D products

- Use the Web form for the support request  
<http://www.siemens.com/automation/support-request> (<http://www.siemens.com/automation/support-request>)

Additional information about our technical support is available in the Internet at:

<http://www.siemens.de/automation/service&support> (<http://www.siemens.com/automation/service&support>)

## Service & Support on the Internet

In addition to our documentation, we offer a comprehensive knowledge base online on the Internet.

<http://www.siemens.de/automation/service&support> (<http://www.siemens.com/automation/service&support>)

### There you will find:

- The newsletter, which provides the latest information on your products.
- The documents you require, using our Service & Support search engine.
- A forum where users and specialists exchange information worldwide.
- Your local service partner for Automation & Drives in our contact database.
- Information about on-site service, repairs and spare parts. And much more is available under "Services".



# Table of contents

	<b>Preface .....</b>	<b>3</b>
<b>1</b>	<b>Product Description .....</b>	<b>11</b>
1.1	Uses of the CP 440 .....	11
1.2	Components Required for a Point-to-Point Connection with the CP 440.....	12
1.2.1	Required Hardware Components .....	12
1.2.2	Required Software Components .....	13
1.2.3	Incompatible CPU versions .....	13
1.3	Design of the CP 440.....	14
1.4	Features of the X27 (RS 422/485) Interface .....	16
<b>2</b>	<b>Basic Principles of Serial Data Transmission .....</b>	<b>17</b>
2.1	Serial Transmission of a Character .....	17
2.2	Transmission Procedure with a Point-to-Point Connection .....	20
2.3	Transmission integrity .....	21
2.4	Data Transmission with the ASCII Driver.....	23
2.4.1	Data Transmission with the ASCII Driver.....	23
2.4.2	Sending Data with the ASCII Driver .....	24
2.4.3	Receiving Data with the ASCII Driver .....	26
2.4.4	Topologies Between the Communication Partners .....	32
2.5	Data Transmission with the 3964(R) Procedure.....	35
2.5.1	The Control Characters of the 3964(R) Procedure .....	35
2.5.2	Block Checksum.....	37
2.5.3	Sending Data with 3964(R) .....	38
2.5.4	Receiving Data with 3964(R) .....	41
2.5.5	Handling Corrupt Data .....	46
<b>3</b>	<b>Commissioning the CP 440 .....</b>	<b>49</b>
<b>4</b>	<b>Mounting the CP 440 .....</b>	<b>53</b>
4.1	CP 440 slots .....	53
4.2	Mounting and Dismounting the CP 440.....	53
4.2.1	Installation steps.....	54
4.2.2	Removal steps.....	54
<b>5</b>	<b>Configuring and Parameter Assignment the CP 440 .....</b>	<b>55</b>
5.1	Assigning parameters to the CP 440 .....	55
5.2	Installing the Programming Interface .....	56
5.3	Parameters for the Communications Protocols.....	56
5.4	Configuration data .....	57
5.4.1	Introduction.....	57

5.4.2	Basic Parameters of the CP 440 .....	57
5.4.3	Configuration Data of the ASCII Driver .....	58
5.4.4	Configuration Data of the 3964(R) Procedure .....	63
5.5	Managing the Parameter Data.....	68
5.6	Firmware updates .....	69
5.6.1	Subsequent Loading of Firmware Updates .....	69
5.6.2	Viewing the Firmware Version.....	71
<b>6</b>	<b>Communication via Function Blocks.....</b>	<b>73</b>
6.1	Overview of the Function Blocks .....	73
6.2	Notes on Program Structure .....	75
6.3	Using the Function Blocks .....	75
6.3.1	S7 Transmits Data to a Communication Partner, 10 SEND_440 FB .....	76
6.3.2	S7 Receives Data from a Communication Partner, 9 RECV_440 FB .....	79
6.3.3	Deleting the Receive Buffer (11 "RES_RECV" FB) .....	82
6.4	Programming the Function Blocks .....	85
6.4.1	Supplying the block parameters .....	85
6.5	General Information on Program Processing .....	88
6.6	Technical Specifications of the Function Blocks .....	89
<b>7</b>	<b>Startup Characteristics and Operating Mode Transitions of the CP 440 .....</b>	<b>91</b>
7.1	Operating Modes of the CP 440.....	91
7.2	Startup Characteristics of the CP 440.....	91
7.3	Behavior of the CP 440 on Operating Mode Transitions of the CPU.....	92
7.4	Behavior of the Sender Line Drivers of the Serial Interface During Particular Operating Modes of the CP 440.....	93
<b>8</b>	<b>Diagnostics with the CP 440 .....</b>	<b>95</b>
8.1	Diagnostics Functions of the CP 440.....	95
8.2	Diagnostics via the display elements of the CP 440 .....	96
8.3	Diagnostics Messages of the Function Blocks.....	97
8.4	Diagnostics via the diagnostic buffer of the CP 440 .....	104
<b>9</b>	<b>Programming Example for Standard Function Blocks .....</b>	<b>107</b>
9.1	General Information.....	107
9.2	Device Configuration.....	107
9.3	Settings .....	108
9.4	Blocks Used .....	108
9.5	Installation, Error Messages.....	111
9.6	Activation, Startup Program and Cyclic Program .....	111
9.6.1	"CP440 SEND RECV" Program Example .....	111
9.6.2	"CP440 1 CYC" Program Example .....	113
9.6.3	"CP440 ASCII BCC" Program Example .....	113



---

9.6.4	"CP440 MASTER" Program Example .....	114
9.6.5	"CP440 SLAVE" Program Example .....	115
<b>10</b>	<b>Technical specifications .....</b>	<b>117</b>
10.1	Technical Specifications of the CP 440 .....	117
10.2	Standards and approvals .....	120
10.2.1	Currently valid markings and approvals .....	120
10.2.2	CE approval .....	121
10.2.3	CCC approval .....	122
10.2.4	UKCA approval .....	122
10.2.5	Explosion protection .....	123
10.2.6	cULus approval .....	124
10.2.7	cULus HAZ. LOC. approval .....	124
10.2.8	cFMus approval.....	124
10.2.9	Approval for Australia and New Zealand .....	125
10.2.10	Approval for the Korea and South Korea .....	125
<b>11</b>	<b>Connecting cables .....</b>	<b>127</b>
11.1	X27 (RS 422/485) Interface of the CP 440.....	127
11.2	Cables .....	127
<b>12</b>	<b>Accessories and order numbers.....</b>	<b>133</b>
12.1	Accessories and Order Numbers .....	133
<b>13</b>	<b>Literature .....</b>	<b>135</b>
13.1	Literature on SIMATIC S7 .....	135
	<b>Index.....</b>	<b>137</b>



# Product Description

## 1.1 Uses of the CP 440

### Applications

The CP 440 communication processor enables you to exchange data between automation devices or computers by means of a point-to-point connection. The CP 440 is designed to transfer short, fast frames.

The following are typically connected to it:

- Scanners, barcode readers
- Sensors
- Weighing scales

### Functionality of the CP 440

The CP 440 communication processor provides the following functionality:

- An integrated MPI (Multipoint) X27 (RS422/485) interface
- Maximum transmission length 400 bytes (also see chapter "Overview of the Function Blocks (Page 73)")
- A transmission rate of up to 115.2 kbps, full-duplex
- Integration of the most important transmission protocols in the module firmware
  - ASCII driver
  - 3964(R) Procedure
- Customization of the transmission protocols by means of parameter assignment with the **CP 440: Point-to-Point Communication, Parameter Assignment** interface

### Uses of the CP 440

The CP 440 communication processor allows point-to-point communication with SIMATIC modules and with non-Siemens products.

### Functions Supported by the Interfaces

Table 1-1 Functions of the CP 440

Function	CP 440	
	RS 422*	RS 485*
3964(R) Procedure	Yes	No
ASCII driver:	Yes	Yes
• XON/XOFF flow control	Yes	No
* The RS 422 and RS 485 are distinguished by their parameter configuration.		

## 1.2 Components Required for a Point-to-Point Connection with the CP 440

### 1.2.1 Required Hardware Components

#### Hardware Components

The following table lists the hardware components required for establishing a point-to-point connection with the CP 440.

Table 1-2 Hardware Components for a Point-to-Point Connection with the CP 440



Components	Function	
Rack	... provides the mechanical and electrical connections of the S7-400.	
Power supply module (PS)	... converts the line voltage (120/230 V AC or 24 V DC) into the operating voltage of 24 V and 5 V DC required to supply the S7-400.	
Central processor unit (CPU) Some CPU versions cannot be used with the CP 440. Accessories: Memory card, backup battery	... executes the user program; communicates via the MPI interface with other CPUs or with a programming device.	
CP 440 communication processor	... communicates via the interface with one or more communication partners.	
Standard cable	... connects the CP 440 communication processor to the communication partner.	
Programming device cable	... connects a CPU to a programming device/PC.	
Programming device or PC	... communicates with the CPU of the S7-400.	

## 1.2.2 Required Software Components

### Software components

The following table lists the software components required for establishing a point-to-point connection with the CP 440.

Table 1-3 Software Components for a Point-to-Point Connection with the CP 440

Components	Function	Diagram	License
STEP 7 software package	... configures, assigns parameters, programs and tests the S7-400.		
Parameter assignment interface: <b>Point-to-Point Communication, Parameter Assignment</b> interface, Version 5.1	... assigns parameters for the interfaces of the CP 440.		
Function blocks (FBs) with programming examples	... controls the communication between CPU and CP 440.		

## 1.2.3 Incompatible CPU versions

### CPU versions

The CP 440 can be operated with all CPU versions except the CPUs listed in the tables below:

Table 1-4 CPU Versions with Which the CP 440 Can Be Used as of the Version Indicated

CPU	Order number
CPU 412-1	6ES7 412-1XF01-0AB0, Release 5
CPU 413-1	6ES7 413-1XG01-0AB0, Release 5
CPU 413-2	6ES7 413-2XG01-0AB0, Release 5
CPU 414-1	6ES7 414-1XG01-0AB0, Release 5
CPU 414-2 with 128k	6ES7 414-2XG01-0AB0, Release 5
CPU 414-2 with 348k	6ES7 414-2XJ00-0AB0, Release 7
CPU 416-1	6ES7 416-1XJ01-0AB0, Release 5
CPU 416-2 with 0.8 M	6ES7 416-2XK00-0AB0, Release 7
CPU 416-2 with 1.6 M	6ES7 416-2XL00-0AB0, Release 7
CPU 416-2 DP ISA Lite CPU 416-2 DP ISA CPU 412-2 DP PCI CPU 416-2 DP PCI	6ES7 616-2PK00-0AB4, Release 3

Table 1-5 CPU Versions with Which the CP 440 Cannot be Used

<b>CPU</b>	<b>Order number</b>
CPU 412-1	6ES7 412-1XF00-0AB0
CPU 413-1	6ES7 413-1XG00-0AB0
CPU 413-2	6ES7 413-2XG00-0AB0
CPU 414-1	6ES7 414-1XG00-0AB0
CPU 414-2 with 128k	6ES7 414-2XG00-0AB0
CPU 416-1	6ES7 416-1XJ00-0AB0

## 1.3 Design of the CP 440

### Interface

The CP 440 communication processor is supplied with an integrated serial X27 (RS422/485) interface.

## Position of Operator Control and Display Elements

The figure shows the arrangement of the controls and indicators on the front panel of the CP 440 communication processor.

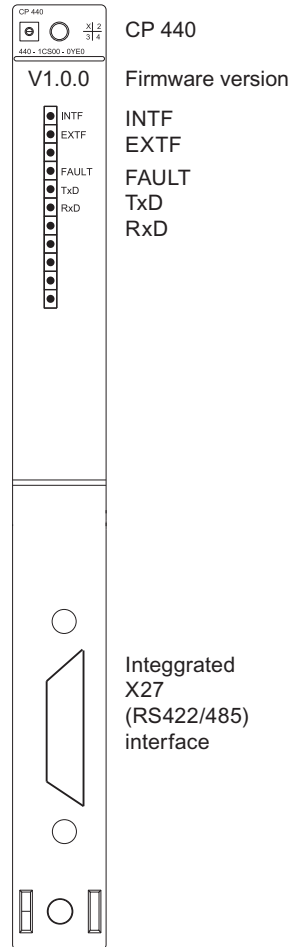


Figure 1-1 Arrangement of the Controls and Indicators on the CP 440 Communication Processor

## LEDs

On the front panel of the

- INTF (red) indicates an internal error
- EXTF (red) indicates an external error
- FAULT (red) Fault display for interface
- TXD (green) Interface sending
- RXD (green) Interface receiving

## Base Connector for S7 Backplane Bus

On the back panel of the CP 440 you will find the base connector for the S7-400 backplane bus.

## 1.4 Features of the X27 (RS 422/485) Interface

The S7-400 backplane bus is a serial data bus via which the CP 440 communicates with the modules of the programmable controller.

### 1.4 Features of the X27 (RS 422/485) Interface

#### Definition

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

#### Properties

The X27 (RS 422/485) interface has the following attributes and is in compliance with the following requirements:

- Type: Differential voltage interface
- Front connector: 15-pin sub-D female with screwed interlock
- Max. transmission rate: 115.2 Kbps
- Max. cable length: 1200 m at 19200 Bps
- Standard: DIN 66259 Parts 1 and 3, EIA-RS 422/485, CCITT V.11
- Degree of protection: IP 00

---

#### Note

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

---

#### Cables

Siemens offers standard cables in various lengths for point-to-point connection between the communication processor and a communication partner.

For order information and the length of standard cables are listed in the appendix "AUTOHOTSPOT".

#### Fabricating Your Own Cables

If you are fabricating your own cables, there are a few points to take into consideration. The appendix "AUTOHOTSPOT" of this manual contains information about this topic as well as the pin assignment of the sub D male connector and the wiring diagrams.



# Basic Principles of Serial Data Transmission

## 2.1 Serial Transmission of a Character

### Introduction

The system provides various networking options for the exchange of data between two or more communication partners. The simplest form of data interchange is via a point-to-point connection between two communication partners.

### Point-to-Point Communication

In point-to-point communication the communication processor forms the interface between a programmable controller and a communication partner. Data are transmitted serially in the point-to-point connection with the CP 440.

### Serial Transmission

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

### Drivers for Bidirectional Data Traffic

The CP 440 itself handles data transmission with communication partners via its serial interface. The CP 440 is equipped with two different drivers for this purpose.

#### **Bidirectional data traffic:**

- ASCII driver
- 3964(R) procedure

The CP 440 executes data transfer via the serial interface depending on the selected driver.

### Bidirectional Data Traffic - Operating Modes

The CP 440 has two operating modes for bidirectional data traffic:

- Half-duplex operation (3964(R) procedure, ASCII driver)  
Data is exchanged between the communication partners but only in one direction at a time. 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)  
Data is exchanged between two or more communication partners in both directions simultaneously. In full-duplex operation, therefore, data can be sent and received at the same time. Every communication partner must be able to operate a send and a receive facility simultaneously.

## 2.1 Serial Transmission of a Character

Only half-duplex mode can be used with an RS 485 (2-wire) setting.

### Asynchronous Data Transmission

With the communication processor, serial transmission occurs asynchronously. The so-called time base 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 length of the start-bit transmission determines the clock pulse. 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 sending and receiving partners before serial transmission can take place. These include:

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

### Character Frame

Data is transmitted between the CP 440 and a communication partner via the serial interface in a character frame. Two data formats are available for each character frame. 7 data bits without a parity bit are not supported. You can set the desired format for data transmission with the programming interface **CP 440: Point-to-Point Communication, Parameter Assignment** interface.

By way of example, the figure below shows the two data formats of the 10-bit character frame.

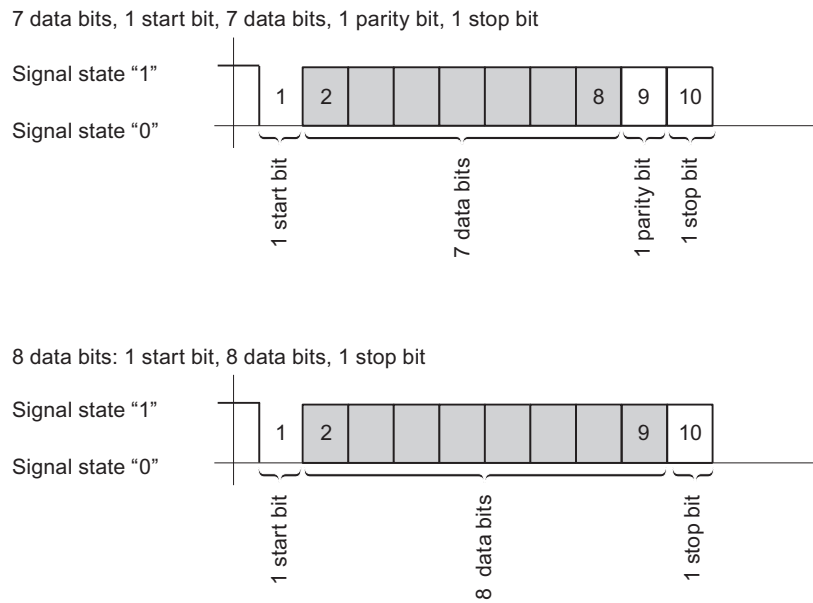


Figure 2-1 10-Bit Character Frame

### 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.

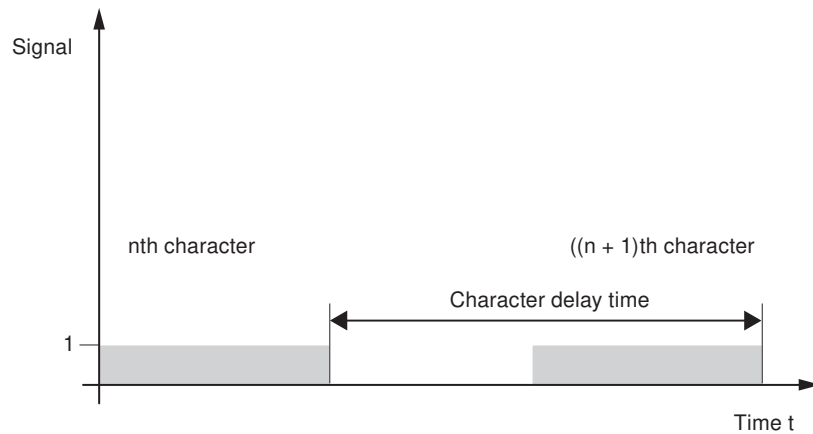


Figure 2-2 Character Delay Time

## 2.2 Transmission Procedure with a Point-to-Point Connection

### Introduction

When data are transmitted, all communication partners must adhere to a fixed set of rules for handling and implementing 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.

### ISO 7-Layer Reference Model for Data Transmission

All communication partners must adhere to a fixed set of rules for handling and implementing data traffic. Such rules are called protocols.

#### A protocol defines the following:

- Operating mode  
Half-duplex or full-duplex operation
- Initiative  
Which communication partners can initiate data transmission and under what conditions
- Control characters  
Which control characters are to be used for data transmission
- Character frame  
Which character frame is 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-link layer
  - Security procedure for the transmission
  - Access modes
3. Network layer
  - Network connections
  - Specifies the addresses for communication between two partners.
4. Transport layer
  - Error-recognition procedure
  - Debugging
  - Handshaking
5. Session layer
  - Establishing communication
  - Communication control
  - 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.

## 2.3 Transmission integrity

**Introduction**

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 440 can use the following protocols:

- ASCII driver
- 3964(R) Procedure

The figure below illustrates how these protocols of the CP 440 fit into the ISO reference model:

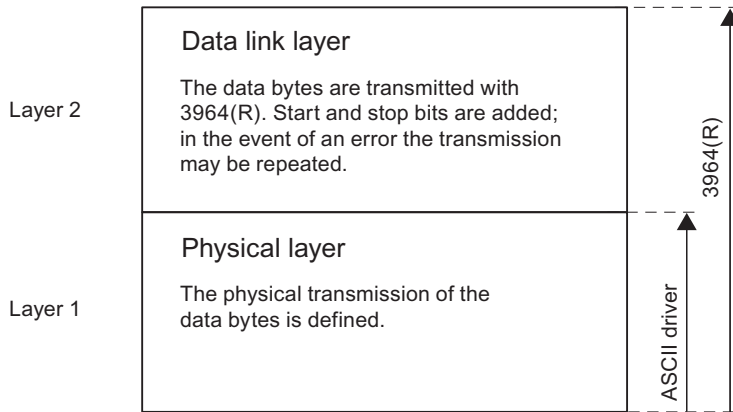


Figure 2-3 Position of the supplied protocols of the CP 440 in the reference model

### 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(R)

Enhanced Data Integrity with the 3964(R) Procedure:

- The Hamming distance with the 3964(R) is 3. This measures the integrity of data transmission.
- The 3964(R) procedure ensures high transmission integrity on the data line. This high integrity is achieved by means of a fixed message-frame set-up and clear-down 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 a 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 3964(R) 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 operation) cannot detect missing zeros (as a whole character) because a zero in the EXOR 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.

## 2.4 Data Transmission with the ASCII Driver

### 2.4.1 Data Transmission with the ASCII Driver

#### Introduction

The ASCII driver controls data transmission via a point-to-point connection between the CP 440 and a communication partner. This driver contains the physical layer (layer 1).

The structure of the message frames is left open through the S7 user passing on the complete send message frame to the CP 440. For the receive direction, the end criterion of a message must be configured. 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.

Both RS422 and RS485 operation are possible.

### RS422 Operation

In RS422 operation, the data is transmitted via four cables (four-wire mode). Two cables (differential signal) are available for the send direction and two for the receive direction. This means you can send and receive data at the same time (full-duplex operation).

### RS485 Operation

In RS485 operation, the data is transmitted via two cables (two-wire mode). The two cables (differential signal) are alternately available for the send direction and the receive direction. This means you can either send or receive data at the same time (half-duplex operation). After a send operation, the cable is immediately switched over to receive (the sender becomes high-impedance). The maximum switchover time is 0.1 ms.

## 2.4.2 Sending Data with the ASCII Driver

### Sending Data

When you send data, you specify the number of user data bytes to be transferred in the "LEN" parameter of the call of the SEND\_440 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 SEND\_440 FB at any time, but the ASCII driver does not begin its output until a period longer than the configured character delay time has elapsed since the last message frame was sent.

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 configured at the FB  
Data is sent up to the length configured at the FB. The last character must be the end-of-text character.
- Send up to the length configured at the FB and automatically append the end-of-text character or characters  
Data is sent up to the length configured at the FB. The end-of-text character(s) is/are 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 frame length", the number of data bytes transferred in the send direction is as specified for the "LEN" parameter of the SEND\_440 FB. 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 parameter assignment interface. The two parameter settings must be identical, in order to ensure correct data traffic. If an end code is not detected when data is sent, a pause equal to the length of the monitoring time is inserted between two message frames to allow the partner to synchronize (identify the beginning of the message frame).



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

---

**Note**

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

---

## Send Operation

The figure below illustrates a send operation.

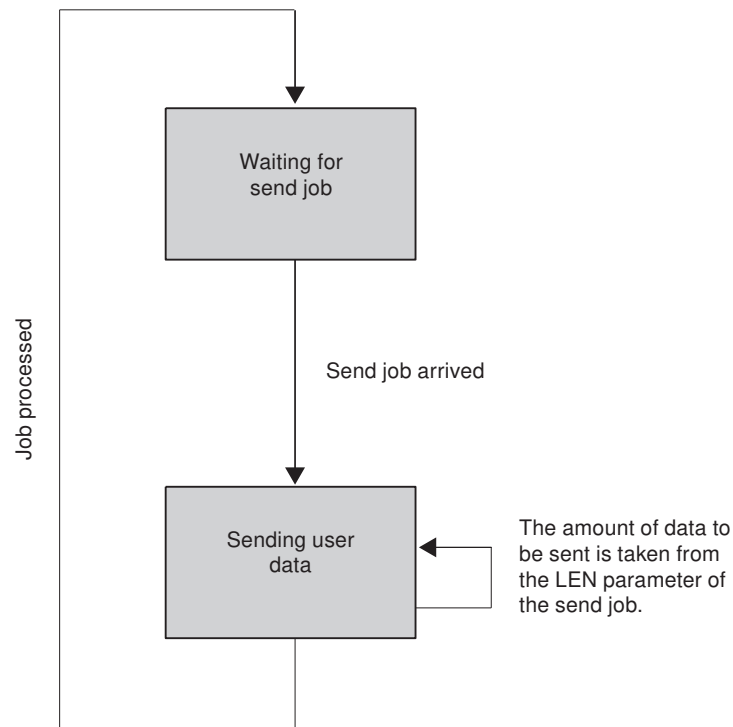


Figure 2-4 Sequence of a send operation

### 2.4.3 Receiving Data with the ASCII Driver

#### Selectable End Criteria

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

- Expiration of the 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 (expiration 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 choice of configured end criterion and 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 the flow control XON/XOFF is used.

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

#### End Criterion "Expiration 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 link takes a send pause within a message frame.

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

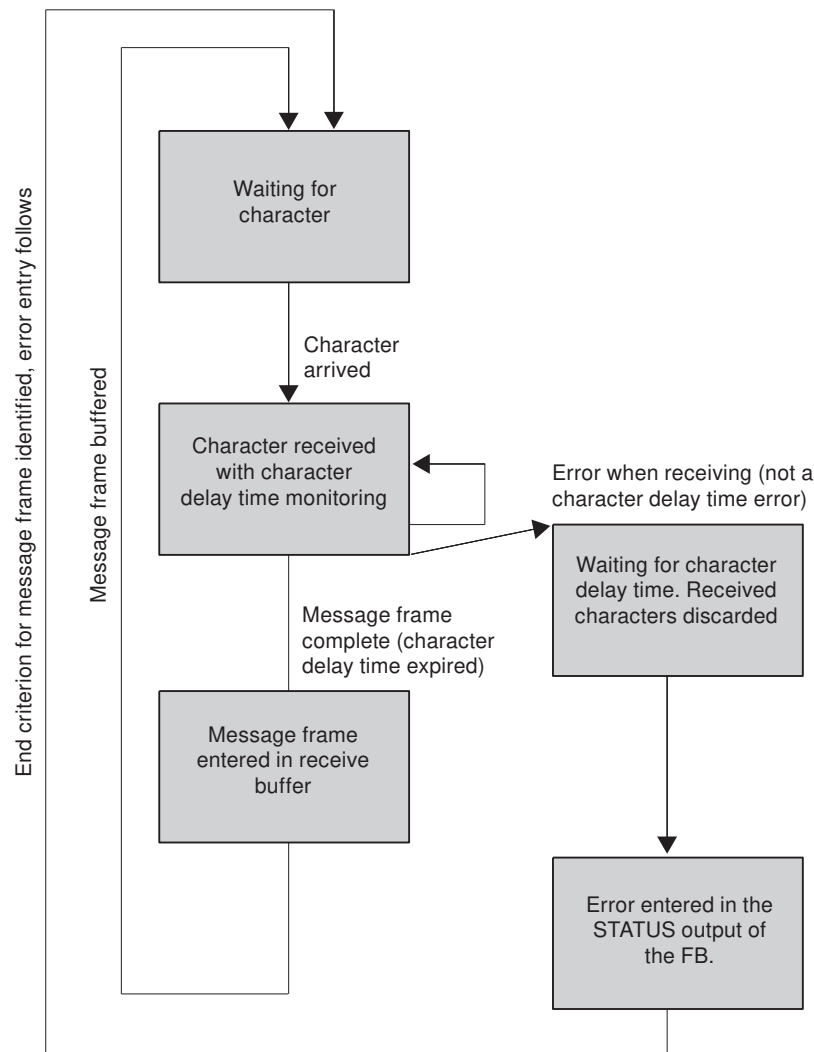


Figure 2-5 Sequence of Receive Operation with End Criterion "Expiration of Character Delay Time"

### End Criterion End-of-Text Character

When data is received, the end of the message frame is recognized when the configured end-of-text character(s) arrive. The following options are available:

- One end-of-text character
- Two end-of-text characters, 1st and 2nd end-of-text characters
- Two end-of-text characters, 1st or 2nd end-of-text character

The received data including the end-of-text character(s) is accepted from the CPU.

If the end-of-text character is missing in the received data, the character delay time elapses during reception and results in a termination of the frame. The character delay time is used in this instance as the monitoring time. 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 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
  - Is discarded if the monitoring time expires at the end of the message frame.
  - Is merged with the next message frame if a new message frame is received before the monitoring time expires.
- End-of-text character not included in message frame:  
The message frame
  - is discarded if the monitoring time expires at the end of the message frame.
  - Is merged with the next message frame if a new message frame is received before the monitoring time expires.

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

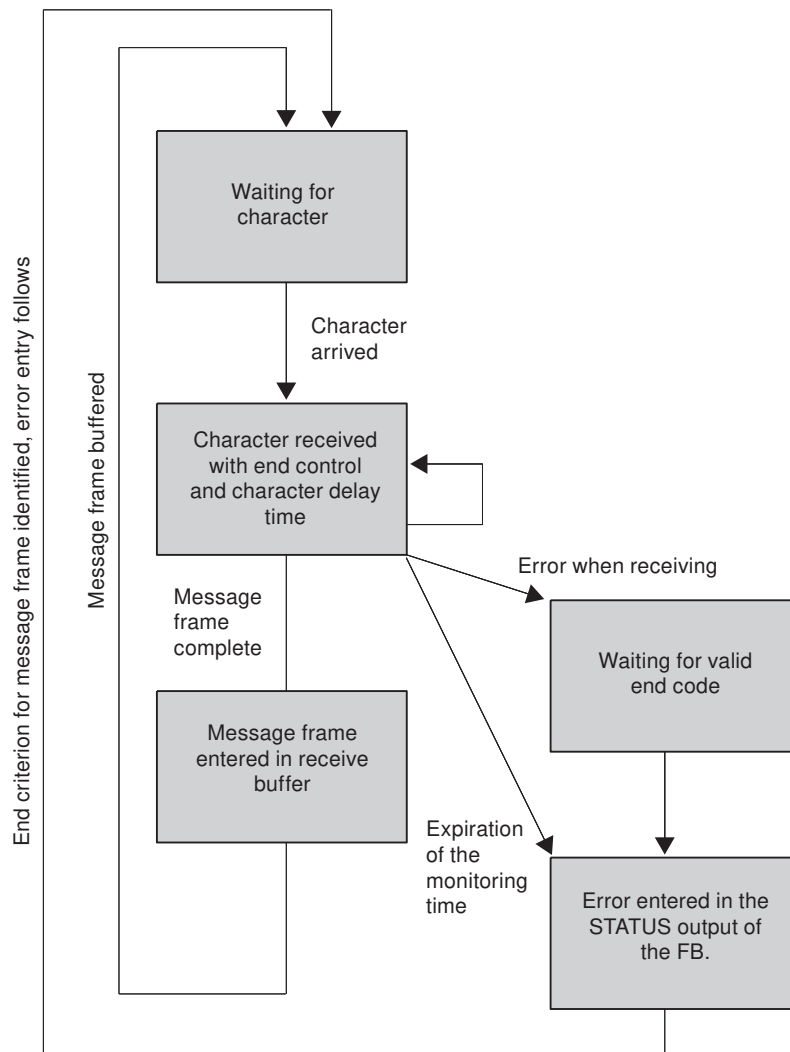


Figure 2-6 Sequence of Receive Operation with "End-of-Text Character" End Criterion

### End Criterion Fixed Message Frame Length

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

If the character delay time expires before the configured number of characters has been reached, the receive operation is terminated. The character delay time is used in this instance as the monitoring time. 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 configured fixed message frame length:

- Message frame length of received characters greater than configured fixed message frame length:  
All characters received after the parametered fixed message frame length is reached
  - is discarded if the monitoring time expires at the end of the message frame.
  - Is merged with the next message frame if a new message frame is received before the monitoring time expires.
- Message frame length of received characters less than parametered fixed message frame length:  
The message frame
  - Is discarded if the monitoring time expires at the end of the message frame.
  - is merged with the next message frame if a new message frame is received before the monitoring time expires.

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

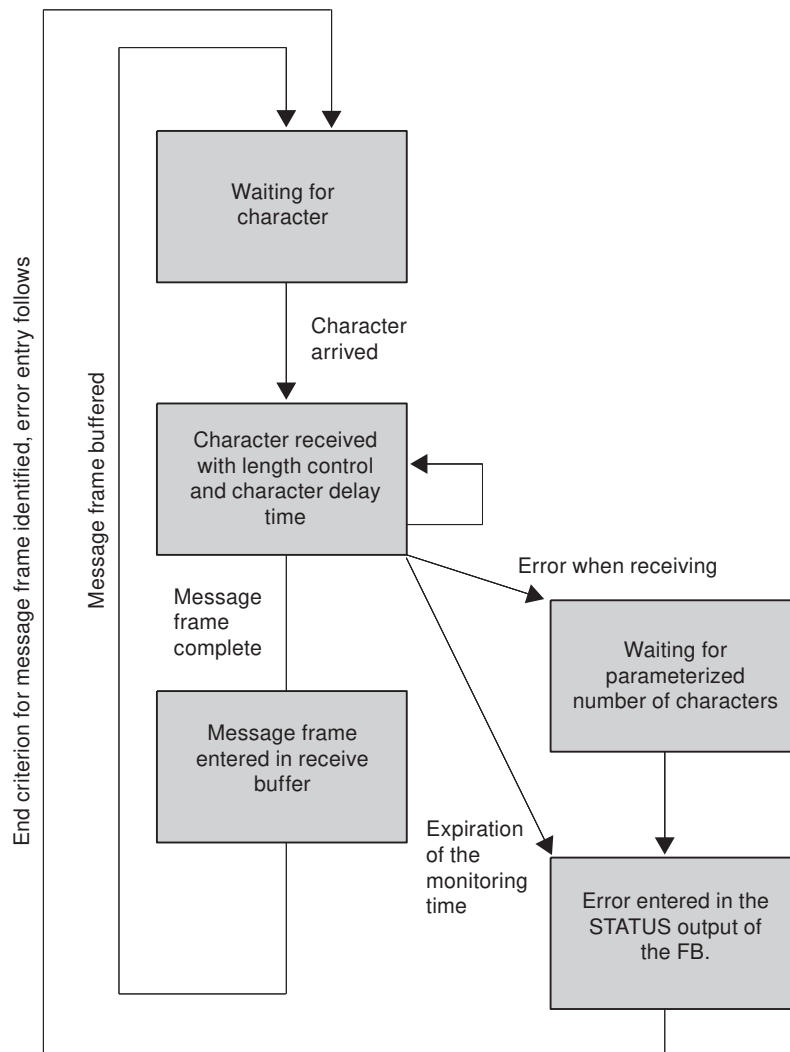


Figure 2-7 Sequence of Receive Operation with "Fixed Message Frame Length" End Criterion

## Receive Buffer on CP 440

The CP 440 receive buffer accommodates 2000 bytes. During the parameter assignment, you can specify whether overwriting of data in the receive buffer should be prevented. You can also specify the value range (1 to 10) for the number of buffered receive message frames or use the complete receive buffer.

You can delete the CP receive buffer at startup. You specify this setting either in the parameter assignment screen or by calling the RES\_RCV function block (see chapter "Communication via Function Blocks (Page 73)").

The receive buffer on the CP 440 is a ring buffer:

- If several message frames are entered in the receive buffer of the CP 440, the following applies: it is always the oldest one that is sent from the CP 440 to the CPU.
- If you only ever want to send the most recent message frame to the CPU, you must set the value "1" for the number of buffered message frames and deactivate the overwrite protection.

---

**Note**

If the constant reading out 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 CPU first receives old message frames from the CP 440 before it receives the most recent one. The old message frames are those on their way when transmission between the CP 440 and the CPU was interrupted, or which had already been received by the FB.

---

### 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. Software handshaking is supported with XON/XOFF in the CP 440.

Data flow control is implemented as follows on the CP 440:

- As soon as the CP 440 is switched by the configuration to the flow control operating mode, it sends the XON character.
- When the configured number of message frames is reached, or alternatively 50 characters before the receive buffer overflows (size of the receive buffer: 2000 bytes), the CP 440 sends the XOFF character. 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 440 sends the XON character.
- If the CP 440 receives the XOF character, the CP 440 interrupts transmission. If an XON character is not received before a configured time has elapsed, the transmission is aborted and an appropriate error message (0708H) is generated at the STATUS output of the function blocks.

### 2.4.4 Topologies Between the Communication Partners

#### Application options

The CP 440 can be used in different topologies in the RS422 and RS485 operating modes.

Distinctions are drawn between connections with:

- Two nodes (**point-to-point**) and
- Several nodes (**multipoint**)



In these cases the CP 440 can be used as:

- **Master** or
- **Slave.**

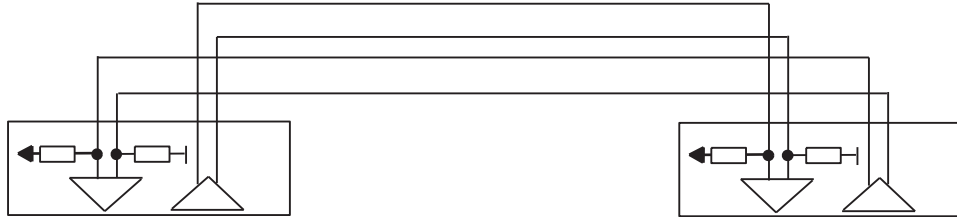


Figure 2-8 RS 422 Point-to-Point

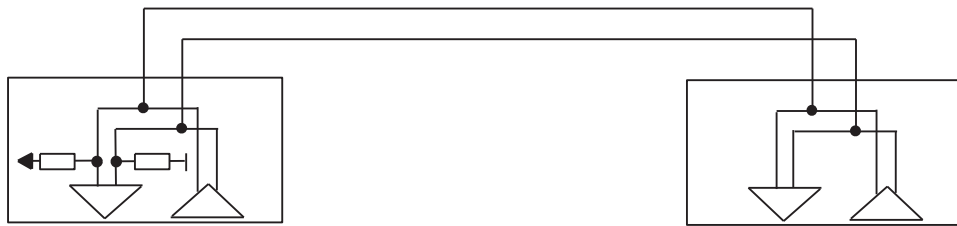


Figure 2-9 RS 485 Point-to-Point

In the case of a **master/slave topology**, there must be an appropriate message frame in the user program. Example: The master sends all the slaves a message frame with address information. All the slaves listen in and compare the address with their own. If the address is the same, the addressed slave sends its answer.

The senders of all slaves must be able to switch to low impedance.

## RS422 Operation

In the case of a master/slave topology in RS422 operation:

- The master's sender is interconnected with the receivers of all the slaves.
- The slaves' senders are interconnected with the master's receiver.
- Only the receiver of the master and the receiver of one slave have a default setting. All the other slaves function without default settings.

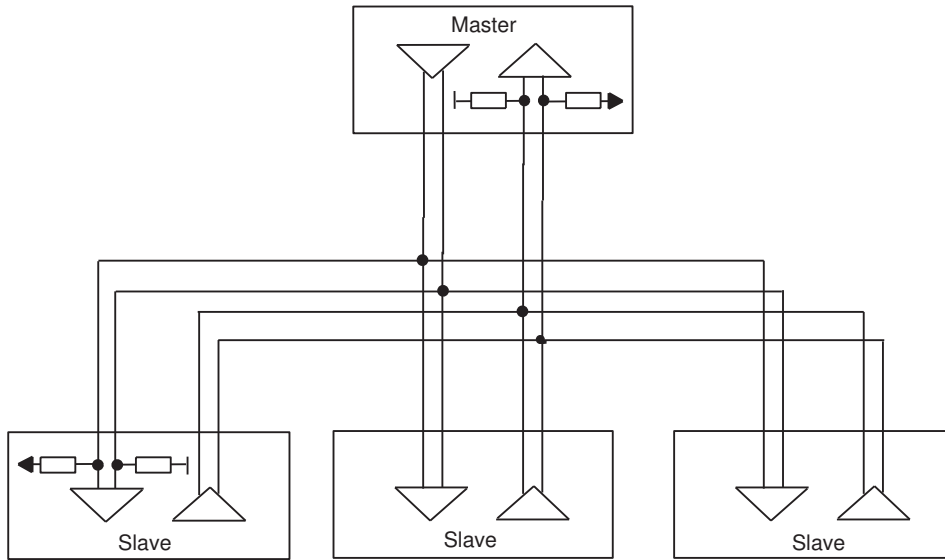


Figure 2-10 RS 422 Multipoint

### RS485 Operation

In the case of a topology in RS485 operation:

- The cable pair is interconnected for the send/receive line of all the nodes.
- Only the receiver of a node has a default setting. All the other modules function without default settings.

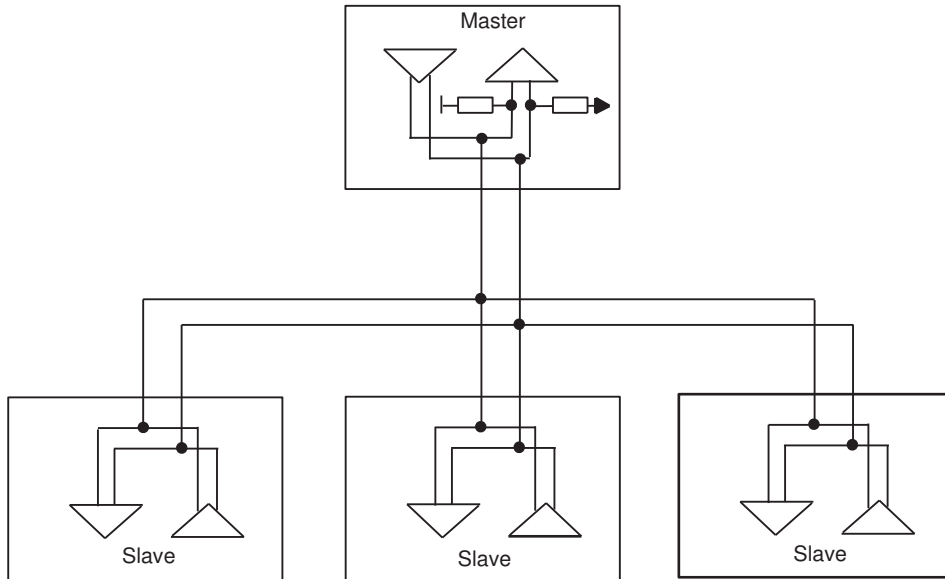


Figure 2-11 RS 485 Multipoint

The settings required for the different topologies can be made in the parameter assignment interface in the "Interface" dialog box.

---

**Note**

When you run the ASCII driver in RS422 multipoint or RS485 mode, you must take steps in the user program to ensure that only one node sends data at any one time. If two users send data simultaneously, the message frame is corrupted.

---

## 2.5 Data Transmission with the 3964(R) Procedure

### Introduction

The 3964(R) procedure controls point-to-point data exchange between the communication processor and a communication partner. As well as the physical layer (layer 1), the 3964(R) procedure also incorporates the data-link layer (layer 2).

### 3964(R) procedure startup

The figure below illustrates the startup of the 3964(R) procedure.

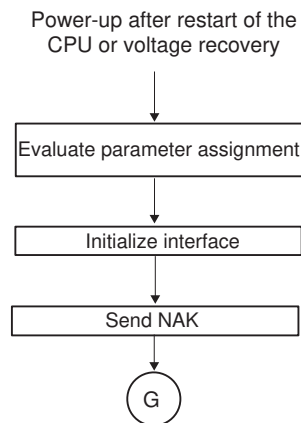


Figure 2-12 Flow Diagram of the Startup of the 3964(R) Procedure

### 2.5.1 The Control Characters of the 3964(R) Procedure

#### Introduction

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

## The Control Characters

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)
- **NAK**Negative Acknowledge

---

### 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.

## 2.5.2 Block Checksum

### Block Checksum

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

Message frame:

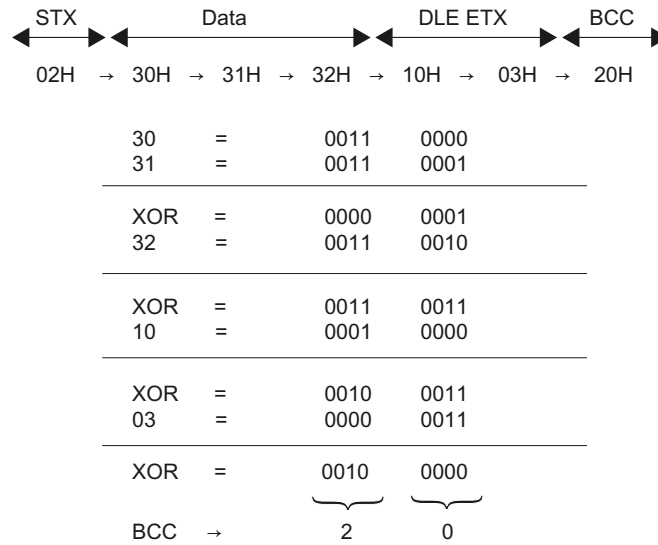


Figure 2-13 Block Checksum

The block checksum is the even longitudinal parity (EXOR operation on 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.

---

### 2.5.3 Sending Data with 3964(R)

#### Transmission Sequence

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

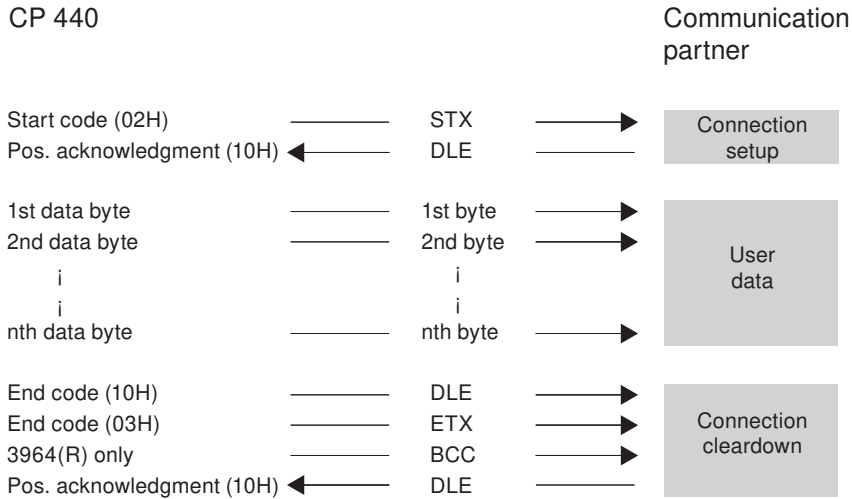


Figure 2-14 Data traffic when sending with the 3964(R) procedure

#### 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 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 connection attempts, the procedure aborts the connection setup and sends the NAK code to the communication partner. The CP 440 reports the error to the SEND\_440 function block (STATUS output parameter).

#### Sending Data

If a connection is successfully established, the user data contained in the output buffer of the CP 440 is sent to the communication partner with the selected 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.

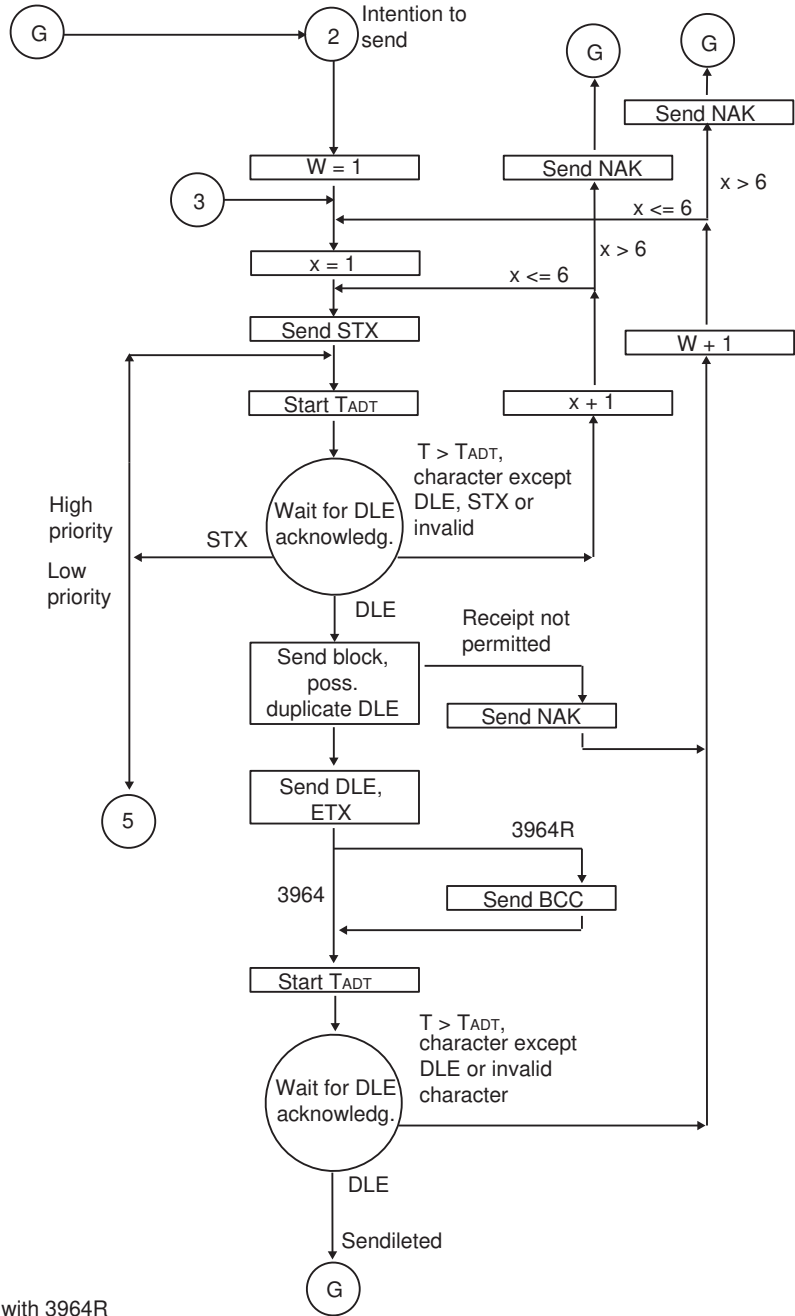
### **Releasing a Send Connection**

Once the contents of the buffer have been sent, the procedure adds the codes DLE, ETX and in the case of 3964(R) only the block checksum BCC as the end code, 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 440 reports the error to the SEND\_440 function block (STATUS output parameter).

**Sending with the 3964(R) procedure**

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



BCC only with 3964R  
 $x$  = setup attempt count  
 $T_{ADT} = 500$  ms (3964R  $T_{ADT}=2$ s)  
 $W$  = transmission attempt count  
 Immediate return to initial state at line break (BREAK)

Figure 2-15 Flow diagram of sending with the 3964(R) procedure

C: Counter for connection attempts

R: Counter for retries



D: Default state  
W: Waiting for character reception

## 2.5.4 Receiving Data with 3964(R)

### Process of Data Transmission when Receiving

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

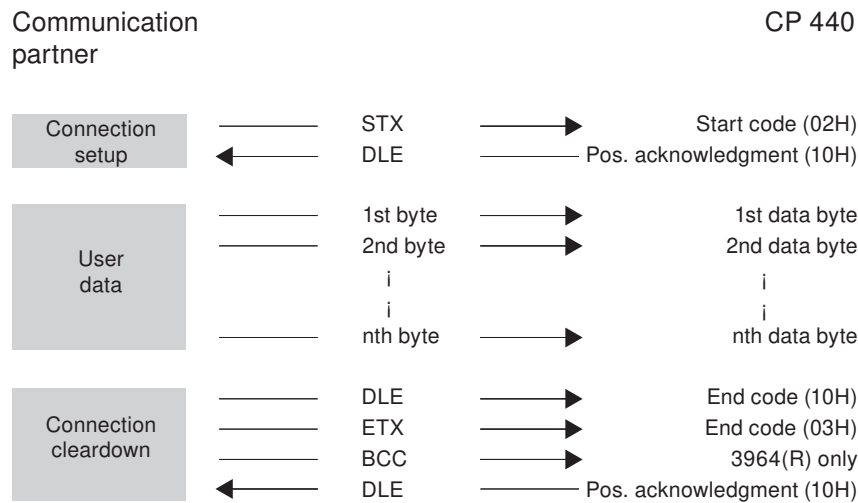


Figure 2-16 Data Traffic when Receiving with the 3964(R) Procedure

#### 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 elapsed, the CP 440 reports the error (error message at the STATUS output of the FB). and the procedure sends a NAK and returns to idle mode. Otherwise, the procedure sends a DLE and receives the data.

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 440 reports the error to the RECV\_440 function block (STATUS output parameter).

## Receiving Data

After a successful connection setup, the receive characters that 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, a NAK is sent to the communication partner. The system program reports the error to the RECV\_440 function block (STATUS output parameter). 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 shut down, then a 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 440 reports the first failed transmission and the final abortion of the receive operation to the RECV\_440 function block (STATUS output parameter).

## Releasing a Receive Connection

When the 3964 procedure detects a DLE ETX character string, it ends the receiving operation and confirms the successfully received block by sending a DLE signal to the communication partner. When errors are found in the received data, it outputs a NAK signal to the communication partner. A repetition is then expected.

If the 3964(R) procedure recognizes the string DLE ETX BCC, it stops receiving. It then 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 3964(R) procedure sends a DLE and returns to idle mode. If the BCC is faulty or a different receiving error occurs, a NAK is sent to the communication partner. A repetition is then expected.

### Receiving with the 3964(R) Procedure

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

#### Receiving with procedure 3964(R) (part 1)

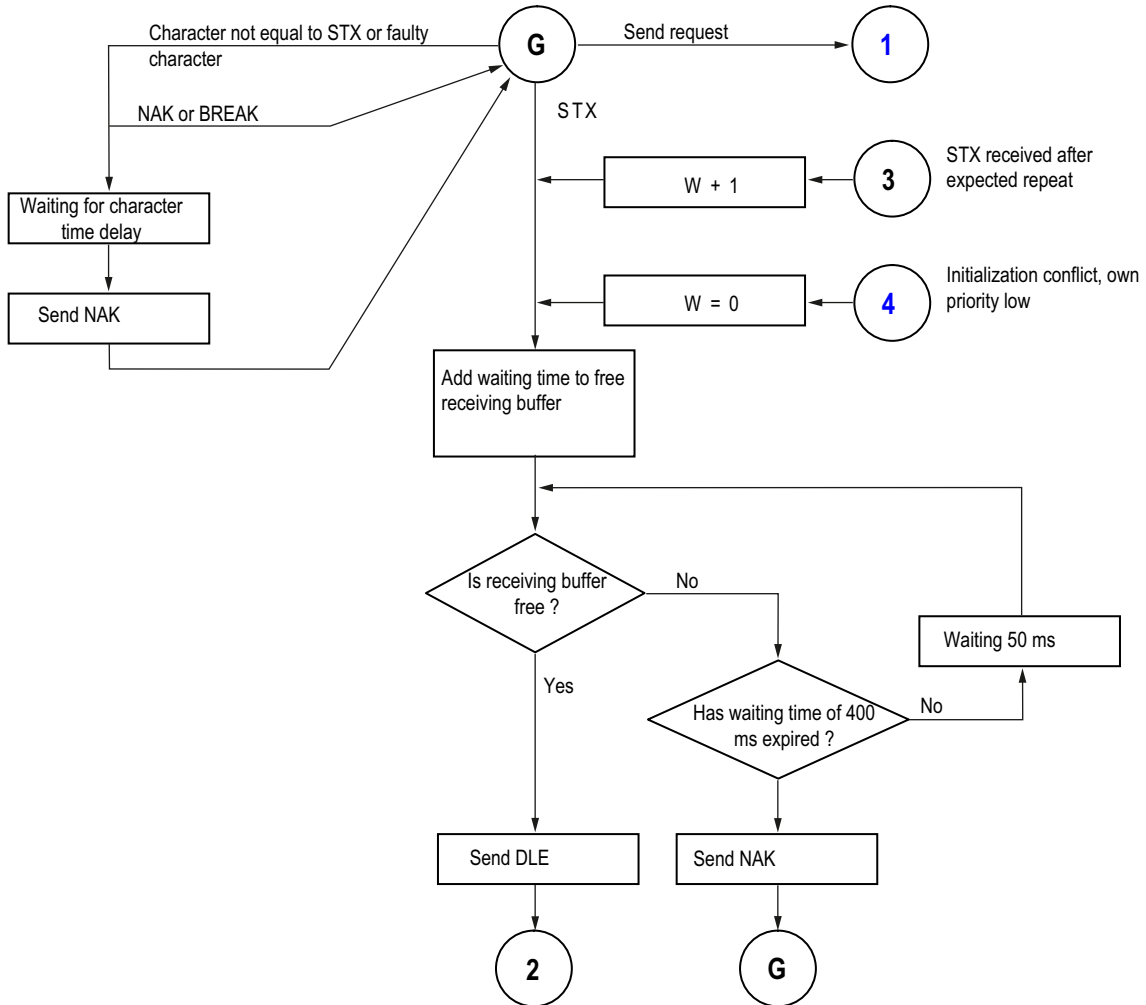


Figure 2-17 Flow diagram of receiving with the 3964(R) procedure (part 1)

R: Counter for retries

D: Default state

**Receiving with the 3964(R) Procedure (part 2)**

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

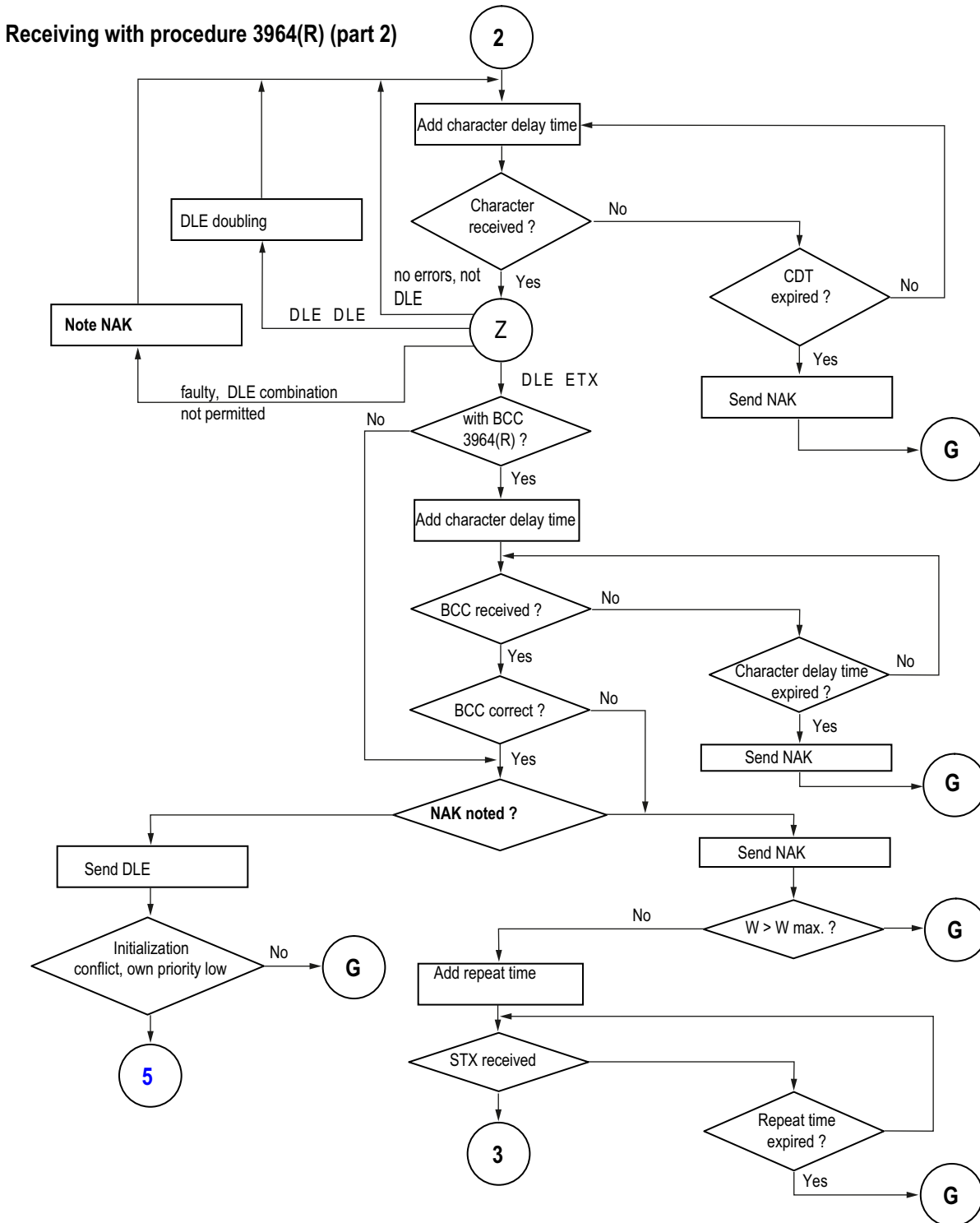


Figure 2-18 Flow diagram of receiving with the 3964(R) procedure (part 2)

R: Counter for retries  
D: Default state  
W: Waiting for character reception

### **Receive buffer on CP 440**

The CP 440 receive buffer accommodates 2000 bytes. During the parameter assignment, you can specify whether overwriting of data in the receive buffer should be prevented. You can also specify the value range (1 to 10) for the number of buffered receive message frames or use the complete receive buffer.

You can delete the CP receive buffer at startup. The setting can be made either by using the parameter assignment interface or by calling the RES\_RCV function block (see Chapter 6).

#### **The receive buffer on the CP 440 is a ring buffer:**

- If several message frames are entered in the receive buffer of the CP 440, the following applies: it is always the oldest one that is sent from the CP 440 to the CPU.
- If you only ever want to send the most recent message frame to the CPU, you must set the value "1" for the number of buffered message frames and deactivate the overwrite protection.

---

#### **Note**

If the constant reading out 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 CPU first receives old message frames from the CP 440 before it receives the most recent one. The old message frames are those on their way when transmission between the CP 440 and the CPU was interrupted, or which had already been received by the FB.

---

### 2.5.5 Handling Corrupt Data

#### Reaction to Corrupt Data

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

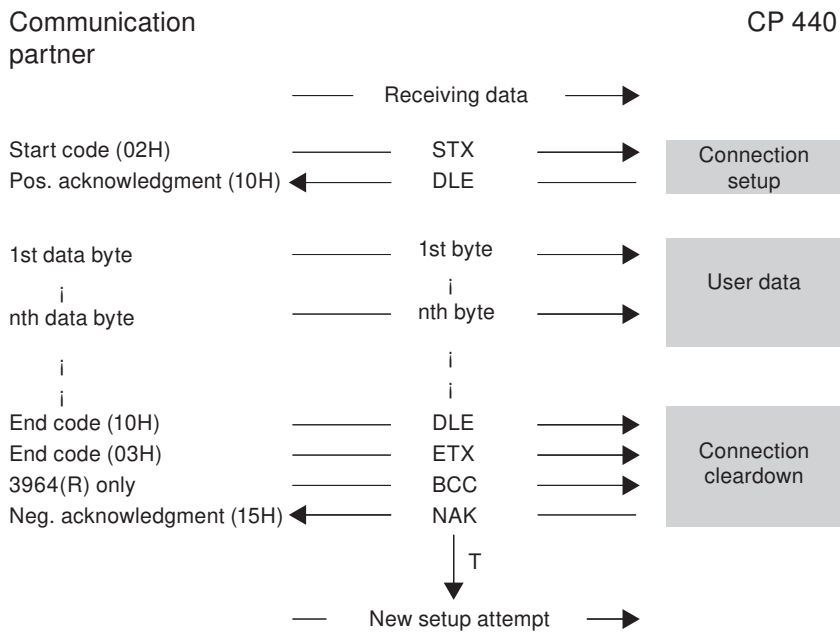


Figure 2-19 Data traffic when receiving corrupt data

When DLE, ETX, BCC is received, the CP 440 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 440 responds with DLE.

Otherwise, the CP 440 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 440 aborts the receive operation.

## Initialization Conflict

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

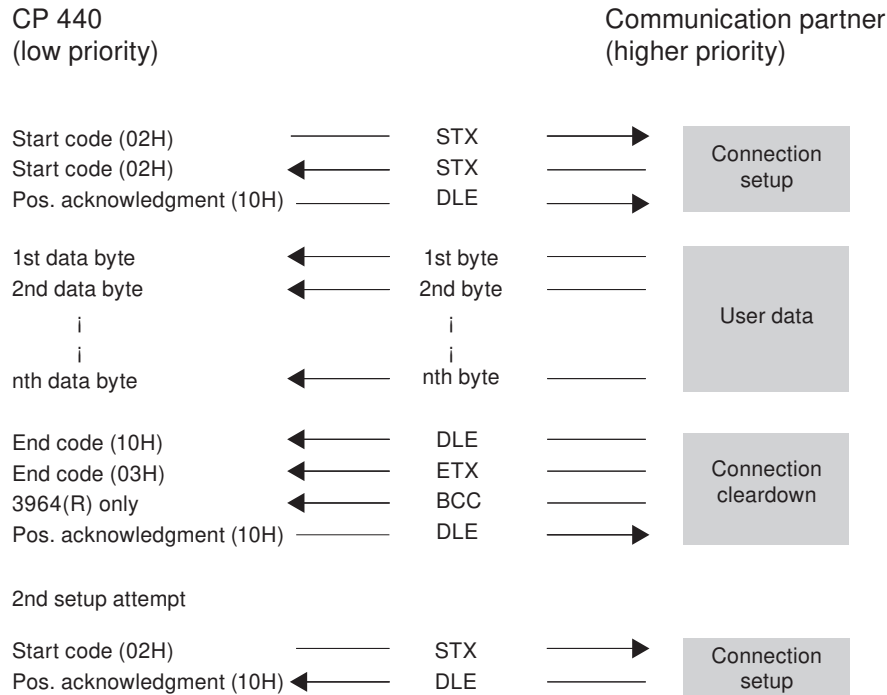


Figure 2-20 Data Traffic During an Initialization Conflict

If a device responds to the communication partner's send request (code STX) within the acknowledgment delay time 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 assign parameters with 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. These error messages are displayed in the STATUS output of the FB.

If the system program regularly reports an error number at the STATUS output of the FB for send and receive repetitions, this indicates occasional trouble affecting the data traffic. The large number of transmission attempts compensates for this, however. In this case you are advised to check the transmission link for possible sources of interference, because frequent repetitions

*2.5 Data Transmission with the 3964(R) Procedure*

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 BREAK occurs on the receive line (receive line interrupted), an error is displayed at the STATUS output of the FB. No repeat is started. The BREAK status 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.



# Commissioning the CP 440

## Step Sequence

Before starting up the CP 440 you will need to perform the following steps in the specified sequence.

1. Mounting the CP 440
2. Configuring the CP 440
3. Assigning parameters to the CP 440
4. Backing up configuration data
5. Creating a user program for the CP 440
6. Commissioning the physical interface

## Mounting the CP 440

Mounting the CP 440 involves inserting it into the mounting rack of your programmable controller.

You can find a detailed description in the chapter "Mounting the CP 440 (Page 53)" of this manual.

## Connecting the CP 440 to the Communication Partner

The connection to your communication partner is made using the cable.

## Configuring the CP 440

Configuration of the CP 440 includes its entry in the configuration table. Configure your CP 440 using the STEP 7 software.

You can find a detailed description in the chapter "Assigning parameters to the CP 440 (Page 55)" of this manual.

## Assigning parameters to the CP 440

Assigning parameters to the CP 440 involves creating the specific parameters of the protocols. Perform parameter assignment of the CP 440 from the programming interface **Point-to-Point Communication, Parameter Assignment**.

You can find a detailed description in the chapter "Parameters for the Communications Protocols (Page 56)" of this manual.

## Backing up Configuration Data

A backup of CP 440 parameter data includes the storage of parameters, their download to the CPU and transfer to the CP 440. Backup your CP configuration using the STEP 7 software.

You can find a detailed description in the chapter "Managing the Parameter Data (Page 68)" of this manual.

## Creating a User Program for the CP 440

CP 440 programming includes the implementation of the CP 440 in the STEP 7 user program of your CPU. Program your CP 440 using the language editors of the STEP 7 software.

A detailed description of programming with STEP 7 is available in the *Programming with STEP 7* manual.

Communication with the help of the CP 440 function blocks is described in the chapter "Communication via Function Blocks (Page 73)".

A comprehensive programming example is available in the chapter "Programming Example for Standard Function Blocks (Page 107)".

## Commissioning the physical interface

If there is no communication with the partner device after configuration has been completed, you should test the connection. To do this, proceed as follows:

1. What might be the possible cause:
  - Is the polarity of the send/receive lines reversed?
  - Are the default settings correct? Several defaults may have been set with different polarity. The default settings may already be integrated in the device on a permanent basis.
  - Missing or wrong terminating resistors?
  - High byte and low byte mixed in the security word (e.g. CRC)?
2. How to proceed:
  - First of all, use the manual to check the line connection:
    - Assignment (see Appendix "AUTOHOTSPOT")
    - Polarity (see Appendix "AUTOHOTSPOT")
    - Default settings (see chapter "Configuration data (Page 57)")
  - Carry out a setup test
3. Carry out the simplest setup possible:
  - Connect only 2 nodes to each another
  - If possible, use 2-wire cable (RS485)
  - Use a short cable
  - Terminating resistors are not required due to the short distance
  - Send first in one direction and then in the other

## 4. Check:

Always monitor the TXD send and RXD receive LEDs during the following operations

Example 1: The polarity of the line is definitely correct

- The default settings (all options) vary
- Check the security word (e.g. CRC)

Example 2: The default settings are definitely correct

- Reverse the connections (note: cross both pairs of lines in the RS422)
- Check the security word (e.g. CRC)

Example 3: Neither correct polarity nor correct default settings known

- Reverse the connections (note: cross both pairs of lines in the RS422)
- If not OK, change the default settings (all options) with an appropriate communication attempt
- If not OK, change the connections back, and change the default settings (all options)
- Check the security word (e.g. CRC)

When setting up again, don't forget to put back the terminating resistors you removed.

## 5. Additional tips:

- If possible, connect an interface tester (possibly the V.24 → RS 422/485 converter) to the cable.
- Check the signal level using a measuring device (measure level to GND - pin 8).
- Some devices do not signal reception if data is being received but the CRC security word is not correct.
- If necessary, replace the module to exclude the possibility of an electrical defect.



## Mounting the CP 440

### 4.1 CP 440 slots

#### Introduction

There are no specific slots reserved for communication modules in the rack of the S7-400 automation system.

#### Positioning the CP in the Rack

The communication processor 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.

#### Further Information

Additional information about the topic of racks is available in the installation manual *S7-400, Automation Systems, Installation*.

### 4.2 Mounting and Dismounting the CP 440

#### Introduction

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

#### Tool

You will need a 3.5 mm cylindrical screwdriver to mount or dismount the communication processor.

---

#### Note

The CP 440 can be hot-plugged and hot-pulled, in other words with voltage applied. This means that the CP 440 can be replaced while the programmable logic controller is in operation. To avoid the CPU going into STOP, OB 83 (insert/remove interrupt) and OB 122 (I/O access error) must be programmed. CP 440 is configured automatically when it is plugged in. The CP 440 then resumes operation.

---

### Installation Guidelines

The general installation guidelines for S7-400 must be followed (see the *S7-400 Automation System, Installation* manual).

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

#### 4.2.1 Installation steps

To mount the communication processor 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. Insert the CP 441 module and tilt it downward.
2. Hang the communication processor in the rack and swing it down.
3. Screw down the module at the top and bottom with a torque of 0.8 to 1.1 Nm.

#### 4.2.2 Removal steps

To remove the communication processor 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.

# Configuring and Parameter Assignment the CP 440

## 5.1 Assigning parameters to the CP 440

### Requirements

The programming interface **Point-to-Point Communication, Parameter Assignment** is installed in the STEP 7 software on your PG/PC.

Before you can enter the communication processor in the configuration table of the STEP 7 software, you must have created a project and a terminal with STEP 7.

### Parameter Assignment Options

You configure and assign parameters to the CP 440 using STEP 7 or the **CP 440: Point-to-Point Communication, Parameter Assignment** interface.

Table 5-1 Configuration Options for the CP 440

Product	Order number	Configurable using the parameter assignment tool	under STEP 7
CP 440	6ES7 440-1CS00-0YE0	As of version 5.1	As of version 5.3

### Configuring the CP 440

Once you have mounted the communication processor, you must inform the programmable controller that it is there. This process is known as "configuration".

In the following, "configuration" refers to the entry of the communication processor in the configuration table of the STEP 7 software. In the configuration table, enter the rack, the slot and the order number of the communication processor. STEP 7 then automatically assigns an address to the CP.

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

### Further Information

How to configure S7-400 modules is described in detail in the *Configuring Hardware and Communication Connections STEP 7* manual.

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

## 5.2 Installing the Programming Interface

### Installation

The **CP 440: Point-to-Point Communication, Parameter Assignment** interface is supplied together with the function blocks and the programming examples on a CD.

To install the engineering tool:

1. Insert the CD into the CD drive of your programming device/PC.
2. In **Microsoft Windows**, open the dialog for installing software by double-clicking the "Add and Remove Programs" icon in the "Control Panel".
3. In the dialog box, select the CD drive and the "Setup.exe" file and start installation.
4. Follow the step-by-step instructions of the Setup program.

## 5.3 Parameters for the Communications Protocols

### Introduction

Once you have entered the communication processor in the configuration table, you must supply parameters to it and its serial interface.

### Parameter assignment

The term "parameter assignment" is used in the following to describe the setting of protocol-specific parameters. This is done using the **CP 440: Point-to-Point Communication, Parameter Assignment** interface.

You start the programming interface by double-clicking the order number (CP 440) in the configuration table or by selecting the CP 440 and selecting the **Edit > Object Properties** menu command. The "Properties - CP 440" dialog box appears.

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 for setting the protocol-specific parameters.

### Further Information

The basic operation of the **Point-to-Point Communication, Parameter Assignment** programming interface is the same for all communication processors and is self-explanatory. For this reason, the parameter assignment interface is not described in detail here.



Also, the on-line help provides sufficient support for working with the parameter assignment interface.

---

#### Note

To create a parameter record for the CP, you must select and save the protocol settings at least once.

---

## 5.4 Configuration data

### 5.4.1 Introduction

#### Introduction

You can set the response of the CP 440 to CPU STOP by means of the basic parameters. By selecting different protocols, you can adjust your CP 440 communication processor to suit the properties of the communication partner.

The sections that follow describe the basic parameters of the CP 440 and the configuration data for the ASCII driver and for the 3964(R) procedure.

### 5.4.2 Basic Parameters of the CP 440

#### Basic parameters

Enter the basic parameters in the STEP 7 HW dialog box "Properties - CP 440 ". Open the dialog box by double-clicking the CP 440 in the configuration table of STEP 7.

The basic parameters are described in the table below.

Table 5-2 Basic Parameters of the CP 440

Parameters	Description	Value Range	Default Value
Reaction to CPU Stop	This parameter controls the storage of the received frames in the receive buffer. You can find more detailed information in the following tables.	<ul style="list-style-type: none"> <li>• Continue</li> <li>• STOP</li> </ul> The transmission process is terminated in both cases.	Continue

5.4 Configuration data

**Affect of the “Reaction to CPU Stop” Parameter on the Storage of the Received Message Frames**

The response depends on whether or not flow control is used.

Table 5-3 Control of Message Frame Storage Without Flow Control

Without Flow Control	Saved Frames	Frame Just Arriving	New Frames
Reaction to CPU Stop: Continue	Retained	Saved; discarded if the buffer is full	Saved until the buffer is full, then discarded
Reaction to CPU STOP: STOP	Retained	Discarded	Discarded

Table 5-4 Control of Frame Storage with Flow Control

With Flow Control	Saved Frames	Frame Just Arriving	New Frames
Reaction to CPU STOP: Continue	Retained	Saved; flow control is activated if the buffer is full.	Saved; flow control is activated if the buffer is full.
Reaction to CPU STOP: STOP	Retained	Further data cannot be received because flow control is activated.	Additional data cannot be received because flow control is activated.

**See also**

- Configuration Data of the 3964(R) Procedure (Page 63)
- Managing the Parameter Data (Page 68)

**5.4.3 Configuration Data of the ASCII Driver**

**Introduction**

Using the configuration data of the ASCII driver, you can adjust the CP440 to suit the communication partner.

**Configuration Data of the ASCII Driver**

With the **CP 440: Point-to-Point Communication, Parameter Assignment** interface, specify the parameters for the physical layer (layer 1) of the ASCII driver. Below you will find a detailed description of the parameters.

---

**Note**

The ASCII driver can be used in four-wire mode (RS 422) and two-wire mode (RS 485). At parameter assignment, you must specify the type of interface (RS 422 or RS 485).

---

## Protocol parameters

The table below describes the protocol parameters.

Table 5-5 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.	<ul style="list-style-type: none"> <li>After character delay time expires</li> <li>On receipt of a fixed number of characters</li> <li>On receipt of end-of-text character</li> </ul>	After character delay time expires																								
Character delay time	The character delay time defines the maximum permitted time between 2 consecutively received characters.	<ul style="list-style-type: none"> <li>1 to 65530 ms</li> </ul> <p>The shortest character delay time depends on the transmission rate</p>	4 ms																								
Monitoring time for missing end ID	The character delay time is used as the monitoring time for a missing end code. This applies to the following settings for the end code <ul style="list-style-type: none"> <li>On Receipt of Fixed Number of Characters</li> <li>On receipt of end-of-text character</li> </ul>	<table border="1"> <thead> <tr> <th>Baud</th> <th>CDT (ms)</th> </tr> </thead> <tbody> <tr><td>• 300</td><td>• 130</td></tr> <tr><td>• 600</td><td>• 65</td></tr> <tr><td>• 1200</td><td>• 32</td></tr> <tr><td>• 2400</td><td>• 16</td></tr> <tr><td>• 4800</td><td>• 8</td></tr> <tr><td>• 9600</td><td>• 4</td></tr> <tr><td>• 19200</td><td>• 2</td></tr> <tr><td>• 38400</td><td>• 1</td></tr> <tr><td>• 57600</td><td>• 1</td></tr> <tr><td>• 76800</td><td>• 1</td></tr> <tr><td>• 115200</td><td>• 1</td></tr> </tbody> </table>	Baud	CDT (ms)	• 300	• 130	• 600	• 65	• 1200	• 32	• 2400	• 16	• 4800	• 8	• 9600	• 4	• 19200	• 2	• 38400	• 1	• 57600	• 1	• 76800	• 1	• 115200	• 1	
Baud	CDT (ms)																										
• 300	• 130																										
• 600	• 65																										
• 1200	• 32																										
• 2400	• 16																										
• 4800	• 8																										
• 9600	• 4																										
• 19200	• 2																										
• 38400	• 1																										
• 57600	• 1																										
• 76800	• 1																										
• 115200	• 1																										
End-of-text character 1 <sup>(1)</sup>	First end code.	<ul style="list-style-type: none"> <li>with 7 data bits: 0 to 7FH (Hex) <sup>(2)</sup></li> <li>with 8 data bits: 0 to FFH (Hex) <sup>(2)</sup></li> </ul>	3 (03H = ETX)																								
End-of-text character 2 <sup>(1)</sup>	Second end code, if specified. You can configure the following for the end code: <ul style="list-style-type: none"> <li>1. end-of-text character AND 2nd end-of-text character</li> <li>1. end-of-text character OR 2nd end-of-text character</li> </ul>	<ul style="list-style-type: none"> <li>with 7 data bits: 0 to 7FH (Hex) <sup>(2)</sup></li> <li>with 8 data bits: 0 to FFH (hex) <sup>(2)</sup></li> </ul>	0																								
Message frame length when received <sup>(3)</sup>	When the end criterion is "fixed message frame length", the number of bytes making up a message frame is defined.	1 to 400 (bytes) (see chapter "Overview of the Function Blocks (Page 73)")	200																								

5.4 Configuration data

Parameter	Description	Value Range	Default Value
Transmission pause between the messages related to the monitoring time	A pause equal to the length of the monitoring time (for a missing end ID) is inserted between two message frames when sending so that the partner can synchronize (detect receipt of the message frame).	<ul style="list-style-type: none"> <li>• Yes</li> <li>• No</li> </ul>	Yes
<p><sup>(1)</sup> Can only be set if the end criterion is an end-of-text character.</p> <p><sup>(2)</sup> Depending on whether you set 7 or 8 data bits for the character frame.</p> <p><sup>(3)</sup> Can only be set if the end criterion is fixed message frame length.</p>			

**Baud Rate / Character Frame**

The table below contains descriptions of and specifies the value ranges of the relevant parameters.

Table 5-6 Baud Rate / Character Frame (ASCII Driver)

Parameter	Description	Value Range	Default Value
Baud rate	Speed of data transmission in bits per second (baud)	<ul style="list-style-type: none"> <li>• 300</li> <li>• 600</li> <li>• 1200</li> <li>• 2400</li> <li>• 4800</li> <li>• 9600</li> <li>• 19200</li> <li>• 38400</li> <li>• 57600</li> <li>• 76800</li> <li>• 115200</li> </ul>	9600
Start bit	During transmission, a start bit is prefixed to each character to be sent.	1 (fixed value)	
Data bits	Number of bits onto which a character is mapped.	<ul style="list-style-type: none"> <li>• 7</li> <li>• 8</li> </ul>	8
Stop bits	During transmission, stop bits are appended to every character to be sent, indicating the end of the character.	<ul style="list-style-type: none"> <li>• 1</li> <li>• 2</li> </ul>	1
Parity	<p>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. This improves data integrity.</p> <p>A parity of "none" means that no parity bit is sent.</p> <p>It is not possible to specify "none" if 7 data bits are set.</p>	<ul style="list-style-type: none"> <li>• none</li> <li>• odd</li> <li>• even</li> </ul>	even

## Data flow control

The table below contains descriptions of the parameters for data flow control. Data flow control is only possible when "Full Duplex (RS 422) Four-Wire Mode Point to Point" is set.

Table 5-7 Data flow control (ASCII driver)

Parameter	Description	Value Range	Default Value
Data flow control	Defines which data flow control procedure is used.	<ul style="list-style-type: none"> <li>None</li> <li>XON/XOFF</li> </ul>	none
XON character <sup>(1)</sup>	Code for XON character	<ul style="list-style-type: none"> <li>with 7 data bits: 0 to 7FH (Hex) <sup>(2)</sup></li> <li>with 8 data bits: 0 to FFH (Hex) <sup>(2)</sup></li> </ul>	11 (DC1)
XOFF character <sup>(1)</sup>	Code for XOFF character	<ul style="list-style-type: none"> <li>with 7 data bits: 0 to 7FH (Hex) <sup>(2)</sup></li> <li>with 8 data bits: 0 to FFH (Hex) <sup>(2)</sup></li> </ul>	13 (DC3)
Waiting for XON after XOFF (wait time for CTS=ON) <sup>(1)</sup>	Period of time for which the CP 440 should wait for the XON code or for CTS="ON" of the communication partner when sending.	<ul style="list-style-type: none"> <li>20 ms to 65530 ms in 10 ms increments</li> </ul>	20000 ms
<sup>(1)</sup> Only for data flow control with XON/XOFF.			
<sup>(2)</sup> Depending on whether you set 7 or 8 data bits for the character frame.			

## Receive buffer on CP

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

Table 5-8 Receive buffer on CP (ASCII Driver)

Parameter	Description	Value Range	Default Value
Delete CP receive buffer at startup	During power up or during transition of the CPU from STOP to RUN the CP receive buffer is deleted.	<ul style="list-style-type: none"> <li>Yes</li> <li>No</li> </ul>	No
Buffered receive message frames	You can specify the number of receive message frames to be buffered in the CP receive buffer or to use the whole buffer. If you use the whole buffer of 2000 bytes, the number of buffered receive message frames depends only on the length of the frames.  If you specify "1" here and deactivate the following parameter "prevent overwrite" <b>and</b> cyclically read the received data from the user program, a current message frame will always be sent to the CPU.	<ul style="list-style-type: none"> <li>1 to 10</li> <li>Use whole buffer</li> </ul>	Use whole buffer
Prevent overwrite	You can use this parameter to prevent data in the receive buffer being overwritten when the buffer is full.	<ul style="list-style-type: none"> <li>Yes</li> <li>No</li> </ul>	Yes

### Operating mode/initial state of the receive line

The table below contains descriptions of the operating mode/initial state of the receive line for the X27 (RS 422/485) interface.

Table 5-9 X27 (RS 422/485) Interface (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).	<ul style="list-style-type: none"> <li>Full Duplex (RS 422) Four-Wire Mode Point-to-Point Operating mode for point-to-point communication in four-wire mode</li> <li>Full Duplex (RS 422) Four-Wire Mode Multipoint Master Operating mode for multipoint communication in four-wire mode if the CP is a master.</li> <li>Full Duplex (RS 422) Four-Wire Mode Multipoint Slave Operating mode for multipoint communication in four-wire mode if the CP is a slave.</li> <li>Half Duplex (RS 485) Two-Wire Mode Operating mode for point-to-point or multipoint communication in two-wire mode. The CP can be a master or slave.</li> </ul>	<ul style="list-style-type: none"> <li>Full Duplex (RS 422) Four-Wire Mode Point to Point</li> </ul>
Initial state of receive line	<p><b>none:</b> This setting only makes sense with bus-capable special drivers.</p> <p><b>R(A) 5V / R(B) 0V</b> Break evaluation is possible with this initial state. (Cannot be set with "Full Duplex (RS422) Four-Wire Mode Multipoint Master" and "Half Duplex (RS485) Two-Wire Mode")</p> <p><b>R(A) 0V / R(B) 5V:</b> This initial state corresponds to the idle state (no sender active). Break detection is not possible with this initial state.</p>	<ul style="list-style-type: none"> <li>None</li> <li>R(A) 5V / R(B) 0V (break detection)</li> <li>R(A) 0V / R(B) 5V</li> </ul>	<ul style="list-style-type: none"> <li>Depends on the operating mode set</li> </ul>

### Initial state of receive line

The following figure illustrates the wiring of the receiver at the X27 (RS 422/485) interface:

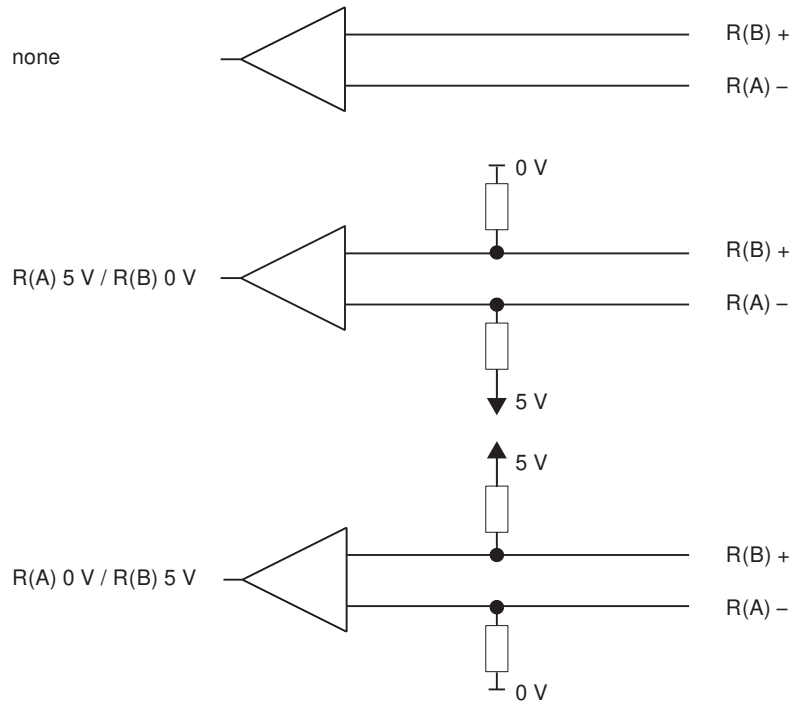


Figure 5-1 Wiring of the Recipient at the X27 (RS 422/485) Interface (ASCII Driver)

### See also

Data Transmission with the ASCII Driver (Page 23)

## 5.4.4 Configuration Data of the 3964(R) Procedure

### Introduction

Using the parameter assignment data of the 3964(R) procedure, you can adjust the CP 440 to suit the properties of its communication partner.

### Configuration Data of the 3964(R) procedure

With the **CP 440: Point-to-Point Communication, Parameter Assignment** interface, specify the parameters for the physical layer (layer 1) and the data-link layer (layer 2) of the 3964(R) procedure. Below you will find a detailed description of the parameters.

---

**Note**

The 3964(R) procedure can only be used in four-wire mode.

---

### Protocol

The following table describes the protocol.

Table 5-10 3964(R) Protocol

Parameter	Description	Default Value
3964 with default values and no block check	<ul style="list-style-type: none"> <li>The protocol parameters are set to default values.</li> <li>If the CP 440 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.</li> </ul>	<b>3964(R) with default values and block check:</b> CDT = 220 ms ADT = 2000 ms Connection attempts = 6 Transmission attempts = 6
3964R with default values and block check	<ul style="list-style-type: none"> <li>The protocol parameters are set to default values.</li> <li>If the CP 440 recognizes the string DLE ETX BCC, it stops receiving. The CP 440 compares the block check character (BCC) received with the length parity calculated internally. If the BCC is correct and no other receive errors have occurred, the CP 440 sends the code DLE to the communication partner (the NAK code is sent if an error occurs).</li> </ul>	
3964 assignable without block check	<ul style="list-style-type: none"> <li>The protocol parameters are programmable.</li> <li>If the CP 440 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.</li> </ul>	
3964R assignable with block check	<ul style="list-style-type: none"> <li>The protocol parameters are programmable.</li> <li>If the CP 440 recognizes the string DLE ETX BCC, it stops receiving. The CP 440 compares the block check character (BCC) received with the length parity calculated internally. If the BCC is correct and no other receive errors have occurred, the CP 440 sends the code DLE to the communication partner (the NAK code is sent if an error occurs).</li> </ul>	



## Protocol parameters

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

Table 5-11 Protocol Parameters (3964(R) Procedure)

Parameter	Description	Value Range	Default Value
Character delay time	The character delay time defines the permissible maximum interval between two incoming characters in a message frame.	20 ms to 65530 ms in 10 ms increments The shortest character delay time depends on the baud rate:	220 ms
		<ul style="list-style-type: none"> <li>• 300 bps</li> <li>• 600 bps</li> <li>• 1200 bps</li> <li>• 2400 to 15200 bps</li> </ul>	
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 65530 ms in 10 ms increments The shortest acknowledgment delay time (ADT) depends on the baud rate:	2000 ms (550 ms with 3964 without block check)
		<ul style="list-style-type: none"> <li>• 300 bps</li> <li>• 600 bps</li> <li>• 1200 bps</li> <li>• 2400 to 15200 bps</li> </ul>	
Connection attempts	This parameter defines the maximum number of attempts the CP 440 is allowed in order to establish a connection.	1 to 255	6
Transmission attempts	This parameter defines the maximum number of attempts to transfer a message frame (including the first one) in the event of an error.	1 to 255	6

5.4 Configuration data

**Baud Rate / Character Frame**

The following table describes the transmission rate/character frame.

Table 5-12 Baud Rate / Character Frame (3964(R) Procedure)

Parameter	Description	Value Range	Default Value
Baud rate	Speed of data transmission in bits per second (baud)	<ul style="list-style-type: none"> <li>• 300</li> <li>• 600</li> <li>• 1200</li> <li>• 2400</li> <li>• 4800</li> <li>• 9600</li> <li>• 19200</li> <li>• 38400</li> <li>• 57600</li> <li>• 76800</li> <li>• 115200</li> </ul>	9600
Start bit	During transmission, a start bit is prefixed to each character to be sent.	1 (fixed value)	
Data bits	Number of bits onto which a character is mapped.	<ul style="list-style-type: none"> <li>• 7</li> <li>• 8</li> </ul>	8
Stop bits	During transmission, stop bits are appended to every character to be sent, indicating the end of the character.	<ul style="list-style-type: none"> <li>• 1</li> <li>• 2</li> </ul>	1
Parity	<p>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. This improves data integrity.</p> <p>A parity of "none" means that no parity bit is sent.</p> <p>It is not possible to specify "none" if 7 data bits are set.</p>	<ul style="list-style-type: none"> <li>• none</li> <li>• odd</li> <li>• even</li> </ul>	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 configure both communication partners with different priorities, i.e. one partner is assigned high priority, the other low.	<ul style="list-style-type: none"> <li>• low</li> <li>• high</li> </ul>	high

## Receive buffer on CP

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

Table 5-13 Receive buffer on CP (3964(R) procedure)

Parameter	Description	Value Range	Default Value
Delete CP receive buffer at startup	During power up or during transition of the CPU from STOP to RUN the CP receive buffer is deleted.	<ul style="list-style-type: none"> <li>• Yes</li> <li>• No</li> </ul>	<ul style="list-style-type: none"> <li>• No</li> </ul>
Buffered receive message frames	<p>You can specify the number of receive message frames to be buffered in the CP receive buffer or to use the whole buffer. If you use the whole buffer of 2000 bytes, the number of buffered receive message frames depends only on the length of the frames.</p> <p>If you specify "1" here and deactivate the following parameter "prevent overwrite" <b>and</b> cyclically read the received data from the user program, a current message frame will always be sent to the CPU.</p>	<ul style="list-style-type: none"> <li>• 1 to 10</li> <li>• Use whole buffer</li> </ul>	<ul style="list-style-type: none"> <li>• Use whole buffer</li> </ul>
Prevent overwrite	You can use this parameter to prevent data in the receive buffer being overwritten when the buffer is full.	<ul style="list-style-type: none"> <li>• Yes</li> <li>• No</li> </ul>	<ul style="list-style-type: none"> <li>• Yes</li> </ul>

## Initial state of receive line

The table below contains a description of the initial state of the receive line for the X27 (RS 422) interface. RS 485 operation is not possible in conjunction with the 3964(R) procedure.

Table 5-14 X27 (RS 422) interface (3964(R) procedure)

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

### Wiring of the recipient at the X27 (RS 422) interface

The figure below shows the wiring of the recipient at the X27 (RS 422) interface:

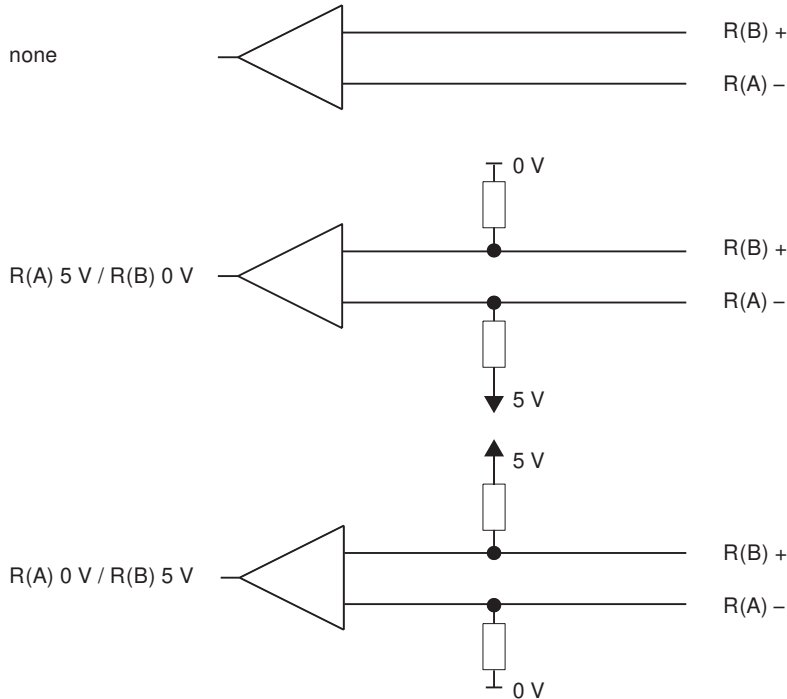


Figure 5-2 Wiring of the recipient at the X27 (RS 422) interface (3964(R) procedure)

## 5.5 Managing the Parameter Data

### Introduction

The configuration and configuration data of the CP 440 is saved in the current project (on the hard disk of the programming device/PC).

### Data management

When you exit the configuration table by selecting the **Station > Save** or **Station > Save As** menu command, the configuration and configuration data (including the module parameters) is automatically saved in the project/user file you have created.

### Loading the configuration and parameters

You can download the configuration and configuration data to the CPU from the programming device (by selecting **PLC > Download**). The CPU accepts the parameters immediately after the download.

The module parameters are automatically transferred to the CP 440 in the following cases:

- When they are downloaded to the CPU and the CP 440 can be reached via the S7-400 backplane bus
- After power up during startup of the CPU, as soon as the CP 440 can be reached via the S7-400 backplane bus

Default settings apply if parameters are not changed.

## Further Information

You can find out how to do the following in the *Configuring Hardware and Communication Connections STEP 7* manual:

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

## 5.6 Firmware updates

### 5.6.1 Subsequent Loading of Firmware Updates

#### Introduction

The communication processor is shipped with basic firmware preinstalled. You can enhance functionality and eliminate errors by downloading firmware updates to system memory of communication processor.

Subsequent loading of firmware updates with the **CP 440: Point-to-Point Communication, Parameter Assignment** interface.

#### Basic Firmware

The CP 440 is supplied with a basic firmware.

#### Requirements

The prerequisites for loading firmware updates are:

- 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 programming interface.
- The instructions accompanying the firmware update always detail the destination directories for the files.

The "..\CP440.nnn" path always identifies the firmware version.

## Load firmware

You upload the firmware update with the aid of the CP 440: **Point-to-Point Communication, Parameter Assignment** interface.

Proceed as follows:

1. Switch the CPU to STOP mode.
2. Start the parameter assignment interface:  
In SIMATIC Manager: **File > Open > Project > Hardware Config > double-click on CP 440 > select the "Parameter" button.**
3. Select the **Options > Firmware Update** menu command.  
**Result:**  
If a connection can be established to the CP 440, the current module firmware status is displayed.  
If there is no firmware loaded on the CP 440, 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 the "Find File ..." button to select the firmware to be loaded (header.upd).  
**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 440. You are prompted for confirmation. The upload procedure is canceled immediately if you click on the "Cancel" button.  
**Result:**  
The new firmware is loaded into the operating system memory of the CP 440. "Done" shows progress in bar-graph form and as a percentage. The module is immediately ready for use as soon as the firmware update is completed.

---

### Note

Before the basic firmware is deleted from the module, the CP 440 checks the order number of the firmware to be downloaded in order to ensure that the firmware is approved for the CP 440.

---

## Power Failure During Firmware Uploading

If there is a failure when the firmware is being uploaded, the process must be repeated. The module then has the state "CP 440 without module firmware".

## See also

Parameters for the Communications Protocols (Page 56)

## 5.6.2 Viewing the Firmware Version

### Viewing the Hardware and Firmware Version

You can view the current hardware and firmware version of the communication processor in **STEP 7** in the "Module Status" dialog. You can open this dialog by:

In SIMATIC Manager: **File > Open > Project > HW Config > Station > Online > double-click on the module of the communication processor.**

### LEDs

LEDs during the download of a firmware update:

Table 5-15 LEDs for firmware update

Status	INTF/EXTF	FAULT	TXD	RXD	Remark	To correct or avoid errors
Firmware update in progress	on	on	on	on	-	-
Firmware update completed	on	off	off	off	-	-
CP 440 without module firmware	flashes (2 Hz)	on	off	off	Module firmware deleted, firmware update was canceled, firmware update still possible	Reloading the firmware
Hardware fault during firmware update	flashes (2 Hz)	off	flashes (2 Hz)	flashes (2 Hz)	Read/write operation failed	Switch power supply to module off and then on again and reload the firmware. Check whether the module is defective.





# Communication via Function Blocks

## Introduction

Communication between the CPU, the CP 440 and a communication partner takes place via the function blocks and the protocols of the CP 440.

## Communication between CPU and CP 440

The function blocks form the software interface between the CPU and the CP 440. They must be called in cycles from the user program.

## Restriction in Multicomputing Operation

In multicomputing operation only one CPU can access the CP because communication between the CPU and CP is not multicomputing-compatible.

## Communication between CP 440 and a Communication Partner

The transmission protocol conversion takes place on the CP 440. The protocol is used to adapt the interface of the CP 440 to the interface of the communication partner.

This enables you to link an S7 programmable logic controller with any communication partner that can handle the ASCII driver and the 3964(R) procedure.

## Calling the Function Blocks in Interrupt OBs

It is not permissible to simultaneously call a function block in OB1 and in the interrupt OB (time OBs). The reason for this is that the function block cannot be called again if an interrupt OB triggers an interruption. If you need to call the function block in OB1 and in the interrupt OB, you must disable the interrupts (SFC 41) in the lower priority OB before the FB call and enable them again after the FBs have executed.

## 6.1 Overview of the Function Blocks

### Introduction

The S7-400 programmable logic controller provides you with a number of function blocks that initiate and control communication between the CPU and the CP 440 communication processor in the user program.

## Function Blocks

The table below contains the function blocks of the CP 440 and describes their purpose.

Table 6-1 Function Blocks of the CP 440

FB/FC	Meaning	Protocol
FB 9 "RECV_440"	The RECV_440 function block allows you to receive data from a communication partner and store it in a data block.	ASCII driver, 3964(R) procedure
FB 10 "SEND_440"	The SEND_440 function block allows you to send all or part of a data block to a communication partner.	ASCII driver, 3964(R) procedure
FB 11 "RES_RECV"	The RES_RECV function block enables you to reset the receive buffer of the CP 440.	ASCII driver, 3964(R) procedure

### Note

**The maximum CP440 frame length was extended from 200 to 400 bytes.**

To use this feature, you have to first install version V5.1.7 or higher from the PtP - Param. During installation, the two function blocks

FB9 V1.1 (RECV\_440)

and

FB10 V1.1 (SEND\_440)

are available in the CP PtP SIMATIC Standard Library in the subfolder CP440/Blocks. Only when using these blocks you can transfer more than 200 bytes (i.e. up to 400 bytes) with your CP440.

### Application in Existing User Projects:

If you want to extend the existing CP440 applications for the operation of up to 400 bytes frame length, you have to **first delete** the concerned CP440 in HW Config and then **reconfigure it with a PtP - Param V5.1.7 or higher**. Furthermore, you have to replace the two function blocks **FB9** and **FB10** (see above) with each new **version V1.1** in your user program.

## Scope of Supply and Installation

The function blocks of the CP 440 are on CD along with the parameter assignment interface, the programming examples and the manual.

The function blocks are installed together with the parameter assignment interface. After installation, the function blocks are stored in the CP 440 library.

You can open the library in STEP 7 SIMATIC Manager by selecting **File > Open > Library** under "CP PTP\CP 440\Blocks".

To work with the function blocks, you only need to copy the required function block to your project.

## See also

Parameters for the Communications Protocols (Page 56)

## 6.2 Notes on Program Structure

### Notes

In order to use the speed of the CP 440 to optimum effect, you should heed the following when you create your user program:

#### For a short cycle:

- Only one run of the SEND\_440 function block with REQ=0 and one run with REQ=1 is necessary for data transmission (creating the positive edge at the REQ input). The calls can be made directly one after the other. This makes it possible to carry out one data transmission per cycle.
- Program the call of the SEND\_440 FB with REQ=1 (this activates the FB) at the end of the program. This ensures optimum use of the time between two cycles, which can be relatively long in a short cycle.

#### For a long cycle:

- Call the SEND\_440 and RECV\_440 FBs several times throughout the program.
- Call the SEND\_440 and RECV\_440 FBs in the time OB (disable the interrupts in the lower priority OBs).

Note when using interrupts (time OBs) that calling the corresponding OB requires quite a lot of time (you can find exact times in your CPU manual). You should therefore check in each case whether the FBs could be processed more effectively by modifying your program.

## 6.3 Using the Function Blocks

### Introduction

The following sections describe what you must take into account when supplying the function blocks with parameters.

### STATUS Indication at the FB

Please note the following with reference to the STATUS display on the function blocks:

---

#### Note

The parameters DONE, NDR, ERROR and STATUS are valid for only one module run. To display the STATUS, you should therefore copy it to a free data area.

---

DONE = '1' means that the request was completed without error.

**This means:**

- With ASCII driver: The job was sent to the communication partner. This does not necessarily mean that the data was received by the communication partner.
- With 3964(R) procedure: The job was sent to the communication partner and acknowledged affirmatively by the communication partner. This does not necessarily mean that the data was forwarded to the partner CPU.

**Jobs which can be processed simultaneously**

Only one SEND\_440 FB, one RECV\_440 FB and one RES\_RECV FB can be programmed in the user program for each CP 440 used.

In addition, you can only use the following data blocks because the states required for the internal execution of the FB are stored in the instance data block:

- 1 instance data block for the SEND\_440 FB
- 1 instance data block for the RECV\_440 FB
- 1 instance data block for the RES\_RECV FB

**Data Consistency**

Please note the following to ensure consistent data transmission:

- 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.

**6.3.1 S7 Transmits Data to a Communication Partner, 10 SEND\_440 FB**

**How FB SEND\_440 Works**

The SEND\_440 FB transfers a data field from a data block, specified by the DB\_NO, DBB\_NO and LEN parameters, to the CP 440. The SEND\_440 FB is called cyclically for data transmission or, alternatively, statically (without conditions) in a time-controlled program.

The data transmission is initiated by a positive edge at the REQ input. A data transmission operation can run over several calls (program cycles), depending on the amount of data involved.

The FB SEND\_440 function block can be called in the cycle when the signal state at the R parameter input is "1". This aborts the transmission to the CP 440 and sets the FB SEND\_440 back to its initial state. Data that has already been received by the CP 440 is still sent to the communication partner. If the R input is statically showing the signal state "1", this means that sending is deactivated.

The LADDR parameter specifies the address of the CP 440 to be addressed.

## Error Display on the SEND\_440 FB

The DONE output shows "request completed without errors". ERROR indicates whether an error has occurred. In STATUS the event number is displayed in the event of an error. If no error occurs the value of STATUS is 0. DONE and ERROR/STATUS are also output in response to a RESET of FB SEND\_440. In the event of an error, the binary result BR is reset. If the block is terminated without errors, the binary result has the status "1".

### Note

The SEND\_440 function block does not have a parameter check. If there are invalid parameters, the CPU branches to the STOP mode.

## Block call

STL representation	LAD representation
CALL SEND_440, I_SEND_440	
REQ : =	SEND_440, I_SEND_440
R : =	
LADDR : =	— EN ENO —
DB_NO : =	— REQ DONE —
DBB_NO : =	— R ERROR —
LEN : =	— LADDR STATUS —
DONE : =	— DB_NO —
ERROR : =	— DBB_NO —
STATUS : =	— LEN —

### Note

The parameters EN and ENO are only present in the graphical representation (LAD or FBD). The block is started with EN = TRUE. If the function is completed without errors, ENO = TRUE is set. To process these parameters, the compiler uses the binary result BR. The binary result is set to the signal state "1" if the block was terminated without errors. If there was an error, the BR is set to "0".

### Assignment in the Data Area

The SEND\_440 FB works with an instance DB (I\_SEND\_440). The DB number is specified in the call. The data in the instance DB cannot be accessed.

#### Note

Exception: If the error (STATUS == W#16#1E0F) occurs, you can consult the SFCERR variable for additional details (see the chapter "Diagnostics Messages of the Function Blocks (Page 97)").

This error variable can only be loaded via a symbolic access to the instance DB.

### Parameters of the SEND\_440 FB

The following table lists the parameters of the SEND\_440 FB.

Table 6-2 Parameters of the SEND\_440 FB

Name	Type	Data Type	Description	Permitted Values Comment
REQ	INPUT	BOOL	Initiates request with positive edge	
R	INPUT	BOOL	Aborts request	Current request is aborted. Sending is blocked.
LADDR	INPUT	INT	Basic address of CP 440	The basic address is taken from STEP 7.
DB_NO	INPUT	INT	Data block number	Send DB No.: CPU-specific, zero is not allowed
DBB_NO	INPUT	INT	Data byte number	0 ≤ DBB_NO ≤ 8190 Transmitted data as of data byte; Offset is CPU-specific
LEN	INPUT	INT	Data length	1 ≤ LEN ≤ 400 (see chapter "Overview of the Function Blocks (Page 73)"), specified in number of bytes
DONE <sup>1</sup>	OUTPUT	BOOL	Request completed without errors	Parameters STATUS == 16#00;
ERROR <sup>1</sup>	OUTPUT	BOOL	Request completed with errors	STATUS parameter contains error details
STATUS <sup>1</sup>	OUTPUT	WORD	Error specification	If ERROR == 1, STATUS parameter contains error details

<sup>1</sup> The parameter is available until the next call of the FB!

### Time Sequence Chart for the SEND\_440 FB

The figure below illustrates the behavior of the DONE and ERROR parameters, depending on how the REQ and R inputs are wired.

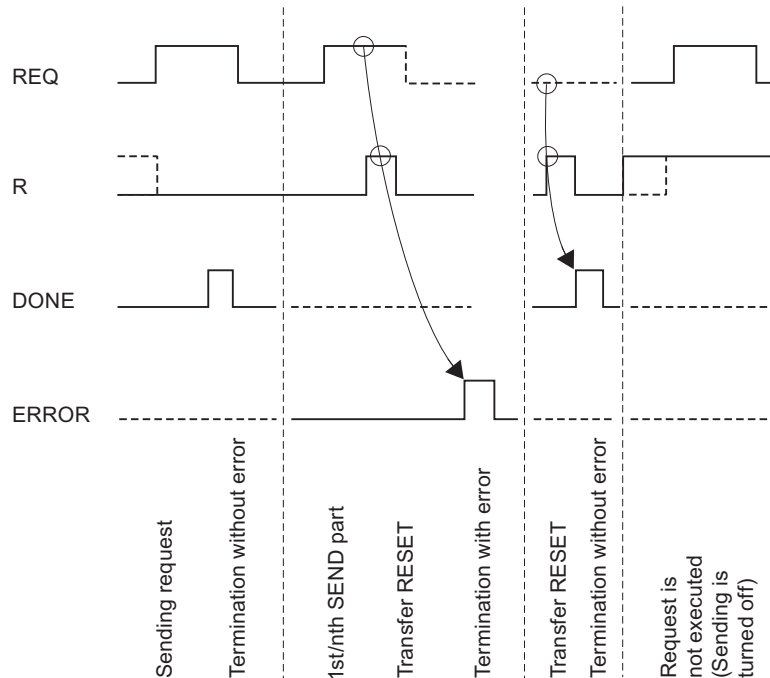


Figure 6-1 Time Sequence Chart for the 10 SEND\_440 FB

#### Note

The REQ input is edge-triggered. A positive edge at the REQ input is adequate. It is not required that the RLO (result of logical operation) is "1" during the whole transmission procedure.

## 6.3.2 S7 Receives Data from a Communication Partner, 9 RECV\_440 FB

### How FB SEND\_440 Works

The RECV\_440 FB transmits data from the CP 440 to an S7 data area specified by the DB\_NO, DBB\_NO and LEN parameters. The RECV\_440 FB is called cyclically for data transmission or, alternatively, statically in a time-controlled program (without conditions).

With the (static) signal state "1" at parameter EN\_R, the software checks whether data can be read by the CP 440. An active transmission can be aborted with signal state "0" at the EN\_R parameter. The aborted receive request is terminated with an error message (STATUS output). Receiving is deactivated as long as the EN\_R parameter has the signal state "0". A data transmission operation can run over several calls (program cycles), depending on the amount of data involved.

6.3 Using the Function Blocks

If the function block detects the signal state "1" at the R parameter, the current send job is aborted and the RECV\_440 FB is set to the initial state. Receiving is deactivated as long as the R parameter has the signal state "1".

The LADDR parameter defines the CP 440 to be addressed.

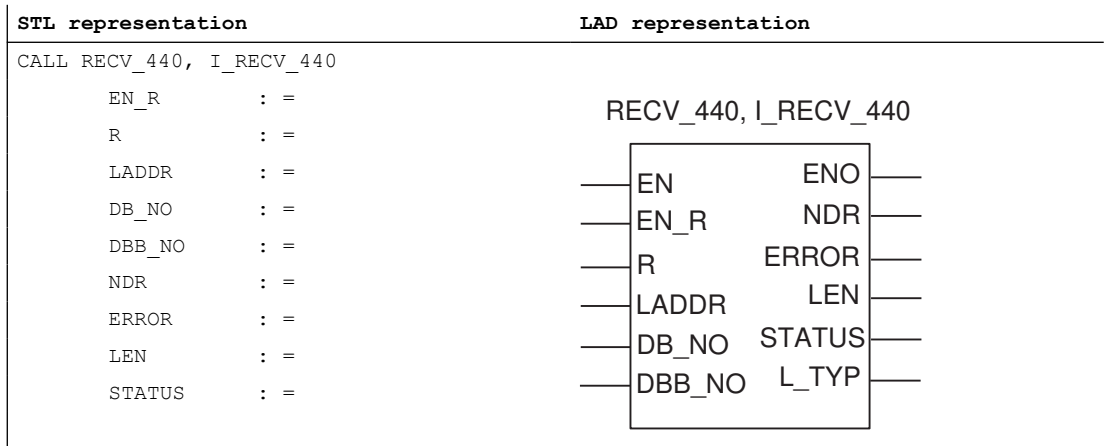
**Error Display on the RECV\_440 FB**

The NDR output shows "request completed without errors/data accepted" (all data read). ERROR indicates whether an error has occurred. In STATUS the event number is displayed in the event of an error. If no error occurs the value of STATUS is 0. NDR and ERROR/STATUS are also output in response to a RESET of FB RECV\_440 (LEN parameter == 16#00). In the event of an error, the binary result BR is reset. If the block is terminated without errors, the binary result has the status "1".

**Note**

The RECV\_440 function block does not have a parameter check. If there are invalid parameters, the CPU branches to the STOP mode.

**Block call**



**Note**

The parameters EN and ENO are only present in the graphical representation (LAD or FBD). The block is started with EN = TRUE. If the function is completed without errors, ENO = TRUE is set. 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".



## Assignment in the Data Area

The RECV\_440 FB works with an instance DB (I\_RECV\_440). The DB number is specified in the call. The data in the instance DB cannot be accessed.

### Note

Exception: If the error (STATUS == W#16#1EOE) occurs, you can consult the SFCERR variable for additional details (see the chapter "Diagnostics Messages of the Function Blocks (Page 97)").

. This error variable can only be loaded via a symbolic access to the instance DB.

## Parameters of the RECV\_440 FB

The following table lists the parameters of the RECV\_440 FB.

Table 6-3 Parameters of the RECV\_440 FB

Name	Type	Data Type	Description	Permitted Values, Comment
EN_R	INPUT	BOOL	Enables data read	
R	INPUT	BOOL	Aborts request	Current request is aborted. Receiving is blocked.
LADDR	INPUT	INT	Basic address of CP 440	The basic address can be found in the configuration table in STEP 7.
DB_NO	INPUT	INT	Data block number	Receive DB No.: CPU-specific, zero is not allowed
DBB_NO	INPUT	INT	Data byte number	Offset is CPU-specific
NDR <sup>1</sup>	OUTPUT	BOOL	Request completed without errors, data accepted	STATUS parameter == 16#00;
ERROR <sup>1</sup>	OUTPUT	BOOL	Request completed with errors	STATUS parameter contains error details
LEN <sup>1</sup>	OUTPUT	INT	Length of message frame received	$1 \leq \text{LEN} \leq 400$ (see chapter "Overview of the Function Blocks (Page 73)"), specified in number of bytes
STATUS <sup>1</sup>	OUTPUT	WORD	Error specification	If ERROR == 1, STATUS parameter contains error details

<sup>1</sup>The parameter is available until the next call of the FB!

### Time Sequence Chart for the RECV\_440 FB

The figure below illustrates the behavior of the NDR, LEN and ERROR parameters, depending on how the EN\_R and R inputs are wired.

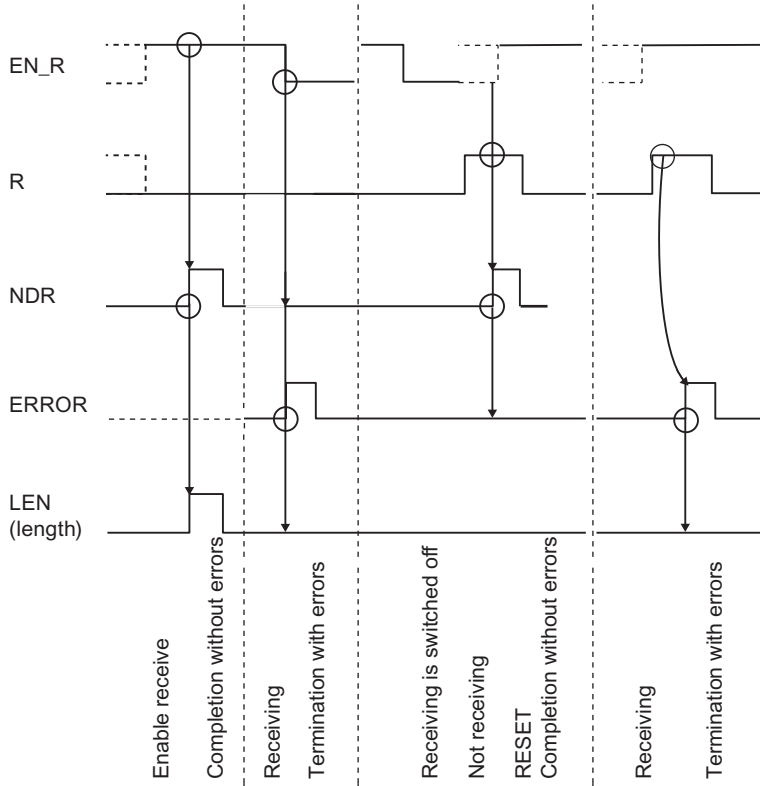


Figure 6-2 Time Sequence Chart for the 9 RECV\_440 FB

**Note**

The EN\_R must be set to static "1". During the receive request, the EN\_R parameter must be supplied with RLO "1" (result of logic operation).

### 6.3.3 Deleting the Receive Buffer (11 "RES\_RECV" FB)

The RES\_RECV FB deletes the entire receive buffer of the CP 440. All the stored message frames are discarded. A message frame coming in at the same time as the RES\_RECV FB call is saved.

The FB is activated by a positive edge at the REQ input. The job can run over several calls (program cycles).

The FB RES\_RECV function block can be called in the cycle when the signal state at the R parameter input is "1". This terminates deletion and sets the RES\_RECV FB back to its initial state.

The LADDR parameter specifies the address of the CP 440 to be addressed.

### Error Display on the RES\_RECV FB

The DONE output shows "request completed without errors". ERROR indicates whether an error has occurred. In STATUS the event number is displayed in the event of an error. If no error occurs the value of STATUS is 0. DONE and ERROR/STATUS are also output in response to a RESET of FB RES\_RECV. In the event of an error, the binary result BR is reset. If the block is terminated without errors, the binary result has the status "1".

---

#### Note

The RES\_RECV function block does not have a parameter check. If there are invalid parameters, the CPU branches to STOP mode.

---

### Block call

STL representation	LAD representation
CALL RES_RECV, I_RES_RECV	
REQ : =	
R : =	
LADDR : =	
DONE : =	
ERROR : =	
STATUS : =	
	RES_RECV, I_RES_RECV 

---

#### Note

The EN and ENO parameters are only present in the graphical representation (LAD or FBD). The block is started with EN = TRUE. If the function is completed without errors, ENO = TRUE is set. To process these parameters, the compiler uses the binary result. The binary result is set to signal state "1" if the block was terminated without errors. If there was an error, the binary result is set to "0".

---

**Assignment in the data area**

The RES\_RECV FB works with an instance DB (I\_RES\_RECV). The DB number is specified in the call. The data in the instance DB cannot be accessed.

**Note**

Exception: If the error STATUS == W#16#1E0F occurs, you can consult the SFCERR variable for more details of the error. This error variable can only be loaded via a symbolic access to the instance DB.

**Parameters of the RES\_RECV FB**

The following table lists the parameters of the RES\_RECV FB.

Table 6-4 Parameters of the RES\_RECV FB

Name	Type	Data type	Description	Permitted Values, Comment
REQ	INPUT	BOOL	Initiates request with positive edge	
R	INPUT	BOOL	Aborts request	Current request is aborted. Sending is blocked.
LADDR	INPUT	INT	Basic address of CP 440	The basic address is taken from STEP 7.
DONE <sup>1</sup>	OUTPUT	BOOL	Request completed without errors	STATUS parameter == 16#00;
ERROR <sup>1</sup>	OUTPUT	BOOL	Request completed with errors	STATUS parameter contains error details
STATUS <sup>1</sup>	OUTPUT	WORD	Error specification	If ERROR == 1, STATUS parameter contains error details
<sup>1</sup> The parameter is available until the next call of the FB!				

### Time Sequence Chart for the RES\_RECV FB

The figure below illustrates the behavior of the DONE and ERROR parameters, depending on how the REQ and R inputs are wired.

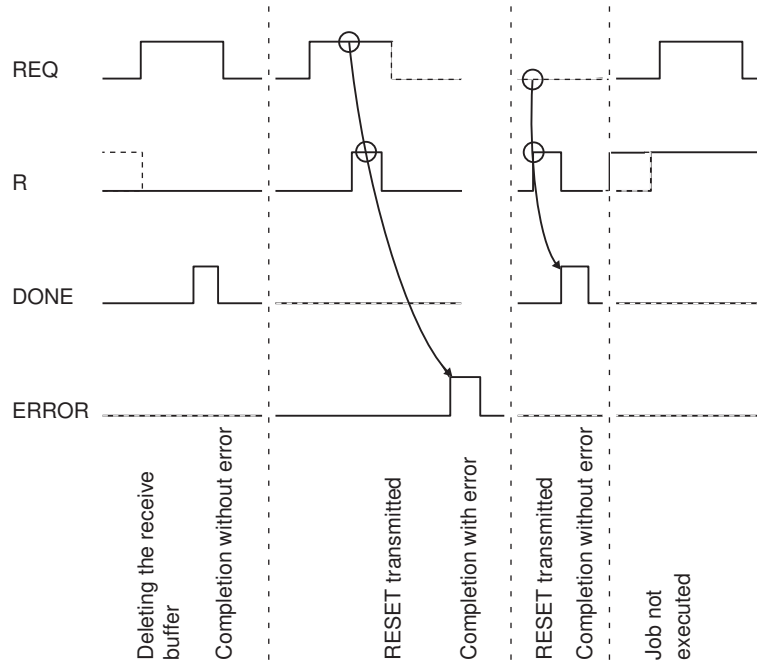


Figure 6-3 Time Sequence Chart for the 11 RES\_RECV FB

#### Note

The REQ input is edge-triggered. A positive edge at the REQ input is adequate. It is not required that the RLO (result of logical operation) is "1" during the whole transmission procedure.

## 6.4 Programming the Function Blocks

### 6.4.1 Supplying the block parameters

#### Direct/Indirect Parameter Assignment

With STEP 7 the data blocks cannot be indirectly assigned parameters (parameters transferred in the currently selected data block) as they can with STEP 5.

All block parameters accept both constants and variables, so the distinction between direct and indirect parameter assignment is no longer necessary in STEP 7.

**Example of "Direct Parameter assignment"**

Call of the FB 10 corresponding to "direct parameter assignment":

```

STL
-----
Network 1:
    CALL
        FB 10, DB10
        REQ          := M 0.6      //Activates SEND
        R            := M 5.0      //Activates RESET
        LADDR        := +336       //Basic address, PB336
        DB_NO        := +11        //Data block DB11
        DBB_NO       := +0         //As of data word DBB 0
        LEN          := +15        //Length 15 bytes
        DONE         := M 26.0     //Terminated without error
        ERROR        := M 26.1     //Terminated with error
        STATUS       := MW 27      //Status word

```

**Example of "Indirect Parameter assignment"**

Call of the FB 10 corresponding to indirect parameter assignment:

```

STL
-----
Network 1:
    CALL
        FB 10, DB10
        REQ          := M 0.6      //Activates SEND
        R            := M 5.0      //Activates RESET
        LADDR        := MW21       //Basic address in MW21
        DB_NO        := MW40       //DB no. in MW40
        DBB_NO       := MW42       //DBB No. in MW42
        LEN          := MW44       //Length in MW44
        DONE         := M 26.0     //Terminated without error
        ERROR        := M 26.1     //Terminated with error
        STATUS       := MW 27      //Status word

```

**Parameter assignment of Data Words**

The specification of data words (partially qualified specification) is not allowed because (depending on the actual operands) the currently selected data block can no longer be determined in the standard function. If a data operand is specified as an actual parameter, the fully qualified specification must always be used.

A fully qualified specification can be either absolute or symbolic. Mixed addressing with fully qualified data operands is rejected by the compiler.

### Example 1

The symbol name for the data block is entered in the symbol table, while the symbol name for the data operand is declared in the corresponding data block.

STL	
DB 20.DBW 0	Absolute fully qualified addressing
CP_DB.SEND_DWNORCV	Symbolic fully qualified addressing

### Example 2

The symbol name of the data block used, DB 20, is "CP\_DB"; the symbol name for the send DB number is "SEND\_DBNO" and is located in data block DB 20 in the data word DBW 0.

The start address of the send frame is "SEND\_DWNO" and is located in the data block DB 20 in DBW 2. The frame length is "SEND\_LEN" and is located in the data block DB 20 in DBW 4.

The variable used for the module address is the memory word "BGADR" (MW21), for the DONE parameter the flag "SEND\_DONE" (M26.0), for the ERROR parameter the memory bit "SEND\_ERROR" (M26.1), and for the STATUS parameter the memory word "SEND\_STATUS" (MW27).

The STL listings for the example are shown on the following page.

### Example of an Absolutely Addressed Actual Operand

Calling FB 10 with absolutely addressed actual operands:

STL	
Network 1:	
CALL	
FB 10, DB10	
REQ	:= M 0.6 //Activates SEND
R	:= M 5.0 //Activates RESET
LADDR	:= MW21 //Basic address in MW21
DB_NO	:= DB20.DBW0 //DB no. in DBW0 of DB20
DBB_NO	:= DB20.DBW2 //From DBB no., located in DBW2 of DB20
LEN	:= DB20.DBW4 //Length located in DBW4 of DB20
DONE	:= M 26.0 //Terminated without error
ERROR	:= M 26.1 //Terminated with error
STATUS	:= MW 27 //Status word

### Example of a "Symbolically Addressed Actual Operand"

Calling FB 10 with symbolically addressed actual operands:

STL	
Network 1:	
CALL	

6.5 General Information on Program Processing

STL		
FB 10, DB10		
REQ	:= SEND_REQ	//Activates SEND
R	:= SEND_R	//Activates RESET
LADDR	:= BGADR	//Basic address
DB_NO	:= CP_DB.SEND_DBNR	//Send DB no.
DBB_NO	:= CP_DB.SEND_DWNR	//Message frame as of data word
LEN	:= CP_DB.SEND_LAE	//Message frame length
DONE	:= SEND_DONE	//Terminated without error
ERROR	:= SEND_ERROR	//Terminated with error
STATUS	:= SEND_STATUS	//Status word

EN/ENO Mechanism

The parameters EN and ENO are only present in the graphical representation (LAD or FBD). The block is started with EN = TRUE. If the function is completed without errors, ENO = TRUE is set. To process these parameters, the compiler uses the binary result (BR).

The binary result is set to the signal state "1" if the block was terminated without errors. If there was an error, the binary result is set to "0".

## 6.5 General Information on Program Processing

### Startup Characteristics of the CP 440 Programmable Logic Controller

The configuration data is created with the aid of the **CP 440: Point-to-Point Communication, Parameter Assignment** parameter assignment and transferred to the CPU with the STEP 7 software. Each time the CPU is started up, the current parameters are transferred to the CP 440 by the system service of the CPU.

### Startup Characteristics, FB-CP 440

Once the connection between the CPU and the CP 440 has been established, the CP 440 must be initialized.

For each function block (SEND\_440, RECV\_440, RES\_RECV) there is a separate startup coordination. Before requests can be actively processed, the accompanying startup procedure must be completed.

### Disabling alarms

In the function blocks the interrupts are not disabled.



## Addressing the Module

The logical basic address is defined via STEP 7 and must be specified by the user under the block parameter LADDR.

## 6.6 Technical Specifications of the Function Blocks

### Introduction

The following lists the technical specifications for the memory requirements, runtimes, minimum number of the CPU cycles and the system functions used.

### Memory requirements

The table below contains the memory requirements of the function blocks of the CP 440.

Table 6-5 Memory Requirements of the Function Blocks (in Bytes)

Block	Name	Version	Load memory	Working memory	Local data	Instance DB
FB 9	RECV_440	1.1	1240	1006	26	36
FB 10	SEND_440	1.1	1062	846	26	36
FB 11	RES_RECV	1.0	894	710	38	26

### Minimum Number of CPU Cycles

The table below describes the minimum number of CPU cycles (FB/FC calls) required to process a "minimum request" (32 bytes SEND/RECEIVE for the transported user data set per program cycle). This applies only in centralized operation.

Table 6-6 Minimum Number of CPU Cycles

	Number of CPU Cycles for Processing ...		
	Completion without error	Completion with error	RESET/RESTART
RECV_440	≥ 2	≥ 2	≥ 3
SEND_440	≥ 2	≥ 2	≥ 3
RES_RECV	≥ 2	≥ 2	≥ 3

Before the CP 440 can process an initiated job after the CPU has changed from STOP to RUN mode, the CP-CPU startup mechanism SEND\_440 must be completed. Any requests initiated in the meantime do not get lost. They are transmitted once the startup coordination with the CP 440 is finished.

Before the CP 440 can receive or submit a frame in the user program after a change in the CPU mode from STOP to RUN, the CP-CPU startup mechanism RECV\_440 must be completed.



# Startup Characteristics and Operating Mode Transitions of the CP 440

## 7.1 Operating Modes of the CP 440

The CP 440 has the operating modes STOP, parameter reassignment and RUN.

### STOP

When the CP 440 is in STOP mode, no protocol driver is active and all send and receive requests from the CPU are given a negative acknowledgment.

The CP 341 remains in STOP mode until the cause of the stop is removed (e.g. break, invalid parameter).

### Parameter reassignment

For parameter reassignment, the protocol driver is initialized.

Sending and receiving are not possible, and send and receive message frames stored in the CP 440 are lost when the driver is restarted. Communication between the CP and the CPU is started afresh (active message frames are aborted).

At the end of the parameter reassignment, the CP 440 is ready to send and receive.

### RUN

The CP 440 processes the requests from the CPU. It provides the message frames received by the communications partner to be fetched by the CPU.

## 7.2 Startup Characteristics of the CP 440

### Startup Phases of the CP 440

The CP 440 start-up is divided into two phases:

- Initialization (CP 440 in POWER ON mode)
- Parameter assignment

### Initialization

As soon as the CP 440 is energized, all module components are initialized.

### Parameter Assignment

Parameter assignment means that the CP 440 receives the module parameters assigned to the current slot as created with the **CP 440: Point-to-Point Communication, Parameter Assignment** programming interface.

Parameter reassignment is performed. The CP 440 is now ready for operation.

## 7.3 Behavior of the CP 440 on Operating Mode Transitions of the CPU

Once the CP 440 has been started up, all the data is exchanged between the CPU and the CP 440 via the function blocks.

### CPU STOP

In CPU STOP mode, communication via the S7 backplane bus is not possible. Any active CP CPU data transmission, including both send and receive message frames, is aborted and the connection is reestablished.

Data traffic at the interface of the CP 440 is continued with the ASCII driver in the case of parameter assignment without flow control. In other words, the current send request is completed. Receiving the data depends on the setting of the basic parameter "Reaction to CPU Stop" (see section "Basic Parameters of the CP 440 (Page 57)").

### CPU STARTUP

On start-up, the CP sends the parameters created with the **CP 440: Point-to-Point Communication, Parameter Assignment** programming interface. The CP 440 only parameter reassignment if the parameters have changed.

### CPU RUN

When the CPU is in RUN mode, sending and receiving are unrestricted. In the first FB cycles following the CPU restart, the CP 440 and the corresponding FBs are synchronized. No new FB is executed until this is finished.

### Points to Note when Sending Message Frames

Message frames can be sent only when the CPU status is RUN.

If the CPU goes into STOP during data transmission from the CPU "CP, the SEND\_440 FB reports after restart that the current program has been interrupted and the job has been terminated due to BREAK/restart/reset.

---

#### Note

The CP 440 does not send data to the communications partner until it has received all data from the CPU.

---

## 7.4 Behavior of the Sender Line Drivers of the Serial Interface During Particular Operating Modes of the CP 440

### Behavior of the Sender Line Drivers

Table 7-1 Behavior of the Sender Line Drivers

Status	Line Drivers
The CP 440 is deenergized	High resistance
The CP 440 is in the expansion rack with its own power supply, but the central controller is switched off.	High resistance
No parameters are assigned to the CP 440.	High resistance
Parameters are assigned to the CP 440; the sender is in an idle state <ul style="list-style-type: none"> <li>• 3964</li> <li>• ASCII, operating mode RS422, multipoint, master</li> <li>• ASCII, operating mode RS422, multipoint, slave</li> <li>• ASCII, RS485</li> </ul>	<ul style="list-style-type: none"> <li>• High level</li> <li>• High level</li> <li>• High level</li> <li>• High resistance</li> </ul>



# Diagnostics with the CP 440

## 8.1 Diagnostics Functions of the CP 440

### Introduction

The diagnostics functions of the CP 440 enable you to quickly localize any errors which occur. The following diagnostics options are available:

- Diagnostics via the display elements of the CP 440
- Diagnosis via the STATUS output of the function blocks
- Diagnostics via the diagnostic buffer of the CP 440

### Display elements (LEDs)

The display elements show the operating mode or possible error states of the CP 440. The display elements give you an initial overview of any internal or external errors as well as interface-specific errors (see chapter "Diagnostics via the display elements of the CP 440 (Page 96)").

The special LEDs, TXD and RXD give you information on the state of the send and receive lines.

The chapter "Firmware updates (Page 69)" provides information on other LED indicators that occur when loading a firmware update.

### STATUS output of the FBs

The function blocks FB\_SEND\_440, FB\_RECV\_440 and RES\_RECV have a STATUS output to indicate an error diagnosis. Reading the STATUS output of the function blocks gives you information on errors which have occurred during communication. You can interpret the STATUS output in the user program (see chapter "Diagnostics Messages of the Function Blocks (Page 97)").

The CP 440 also enters the diagnostic events at the STATUS output in its diagnostic buffer.

### Diagnostic Buffer of the CP 440

All the CP 440's errors are entered in its diagnostic buffer.

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 user-relevant information of the CP diagnostic buffer (see chapter "Diagnostics via the diagnostic buffer of the CP 440 (Page 104)").

## 8.2 Diagnostics via the display elements of the CP 440

### Display Elements of the CP 440

The display elements of the CP 440 provide information on the CP 440. The following display functions can be distinguished:

- Group error displays
  - INTF internal error
  - EXTF external error
- Special displays
  - TXD (green) sending active; lights up when the CP 440 is sending user data via the interface
  - RXD (green) receiving active; lights up when the CP 440 is receiving user data via the interface
- Interface fault LED
  - FAULT interface error

### 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 440 Display Elements

Error Display	Error Description	To correct or avoid errors
INTF comes on	CP 440 signals internal fault, e.g. protocol fault	Read the STATUS of the FB, or read the diagnostic buffer of the CP 440.
EXTF comes on	CP 440 signals external fault, e.g. break on the line.	Read the STATUS of the FB, or read the diagnostic buffer of the CP 440.
FAULT off	Interface ready for operation	-
FAULT flashing slowly	Interface initialized and ready for operation but communication via S7-400 backplane bus not possible.	Check the configuration for incorrect entries (e.g. slot, ID no., etc.).
FAULT flashing fast	Incorrect parameter	Check the parameter settings in the <b>CP440: Point-to-Point Connection Parameter Assignment</b> interface.
FAULT comes on	No interface parameters	Assign parameters with the <b>CP440: Point-to-Point Connection Parameter Assignment</b> interface.
INTF and FAULT light up at the same time	CPU reset (module not configured/has not been assigned parameters)	Configure and assign parameters to the CP 440 and load the HW configuration to the CPU.

### See also

Subsequent Loading of Firmware Updates (Page 69)



## 8.3 Diagnostics Messages of the Function Blocks

### Error Diagnostics

Each function block has a STATUS parameter for error diagnostics. The STATUS message numbers always have the same meaning, irrespective of which function block is used.

### Numbering Scheme for Event Class/Event Number

The figure below illustrates the structure of the STATUS parameter.

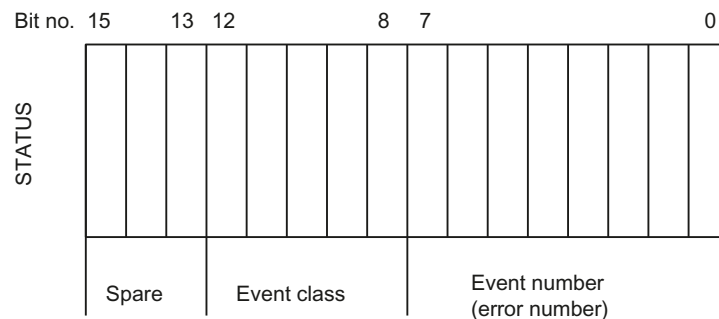


Figure 8-1 Structure of the STATUS Parameter

### Example

The figure below illustrates the contents of the STATUS parameter for the event "Request aborted due to complete restart, restart or reset" (event class: 1EH, event number 0DH).

Event: "Request canceled due to complete restart, restart, or reset"

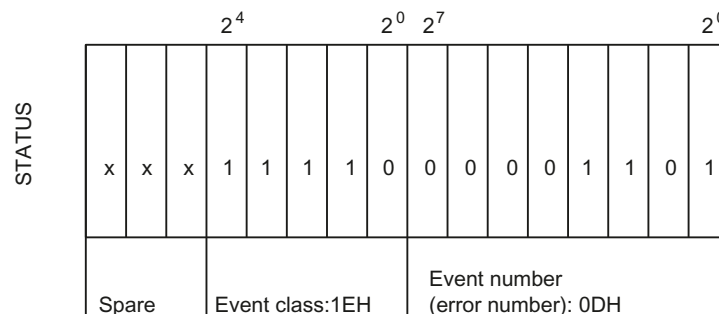


Figure 8-2 Example: Structure of the STATUS Parameter for the Event "Request Aborted Due to Complete Restart, Restart or Reset"

## Event Classes

The table below describes the various event classes and numbers.

Table 8-2 Event Classes and Event Numbers

Event Class 0 (00H): "CP startup"		
Event No.	Event Text	To correct or avoid errors
(00)01H	Initialization of the CP completed	-
(00)03H	Point-to-point parameters applied	-
(00)04H	Parameter already on CP (timers match)	-
(00)07H	Status transition: CPU to STOP	-
(00)08H	Status transition: CPU to RUN/STARTUP	-

Event Class 2 (02H): "Initialization error"		
Event No.	Event Text	To correct or avoid errors
(02)01H	No parameters Parameter memory empty or has unknown contents	Load interface parameters.
(02)04H	Impermissible character frame	Correct the impermissible parameter assignment, load the parameters onto the module, and carry out a cold restart.
(02)05H	Impermissible transmission rate	Correct the impermissible parameter assignment, load the parameters onto the module, and carry out a cold restart.
(02)0FH	Invalid parameter assignment detected at start of parameter communication. Interface could not be parametered.	Correct invalid parameters and restart.
(02)12H	Impermissible <ul style="list-style-type: none"> <li>• Monitoring time</li> <li>• Character Delay Time</li> </ul> With 3964(R) only: <ul style="list-style-type: none"> <li>• Acknowledgment delay time</li> </ul>	Correct the impermissible parameter assignment, load the parameters onto the module, and carry out a cold restart.

Event Class 5 (05H): "Error processing a CPU request"		
Event No.	Event	To correct or avoid errors
(05)01H	Current request aborted as a result of CP restart.	No remedy is possible at POWER ON. When changing the parameters of the CP in 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 parameters are not set).	Set the parameters for the device interface.

Event Class 5 (05H): "Error processing a CPU request"		
Event No.	Event	To correct or avoid errors
(05)0EH	<ul style="list-style-type: none"> <li>Invalid message frame length</li> <li>or</li> <li>The configured end-of-text characters did not occur within the maximum permissible length.</li> </ul>	<ul style="list-style-type: none"> <li>The message frame is &gt; 400 bytes (see chapter "Overview of the Function Blocks (Page 73)"). Select a shorter message frame length.</li> <li>or</li> <li>Add the end-of-text characters at the place you want in the send buffer.</li> </ul>
(05)17H	Transmission length > 400 bytes (see chapter "Overview of the Function Blocks (Page 73)") is too large for the CP.	Split the job up into several shorter jobs.
(05)1DH	Send/receive job interrupted by: <ul style="list-style-type: none"> <li>Communication block reset</li> <li>Parameter reassignment</li> </ul>	Repeat the call for the communication block.
(05)22H	A new SEND job has been started, although the old job has not yet been completed.	Only start the new SEND job when the old job has been completed with DONE or ERROR.

Event Class 7 (07H): "Send error"		
Event No.	Event	To correct or avoid errors
(07)01H	With 3964(R) only: Sending the first repetition: <ul style="list-style-type: none"> <li>An error was detected during transmission of the message frame, or</li> <li>The partner requested a repetition by means of a negative acknowledgment code (NAK).</li> </ul>	A repetition is not an error, however, it can be an indication that there are disturbances on the transmission line or a malfunction of the partner device. If the message frame has still not been transmitted after the maximum number of repetitions, an error number describing the first error that occurred is output.
(07)02H	With 3964(R) only: Error during connection setup: After STX was sent, NAK or any other code (except for DLE or STX) was received.	Check for malfunctioning of the partner device, possibly using an interface test device switched into the transmission line.
(07)03H	With 3964(R) only: 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 on the send line, for example. Identify the malfunctioning of the partner device, possibly by using an interface test device switched into the transmission line.
(07)04H	With 3964(R) only: Termination by partner: During current send operation, one or more characters were received by partner.	Check whether the partner is also showing an error, possibly because not all transmission data has arrived (e.g. due to break on line) or due to serious faults or because the partner device has malfunctioned. Identify the malfunction, possibly by using an interface test device switched into the transmission line.
(07)05H	With 3964(R) only: Negative acknowledgment during sending	Check if 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. Identify the malfunction, possibly by using an interface test device switched into the transmission line.

Event Class 7 (07H): "Send error"		
Event No.	Event	To correct or avoid errors
(07)06H	With 3964(R) only: Error at end of connection: <ul style="list-style-type: none"> <li>Partner rejected message frame at end of connection with NAK or a random string (except for DLE), or</li> <li>Acknowledgment code (DLE) received too early.</li> </ul>	Check whether the partner is also showing an error, possibly because not all transmission data has arrived (e.g. due to break on line) or due to serious faults or because the partner device has malfunctioned. Identify the malfunction, possibly by using an interface test device switched into the transmission line.
(07)07H	With 3964(R) only: 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. Identify the malfunction, possibly by using an interface test device switched into the transmission line.
(07)08H	For ASCII driver only: The 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 parameters.
(07)09H	With 3964(R) only: Connection setup not possible. Number of permitted setup attempts exceeded.	Check the interface cable or the transmission parameters. Also check that receive function between CPU and CP is correctly configured at the partner device.
(07)0AH	With 3964(R) only: The data could not be transmitted. The permitted number of transfer attempts was exceeded.	Check the interface cable or the transmission parameters.
(07)0BH	With 3964(R) only: Initialization conflict cannot be solved because both partners have high priority.	Change the parameter assignment.
(07)0CH	With 3964(R) only: Initialization conflict cannot be solved because both partners have low priority.	Change the parameter assignment.

Event Class 8 (08H): "Receive error"		
Event No.	Event	To correct or avoid errors
(08)01H	With 3964(R) only: Expecting the first repetition: An error was recognized on receiving a telegram and the CP requested repetition from the partner via a negative acknowledgment (NAK).	A repetition is not an error, however, it can be an indication that there are disturbances on the transmission line or a malfunction of the partner device. If the message frame has still not been transmitted after the maximum number of repetitions, an error number describing the first error that occurred is output.
(08)02H	With 3964(R) only: Error during connection setup: <ul style="list-style-type: none"> <li>In idle mode, one or more random codes (other than NAK or STX) were received, or</li> <li>after an STX was received, partner sent more codes without waiting for response DLE.</li> </ul> After the partner has signaled POWER ON: <ul style="list-style-type: none"> <li>While partner is being activated, CP receives an undefined code.</li> </ul>	Identify the malfunction on the partner device, possibly by using an interface test device is switched into the transmission line.
(08)05H	With 3964(R) only: Logical error during receiving: After DLE was received, a further random code (other than DLE or ETX) was received.	Check if partner DLE in message frame header and in data string is always in duplicate or the connection is released with DLE ETX. Identify the malfunction on the partner device, possibly by using an interface test device is switched into the transmission line.
(08)06H	Character delay time exceeded: <ul style="list-style-type: none"> <li>Two successive characters were not received within character delay time, or</li> </ul> With 3964(R) only: <ul style="list-style-type: none"> <li>1. character after sending of DLE during connection setup was not received within character delay time.</li> </ul>	Partner device faulty or too slow. Identify the malfunction, possibly by using an interface test device switched into the transmission line.
(08)07H	Impermissible message frame length: A message frame of length 0 was received.	Receipt of a message frame of length 0 is not an error. Check why the communication partner is sending message frames without user data.
(08)08H	With 3964(R) only: Error in block check character (BCC) Internally calculated value of BCC does not match BCC received by partner at end of connection.	Check whether connection is badly damaged; in this case you may also occasionally see error codes. Identify the malfunction on the partner device, possibly by using an interface test device is switched into the transmission line.
(08)09H	With 3964(R) only: Waiting time for block repetition has expired	Configure the same block wait time in the communication partner as for the CP 440. Identify the malfunction in the communication partner, possibly by using an interface test device switched into the transmission line.
(08)0AH	There is no free receive buffer available: No empty receive buffer was available to receive data.	The RECV_440 FB must be called more frequently.

## 8.3 Diagnostics Messages of the Function Blocks

Event Class 8 (08H): "Receive error"		
Event No.	Event	To correct or avoid errors
(08)0CH	Transmission error: <ul style="list-style-type: none"> <li>Transmission error (parity error, stop bit error, overflow error) detected.</li> </ul> With 3964(R) only: <ul style="list-style-type: none"> <li>If faulty character is received in idle mode, the error is reported immediately so that disturbances on the transmission line can be detected early.</li> </ul> With 3964(R) only: <ul style="list-style-type: none"> <li>If this occurs during send or receive operation, repetitions are initiated.</li> </ul>	Faults on the transmission line cause message frame repetitions, thus lowering user data throughput. Danger of an undetected error increases. Change your system setup or the line routing.  Check the cable of the communications partner or check whether both devices have same setting for baud rate, parity and number of stop bits.
(08)0DH	BREAK: Receive line to partner is interrupted.	Reconnect or switch partner on again.
(08)0EH	Receive buffer overflow when flow control not enabled.	The function block to be received must be called more often in the user program, or communication with flow control must be configured.
(08)10H	Parity error	Check the cable of the communications partner or check whether both devices have same setting for baud rate, parity and number of stop bits.
(08)11H	Character frame error	Check the cable of the communications partner or check whether both devices have same setting for baud rate, parity and number of stop bits.  Change your system setup or the line routing.
(08)12H	For ASCII driver only: More characters were received after the CP had sent XOFF or set CTS to OFF.	Reassign the parameters for the communication partner, or deal with the data in the CP more quickly.
(08)14H	For ASCII driver only: A message frame or several message frames have got lost, because you have been working without flow control.	Work with flow control as much as you can. Use the entire receive buffer. Set the "Reaction to CPU Stop" parameter in the basic parameters to "Continue".
(08)16H	The length of a received message frame was longer than the maximum specified length.	Correction required at the partner
(08)20H	Static error when calling the SFC RD_REC. The RET_VAL return value of the SFC is provided in the SFCERR variable in the instance DB for evaluation.	Analyze the SFCERR variable from the instance DB.

Event class 11 (0B8H): "Warnings"		
Event No.	Event	To correct or avoid errors
(0B)01H	Receive buffer more than 2/3 full	Call the receive block more often to avoid the receive buffer overflowing.

Event class 128 (80H): "Module error"		
Event No.	Event	To correct or avoid errors
(80)00H	Module firmware not found	Perform a firmware update

## Display and evaluate STATUS output

You can display and interpret the actual operands in the STATUS output of the function blocks.

### Note

An error message is only output if the ERROR bit (request completed with error) is set. In all other cases the STATUS word is zero.

## Event Class 30

Event class 30 contains error messages which might occur during communication between the CP 440 and the CPU via the S7 backplane bus.

The table below describes event class 30.

Table 8-3 Event Class 30

Event class 30 (1EH): "Error during communication between CP and CPU"		
Event No.	Event	Further Information/Remedy
(1E)0DH	<ul style="list-style-type: none"> <li>Job aborted due to a cold restart, warm restart or reset</li> <li>Module firmware does not exist</li> </ul>	<ul style="list-style-type: none"> <li>Repeat the job.</li> <li>Check whether the module is inserted.</li> </ul>
(1E)0EH	Static error when the RD_REC SFC was called. Return value RET_VAL of SFC is available for evaluation in SFCERR variable in instance DB.	Load SFCERR variable from instance DB.
(1E)0FH	Static error when the WR_REC SFC was called. Return value RET_VAL of SFC is available for evaluation in SFCERR variable in instance DB.	Load SFCERR variable from instance DB.
(1E)41H	Number of bytes set in LEN parameter of FBs illegal.	Keep to the value range of 1 to 400 bytes (see chapter "Overview of the Function Blocks (Page 73)").

## Calling the SFCERR Variable

You can obtain more information on errors 14 (1E0EH) and 15 (1E0FH) in event class 30 by means of the SFCERR variable.

You can load the SFCERR variable from the instance DB belonging to the corresponding function block.

The error messages entered in the SFCERR variable are described in the section on the system functions SFC 58 "WR\_REC" and SFC 59 "RD\_REC" in the *System Software for S7-300/400, System and Standard Functions* reference manual.

## See also

Serial Transmission of a Character (Page 17)

## 8.4 Diagnostics via the diagnostic buffer of the CP 440

### Diagnostic Buffer of the CP 440

The CP 440 has its own diagnostic buffer, in which all the diagnostic events of the CP 440 are entered in the order in which they occur.

The following are displayed in the diagnostic buffer of the CP 440:

- Initialization and parameter 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 440 or to trace the occurrence of individual diagnostic events.

---

#### Note

The diagnostic buffer is a ring buffer for a maximum of 10 diagnostic entries. When the diagnostic buffer is full, the oldest entry is deleted when a new entry is made in it. This means that the most recent entry is always first. The contents of the diagnostics buffer are lost in the event of a POWER OFF or when the CP 440 is reconfigured.

It is not possible to display the time.

---

### Reading the Diagnostic Buffer at the Programming Device

The contents of the diagnostic buffer of the CP 440 can be read by means of the STEP 7 information functions.

All the user-relevant information in the CP diagnostic buffer is displayed to you in the "Diagnostic Buffer" dialog box on the "Module Information" tab. You can access the "Module Information" tab under STEP 7 from SIMATIC Manager.

Prerequisite: 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).

#### To do this, proceed as follows:

1. Open the relevant SIMATIC 400 station (double-click or select menu command **Edit > Open**).
2. Then open the "Hardware" object (double-click or select menu command **Edit > Open**).  
**Result:**The window containing the configuration table appears.
3. Select the CP 440 in the configuration table.



4. Select **PLC > Module Information**.

**Result:** The "Module Information" tab appears for the CP 440. The "General" tab is displayed by default the first time you call it.

5. Select the "Diagnostic Buffer" tab.

**Result:** The latest diagnostic events of the CP 440 are displayed in plain text on the "Diagnostic Buffer" tab. Any "result details" on the cause of the problem appears in the lower part of the tab.

The event's numeric code is displayed in the "Event ID" field. The 16#F1C8 prefix is fixed. By clicking the "Help on Event" button you can display the help text on the event text.

If you click the "Update" button, the current data is read from the CP 440. By clicking the "Help on Event" button you can display a help text on the selected diagnostic event with information on error correction.



# Programming Example for Standard Function Blocks

## 9.1 General Information

### Introduction

The project example given here and contained in the CP440\_PtP\_Com project describes the standard functions for operating the CP 440 communication processor.

The example can be executed with the minimum hardware equipment. The STEP 7 function monitor/modify variables is also used (e.g. to modify transmitted data).

### Objective

The project example is intended to do the following:

- aims to show the most important functions,
- to enable the correct functioning of the connected hardware to be checked (the example is therefore simple and easy to follow)
- can easily be extended for your own purposes.

**The project example consists of the following five separate components:**

- CP440 SEND RECV: Interconnection with SEND and RECV for ASCII/3964(R)
- CP440 1CYC: Interconnection with SEND and RECV for ASCII/3964(R), cyclic job processing
- CP440 ASCII BCC: Interconnection for ASCII with block check formation
- CP440 MASTER: Master station for interconnection with ASCII protocol
- CP440 SLAVE: Slave station for interconnection with ASCII protocol

The CP 440 is assigned parameters by the CPU when the latter is started up (system service).

## 9.2 Device Configuration

### Application

To try out the program example, you could use the following devices:

- One S7-400 PLC (mounting rack, power supply, CPU)
- One CP 440 module with a communications partner (e.g. a second CP), or you could plug in a "short-circuit connector", i.e. the send line is bridged to the receive line
- A programming device (e.g. PG 740)

## 9.3 Settings

### Settings in the CPU by means of STEP 7

Use STEP 7 to configure your controller as follows.

- Slot 1: Power supply
- Slot 2: CPU
- Slot 4: CP 440, start address 512
- Slot 5: CP 440, start address 528

There is no CP 440 on slot 5 in the "CP440 MASTER" and "CP440 SLAVE" examples.

### Settings on the CP 440

You cannot make any hardware settings on the CP 440.

Use STEP 7 to configure all relevant data, including the parameters for the CP 440 with the **CP 440: Point-to-Point Communication, Parameter Assignment** interface and upload them to the CPU.

You can run the "CP440 SEND RECV" or "CP440 1CYC" program example without making any changes in the user program with the following:

- 3964(R) Procedure
- ASCII driver with "on expiry of character delay time" end criterion
- ASCII driver with "on receipt of fixed message frame length" end criterion.

For the ASCII driver with the "on receipt of the end character(s)" end criterion, you must also program the end codes.

## 9.4 Blocks Used

### Blocks Used

The following tables contain a list of the blocks used for the program examples.

Table 9-1 CP 440 SEND RECV

Block	Symbol	Description
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Cold restart processing
DB 21	SEND IDB	Instance DB for SEND_440 FB
DB 22	RECV IDB	Instance DB for RECV_440 FB
DB 40	SEND WORK DB	Work DB for the standard FB 10
DB 41	RECV WORK DB	Work DB for the standard FB 9
DB 42	SEND SRC DB	Send data block

Block	Symbol	Description
DB 43	RECV_DST_DB	Receive data block
FB 9	RECV_440	Receive standard FB for data
FB 10	SEND_440	Send standard FB for data
FC 21	SEND	Sending Data
FC 22	RECEIVE	Receiving data

Table 9-2 CP 440 1 CYC

Block	Symbol	Description
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Cold restart processing
DB 21	SEND_IDB	Instance DB for SEND_440 FB
DB 22	RECV_IDB	Instance DB for RECV_440 FB
DB 40	SEND_WORK_DB	Work DB for the standard FB 10
DB 41	RECV_WORK_DB	Work DB for the standard FB 9
DB 42	SEND_SRC_DB	Send data block
DB 43	RECV_DST_DB	Receive data block
FB 9	RECV_440	Receive standard FB for data
FB 10	SEND_440	Send standard FB for data
FC 21	SEND	Sending Data
FC 22	RECEIVE	Receiving data

Table 9-3 CP 440 ASCII BCC

Block	Symbol	Description
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Cold restart processing
FB 9	RECV_440	Receive standard FB for data
FB 10	SEND_440	Send standard FB for data
FC 21	SEND	Sending Data
FC 22	RECEIVE	Receiving data
FC 23	GEN_BCC	Creates BCC for the sender
FC 24	CHK_BCC	Checks the BCC when data is received
DB 21	SEND_IDB	Instance DB for SEND_440 FB
DB 22	RECV_IDB	Instance DB for RECV_440 FB
DB 40	SEND_WORK_DB	Work DB for the standard FB 10
DB 41	RECV_WORK_DB	Work DB for the standard FB 9
DB 42	SEND_SRC_DB	Send data block
DB 43	RECV_DST_DB	Receive data block

9.4 Blocks Used

Table 9-4 CP 440 MASTER

Block	Symbol	Description
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Cold restart processing
FB 9	RECV_440	Receive standard FB for data
FB 10	SEND_440	Send standard FB for data
FC 1	SLAVE01	Main program for communication to slave 01
FC 2	SLAVE02	Main program for communication to slave 02
FC 11	SEND SL01	Sending Data
FC 12	SEND SL01	Sending Data
FC 21	RECEIVE SL01	Receiving data
FC 22	RECEIVE SL02	Receiving data
DB 9	RCV IDB	Instance DB for RECV_440 FB
DB 10	SEND IDB	Instance DB for SEND_440 FB
DB 31	SEND WORK DB SL01	Work DB for the standard FB 10
DB 32	SEND WORK DB SL02	Work DB for the standard FB 10
DB 41	RCV WORK DB SL01	Work DB for the standard FB 9
DB 42	RCV WORK DB SL02	Work DB for the standard FB 9
DB 51	SEND SRC DB SL01	Send data block
DB 52	SEND SRC DB SL02	Send data block
DB 61	RECV DST DB SL01	Receive data block
DB 62	RECV DST DB SL02	Receive data block

Table 9-5 CP 440 SLAVE

Block	Symbol	Description
OB 1	CYCLE	Cyclic program processing
OB 100	RESTART	Cold restart processing
FB 9	RECV_440	Receive standard FB for data
FB 10	SEND_440	Send standard FB for data
FB 40	SLAVE FB	Slave for the CP 440
DB 9	RCV IDB	Instance DB for RECV_440 FB
DB 10	SEND IDB	Instance DB for SEND_440 FB
DB 40	IDB SLAVE 01	Instance DB for slave 01
DB 51	SEND SRC DB SL01	Send data block
DB 52	SEND SRC DB SL02	Send data block
DB 61	RECV DST DB SL01	Receive data block
DB 62	RECV DST DB SL02	Receive data block
DB 140	IDB SLAVE 02	Instance DB for slave 02

## 9.5 Installation, Error Messages

### Scope of Supply and Installation

The project example of the CP 440 and the **CP 440: Point-to-Point Communication, Parameter Assignment** interface is supplied together with the function blocks and this manual on a CD.

The program examples are installed together with the parameter assignment interface and described in the chapter "Installing the Programming Interface (Page 56)". After installation, the examples are stored in the following project:

#### **CP440\_PtP\_Com**

Open the project in the STEP 7 SIMATIC Manager by selecting **File > Open > Project**.

The project example is available both compiled and as a source file. A list of all the symbols used in the example is also included.

If there is no second CP 440 available to serve as a communication partner, you have to delete the CP 440 in HW Config by selecting **Edit > Delete**. The call of the FC 22 (FC for Receive) must also be commented out of the examples "CP440 SEND RECV" , "CP440 1CYC" and "CP440 ASCII BCC" in OB 1.

### Download to the CPU

The hardware for the example is completely set up and the programming device is connected.

After the overall reset of the CPU (operating mode STOP), transfer the complete example to the user memory. Then use the operating mode switch to change from STOP to RUN.

### Malfunction

If an error occurs at startup, the cyclically processed block calls are terminated with an error.

If there is an error message, the parameter output ERROR of the modules is set. A more precise error description is then stored in the STATUS parameter of the blocks. If STATUS contains one of the error messages 16#1E0E or 16#1E0F, the exact error description is stored in the SFCERR variable in the instance DB.

## 9.6 Activation, Startup Program and Cyclic Program

### 9.6.1 "CP440 SEND RECV" Program Example

#### Activation, Startup Program

The start-up program is located in the OB 100.

The control bits and the counters are reset in the startup procedure.

## Cyclic Program

The cyclic program is defined in the organization block OB 1.

In the example, the function blocks FB 9 "RECV\_440" and FB 10 "SEND\_440" work together with the functions FC 21 and FC 22 as well as with the data blocks DB 21 and DB 22 as instance DBs and DB 42 and DB 43 as send or receive DBs.

In the example the function blocks are assigned parameters partly via constants and partly via symbolically addressed actual operands.

## Data Transmission

Data transmission takes place from the CP 440 on slot 4 to the CP 440 on slot 5. If you work with other communication partners, the FC 22 (RECEIVE) is not called.

### Description of FC 21 (SEND)

#### The "Generate edge SEND\_REQ" program section:

SEND\_440 is run through once at the start with SEND\_REQ=0. SEND\_REQ is then set to 1. The SEND\_440 job is started when a signal state change from "0" to "1" is detected at the SEND\_REQ control parameter.

When SEND\_DONE=1 or SEND\_ERROR=1, SEND\_REQ is reset to 0.

#### "SEND\_DONE=1" program section:

If the transfer is successful, SEND\_DONE is set to "1" at the parameter output of SEND\_440.

To distinguish between consecutive transfers, a send counter (SEND\_COUNTER\_OK) is included in data word 0 of the source block DB 42.

#### "SEND\_ERROR=1" program section:

If SEND\_440 runs through with SEND\_ERROR=1, the SEND\_COUNTER\_ERR error counter is incremented in data word 2. The SEND\_STATUS is copied, because it is overwritten with 0 in the next run, which means it will not be possible to read it out.

### Description of FC 22 (RECEIVE)

#### The "Enable Receive Data" program section:

Before data can be received, the receive enable (RECV\_EN\_R) at block RECV\_440 must be set to "1".

#### "RECV\_NDR=1" program section:

When RECV\_NDR is set, new data has been received and the RECV\_COUNTER\_OK receive counter is incremented.

#### "RECV\_ERROR=1" program section:

If execution is unsuccessful (i.e. if the error bit is set at the parameter output of RECV\_440), the RECV\_COUNTER\_ERR error counter is incremented. The RECV\_STATUS is copied, because it is overwritten with 0 in the next run and it will thus not be possible to read it out.

All relevant values can be observed for test purposes in the variable table.



## 9.6.2 "CP440 1 CYC" Program Example

This example is set up identically to the example "CP440 SEND RECV". However, the FC 21 has been changed to allow a SEND job to be processed cyclically.

### Description of FC 21 (SEND)

The procedure for processing a SEND job cyclically is as follows:

- At each block call, SEND\_440 is run through with SEND\_REQ=0. If neither SEND\_DONE nor SEND\_ERROR are present, the block is terminated.
- If SEND\_DONE is present, SEND\_440 is run through again with SEND\_REQ=1.
- If SEND\_ERROR is present, SEND\_440 is run through with SEND\_REQ=0 and then with SEND\_REQ=1.

To ensure that SEND\_440 can be run through with SEND\_REQ=1 in the very first pass, the SEND\_STARTUP\_ENDED bit is also evaluated in the program section "Check if SEND\_DONE or SEND\_ERROR or not first Cycle". This bit is at 0 during the first pass and is then fixed at 1.

### See also

"CP440 SEND RECV" Program Example (Page 111)

## 9.6.3 "CP440 ASCII BCC" Program Example

### Structure of the Program Example

This program example only makes sense for the ASCII driver.

This example is set up identically to the "CP440 SEND RECV" example but also contains the following:

- A block check sum is generated in the send section by calling FC 23 (GEN BCC)
- The block check sum is checked in the receive section with the call of FC 24 (CHK BCC)

### Description of the FC 23 (GEN BCC)

The "DB\_NO" input parameter specifies the DB in which the data to be sent is stored. The "LEN" input parameter specifies the length of the data to be sent. The last two bytes should be reserved for the block check (stored as an ASCII value).

The block check is executed by means of an XOR logical operation of all the bytes to be sent (LEN -2). The XOR logical operation occurs in the "LOOP". The relevant byte is loaded with the L DBB [#d\_loop\_akt] command. The "#d\_loop\_akt" offset must be specified as a bit offset. After "LOOP" is executed, the block check is in the low byte of the "#w\_bcc\_value" variable. The block check is then encoded in two ASCII characters as described in the example below.

### Example

Generation of BCC HI ASCII characters "32" =2

```
#w_bcc_value      0025
UW w#16#F0       00F0      //Hide lower half-byte
                  = 0020

SLW 4            0200      //Push a half-byte to the left
OW w#16#3000     3000      //Request 3000 Hex
                  = 3200
```

Generation of BCC LO ASCII characters "35" =5

```
#w_bcc_value      0025
UW w#16#F        000F      //Hide upper half-byte
                  = 0005

OW w#16#30       0030      //Request 30 Hex
                  = 0035
```

The BCC is then written in word 20 of the data block (L DBW [#d\_loop\_akt] )

### Description of the FC 24 (CHK BCC)

The program of the FC24 is largely identical to that of the FC23. At the end, the calculated block check is compared with the block check of the received data. The #RETVAL output parameter is output:

- 0: Block check OK or
- -1: Block check error

### 9.6.4 "CP440 MASTER" Program Example

This example describes the communication between a master and several slaves. You can find the program sections for the individual slaves in the "CP440 SLAVE" program example.

In OB1 the pattern is set for communication with a partner. The branch table in the "Select Slave" program section is used for this. Depending on the value in MW 8, either communication to slave 1 (FC 1 call) or 2 (FC 2 call) is selected. The program can be adapted to include additional slaves.

In FC 1 a send job ("execute new Send-Job" program section with FC 11 call) and the associated receive job ("execute Recv-Job" program section with FC 21 call) are called by the slave for the response. The counter "START\_TIMER" for monitoring the response is started with the send job. If the slave does not respond within the set monitoring time, the timed period expires and there is a switch to the next slave.

### 9.6.5 "CP440 SLAVE" Program Example

In FB 40, RECV is called to query whether a job has been received from the master. For this purpose, in the receive message frame the slave address is compared with its own address. As soon as there is a job, the reply is sent to the master with the SEND call.

The parameters can be adapted for other slaves in OB 1.



## Technical specifications

### 10.1 Technical Specifications of the CP 440

#### General technical specifications

The following table contains the general technical specifications of the CP 440.

You will find additional general technical specifications on the SIMATIC S7-400 in the *S7-400 Automation System, Module Specifications* reference manual, *General Technical Specifications*, and the *S7-400 Automation System, Installation* installation manual.

- Electronic compatibility
- Shipping and storage conditions
- Mechanical and climatic environmental conditions
- Information on insulation testing, safety class and degrees of protection
- Certification

Technical specifications	
Dimensions W x H x D	29 x 290 x 210 mm
Weight	0.3 kg
Current consumption from the backplane bus	Max. 0.36 A at 5 V, typically 0.33 A at 5 V
Power loss	Max. 1.9 W, typically 1.7 W
Display elements	LEDs for transmitting (TXD), receiving (RXD) and interface fault (FAULT) Group alarm LEDs for internal fault (INTF) and external fault (EXTF)
Diagnostic functions	<ul style="list-style-type: none"> <li>• Indicators for internal and external faults</li> <li>• Diagnostic information dump</li> </ul>
Supplied protocol drivers	<ul style="list-style-type: none"> <li>• ASCII driver</li> <li>• 3964(R) procedure</li> </ul>
Transmission rates with 3964(R) protocol	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 bps
Transmission rates with ASCII driver	300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 bps
Character frame	<ul style="list-style-type: none"> <li>• Number of bits per character (7 or 8)</li> <li>• Number of start/stop bits (1 or 2)</li> <li>• Parity (none, even, odd); at 7 bits per character, only "even" or "odd" can be set for parity.</li> </ul>

10.1 Technical Specifications of the CP 440

**Technical specifications X27 (RS 422/485) interface**

The following table contains the technical specifications of the X27 (RS 422/ 485) interface of the CP 440-RS 422/485.

Table 10-1 Technical specifications of the X27 (RS 422/485) interface

<b>Technical specifications</b>	
Interface	RS 422 or RS 485, 15-pin sub D female
RS 422 signals RS 485 signals	TXD (A), RXD (A), TXD (B), RXD (B), GND R/T (A), R/T (B), GND all isolated against the S7-internal power supply (backplane bus) and the external 24 V DC supply
Max. distance	1200 m
Max. baud rate	115200 bps

**Technical specifications of the 3964(R) procedure**

The following table contains the technical specifications of the 3964(R) procedure.

Table 10-2 Technical specifications of the 3964(R) procedure

<b>3964(R) procedure with default values</b>	
Max. message frame length	400 bytes (see chapter "Overview of the Function Blocks (Page 73)")
Parameter	can be assigned parameters: <ul style="list-style-type: none"> <li>• With/without block check character</li> <li>• Priority: low/high</li> <li>• Transmission rate: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 bps</li> <li>• Character frame: 9, 10, 11 or 12 bits</li> <li>• Initial state of receive line: none, R(A)5V/R(B)0V, R(A)0V/R(B)5V</li> <li>• Number of buffered message frames: 1 to 10, use entire buffer</li> </ul>
<b>3964(R) procedure, can be assigned parameters</b>	

<b>3964(R) procedure with default values</b>	
Max. message frame length	400 bytes (see chapter "Overview of the Function Blocks (Page 73)")
Parameter	can be assigned parameters: <ul style="list-style-type: none"> <li>• With/without block check character</li> <li>• Priority: low/high</li> <li>• Transmission rate: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 bps</li> <li>• Character frame: 9, 10, 11 or 12 bits</li> <li>• Character delay time: 20 ms to 65530 ms in 10 ms increments</li> <li>• Acknowledgment delay time: 20 ms to 65530 ms in 10 ms increments</li> <li>• Number of connection attempts: 1 to 255</li> <li>• Number of transmission attempts: 1 to 255</li> <li>• Initial state of receive line: none, R(A)5V/R(B)0V, R(A)0V/R(B)5V</li> </ul>

### Technical specifications ASCII driver

The following table contains the technical specifications of the ASCII driver.

Table 10-3 Technical specifications of the ASCII driver

<b>ASCII driver</b>	
Max. message frame length	400 bytes (see chapter "Overview of the Function Blocks (Page 73)")
Parameter	can be assigned parameters: <ul style="list-style-type: none"> <li>• Transmission rate: 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200 bps</li> <li>• Character frame: 9, 10, 11 or 12 bits</li> <li>• Character delay time: 1 ms to 65530 ms in 1-ms increments</li> <li>• Flow control: none, XON/XOFF</li> <li>• XON/XOFF characters (only with "flow control" = "XON/XOFF")</li> <li>• Wait for XON after XOFF (waiting time for CTS=ON): 20 ms to 65530 ms in 10-ms increments</li> <li>• Number of buffered message frames: 1 to 10, use entire buffer</li> <li>• Disable overwrite: yes/no</li> <li>• Indicator for end of receive message frame:                             <ul style="list-style-type: none"> <li>– After character delay time expires</li> <li>– On receipt of end-of-text character(s)</li> <li>– On receipt of a fixed number of characters</li> </ul> </li> </ul>
<b>ASCII driver with end of message frame recognition after character delay time expires</b>	
Parameter	No further parameter assignment necessary. End of message frame is recognized when the configured character delay time expires.
<b>ASCII driver with end of message frame recognition by assigned parameters end character</b>	

<b>ASCII driver</b>	
Parameter	can also be assigned parameters: <ul style="list-style-type: none"><li>• Number of end-of-text characters: 1 and/or 2</li><li>• Hex code for first/second end-of-text character</li></ul>
<b>ASCII driver with end of message frame recognition by configured message-frame length</b>	
Parameter	can also be assigned parameters: <ul style="list-style-type: none"><li>• Message frame length: 1 to 400 bytes (see chapter "Overview of the Function Blocks (Page 73)")</li></ul>

**See also**

Technical Specifications of the Function Blocks (Page 89)

## 10.2 Standards and approvals

### 10.2.1 Currently valid markings and approvals

#### Introduction

This section contains:

- The standards and test values to which the coupling module CP 440 for point-to-point connections complies and which it fulfills.
- The test criteria according to which the CP 440 was tested.

#### Technical specifications of the coupling module CP 440 for point-to-point connections

You can find the technical specifications of the coupling module CP 440 for point-to-point connections in the equipment documentation. If there are deviations between the statements in this document and the equipment documentation, the statements in the equipment documentation take priority.

#### Validity of the information on the components


<b>NOTICE</b>
<b>Markings and approvals</b> In the documentation, you can find the markings and approvals which are generally possible or planned in the system. However, only the marking or approval printed on the components of the automation system is valid.



## Reference

You can find the certificates for the markings and approvals on the Internet under Service&Support.

## Safety instructions

 <b>WARNING</b>
<b>Personal injury and damage to property may occur</b>
Note the following information for use in hazardous areas:
<ul style="list-style-type: none"><li>• Personal injury and material damage can occur if you establish or disconnect an electrical circuit while the automation system is in operation.</li><li>• Always ensure to de-energize the system before removing the plug connections.</li></ul>

---

### Note

The coupling module CP 440 for point-to-point connections is intended for use in industrial areas. If used in residential areas, it may interfere with radio/TV reception.

---

## 10.2.2 CE approval

### Introduction



The coupling module CP 440 for point-to-point connections fulfills the requirements and protective aims of the following EC directives and conforms with the harmonized European standards (EN) that have been published for programmable logic controllers in the Official Journals of the European Community:

- Low Voltage Directive
- EMC Directive
- Explosion Protection Directive

The EC Declaration of Conformity can be downloaded from the Internet.

### Low Voltage Directive

2014/35/EU "Electrical Equipment Designed for Use within Certain Voltage Limits" (Low Voltage Directive)

The coupling modules CP 440 for point-to-point connections are tested according to the requirements of EN 61131-2, which fall under the Low Voltage Directive.

### EMC Directive

2014/30/EU "Electromagnetic Compatibility" (EMC Directive)

#### Use in industry

SIMATIC products are designed for industrial applications.

Area of application	Interference requirements	Immunity requirements
Industry	EN 61000-6-4 + A1	EN 61000-6-2

### Adhere to the installation guidelines

SIMATIC products meet the requirements if you comply with the installation guidelines described in the manuals during installation and operation.

#### 10.2.3

#### CCC approval



##### Certificate:

2020322309002744

Ex nA IIC T4 Gc

##### According to the following standards:

- GB 3836.1-2010 (Explosive atmospheres - Part 1: Equipment - General requirements)
- GB 3836.8-2014 (Explosive atmospheres-Part 8: Equipment protection by type of protection "n")

#### 10.2.4

#### UKCA approval



DEKRA 21UKEX0018 X

Importer UK:


Siemens plc

Manchester M20 2UR

## 10.2.5 Explosion protection

### ATEX approval DEKRA 21ATEX0010 X



Type examination certificate number	DEKRA 21ATEX0010 X	
Standards	EN IEC 60079-0	
	EN 60079-7	
Marking		II 3 G Ex ec IIC T4 Gc
The certificate is valid for the "DEKRA 21ATEX0010 X" products listed in the certificate.		

#### Special conditions

- The device may only be used in areas with a pollution degree of no more than 2 according to EN 60664-1.
- The modules must be installed in a suitable enclosure which provides a degree of protection of at least IP54 in accordance with EN 60079-7, taking into account the ambient conditions of use.
- Measures must be taken to prevent exceeding the rated voltage by more than 119 V of transient disturbance voltages.

### IECEx approval IECEx DEK 21.0008X

Certificate number	IECEx DEK 21.0008X	
Standards	IEC 60079-0	
	IEC 60079-7	
Marking	Ex ec IIC T4 Gc	
The certificate is valid for the "IECEx DEK 21.0008X". products listed in the certificate.		

#### Special conditions

- The device may only be used in areas with a pollution degree of no more than 2 according to IEC 60664-1.
- The modules must be installed in a suitable enclosure which provides at least IP54 degree of protection in accordance with IEC 60079-7 and taking into account the ambient conditions of use.
- Measures must be taken to prevent exceeding the rated voltage by more than 119 V of transient disturbance voltages.

### 10.2.6 cULus approval



Underwriters Laboratories Inc. in accordance with:

- UL 508 (Industrial Control Equipment)
- CSA C 22.2 No. 142 (Process Control Equipment)

### 10.2.7 cULus HAZ. LOC. approval



**Underwriters Laboratories Inc. in accordance with:**

- UL 61010-2-201 (Industrial Control Equipment)
- CSA / CAN 61010-2-201 (Process Control Equipment)
- ANSI/ISA 12.12.01
- CSA C22.2 No. 213 (Hazardous Location)

Approved for use in:

Class I, Division 2, Group A, B, C, D Tx;

Class I, Zone 2, Group IIC Tx

### 10.2.8 cFMus approval



Factory Mutual Research (FM) in accordance with:

- Approval Standards FM Class 3600:2018, FM Class 3611:2018, FM Class 3810:2018,
- ANSI/UL 121201.2017,
- ANSI/UL 61010-1:2018

APPROVED for use in:

- Class I, Division 2, Group A, B, C and D
- Class I, Zone 2, Group IIC, hazardous (classified) locations
- Ordinary (unclassified) locations with an ambient temperature rating of 0 °C to + 70 °C, indoor environments

### 10.2.9 Approval for Australia and New Zealand



The SIMATIC S7-400 product series meets the requirements of the standard EN 61000-6-4:2007 + A1:2001.

### 10.2.10 Approval for the Korea and South Korea



The SIMATIC S7-400 product series complies with Korean safety standards:  
Registration of Broadcasting and Communication Equipment KCC-REM-S49-S7400

Note that this device corresponds to limit class A in terms of the emission of radio frequency interference. This device can be used in all areas, except residential areas.

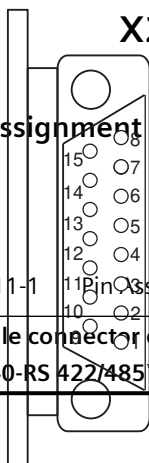
이 기기는 업무용(A급) 전자파 적합기기로서 판매자 또는 사용자는 이 점을 주의하시기 바라며 가정 외의 지역에서 사용하는 것을 목적으로 합니다.



## Connecting cables

### 11.1 X27 (RS 422/485) Interface of the CP 440

#### Pin Assignment



The table below shows the pin assignment for the 15-pin subminiature D female connector in the front panel of the CP 440.

Table 11-1 Pin Assignment for the 15-Pin Subminiature D Female Connector of the Integrated Interface of the CP 440

Female connector on CP 440-RS 422/485	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 (isolated)
	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	-	-	-

\* View from the front

### 11.2 Cables

#### Connecting cables

If you make 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 housings. A large surface area of both sides of the cable shield must be in contact with the connector casing. You are advised to use Siemens V42 254 shielded connector casings.



**CAUTION**

Never connect the cable shield with the GND, as this could destroy the submodules. GND must always be connected on both sides (pin 8), otherwise the submodules could again be destroyed.

**In the Following**

The following pages contain examples of connecting cables for a point-to-point connection between the CP 440 and S7 modules or the SIMATIC S5.

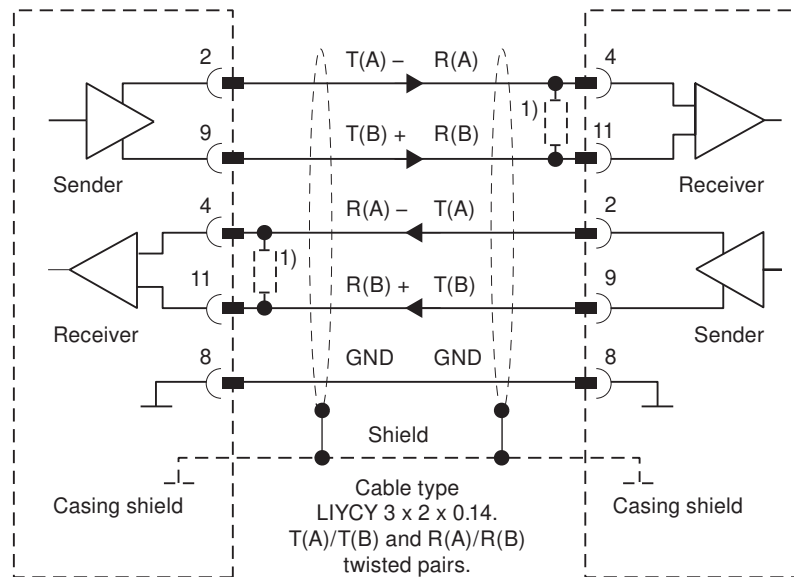
**Connecting cable X 27 (S7 (CP 440) - CP 340/CP 341/CP 440/CP 441)**

The figure below illustrates the cable for a point-to-point connection between a CP 440 and a CP 340/CP 341/CP 440/CP 441 for RS 422 operation.



For the connecting cables you require the following male connectors:

- At the CP 440 end: 15-pin sub D male connector with screw-locking
- At the communication partner: 15-pin sub D male connector with screw-locking



1) In the case of cables longer than 50 m you must solder in a terminating resistor of approx. 330  $\Omega$  on the receiver for trouble-free traffic.

Figure 11-1 X27 Cable for CP 440 - CP 340/CP 341/CP 440/CP 441 for RS 422 Operation (Four-Wire)

#### Note

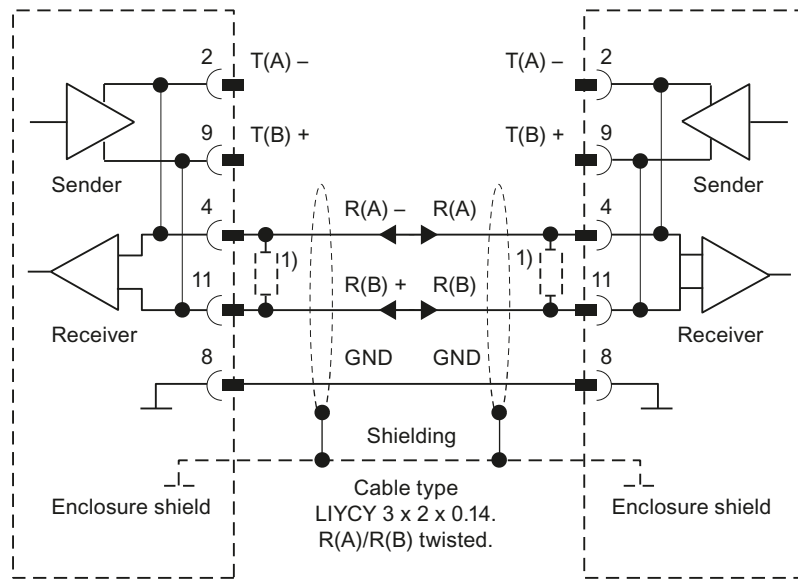
This cable type can be used in the following lengths for the CP 440 as communication partner: max. 1200 m at 19 200 Baud, max. 500 m at 38 400 Baud max., max. 200 m at 115 200 Baud

### Connecting cable X 27 (S7 (CP 440) - CP 340/CP 341/CP 440/CP 441)

The figure below illustrates the cable for a point-to-point connection between a CP 440 and a CP 340/CP 341/CP 440/CP 441 for RS 485 operation.

For the connecting cables you require the following male connectors:

- At the CP 440 end: 15-pin sub D male connector with screw-locking
- At the communication partner: 15-pin sub D male connector with screw-locking



1) To ensure interference-free data exchange with line lengths > 50 m you must solder in a terminating resistance of approx. 330 Ω on the receiver side.

Figure 11-2 X27 Cable for CP 440 - CP 340/CP 341/CP 440/CP 441 for RS 485 Operation (Two-Wire)

**Note**

The previous figure shows the wiring if you want to make the connecting cable yourself. In both RS 485 mode (two wire) and RS 422 mode (four wire) you can also use Siemens connecting cables. The figure below illustrates the internal wiring in the connecting cable.

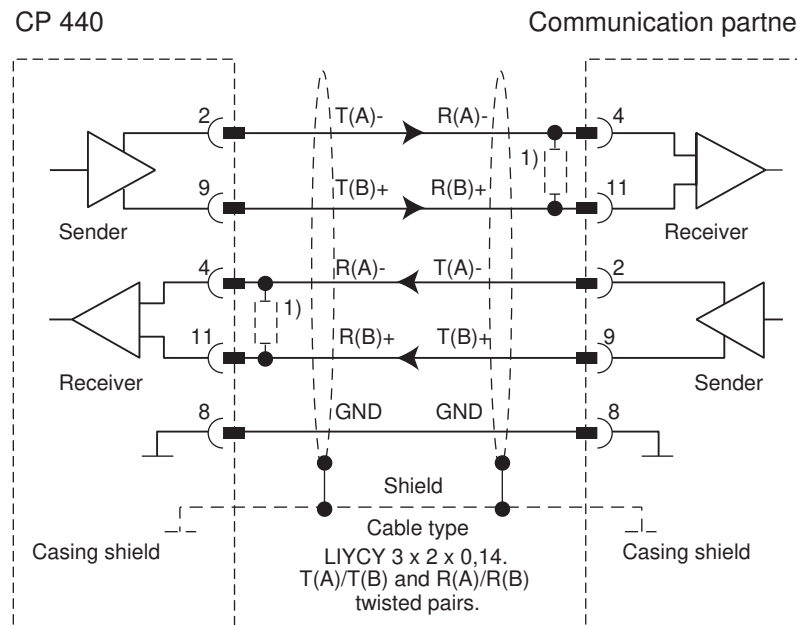
The jumpers 2-4 and 9-11 are "installed" by parameter assignment of the CP.

**Connecting cable X 27 (S7 (CP 440) - CP 544, CP 524, CPU 928B, CPU 945, CPU 948)**

The figure below illustrates the cable for a point-to-point connection between a CP 440 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 the CP 440 end: 15-pin subminiature D male with screw interlock
- At the communication partner: 15-pin subminiature D male with clip fixing



1) In the case of cables longer than 50 m you must solder in a terminating resistor of 330  $\Omega$  on the recipient side.

Figure 11-3 X27 Cable CP 440 - CP 544, CP 524, CPU 928B, CPU 945, CPU 948 for RS 422 Operation (Four-Wire Mode)



## Accessories and order numbers

### 12.1 Accessories and Order Numbers

#### Module

Table 12-1 Order Numbers of the CP 440 Module

Product	Order Number
CP 440	6ES7 440-1CS00-0YE0

#### Cables

Cables are available of the preferential lengths: 5 m, 10 m and 50 m.

Table 12-2 Order Numbers of the Cables

Connecting cables for	Specification	Order Number
CP 440 - CP 340		
CP 440 - CP 341		
CP 440 - CP 440		
CP 440 - CP 441		
X27 (RS 422) interface	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



## Literature

### 13.1 Literature on SIMATIC S7

#### Literature on SIMATIC S7

The following page gives you an overview of the manuals that you need to configure and program the S7-400.

#### Manuals for configuring and commissioning

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. The following table also provides you with an overview of the documentation to **STEP 7**.

Table 13-1 Manuals for Configuring and Programming the S7-400

Title	Contents
Manual Programming with STEP 7 ( <a href="http://support.automation.siemens.com/WW/view/en/18652056">http://support.automation.siemens.com/WW/view/en/18652056</a> )	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.
Manual Configuring Hardware and Communication Connections with STEP 7 ( <a href="http://support.automation.siemens.com/WW/view/en/18652631">http://support.automation.siemens.com/WW/view/en/18652631</a> )	This STEP 7 manual explains the principles behind the use and functions of the STEP 7 automation software. It will provide both first-time users of STEP 7 and those with knowledge of STEP 5 with an overview of the procedures for configuring, programming and starting up an S7-300/400. When working in the software, users can access the relevant sections of the online help where they will find specific support for its application.

## 13.1 Literature on SIMATIC S7

Title	Contents
Reference Manual Instruction list (IL) for S7-300/400 ( <a href="http://support.automation.siemens.com/WW/view/en/18653496">http://support.automation.siemens.com/WW/view/en/18653496</a> )	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.
Reference Manual Ladder Diagram (LAD) for S7-300/400 ( <a href="http://support.automation.siemens.com/WW/view/en/18654395">http://support.automation.siemens.com/WW/view/en/18654395</a> )	When working with the software, you can use the on-line help, which provides you with detailed information on editors and compilers.
Reference Manual Function block diagram (FBD) for S7-300/400 ( <a href="http://support.automation.siemens.com/WW/view/en/18652644">http://support.automation.siemens.com/WW/view/en/18652644</a> )	
Reference Manual S7-SCL for S7-300/400 ( <a href="http://support.automation.siemens.com/WW/view/en/5581793">http://support.automation.siemens.com/WW/view/en/5581793</a> ) <sup>1</sup>	
Manual S7–GRAPH for S7-300/400 Programming Sequential Control Systems ( <a href="http://support.automation.siemens.com/WW/view/en/1137630">http://support.automation.siemens.com/WW/view/en/1137630</a> ) <sup>1</sup>	
Manual Programming S7–HiGraph State Graphs ( <a href="http://support.automation.siemens.com/WW/view/en/1137299">http://support.automation.siemens.com/WW/view/en/1137299</a> ) <sup>1</sup>	
Manuals CFC for SIMATIC S7 ( <a href="http://support.automation.siemens.com/WW/view/en/15236182">http://support.automation.siemens.com/WW/view/en/15236182</a> ) <sup>1</sup>	
Reference Manual System and Standard Functions for S7-300/400 ( <a href="http://support.automation.siemens.com/WW/view/en/1214574">http://support.automation.siemens.com/WW/view/en/1214574</a> )	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.
<sup>1</sup> Add-on packages for S7–300/400 system software	



# Index

## 3

- 3964 procedure
  - Receive buffer, 45
- 3964 procedure programmable, 64
- 3964 procedure with default values, 64
- 3964(R) procedure
  - Parameter, 63
  - Technical specifications, 118
- 3964(R) Procedure, 35
  - Initialization conflict, 47
  - Priority, 36
  - Procedure errors, 47
- 3964(R) procedure startup, 35

## A

- Absolutely addressed actual parameter, 87
- Acknowledgment delay time (ADT), 65
- Actual operand
  - absolutely addressed, 87
  - symbolically addressed, 87
- Addressing the module, 89
- ASCII driver, 23
  - Parameter, 58
  - Receive buffer, 31
  - receiving data, 26
  - Sending Data, 24
  - Technical specifications, 119

## B

- Basic parameters, 57
- Baud rate
  - 3964 procedure, 66
  - ASCII driver, 60
- Bidirectional data traffic, 17
- Buffered receive message frames, 61, 67

## C

- Cable shield, 128
- Cables, 49
  - Order Numbers, 133
- Certification, 117
- Character delay time, 24, 26, 59, 65

- Character delay time (CDT), 19
- Character frame, 18, 60, 66
- Code Transparency, 26
- Commissioning the physical interface, 50
- Communication protocols
  - Parameter assignment, 56
- Communication via Function Blocks, 73
- Configuration data, 57
  - 3964(R) procedure, 63
  - ASCII driver, 58
- Connecting cables, 127
- Connection attempts, 65
- Control and display elements, 15
- CP 440
  - Applications, 11
  - Diagnostic buffer, 95, 104
  - Diagnostic functions, 95
  - functionality, 11
  - installing, 53
  - Operating modes, 91
  - removing, 53
  - Setup, 14
  - Startup behavior, 91
  - Technical specifications, 117
- CPU RUN, 92
- CPU Startup, 92
- CPU STOP, 92
- CPU versions, 13

## D

- Data bits, 60, 66
- Data Consistency, 76
- Data flow control, 61
- Data flow control/Handshaking, 32
- Data management, 68
- Default setting, 62
- Default settings, 33, 50
- Diagnostic buffer, 95, 104
- Diagnostic functions, 95, 117
- Diagnostic messages, 97
- Diagnostics
  - Diagnostic buffer, 104
  - Display elements, 96
- Dimensions, 117
- Direct parameter assignment, 85
  - Example, 86
- Disabling alarms, 88
- Display elements, 96

Display elements (LEDs), 15, 95

## E

EN/ENO Mechanism, 88  
End criterion, 24, 26  
    End-of-text character, 27  
    Expiration of the character delay time, 26  
    Fixed message frame length, 29  
End-of-text character, 24, 59  
Error messages of the display elements, 96  
Event class, 97  
Event Class 30, 103  
    errors in communication between the CP and the CPU, 103  
Event number, 97  
EXTF, 96

## F

FAULT, 96  
FB 10, 74  
FB 11, 74  
FB 9, 74  
FB RES\_RECV  
    Assignment in the Data Area, 84  
    Deleting the receive buffer, 82  
    Error display, 83  
    Parameters, 84  
    Time sequence chart, 85  
Fixed message frame length, 24, 29  
Four-wire mode, 62  
Full duplex, 62  
Full-duplex operation, 17  
Function blocks, 73  
    Application, 75  
    Diagnostic messages, 97  
    Overview, 73  
    Technical specifications, 89  
Functionality of the CP 440, 11  
Functionality of the CP 440, 11

## G

Group alarm LED, 96

## H

Half duplex, 62  
Half-duplex operation, 17

Hardware Components, 12

## I

Indicator for end of receive message frame, 59  
Indirect parameter assignment, 85  
    Example, 86  
Initial state of receive line, 62, 67  
Initialization, 91  
Initialization conflict, 47  
Installation, 56  
Instance data block, 76, 78, 81, 84  
Interface  
    X27 (RS 422/485), 127  
Interface (physical), 50  
Interface fault LED, 96  
Interfaces, 12  
Interrupt OBs, 73  
INTF, 96  
ISO 7-Layer Reference Model, 20

## J

Jobs which can be processed simultaneously, 76

## L

LADDR, 89  
LEDs, 15, 71  
Literature, 135  
Load firmware, 70  
Loading the configuration and parameters, 68

## M

Master, 33  
Memory requirements, 89  
Message frame length, 59  
Minimum Number of CPU Cycles, 89  
Monitoring time for missing end ID, 59  
Mounting, 53  
Mounting the CP 440, 53  
Multicomputing operation, 73  
Multipoint, 32

## O

OB 122  
    I/O access error, 53  
OB 83 (insert/remove interrupt), 53

Operating mode transitions, 92  
 Order number, 55  
 Order Number, 133

## P

Parameter assignment, 56, 92  
   direct, 85  
   indirect, 85  
   of data words, 86  
 Parameter assignment interface, 13  
 Parameter reassignment, 91  
 Parity, 60, 66  
 Pin Assignment, 127  
 Point-to-point, 32  
 Polarity, 50  
 Power input, 117  
 Power loss, 117  
 Prevent overwrite, 61  
 Priority, 36, 66  
 Procedure, 20  
 Procedure errors, 47  
 Program processing, 88  
 Programming device (PG), 12  
 Programming device cable, 12  
 Protocol, 64  
 Protocol parameters, 59, 65

## R

Reading the diagnostic buffer, 104  
 Receive buffer, 31, 45, 61, 67  
 receiving data  
   ASCII driver, 26  
 Receiving with the 3964(R) procedure, 43  
 RECV\_440, 74  
 RECV\_440 FB  
   Assignment in the Data Area, 81  
   Error display, 80  
   Parameters, 81  
   Time sequence chart, 82  
 Removing the CP 440, 53  
 RES\_RECV, 74  
 RS 422, 62  
 RS 422/485, 16  
 RS 485, 62  
 RS422 operation, 24  
 RS485 operation, 24  
 RUN, 91

## S

SEND\_440, 74  
 SEND\_440 FB, 76  
   Assignment in the Data Area, 78  
   Error display, 77  
   Parameters, 78  
   Time sequence chart, 79  
 Sender line drivers, 93  
 Sending Data  
   ASCII driver, 24  
 Sending with the 3964(R) procedure, 40  
 SFC 41, 73  
 SFCERR, 103  
 SFCERR variable, 103  
 Slave, 33  
 Slots, 53  
 Software components, 13  
 Special displays, 96  
   RXD, 95  
   TXD, 95  
 Special features  
   when sending message frames, 92  
 Standard cable, 12  
 Start bit, 60, 66  
 Startup behavior  
   CP 440 FBs, 88  
   programmable logic controller (CP 440), 88  
 STATUS on the FB, 75  
 STATUS output, 103  
 STATUS outputs of the FBs, 95  
 STEP 7, 55  
 Steps, 49  
 STOP, 91  
 Stop bits, 60, 66  
 Switchover time, 24  
 Symbolically addressed actual parameter, 87

## T

Technical specifications, 117  
 Time OBs, 73  
 Topologies, 32  
 Transmission attempts, 65  
 Transmission integrity, 21  
   with ASCII driver, 22  
 Transmission pause, 60  
 Two-wire mode, 62

## U

Uses of the CP 440, 11

## W

Weight, 117

## X

X27 (RS 422/485) interface, 127

    Technical specifications, 118

X27 interface

    Definition, 16

    Properties, 16

XOFF code, 61

XON code, 61

XON/XOFF, 61