SIEMENS

SIMATIC

CP 444
Communication Processor

Installation and Parameter Assignment

Manual

This manual has the order number:
6ES7444-2AA00-8BA0

Edition 09/2001
A5E00133845-01

Safety Guidelines

This manual contains notices intended to ensure personal safety, as well as to protect the products and connected equipment against damage. These notices are highlighted by the symbols shown below and graded according to severity by the following texts:

Danger

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

Warning

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

Caution

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

Caution

indicates that property damage can result if proper precautions are not taken.

Notice

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

Qualified Personnel

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

Correct Usage

Note the following:

Warning

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

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

Trademarks

SIMATIC®, SIMATIC HMI® and SIMATIC NET® are registered trademarks of SIEMENS AG.
Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

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

Disclaim of Liability

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

Preface

Purpose of the Manual

The information provided in this manual tells you how to connect the S7-400 programmable controller to the SIMATIC Industrial Ethernet cellular and area network.

Contents of the Manual

This manual describes the hardware and software of the CP 444 communication processor and how it is integrated in an S7-400 programmable controller. It is divided into a main section and a reference section (appendices).

The following subjects are covered:

- The fundamentals of the MAP 3.0 protocol profile
- Starting up the CP 444
- Mounting and wiring the CP 444
- Parameterizing the CP 444
- Programming the communication
- Troubleshooting
- Attributes and technical specifications

This manual builds on the information in the installation manual *S7-400/M7-400 Programmable Controllers, Hardware and Installation*. You must read and comply with the information in the installation manual in order to run the CP 444 communication processor in an S7-400 programmable controller.

Scope of the Manual

This manual is relevant to the following module and software:

Module/Software	Order Number	As of Release/Version
CP 444 communication processor	6ES7 444-1MX00-0XE0	1
CP 444 configuration package consisting of: • The *CP 444 Communication Processor, Installation and Parameter Assignment* manual • A floppy disk with: – The *Configuration for CP 444* parameterization interface – A library containing function blocks – A programming example	6ES7 444-1MX00-7□G0 German A English B	1

This manual describes the CP 444 communication processor valid at the time of its publication. We reserve the right to describe changes to the module's functionality in a product information sheet.

Target Group

This manual is aimed at readers who want to plan, set up or put into operation a connection between an S7-400 programmable controller and SIMATIC Industrial Ethernet. We assume that you already have experience with and know how to use SIMATIC Industrial Ethernet and the MAP 3.0 protocol profile.

To parameterize the CP 444 and program the functional blocks, you must know how to use *STEP 7*.

Easy Access to Information

To help you find the information you require quickly and easily, the manual offers the following:

- A comprehensive table of contents followed by lists of all figures and tables which appear in the manual.
- In the main body of the text, the information in the left-hand column of each page summarizes the contents of each section.
- Finally, a comprehensive index allows quick access to information on specific subjects.

Other Manuals Required

Appendix D contains a list of other manuals and brochures on the subject of S7-400 and programmable controllers.

Electronic Manuals

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

Standards, Certificates and Approvals

The CP 444 fulfills the requirements and criteria of IEC 1131, Part 2, and the requirements for the CE marking. It has both CSA and UL certification.

You will find information on the relevant certificates, approvals and standards in Appendix A.2.

Recycling and Disposal

The CP 444 is an environment-friendly product. Its features include the following:

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

The low-contaminant materials used mean that the CP 444 can be recycled.

To recycle and dispose your old equipment in a manner that is not damaging to the environment, contact:

Siemens Aktiengesellschaft
Anlagenbau und Technische Dienstleistungen
ATD ERC Essen Recycling/Remarketing
Fronhauser Str. 69
D-45 127 Essen

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

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

Additional Support

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

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

We offer a range of courses to help get you started with the SIMATIC S7 programmable controller. Please contact your local training center or the central training center in Nuremberg, D-90027 Germany,
Tel. +49 911 895 3154.

Up-to-Date Information at All Times

Continually updated information is available on the internet at *http://www.ad.siemens.de*.

In addition, SIMATIC Customer Support provides you with current information and downloads that can be useful to you when using SIMATIC products:

- On the Internet at http://www.ad.siemens.de/simatic-cs
- From the SIMATIC Customer Support mailbox (German) on + 49 (911) 895-7100 or from the SIMATIC Customer Support BBS (English)

  To call the mailbox, use a modem with up to V.34 (28.8 Kbps) and set its parameters as follows: 8, N, 1, ANSI. Alternatively, use ISDN (x.75, 64 Kbps).

You can contact SIMATIC Customer Support by phone on + 49 (911) 895-7000 and by fax on + 49 (911) 895-7002. You can also send inquiries by e-mail on the Internet or to the above-mentioned mailbox.

Contents

Product Overview

1

Purpose of This Chapter

After you have read this chapter, you will know the possible uses and the design of the CP 444.

Chapter Overview

This chapter is divided into the following sections:

In Section	You Will Find	on Page
1.1	Uses of the CP 444	1-2
1.2	Components Required to Use the CP 444	1-3
1.3	Design of the CP 444	1-5

1.1 Uses of the CP 444

Introduction

The CP 444 communication processor allows you to connect an S7-400 programmable controller to the SIMATIC Industrial Ethernet cellular and area network.

The CP 444 provides an S7-400 programmable controller with communication access using MMS (Manufacturing Message Specification /1/) services) in accordance with the MAP 3.0 communication standard/2/.

Functionality of the CP 444

The CP 444 communication processor offers you the following functionality:

- An interface for connecting to SIMATIC Industrial Ethernet in accordance with the IEEE 802.3 Ethernet standard
- Communication via the interface for open interconnection by means of MMS services for device monitoring purposes (VMD services) and language-neutral data transfer (variable services)
- Transmission rate via the interface: 10 Mbps in accordance with IEEE 802.3
- Power supply and communication with the S7-400 programmable controller via the S7-400 backplane bus
- Module configuration with the *STEP 7* base software of SIMATIC S7
- Configuration of the communication link (associations, variables) with the *Configuration for CP 444* parameterization interface

Uses of the CP 444

The MAP 3.0 protocol profile, which is often used in production, is implemented in the CP 444. MAP 3.0 allows ”open communication” between programmable controllers of different types (e.g. from different vendors).

This makes it possible, for example, to integrate Siemens subnets in networks with the Manufacturing Automation Protocol (MAP) architecture.

/1/ ISO/IEC 9506-4, Industrial automation systems - Manufacturing Message Specification - Part 4
/2/ MAP 3.0 1988; Manufacturing Automation Protocol, Version 3.0

1.2 Components Required to Use the CP 444

Introduction

To provide an S7-400 programmable controller with MAP communication access using the CP 444 communication processor, you require certain hardware and software components.

Hardware Components

The following table lists the hardware components required for a MAP communication connection with the CP 444.

Table 1-1 Hardware Components for a MAP Communication Connection

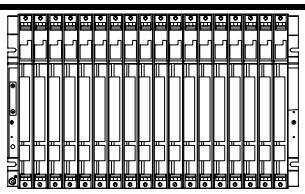
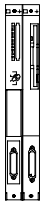
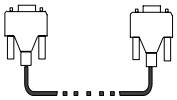
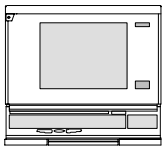
Components	Function	Diagram
Mounting rack	... provides the mechanical and electrical connections between S7-400 modules.	
Power supply module (PS) Accessories: backup battery	... converts the line voltage (120/230 V AC and 24 V DC) into the operating voltage of 24 V and 5 V DC 24 required to supply the S7-400.	
CPU Accessories: memory card	... executes the user program; communicates via the MPI interface with other CPUs or with a programming device.	
CP 444 communication processor	... has an interface; provides the S7-400 with MAP communication access.	
Transceiver connecting cable	... connects the CP 444 communication processor with SIMATIC Industrial Ethernet (transceiver).	

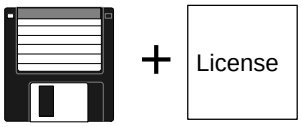
Table 1-1 Hardware Components for a MAP Communication Connection, continued

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

Software Components

The following table lists the software components required for a MAP communication connection with the CP 444.

Table 1-2 Software Components for a MAP Communication Connection

Components	Function	Diagram
STEP 7 software package	... configures, parameterizes, programs and tests the S7-400.	
The *Configuration for CP 444* parameterization interface	... parameterizes the MAP communication connection (associations, variables).	
Function blocks (FBs)	... control data transfer and the variable services.	
Programming example	... indicates how communication with the CP 444 can be programmed using MAP 3.0.	

1.3 Design of the CP 444

Introduction

The CP 444 communication processor has an interface for connection to SIMATIC Industrial Ethernet. The display elements on the front of the module provide information on the operating state of the CP 444.

Arrangement of the Control and Display Elements

Figure 1-1 shows the arrangement of the control and display elements on the front panel of the CP 444 communication processor.

The elements labeled in gray in Figure 1-1 are not used for the Ethernet connection of the CP 444.

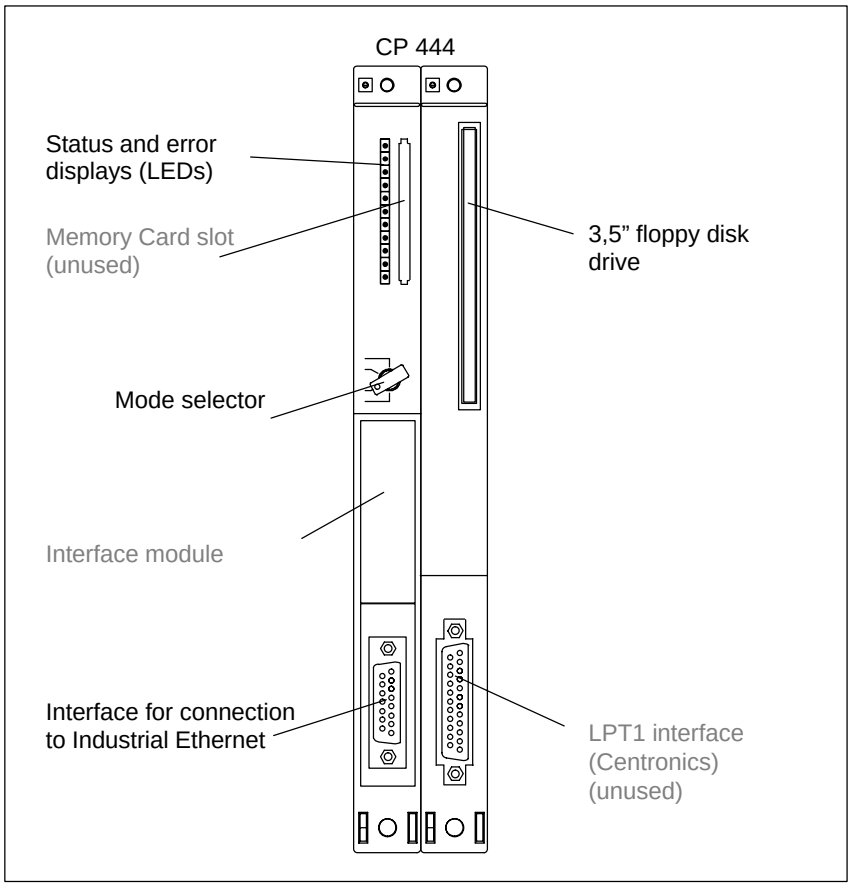


Figure 1-1 Arrangement of the Control and Display Elements for the CP 444 processor

Mode Selector

With the CP 444 mode selector you can create a hardware reset via MRES. How to perfrom a CP 444 hardware reset is described in section 10.3. In normal mode the mode selector must be in the RUN-P position.

LED Displays

The following LED displays are on the front panel of the CP 444:

- **INTF** (red) Indication of an internal fault
- **EXTF** (red) Indication of an external fault
- **SD** (green) (unused)
- **HD** (green) The hard disk in the CP 444 is being accessed
- **MRDY** (yellow) The CP 444 is currently providing an MMS service
- **MERR** (yellow) The CP 444 is requesting an MMS service, or an error has occurred during execution of an MMS service
- **RUN** (green) The CP 444 is running
- **STOP** (yellow) Access to the CP 444 is blocked

The operating modes and errors indicated by these LEDs are described in detail in Section 11.2.

Interface for Ethernet Connection

The 15-pin subminiature D female connector is a communication interface that complies with the IEEE 802.3 Ethernet standard. You connect the transceiver connecting cable for an SIMATIC Industrial Ethernet connection to this interface. You will find the pin assignment of the interface in Appendix A.1.

Note

The 15-pin subminiature D female connector of the CP 444 does not provide power (+15 V) for a transceiver.

Floppy Disk Drive

You use the floppy disk drive (with a 3.5” floppy disk inserted) to record an MMS trace log. This log data provides a step-by-step record of the connection setup process, for example, and enables connection problems to be localized. Error/fault analysis using the MMS trace is described in Chapter 11.4.

Module Slots, LPT1 Interface

The module slot for a memory card, the module slot for an interface module and the LPT1 interface are not required for when the CP 444 is used to provide MAP communication access to a S7-400 and therefore remain unused (labeled in gray in Figure 1-1). The module slot for an interface module can be used for a VGA module.

Base Connector for the S7 Backplane Bus

On the back of the CP 444 there is a base connector for connection to the S7-400 backplane bus. The S7-400 backplane bus supplies the CP 444 with the necessary voltage and is the serial data bus via which the CP 444 communicates with the modules of the programmable controller.

Fundamentals of the MAP 3.0 Protocol Profile

2

Purpose of This Chapter

Once you have read this chapter, you will have a grasp of the essentials as regards the MAP 3.0 protocol profile for the use of the CP 444. The terms ”MMS object”, ”MMS service” and ”virtual programmable controller (VMD)” are briefly introduced.

If you are already familiar with the MAP 3.0 protocol profile, you need not read this chapter.

MAP 3.0

The CP 444 communication processor is an interface module that provides an S7-400 programmable controller with access to the SIMATIC Industrial Ethernet communication network by means of the MAP 3.0 protocol profile.

Chapter Overview

This chapter is divided into the following sections:

2.1 MAP 3.0 Protocol Profile

Introduction

Standardized procedures and protocols are indispensable if the devices and systems of different vendors are to be networked at a reasonable price in manufacturing environments. The MAP 3.0 (Manufacturing Automation Protocol) protocol profile has been developed for open industrial communication.

MAP 3.0

The MAP 3.0 protocol profile is based on the ISO/OSI (International Standard Organisation/Open System Interconnection) reference model of open communication. MAP 3.0 describes the communication protocols of all 7 layers of the ISO/OSI reference model.

ISO/OSI Reference Model

The seven layers of the ISO/OSI reference model are subdivided into transport protocols (layers 1 to 4) and application protocols (layers 5 to 7).

Table 2-1 The Seven Layers of the ISO/OSI Reference Model

	Layer	Designation	Function
Application protocols	7	Application layer	Information processing
	6	Presentation layer	Abstract data representation, encoding
	5	Session layer	Dialog control, restart procedures
Transport protocols	4	Transport layer	Data transmission independently of the network type
	3	Network layer	Setup of the transmission path, optimal path selection
	2	Data-link layer	Reliable transmission between network nodes with error control
	1	Physical layer	Maintenance of the physical connection

MAP 3.0 Protocols

Open communication between devices of different types is only possible when the protocol profile implements all 7 layers of the ISO/OSI reference model.

The MAP 3.0 protocol profile on the CP 444 therefore comprises the functionality of all 7 layers. MAP 3.0 uses the Ethernet bus with the CSMA/CD access method based on IEEE 802.3 on layers 1 and 2 and the MMS communication standard on the application layer (layer 7).

Table 2-2 Protocol Profile of the CP 444 Communication Processor

Layer	Protocol/Standard
7b	MMS ISO 9506
7a	Application protocol based on ISO 8650/2 ACSE
6	Presentation protocol based on ISO 8823 kernel and ASN.1 ISO 8824/25
5	Session protocol based on ISO 8327 kernel (full duplex)
4	Transport protocol based on ISO 8073 class 4
3	Internet protocol based on ISO 8473 and ES/IS ISO 9542
2	LLC protocol based on ISO 8802-2 (IEE 802.2)
1	Ethernet bus with CSMA/CD based on ISO 8802-3 (IEEE 802.3)

MMS (Manufacturing Message Specification) = communication standard for information processing

MMS

MMS (Manufacturing Message Specification) represents the protocol of the application layer 7b and is the international standard ISO 9506.

MMS enables the implementation of communication between different application systems and a programmable controller. MMS permits the specific properties of end devices (e.g. personal computers, programmable logic control systems and production control computers) to be covered by uniform, standardized system and data representation.

2.2 MMS Models, Objects and Services

MMS Model

MMS provides you with standardized mechanisms that allow programmable controllers to be addressed by means of abstract models.

MMS provides objects (MMS objects) that represent the various automation components required to model a programmable controller. These objects are, in turn, defined by attributes. The objects are assigned specific services (MMS services) by means of which they can be manipulated.

The standardization of services, objects, attributes, parameters and statuses ensures open communication.

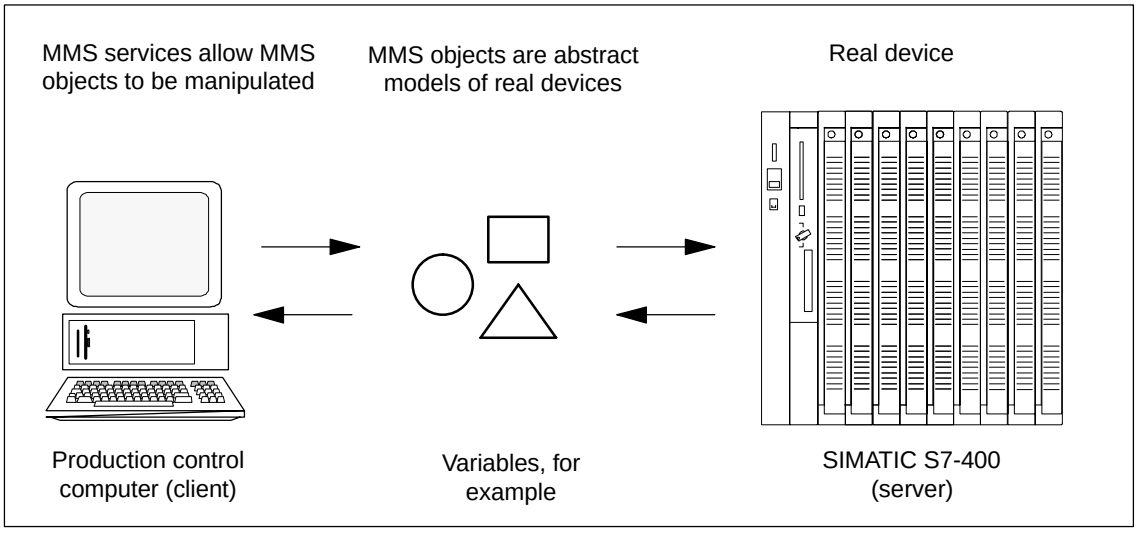


Figure 2-1 MMS Model of a Communication Example: Production Control Computer ↔ SIMATIC S7-400

An Example

A measured value is, for example, an MMS object that belongs to the variable class. This MMS object can be addressed by means of the MMS “Read” and ”Write” variable services. These MMS services ensure that the MMS object is transmitted in a coherent form, regardless of the type of representation and interpretation of the end device.

Client/Server Relationship

The MMS services always assume two communication partners that communicate with each other. The communication involves one partner making available MMS objects that the other communication partner accesses by means of MMS services. The communication partner that provides the MMS objects is the server (object manager). The communication partner that accesses the MMS objects is the client (object user).

The client requests an MMS service from the server to access MMS objects. The server provides the requested MMS service and gives the result to the client.

However, the server can in certain cases also send an MMS service to the client in order to provide notification of the state change of an MMS object, for example. The server still remains the server in spite of the fact that it is sending an MMS service to the client. The server acts on its own initiative in this case; an MMS service like this does not affect an MMS object in the client.

An Example

A control computer (client) uses the MMS ”Read” variable service to request the transmission of a process value in a programmable controller. To do this, it uses the association to the application process in an S7-400. The application process in the S7-400 is the server that makes the ”Read” variable service available to the control computer (see Figure 2-1).

Alternating Role Play

A communication partner can be both a client and a server. In other words, the communication partner can both request MMS services (as a client) and provide MMS services (as a server).

2.2.1 The ”Virtual Programmable Controller” (VMD)

Virtual Programmable Controller (VMD)

The prerequisite for open communication between different programmable controllers is a standardized interface between a programmable controller and its communication partner. In other words, there must be a common basis for communication.

There are a great many components of different types (e.g. personal computers, programmable logic control systems and production control computers), so it is necessary to represent the various hardware and software structures uniformly. This representation of a real programmable controller as an abstract model is referred to as a ”virtual programmable controller” or VMD (Virtual Manufacturing Device).

You can imagine the VMD as the outer skin of the real programmable controller (see Figure 2-2). Data exchange between the real programmable controller and an external communication partner is only possible via the standardized interface of this outer skin.

The structure and functionality of the real device are described by means of MMS objects. MMS provides services for these objects that allow these objects to be manipulated and thus the real programmable controller to be addressed.

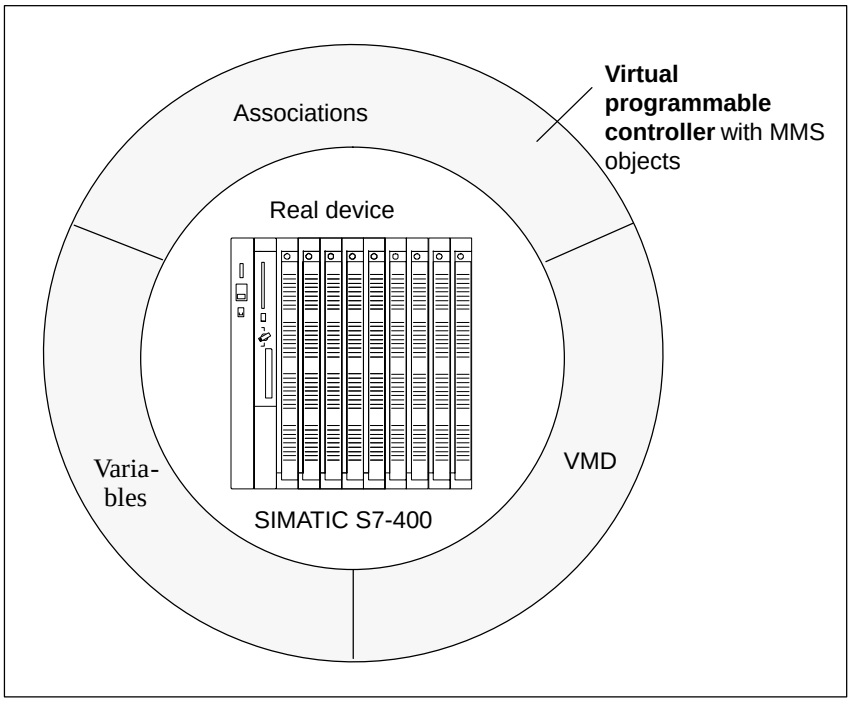


Figure 2-2 SIMATIC S7-400 as Virtual Programmable Controller (VMD)

CP 444 and VMD

What the CP 444 communication processor essentially does is to map an S7-400 to a VMD.

In the SIMATIC, every CP 444 is considered, together with its assigned programmable controller (in which the CP 444 is located), to be a single VMD.

A VMD always consists of a CP 444 and (in the case of multiprocessor operation) up to four CPUs. It is possible to use several CP 444s in a programmable controller. Since the communication processors all work independently of each other, every CP 444 can be seen as a VMD.

2.2.2 Overview of the MMS Objects

MMS Objects

The following MMS objects are defined for communication with the CP 444 (see also Figure 2-2):

Table 2-3 MMS Objects of an S7-400 When the CP 444 Is Used

Object	Meaning
Association	This is the communication channel between two application processes.
VMD	This is the SIMATIC S7-400 itself. The device state and device properties of the S7-400 can be obtained in a standardized form by means of MMS services.
Variable	This is a readable and writable data area in the SIMATIC S7-400. MMS services allow variables to be read or written.

Associations

As far as the user is concerned, communication with the application processes of the communication partners takes place via logical channels (associations).

The view of the communication partner and its automation task is defined by means of these associations. The user must specify which automation task he or she wants to address by means of which association and whether the association is to be set up passively or actively.

Variables

Variables are MMS objects by means of which user data can be represented. Variables are identified by names and assigned a data type description. The data type description allows a standardized view of the data to be obtained throughout the system. The data type can be simple or complex.

Variables can have two different areas of validity:

- **VMD-specific variables**

  These variables apply throughout the whole virtual programmable controller (VMD). VMD-specific variables can be accessed by means of any association.

- **Association-specific variables**

  These variables are assigned to a single association. Association-specific variables can only be accessed by means of this association.

2.2.3 Overview of the MMS Services

MMS Services

The following services implemented on the CP 444 can be addressed for communication with the CP 444:

Table 2-4 MMS Services of an S7-400 When the CP 444 Is Used

Services	Meaning	Described in
Association management (services)	... for setting up, maintaining and clearing an association Applications that are to communicate with each other must set up, maintain and clear a logical connection (association).	Chapter 5
VMD services	... for device monitoring The VMD (Virtual Manufacturing Device) services allow information to be obtained on the properties and state of a VMD (e.g. the objects that exist and the operating state of the devices).	Chapter 6
Variable services	... for language-independent data traffic Variable services are services for reading and writing the values of variables. This data may be simple (e.g. integer) or complex (e.g. structure). A standard syntax is defined for data type description; this overcomes language barriers in data type description (an S7 data block is thus readable in a control computer).	Chapter 7

An Example

A measured value is, for example, an MMS object that belongs to the variable class. Services defined in this class are ”Read”, ”Write” and ”InformationReport”.

Setting Up the CP 444

3

Procedure

To set up the CP 444, you will need to do the following things in the order given:

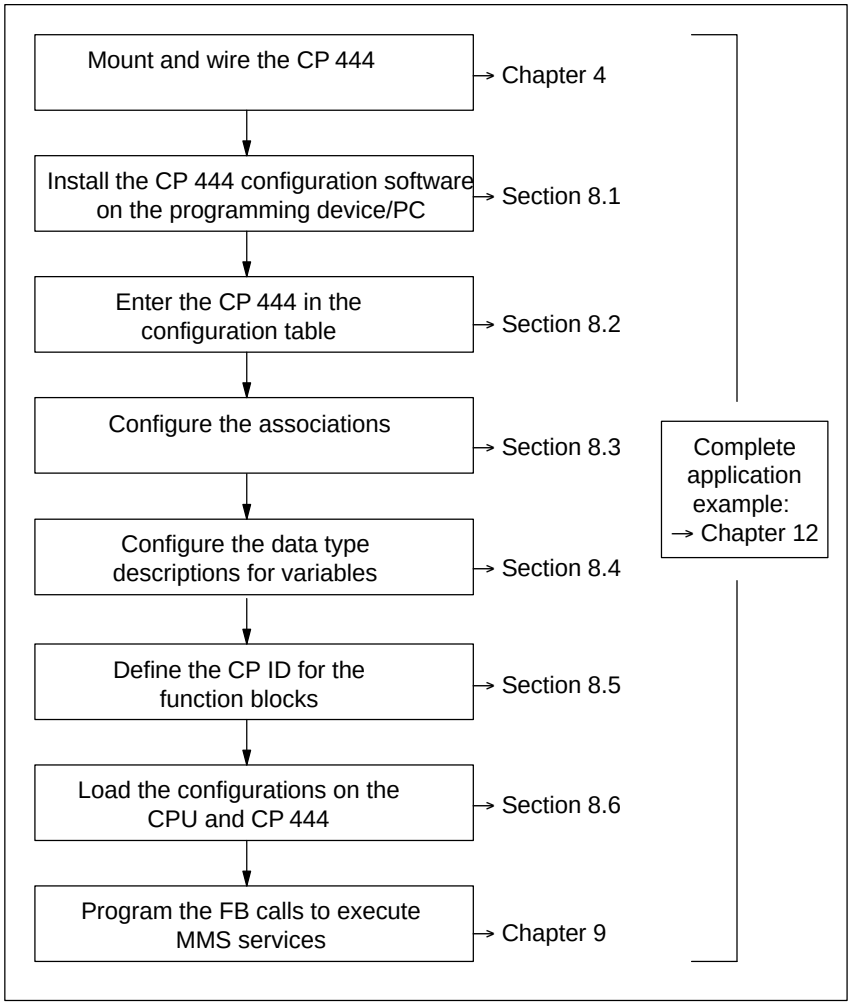


Figure 3-1 Procedure at Setup

Mounting and Wiring

Mounting the CP 444 involves installing the CP 444 in your programmable controller's mounting rack.

Installing the Configuration Software

Before you can configure and parameterize the CP 444 under *STEP 7*, you must install on the programming device/PC the software supplied with the CP 444 configuration package (3.5” disks).

Configuring the CP 444

Configuring the CP 444 involves entering it in the *STEP 7* configuration table. You cannot enter a CP 444 in the configuration table until you have successfully installed the software supplied.

Configuring the Associations

To set up associations between the CP 444 and the remote communication partner, you must store address information and connection-specific parameters on the CP 444.

These parameters are specified in a text file using the *Configuration CP 444* parameterization interface and then compiled in a specific format.

Configuring the Variables

In order to enable the exchange of user data between the CP 444 and the remote communication partner, you must configure variables and their data type descriptions.

The data type descriptions of the variables are specified in another text file using the *Configuration for CP 444* parameterization interface and then compiled in a specific format.

Defining the CP ID

The connection between the CPU and the connected CP 444 is defined with the CP ID. You reference the CP ID at the input of the function blocks for the request of MMS services.

Loading the Configurations

Once configuration is complete, the configuration and parameterization data is loaded on the CPU and the CP 444 while they are in the STOP state using *STEP 7*.

Programming the Execution of the MMS Services

You program the request of an MMS service in the user program of the CPU. Various function blocks are available for the MMS services. You merely have to call them in the user program in order to execute a service. You reference the destination or source of the data requested/for transfer at the inputs/outputs of the function blocks.

The user program is programmed using the language editors of the *STEP 7* software.

Application Example

On the disk containing the configuration software for the CP 444 you will also find a complete application example illustrating how to use the variable services. More information on this application example is given in Chapter 12.

Mounting and Wiring the CP 444

4

Purpose of This Chapter

Once you have read this chapter, you will have all the information you require in order to install the CP 444 in an S7-400 and wire the 15-pin communication interface of the CP 444.

Installation Guidelines

The CP 444 is one of the modules of the S7-400 programmable controller. The following applies to using the CP 444:

Note

The general guidelines for installing and setting up the S7-400 must be complied with (see the *S7-400/M7-400 Programmable Controllers, Hardware and Installation* manual).

Chapter Overview

This chapter is divided into the following sections:

In Section	You Will Find	on Page
4.1	Slots of the CP 444	4-2
4.2	Installing and Removing the CP 444	4-3
4.3	Wiring the Interface	4-6

4.1 Slots of the CP 444

Introduction

The following section describes the rules you must observe when installing the CP 444 communication processor in a rack.

Possible Racks

The CP 444 can be installed in the following S7-400 racks:

- UR1, UR2
- CR2

Possible Slots

Communication processors do not occupy any specific slots in the S7-400 programmable controller's rack.

The CP 444 requires two slots and can be connected to any of the slots in the racks with the following exception: The power supply module occupies slots 1 to 3 in all racks, depending on the width.

The number of plug-in CP 444 communication processors is limited:

- By the number of possible slots in the racks
- By the power consumption of the CP 444 from the S7-400 backplane bus (max. 3.1 A)

For further information on slots, see /4/.

/4/ *S7-400/M7-400 Programmable Controllers, Hardware and Installation* manual

4.2 Installing and Removing the CP 444

Introduction

When installing and removing the CP 444, you must adhere to the rules set out below.

Caution

The CP 444 is a compact module; you should not dismantle it into its component parts!

Tool

To install and remove the CP 444, you require a 3.5 mm cylindrical screwdriver.

Installation Sequence

To install the CP 444 in a rack, proceed as follows:

1. Disconnect the power supply module from the mains.
2. Remove the covers from the slots on the rack to which you want to connect the CP 444. To do this, hold the cover at the places marked, and pull it forward.
3. Hang the CP 444 on, and swing the module downward (see Figure 4-1).

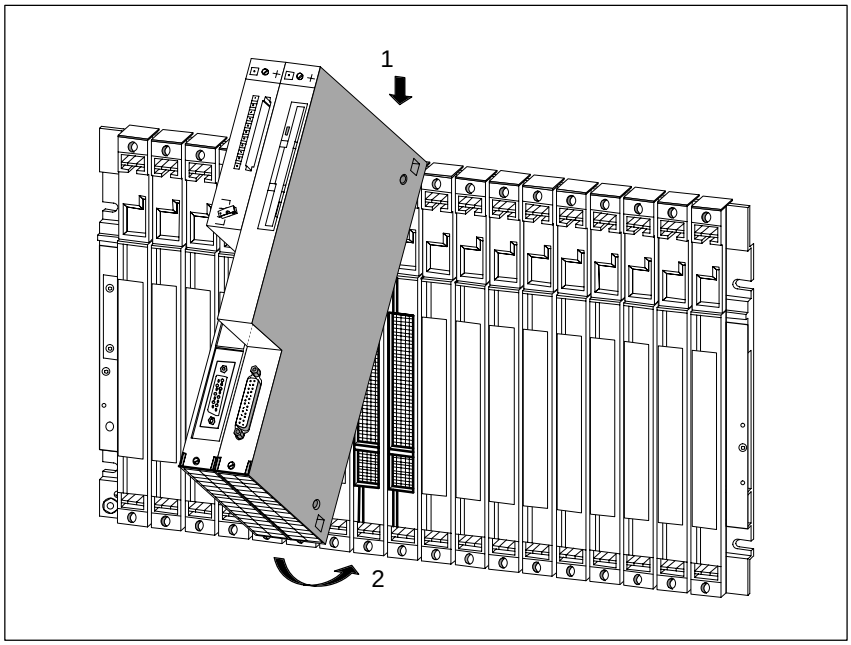


Figure 4-1 Hanging the CP 444 On and Swinging In

4. Screw the module on at the top and bottom with a torque of 0.8 to 1.1 Nm (see Figure 4-2).

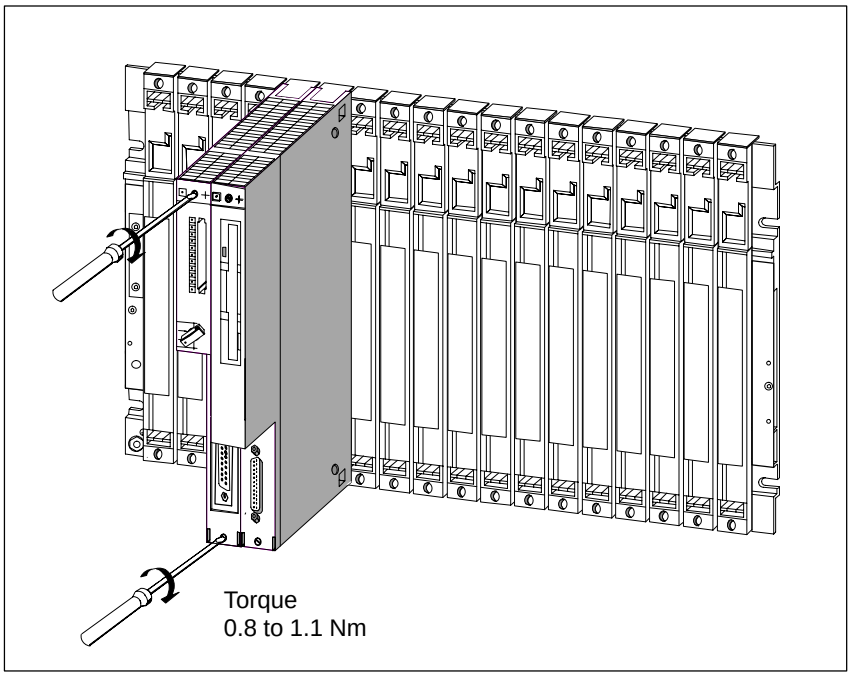


Figure 4-2 Screwing the CP 444 onto the Rack

5. Insert the key in the CP 444's mode selector, and set the mode selector to the **RUN-P** position (see Figure 4-3).

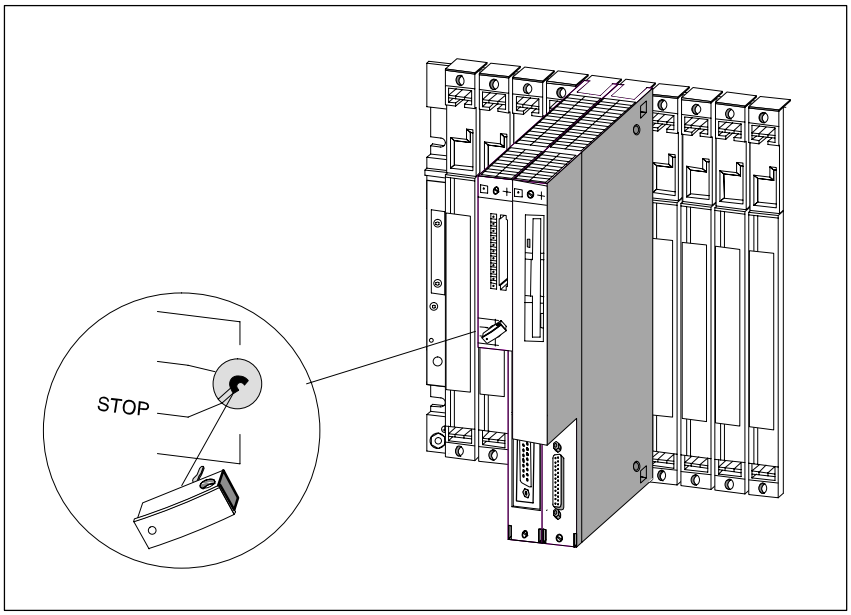


Figure 4-3 Inserting the Key in and Setting the CP 444

Removal Sequence

To remove the CP 444 from a rack, proceed as follows:

1. Use the keyswitch to switch the CPU and the communication processor to STOP.
2. Set the standby switch of the power supply module to position ⏻ (output voltage: 0V).
3. Undo the screws at the top and bottom of the module. Figure 4-4 shows the position of the screws on the module.
4. Swing the module upward, and unhook it (see Figure 4-4).

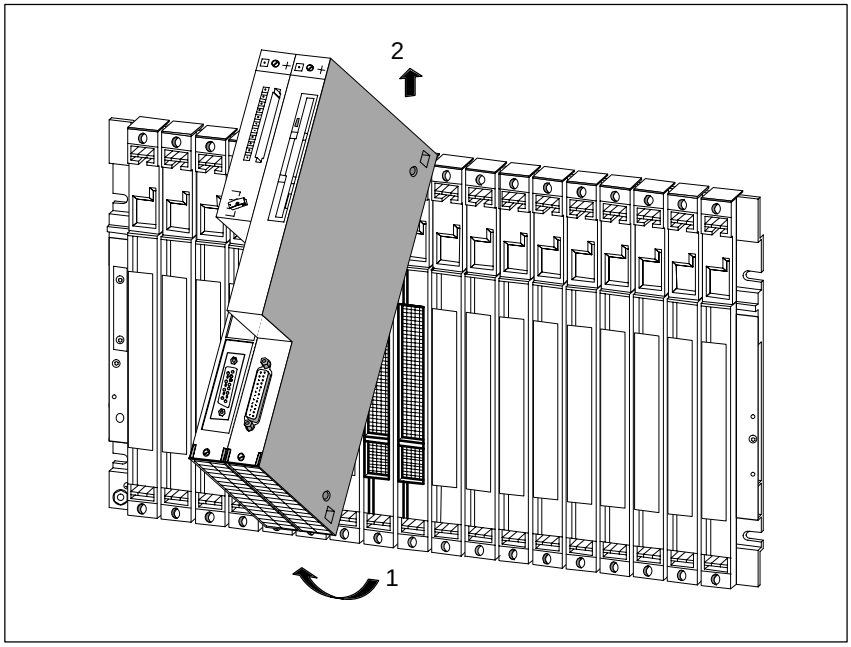


Figure 4-4 Swinging the CP 444 Out and Unhooking It

5. Replace the covers of the slots on the rack to which the CP 444 was connected.

4.3 Wiring the Interface

Interface for Connection to Ethernet

The 15-pin subminiature D female connector on the front of the CP 444 module (see Figure 1-1 in Section 1.3) is a communication interface conforming to the IEEE 802.3 Ethernet standard.

You connect the transceiver connecting cable for a SIMATIC Industrial Ethernet connection to this interface (transceiver).

No Transceiver Power Supply

The 15-pin subminiature D female connector of the CP 444 does not provide power (+ 15 V) for a transceiver.

Note

When the CP 444 is connected to a transceiver, the transceiver must also be connected via a separate power supply.

Standard Connecting Cables

Siemens provides standard connecting cables for the connection to SIMATIC Industrial Ethernet.

You can connect the following cable types to the 15-pin subminiature D female connector of the CP 444:

- 727-1 connecting cable for systems with AUI (3.2 to 50 m)
- Industrial twisted-pair installation cable (2 to 100 m)

The order numbers and possible lengths of the standard cables are given in Appendix B.

Making Your Own Connecting Cables

If you want to make your own connecting cables to the transceiver, you will find the assignment of the 15-pin subminiature D female connector in Appendix A.1.

Please note that you must use only shielded connector casings. To ensure EMC (electromagnetic compatibility), the cable shielding must be connected to each connector casing over a large area.

Other Interfaces

None of the other interfaces visible on the front of the CP 444 module are required when the CP 444 is used to provide an S7-400 with MAP communication access.

These interfaces (interface submodule, memory card slot and LPT1 interface) therefore remain unused (they are labeled in gray in Figure 1-1).

Association Management (Services) 5

Purpose of This Chapter

This chapter provides you with the information you require about the MMS services for association management. You will learn:

- Which services are executed by the CP 444
- The configuration settings for these services

This chapter explains how to configure the CP 444 with the *Configuration for CP 444* parameterization interface, which is described in Chapter 8.

Introduction

If applications are to communicate with each other, they must set up, maintain and clear a logical connection (association).

The CP 444 provides association management services that are required for communication on the application layer (layer 7 of the ISO/OSI reference model).

The CP 444 executes these services autonomously.

Services

The CP 444 provides the following services for association management:

Table 5-1 Services for Association Management

Service	Purpose	Comment
Initiate	Sets up an association.	The association can be set up passively or actively (initiation type).
Conclude	Concludes an association.	The association can be concluded by the remote communication partner.
Abort	Aborts an association.	In the event of serious errors **and** when configuration data is loaded, the CP 444 initiates an Abort automatically. The CP 444 can also receive an Abort from the remote communication partner.

Initiative

The association management services are executed by the CP 444 without being activated by the user program (CPU).

The parameters required to execute the services are entered using the *Configuration for CP 444* parameterization interface when the associations are defined (see Section 8.3) and are stored in the CP 444 (see Section 8.6).

The associations are set up when the CP 444 starts up. The initiation of an association depends on:

- The initiation type configured
- The address information configured

Note

The association is initiated when the destination address information (remote entry) of the active application process and the source address information (local entry) of the passive application process match.

Initiation Type

There are two association initiation types for the CP 444:

- **Active initiation**

  The association is initiated immediately after the power is switched on or the CP 444 is started. If the connection is interrupted, the CP 444 initiates automatic reinitiation.

- **Passive initiation**

  The initiation of the association is started by the remote communication partner (e.g. by a production control computer). In other words, the CP 444 expects the connection to be set up.

VMD Services

6

Purpose of This Chapter

This chapter describes the MMS services for device monitoring, known as VMD services. You will learn:

- Which VMD services are executed by the CP 444
- What you have to program in the user program of the CPU

This chapter explains how to program the VMD services using *STEP 7*, which is described in Chapter 9. If you do not use any VMD services, for example, you only use variable services, you do not need to read this chapter.

Chapter Overview

This chapter is divided into the following sections:

6.1 Overview of the VMD Services

Introduction

The VMD (Virtual Manufacturing Device) services allow information to be obtained on the properties and status of a VMD (e.g. the objects that exist and the operating status of the devices, etc.).

In the SIMATIC, every CP 444, together with its assigned programmable controller (in which the CP 444 is located) is considered to be a single VMD.

Client/Server Functionality

The VMD services for virtual manufacturing devices (VMDs) enable a client to obtain information on the status or attributes of the VMD in the server. If required, the server can also report the status to a client spontaneously (without being requested).

The information can then be further processed on the client, so that, in the case of a control computer, for example, an overview of the overall status of the system can be obtained.

VMD Services

The CP 444 can make available the following VMD services for virtual manufacturing devices (VMDs) as a client and/or server:

Table 6-1 VMD Services

Service	Purpose	Server	Client
Status	Checks the status of the VMD	×	×
UnsolicitedStatus	Reports the status of the VMD without a request	×	
GetNameList	Obtains the name list	×	
Identify	Identifies the VMD	×	×
GetCapabilityList	Obtains the capability list	×	

× The CP 444 can support this service as a server or client.

Initiative

When the CP 444 provides VMD services as a server, it executes them independently.

VMD services it provides as a client must be programmed by means of an FB call in the user program of the CPU. Chapter 9 of this manual describes all the required function blocks (FBs) and how they are programmed in the user program.

6.2 General VMD Services for an S7-400 as a Server

Introduction

The server functionality of the CP 444 allows it to provide services that enable the communication partner (client) to obtain information on the status of the VMD (CP 444 and CPU).

Services

The following VMD services are supported for a remote communication partner (client) accessing the CP 444 (server):

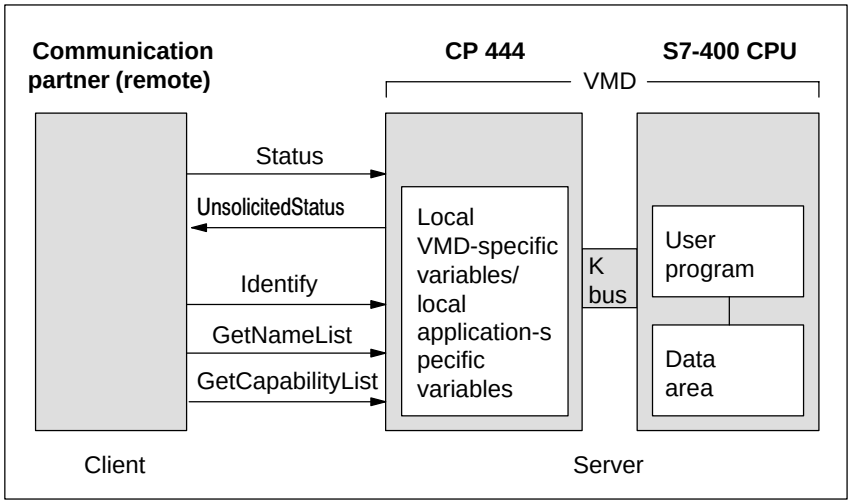


Figure 6-1 VMD Services for Access by Remote Communication Partners

Status

The ”Status” service allows the remote communication partner (client) to request information on the physical and logical status of the VMD managed in the CP 444 (server). The CP 444 sends back the requested information (e.g. whether the CPU is in RUN or STOP mode).

Initiative: The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

Table 6-2 Possible Responses of the CP 444 to a ”Status” Service Request

Physical Status (VMD)	Logical Status (VMD)	Meaning
Operational	State-Changes-Allowed	CPU in RUN: • All MMS services allowed
Partially-Operational	State-Changes-Allowed	CPU in STOP: • All MMS services allowed
Inoperable	Limited-Services-Permitted	No connection between the CP 444 and CPU: • All VMD services allowed • Variable services **not** allowed

UnsolicitedStatus

The ”UnsolicitedStatus” service allows the CP 444 (server) to report to the communication partner (client) the physical and logical status of its own VMD on its own initiative (spontaneously) (see Table 6-2 for the possible messages).

Initiative In the event of a status change (STOP, RUN), the CP 444 is notified by the CPU. The CP 444 then sends the new status to the communication partner. No programming is necessary in the user program of the CPU at the server end.

Identify

The remote communication partner (client) uses the ”Identify” service to request information on the attributes of the VMD managed in the CP 444 (server). These attributes might be, for example, the name of the vendor of the programmable controller, the module designation of the CP 444 and the revision identifier (firmware status) of the CP 444.

Initiative: The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

Table 6-3 Possible Responses of the CP 444 to an ”Identify” Service Request

Information	Value reported
Vendor Name	Siemens AG
Model Name	CP 444
Revision Identifier	<Version> <Datum>

GetNameList

This service is available only with the server functionality of the CP 444. The information requested by means of this service is sent by the CP 444 independently to the requester of the service. The response information sent by the CP 444 might contain, for example, the list of all the variables defined in and managed by the CP 444.

Initiative The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

GetCapabilityList

This service is only available with the server functionality of the CP 444. The information requested by means of this service is sent by the CP 444 independently to the requester of the service. The CP 444 sends an empty list.

Initiative The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

6.3 General VMD Services for an S7-400 as a Client

Introduction

The client functionality of the CP 444 allows it to provide services that enable it to obtain information on the status of the VMD of the communication partner.

Services

The following VMD services are supported for the CP 444 (client) accessing a remote communication partner (server):

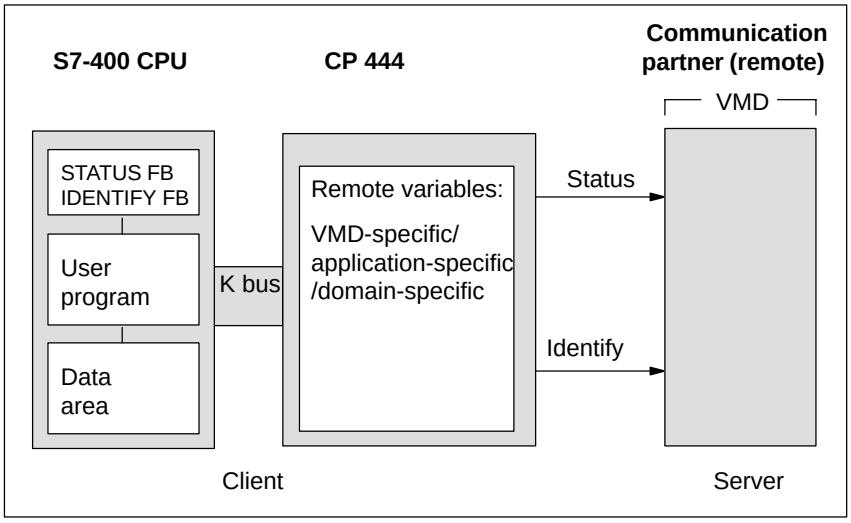


Figure 6-2 VMD Services for Access to Remote Communication Partners

Status

The CP 444 (client) uses the ”Status” service to request information on the physical and logical status of the VMD of the communication partner (server). The communication partner must send the requested information.

Initiative: The initiative is taken by the CP 444. The call of the STATUS function block (FB) must be programmed in the user program of the CPU (see Section 9.6).

Identify

The CP 444 (client) uses the ”Identify” service to request information on the attributes of the VMD of the communication partner (server). These attributes might be, for example, the name of the vendor, the module designation and the release status of the communication partner.

Initiative The initiative is taken by the CP 444. The call of the IDENTIFY function block (FB) must be programmed in the user program of the CPU (see Section 9.3).

Variable Services

7

Purpose of This Chapter

This chapter describes the MMS services for language-independent data traffic, known as variable services. You will learn:

- Which variable services are executed by the CP 444
- The configuration settings for these services
- What you have to program in the user program of the CPU

This chapter explains how to configure the CP 444 using the *Configuration for CP 444* parameterization interface, which is described in Chapter 8, and how to program the variable services using *STEP 7*, which is described in Chapter 9.

If you do not use any variable services, for example, you only want to request the device status (VMD services), you do not need to read this chapter.

Chapter Overview

This chapter is divided into the following sections:

7.1 Overview of the Variable Services

Introduction

Variable services are services for reading and writing the values of variables. This data may be simple (e.g. integers) or complex (e.g. structures).

A standard syntax has been defined for data type description; this overcomes language barriers at data type description and enables data exchange regardless of the end system involved (an S7 data block can be read in a control computer, for example). The variables are ”converted” to the format of the respective end system both at the client and the server end.

Client/Server Functionality

As far as the variable services are concerned, this affects the client/server functionality as follows:

Client

The S7-400 with the connected CP 444 functions as a client when you want to access variables defined in another (remote) communication partner. The name and data type description of the variables are required to access these variables. This structural information is stored on the CP 444.

Server

The S7-400 with the connected CP 444 functions as a server when you want to provide services for read and write access to variables. The variables are managed by the CP 444 in accordance with the configured structural information (name, data type description).

Variable Services

The CP 444 can make available the following variable services for virtual manufacturing devices (VMDs) as a client and/or server:

Table 7-1 Variable Services

Service	Purpose	Server	Client
Read	Reads a variable.	×	×
Write	Writes a variable.	×	×
InformationReport	Reports a variable.	×	×
GetVariableAccess-Attributes	Obtains variable attributes.	×	

× The CP 444 can support this service as a server or client.

Initiative

Some variable services are executed independently by the CP 444. Other variable services must be programmed by means of an FB call in the user program of the CPU. Chapter 9 of this manual describes all the required function blocks (FBs) and how they are programmed in the user program.

The structural information of the variables for executing the variable services is configured using the *Configuration for CP 444* parameterization interface (see Section 8.4) and stored in the CP 444 (see Section 8.6). Depending on the configured scope of validity of the variable (VMD-specific, association-specific), a variable is assigned to the whole VMD or to a specific association.

VMD-Specific Variables

Since an S7-400 programmable controller always represents a VMD in SIMATIC S7, “VMD-specific” means that the variable is valid and known in the whole S7-400.

- **Access:**

  Access is permitted via any association. A VMD-specific variable is visible from every communication partner. In other words, any communication partner can access this variable via any association.

- **Application:**

  Global lists or variables accessed by different communication partners.

Association-Specific Variables

Association-specific variables are assigned to a specific association.

- **Access:**

  The variable is visible only via this association. In other words, it is only possible to access the variable via this association.

- **Application:**

  The association can provide access to one of several application processes of a VMD. Use and access to data areas can be restricted by means of assignment to an association and thus to a specific application process.

7.2 Variable Services for an S7-400 as a Server (Local Variables)

Introduction

The server functionality of the CP 444 allows it to provide variable services that enable the communication partner (client) to access data areas in the CPU (server).

Data Type Description

The data type description of the variables is configured using the *Configuration for CP 444* parameterization interface and stored on the CP 444.

We refer to **local** variables in configuration because the source of the variables to be read or the destination of the variables to be written is in the local VMD. You configure these variables in the following parameter blocks, depending on the area of validity (see Section 8.4):

- **`%Begin_LocalVmdSpecificVariables`**
- **`%Begin_LocalAssociationSpecificVariables`**

Data Management

The real variables themselves or the buffer for the variables must be made available in the data area of the CPU.

Services

The following variable services are supported for a remote communication partner (client) accessing data areas in the CPU (server):

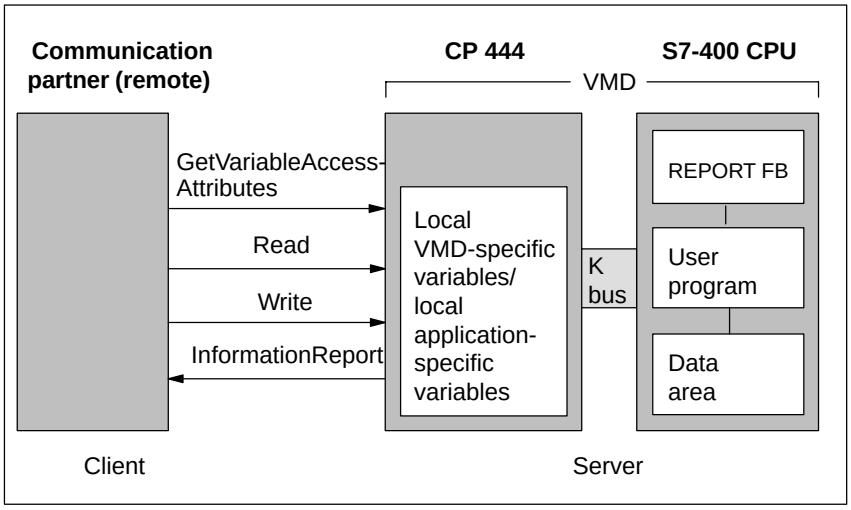


Figure 7-1 Variable Services for Access by Remote Communication Partners

GetVariable AccessAttributes

The remote communication partner (client) uses the "GetVariableAccessAttributes" service to request the CP 444 (server) to send information on the attributes of a specific local variable (e.g. the scope of validity).

Initiative: The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

Read

The remote communication partner (client) uses the "Read" service to obtain the value of a variable in the CPU (server). The variables that are read must be configured as local variables on the CP 444 (server). The scope of validity is defined as well.

Initiative: The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

Write

The communication partner (client) uses the "Write" service to transfer data to the CP 444 (server). The CP 444 causes the specified variable to be overwritten in the CPU with the value transferred. The variables that are written must be configured as local variables on the CP 444 (server). The scope of validity is defined as well.

Initiative: The initiative is taken by the remote communication partner (client). No programming is necessary in the user program of the CPU at the server end.

InformationReport

The CP 444 (server) uses the "InformationReport" service to send data type descriptions and variable values to the remote communication partner (client) without being explicitly requested by the latter to do so. The relevant variables must be configured as local variables (with scope of validity) on the CP 444 (server).

Initiative: The initiative is taken by the CP 444. The call of the REPORT function block (FB) must be programmed in the user program of the CPU (see Section 9.5).

7.3 Variable Services for an S7-400 as a Client (Remote Variables)

Introduction

The client functionality of the CP 444 allows it to provide variable services that enable it to access data areas of the communication partner (server).

Data Type Description

The data type description of the variables is configured using the *Configuration for CP 444* parameterization interface and stored on the CP 444.

We refer to **remote** variables in configuration because the source of the variables to be read or the destination of the variables to be written is the communication partner (VMD). You configure these variables in the following parameter blocks, depending on the area of validity (see Section 8.4):

- **`%Begin_RemoteVariables`**

Data Management

A buffer for the data read or written must be made available in the data area of the CPU.

Services

The following variable services are supported for the CP 444 (client) accessing data areas on the remote communication partner (server):

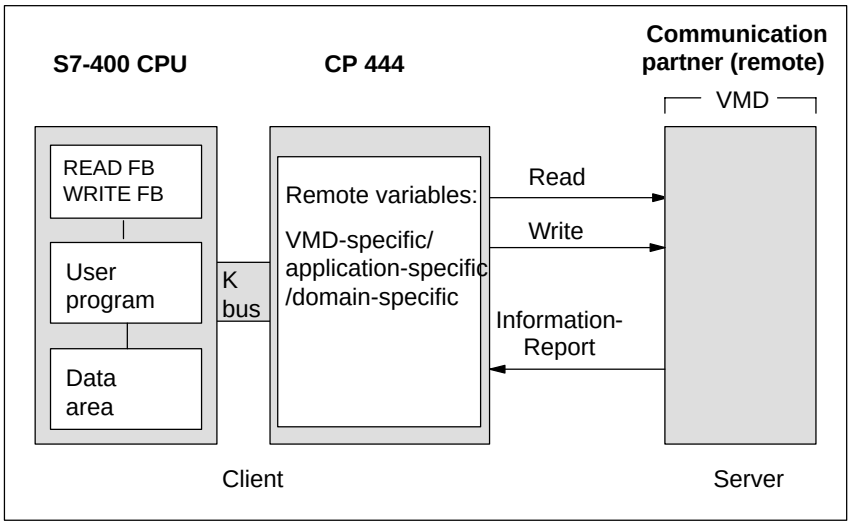


Figure 7-2 Variable Services for Access to Remote Communication Partners

Read

The CP 444 (client) uses the “Read” service to obtain the value of a variable in the remote communication partner (server). The data of the variables is transferred from the communication partner to the CP 444 and entered in the data area of the CPU. The variables to be read must be configured as remote variables (with scope of validity) on the CP 444 (client).

Initiative: The initiative is taken by the CP 444. The call of the READ function block (FB) must be programmed in the user program of the CPU (see Section 9.4).

Write

The CP 444 (client) uses the “Write” service to transfer data to the remote communication partner (server). The data is transferred from the CP 444 to the communication partner and entered, overwriting the corresponding local variable of the communication partner. The variables to be written must be configured as remote variables (with scope of validity) on the CP 444 (client).

Initiative: The initiative is taken by the CP 444. The call of the WRITE function block (FB) must be programmed in the user program of the CPU (see Section 9.7).

InformationReport

The remote communication partner (server) uses the ”InformationReport” service to send data type descriptions and variable values to the CP 444 (client) without being explicitly requested by the latter to do so. The relevant variables must be configured as remote variables (with scope of validity) on the CP 444 (client).

Initiative: The initiative is taken by the remote communication partner (server). No programming is necessary in the user program of the CPU at the client end.

7.4 Coordinated and Uncoordinated Accessing of Variables

Introduction

You can control write and read access to variables at the **server end**. You program access coordination in the user program of the CPU with the ACCESS FB function block.

Access Protection Options

Depending on the programming of the ACCESS FB call in the user program of the CPU, the following action is possible:

- Consistent transfer

  You can use the ACCESS FB to ensure that the user program always accesses fully updated variables (”Write” service) or that the user program does not change the variables as long as reading of the variable is still active (”Read” service).

  This prevents incompletely transferred data areas from being accessed as a result of the cyclical processing of the user program in the CPU (server).

- Coordinated access

  In addition to ensuring consistent transfer, you control variable access by temporarily blocking or enabling read or write access.

  This means that the server can, for example, protect a data record from being read again until a specific control sequence has been processed in the user program.

- With FB ACCESS controllable services.

  FB ACCESS makes the coordinated access of server variables possible with the services “Read” and “Write.” The service “Information Report” is not affected by FB ACCESS. Write access of server variables with FB Report is possible at all times.

Programming the ACCESS FB

The description of FB ACCESS in Section 9.2 of the user program provides information on the structure of the user program.

Principle

You control access to variables by setting and resetting control bits and querying control bits in an ACCESS-DB data block. You reference the data block number of the ACCESS-DB at the input of the ACCESS FB.

The ACCESS FB must be called at least once in each cycle of the user program.

7.5 Addressing Local and Remote Variables

Introduction

This section provides you with an overview of the addressing of local and remote variables, depending on the variable service and the client/server functionality.

Overview

The table below indicates where the location of the variable values must be made known and at which end (server/client) a function block (FB) has to be programmed in the user program.

Table 7-2 Addressing Local and Remote Variables

Server/Client	Variable Service	FB Call Necessary	Configuration of S7 Addresses for Variables	
			Parameters in Text File (S7Addr)	Input at FB (RD_1, SD_1)
Server (local variables)	Read (receive)	–*	×	
	Write (receive)	–*	×	
	InfomationReport (send)	REPORT FB		×
Client (remote variables)	Read (send)	Read FB		×
	Write (send)	Write FB		×
	InfomationReport (receive)	–*	×	

* A function block (FB) is not required for uncoordinated access.
Cyclical calling of the ACCESS FB is necessary for coordinated access.

Configuration

In the text file for variables you must configure for the CP 444 the complete data type descriptions and S7 addresses of both the **local** and the remote **remote** variables (see Section 8.4.3).

Note, however, that, in the case of the ”InfomationReport (send)”, ”Read (send)” and ”Write (send)” variable services, the S7 address (data block/data word number) referenced at the input of the function block becomes effective (see Table 7-2).

Configuring and Parameterizing the CP 444

8

Purpose of This Chapter

After working through this chapter, you will have completed the configuration of the CP 444 under *STEP 7* using the configuration software supplied.

Procedure

To configure and parameterize the CP 444, proceed as follows:

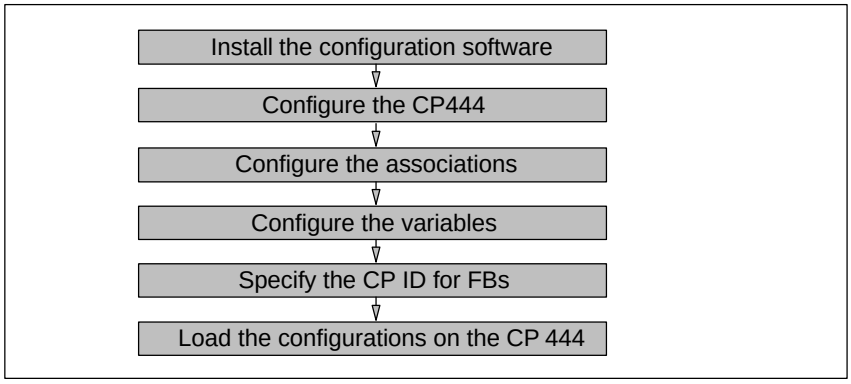


Figure 8-1 Configuration Procedure

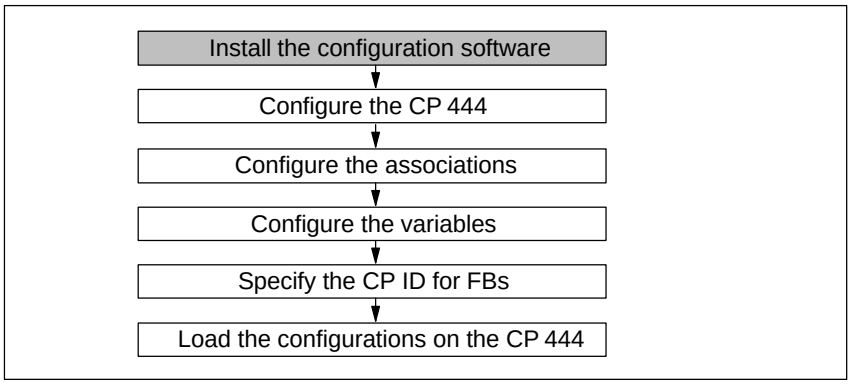
Chapter Overview

The various steps of the procedure are described in the following sections.

In Section	You Will Find	on Page
8.2	Installing the Configuration Software Under STEP 7	8-3
8.2	Configuring the CP 444	8-3
8.3	Configuring the Associations	8-4
8.4	Configuring the Variables	8-16
8.5	Specifying the CP ID	8-27
8.6	Loading the Configuration Data	8-28

8.1 Installing the Configuration Software Under STEP 7

Where Am I?



Introduction

Before you can configure and parameterize the CP 444 under *STEP 7*, you must install on the programming device/PC under *STEP 7* the software supplied with the CP 444 configuration package (3.5” disks).

Note

You cannot enter a CP 444 in the *STEP 7* configuration table until you have successfully installed the software supplied.

Prerequisite

A standard version of *STEP 7* Version 4.0 (or higher) must be installed on your programming device/PC.

Installation

To install the software, proceed as follows:

1. Under *Windows 95/NT*, start the dialog for installing software by double-clicking the ”Add/Remove Programs” icon in ”Control Panel”.
2. Click ”Install”.
3. Insert floppy disk 1 in the floppy disk drive of your programming device/PC, and then click ”Next”. *Windows 95/NT* looks independently for the **`setup.exe`** installation program.
4. Follow the step-by-step instructions of the installation program.

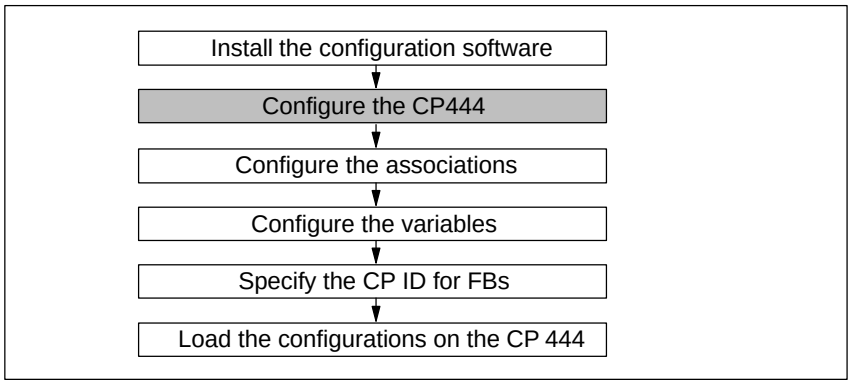
Result: The software required for the CP 444 is installed under *STEP 7*. The *Configuration for CP 444* parameterization interface is located in the following directory: **`SIEMENS\STEP7\S7wfm`**

Current Information

For current information and installation notes for the software, please refer to the **`Liesmich.wri`** file (German) or **`readme.wri`** file (English) in your installation directory.

8.2 Configuring the CP 444

Where Am I?



Introduction

Once you have installed the configuration software under *STEP 7* on the programming device/PC, you can begin configuring the CP 444 by entering the module in the *STEP 7* configuration table.

Prerequisite

Before you enter the CP 444 in the configuration table, you must use *STEP 7* to create a project and the SIMATIC 400 station to be assigned to the CP 444.

Configuration

To configure the CP 444, proceed as follows:

1. Open the rack of the SIMATIC 400 station in which the CP 444 is to be installed.

   Result: The slots of the rack are displayed.

2. Select the CP 444 from the module catalog.

3. Drag the module to the appropriate line in the configuration table.

Result: *STEP 7* automatically assigns an address to the CP 444. The CPU is then able to find the CP 444 in its slot on the rack by means of its address.

Reference Material

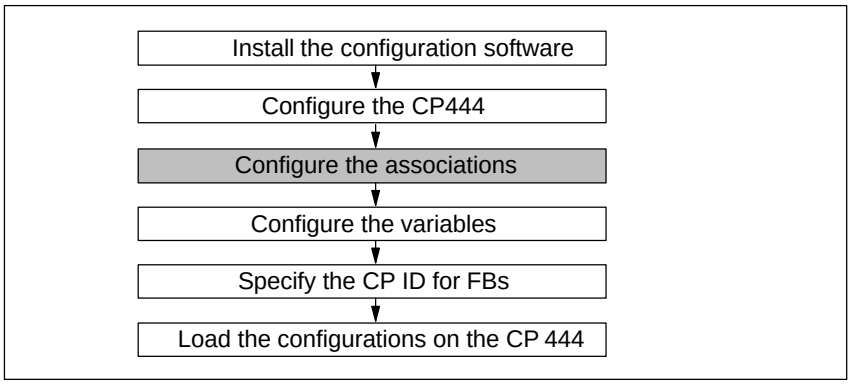
How to configure S7-400 modules is described in detail in the *STEP 7* /5/ User Manual.

In addition, the on-line help system of *STEP 7* provides you all the help you will need to configure an S7-400 module.

/5/ *Standard Software for S7 and M7, STEP 7* user manual

8.3 Configuring the Associations

Where Am I?



Introduction

Once you have entered the CP 444 in the configuration table, configure the associations and variables.

Principle

All the parameters of the associations of the CP 444 must be stored in a text file. You generate the text file by means of the *Configuration for CP 444* parameterization interface.

The parameterization interface provides a text editor. You can use the text editor (see Figure 8-3) either to create a new text file containing parameters or to open and edit an existing text file (sample file).

Once you have finished editing the text file, you compile the text file into a specific format. This format is required to store the parameters in a module SDB of the CP 444 (see Section 8.6).

Starting the Parameterization Interface

You start the parameterization interface by double-clicking the CP 444 in the configuration table or by selecting the CP 444 and choosing the **Edit > Object Properties** menu command.

Result: The ”Properties – CP 444” dialog box appears.

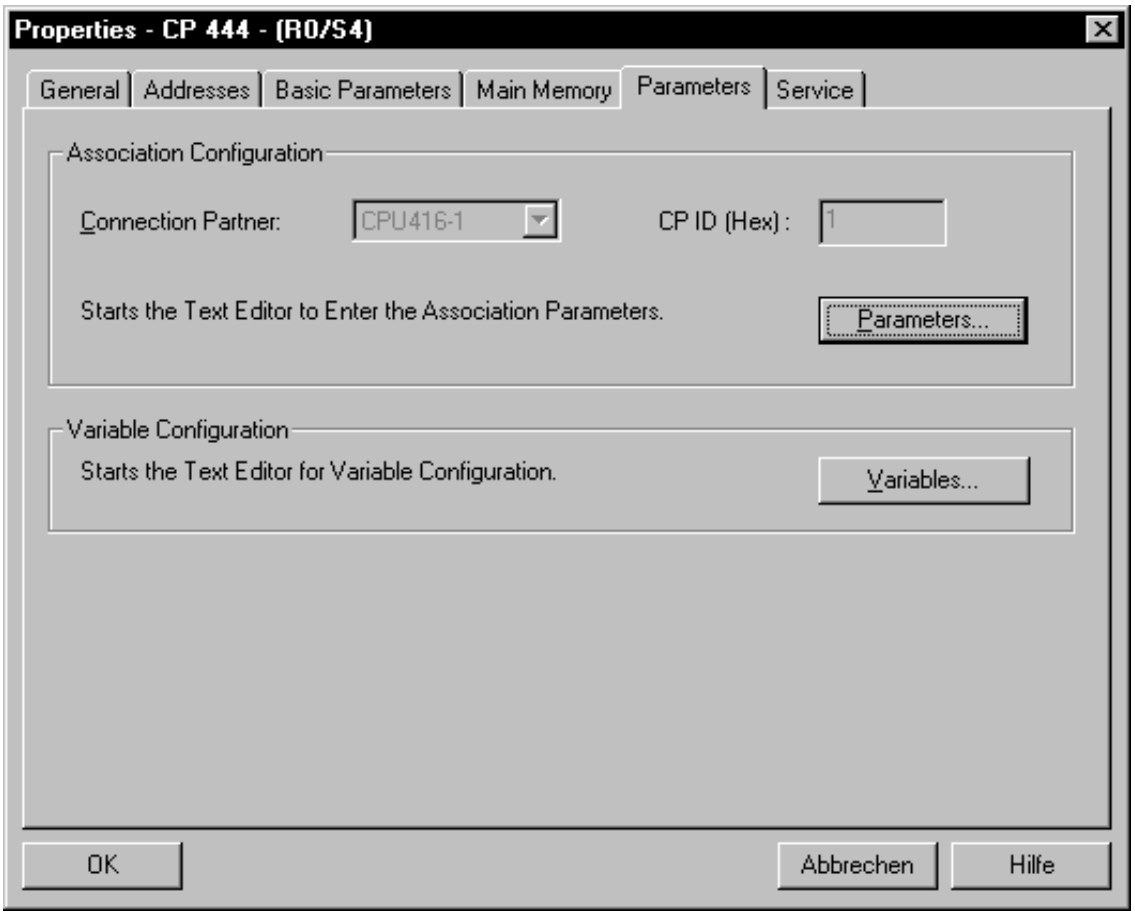


Figure 8-2 “Properties - CP 444” Dialog Box (”Parameters” Tab)

Editing a Text File

To edit the text file for the associations, proceed as follows:

1. Change to the “Parameters” tab. To do this, simply click the ”Parameters” tab label near the top of the dialog box.

   Result: The ”Parameters” tab appears in the foreground (see Figure 8-2).

2. In the ”Connection Partner” list box, select the CPU to which you want to create a K bus connection (see also Section 8.5).

3. Click the “Parameters” button.

   Result: The window of a text editor appears. You use this text editor to create the text file (Extension .con) containing the settings of the associations.

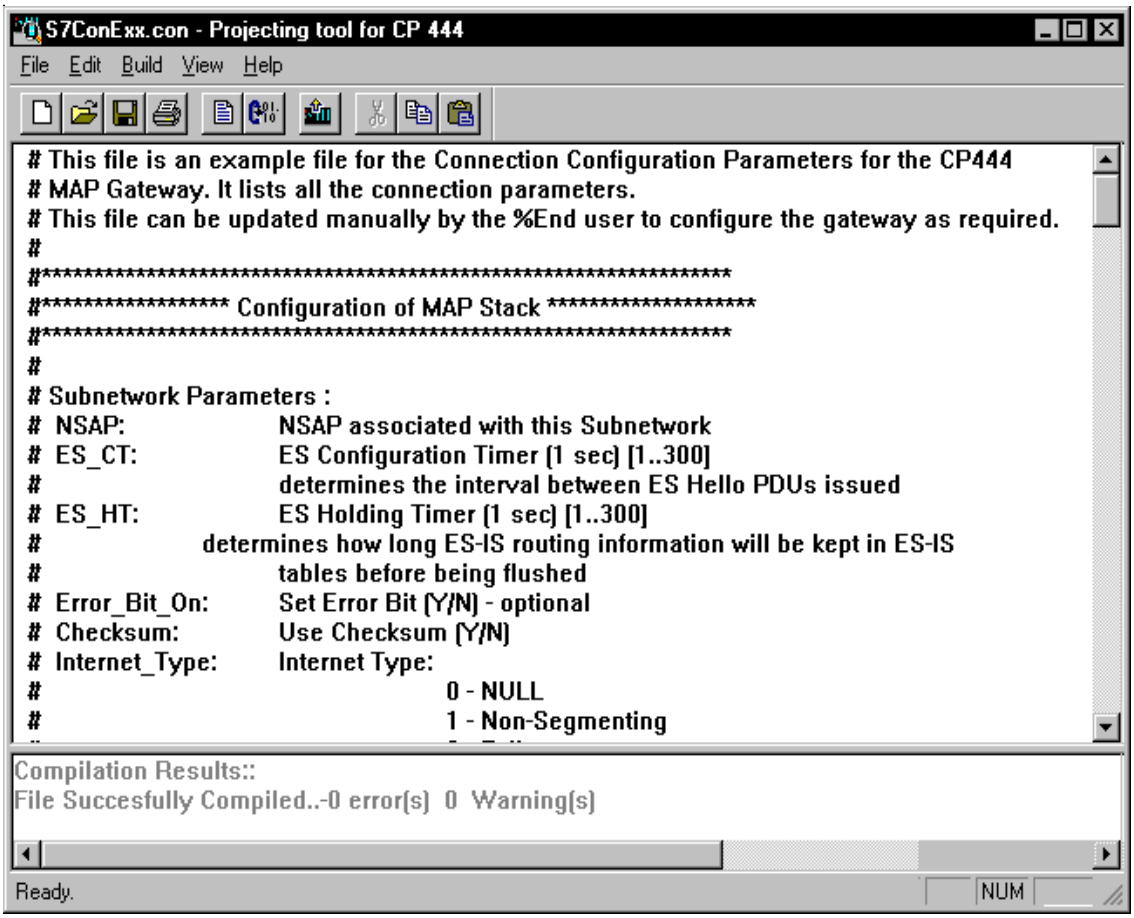


Figure 8-3 “Name.con - Configuration for CP 444” Dialog Box (Associations)

4. Choose **File > Load Example** to open the sample file.

   Result: The contents of the sample file appear with the default parameters. The parameters are subdivided into separate parameter blocks.

5. Enter the parameters and parameter values in accordance with the requirements of the associations.

   – Structure of the parameter blocks: See Figures 8-4 and 8-5

   – Purpose of the parameters and possible values: See Tables 8-1 to 8-7.

6. Choose **File > Save As** to save the entries in another text file.

7. After you have entered all the parameters, choose **Build > Compile**.

   Result: The current (open) text file is converted to the format required to create the module SDB.

Incorrect Entries

If you make incorrect entries in the text file, error messages are generated and displayed in the lower part of the text editor window (see Figure 8-3). If you **double-click an error message**, the relevant line in the text file is displayed and selected.

Not until all the errors have been eliminated is the text file converted to the SDB-specific format.

Structure of the Text File

The parameters are assigned to different parameter blocks. The structure of the parameter blocks in the text file is as follows:

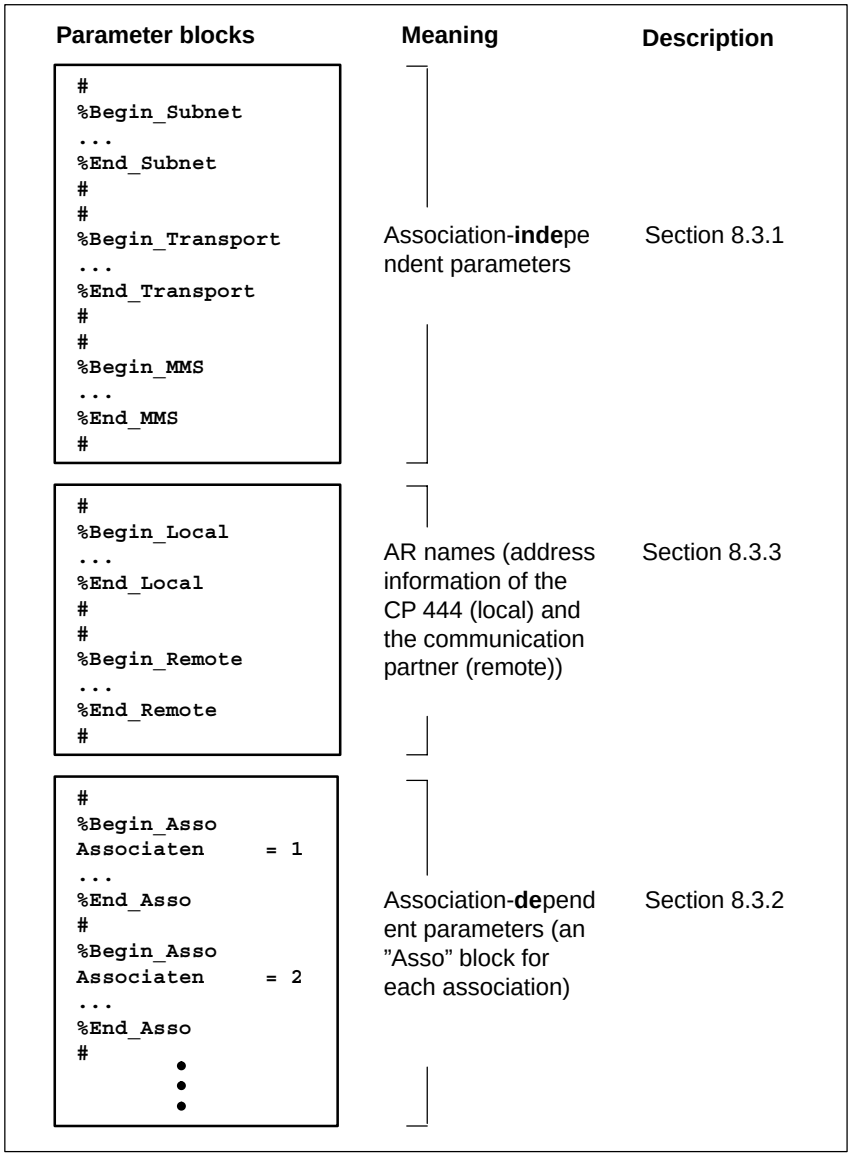


Figure 8-4 Structure of the Text File (Associations)

Structure of a Parameter Block

The following applies to each parameter block:

- Every parameter block has a fixed keyword at the beginning (**%Begin_**) and end (**%End_**).
- The keywords must be followed by the name of the parameter block (see Tables 8-1 to 8-7).
- Names are reserved for the parameters in a parameter block, by means of which they are identified. The parameter value is referenced by the equals sign (=) after the parameter name.
- Lines that begin with the number sign (#) are for entering comments.

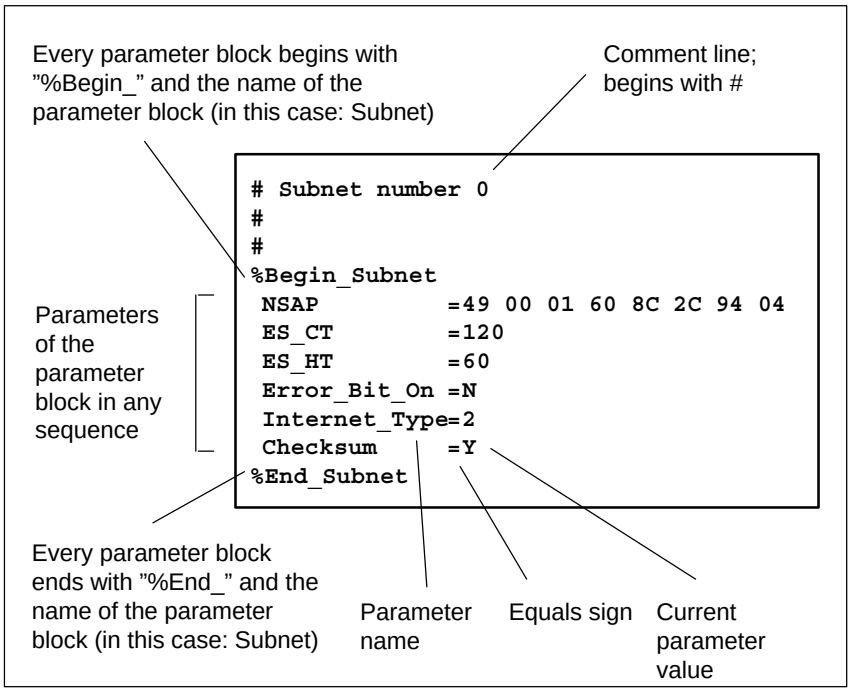


Figure 8-5 Structure of the Parameter Blocks in the Text File (Associations)

Help

Select the first line of a parameter block in the text file (e.g. "%Begin_Subnet"), and then press the F1 key. Information appears on the parameters that can be edited in this parameter block.

8.3.1 Association-Independent Parameters

Association-Independent

This section describes the parameters that must be configured globally for associations.

The following parameter blocks must be edited in accordance with the OSI layers (layers 3 to 7) of the reference model (see Section 2.1):

- Layer 3 - Network layer: ”Subnet” parameter block
- Layer 4 - Transport layer: ”Transport” parameter block
- Layer 7 - Application layer: ”MMS” parameter block

“Subnet” Parameter Block

The table below describes all the parameters in the “Subnet” parameter block.

Table 8-1 “Subnet” Parameter Block

Parameter	Meaning	Value Range	Default Value	Mandatory (M)/ Optional (O)
NSAP	NSAP assigned to the CP 444 (local). NSAP = access ID for the services of the network layer. The NSAP address identifies a CP 444 in a WAN (Wide Area Network). 49 – AFI 0001 – AID 00609716E69B – STAID (Can only occur once in the network.) 01 – NSEL	Up to 40 ASCII characters in hexadecimal format	49 00 01 00 60 97 16 E6 9B 01	M
ES_CT	ES Configuration Timer: Interval at which the end system sends its ES-Hello-PDU.	1 to 300 (s)	60	M
Error_Bit_On	Set error bit	Y/N	N	O
Internet_Type	Internet type	0: NULL 1: Not segmented 2: Full	2	O
Checksum	Use checksum	Y/N	Y	M

"Transport" Parameter Block

The table below describes all the parameters in the "Transport" parameter block.

Table 8-2 "Transport" Parameter Block

Parameter	Meaning	Value Range	Default Value	Mandatory (M)/ Optional (O)
Max_TPDU_Size	Maximum size of the transport PDU	2^7 to 2^{10} (bytes)	1024	M
Max_Retrans	Maximum number of retransmissions for a transport PDU before a disconnect request is output	1 to 19	6	M
Retrans_Time	Maximum retransmission time (T1) for a transport PDU	10 to 1000 (s) in 0.1 s units	100	M
Ack_Time	Maximum time that can elapse between the receipt of a transport PDU and the sending of a corresponding receipt acknowledgment	10 to 1000 (s) in 0.1 s units	10	M
Window_Time	Maximum time to elapse before "up-to-date window information" is transmitted	10 to 1000 (s) in 0.1 s units	100	M
Extended_Format	Suggest extended format	Y/N	Y	M
Checksum	Suggest checksum	Y/N	N	M

"MMS" Parameter Block

The table below describes all the parameters in the "MMS" parameter block.

Table 8-3 "MMS" Parameter Block

Parameter	Meaning	Value Range	Default Value	Mandatory (M)/ Optional (O)
Connect_Retrans_Time	Retransmission time for sending an MMS-Initiate-Request	10 to 1000 (s) in 0.1 s units	100	O

8.3.2 Association-Dependent Parameters

Association-Dependent

This section describes the parameters that must be configured specifically for each association of the CP 444.

Initiation Type

An association can be initiated passively or actively. There is always an ”active side” to an association (which makes the initiation request) and a ”passive side” (which accepts the initiation request).

Consequently, two types of association initiation can be configured for the CP 444:

- **Active initiation**

  The association is initiated immediately after the power is switched on or the CP 444 is started. If the connection is interrupted, the CP 444 initiates automatic reinitiation.

- **Passive initiation**

  The initiation of the association is started by the remote communication partner (e.g. by a production control computer). In other words, the CP 444 expects the connection to be set up.

You specify the initiation type by means of the ”Asso_Mode” parameter in the ”Asso” parameter block.

Status Bit

A status bit (data type: BOOL) allows you to evaluate the status of the association in the user program of the CPU.

You use the ”Asso_Flag” parameter in the ”Asso” parameter block to specify the address of the status bit.

Table 8-4 Status Bit of the Association

Value	Meaning
Status bit = FALSE	Association interrupted
Status bit = TRUE	Association created

"Asso" Parameter Block

The table below describes all the parameters in the "Asso" parameter block.

Table 8-5 "Asso" Parameter Block

Parameter	Meaning	Value Range	Mandatory (M)/ Optional (O)
Association[1]	Number of the association:	0 to 255	M
L_AR_Name	Local AR name: Defines the local end point of the association	Up to 32 characters	M
Asso_Mode	Initiation type for the association	Passive/Active	M
Asso_Flag	Status bit of the association: The Asso_Flag is only updated when the status of the association changes.	Bit address in the data block	O
MMS_Prot_Ver	Protocol used by the client (VMD) to send its PDUs The CP 444 supports both MMS-IS and MMS-DIS.	IS/DIS	These parameters are only mandatory when the "Asso_Mode = Active" initiation type is parameterized.
L_AR_Name	Remote AR name: Defines the remote end point of the association	Up to 32 characters	
MMS_PDU_Size	Maximum size of the MMS PDU	Up to16 KB	

1 This number is the first part of the ID at the input of the function blocks (see Chapter 9).

8.3.3 AR Names (Address Information)

AR Names

The AR names comprise the full address information (address book) in accordance with the OSI layers (layers 3 to 7) of the reference model (see Section 2.1) that are required for the initiation of the associations.

Initiation of the Associations

The initiation of an association is dependent on the initiation type selected (see Section 8.3.2) and the address information configured.

An association is initiated when the destination address information (remote entry) of the active application and the source address information (local entry) of the passive application match.

The ”active side” specifies in the initiation request its local address information (source) **and** the address information of the partner application (destination). The destination address information must be fully specified. In other words, it must have the full address information of layers 3 to 7 so that the initiation request can be delivered to and accepted by the partner application.

You specify the local address information in the ”Local” parameter block and the address information of the partner application in the ”Remote” parameter block.

“Local” Parameter Block

The table below describes all the parameters in the “Local” parameter block.

Table 8-6 “Local” Parameter Block

Parameter	Meaning	Value Range	Example	Mandatory (M)/ Optional (O)
AR_Name	Local AR name: Local end point of the association	Up to 32 characters	Local_Client	M
AP-Title	Application process title (decimal numbers separated by spaces)	Array of up to 16 integers	1 2 30 1	O
AP_InvokeID	Application process invocation ID: Freely definable ID of the type unsigned integer 32 **Note:** The entry must match the remote entry for the communication partner.	Integer	100	O
AE_Qualifier	Application entity qualifier: Type: unsigned integer 32	Integer	1	O
AE_InvokeID	Application entity invocation ID: Freely definable ID of the type unsigned integer 32 **Note:** The entry must match the remote entry for the communication partner.	Integer	1	O

Table 8-6 “Local” Parameter Block, continued

Parameter	Meaning	Value Range	Example	Mandatory (M)/ Optional (O)
PSel	Presentation selector: Access ID for the presentation service The access ID is freely definable. **Note:** For local applications of the same type (e.g. for all MMS applications), the value should be the same if these applications are addressed by AE title. If not, a different value must be configured for each application. The entry must match the remote entry for the communication partner.	Up to 32 ASCII characters in hexadecimal format	0010	M
SSel	Session selector: Access ID for the service of the session layer The access ID is freely definable. **Note:** The entry must match the remote entry for the communication partner.	Up to 32 ASCII characters in hexadecimal format	0010	M
TSel	Transport selector: Access ID for the transport service The access ID is freely definable. **Note:** The entry must match the remote entry for the communication partner.	Up to 64 ASCII characters in hexadecimal format	0010	M

“Remote” Parameter Block

The table below describes all the parameters in the “Remote” parameter block.

Table 8-7 “Remote” Parameter Block

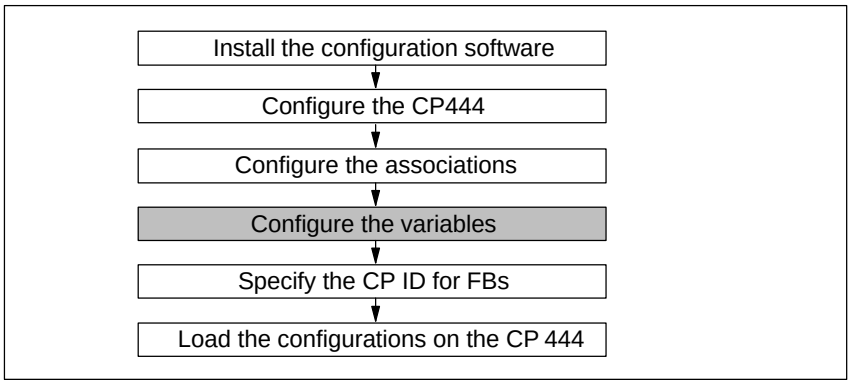
Parameter	Meaning	Value Range	Example	Mandatory (M)/ Optional (O)
AR_Name	Remote AR name: Remote end point of the association	Up to 32 characters	Remote_1	M
AP-Title	Application process title (decimal numbers separated by spaces)	Array of up to 16 integers	1 2 30 1	O
AP_InvokeID	Application process invocation ID: Freely definable ID of the type unsigned integer 32 **Note:** The entry must match the local entry for the CP 444.	Integer	100	O
AE_Qualifier	Application entity qualifier: Type: unsigned integer 32	Integer	1	O

Table 8-7 "Remote" Parameter Block, continued

Parameter	Meaning	Value Range	Example	Mandatory (M)/ Optional (O)
AE_InvokeID	Application entity invocation ID: Freely definable ID of the type unsigned integer 32 **Note:** The entry must match the local entry for the CP 444.	Integer	1	O
PSel	Presentation selector: Access ID for the presentation service The access ID is freely definable. **Note:** For remote applications of the same type (e.g. for all MMS applications), the value should be the same if these applications are addressed by AE title. If not, a different value must be configured for each application. The entry must match the local entry for the CP 444.	Up to 32 ASCII characters in hexadecimal format	0012	M
SSel	Session selector: Access ID for the service of the session layer The access ID is freely definable. **Note:** The entry must match the local entry for the CP 444.	Up to 32 ASCII characters in hexadecimal format	0012	M
TSel	Transport selector: Access ID for the transport service The access ID is freely definable. **Note:** The entry must match the local entry for the CP 444.	Up to 64 ASCII characters in hexadecimal format	0012	M
NSAP	NSAP assigned to the remote communication partner NSAP = access ID for the services of the network layer. The NSAP address identifies the remote communication partner in the WAN (Wide Area Network).	Up to 40 ASCII characters in hexadecimal format	49 00 02 08 00 06 01 B0 55 01	M
Static_Route	Recording of the static route	Y/N	N (default value)	O
SNPA	MAC address of the remote communication partner	Up to 12 ASCII characters in hexadecimal format	–	M if "Static_Route = Y"

8.4 Configuring the Variables

Where Am I?



Introduction

The CP 444 manages the variables accessible to an external communication partner (local variables). In addition, the CP 444 must know the structural information (name, data type description) of the variables of the communication partner (remote variables).

Variable configuration involves the definition of the local **and** remote variables and the assignment to the addresses of the S7 CPU.

Principle

Like the parameters for the associations, the configuration data of the variables must be stored in a text file. You generate the text file by means of the *Configuration for CP 444* parameterization interface.

The parameterization interface provides a text editor. You can use the text editor (see Figure 8-7) either to create a new text file containing data type descriptions or to open and edit an existing text file (sample file). The sample file contains a number of variable configuration examples, indicating the entries that have to be made.

Once you have finished editing the text file, you compile the text file into a specific format. This format is required to store the configuration data in a module SDB of the CP 444 (see Section 8.6).

Starting the Parameterization Interface

You start the parameterization interface by double-clicking the CP 444 in the configuration table or by selecting the CP 444 and choosing the **Edit > Object Properties** menu command.

Result: The ”Properties – CP 444” dialog box appears.

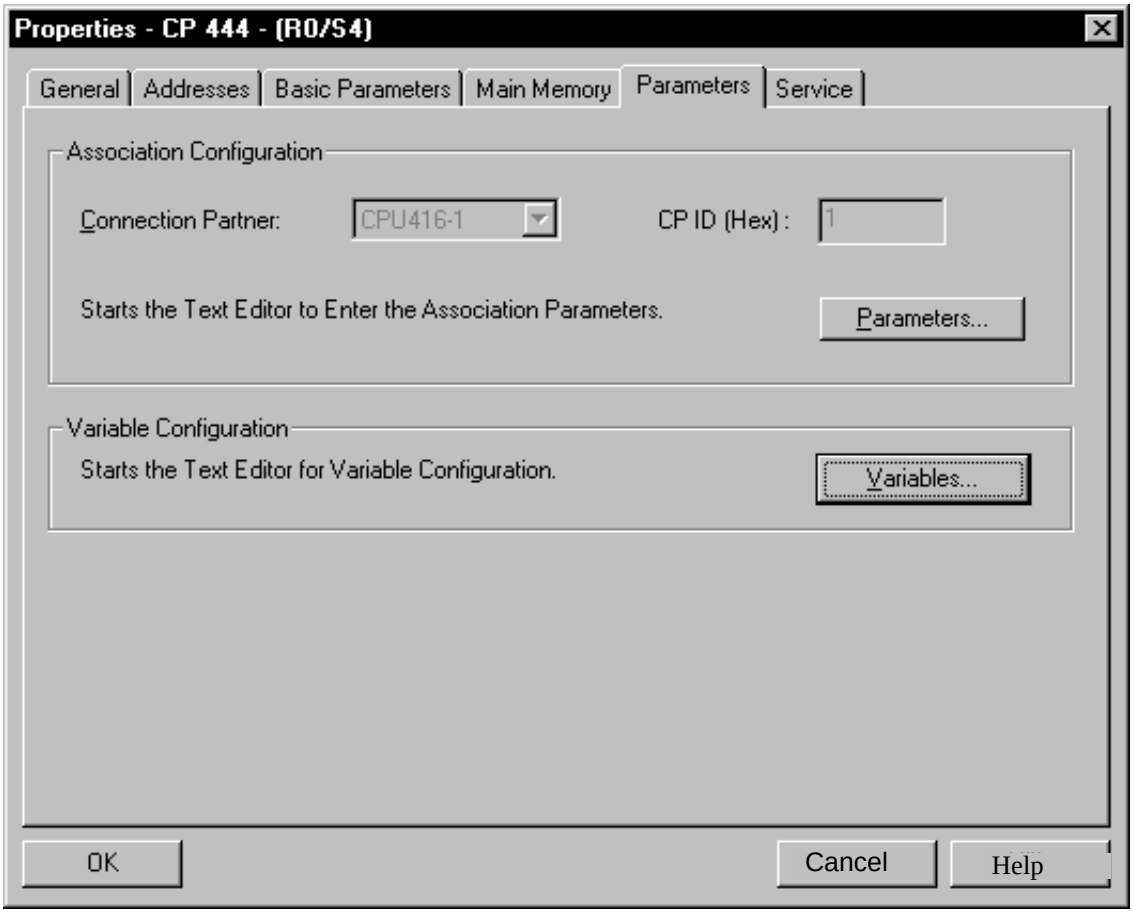


Figure 8-6 “Properties - CP 444” Dialog Box (”Parameters” Tab)

Editing a Text File

To edit the text file for the variable configuration, proceed as follows:

1. Change to the “Parameters” tab. To do this, simply click the ”Parameters” tab label near the top of the dialog box.

   Result: The ”Parameters” tab appears in the foreground (see Figure 8-6).

2. Click the “Variables” button.

   Result: The window of a text editor appears. You use this text editor to create the text file (Extension .var) containing the configuration data for the variables.

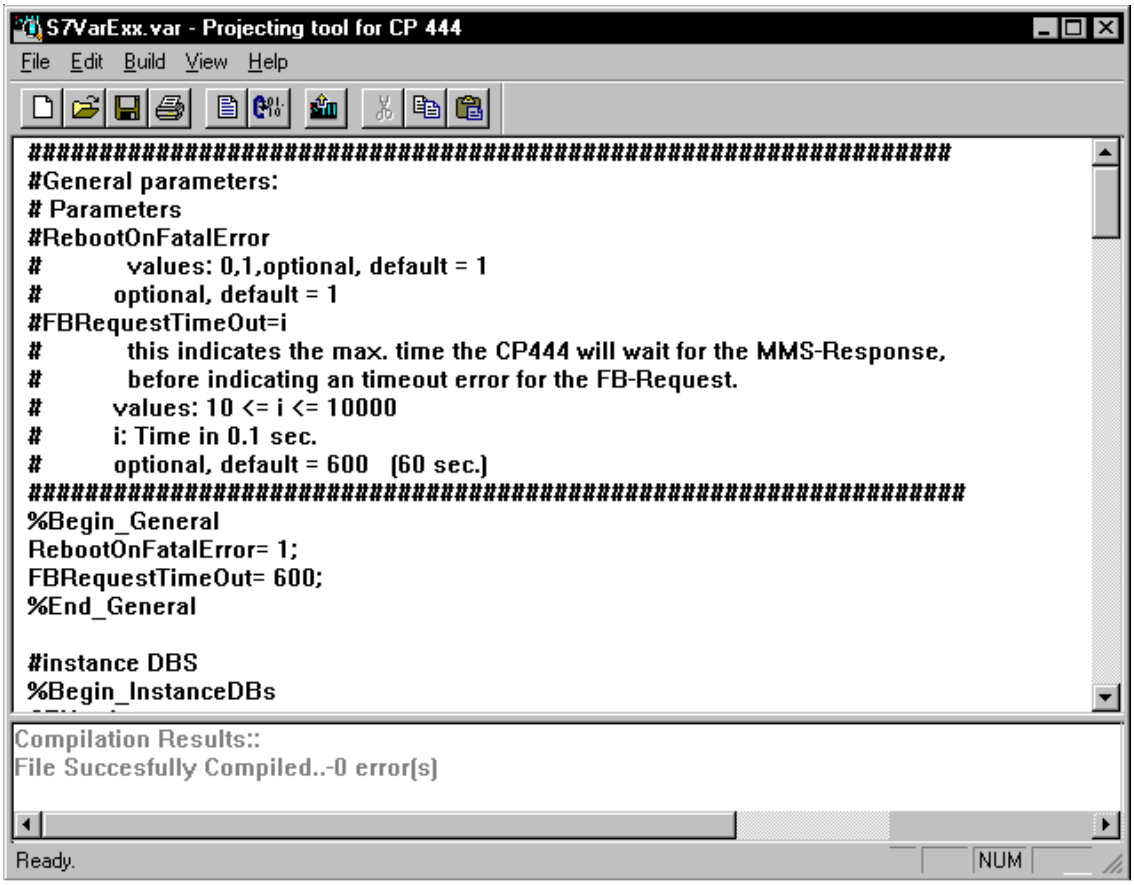


Figure 8-7 "Name.var - Configuration for CP 444" Dialog Box (Variables)

3. Choose **File ▸ Load Example** to open the sample file.

   Result: The contents of the sample file appear with examples for different variable types.

4. Configure the variables of your associations as in the examples provided:
   - Structure of the variable parameters (see Figures 8-8 to 8-11)
   - Purpose of the parameters and possible values (see Tables 8-10 to 8-13)
5. Choose **File > Save As** to save the entries in another text file.
6. After you have entered all the configuration data, choose **Build > Compile**.

   Result: The current (open) text file is converted to the format required to create the module SDB.

Incorrect Entries

If you make incorrect entries in the text file, error messages are generated and displayed in the lower part of the text editor window (see Figure 8-7). If you **double-click an error message**, the relevant line in the text file is displayed and selected. Not until all the errors have been eliminated is the text file converted to the SDB-specific format.

Structure of the Text File

The variable parameters are assigned to different parameter blocks. The structure of the parameter blocks in the text file is as follows:

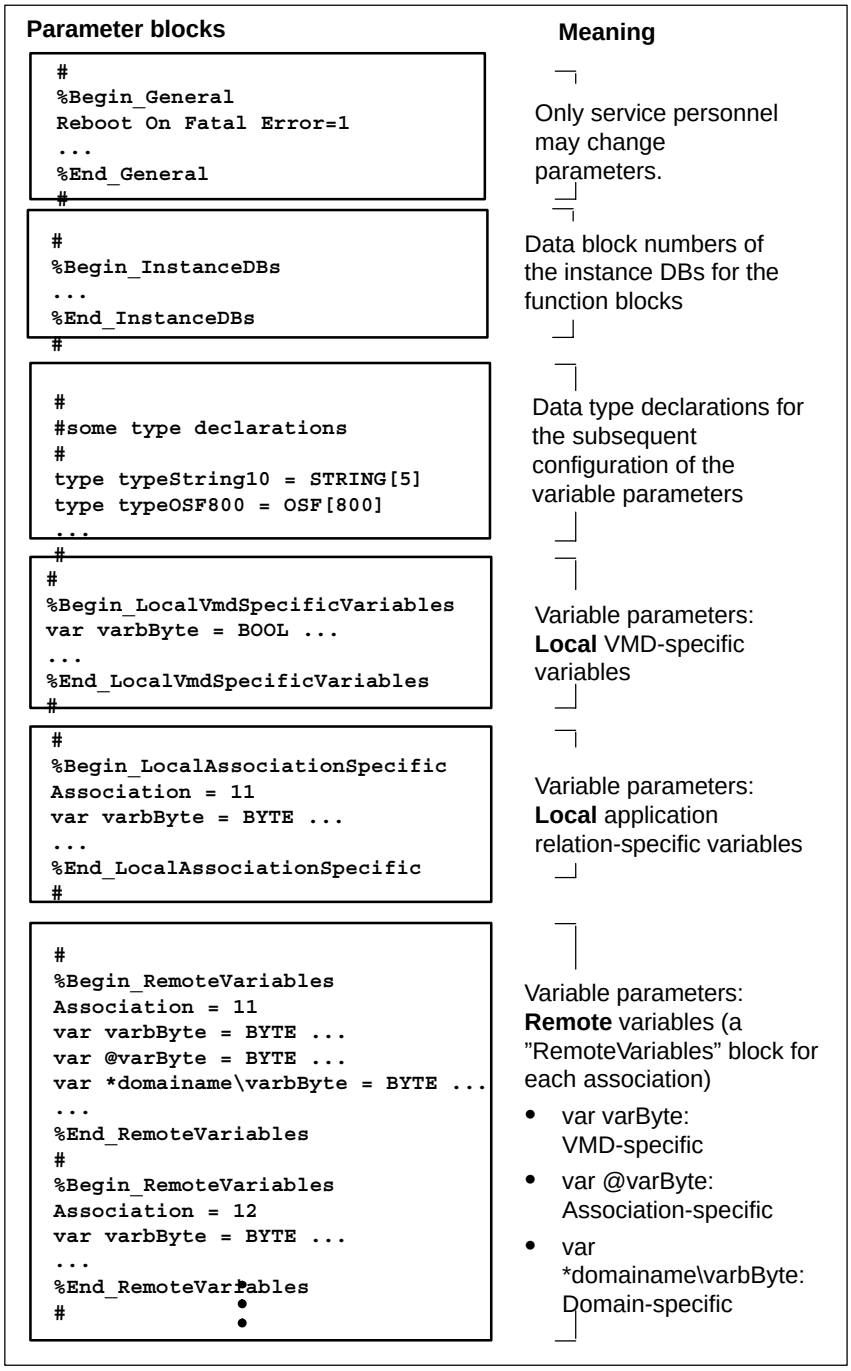


Figure 8-8 Structure of the Text File (Variables)

Help

Select the first line of a variable block (e.g. "%Begin_RemoteVariables"), and then press the F1 key. Information appears on the parameters that can be edited in this variable block.

8.4.1 Numbers of the Instance DBs

Introduction

In the **%Begin_InstanceDBs** block you tell the CP 444 the data block numbers of the instance DBs of the function blocks for the MMS services. These block numbers must match what is specified for the FB call in the user program of the CPU (see Chapter 9).

You can edit this block at any point in the text file. The numbers of the instance DBs are at the beginning of the sample text file.

Structure

Figure 8-9 shows you the structure of the “InstanceDBs” block in the text file:

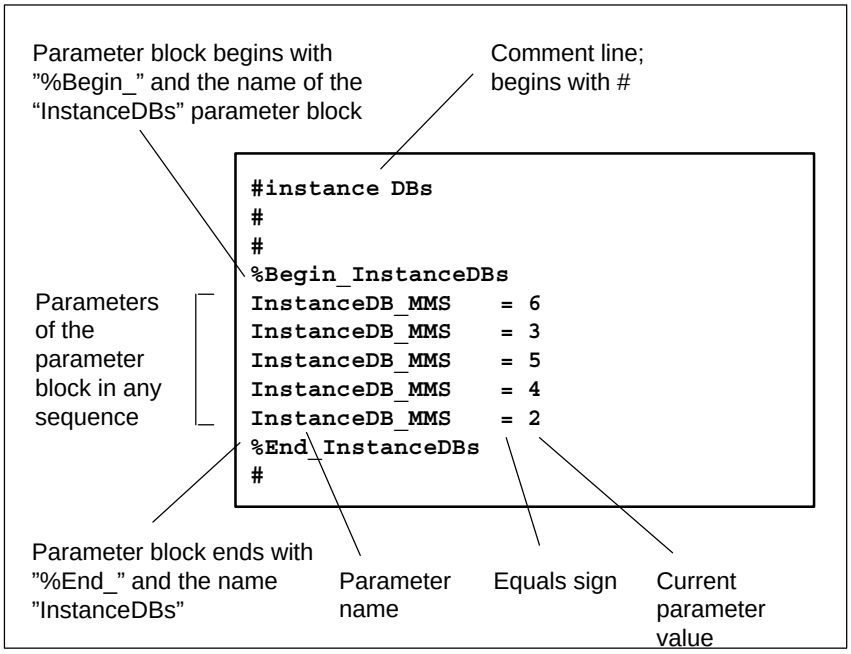


Figure 8-9 Structure of the ”InstanceDBs” Block (Variables)

Parameters

Table 8-8 describes the configuration data:

Table 8-8 Numbers of the Instance DBs

Name	Meaning	Value Range	Example (See Figure 8-9)
InstanceDB_MMS	You specify this parameter for every function block. If a function block of the same type (e.g. READ FB) is called several times in the user program, specify this parameter for each READ FB.	Number of the instance data block (CPU-specific) A maximum of 32Instance DBs can be configured	6

8.4.2 Data Type Declarations

Introduction

Before you enter a variable block with data type descriptions, all the data types used in the variable block must be declared in the text file **before** the variable block.

There is no variable block for the data type declarations. The declarations can be anywhere in the text file except in a variable block. The declarations are valid for all the subsequent variable blocks (local or remote).

Structure

Figure 8-10 shows the structure of the data type declarations in the text file:

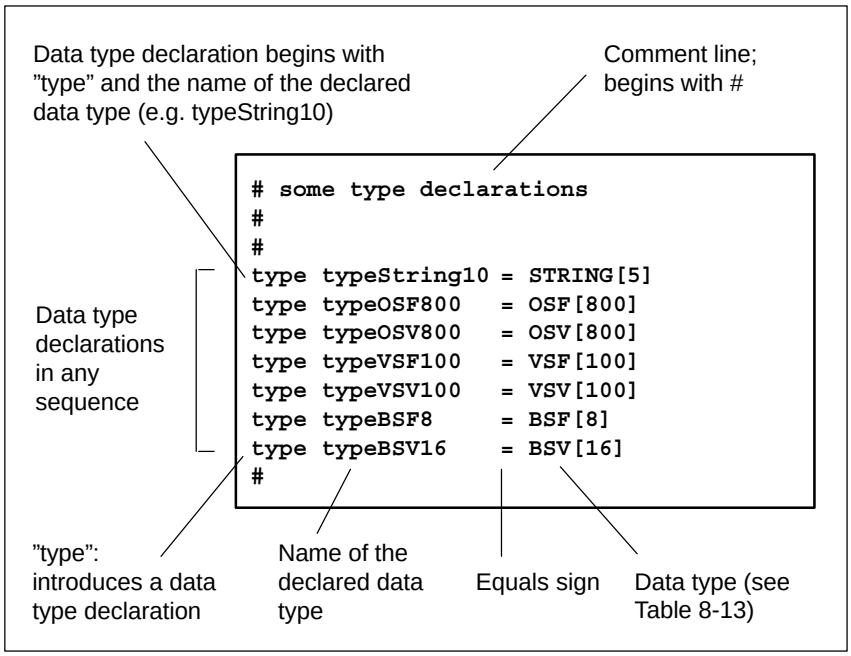


Figure 8-10 Structure of the Data Type Declarations (Variables)

Parameters

Table 8-9 describes the configuration data. **Note** that only the complex data types (see Table 8-13) have to be declared in the text file. The standard data types (see Table 8-12) can be used in the variable blocks without being declared.

Table 8-9 Data Type Declarations

Name	Meaning	Value Range	Example (See Figure 8-10)
type <name>	For <name> you enter the name of the data type declared (32 characters).	Complex data types (see Table 8-13)	STRING[5]

8.4.3 Variable Parameters

Introduction

You enter the variable parameters in variable blocks. The following variable blocks can be edited:

- **`%Begin_LocalVmdSpecificVariables`**

  Configuration of all local VMD-specific variables

- **`%Begin_LocalAssociationSpecificVariables`**

  Configuration of all local association-specific variables

- **`%Begin_RemoteVariables`**

  Configuration of all remote variables (VMD-specific, association-specific and domain-specific)

Structure of a Variable Block

The following applies to each variable block:

- Before you enter a variable block with data type descriptions, all the data types used in the variable block must be declared in the text file **before** the variable block (see Section 8.4.2).

Note

Only the complex data types (see Table 8-13) have to be declared in the text file. The standard data types (see Table 8-12) can be used in the variable blocks without being declared.

- The block containing the declared data types is followed by the variable blocks containing the actual variable parameters.
- Every variable block has a fixed keyword at the beginning (**%Begin_**) and end (**%End_**).
- The keywords must be followed by the name of the variable block (see above).
- Names are reserved for the parameters in a variable block, by means of which they are identified. The parameter value is referenced by the equals sign (=) after the parameter name.
- A line is reserved for each variable. The parameters of a variable are separated from each other by the ”|” character.
- Lines that begin with the number sign (#) are for entering comments.

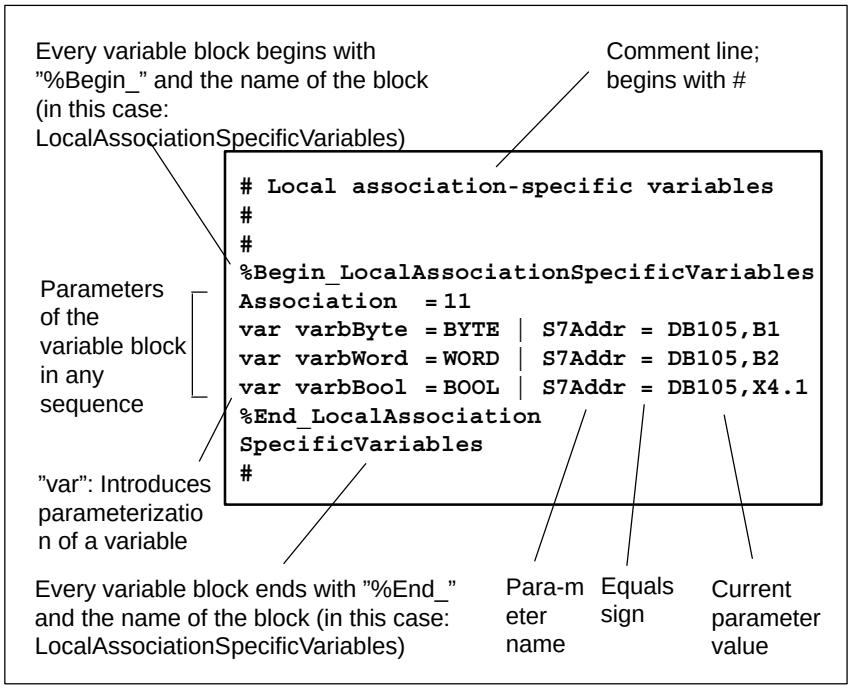


Figure 8-11 Structure of the Variable Blocks (Variables)

Variable Parameters (Local)

The table below describes the configuration data for local variables.

Table 8-10 Variable Parameters (Local)

Name	Meaning	Value Range	Examples
Association[1]	Number of the association:	0 to 255	11
var <name>	For <name> you enter the name of the variables (32 characters).	Name of the data type you declared (see Figure 8-10) or Name for standard data type (see Table 8-12)	typeString10 or BYTE
S7Addr	Destination or source address of the variable value (data block number DBx and byte address By). See also Section 7.5, "Addressing Local and Remote Variables"	DBx,By (CPU-specific)	DB105,B1
Access	Access coordination: • UR /UW = Uncoordinated read and write (no ACCESS FB) • CR /CW: Coordinated read and write (ACCESS FB use)	• UR /UW[2] • CR /CW[2]	UR /UW
AccessCtrl (optional)	Shows the byte address of the status bits in the access DB during CR/CW access.	Bx	B1

1 This parameter occurs only in the "%Begin_LocalAssociationSpecificVariables" variable block. It does not occur in the "%Begin_LocalVmdSpecificVariables" variable block.

2 Combinations, such as UR/CR or UW/CW, for example, are not possible.

Variable Parameters (Remote)

The table below describes the configuration data for remote variables. VMD-specific, association-specific and domain-specific are distinguished from each other by means of the prefixes "@" and *domainame" before the actual variable names.

Table 8-11 Variable Parameters (Remote)

Name	Meaning	Value Range	Examples
Association	Number of the association:	0 to 255	11
var <name>	For <name> you enter the name of a VMD-specific variable (32 characters).	Name of the data type you declared (see Figure 8-10) or Name for standard data type (see Table 8-12)	typeString10 or BYTE
var @<name>[1]	For <name> you enter the name of an association-specific variable (32 characters).		
var *domainame\<name>	For <name> you enter the name of a domain-specific variable (32 characters).		
S7Addr	Destination or source address of the variable value (data block number DBx and byte address By). See also Section 7.5, "Addressing Local and Remote Variables"	DBx,By (CPU-specific)	DB105,B1

1 deutsch: Um Applikationsbeziehungs-spezifische Variablen im Variablen–DB zudeklarieren, muß das Zeichen "@" in den DB-Editor mit Copy und Paste aus einem beliebigen Text eingefügt werden. Die direkte Eingabe über die Tastatur ist hier nicht möglich.

Data Types (Standard Types)

The table below describes standard data types supported by the CP 444. These do **not** have to be declared in the text file beforehand.

Table 8-12 Standard Data Types

Value in Text File	Corresponding MMS Data Type	Corresponding Data Type in S7	Comment
BOOL	Boolean	BOOL	Boolean: FALSE = ”0”, TRUE = ”1”
BYTE	Unsigned8	BYTE	Unsigned integer: 8 bits
WORD	Unsigned16	WORD	Unsigned integer: 16 bits
DWORD	Unsigned32	DWORD	Unsigned integer: 32 bits
INT	Integer16	INT	Integer: 16 bits
DINT	Integer32	DINT	Integer: 32 bits
CHAR	Fixed Visible String	CHAR	Printable character deutsch: (7-Bit-ASCII ohne Bereich 31 bis 127)
REAL	Floating Point	REAL	Floating-point number

Data Types (Structures)

The table below describes complex data types supported by the CP 444. These have to be declared in the text file before the variable blocks.

Table 8-13 Complex Data Types

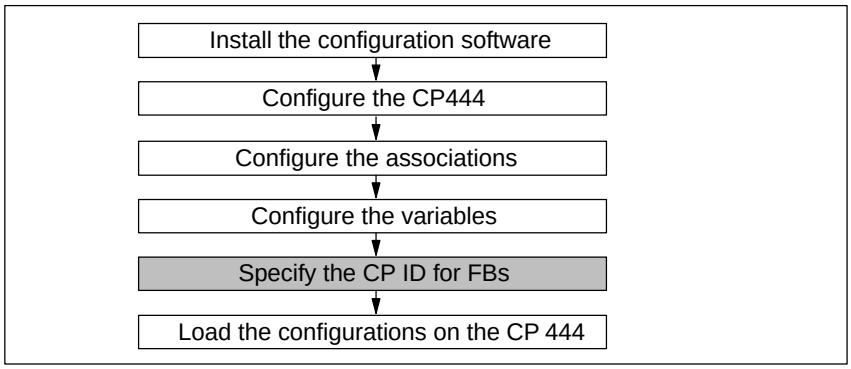
Value in Text File	Corresponding MMS Data Type	Corresponding Data Type in S7	Comment
STRING[len]	Visible String (variable)	STRING[len]	[len] is the maximum number of bytes in the string.
OSF[len]	Octet String (fixed)	ARRAY[1..len] BYTE	[len] is the maximum number of bytes in the string.
OSV[len]	Octet String (variable)	STRUCT len WORD data ARRAY[1..len] BYTE END_STRUCT	[len] is the maximum number of bytes in the string. The first word contains the current number.
VSF[len]	Visible String (fixed)	ARRAY[1..len] CHAR	[len] is the maximum number of bytes in the string. deutsch: Nur druckbare Zeichen (7-Bit-ASCII ohne Bereich 31 bis 127)
VSV[len]	Visible String (variable)	STRUCT len WORD data ARRAY[1..len] CHAR END_STRUCT	[len] is the maximum number of bytes in the string. The first word contains the current number. deutsch: Nur druckbare Zeichen (7-Bit-ASCII ohne Bereich 31 bis 127)
BSF[len]	Bit String (fixed)	ARRAY[1..len] BOOL	Any number of bits. [len] is the maximum number of bits.

Table 8-13 Complex Data Types, continued

Value in Text File	Corresponding MMS Data Type	Corresponding Data Type in S7	Comment
BSV[len]	Bit String (variable)	STRUCT len WORD data ARRAY[1..len] BOOL END_STRUCT	Any number of bits. The first word contains the current length. [len] is the maximum number of bits.
ARRAY[len]	Array	ARRAY[1..len]	An array is a field of elements of the same type. The number of elements must be specified.
{...}	Structur	STRUCT ... END_STRUCT	Structures can contain different components (e.g. IN, UN ..). Each of the components in the structure must be specified individually. The data types OSV, VSV and BSV must be the last item in structures.

8.5 Specifying the CP ID

Where Am I?



Introduction

The CP ID is specified automatically by the *Configuration for CP 444* parameterization interface at parameterization of the associations and displayed in the ”CP ID” file on the ”Parameters” tab (see Figure 9-1).

The CP ID is the second part of the connection ID. You have to reference the connection ID in the user program of the CPU at the ”ID” input of the function blocks for the execution of MMS services (see Chapter 9).

Defining the CP ID

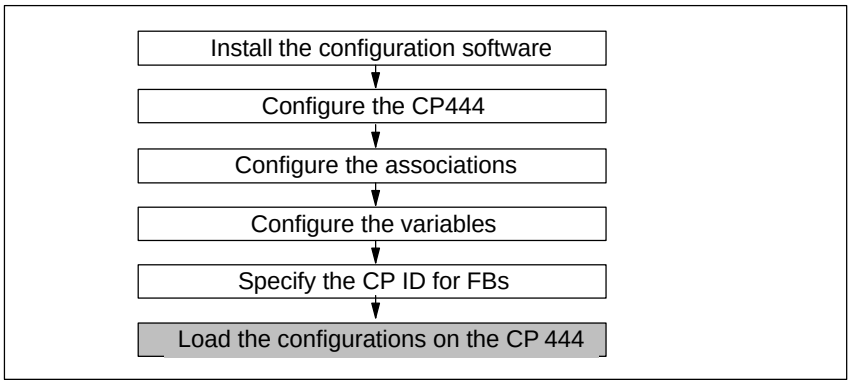
To create the CP ID, proceed as follows:

1. In the ”Connection Partner” list box on the ”Parameters” tab, select the CPU to which you want to create a K bus connection.
2. Click the “Parameters” button.
   Result: The window of the text editor for parameterizing the associations (described in Section 8.3) appears.
3. Once you have completed parameterization, choose **File > Exit** to terminate the text editor.
   Result: The current CP ID of the CP 444 appears in the ”CP ID” field. ”CP ID” and ”Connection Partner” can no longer be changed in the “Parameters” tab.
4. Close the configuration table, and save the configuration data in your project.
5. In SIMATIC Manager, open the connection table of the CPU.
   Result: The K bus connection is visible in the connection table. You can change the CP ID preset by the parameterization interface in the ”Local ID (Hex)” field or delete the K-bus connection.
6. Choose the **Connection Table > Save and Compile** menu command.
7. Set the ”Compile and Test All” option, and then click OK.

The parameters for the K bus connection between the CPU and the CP 444 have thus been created in the correct format and can be loaded on the CPU.

8.6 Loading the Configuration Data

Where Am I?



Introduction

The configuration and parameterization data of the CP 444 is stored in the current project (on the hard disk of the programming device/PC).

Data Management

When you exit the configuration table (see Section 8.2) and the connection table (see Section 8.5), all the configuration and parameterization data is stored automatically in the STEP 7 project you have created.

Loading the Configuration Data

You can now use *STEP 7* to load the configuration and parameterization data on-line from the programming device to the CPU and the CP 444 (**PLC > Download**) menu command). The CPU and the CP 444 accept the parameters immediately they are loaded.

The LEDs on the front of the CP 444 indicate that a load operation is in progress (see Section 11.2).

Note

If the CPU is to remain in RUN mode when the configuration data is loaded, the insert/remove OB (OB 83) must be loaded on the CPU.

Reading the Configuration Data

You can display on the programming device/PC the current parameterization data on the CP 444. To do this, choose the **File > Load Parameters** menu command in the *Configuration for CP 444* parameterization interface. The CP 444 loads the current parameters immediately and displays them in the form of the familiar text file (see Section 8.3 and 8.4).

Additional Information

In the STEP 7 user manual /5/ you will find a detailed description of how to do the following for modules connected to a programmable controller:

- Save the configuration and parameters.
- Load the configuration and parameters.
- Read, modify and copy the configuration and parameters.

/5/ *Standard Software for S7 and M7, STEP 7* User Manual

Programming the Function Blocks for MMS Services

9

Purpose of This Chapter

Once you have read this chapter, you will have all the information you require to program the function blocks of the CP 444:

- The purpose and properties of the function blocks
- The parameters of the blocks and what they are for
- Commented function block calls

Introduction

The CP 444 configuration package includes a number of function blocks by means of which you can initiate MMS services on an event-driven basis using the user program of the CPU involved.

The function blocks are already programmed; you need only to call and parameterize them in the user program.

Chapter Overview

This chapter is divided into the following sections:

9.1 Overview of the Function Blocks and Conventions

Introduction

The function blocks described below are part of the CP 444 configuration package. Once the configuration software is installed, you will find them in the **CP444** library of SIMATIC Manager.

Installation

The installation of the CP 444 configuration software is described in Section 8.1.

Function Blocks

The table below lists the function blocks for the CP 444 and indicates what they are for.

Table 9-1 Function Blocks for MMS Services

FB	Name	Purpose
FB 1	ACCESS	Sets access rights of local variables
FB 2	IDENTIFY	Identifies the remote communication partner
FB 3	READ	Reads remote variables
FB 4	REPORT	Reports local variables to the remote communication partner without acknowledgment
FB 5	STATUS	Reads status information on the remote communication partner
FB 6	WRITE	Writes remote variables
FB 7	ABORT	Aborts an association

Memory Allocation

To give you optimal utilization of the user memory of your CPU, the READ, REPORT and WRITE function blocks have different memory allocations. Please always use the function block that only allocates as much memory as you require to transfer the user data.

The function blocks are in different folders in the **CP444** library in accordance with the memory allocation. The table below gives the names of the folders and the maximum volume of user data of the function blocks that can be transferred.

Table 9-2 Volume of User Data of the Function Blocks that Can Be Transferred

Name of the Container	User Data that Can Be Transferred
CP 444-235	235 bytes
CP 444-4096	4096 bytes

Working with the Library

You open the **CP444** library in *STEP 7* SIMATIC Manager by choosing the **File** > **Open** > **Library** menu command. When working with the function blocks, you need only copy the required function block to your project's CPU program.

Abbreviations for Memory Areas

The following abbreviations are used for the memory areas for the function blocks described in this chapter:

Abbreviation	Type
I	Input
O	Output
M	Memory marker
L	Local data
D	Data block area
T	Timer
C	Counter

Connection ID

The connection ID must be specified for the ”ID” parameter for the function blocks described in this chapter. The connection ID references the local and remote communication partner. The connection ID consists of two parts, which you specify when you parameterize the CP 444. Figure 9-1 shows how the connection ID is made up.

Please note that the CP ID cannot be changed dynamically since it is evaluated once only, at CPU startup (see also the chapter entitled ”Communication SFBs for Configured Connections” in /6/).

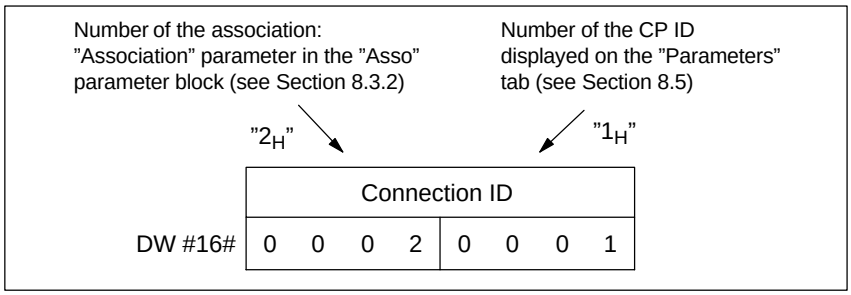


Figure 9-1 Composition of the Connection ID

Note

Only the CP_ID is required for the FB ACCESS function block.

Status and Error Messages

The structure of the status and error messages for the function blocks described in this chapter is identical. You will find explanations of the output parameters and an overview of the error messages in Section 9.9.
If the status message “W#16#1204” (“Asso Queue Overflow” see section 9.9) occurs for data types which are sent very quickly to the CP444 e.g. data type “Bool”, the call up frequency for the function block used should be lowered.

/6/ Reference Manual: *System Software for S7-300 and S7-400, System and Standard Functions*

9.2 ACCESS Function Block

Introduction

The ACCESS function block allows you from the user program to block and release the access rights to variables. You can also determine whether the communication partner has edited the variable or is currently editing it.

By assigning the access rights selectively you can coordinate the access to variables and achieve consistent data interchange. You define the access rights in a DB (ACCESS-DB).

Coordinated Access

To ensure coordinated access you control access to variables by blocking or releasing read or write access temporarily.

In this way, the server end can, for example, protect a data record from being read again until a specific control sequence has been processed in the user program.

Consistent Transfer

In order to ensure consistent data transfer, the user program cannot update/edit variables until the control bit DONE (in ACCESS-DB) is set.

This makes it impossible, for example, for the partner to write or read incompletedata areas.

Important to Note

The ACCESS function block must be called at least once in each cycle of the user program.

Requirement

The instance DB assigned to the ACCESS FB is declared.

Block Diagram of the ACCESS FB

The block diagram shows the call interface of the FB ACCESS.

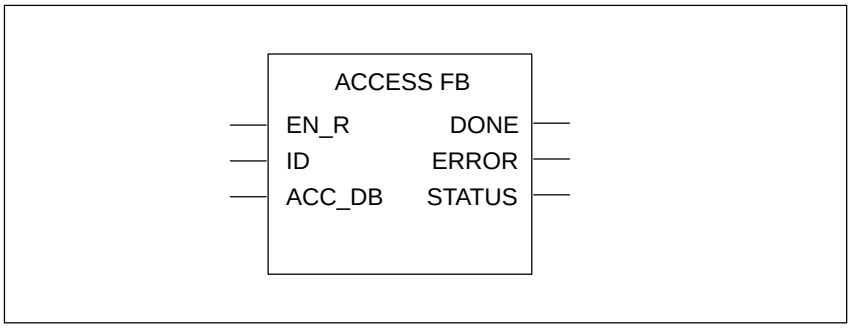


Figure 9-2 Block Diagram of the ACCESS FB

ACCESS FB Parameters

You will find the parameters of the ACCESS FB in the table below.

Table 9-3 ACCESS FB Parameters

Parameter	Declaration	Data Type	Storage Area	Description
EN_R	INPUT	WORD	E, A, M, D, L	There is no edge evaluation. TRUE can be specified.
ID	INPUT	WORD	E, A, M, D, L	Number of the **CP-ID** (see Section 9.1)
ACC_DB	INPUT	WORD	E, A, M, D, L	Number of the data block used
DONE	OUTPUT	BOOL	E, A, M, D, L	The bit is set by the CPU when the request has been processed without error.
ERROR	OUTPUT	BOOL	E, A, M, D, L	The bit is set by the CPU when an error has occurred.
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are described (see Section 9.9)

Specifying Access Rights in the ACCESS-DB

You can specify the access right for each variable. All you need to do is define the read and write rights (READ/WRITE) and define two control bits (IN_USE/DONE) in the so-called ACCESS-DB. You reference the block number of the ACCESS-DB at the “ACC_DB” input parameter of the ACCESS FB.

Table 9-4 Overview of the Parameters in the ACCESS-DB

Parameter	Data Type	Meaning
READ	BOOL	TRUE = reading permitted
WRITE	BOOL	TRUE = writing permitted
IN_USE	BOOL	TRUE = partner currently processing the variable
DONE	BOOL	TRUE = partner has completed processing the variable
READ_DONE	BOOL	TRUE = reading completed
WRITE_DONE	BOOL	TRUE = writing completed
INFOREP_DONE	BOOL	TRUE = transmission completed

Behavior at Restart of the CPU

If the data transfer is interrupted by a restart of the CPU (RUN → STOP → RUN), the IN_USE control bit or bits in the ACCESS-DB is/are not automatically reset. In case there is a restart of the CPU, you should therefore always set the IN_USE control bit(s) to FALSE in the user program.

Block Call, Example in STL

The following program sequence shows a block call with appropriate parameterization. Please note that the ACCESS FB must be called at least once in each cycle of the user program.

```
AWL                                  Erläuterung
call FB 1, DB 22                     ACCESS block call with instance DB
(
EN_R     := M 1.0                    Release signal for variable accesses
ID       := W#16#1                   Connection ID matched with configuration
ACC_DB   := W#16#5                   Addresses the ACCESS DB = DB 5
DONE     := M 1.1                    Confirmation of execution
ERROR    := M 1.2                    Indicates error during execution
STATUS   := MW 20);                  Detailed error decoding
```

ACCESS-DB, Example

The following example shows the definition of the access right in the ACCESS-DB for the variables MOTOR1 and MOTOR2.

Table 9-5 Definition of the Access Rights in the ACCESS-DB

Address	Name	Type	Initial Value	Comment
0.0		STRUCT		
+0.0	MOTOR1	STRUCT		Name of the variable
+0.0	read	BOOL	TRUE	Read access right
+0.1	write	BOOL	FALSE	Write access right
+0.2	in_use	BOOL	FALSE	In processing
+0.3	done	BOOL	FALSE	Processing completed
+0.4	read_done	BOOL	FALSE	Reading completed
+0.5	write_done	BOOL	FALSE	Writing completed
+0.6	Inforep_done	BOOL	FALSE	Transmission completed
=2.0		END_STRUCT		
+2.0	MOTOR2	STRUCT		Name of the variable
+0.0	read	BOOL	FALSE	Read access write
+0.1	write	BOOL	FALSE	Write access write
+0.2	in_use	BOOL	FALSE	In processing
+0.3	done	BOOL	FALSE	Processing completed
+0.4	read_done	BOOL	FALSE	Reading completed
+0.5	write_done	BOOL	FALSE	Writing completed
+0.6	Inforep_done	BOOL	FALSE	Transmission completed
=2.0		END_STRUCT		
=4.0		END_STRUCT		

9.3 IDENTIFY Function Block

Introduction

The IDENTIFY function block allows you to get the following information from the partner device:

- The name of the vendor
- The name of the model
- The revision of the device

Depending on this information you can, for example:

- Set the local program function for the performance and behavior of the communication partner
- Arrange/organize the required parameters in the user program

Requirement

The instance DB assigned to the IDENTIFY FB is declared.

IDENTIFY FB Block Diagram

The block diagram shows the call interface of the IDENTIFY FB.

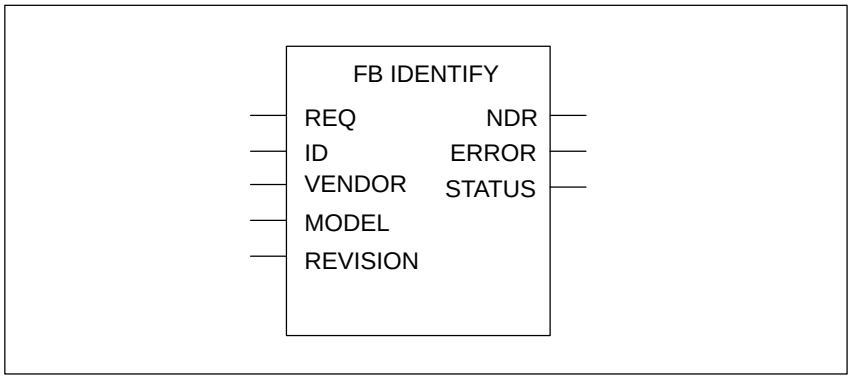


Figure 9-3 Block Diagram of the IDENTIFY FB

IDENTIFY FB Parameters

The table below lists the parameters of the IDENTIFY FB.

Table 9-6 Parameters of the IDENTIFY FB

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	E, A, M, D, L	Bit that activates the processing of the job at a positive edge
ID	INPUT	DWORD	E, A, M, D, L	Connection ID: Number by means of which the connection is uniquely identified (see Section 9.1).
NDR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the recipient when the job is processed without errors
ERROR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU if an error occurs
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are referenced (see Section 9.9).
VENDOR	INOUT	STRING[64]	D	The name of the vendor
MODEL	INOUT	STRING[16]	D	The name of the model
REVISION	INOUT	STRING[16]	D	The revision of the device

Block Call; Example in STL

The following program sequence shows a block call with corresponding parameterization.

```
STL                                  Explanation
call FB 2, DB 22                     IDENTIFY block call with instance DB
(
REQ      := M 1.0                    Edge signal for the execution of the FB
ID       := DW#16#10001              Connection ID compared with configuration
NDR      := M 1.1                    Indicates when new data has been received
ERROR    := M 1.2                    Indicates errored execution
STATUS   := MW 20                    Detailed error decoding
VENDOR   := "SLAVE2".VENDOR_ABBILD   Data area for the name of the vendor
MODEL    := "SLAVE2".MODEL_ABBILD    Data area for the model
REVISION := "SLAVE2".REV_ABBILD      Data area for the revision
          );
```

```
Additional information

"SLAVE2"
is the symbolic name of a data block. This name is defined in the associated symbol
table.
VENDOR_ABBILD, MODEL_ABBILD and REVISION_ABBILD
are variables of the STRING data type. These are defined in the "SLAVE2" data block.
```

9.4 READ Function Block

Introduction

The READ function block reads the data of a variable from a specified data area of the communication partner. The data read is stored locally in a data block, an area in the process image of the inputs/outputs or in a bit memory address area.

Requirements:

... for calling the READ FB in the user program:

- The instance DB assigned to the READ FB is declared.
- The variable is configured as a remote variable with the CP 444 (see Section 8.4).

READ FB Block Diagram

The block diagram shows the call interface of the READ FB.

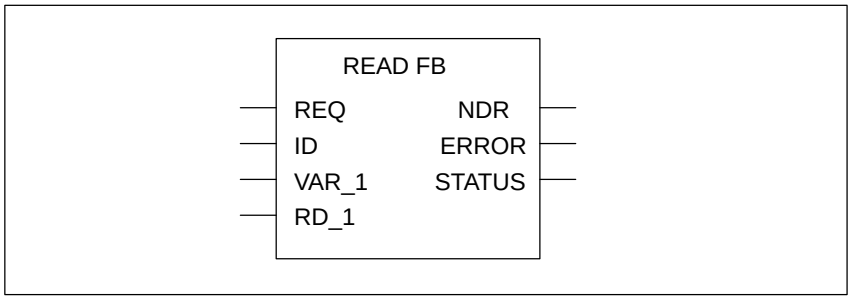


Figure 9-4 Block Diagram of the READ FB

READ FB Parameters

The table below lists the parameters of the READ FB.

Table 9-7 Parameters of the READ FB

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	E, A, M, D, L	Bit that activates the processing of the job at a positive edge
ID	INPUT	DWORD	E, A, M, D, L	Connection ID: Number by means of which the connection is uniquely identified (see Section 9.1).
VAR_1	INPUT	ANY	E, A, M, D, L	Pointer to the name of a variable to be read
RD_1	INPUT	ANY	E, A, M, D, L, T, Z	Pointer to the address of a variable to be read Example: P#DB17.DBX0.0 BYTE 16
NDR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the recipient when the job is processed without errors
ERROR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU if an error occurs
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are described (see Section 9.9).

Block Call; Example in STL

The following program sequence shows a block call with corresponding parameterization.

```
STL                                 Explanation
call FB 3, DB 29                    READ block call with instance DB
(
REQ     :=      M 1.0               Edge signal for the execution of the FB
ID      :=      DW#16#10001         Connection ID compared with configuration
VAR_1   :=      "SLAVE2".INDEX      Addresses communication variable to be read
RD_1    :=      "PROZESS".ABBILD    Addresses data area to receive transferred data
NDR     :=      M 1.1               Execution confirmation
ERROR   :=      M 1.2               Indicates errored execution
STATUS  :=      MW 20               Detailed error decoding
                );
```

```
Additional information

"SLAVE2".INDEX
is the symbolic name of a structural element of a data area defined and stored as a
communication variable on the communication partner.

"PROZESS".ABBILD
is a locally declared variable in the "PROZESS" DB that contains the value read as the
destination data area.
```

9.5 REPORT Function Block

Introduction

The REPORT function block allows a server to transfer variables unacknowledged.

The structure of the variables must be defined at configuration.

Requirements:

... for calling the REPORT FB in the user program:

- The instance DB assigned to the REPORT FB is declared.
- The variable is configured as a local variable with the CP 444 (see Section 8.4).

REPORT FB Block Diagram

The block diagram shows the call interface of the REPORT FB.

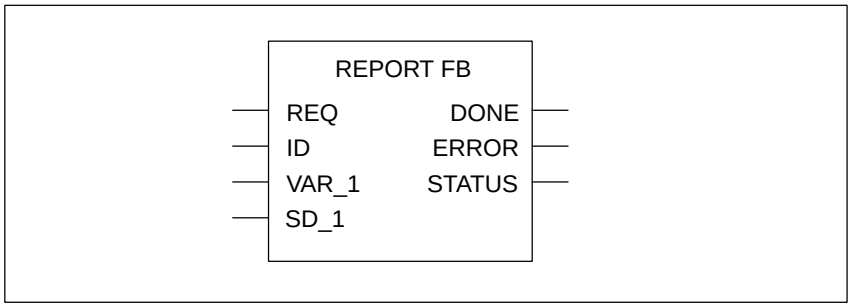


Figure 9-5 Block Diagram of the REPORT FB

REPORT FB Parameters

The table below lists the parameters of the REPORT FB.

Table 9-8 Parameters of the REPORT FB

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	E, A, M, D, L	Bit that activates the processing of the job at a positive edge
ID	INPUT	DWORD	E, A, M, D, L	Connection ID: Number by means of which the connection is uniquely identified (see Section 9.1).
VAR_1	INPUT	ANY	E, A, M, D, L	Pointer to the name of a variable to be reported
SD_1	INPUT	ANY	E, A, M, D, L, T, Z	Pointer to the address of a variable to be reported Example: P#DB17.DBX0.0 BYTE 16
DONE	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU when the job is processed without errors
ERROR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU if an error occurs
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are described (see Section 9.9).

Block Call; Example in STL

The following program sequence shows a block call with corresponding parameterization.

```
STL                                 Explanation
call FB 4, DB 28                    REPORT block call with instance DB
(
REQ     :=      M 1.0               Edge signal for the execution of the FB
ID      :=      DW#16#10001         Connection ID compared with configuration
VAR_1   :=      "SLAVE2".INDEX      Addresses communication variable to be read
SD_1    :=      "SLAVE2".VAR_54     Addresses data area to send transferred data
DONE    :=      M 1.1               Execution confirmation
ERROR   :=      M 1.2               Indicates errored execution
STATUS  :=      MW 20);             Detailed error decoding
```

```
Additional information

"SLAVE2".INDEX
is the symbolic name of a structural element of a data area defined and stored as a
communication variable on the communication partner.

"SLAVE2"
is the symbolic name of a data block. This name is defined in the associated symbol
table.
VAR_54
is a variable defined in the "SLAVE2" DB that is also defined as a communication
variable in the symbol table. VAR_54 is the variable to be reported.
```

9.6 STATUS Function Block

Introduction

The STATUS function block allows information on the status of the communication partner to be requested.

Distinctions are drawn between the following:

- The logical status of the communication partner
- The physical status of the communication partner
- Device-specific detailed information

Requirement

The instance DB assigned to the STATUS FB is declared.

STATUS FB Block Diagram

The block diagram shows the call interface of the STATUS FB.

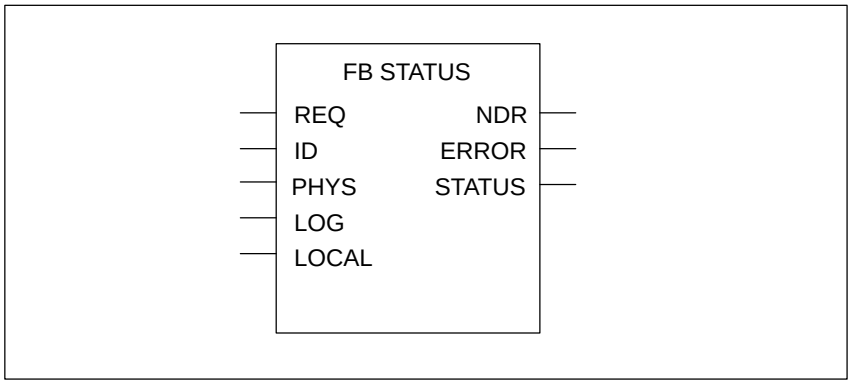


Figure 9-6 Block Diagram of the STATUS FB

STATUS FB Parameters

The table below lists the parameters of the STATUS FB.

Table 9-9 Parameters of the STATUS FB

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	E, A, M, D, L	Bit that activates the processing of the job at a positive edge
ID	INPUT	DWORD	E, A, M, D, L	Connection ID: Number by means of which the connection is uniquely identified (see Section 9.1).
VAR_1	INPUT	ANY	E, A, M, D, L	Pointer to the name of a variable to be read
SD_1	INPUT	ANY	E, A, M, D, L, T, Z	Pointer to the address of a variable to be read Example: P#DB17.DBX0.0 BYTE 16
NDR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the recipient when the job is processed without errors
ERROR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU if an error occurs
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are described (see Section 9.9).
PHYS	INOUT	BYTE	E, A, M, D, L	Physical status
LOG	INOUT	BYTE	E, A, M, D, L	Logical status
LOCAL	INOUT	BYTE[16]	E, A, M, D, L	MMS Standard: Local Details (You will find detailed information on this in the description of the partner.)*

* If the communication partner is a CP 444, you will find the responses of the CP 444 in Table 6-2.

PHYS Parameter for the Physical Status

Table 9-10 shows the possible codes for the PHYS output parameter.

Table 9-10 Codes of the PHYS Parameter for the Physical Status of the Communication Partner

Code for PHYS	Physical status
10H	OPERATIONAL
11H	PARTIALLY OPERATIONAL
12H	INOPERABLE
13H	SUPPORT-SERVICES ALLOWED

LOG Parameter for the Logical Status

Table 9-11 shows the possible codes for the LOG output parameter.

Table 9-11 Codes of the LOG Parameter for the Logical Status of the Communication Partner

Code for LOG	Logical status of the communication partner
00H	STATE-CHANGES-ALLOWED
01H	NO-STATE-CHANGES-ALLOWED
02H	LIMITED-SERVICES-PERMITED (e.g. Status and Identify)
03H	SUPPORT-SERVICES-PERMITTED (e.g. Start, Stop, Reset)

Block Call; Example in STL

The following program sequence shows a block call with corresponding parameterization.

```
STL                                 Explanation
call FB 5, DB 21                    STATUS block call with instance DB
(
REQ    :=  M 1.0                    Edge signal for the execution of the FB
ID     :=  DW#16#10001              Connection ID compensated with configuration
NDR    :=  M 1.1                    Indicates when new data has been received
ERROR  :=  M 1.2                    Indicates errored execution
STATUS :=  MW 20                    Detailed error decoding
PHYS   :=  MB 22                    Data area for the physical status
LOG    :=  MB 23                    Data area for the logical status
LOCAL  :=  P#DB18.DBX0.0 WORD8);    Data area for "local detail"
```

9.7 WRITE Function Block

Introduction

The WRITE function block transfers the data of a variable from a specified local data area to a data area of the communication partner. The local data area can be a data block, an area in the process image of the inputs/outputs or a bit memory address area.

Requirements:

... for calling the WRITE FB in the user program:

- The instance DB assigned to the WRITE FB is declared.
- The variable is configured as a remote variable with the CP 444 (see Section 8.4).

WRITE FB Block Diagram

The block diagram shows the call interface of the WRITE FB.

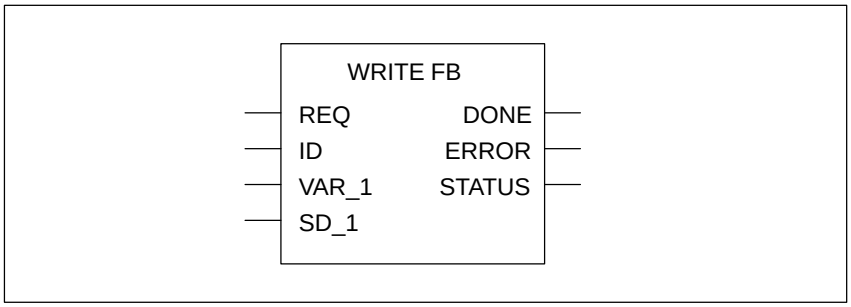


Figure 9-7 Block Diagram of the WRITE FB

WRITE FB Parameters

The table below lists the parameters of the WRITE FB.

Table 9-12 Parameters of the WRITE FB

Parameter	Declaration	Data Type	Memory Area	Description
REQ	INPUT	BOOL	E, A, M, D, L	Bit that activates the processing of the job at a positive edge
ID	INPUT	DWORD	E, A, M, D, L	Connection ID: Number by means of which the connection is uniquely identified (see Section 9.1).
VAR_1	INPUT	ANY	E, A, M, D, L	Pointer to the name of a variable to be sent that is intended to be written by a partner.
SD_1	INPUT	ANY	E, A, M, D, L, T, Z	Pointer to the address of a variable to be sent. Example: P#DB17.DBX0.0 BYTE 16
DONE	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU when the job is processed without errors.
ERROR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU if an error occurs.
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are described (see Section 9.9).

Block Call; Example in STL

The following program sequence shows a block call with corresponding parameterization.

```
STL                                  Description
call FB 6, DB 28                     WRITE block call with instance DB
(
REQ     :=      M 1.0                Edge signal for the execution of the FB
ID      :=      DW#16#10001          Connection ID compared with configuration
VAR_1   :=      "SLAVE2".INDEX       Addresses communication variable to be read
SD_1    :=      "PROZESS".ABBILD     Addresses data area to send transferred data
DONE    :=      M 1.1                Execution confirmation
ERROR   :=      M 1.2                Indicates errored execution
STATUS  :=      MW 20                Detailed error decoding
                );
```

```
Additional information

"SLAVE2".INDEX
is the symbolic name of a structural element of a data block defined and stored as a
communication variable on the communication partner.

"PROZESS".ABBILD
is a locally declared variable in the "PROZESS" DB that contains the value to be
written as the source data area.
```

9.8 ABORT Function Block

Introduction

The ABORT function block allows the defined abortion of an association from the user program. This can be useful, for example, in order to synchronize a number of applications after an error situation.

Requirement

The instance DB assigned to the ABORT FB is declared.

ABORT FB Block Diagram

The block diagram below shows the call interface of the ABORT FB.

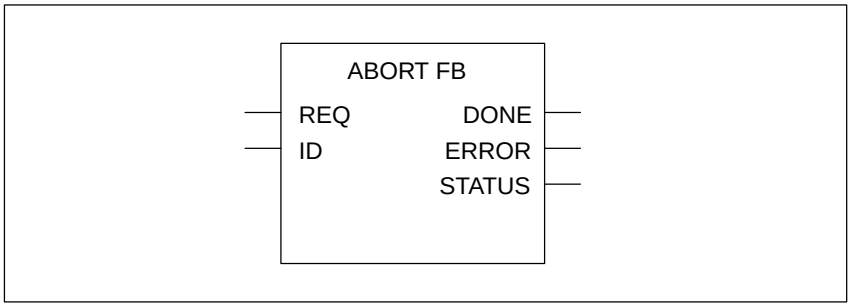


Figure 9-8 Block Diagram of the ABORT FB

ABORT FB Parameters

The following table contains the parameters of the ABORT FB.

Table 9-13 Parameters of the ABORT FB

Parameter	Declaration	Data Type	Storage Area	Description
REQ	INPUT	BOOL	E, A, M, D, L	Bit that activates the processing of the job at a positive edge.
ID	INPUT	DWORD	E, A, M, D, L	Connection ID: Number by means of which the connection is uniquely identified (see Section 9.1).
DONE	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU when the job is processed without errors
ERROR	OUTPUT	BOOL	E, A, M, D, L	Bit set by the CPU when an error occurs
STATUS	OUTPUT	WORD	E, A, M, D, L	Number by means of which status information and error causes are described (see Section 9.9).

Block Call, Example in STL

The following program sequence shows a block call with corresponding parameterization.

```
STL                                 Description
call FB 7, DB 28                    ABORT block call with instance DB
(
REQ      := M 1.0                   Edge signal for the execution of the FB
ID       := DW#16#50001             Connection ID, abortion of asociation no. 5
DONE     := M 1.1                   Execution confirmation
ERROR    := M 1.2                   Indicates errored execution
STATUS   := MW 20);                 Detailed error decoding
```

9.9 Status and Error Information of the Function Blocks

Introduction

This chapter describes the output parameters that are valid for all function blocks and access MMS services of the CP 444.

These are the output parameters DONE and NDR, ERROR and STATUS.

DONE and NDR

The DONE and NDR output parameters are:

- **1**, when job processing is **error-free**
- 0, when the job is currently running or has been completed with errors

ERROR

The ERROR output parameter is:

- **0**, when the job is completed **without errors**
- 1, when the processing of the job is terminated with an error (the STATUS output parameter provides detailed information)

STATUS

If an error occurs during execution of a function block (ERROR = 1), an error code is entered in the STATUS output parameter. Distinctions are drawn between the error codes below on the following basis:

- Local errors
- ServiceErrors
- DataAccessErrors

9.9.1 Error Codes for Local Errors

Introduction

Errors that occur when an MMS job is created or during processing of a received MMS acknowledgment on the local CP 444 are referred to as local errors. The error code of a local error is output in the STATUS output parameter and is located in the area from W#16#0001 to W#16#7FFF.

Error Codes for Local Errors

Table 9-14 shows the error codes for local errors that the CPU can display in the STATUS output parameter.

Table 9-14 Error Codes in the STATUS Output Parameter

STATUS (W#16#...)	Class	Meaning
0001	Block	Communication problem: For example, the K bus connection has not been set up.
0002	Block	Function cannot be executed: Either a negative acknowledgment from the CP 444 or an error in the sequence.
0003	Block	R_ID is not known on this K bus connection; the resource record does not exist.
0004	Block	The receive data area is too short, or the data types do not match.
0005	Block	A reset request has arrived from the CP 444.
0006	Block	Corresponding job processing in the CP 444 has the status DISABLED, or a reset request has arrived from the CP 444; consequently, the transfer is incomplete.
0007	Block	Corresponding job processing in the CP 444 does not have the correct status, e.g. instance DB is not defined (see section 8.4.1)
0008	Block	Job processing of the CP 444 reports user memory access error.
000A	Block	Access to the local user memory not possible (the DB has been deleted, for example).
000B	Block	Warning: The job is already running.
000C	Block	When an underlying BSEND or BRCV SFB was called, an instance DB not belonging to SFB 12 / SFB 13 was specified, or a global DB was used rather than an instance DB.
0011	Block	Warning: The job is receiving asynchronous data.
0012	Block	R_ID already exists in the connection. For example, there are two FB calls with the same instance DB.
0502	Request Association	Only in the case of the WRITE FB: data length of the FB exceeded
1000	Request Association	Error not specified in any more detail

Table 9-14 Error Codes in the STATUS Output Parameter, continued

STATUS (W#16#...)	Class	Meaning
1001	Request Association	Invalid MMS association
1002	Request Association	MMS association not set up
1003	Request Association	MMS PDU length inadequate
1100	Request Service	Error not specified in any more detail
1101	Request Service	Job contains unknown parameters.
1200	Request Variable	Error not specified in any more detail
1201	Request Variable	Data type does not match the variable definition (e.g. Write-Request, InformationReport)
1202	Request Variable	The MMS variable has not been configured.
1203	Request Variable	The data format in the CPU does not match the type of the variable (e.g. invalid characters in the DB when the type is Visible String)
1204	Request Service	Asso queue overflow For example, there are too many jobs for this association in the queue.
1205	Request Service	MMS stack overflow
2001	Response Association	Timeout: remote communication partner is not responding.
2002	Response Association	MMS abort received
2003	Response Association	deutsch: MMS-PDU-Länge nicht ausreichend
2100	Response Service	Error not specified in any more detail
2101	Response Service	Response contains unknown parameters.
2200	Response Variable	Error not specified in any more detail
2201	Response Variable	Data type does not match the variable definition (e.g. Read-Response)

9.9.2 Error Codes for ServiceErrors

Introduction

Errors that occur on a remote communication partner and are received by the CP 444 with an MMS error acknowledgment are referred to as ServiceErrors. The error code of a ServiceError is output in the STATUS output parameter. It can range from W#16#8000 to W#16#823F and from W#16#824A to W#16#8DFF.

For a detailed description of the cause of the error, please refer to the documentation of the remote communication partner.

Decoding the Error Code

The MMS codes for ErrorClass and ErrorCode are encoded in the error code of the ServiceError.

You obtain these codes by subtracting W#16#8000 from the error code displayed; the first byte contains the value for ErrorClass and the second the value for ErrorCode.

Example

Table 9-15 shows how to obtain the MMS code for ErrorClass and ErrorCode from the error code.

Table 9-15 MMS Code for ErrorClass and ErrorCode

Error Code in the STATUS Output Parameter	Error Code Minus W#16#8000	MMS Code for ErrorClass	MMS Code for ErrorCode
8001	0001	0	1
8B03	0B03	B	3

Error Codes for ServiceErrors

Table 9-16 shows an overview of the error codes for ServiceErrors that are reported by the CP 444 and can be displayed in the STATUS output parameter.

Table 9-16 Error Codes for ServiceErrors and Their Meaning

STATUS (W#16#...)	ErrorClass	ErrorCode	MMS Name
VMD-State ErrorClass			
8000	0	0	Other
8001	0	1	VMD-State-Conflict
8002	0	2	VMD-Operational-Problem
8003	0	3	Domain-Transfer-Problem
8004	0	4	State-Machine-ID-Invalid
Application Reference ErrorClass			
8100	1	0	Other
8101	1	1	Application-Unreachable
8102	1	2	Connection-Lost

Table 9-16 Error Codes for ServiceErrors and Their Meaning, continued

STATUS (W#16#...)	ErrorClass	ErrorCode	MMS Name
8103	1	3	Application-Reference-Invalid
8104	1	4	Context-Unsupported
Definition ErrorClass			
8200	2	0	Other
8201	2	1	Object-Undefined
8202	2	2	Invalid-Address
8203	2	3	Type-Unsupported
8204	2	4	Type-Inconsistent
8205	2	5	Object-Exists
8206	2	6	Object-Attribute-Inconsistent
Resource ErrorClass			
8300	3	0	Other
8301	3	1	Memory-Unavailabe
8302	3	2	Processor-Resource-Unavailable
8303	3	3	Mass-Storage-Unavailable
8304	3	4	Capability-Unavailable
8305	3	5	Capability-Unknown
Service ErrorClass			
8400	4	0	Other
8401	4	1	Primitives-Out-Of-Sequence
8402	4	2	Object-State-Conflict
8403	4	3	Client PDU-Size (DIS only)
8404	4	4	Continuation-Invalid
8405	4	5	Object-Constraint-Conflict
Service Preempt ErrorClass			
8500	5	0	Other
8501	5	1	Timeout
8502	5	2	Deadlock
8503	5	3	Cancel
Time-Resolution ErrorClass			
8600	6	0	Other
8601	6	1	Unsupported-Time-Resolution

Table 9-16 Error Codes for ServiceErrors and Their Meaning, continued

STATUS (W#16#...)	ErrorClass	ErrorCode	MMS Name
Access ErrorClass			
8700	7	0	Other
8701	7	1	Object-Access-Unsupported
8702	7	2	Object-Non-Existent
8703	7	3	Object-Access-Denied
8704	7	4	Object-Invalidated
Initiate ErrorClass			
8800	8	0	Other
8803	8	3	Max-Services-Outstanding-Calling-Insuficient
8804	8	4	Max-Services-Outstanding-Called-Insuficient
8805	8	5	Service-CBB-Insufficient
8806	8	6	Parameter-CBB-Insufficient
8807	8	7	Nesting-Level-Insufficient
Conclude ErrorClass			
8900	9	0	Other
8901	9	1	Further-Communication-Required
Cancel ErrorClass			
8A00	10	0	Other
8A01	10	1	Invoke-ID-Unknown
8A02	10	2	Cancel-Not-Possible
File ErrorClass			
8B00	11	0	Other
8B01	11	1	Filename-Ambiguous
8B02	11	2	File-Busy
8B03	11	3	Filename-Syntax-Error
8B04	11	4	Content-Type-Invalid
8B05	11	5	Position-Invalid
8B06	11	6	File-Access-Denied
8B07	11	7	File-Non-Existent
8B08	11	8	Duplicate-Filename
8B09	11	9	Insufficient-Space-In-Filestore
Others ErrorClass			
8C00	12		Other

9.9.3 Error Codes for DataAccessErrors

Introduction

Errors that occur when a variable is accessed and that are received with an MMS read acknowledgment or an MMS write acknowledgment are referred to as DataAccessErrors. The error code of a DataAccessError can range from W#16#8240 to W#16#8249.

For a detailed description of the cause of the error, please refer to the documentation of the remote communication partner.

Decoding the Error Code

The MMS code for the DataAccessError is encoded in the error code. You obtain the MMS code of a DataAccessError by subtracting W#16#8240 from the error code displayed.

Example

Table 9-17 shows how to obtain the MMS code of the DataAccessError from the error code displayed.

Table 9-17 Obtaining the MMS Code for DataAccessErrors

Error Code in the STATUS Output Parameter	Error Code Minus W#16#8240	MMS Code for DataAccessError
8241	0001	1
8247	0007	7

Error Codes for DataAccessErrors

Table 9-18 shows an overview of the error codes for DataAccessErrors that are reported by the CP 444 and can be displayed in the STATUS output parameter.

Table 9-18 Error Codes for DataAccessErrors and Their Meaning

STATUS (W#16#...)	DataAccess Error	MMS Name
8240	00	Object Invalidated
8241	01	Hardware fault
8242	02	Temporarily Unavailable
8243	03	Object Access Denied
8244	04	Object Undefined
8245	05	Invalid Address
8246	06	Type Unsupported
8247	07	Type Inconsistent
8248	08	Object Non Existent

9.10 Technical Specifications of the Function Blocks

Storage Space Requirements

The table below indicates the storage space requirements of the function blocks of the CP 444.

Table 9-19 Storage Space Requirements of the Function Blocks in Bytes

Block	Name	Version	Transferable User Data	Load Memory	Work Memory	Local Data	Load Memory Instance DB	Work Memory Instance DB
FB 1	ACCESS	1.00	4096	2718	2326	106	4458	4200
FB 2	IDENT	1.00	235	1664	1328	136	470	196
FB 3	READ	1.60	235	2406	1984	130	612	338
		1.60	4096	2406	1984	130	4470	4196
FB 4	REPORT	1.60	235	2414	1988	142	638	358
		1.60	4096	2414	1988	142	4564	4284
FB 5	STATUS	1.60	235	1666	1358	112	444	190
FB 6	WRITE	1.60	235	2414	1988	142	638	358
		1.60	4096	2414	1988	142	4564	4284
FB 7	ABORT	1.00	–	1142	872	76	310	80

Start-Up and Operating Characteristics of the CP 444

10

Purpose of This Chapter

After reading this chapter, you will know the prerequisites for running the CP 444 and how the CP 444 behaves after it is switched on.

Chapter Overview

This chapter is divided into the following sections:

10.1 Start-Up Characteristics of the CP 444

Prerequisite

So that the CP 444 can start up, its mode selector must be in the RUN-P position. The position of the CPU's mode selector is irrelevant to the start-up of the CP 444.

Start-Up

The start-up includes complex self-tests of the hardware and software configuration, initialization and connection setup to the CPU. The entire start-up takes approximately 80 to 120 seconds.

Meaning of the LEDs at Start-Up

The MRDY and MERR LEDs allow you to follow the progress of the CP 444's start-up.

Table 10-1 Meaning of the MRDY and MERR LEDs at Start-Up

MRDY	MERR	Meaning
Off	Off	Start of CP 444 initialization
Flashing	Flashing	Self-test and initialization running
On	On	Self-test and MMS initialization completed.
On	Off	K bus connection to the CPU set up
Off	On	Error in the CP 444. A reset is required, or the configuration must be reloaded.

Important

Please note the following in relation to the behavior of the CP 444:

Note

After the power is switched on, the CP 444 requires approximately 80 to 120 seconds for its start-up (initialization, hardware and memory tests).

During this time, the function blocks called in the user program of the CPU are not processed by the CP 444. You can read a corresponding error message at the STATUS output parameter of the FB called (error code W#16#0001).

10.2 Operating Characteristics of the CP 444

Introduction

In the following, it is assumed that the CP 444 has completed start-up processing and is in RUN.

Data interchange depends on the operating status of the CPU. The description below indicates the communication that is possible when the CPU is in RUN and STOP.

CPU and CP 444 in RUN

If both the CPU and CP 444 are in RUN, the CPU and the remote communication partner can request all the variable services and VMD services supported by the CP 444.

CPU and CP 444 in STOP

Communication direction: CPU > CP 444 and CP 444 > CPU

If the CPU is in STOP, the CP 444 processes all services that do not require a function block.

Communication direction: CP 444 > remote communication partner

Jobs received by the CP 444 before the CPU is switched to STOP continue to be processed by the CP 444 and transferred to the remote communication partner.

Communication direction: remote communication partner > CP 444

If the CP 444 receives the data for a VMD service from a remote communication partner, the CP 444 processes this job (since the CPU is not involved in VMD services).

If the CP 444 receives the data for a variable service from a remote communication partner, this service is aborted with the error message W#16#8242 (see Table 9-18) to the remote communication partner.

10.3 Resetting the CP 444 Using the Mode Selector

Introduction

If the CP 444 cannot be switched to RUN as described in Section 10.1, this generally means that there is a correctable error in the internal memory. To correct this error, you have to reset the CP 444.

You reset the CP 444 using the mode selector. You must adhere to a particular sequence when doing this.

Resetting the CP444 (Hardware Reset)

To reset the CP 444 using the mode selector, proceed as follows:

1. Turn the mode selector to STOP.

   Result: The STOP LED comes on.

2. Turn the mode selector to MRES, and keep it in this position.

   Result: The STOP LED goes off and comes on again after about 3 seconds.

3. Within 3 seconds of the STOP LED coming on again, turn the mode selector back to STOP, then to MRES and back to STOP again.

   Result: The STOP LED flashes for about 3 seconds at 2.5 Hz (rapid flashing) and then remains on. The hardware reset is carried out.

   If the STOP LED does not flash, of if other LEDs come on or flash, repeat steps 2 and 3.

Note

If you want the CPU to remain in RUN mode when the CP 444 is reset, the insert/remove OB (OB 83) must be loaded.

Diagnostics with the CP 444

11

Purpose of This Chapter

This chapter describes the diagnostic functions of the CP 444. You will find here all the information you need for diagnostics using:

- The display elements (LEDs) of the CP 444
- The MMS error messages of the CP 444 to remote communication partners

Status Output at the FB

The interpretation of the error numbers at the STATUS output of the function blocks is described in Section 9.9.

Chapter Overview

This chapter is divided into the following sections:

11.1 Diagnostic Functions of the CP 444

Introduction

The diagnostic functions of the CP 444 enable you to localize quickly any errors that occur. The following diagnostic options are available:

- Diagnostics using the display elements of the CP 444
- Diagnostics using the STATUS output of the function blocks
- Diagnostics using the MMS error messages of the CP 444 to remote communication partners
- Diagnostics using the MMS trace function

Display Elements (LEDs)

The display elements indicate the operating status and any error statuses of the CP 444. The display elements give you an initial overview of any internal or external errors as well as interface-specific errors (see Section 11.2).

STATUS Output of the FBs

Every function block has the STATUS output parameter for diagnostics and responding to errors in the user program. An error code is output to it, by means of which you get detailed information on errors and events.

By reading the STATUS output of the function blocks you get detailed information on errors/events that have occurred during communication between the CP 444, the assigned CPU and the communication partner connected on this connection.

Note

The interpretation of the error numbers at the STATUS output of the function blocks is described in Section 9.9.

MMS Error Messages

If a remote communication partner requests MMS services that the CP 444 cannot execute, the CP 444 sends an error code to the remote communication partner. To enable you to carry out detailed error analysis in the remote communication partner, the MMS error messages of the CP 444 are listed in Section 11.3.

MMS Trace

Using the MMS trace you can record an MMS trace log. The log provides a step-by-step record of the connection setup process, for example, and enables connection problems to be localized.

11.2 Diagnostics Using the Display Elements of the CP 444

Introduction

The LEDs of the CP 444 indicate to you the error and operating status of the CP 444 at a glance.

Meaning of the Status and Error Displays

Table 11-1 explains the status and error displays of the CP 444.

Table 11-1 Meaning of the Status and Error Displays of the CP 444

Display	Meaning	Explanation
INTF (red)	Internal error message	Comes on at: • Hardware faults • Firmware errors • Parameterization errors
EXTF (red)	External error message	Comes on at: • Peripheral errors
SD (green)	(unused)	–
HD (green)	Hard disk access	Comes on at read or write access
MRDY (yellow)	CP 444 has connection to CPU	The MRDY and MERR LEDs have an additional meaning during start-up (see Table 11-2), data transfer (see Table 11-3) and loading of the configuration data (see Table 11-4).
MERR (yellow)	Error in the MMS task	
RUN (green)	"**RUN**" status display	Comes on when the system software is loaded and the CP 444 can exchange data with the remote communication partner.
STOP (yellow)	"**STOP**" status display	Comes on when there is no communication between the CP 444 and the CPU.
All LEDs come on briefly when the CP 444 is switched on or reset (self-test).		

MRDY and MERR at Start-Up

After the CP 444 is switched on in RUN, the CP 444 tests the hardware and prepares itself for operation. The MRDY and MERR LEDs allow you to follow the progress of the CP 444's start-up.

Table 11-2 Meaning of the MRDY and MERR LEDs at Start-Up

MRDY	MERR	Meaning
Off	Off	Start of CP 444 initialization
Flashing	Flashing	Initialization is running
On	On	Self-test and MMS initialization completed.
On	Off	Connection established to the CPU
Off	On	Error in the CP 444. A reset is required, or the configuration must be reloaded.

MRDY and MERR in RUN

When the CP 444 is in RUN, the CP 444 can send and receive data. The MRDY and MERR LEDs allow you to follow the progress of the transmission and receipt.

Table 11-3 Meaning of the MRDY and MERR LEDs in RUN

MRDY	MERR	Meaning
Flashing	Off	CP 444 is receiving an MMS service
Off	Flashing	CP 444 is sending an MMS service

MRDY and MERR When Loading the Configuration Data

When the CP 444 is in STOP, you can load the configuration data. The MRDY and MERR LEDs allow you to follow the progress of the load operation.

Table 11-4 Meaning of the MRDY and MERR LEDs at Loading

MRDY	MERR	Meaning
On	On	Configuration data is being loaded
On	Off	Loading completed, connection exists to CPU
Off	On	Error during loading

11.3 MMS Error Messages of the CP 444 to Remote Communication Partners

Introduction

If a remote communication partner requests MMS services that the CP 444 cannot execute, the CP 444 sends an error code to the communication partner. The error codes are subdivided into ServiceErrors and DataAccessErrors.

ServiceError

The ServiceError is the error code of an MMS error acknowledgment and specifies the error that has occurred in the CP 444. MMS subdivides the ServiceError into ErrorClass and ErrorCode.

You need the error code for ErrorClass and ErrorCode in order to respond to the situation in the remote communication partner.

DataAccessError

The DataAccessError is the error code of an MMS read acknowledgment or an MMS write acknowledgment and specifies errors reported when variables are accessed.

11.3.1 Error Codes for ServiceError (ErrorClass and ErrorCode)

Error Codes for ServiceError

ServiceError consists of two components, the ErrorClass and the ErrorCode. Table 11-5 contains a list of the possible error codes for ErrorClass and ErrorCode that the CP 444 can send to the remote communication partner.

Table 11-5 Error Codes for ServiceError and Their Meaning

ErrorClass	ErrorCode	MMS Name	Comment
VMD-State ErrorClass			
0	0	Other	Error not specified in any more detail
0	1	VMD-State-Conflict	Service not permitted with current VMD status
0	2	VMD-Operational-Problem	Service not possible, since the CP 444 cannot reach the CPU
Definition ErrorClass			
2	0	Other	Error not specified in any more detail
2	1	Object-Undefined	Object not defined
2	3	Type-Unsupported	Type not supported
2	4	Type-Inconsistent	Type not consistent
2	6	Object-Attribute-Inconsistent	Object attribute not consistent
Resource ErrorClass			
3	0	Other	Error not specified in any more detail
3	1	Memory-Unavailabe	Memory problems
Access ErrorClass			
7	0	Other	Error not specified in any more detail
7	1	Object-Access-Unsupported	Access to object not supported
7	2	Object-Non-Existent	Object does not exist
Initiate ErrorClass			
8	0	Other	Error at Initiate not specified in any more detail
8	3	Max-Services-Outstanding-Calling-Insufficient	Max-Services-Outstanding-Calling parameter insufficient
8	4	Max-Services-Outstanding-Called-Insufficient	Max-Services-Outstanding-Called insufficient
8	5	Service-CBB-Insufficient	Service-CBB-Insufficient insufficient
8	6	Parameter-CBB-Insufficient	Parameter-CBB insufficient
8	7	Nesting-Level-Insufficient	Nesting-Level insufficient

Table 11-5 Error Codes for ServiceError and Their Meaning, continued

ErrorClass	ErrorCode	MMS Name	Comment
Conclude ErrorClass			
9	0	Other	Error at Conclude not specified in any more detail
9	1	Further-Communication-Required	Conclude cannot be executed because services are still being processed
Cancel ErrorClass			
10	2	Cancel-Not-Possible	Cancel is not possible

11.3.2 Error Codes for DataAccessError

Error Codes for DataAccessError

DataAccessError describes the error that can occur when a variable is accessed. Table 11-6 contains a list of the possible error codes that the CP 444 can send to the remote communication partner.

Table 11-6 Error Codes for DataAccessError and Their Meaning

DataAccessError	MMS Name	Comment
00	Object Invalidated	Variable is invalid
01	Hardware fault	Hardware fault (e.g. CPU does not exist)
02	Temporarily Unavailable	The variable cannot be accessed because access via the user program is blocked (bit in access DB reset)
03	Object Access Denied	The variable cannot be accessed because it was configured with the READ-ONLY attribute
04	Object Undefined	Variable is not configured
05	Invalid Address	The data block that should contain the variable does not exist or is too short
06	Type Unsupported	Variable type is not supported
07	Type Inconsistent	Variable type is inconsistent
08	Object Non Existent	Variable does not exist

11.4 Error/Fault Analysis Using the MMS Trace

Introduction

You can use the MMS trace to record an MMS trace log. The log provides a step-by-step record of the connection setup process, for example, and enables connection problems to be localized.

Principle

You start the trace using the *Configuration for CP 444* parameterization interface. The recorded log is saved on the inserted 3.5” floppy disk in the CP 444. You can then load the log data on the programming device/PC and display it in plain text.

Starting the Parameterization Interface

You start the parameterization interface by double-clicking the CP 444 in the configuration table or by selecting the CP 444 and choosing **Edit > Object Properties**.

Result: The ”Properties – CP 444” dialog box appears.

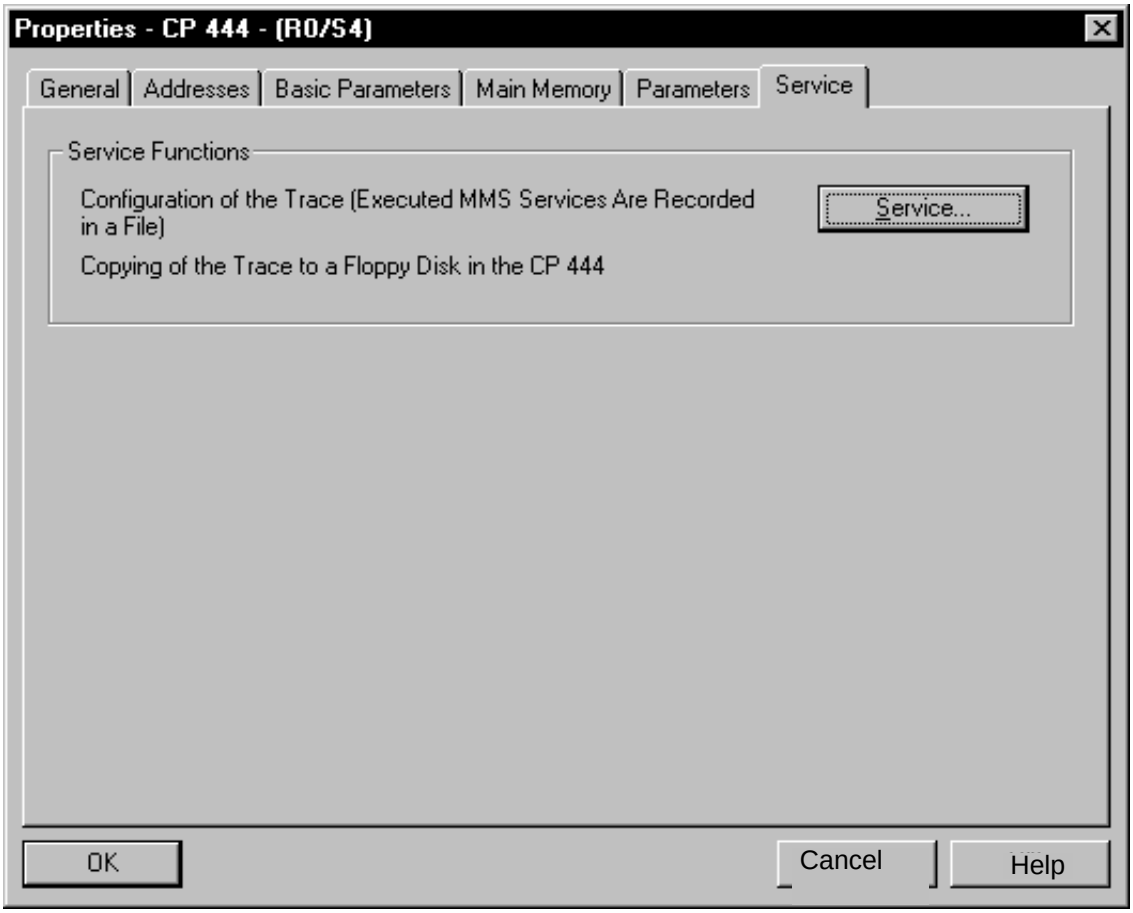


Figure 11-1 “Properties - CP 444” Dialog Box (”Service” Tab)

Recording and Displaying the MMS Trace Log

To record an MMS trace log, proceed as follows:

1. Change to the “Service” tab by simply clicking the “Service” tab label in the upper part of the dialog box.

   Result: The ”Service” tab appears in the foreground (see Figure 11-1).

2. Click the “Service” button.

   Result: The “Diagnostics” dialog box appears.

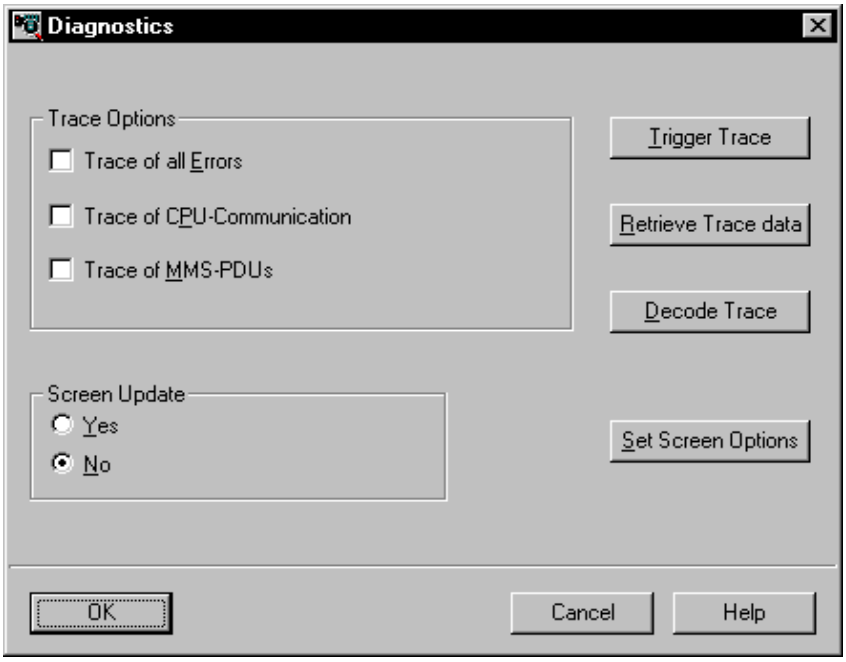


Figure 11-2 “Diagnostics” Dialog Box

3. Set the trace options. To get more information on this, click the ”Help” button.
4. Start the trace by clicking the ”Trigger Trace” button.

   Result: The data of the MMS trace log is saved in a file on the hard disk.

5. To stop the recording, clear the check box in the ”Trace Options” group, and click the ”Trigger Trace” button again.
6. To retrieve the trace data, proceed as follows:

   – Insert a floppy disk in the CP 444’s drive.

     Note: All the data on the floppy disk will be deleted.

   – Click the ”Retrieve Trace Data” button.

   Result: The trace data is compressed and saved on the floppy disk. The access LED on the floppy disk drive indicates when the data transfer is completed.

7. To evaluate the trace data, insert the floppy disk containing the recorded trace data in the drive of your programming device/PC, and click the ”Decode Trace” button.

   Result: The trace data is decompressed and displayed in the Wordpad text editor.

Screen Update

Default = NO

To activate screen updating, set YES and load the change in the CP 444 by means of ”Set Screen Options”. This sets the screen update to 5 seconds.

Example 1: Trace of All Errors

Trace data

```
mms_op_error returns : 0x6702
 Error mv_write_variables: MMS VMI : Type Name
 +++ CFB WRITE MMS REQUEST FAILED-DATA DEFINITION. ON PLC MAY NOT BE O.K!
 +++ Local Error = 0x1203


+++CFB STATUS  MMS REQUEST FAILED-CHANNEL NOT ASSOCIATED
+++ Local Error = 0x1002

+++CFB IDENTIFY MMS RESPONSE FAILED
+++MMS Service Error = 0x8002


Error of Status = MMS RESP : Connection Terminated
Error code = 0x6604
CFB STATUS MMS RESPONSE FAILED
MMS Service Error = 0x8002
```

Example 2: Trace of CPU Communication and Trace of MMS-PDUs

Trace data

```
******************
Logging started DD\MM\YY 5\1\1994 Time 22:46:58:700
***************

*********************Time Elapsed in secs : millisecs = 2 : 2733 *********************
Creation of the VMD remote variables without errors
Creating Remote VMDs for 8 channels.
oder
ERROR Creating Remote VMDs.
Detailed information with ”Trace of All Errors”

***************** ***Time Elapsed in secs : millisecs = 12 : 11974 ******************
CP <-> CPU connection setup
CPU = 1
Local TSAP = 1005
Remote TSAP = 1003
Remote Address = 0100000003000000000000100
Negotiated Packet Size for K-Bus = 480
```

Trace data (continued)

```
**********Time Elapsed in secs : millisecs = 12 : 12407 **********
Ascertainment of the operating mode of the CPU
after s7ag_read_szl() num Bytes = 28
KBUS DATA DUMP::
04 24 00 01 00 14 00 01 51 40 FF 08 00 00 00 00
00 00 00 00 94 01 20 18 41 37 51 55

**********Time Elapsed in secs : millisecs = 13 : 12754 **********
Connection to the CPU successfully set up
Log Start DB open successful = 1, lret=0x0

**********Time Elapsed in secs : millisecs = 13 : 13199 **********
Initialization from the remote partner for passive association 5
MMS INDICATION RECEIVED
Channel = 5
Context = 01
Invoke ID = 32773
Opcode = 83
Operation is : INITIATE

**********Time Elapsed in secs : millisecs = 13 : 13207 **********
Response of the local CP for the initialization of association 5
ISSUING MMS RESPONSE
Channel = 5
Context = 01
Invoke ID = 32773
Opcode = 83
Operation is : INITIATE




**********Time Elapsed in secs : millisecs = 14 : 13509 **********
Initialization to the remote partner for active association 0
ISSUING MMS REQUEST
Channel = 0
Context = 01
Invoke ID = 32768
Opcode = 83
Operation is : INITIATE

**********Time Elapsed in secs : millisecs = 14 : 13656 **********
Response of the remote partner to the initialization of association 0
MMS CONFIRM RECEIVED
Channel = 0
Context = 01
Invoke ID = 32768
Opcode = 83
Elapsed Time : 0 Seconds
Operation is : INITIATE

or remote partner does not respond to the initialization of association 0
Abort indication for channel = 0
GW abort,AP_abort

**********Time Elapsed in secs : millisecs = 37 : 36919 **********
Updating of the Asso_Flag for association 4
Update for Channel number  : 4
No of bytes written from the PLC  : 1
KBUS DATA DUMP::
FF
lret of s7-write : 00
```

Trace data (continued)

```
**********Time Elapsed in secs : millisecs = 67 : 66760 **********
Transmission initiation by CFB
No of bytes read from the PLC  : 146
KBUS DATA DUMP::
00 00 01 A0 00 8C 00 01 00 03 00 01 00 00 00 00
00 00 00 00 00 00 *00 0D* 53 5F 42 49 4E 5F 4D 45 *00 0D* Length of the variable name
followed by name in hex.
53 53 41 47 45 *20* /00 6A/ 00 01 00 6A 53 5F 42 49  *20* Filler byte = even-numbered
variable name
4E 5F 4D 45 53 53 41 47 45 2F 56 53 46 5B 32 36   /00 6A/ Length of the net data
followed by net data
5D 2D 45 4E 44 45 00 00 00 00 FF 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 FF 00 00 00 00
00 00

**********Time Elapsed in secs : millisecs = 67 : 66809 **********
Textual description of the CFB initiation
BRCV INDICATION RECEIVED in CFBPARSE.C.
r_id = 416                Instance DB no.
Channel = 1               Association no.
Length of Variable : 13   Length of the variable name
Operation is : CFB Write  Type of initiation
Variable = S_BIN_MESSAGE  Variable name

**********Time Elapsed in secs : millisecs = 67 : 66851 **********
WRITE order to remote partner
ISSUING MMS REQUEST
Channel = 1
Context = 01
Invoke ID = 1
Opcode = 5
Operation is : WRITE

**********Time Elapsed in secs : millisecs = 67 : 67117 **********
Acknowledgment from remote partner
MMS CONFIRM RECEIVED
Channel = 1
Context = 01
Invoke ID = 1
Opcode = 5
Elapsed Time : 0 Seconds
Operation is : WRITE
CFB WRITE SUCCESSFUL

**********Time Elapsed in secs : millisecs = 67 : 67153 **********
Acknowledgment to the WRITE CFB
No of bytes sent to the PLC  : 4
KBUS DATA DUMP::
00 00 00 00

---------------------------------- END OF EXAMPLES ----------------------------------
```

Programming Example: Variable Services 12

Programming Example:

The floppy disk containing the CP 444 configuration software also contains a complete programming example of the use of the variable services.

Installation

The programming example is installed together with the CP 444 configuration software (see Section 8.1). After installation you will find it in the two projects **`Ag1`** and **`Ag2`**. You open a project by choosing the **File > Open > Project** menu command.

Description

The programming example is not described in this edition of the manual. You will find a detailed description of how to configure the CP 444 using the *Configuration for CP 444* parameterization interface and how to program the user program using *STEP 7* in the beispiel.doc file.

You will find **`beispiel.doc`** in the directory of the **`Ag1`** project (access via **`SIEMENS\STEP7\EXAMPLE\CP444AG1`** in Windows95 Explorer).

Technical Specifications

A

Chapter Overview This chapter is divided into the following sections:

In Section	You Will Find	on Page
A.1	Technical Specifications of the CP 444	A-2
A.2	Certification and Application Areas	A-4

A.1 Technical Specifications of the CP 444

CP 444 Technical Specifications

The table below contains the technical specifications of the CP 444.

Table A-1 Technical Specifications of the CP 444

Technical Specifications	
Dimensions W × H × D (mm)	50 × 290 × 210
Weight	Approx. 2.08 kg
Current input from backplane bus	Max. 3.1 A
Power loss	Typically 15.6 W
Interface	
Front connector (IF 2)	Screw-type 15-pin subminiature D female connector in compliance with the IEEE 802.3 Ethernet standard
Transmission rate	10 Mbps in accordance with IEEE 802.3
Line length (to transceiver)	Max. 50 m with connecting cable 727-1 Max. 100 m with industrial twisted-pair installation cable

For additional technical data, see the Reference Manual:
S7-400 and M7-400, Programmable Controllers, Module Specifications; Chapter 1.

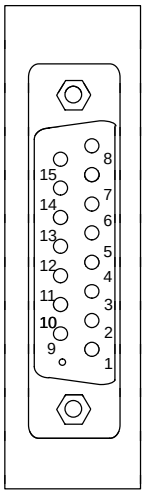
The technical data of the MSM 478 in Section 13.4.2 also apply.

Pin Assignment: 15-Pin Subminiature D Female Connector

The table below contains the assignment of the 15-pin subminiature D female connector (IF 2) on the front panel of the CP 444.

The pins are assigned in accordance with the IEEE 802.3 Ethernet standard.

Table A-2 Pin Assignment of the 15-Pin Subminiature D Female Connector

Female Connector on CP 444*	Pin	Designation	Meaning
	1	MEXT	External ground, shield
	2	CLSN	Collision (+)
	3	TRMT	Transmit (+)
	4	M	Ground (reference potential)
	5	RCV	Receive (+)
	6	M	Ground (reference potential)
	7	–	–
	8	–	–
	9	CLSN_N	Collision (−)
	10	TRMT_N	Transmit (−)
	11	–	–
	12	RCV_N	Receive (−)
	13	–	–
	14	–	–
	15	–	–

* View from the front

Transceiver Connecting Cables

If you want to make your own connecting cables for the transceiver, **please note** that you must only use shielded connector casings. To ensure EMC (electromagnetic compatibility), the cable shielding must be connected to each connector casing over a large area.

Note

The 15-pin subminiature D female connector of the CP 444 does not provide power (+15 V) for a transceiver. When the CP 444 is connected to a transceiver, the transceiver must also be connected via a separate power supply.

A.2 Certification and Application Areas

Introduction

This chapter gives the certificates and approvals of the CP 444 and the most important standards it complies with.

IEC 1131

The CP 444 communication processor fulfills the requirements and criteria of IEC 1131, Part 2.

CE Marking

CE

Our products fulfill the requirements and safety objectives of the following EC Directives and comply with the harmonized European standards (EN) published for stored-program controllers in the official journals of the European Communities:

- 89/336/EEC Electromagnetic Compatibility Directive (EMC Directive)
- 73/23/EEC Low Voltage Directive (for electrical equipment)

The EC Declarations of Conformity are available to the relevant authorities at the following address:

Siemens Aktiengesellschaft
Bereich Automatisierungstechnik
A&D AS E 14
Postfach 1963
D-92209 Amberg

Area of Application

In accordance with this CE marking, the CP 444 has the following area of application:

Area of Application	Requirements	
	Emitted Interference	**Noise Immunity**
Industry	EN 50081-2 : 1993	EN 50082-2 : 1995

UL Recognition

UL Recognition Mark
Underwriters Laboratories (UL) to
Standard UL 508, Report E 85972

CSA Certification

CSA Certification Mark
Canadian Standard Association (CSA) to
Standard C22.2 No. 142, Report No. LR 63533

FM Approval

The following approval has been obtained for the S7-300:

Factory Mutual Approval Standard Class Number 3611, Class I, Division 2, Group A, B, C, D.

Warning

Personal injury or property damage can result.

In hazardous areas, personal injury or property damage can result if you create or break an electrical circuit during operation of an S7-300 (for example, by means of plug-in connections, fuses, switches).

Do not create or break live electric circuits unless you are certain there is no danger of explosion.

Warning

WARNING - DO NOT DISCONNECT WHILE CIRCUIT IS LIVE UNLESS LOCATION IS KNOWN TO BE NON-HAZARDOUS.

Accessories and Order Numbers

B

Accessories and Order Numbers

The table below provides you with an overview of the accessories and order numbers of the CP 444:

Table B-1 Accessories and Order Numbers of the CP 444

Product		Order Number
CP 444 Communication Processor		6ES7 444-1MX00-0XE0
CP 444 configuration package consisting of: • The *CP 444 Communication Processor, Installation and Parameter Assignment* manual • A floppy disk with: – The *Configuration for CP 444* parameterization interface – A library containing function blocks – Programming example		6ES7 444-1MX00-7□G0 ↑ German A English B
727-1 connecting cable for systems with AUI	3.2 m	6ES5 727-1BD20
	10 m	6ES5 727-1CB00
	15 m	6ES5 727-1CB50
	20 m	6ES5 727-1CC00
	32 m	6ES5 727-1CD20
	50 m	6ES5 727-1CF00
Industrial twisted-pair installation cable	2 m	6XV1 850-0BH20
	5 m	6XV1 850-0BH50
	8 m	6XV1 850-0BH80
	12 m	6XV1 850-0BN12
	15 m	6XV1 850-0BN15
	20 m	6XV1 850-0BN20
	30 m	6XV1 850-0BN30
	40 m	6XV1 850-0BN40
	50 m	6XV1 850-0BN50
	60 m	6XV1 850-0BN60
	70 m	6XV1 850-0BN70
	80 m	6XV1 850-0BN80
	90 m	6XV1 850-0BN88
	100 m	6XV1 850-0BT10

Overview of the MMS Services Supported C

Chapter Overview

This chapter provides you with an overview of which MMS services are supported by the CP 444. It is divided into the following sections:

C.1 PICS Part One: Implementation Information

Implementation Information

Table C-1 shows you the device information of the CP 444.

Table C-1 Device Information of the CP 444

Implementation Information	CP 444
Implementation Vendor Name	Siemens AG
Implementation Model Name	CP 444
Implementation Revision Identifier	<firmware version>
Machine Name(s) and Version Number(s)	
Operating System(s)	
MMS abstract syntax	MMS core
MMS Version Number Supported	MMS-DIS, MMS-IS
MMS CS abstract syntaxes	
MMS CS Version Number Supported	
Calling MMS-user (indicate ”Yes” or ”No”)	YES
Called MMS-user (indicate ”Yes” or ”No”)	YES
List of Standardized Names	

C.2 PICS Part Two: Service Conformance Building Blocks

Introduction

The tables below contain the service conformance blocks of the CP 444.

The last column in each table indicates whether the service is supported as a client, a server or both. A dash (–) means the service is not supported by the CP 444.

Environment Management

Table C-2 provides you with an overview of the environment management services (for association management).

Table C-2 Environment Management Services

Environment Management Services	Server, Client or Both
Initiate	Both
Conclude	Server
Abort	Both

VMD Support Services

Table C-3 provides you with an overview of the VMD support services (general VMD services).

Table C-3 VMD Support Services

VMD Support Services	Server, Client or Both
Unsolicited Status	Server
Status	Both
GetNameList	Server
Identify	Both
Rename	–
GetCapabilityList	Server

Variable Access Services

Table C-4 provides you with an overview of the variable access services.

Table C-4 Variable Access Services

Variable Access Services	Server, Client or Both
Read	Both
Write	Both
InformationReport	Both
GetVariableAccessAttributes	Server
DefineNamedVariable	–
DefineScatteredAccess	–
GetScatteredAccessAttributes	–
DeleteVariableAccess	–
DefineNamedVariableList	–
GetNamedVariableListAttributes	–
DeleteNamedVariableList	–
DefineNamedType	–
GetNamedTypeAttributes	–
DeleteNamedType	–

Domain Management Services

Table C-5 provides you with an overview of the domain management services.

Table C-5 Domain Management Services

Domain Management Services	Server, Client or Both
InitiateDownloadSequence	–
DownloadSegment	–
TerminateDownloadSequence	–
InitiateUploadSequence	–
UploadSegment	–
TerminateUploadSequence	–
RequestDomainDownload	–
RequestDomainUpload	–
LoadDomainContent	–
StoreDomainContent	–
DeleteDomain	–
GetDomainAttributes	–

Program Invocation Services

Table C-6 provides you with an overview of the program invocation services.

Table C-6 Program Invocation Services

Program Invocation Services	Server, Client or Both
CreateProgramInvocation	–
DeleteProgramInvocation	–
Start	–
Stop	–
Resume	–
Reset	–
Kill	–
GetProgramInvocationAttributes	–

Event Management Services

Table C-7 provides you with an overview of the event management services.

Table C-7 Event Management Services

Event Management Services	Server, Client or Both
DefineEventCondition	–
DeleteEventCondition	–
GetEventConditionAttributes	–
ReportEventConditionStatus	–
AlterEventConditionMonitoring	–
TriggerEvent	–
DefineEventAction	–
DeleteEventAction	–
GetEventActionAttributes	–
ReportEventActionStatus	–
DefineEventEnrollment	–
DeleteEventEnrollment	–
AlterEventEnrollment	–
ReportEventEnrollmentStatus	–
GetEventEnrollmentAttributes	–
AcknowledgeEventNotification	–
AttachToEventCondition	–
EventNotification	–
GetAlarmSummary	–
GetAlarmEnrollmentSummary	–

Semaphore Management Services

Table C-8 provides you an overview of the semaphore management services.

Table C-8 Semaphore Management Services

Semaphore Management Services	Server, Client or Both
TakeControl	–
RelinquishControl	–
DefineSemaphore	–
DeleteSemaphore	–
ReportSemaphoreStatus	–
ReportPoolSemaphoreStatus	–
ReportSemaphoreEntryStatus	–
AttachToSemaphore	–

Journal Management Services

Table C-9 provides you an overview of the journal management services.

Table C-9 Journal Management Services

Journal Management Services	Server, Client or Both
ReadJournal	–
WriteJournal	–
InitializeJournal	–
CreateJournal	–
DeleteJournal	–
ReportJournalStatus	–

File Access Services

Table C-10 provides you with an overview of the file access services.

Table C-10 File Access Services

File Access Services	Server, Client or Both
FileOpen	–
FileRead	–
FileClose	–
FileRename	–
FileDelete	–
FileDirectory	–

C.3 PICS Part Three: Parameter Conformance Building Blocks

Parameter Conformance Building Blocks

The CP 444 supports the following building blocks:

Table C-11 Parameter Conformance Building Blocks

Conformance Building Blocks	Support, Value
STR1	Yes
STR2	Yes
NEST	8
VNAM	Yes
VADR	No
VALT	No
VSCA	No
TPY	No
VLIS	No
REAL	No
AKEC	No
CEI	No

Parameter Meanings

If the corresponding bit is set, the following applies:

STR1 – The ”Array” data type is supported.

STR2 – The ”Structure” data type is supported.

VNAM – Access to variables via names is supported.

SIMATIC S7 Reference Literature

D

Literature Referenced in This Manual

/1/ ISO/IEC 9506-4, Industrial automation systems - Manufacturing Message Specification - Part 4 - Companion Standard for Numerical Control (1992-12-15), available from:

Normen der International Organization for Standardization (ISO)
Beuth-Verlag
Burggrafenstraße 6
10787 Berlin

/2/ MAP 3.0 1988; Manufacturing Automation Protocol, Version 3.0 including 1991 Supplement (1991-04-02), available from:

European MAP/TOP Users Group (EMUG) Secretariat
Institute for Industrial Information Technology Limited
Innovation Centre, University of Wales,
Swansea SA2 8PP
United Kingdom

/3/ *SProgramming with STEP 7* Manual

/4/ *S7-400/M7-400 Programmable Controllers, Hardware and Installation* – Installation Manual

/5/ *Configuring Hardware and Communication Connecttions STEP 7* – Manual

/6/ *System Software for S7-300 and S7-400, System and Standard Functions* – Reference Manual

Literature on SIMATIC S7

On the following pages, you will find a comprehensive review of:

- Manuals that you require for configuring and programming the S7-400
- Technical overviews that give you the fundamentals on SIMATIC S7 and *STEP 7*
- Reference books whose scope goes beyond S7-400

Manuals on Configuration and Programming

Extensive user documentation is available to assist you in configuring and programming the S7-400. You can select and use this documentation as required. Table D-1 provides you with an overview of the *STEP 7* documentation.

Table D-1 Manuals You Require for Configuring and Programming the S7-400

Title	Contents
Programming with STEP 7 *Manual*	The programming manual gives you the fundamentals on the design of the operating system and of an S7 CPU user program. For novice users of an S7-300/400 it provides an overview of the programming principles on which the design of user programs is based.
Configuring Hardware and Communication Connections STEP 7 *Manual*	The STEP 7 manual explains the principles for using the STEP 7 automation software and its functions. Novice users of STEP 7 and experienced users of STEP 5 are provided with an overview of the configuration, programming and setup of an S7-300/400. When working with the software, you can use the on-line help system, which provides detailed information on how to use the software.
Statement List (STL) for S7-300 and S7-400 *Programming Manual*	The manuals for the STL, LAD and SCL packages each comprise the user manual and the language description. To program an S7-300/400, you need only one of the languages, but, if required, you can switch between languages within a project. If you are using the language for the first time, it is advisable to use the manual to familiarize yourself with the programming principles. When working with the software, you can use the on-line help system, which provides detailed information on how to use the associated editors/compilers.
Ladder Logic (LAD) for S7-300 and S7-400 *Programming Manual*	
Structure Control Language (SCL)[1] for S7-300 and S7-400 *Programming Manual*	
GRAPH[1] for S7-300 and S7-400 Programming Sequential *Manual*	The GRAPH, HiGraph and CFC languages provide additional ways of implementing sequential function charts, state diagrams or graphical interconnections of blocks. Each manual comprises instructions for the user and a language description. If you are using the language for the first time, it is advisable to consult the manual to familiarize yourself with the programming principles. When working with the software, you can also use the on-line help system (except with HiGraph), which provides detailed information on how to use the editors/compilers.
HiGraph[1] for S7-300 and S7-400 Programming State Graphs *Manual*	
Continuous Function Charts[1] for S7 and M7 *Programming Continuous Function Charts* *Manual*	
System Software for S7-300 and S7-400 Systems and Standard Functions *Reference Manual*	The S7 CPUs have system and standard functions integrated in the operating system that you can use when programming in any of the available languages (STL, LAD and SCL). The manual provides you with an overview of the functions available with S7 and, for reference purposes, detailed interface descriptions for use in your user program.

1 Optional packages for S7-300/400 system software

Technical Overviews

Table D-2 contains technical overviews that provide you with an overview of the S7-400 and *STEP 7*.

Table D-2 Technical Overviews for SIMATIC S7 and STEP 7

Technical Overviews
S7-400 Programmable Controller *Configuration and Application*
S7-300/400 Programmable Controllers *Programming*

Reference Books

Table D-3 contains a selection of reference books you can order either directly from Siemens or from a bookstore.

Table D-3 List of Reference Books

Title of Book	Siemens Order Number	Order Number at Bookstore
MMS: A Communication Language for Manufacturing Project 7096 - CCE-CNMA - Volume 2 ESPRIT Consortium CCE-CNMA (Eds.) Springer-Verlag	–	ISBN 3-540-59061-7
Speicherprogrammierbare Steuerungen, Grundbegriffe Siemens-AG, Berlin und München, 1989	A19100-L531-F913	ISBN 3-8009-8031-2
SPS Speicherprogrammierbare Steuerungen vom Relaisersatz bis zum CIM-Verbund Eberhardt E. Grötsch Oldenbourg Verlag; München, Wien 1989	A19100-L531-G231	ISBN 3-486-21114-5
Speicherprogrammierbare Steuerungen SPS; Band 1: Verknüpfungs- und Ablaufsteuerungen; von der Steuerungsaufgabe zum Steuerungsprogramm Günter Wellenreuther, Dieter Zastrow Braunschweig (3. Auflage) 1988	–	ISBN 3-528-24464-X
Steuern und Regeln mit SPS Andratschke, Wolfgang Franzis-Verlag	–	ISBN 3-7723-5623-0

Index

Numbers

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W