

SIEMENS

Introduction
C79000-B8976-C049/01

1

The TF Model and the TF Services

2

The TF Interface on the
CP 143 TF

3

TF Variable Services

4

TF Domain and PI Services
Implementing a CIM Network

5

Supplementary Services

6

SINEC

Non-Open Services for
Serial Transfer

7

CP 143 TF with NCM COM 143 TF

Configuring and Testing the TF
Interface

8

PGLOAD

9

REQUEST EDITOR

10

Manual

Example Programs

11

6GK1970 1AC43-0AA1

Release 01
Volume 2 of 2

Appendix

A

TF Error Numbers

B

Protocol implementation
Conformance Statements (PICS)

Abbreviations

C

Index

D

List of Further Reading

E

SINEC is a trademark of Siemens AG
Siemens Aktiengesellschaft

Wir haben den Inhalt der Druckschrift auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so daß wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in der Druckschrift werden jedoch regelmäßig überprüft. Notwendige Korrekturen sind in den nachfolgenden Auflagen enthalten. Für Verbesserungsvorschläge sind wir dankbar.

We have checked the contents of this manual for agreement with the hardware 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 welcome.

Technical data subject to change.

Nous avons vérifié la conformité du contenu du présent manuel avec le matériel et le logiciel qui y sont décrits. Or, des divergences n'étant pas exclues, nous ne pouvons pas nous porter garants pour la conformité intégrale. Si l'usage du manuel devait révéler des erreurs, nous en tiendrons compte et apporterons les corrections nécessaires dès la prochaine édition. Veuillez nous faire part de vos suggestions.

Nous nous réservons le droit de modifier les caractéristiques techniques.

Technische Änderungen vorbehalten.
Weitergabe sowie Vervielfältigung dieser Unterlage, Verwertung und Mitteilung ihres Inhalts nicht gestattet, soweit nicht ausdrücklich zugestanden. Zuwiderhandlungen verpflichten zu Schadenersatz. Alle Rechte vorbehalten, insbesondere für den Fall der Patenterteilung oder GM-Eintragung.

Copyright © Siemens AG 1993
All Rights Reserved

The reproduction, transmission or use of this document or its contents is not permitted without express written authority. Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility or design, are reserved.

Copyright © Siemens AG 1993
All Rights Reserved

Toute communication ou reproduction de ce support d'informations, toute exploitation ou communication de son contenu sont interdites, sauf autorisation expresse. Tout manquement à cette règle est illicite et expose son auteur au versement de dommages et intérêts. Tous nos droits sont réservés, notamment pour le cas de la délivrance d'un brevet ou celui de l'enregistrement d'un modèle d'utilité.

Copyright © Siemens AG 1993
All Rights Reserved

SIEMENS

SINEC

CP 143 TF with NCM COM 143 TF

Description

C79000-B8976-C049-01

Note

We would point out that the contents of this product documentation shall not become a part of or modify any prior or existing agreement, commitment or legal relationship. The Purchase Agreement contains the complete and exclusive obligations of Siemens. Any statements contained in this documentation do not create new warranties or restrict the existing warranty.

We would further point out that, for reasons of clarity, these operating instructions cannot deal with every possible problem arising from the use of this device. Should you require further information or if any special problems arise which are not sufficiently dealt with in the operating instructions, please contact your local Siemens representative.

General

This device is electrically operated. In operation, certain parts of this device carry a dangerously high voltage.

Failure to heed warnings may result in serious physical injury and/or material damage.

WARNUNG !



Only appropriately qualified personnel may operate this equipment or work in its vicinity. Personnel must be thoroughly familiar with all warnings and maintenance measures in accordance with these operating instructions.

Correct and safe operation of this equipment requires proper transport, storage and assembly as well as careful operator control and maintenance.

Personnel qualification requirements

Qualified personnel as referred to in the operating instructions or in the warning notes are defined as persons who are familiar with the installation, assembly, startup and operation of this product and who possess the relevant qualifications for their work, e.g.:

- Training in or authorization for connecting up, grounding or labelling circuits and devices or systems in accordance with current standards in safety technology;
- Training in or authorization for the maintenance and use of suitable safety equipment in accordance with current standards in safety technology;
- First Aid qualification.

Contents

1	Introduction	1-1
1.1	Significance of the SINEC Technological Functions	1-3
1.2	Notes on Using the Manual and Overview of the Contents of Volumes 1 and 2	1-5
1.2.1	Chapter Overview - Volume 2	1-5
1.2.2	Conventions Used in the Manual	1-9
1.2.3	User Groups and Topics	1-10
2	The TF Model and the TF Services	2-1
2.1	The Range of Performance of SINEC TF	2-2
2.1.1	Communication Requirements	2-2
2.1.2	Overview of the TF Services in SINEC H1-TF	2-3
2.2	SINEC TF Communication Model	2-5
2.2.1	Overview of the Architecture	2-5
2.2.2	The TF View of the Programmable Logic Controller	2-7
2.2.3	The Virtual Manufacturing Device (VMD)	2-9
2.2.4	Domain	2-10
2.2.5	Program Invocation (PI)	2-11
2.2.6	Variable	2-12
2.2.7	Application Association	2-15
2.2.8	Client and Server Associations	2-19
2.3	Selecting TF Services	2-21
2.3.1	VMD Services	2-21
2.3.2	Domain Services	2-23
2.3.3	Program Invocation (PI) Services	2-28
2.3.4	Variable Services	2-30
2.3.5	Application Association Management	2-34
2.3.6	Clock Services	2-36
2.3.7	Serial Transfer	2-37

3	The TF Interface on the CP 143 TF	3-1
3.1	The Principle of the TF Link PLC - CP	3-2
3.1.1	Using the Handling Blocks on the CP	3-4
3.2	General Client Interface for Calling TF Services	3-6
3.2.1	Job Buffer	3-6
3.2.2	Sequence on the Client Interface	3-10
3.3	General Server Interface	3-14
4	TF Variable Services	4-1
4.1	Basics of the Variable Services	4-2
4.1.1	Description and Management of Variables	4-2
4.1.2	Scope of Variables in a SIMATIC S5 Programmable Logic Controller	4-4
4.1.3	Checklist for the Application	4-9
4.2	Service Description	4-10
4.2.1	Read Variable (Client)	4-10
4.2.2	Read Variable (Server)	4-16
4.2.3	Write Variable (Client)	4-17
4.2.4	Write variable (server)	4-25
4.2.5	Information Report (Client)	4-26
4.2.6	Information Report (Receiver)	4-32
4.3	Read and Write Variable with the Option of Addressing via a Free Format Address	4-33
4.3.1	Client Interface	4-34
4.3.2	Server Interface	4-39
4.4	TF Data Types in SIMATIC S5	4-40

5	TF Domain and PI Services Implementing a CIM Network	5-1
5.1	Domain Services	5-2
5.1.1	Load Domain Content	5-8
5.1.2	Store domain content	5-15
5.1.3	Delete Domain Content (Client)	5-20
5.1.4	Get Domain Attributes (Client)	5-22
5.1.5	Domain Services (Server)	5-28
5.2	Program Invocation Services	5-29
5.2.1	PLC Program Structure, Status Transitions	5-30
5.2.2	General Sequence of a Status Change	5-37
5.2.3	Interface of the PLC Program to the PI Services on the Server	5-39
5.2.4	Start-up, Installation	5-45
5.2.5	Create Program Invocation (Client)	5-46
5.2.6	Create Program Invocation (Server)	5-50
5.2.7	Delete Program Invocation (Client)	5-51
5.2.8	Delete Program Invocation (Server)	5-54
5.2.9	Start, Stop, Resume, Reset, Kill Program Invocation and Local Program Stop (Client)	5-55
5.2.10	Start, Stop, Resume, Reset, Kill a Program Invocation (Server)	5-58
5.2.11	Points to Note when Starting and Stopping the PLC using the System PI	5-59
5.2.12	Get Program Invocation Attributes (Client)	5-60
5.2.13	Get Program Invocation Attributes (Server)	5-64
6	Supplementary Services	6-1
6.1	Application Association Management	6-2
6.1.1	Definition of Application Associations	6-2
6.1.2	Connection Establishment	6-5
6.1.3	Connection Termination	6-8
6.1.4	Special Connections	6-9
6.2	VMD Services for Virtual Manufacturing Devices	6-10
6.2.1	Status of the Virtual Device (Client)	6-11
6.2.2	Status of the Virtual Device (Server)	6-13
6.2.3	Unsolicited VMD Status (Initiator)	6-17

6.2.4	Unsolicited VMD Status (Receiver)	6-18
6.2.5	Identify Virtual Manufacturing Device (Client)	6-19
6.2.6	Identify VMD (Server)	6-21
6.3	Configuration Jobs	6-23
6.4	Clock Services	6-28
6.4.1	Network Topology, Clock Master/Slave Functions	6-31
6.4.2	Accuracy	6-35
6.4.3	How the Clock Functions	6-36
6.4.4	Several CP 143 TFs on One SINEC H1 Bus without a SINEC Time Transmitter	6-38
6.4.5	CP 143 TF on a SINEC H1 Bus with SINEC Time Transmitter	6-39
6.4.6	Setting and Reading the Time in the Programmable Logic Controller	6-40
6.4.7	Setting and Reading the Time with COM 143	6-45
6.4.8	Restrictions / Tips	6-48
6.4.9	Accuracy	6-49
7	Non-Open Services for Serial Transfer	7-1
7.1	Overview of the Functions and Services	7-2
7.2	Read Byte String (Client)	7-4
7.3	Write Byte String (Client)	7-7
7.4	Read/Write Byte String (Server)	7-13
7.5	Transparent Data Exchange (Client)	7-17
7.6	Transparent Data Exchange (Server)	7-21
7.7	Addendum to Transparent Data Exchange	7-24
7.7.1	Status Word of the TRADA on the Server	7-24
7.7.2	Example of a Program for Evaluating the Bits of the ANWZ on the TRADA	7-25

8	Configuring and Testing the TF Interface	8-1
8.1	Overview	8-2
8.2	Configuring the TF interface	8-3
8.2.1	Edit ... Application Associations	8-3
8.2.2	Edit VMD Variables	8-17
8.2.3	Edit Fileserver AA	8-20
8.2.4	Edit VMD Configuration	8-21
8.3	Documenting the Configuration	8-23
8.3.1	Application association list	8-23
8.3.2	VMD list	8-23
8.3.3	Server list	8-23
8.3.4	All	8-23
8.4	Transferring Configured Data	8-25
8.4.1	Transfer Transfer Database	8-25
8.5	Testing the TF interface	8-26
8.5.1	Outputting Status All	8-27
8.5.2	Output Single Status	8-33
8.5.3	Trace Functions	8-34
8.5.4	Job Test	8-35
9	PGLOAD	9-1
9.1	Overview	9-2
9.1.1	Adapting Programmable Logic Controllers to the Process with PGLOAD	9-2
9.1.2	Range of Functions	9-4
9.2	Description of the Tool	9-5
9.3	Functional Description	9-7
9.3.1	System Configuration and Device Functions	9-7
9.3.2	Configuring Links and Selection Functions	9-8
9.3.3	Transfer Functions	9-9
9.3.4	Host Computer Functions	9-11

9.4	PGLOAD - Application	9-13
9.4.1	PGLOAD Initialization	9-13
9.4.2	Link Configuration / Server Selection	9-15
9.4.3	Link Configuration / PLC links	9-18
9.4.4	Using Transfer Functions	9-20
9.4.5	Using Host Computer Functions	9-23
10	Request Editor	
	User-Friendly User Interface for Generating Job Buffers	10-1
10.1	Overview	10-2
10.1.1	Mode of Operation and Requirements	10-2
10.1.2	Meaning of the Job Buffer	10-3
10.2	Description of the Request Editor	10-5
10.3	Request Editor Init	10-7
10.4	Specifying the Job Buffers for TF Services	10-9
10.4.1	Create Job buffer	10-9
10.4.2	Type Selection Screen Form for TF and Other Services	10-12
10.4.3	Variable Services	10-16
10.4.4	Domain Services	10-30
10.4.5	Program invocation services	10-37
10.4.6	VMD Services	10-48
10.4.7	Transparent Data Exchange (non-open services)	10-53
10.4.8	Other Jobs	10-61
10.5	Displaying and Evaluating the Job Buffer Overview	10-70
10.6	Delete Data Block	10-73
10.7	Documenting Job Buffers	10-73
10.7.1	Documentation All	10-73
10.7.2	Documentation Overview	10-73
10.7.3	Documentation Job Buffers	10-73

11	Example Programs	11-1
11.1	Overview and Requirements	11-1
11.2	Example 1: Using Variable Services	11-3
11.2.1	Task	11-3
11.2.2	Defining Variables	11-5
11.2.3	TF Services Required	11-7
11.2.4	Creating the Client Configuration File	11-9
11.2.5	Creating the Server Configuration File	11-13
11.2.6	Creating the Job Buffers with the Request Editor	11-17
11.2.7	PLC Programs	11-25
11.2.8	Starting Up	11-41
11.2.9	Monitoring the Process at the PG	11-41
11.3	Example 2: Using the Domain and Program Invocation Services	11-42
11.3.1	Task for the Domain Services	11-42
11.3.2	Tasks for the Program Invocation Services	11-43
11.3.3	Preparing Programs and Data	11-45
11.3.4	Executing Domain and PI Services	11-61
11.4	Example 3: Transparent Data Exchange with Acknowledgement (T-DQ)	11-67

Appendix

A	TF Error Numbers used by the CP 143	A-1
A.1	Preface	A-1
A.2	Errors Arranged According to Service Groups	A-2
A.2.1	Non-Service Dependent Errors (NONS)	A-2
A.2.2	Errors in the General Services (GEN)	A-5
A.2.3	Errors in Application Association Management (APPL)	A-7
A.2.4	Errors in the Variable Services(VAR)	A-8
A.2.5	Errors in the Domain Services (DOM)	A-12
A.2.6	Errors in the Program Invocation Services	A-15

A.2.7	Errors during Serial Transfer(SER)	A-17
A.2.8	Errors during Configuration (CONF)	A-18
B	Protocol Implementation Conformance Statements (PICS)	B-1
C	Abbreviations	C-1
D	Index	D-1
E	Further Reading	E-1
□		

1 Introduction

The second volume of the "CP 143 TF with COM 143 TF" manual describes the protocol and the services for open, heterogeneous communication with the CP 143 TF communications processor.

The CP 143 TF communications processor implements the connection of SIMATIC S5 programmable logic controllers to the cell and area network SINEC H1/H1FO.

Within industrial automation, the protocol for the application layer (layer 7) of the OSI reference model is fixed uniformly in the SINEC communications system for all automation systems; the SINEC technological functions.

The manual describes the services and scope of the SINEC technological functions (SINEC TF) for SIMATIC S5 programmable logic controllers. Following this, the implementation of SINEC TF on the CP 143 TF and the interface to the S5 user programs are explained in detail.

This second volume contains information on the following topics:

The SINEC technological functions (TF) model

- Criteria for selecting the TF services
- General client and server interface on the CP 143 TF.
- Detailed description of the individual TF services
- Configuring and testing the CP 143 TF with the COM 143 TF configuration tool.
- PG support of TF domain and PI services with the PGLOAD tool.
- User-friendly formulation of TF jobs with the PG tool TF REQUEST EDITOR
- Examples of incorporating the TF services in SIMATIC S5 user programs

Volume 2 of the manual builds on the information from volume 1 which describes the basics of communication with the CP 143 TF and the basics of configuration with the COM 143 TF configuration tool

The introduction to Volume 1 also contains a simple example of using the TF services.

1.1 Significance of the SINEC Technological Functions

Overview

The SINEC technological functions (TF) form the application protocol for communication in a heterogeneous automation network. They provide the user with services to allow problem-free interaction between different automation components (e.g. PLC, NC controls, robots, open-loop controllers, PCs, mini-computers and host computers etc.) in the cell and area network SINEC H1/H1FO. TF services also allow the exchange of information (messages) using a standard language. The standardization is intended to permit the implementation of open systems, reducing the time and expense required for the software engineering.

Definition based on MMS

The basis on which the TF services are defined is the only international standard for application protocols in the area of industrial automation: ISO 9506, MMS (Manufacturing Message Specification).

CIM is supported

The SINEC technological functions are a further step in the direction of CIM (Computer Integrated Manufacturing), i.e. computer controlled, fully automatic manufacturing, since the integration of automation components in a CIM network is impossible without communication.

Advantages of SINEC TF

The uniform, standardized language for exchange of information has the following advantages:

- > The use of TF services for the exchange of information makes the job of the programmer much easier. The protocol "hides" the specific characteristics of the end system behind a standardized, uniform representation of the system and the data. This means that negotiations between programmers regarding system structures and methods of representation are no longer necessary. The programmer can concentrate on implementing his own particular tasks.
- > The simple integration of components of other manufacturers is made possible by TF.

- The protocol is independent of the underlying communication system: SINEC L2, SINEC H1 or SINEC MAP. This provides flexibility in program development (the system grows with the requirements of the user) and also means a reduction of training costs.
- Gateways can be implemented without problems.
- By using TF, the time and expense of software development can be greatly reduced.

1.2 Notes on Using the Manual and Overview of the Contents of Volumes 1 and 2

1.2.1 Chapter Overview - Volume 2

Chapter 2

The TF model and the TF services

This chapter describes the model underlying the SINEC technological functions. It illustrates the uses of defining an abstract model for a programmable logic controller, the virtual manufacturing device known as VMD.

The chapter also introduces the TF services in sufficient depth for you to select the services you require for your task.

Chapter 3

The TF interface on the CP 143 TF

This chapter describes the modelling of the TF services on the CP 143 TF. You will also learn how to use the TF interface of the CP 143 TF.

Chapter 4

TF variable services

This chapter contains a description of the interface for the TF variable services both for the client and server functions of a PLC.

Chapter 5

TF domain and PI services: services for implementing a CIM network

This chapter describes the use of the TF services for controlling, monitoring and supplying data to the PLC.

Chapter 6

Supplementary services

This chapter describes the handling of application associations on the CP 143 TF and the VMD services with which information can be obtained about a PLC. Additional functions such as configuration jobs and the clock functions are also described.

Chapter 7

Non-open services for serial transfer

Serial data transfer is available for simple communication tasks. This chapter describes the interface for these services.

Chapter 8

Configuring and testing the TF interface

In this chapter you will find a description of the configuration, test and documentation functions required for initiating and operating application associations and the TF services via these associations.

Chapter 9

The PGLOAD tool

The PGLOAD tool provides user-friendly and nevertheless simple functions for addressing programmable logic controllers (PLCs) via the MMS standard TF interface.

PGLOAD is also required to structure a PLC within the framework of the TF services with domain and program invocation objects.

This chapter explains how to use the PGLOAD tool for the following purposes:

- to supply PLCs with programs using the TF domain services directly or via the file server to match the PLC to the current tasks in the process
- to monitor and control PLCs using TF program invocation services

Chapter 10

The REQUEST EDITOR tool

The REQUEST EDITOR tool supports you when creating the job buffer you require to program TF communication services on your SIMATIC programmable logic controller.

This chapter explains the functions and how to use the tool. It is also suitable as an introduction for first-time users and as a reference work when configuring various TF services.

Chapter 11

Examples of programs

This chapter will familiarize you with the TF interface on the SINEC H1 bus system as designed for SIMATIC S5. The services and assignment of parameters to the CP 143 using the COM 143 software package are dealt with in detail.

Appendix

A TF Error Numbers

B Protocol Implementation Conformance Statement PICS

C Abbreviations

D Index

E Further Reading

Overview of Volume 1:

Description of basic CP configuration, the CP transport interface and general information about the construction, installation and operation of the CP 143 TF.

"COM 143 TF Configuration Tool" Supplement

An overview of the functions and screens.

1.2.2 Conventions Used in the Manual

The manual uses the following symbols in the text:

✓ This character indicates an action for you to undertake.



This character highlights important notes and dangers.

mm Dimensions in diagrams and scale drawings are specified in millimetres.

M xx

This note in the margin indicates the screen number which you can refer to in the "COM 143 TF Configuration Tool" supplement.

Prior requirements

To understand the examples you should have

- STEP 5 programming experience and
- basic knowledge of using handling blocks (HDBs). A description of the HDBs can be found in the manual for your programmable logic controller or in separate descriptions for the programmable logic controllers.

Training courses

Siemens provides comprehensive training opportunities for SINEC users.

For more detailed information contact your Siemens office.

1.2.3 User Groups and Topics

The CP143 TF with COM143 TF manual is intended for the following users:

- Users wishing to configure SINEC H1/H1FO cell and area networks with SIMATIC S5 stations
- Programmers of communicating application programs for SIMATIC S5 programmable logic controllers
- Personnel installing SINEC H1FO cell and area networks with SIMATIC S5 stations

The manual contains the information required for the following:

- Assigning parameters to the CP 143
- Configuring links on the transport layer and application layer
- Configuring communication objects such as variables, domains and program invocations
- Operating the interfaces supported by the CP 143
- Configuring job buffers in the user programs of the SIMATIC S5 programmable logic controller for communication on the application layer
- Configuration of host computer functions for domain and program invocation services on the programming devices

2 The TF Model and the TF Services

To allow you to use the TF services of the CP 143 TF, this chapter explains the communications model and the range of services of the SINEC technological functions.

2

If you are familiar with the TF communication model, you can skip this chapter. The description of the TF services will nevertheless be useful in helping you to select the services for your tasks.

At the end of this chapter you will have learnt about the following:

- the architecture of the communication system
- the model of SINEC TF with its objects
- the TF services supported by the CP 143 TF
- the terminology used in the TF model
- the uses of the TF services
- the TF services you will be able to use for your task.

Note:

Only TF specifications actually implemented on the communications processor are described.

2.1 The Range of Performance of SINEC TF

2.1.1 Communication Requirements

Message-oriented communication,

Within industrial communication, a distinction is made between data-oriented and message-oriented communications protocols. While data-oriented protocols handle pure bit or byte streams, message-oriented protocols handle the content. The receiver of a message must perform a **service described in the protocol**. Message-oriented communication therefore goes beyond the simple "transfer of data"

Open communication

The basic idea behind "open communication" is to allow programmable logic controllers of different manufacturers to communicate with each other. By using a common specification, devices from different manufacturers can be integrated into one system. The TF specification describes how the message is exchanged.

It also specifies certain types of message to allow uniform and comprehensible transfer of service requests.

"Openness" is guaranteed by standardizing services, objects, attributes, parameters and statuses with SINEC technological functions (TF). The SINEC protocols allow SIEMENS subnetworks to be integrated in networks with the international manufacturing automation protocol (MAP) architecture therefore allowing an open system.

An example

A measured value is an object belonging to the class of variables. This object can be addressed using the variable services "read variable" and "write variable". The TF services ensure an understandable transmission of the object regardless of its format and analysis in the end system.

2.1.2 Overview of the TF Services in SINEC H1-TF

SINEC TF is made up of the services conforming to MMS and to the open services. The latter are only available in the SINEC AP protocol architecture described here.

2

TF services:

➤ VMD services

With the services for the **Virtual Manufacturing Device (VMD)**, information about the characteristics and the status of a VMD can be requested (which services the device can perform, which objects exist etc.).

➤ Domain services

Domains are task-oriented program or data areas. Using the domain services, programs and data can be transmitted. The transmission can also be initiated by a third party, for example to transfer programs from a fileserver and load them on a PLC.

➤ Program invocation services

The program invocation models an executable program section. Services are specified for creating, starting, stopping and deleting program invocations.

➤ Variable services

Variable services are services for writing and reading the values of variables. These data can range from simple (integer) to complex (structures). A uniform syntax is defined to describe data structures so that language barriers occurring in the data type description are avoided (in the example: the S5 data block can be read in the host computer).

➤ Application association management

Applications wanting to communicate with each other can initiate, maintain and terminate a logical link, known as an application association.

Additional services only available for SINEC H1 (non-open services):**> Clock services**

The time on the clock chip on the CP can be set and synchronized. The time can also be transmitted as a SINEC synchronization frame.

> Serial transfer

For simple data transfer, the serial transfer services are available. Data is transferred without address information and without structure information. As explained in the introduction, this transfer is data-oriented as opposed to message-oriented.

2.2 SINEC TF Communication Model

2.2.1 Overview of the Architecture

2

The diagram below illustrates the architecture of the communication system with the application above it. The meaning of the layers of the model is explained in the introduction in Volume 1 of this manual.

The description in Volume 2 involves the access to communication via the TF interface. As can be seen in the diagram, not only the application program in the programmable logic controller, but also the programming device uses the communications access via the TF interface with special services (PGLOAD).

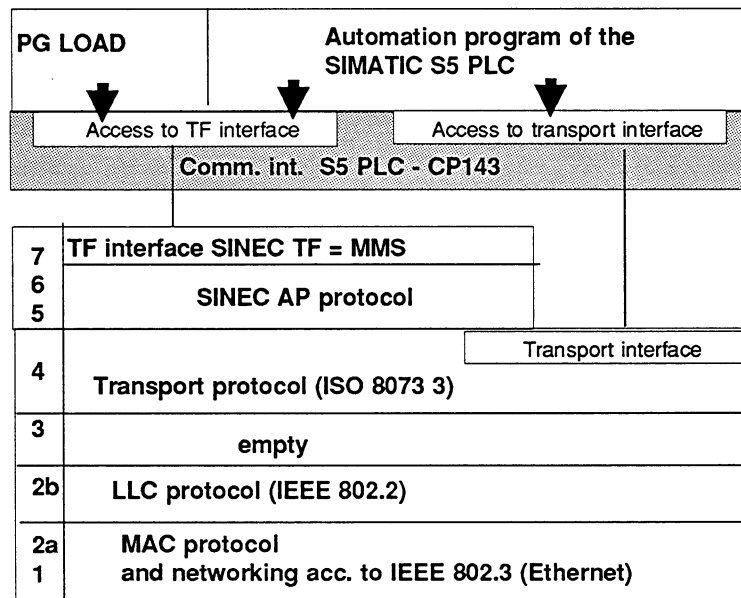


Fig. 2.1 Interface and Protocol Profile of the Communications Processor SINEC CP143 TF

Key to Figure 2.1 (for descriptions refer to the introduction in Volume 1)**TF: Technological Functions**

The SINEC TF interface handles the encapsulation and processing of the TF PDUs and the service-oriented execution of a job or an acknowledgment /4/. The SINEC TF interface is designed so that it has no application protocol-specific elements which cannot be modelled on the MAP 3.0/MMS application protocol. This means that the user software also runs identically on the MAP protocol stack providing conformity rules are adhered to.

PGLOAD for

- > Loading/deleting the PLC
- > Host computer functions

AP: Automation Protocol

SINEC AP handles the protocol for layers 5-7 /3/.

Transport:

Transport layer for SINEC H1 based on the ISO transport protocol.

MAC: Media Access Control

2.2.2 The TF View of the Programmable Logic Controller

Intention

This section explains the view adopted by the application when using the TF communication model and which possible methods of device communication result. In this respect, the functions, model and terminology used are important.

Programmable logic controllers consist of objects

A real programmable logic controller is modelled on a virtual manufacturing device (VMD) and its objects with the intention of uniform task-oriented communication. The device and its services are only available using communication with these objects.

The following objects are defined:

- VMD (Virtual Manufacturing Device):
This is the programmable logic controller (PLC) itself
- Domain
This is a loadable data or program area on the device (PLC)
- Program invocation (PI)
This is an executable, controllable task on the device (PLC)
- Variable
This is a data area on the device (PLC) that can be read or written to
- Application association
This is the communication channel between two applications

The following sections explain these objects and how they are modelled on the SIMATIC S5 programmable logic controller.

TF objects can be addressed using services

During communication between an application process and an application process on a different device, the locally existing objects must be made available to the other device using services. Services are operations that can be performed on the objects listed above. Example: a measured value is an object belonging to the class of variables. Defined services for this class are "read variable", "write variable", and "information report".

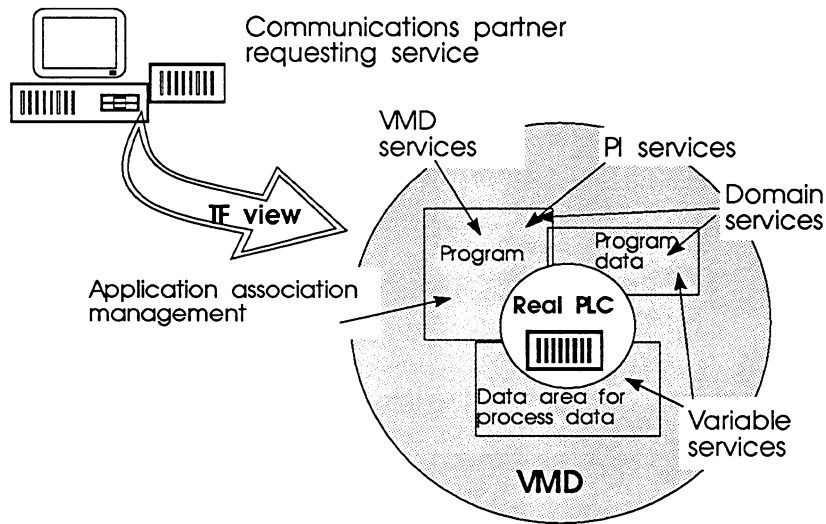


Fig. 2.2 The TF/VMD View of the Real PLC

Types of object

The communications objects in TF can either be static or can be created dynamically due to the communication. Static means that the communication objects must be configured. Dynamic objects are created through the communication.

> Static types of object

Static communications objects are created when the device starts up. These structures remain unchanged in the application area during operation. They can neither be modified nor deleted during operation (see VMD-specific objects).

> Dynamic types of object

Dynamic communications objects are created while the device is operating. These objects can be loaded, modified and deleted during operation. Access to these objects therefore depends on them existing (see domain and link-specific objects).

TF services and object classes

The following sections describe the object classes assigned to the individual TF services.

2.2.3 The Virtual Manufacturing Device (VMD)

Purpose

The virtual manufacturing device (VMD) represents a standardized model or simulation of the physical device. It is described by the objects it contains and the characteristics of the physical device which it models with a standardized view. It therefore provides an interface via the VMD services for standardized interrogation of the device status and device characteristics.

2

SIMATIC S5 and VMD

The essential function of the communications processor (CP) is to model the VMD on the programmable logic controller. The conversion of the functions (required by the protocol) of the VMD on the programmable logic controller (PLC) is therefore performed in the communications processor.

Within SIMATIC S5, each communications processor along with its programmable controller (the PLC in which the communications processor is plugged in) is considered as a single VMD. A VMD always contains a communications processor and (in the case of multiprocessor PLCs) up to four CPUs. Regardless of the number of CPUs, it is, however, possible to use several CPs in a PLC rack. Since the communications processors all operate independently of each other, each can be considered as a VMD itself.

2.2.4 Domain

Purpose

A domain is the model of programs, program sections or program data used for communication. Domains can be loaded and are therefore normally generated by transferring them to the programmable controller. The names of the domains are unique within a VMD. For the SIMATIC S5 PLCs, this means:

Domains can be the following:

Logical management units for variables

A domain as a logical management unit defines a scope (area of validity) for variables. The name of a variable then consists of the domain name and variable name.

Containers for program code and/or data

A domain can consist of program code or data. In the context of TF, these contents are transparent, i.e. their meaning is only known to the applications which use the domain (e.g. not to a fileserver).

SIMATIC S5 and domains

A domain normally consists of "STEP 5" blocks, stored by the user in a program file using the S5-DOS programming package "LAD, CSF, STL". Additional variables can also be assigned to a domain (see Variable). In addition to these "loadable" domains, there are also "static" domains. Static domains can neither be loaded nor deleted.

Up to eight domains can be loaded on a SIMATIC S5 programmable logic controller.

2.2.5 Program Invocation (PI)

Purpose

A program invocation is the grouping of all domains into an executable program. The domains can either be loaded ("loadable" domains) or they are always present ("static" domains). Program invocations are therefore logical units capable of controlling user programs (put together in a PI) via PI services. They can, for example, be started or stopped.

2

PIs have statuses

PI objects are characterized by their statuses and the status changes brought about by the PI services and by running the program.

Identification and assignment to domains

Program invocations are identified uniquely within the VMD by names. Program invocations existing simultaneously can, if necessary, use the same domain(s).

SIMATIC S5 and PIs

SIMATIC S5 PLCs distinguish between a system PI and a user PI. The system PI is by definition always present. By addressing this PI, start/stop instructions can be issued to the PLC. By means of the user PI, a user program loaded on the PLC is addressed. This user program is formed by the loaded domains. It is also possible, if the domain services are not used, to address the PLC user program as a user PI.

2.2.6 Variable

Purpose

Variables are objects that model the data of a user program. Variables are identified by names and contain a description of their structure. The description allows a standardized representation of the data, uniform throughout the system. The structure of the data can be either simple or complex. Via the communications system, variables can be read or written to.

Variable characteristics

Variables are identified by the following characteristics:

- Variable name:
Each variable has an identifier (ASCII string), with which the object is accessed.
- Variable description
The structure description of a variable is entered in an object description.
- Scope:
Variables are assigned to a particular scope. Scope means the assignment of a variable to a VMD, to an application association or to a domain which represents a form of "cocoon" around the variable. Access is only possible to the variable by specifying the name of the "cocoon". Within a scope, variable names must be unique.
- Access rights:
Both read and write access is possible via the network. This means that the values of variables can be modified by another station on the network. If this is undesirable, the write access can be disabled.

SIMATIC S5 and variables

In a SIMATIC S5 PLC, data of a STEP 5 user program is accessed via variables that have a name. The following features are used for managing and handling variables in the S5 PLCs:

- S5 address:
A variable in the SIMATIC PLC is always at a fixed S5 address. This address must be located in the data block area (or extended data block area).
- Status word address:
Each variable in the SIMATIC S5 device is assigned a status word (condition codeword) address. This address contains information about access to the variable via the network.
In the SIMATIC PLC, it is also possible to prevent access to the variable at certain times by manipulating the status word.
- Interface number:
This attribute specifies which interface number must be used on the communications processor to be able to access the variable.
In multiprocessor PLCs, this allows the variables to be distributed on various CPUs.
- Local and remote variables
With the attributes listed above, local variables can be defined and specified. Local variables are objects which exist in the local station. Other stations can access these objects via the network. Remote variables are variables on a different device which the local device can read or write. Structure information must be stored for this access. The definition of local and remote variables is therefore supported by the configuration tools.
- Static and dynamic variables
Static local variables are programmed with the COM 143 TF configuration tool ("VMD variable editor"). Dynamic variables generated as a result of communication are known as "domain-specific" variables (i.e. variables whose existence depends on the existence of a domain). These are defined using the PGLOAD tool (component of COM 143 TF).

Variants of variable structures

The following variants in the structure description of variables must be distinguished:

- Variable with **standard data type**: these elements are of a predefined data type.
- Variable record: a list of differently structured components of any type.
- Variable array: a list of elements with the same structure.

Access protection mechanisms for variables

In automation systems, access protection is often necessary for the safe operation of equipment. The CP 143 TF provides the following types of access protection:

- explicit protection with a R/W identifier for each individual variable set when configuring variables
- implicit protection using the scope of a variable
- temporary protection by means of the variable status word.

2.2.7 Application Association

Purpose

From the point of view of the user, communication with the application processes of the communications partners takes place via logical channels (application associations). These application associations define the view of the communications partner and its automation task. VMD objects can be addressed only via application associations.

Prior to the communication, the user must specify which automation task is to be addressed via which application association. The PDUs therefore contain the meaning of the message as well as the address information allowing the simulation on the application on the server side.

Initiation, use and conclusion of application associations

During the initiate phase, a link request is indicated to the remote application process (initiate service). The link establishment request includes the services to be used in the data transfer phase, the maximum frame size, the number of parallel services (context) and the required type of link as well as any other options required on the link.

If the remote application process agrees to the link establishment request, it sends a confirmation of the link establishment request to the issuer of the job. Following this, both application processes are in the data transfer phase and can communicate with each other according to the agreed restrictions (context).

An existing link can be monitored for availability by the link supervision. The specified supervision times must be the **same** on both the local and remote end points.

A link is cleared by a link termination function. Following termination, data exchange can only be resumed following a new link establishment.

SIMATIC S5 and application associations

With SIMATIC S5 PLCs the communication path between two applications is described by the application association.

The name of an application association must be unique within the network. When an application association is configured, the specification of the link name is supported by the configuring tool. This makes use of the fact that the access to communication with SIMATIC S5 is via the interface number and job number. This procedure is familiar from the transport interface of the CP 143 TF.

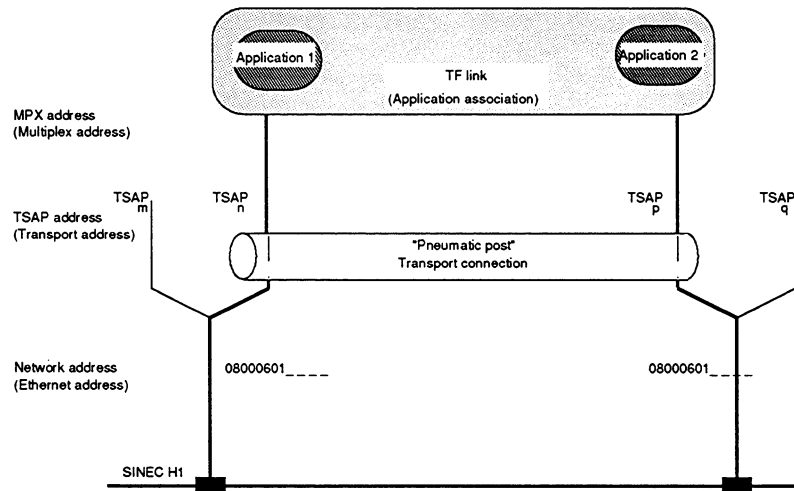


Fig. 2.3 Application Associations - Access to the Transport Services

An application association between two CP 143 TF communications processors of stations X and Z with transport access points A and H and the application access Mn is described uniquely in both stations using the following parameters:

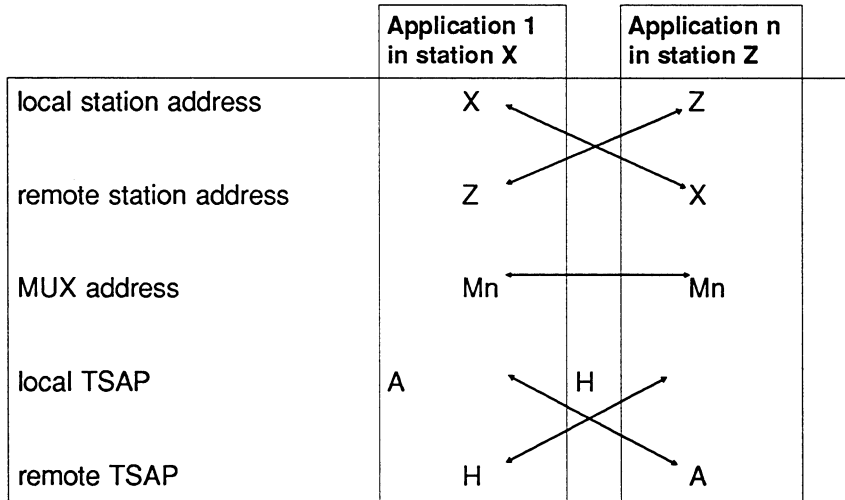


Fig. 2.4 Parameters for an Application Association

Just as the transport links, these parameters are stored in the data link blocks and SYSID blocks on stations X and Z. The following figure illustrates the parameter storage for application associations:

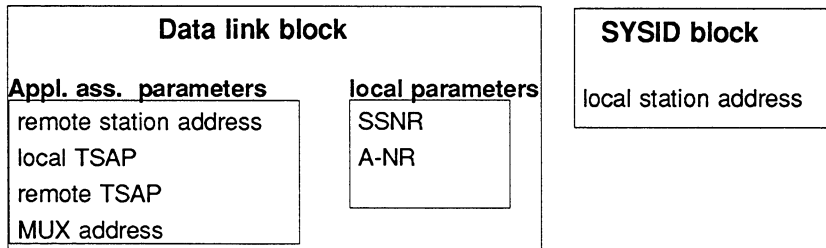


Fig. 2.5 Data Link Block and SYSID Block with TF Links

Handling application associations with SIMATIC S5

In the client and server role, the CP 143 provides the following TF services:

- initiate application association
- abort application association

In the server role, the following service is provided:

- conclude application association

The TF services for managing application associations are executed via the CP 143 in the main without being triggered by the user program. The required parameters for executing the services are entered with COM 143 when defining the application associations and saved on the communications processor. Depending on the type of link (static or dynamic), the application associations are initiated when the modules are started or when a job is initiated for productive communication.

An explicit client interface for initiating and concluding application associations by the PLC user program is not provided. The PLC user program can only abort an existing application association by calling the CP handling block "RESET".

2.2.8 Client and Server Associations

The principle

The use of application associations with SINEC TF is based on the client/server principle. This principle defines two communications partners:

- For communication purposes, a client is an application process which uses the functions of a virtual manufacturing device (VMD) of a remote application process.
- The server is the application process that makes the functions of its virtual manufacturing device (VMD) available to the client. The service can be requested or may be provided spontaneously (process values may be "reported" depending on the technological process).

Example

A host computer requests the transfer of a process value in a PLC using the read variable TF service. It uses the application association to the PLC process in device Y. The PLC process in device Y is the server which provides the read variable service to the host computer.

Changing roles of a process

An application process can function both as client and server. This means that the process can request services (client) and provide services (server). This reflects the typical integration of manufacturing devices in the hierarchical organization of a CIM network.

The following figure illustrates the layers of a CIM network. Communication between devices takes place both within a layer and between the different layers.

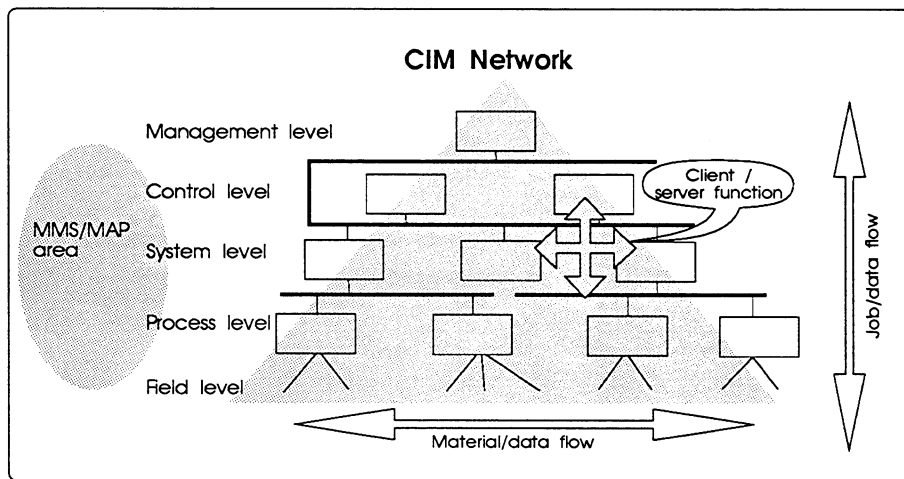


Fig. 2.6 Hierarchical Structure of a CIM Network; Devices with Client and Server Role

Client and server functions with SIMATIC S5

The TF communications functions of SIMATIC S5 devices are designed to support client and server functions. For your application, the following is available:

- the client interface, to formulate service requests within the PLC program
- the server functions for communications services provided in the form of a CP program to be able to execute TF service requests

2.3 Selecting TF Services

2.3.1 VMD Services

2

Meaning

The VMD object forms a cocoon around the other objects of a VMD and its services. Externally, it must be considered the central access point via which information about the characteristics and statuses of the device can be obtained. In keeping with this, the services are primarily information functions.

The general TF services for virtual manufacturing devices allow a client to request information about the status or attributes of a virtual manufacturing device (VMD) on the server. In some circumstances, the server can report the status to a client without a request. The information can be processed further on the client, for example to provide an overview of the whole system status in a control room.

Overview

The CP 143 can process the following general services for virtual manufacturing devices (VMD) both in the role of client and server.

- Get the status of a virtual device (status)
- Report the status of a virtual device (unsolicited status)
- Identify virtual device

The following general services can only be processed by the CP 143 TF in the role of server:

- Get name list and
- Get capability list

Description of the VMD services

Status (get the status of a virtual device)

Using the "Status" service, a client requests information about the physical and logical status of the virtual manufacturing device managed on the server. The server sends the requested information (e.g. whether the "real" manufacturing device or the communications processor of the server is in the RUN or STOP mode or whether the PLC and CP are not synchronized) in the acknowledgment.

Unsolicited status (of the virtual device)

With this job, a server program can report the logical and physical status of the VMD on its own initiative. The CP 143 can receive this spontaneously transmitted information (client function) or send the information (as server).

Identify (virtual device)

A client can request information about the attributes of a virtual manufacturing device (VMD) using the "Identify" service. These attributes can, for example, include the identifier of the vendor of the manufacturing device, the device ID (for the CP 143 the module ID) and the version of the communications processor.

Get name list and get capability list

The TF services "Get name list" and "Get capability list" are only provided by the CP 143 in the server role, since the client role requires mass memory. The information requested with these services is sent to the client automatically by the CP 143 without support of the S5 program. The reply information sent by the CP 143 might include, for example, a list of all communications objects (PI, domain, variable) defined on the server and managed by it.

2.3.2 Domain Services

Meaning

A CIM network requires not only the transmission of pure data or variable objects. The domain services make the VMD (the PLC) a flexible device which can be adapted to the task in hand. This means that programs and data areas can be exchanged online. Objects (domains) of the device can be loaded on the device via the communications networks and, if necessary, saved. The services are kept so flexible that the data must not necessarily be on the coordinating device (the client) but that a fileserver can also be included as a "third party" communications partner.

The TF services for handling domains and PIs on the CP 143 allow the incorporation of SIMATIC S5 programmable controllers and programming devices in a CIM network via the SINEC H1 bus system.

Overview

The following TF domain services are supported by the CP 143 TF:

- Download (server role)
- Upload (server role)
- Load_domain_content (client and server role)
- Store_domain_content (client and server role)
- Delete_domain_content (client and server role)
- Get_domain_attributes (client and server role)

Initiative

The PLC can initiate a load or store service on its own initiative or at the instigation of a "third party".

All the TF domain services can also be used on the local station. In this case the SIMATIC S5 PLC acting as the client triggers the loading of domains from a TF fileserver to itself using a special job number (ANR 205). These local services can, however, only be executed by the master CPU.

Variables in the 'domain' scope

Variable objects with a domain-specific scope are also generated or deleted when a domain is loaded or deleted.

Notes on applications

Several configurations are possible for the SIMATIC S5 system:

1.) Downloading or uploading from the PG.

In the simplest case, the domain services are triggered by a SIMATIC programmer. The S5-DOS program file (name: xxxxxxST.S5D) contains the blocks to be loaded and to be handled as a domain. Using the supplementary PG package, PGLOAD, this S5-DOS program file is transferred to the programmable controller as a load file using the TF domain services. Loaded domains can also be saved for archiving purposes, i.e. transferring them back from the programmable controller to the programmer. These services can also be used to delete domains.

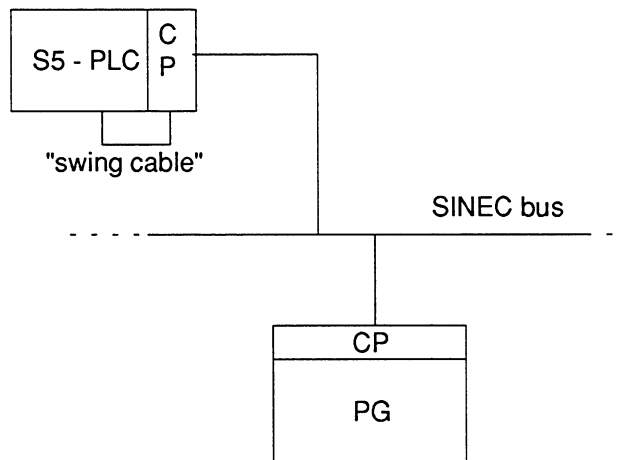


Fig. 2.7 Domain Services with the PG - 'Download/Upload'

2.) Third party association with PG and fileserver

A second configuration can be achieved if a station with a large memory is used in the network as an archive computer. This archive computer, known as the "TF fileserver" stores the files that will be loaded later as domains.

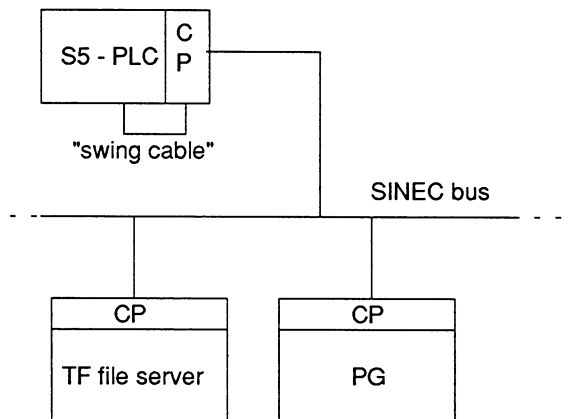


Fig. 2.8 Third Party Association with PG and File Server

The SIMATIC programmer can now request the SIMATIC CP to load a particular file from the TF fileserver into the programmable controller. During this procedure, information about the domain is stored on the communications processor. The programmer can also request the SIMATIC PLC to save a loaded domain on a fileserver or to delete the domain.

File transfer (S5-DOS program files) from SIMATIC programmers to the TF fileserver can be requested at the PG using the TF services. The files can, however, also be transferred from other data media (e.g. with a diskette).

This configuration is known as a "third party association" since the PG requests the manufacturing device to load data from a third station.

3.) Third party association with host computer and fileserver

The third configuration replaces the SIMATIC programming device with a host computer. If this host computer can handle the TF protocol, it can trigger loading and archiving procedures with the TF domain services between a SIMATIC PLC and TF fileserver. A SIMATIC S5 programmable controller can also be used as the host computer.

In this case, the program files are transferred to the TF fileserver as in the second configuration.

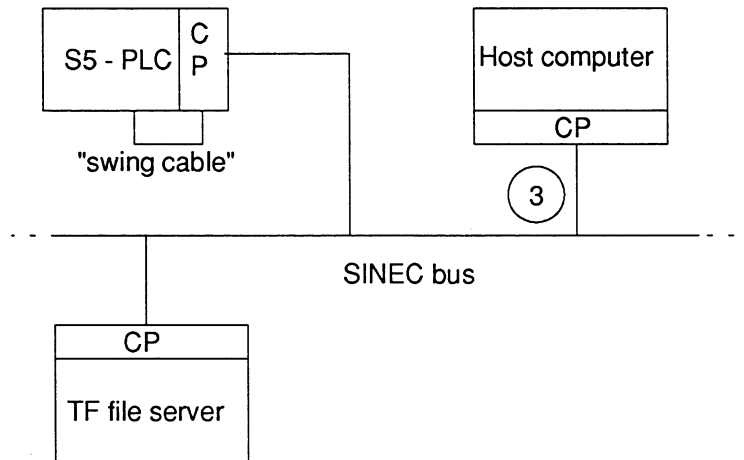


Fig. 2.9 Third Party Association with any Host and File Server

Description of the domain services

Download

A host computer (not an S5 PLC) uses this job to download a domain file to an S5 PLC.

2

Upload

A host computer (not an S5 PLC) uses this job to upload a domain file from an S5 PLC, for example for archiving purposes.

Load domain content

With the TF "load domain content" service, a client requests a server to load a domain. The domain is located in a file stored on the TF fileserver.

Store domain content

With the TF "store domain content" service, a client requests a server to save a loaded domain in a file on the TF fileserver.

Delete domain content

With the TF "delete domain content" service, a loaded domain is deleted in a server. All the variables belonging to the domain are also deleted.

Get domain attributes

A client can request the characteristics of a domain, i.e. its attributes, with the TF service "get domain attributes". The server sends the information back to the client in the acknowledgement (e.g. whether the domain can be deleted or whether it is used by a program invocation).

2.3.3 Program Invocation (PI) Services

Meaning

The PI services provide a view of the programs of the VMD to be executed. They allow the task-oriented grouping and use of domains.

The effects of the services in the real device depend on the device-specific or operating system-specific response in terms of the management of programs.

PI objects are characterized by their statuses and by the status changes brought about by the PI services and program execution.

When controlling programs using PI services, two aspects must be taken into account:

- System PI:
The services control the status of the programmable controller (PLC STOP-RUN)
- User PI:
The services control and monitor the application loaded on the programmable controller.

Overview

The following TF-PI services are supported by the CP 143:

- Create program invocation
- Delete program invocation
- Start, stop, reset, resume, kill program invocation (PI) and local program stop
- Get program invocation attributes

The role of the SIMATIC S5 PLC

PI services can be executed by a local programmable logic controller both on itself and on a remote partner. The programmable logic controller transfers the parameters required for the job to the communications processor when the service is initiated. If the PI services are triggered by the programmable controller on itself, then a special job number must be used (ANR 205). These local services can, however, only be executed by the master CPU.

The task of the CP 143 TF

The modelling of the PI services on the SIMATIC S5 PLC is implemented by the communications processor.

Descriptions of the PI services**Create PI**

With the "create PI" TF service, a client requests a server to generate a program invocation.

Delete PI

With the "delete PI" TF service, a client requests a server to delete a program invocation.

Start, stop, reset, resume, abort PI and local program stop

A client uses the "start, stop, reset, resume, abort PI and local program stop" TF services to control the status of the program invocation in a server.

Get PI attributes

A client requests the characteristics of a program invocation, i.e. its attributes using the "get PI attributes" TF service. The server sends this information back to the client in the acknowledgement (e.g. whether the program invocation can be deleted or the current status of the program invocation).

2.3.4 Variable Services

Meaning

With the variable services and with the standardized transfer of variable objects (simply known as variables), SINEC TF provides a neutral view of end system-dependent variables. By using the TF variable services, variable data can be exchanged independent of the end system.

The standardized representation of the data types must not be confused with standardization of the data contents, i.e. the semantics. The standardization simply involves a conversion of variables to the format of the end system, both on the client and on the server.

Overview

The following variable services are available:

- > Read variable
- > Write variable
- > Information report
- > Get variable attributes

Client and server view in the variable services

The variable services have the following characteristics:

- > Client:
The SIMATIC S5 programmable logic controller functions as the client wanting to access variables defined on another station.

For the SIMATIC S5 to access a variable (SIMATIC S5 as client), the name and the description of the object are required. These are transferred to the communications processor as parameters when the service is activated. Complex descriptions may be configured as remote definitions.

- > Server:
The SIMATIC S5 programmable logic controller functions as the server, and manages the variable defined locally on it.

The variables are managed by the communications processor. The variable attributes (name, scope, description of the variable etc.) are configured for the CP 143 TF and stored on the CP.

The assignment of a variable to the real object in the server (e.g. data block) is made on the SIMATIC S5 by configuration with the COM 143 package or with the additional PG package, PGLOAD. The conversion when the variable is accessed is the responsibility of the communications processor in the role of server.

Variable service descriptions

Read Variable

With the "read variable" TF service, a client requests the value of a variable from the server. The server sends this value in the acknowledgement.

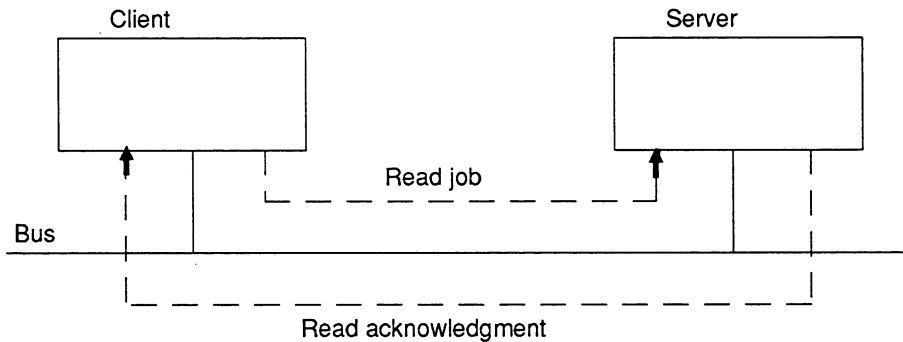


Fig. 2.10 Read Variable

Write Variable

With the "write variable" TF service, the client transfers data to a server. The server overwrites the variable specified in the job with the value transferred by the client. The service is acknowledged by the server. The acknowledgement tells the client whether the service was successful or not.

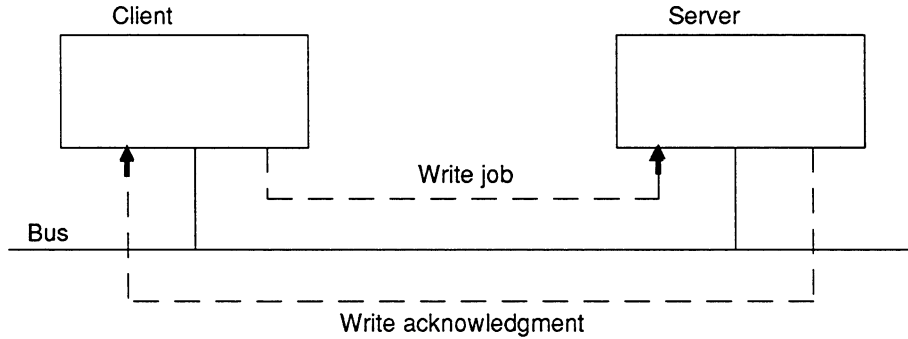


Fig. 2.11 Write Variable

Information Report

With the "information report" TF service, the server sends descriptions of variables and values of variables to the client without an explicit request. This job is not logically acknowledged (by layer 7). When using this job, remember that when the service is called in the PLC program, the communications processor accesses a local object that must be configured with the COM 143 package

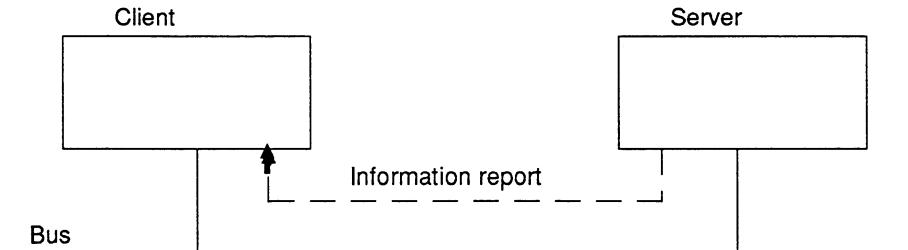


Fig. 2.12 Information Report

With the "get variable attributes" service the client requests the server to send information about the attributes of a particular variable (e.g. scope, description of the variable etc.) in the acknowledgement.

This service is only available from the SIMATIC S5 side with the PLC operating as a server. The PLC program cannot request this service. The communications processor can, however, process a job and return the attributes of a variable defined on it (configured in COM 143) in the acknowledgement.

2.3.5 Application Association Management

Meaning

The TF services for managing application associations supply the infrastructure required for communication at the application layer according to the OSI reference model.

Overview

The CP 143 TF provides the following services when functioning as either client or server:

- > initiate application association
- > abort application association

The following service is provided in the server role:

- > conclude application association

Initiative

The TF services for managing application associations are executed by the CP in most cases without being triggered by the user program. The parameters required to execute the services are entered when the application associations are defined using COM 143 TF and stored on the communications processor. Depending on the type of link (static or dynamic), the application relations are initiated when the modules are started or after a job for the application relation is triggered.

An application association can only be initiated after the corresponding "pneumatic post (transport) link" has been set up. When initiating the application association, a series of parameters are transferred which are significant for a heterogeneous communications network. These indicate whether non-open services (serial transfer) or variables with a certain nesting level are supported.

The role of the user program

An explicit client interface for initiating and concluding application associations by the PLC user program is not provided. An existing application association can simply be aborted by the PLC user program by calling the CP handling block "RESET".

2.3.6 Clock Services

Meaning

Apart from the already mentioned TF services, the CP 143 TF also provides clock services.

The integrated clock chip can access a system clock synchronized via the network. The programmable controller accesses the system clock via a special job number (218) using the CP handling blocks SEND and RECEIVE to write or read the time.

Synchronization

The clock function can be synchronized by the CP 143 TF (clock master) or by another station, for example the SINEC clock transmitter.

Reliability

For the CP clock master function the CP 143 has a redundant design to maintain synchronization if the clock master fails.

2.3.7 Serial Transfer

Meaning

For simple data transfer, the CP 143 provides the TF services of the function class "serial transfer". This function class is distinguished by the following characteristics:

Data is exchanged between the client and server without address information or parameters relating to the meaning of the data.

The serial transfer services are known as "non-open services". They are not included in the scope of functions of the international standard (MMS standard) and can therefore not be modelled on MMS services.

Advantages and Restrictions

The serial transfer services allow the greatest freedom in configuring application associations. The users must, however, negotiate the meaning and further processing of the data.

Compared with direct use of layer 4, when using the serial transfer services, the user can make use of the TF infrastructure. This ensures for example increased reliability with the logical acknowledgment of messages, the timed and logical monitoring of TF jobs and the possibility for better utilization of a transport connection by multiplexing at the application layer.

Overview

The following services are available for data transfer:

- read byte string
- write byte string
- transparent data exchange with and without acknowledgment

Service descriptions of serial transfer

Read/write byte string

The byte string services are used for "unidirectional" data transfer, i.e. data are transmitted in only one direction: with the "read byte string" service in the acknowledgement frame, and in the "write byte string" service with the request frame.

When the "Read byte string" service is called, the client requests data from the server. The request frame itself does not contain any data. The client receives the data from the server in the acknowledgment.

With the "Write byte string" service, the client transfers data to a server. The client can decide whether the received data should be acknowledged or not.

Transparent data exchange

With the "transparent data exchange" service, data exchange can be bi-directional. Data can be transmitted both in the request frame and in the acknowledgement. The client can decide whether or not an acknowledgement is required.

STF service	Job PDU	Acknowledgment PDU
Read byte string	Data request frame without data	Acknowledgment frame with requested data
Write byte string	Data frame	Acknowledgment frame without data or no acknowledgment
Transparent data exchange	Data frame	Acknowledgment frame with or without data or no acknowledgment

Fig. 2.13 Overview of the Serial Transfer Services

□

3 The TF Interface on the CP 143 TF

This chapter explains the use of the TF services integrated on the CP 143 TF.

The chapter describes aspects important for the use of all TF services. Information for specific services can be found in the following chapter.

The chapter explains the following:

- how to use handling blocks for the TF services
- the differences between the client/server view on the TF interface
- the significance of configuring the CP 143 TF with the COM 143 TF configuration tool
- the aspects covered by programming on the PLC
- which non-service dependent parameters are transferred or requested on the TF interface

With this information, you will be able to use the next chapter as a reference section to locate the service descriptions you require for your task.

3.1 The Principle of the TF Link PLC - CP

Intention

Chapter 4, Volume 1 explains the basics of the PLC-CP link. This section expands the information in terms of the TF interface.

The CP is addressed using handling blocks (HDBs)

Control of the CP by the STEP 5 user program is via the dual-port RAM (DPR) using handling blocks (HDBs).

Overview

The following system calls (i.e. HDBs) are used for the TF services:

- > SEND-direct Transfers data to the CP
- > SEND-ALL Transfers data from the PLC to the CP
- > RECEIVE-ALL Data is accepted by the PLC from the CP
- > RESET Resets an application association
- > CONTROL Requests the status of a job
- > SYNCHRON Synchronizes start-up between the PLC and CP

Address information

All these handling blocks must be supplied with an interface number and a job number (exception: the SYNCHRON HDB only requires an interface number).

Essentially, the main link between the STEP 5 user program and a particular action on the CP is a combination of SSNR/ANR. To prevent the system responding incorrectly, make sure that an SSNR/ANR combination is only assigned once in an S5 PLC.

Exclusion

The STEP 5 user cannot access the CPs directly (by avoiding use of the handling blocks).

TF jobs

A TF function is called by an application program using the job buffers. These job buffers are referenced in the handling blocks and transferred via the dual-port RAM to the CP.

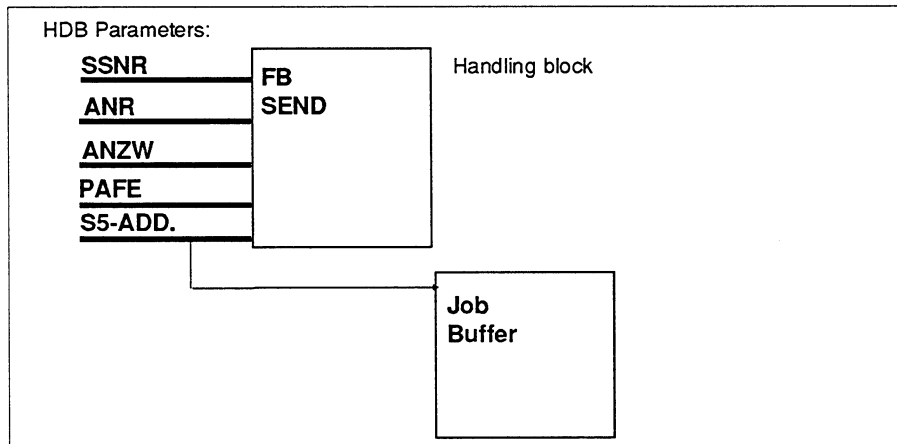


Fig. 3.1 Principle of Referencing the TF Job Buffer

Extended status word

The status word for the TF interface is extended by a third word for TF error recognition. The meaning of the first two status words remains unchanged (for selecting the status word, refer to the detailed description in the Section "Description of the HDB Call Parameters", Page 11).

Note: details of the handling blocks of the individual programmable logic controllers, particularly when the blocks are integrated in the operating system, can be found in the descriptions of the specific programmable logic controllers.

3.1.1 Using the Handling Blocks on the CP

This section describes the use of the handling blocks for transferring TF jobs to the CP. Further information about this topic can be found in Volume 1 of this manual; particularly the processing of the RESET, SYNCHRON and CONTROL HDBs.

Processing Send/Receive

Example of a TF job - read and write variable

The description of a write and read job illustrates how this functions.

Using the SEND-DIRECT HDB, the user transfers the job buffer to the CP. This informs the CP 143 TF that the user wants to send a data record (write job) or that it is ready to receive a message (read job) and also where the received data should be stored.

Once the job buffer arrives on the CP, the PDU for the particular job is created automatically by the CP, at the same it makes sure that no further job can be triggered with the SEND-DIRECT HDB for this ANR (ANZW=job active=00X2H).

Write Variable Job

With a write job, the data for transmission are requested from the PLC using the background communication (SEND-ALL), entered into the PDU and sent to the remote partner. When the data arrives, it is compared with the configured data description. Following this, the data is made available to the PLC by the background communication (RECEIVE-ALL) and transferred to the appropriate data block in the PLC. The job is then acknowledged. The initiator of the service evaluates the acknowledgement and terminates the job accordingly:

- Positive acknowledgment:
the job was completed successfully and "job complete without error" is entered in the SEND-DIRECT ANZW= 00X4H, allowing the job to be triggered again.
- Negative acknowledgment:
the job could not be executed. "Job complete with error" is entered in the SEND-DIRECT ANZW and "remote error" =09X8H, can be found in the

TF error status word (ERRCLS/ERRCOD). The job can then be triggered again.

Read Variable Job

With the read job, the PDU for the specific job is transmitted to the remote partner. The remote CP evaluates the PDU and requests the data from the PLC via the background communication (SEND-ALL). Once the PDU contains the required data, it is transmitted with a positive acknowledgement to the caller (service initiator). The data is then transferred to the PLC using the background communication (RECEIVE-ALL).

3

- Positive Acknowledgment:
After successful transfer of the data, "job complete without error" (=00X4H) is entered in the SEND-DIRECT ANZW and the job can be triggered again.
- Negative Acknowledgment:
If a negative acknowledgement is received from the remote partner or if the data cannot be transferred to the PLC, "job complete with error" is entered in the SEND-DIRECT ANZW and "remote error" (= 09X8H) is set. The detailed error ID can then be read from the TF error status word (ERRCLS/ERRCOD). Following this, the job can be triggered again.

3.2 General Client Interface for Calling TF Services

3.2.1 Job Buffer

Meaning

A TF function is called by an application program using the "job buffers". These job buffers are transferred via the dual-port RAM to the communications processor using standard handling blocks. The job buffer itself is used to transfer the parameters required to execute the service correctly on the communications processor.

Location and Formats

Job buffers must be located in data block or extended data block areas and are restricted to a maximum length of 256 bytes. Each job buffer consists of a general section and a service-specific section.

Support

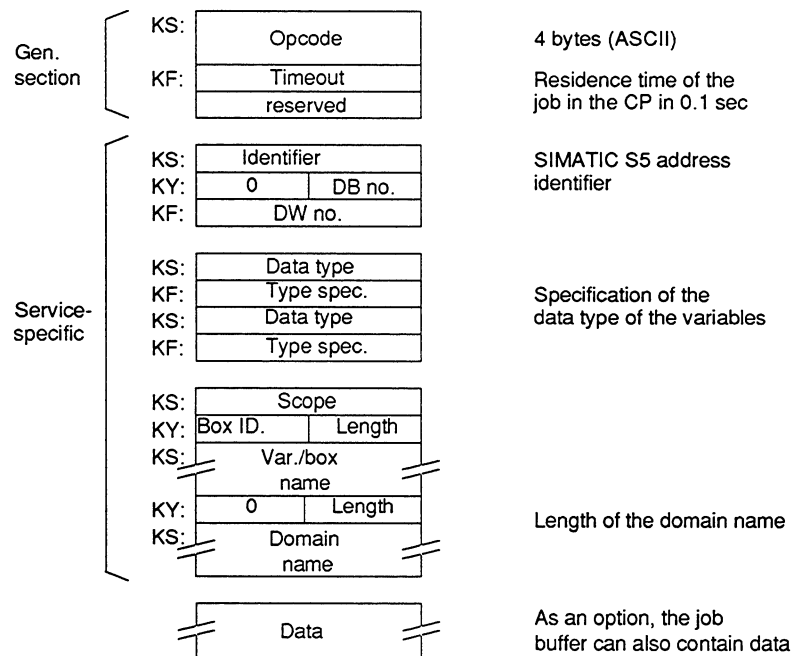
With the REQUEST-EDITOR, the S5 user has a tool to support the creation of job buffers. Using this tool ensures the syntactical correctness of the job buffers.

Structure

Figure 3.2 illustrates the basic structure of a job buffer using the example of the variable services.

The general section applies to all TF services. (open and non-open services).

The structure of the service-specific section of the job buffer differs depending on the TF service. A detailed description can be found in the section describing the individual services. The same applies to the services for simple data transfer supported by the communications processor (non-open AP services).



3

Fig. 3.2 Structure of the Job Buffer based on the Example of the Variable Services

Special Features

It is nevertheless possible to generate job buffers without using the TF editor.

It is also possible to define certain parameters in the memory card of the CP (local and remote definition, see Configuring, Test) instead of in the job buffer (PLC program). If you configure a variable completely, the job buffer only needs to contain the scope and the object name.

The parameters specified in the job buffers always have priority over the parameters programmed in COM 143 or in the COM 143 tool PGLOAD.

Description of the General Section

The following descriptions are based on Figure 3.2

Opcode: 2 words, format: KS
 In the opcode, the user encodes the required TF service.
 The coding is made using four ASCII characters and can
 be seen in the following table.

V-RE	Read variable
V-WR	Write variable
V-IN	Information report
D-LO	Load domain content
D-ST	Store domain content
D-DE	Delete domain content
D-GE	Get domain attributes
P-CR	Create PI
P-ST	Start PI
P-RE	Resume PI
P-SP	Stop PI
P-RS	Reset PI
P-AB	Kill PI
P-HL	Local program stop
P-GE	Get PI attributes
P-DE	Delete PI
M-ST	Status
M-SU	Unsolicited status
M-ID	Identify VMD
B-RQ	Read byte string
B-WQ	Write byte string with acknowledgment
B-WO	Write byte string acknowledgment
B-WI	Request byte string length
T-DQ	Transparent data exchange with acknowledgment
T-DO	Transparent data exchange without acknowledgments
A-CF	Configure ANZW [local]

The meaning of the individual services and how they are modelled on the SIMATIC PLC is described in the relevant section of this manual.

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum residence time of the job in the interface module). This is specified in multiples of 0.1 sec. If the job is completed within the specified time, the parameter is irrelevant.

If the residence time of the job elapses without an acknowledgement arriving, the following actions are initiated by the interface module:

1. The job status is set to "job complete with error" (can be obtained with a "Control" call in the PLC).

Error number in status word: D
TF error number: 2041 (see Appendix)

2. If the acknowledgment arrives later it is destroyed.

3. Link terminated and static link automatically re-established.

reserved: 1 word, not available to the user.

3.2.2 Sequence on the Client Interface

Parameters for the HDB

The configured job buffer for a TF job is transferred by calling a SEND-direct job in the S5 user program. The call parameters are as follows:

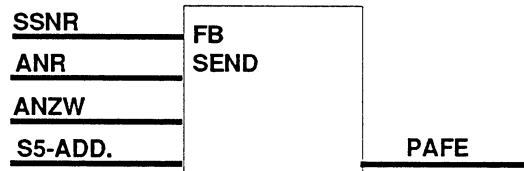


Fig. 3.3 Call Parameters "SEND Direct"

Support by the Request Editor

After the job buffers have been configured, the COM 143 tool TF-REQUEST-EDITOR supplies the parameters for calling the SEND-direct job.

Sequence and messages

After transferring the buffer, the status of the job (status byte in the dual-port RAM) is set to "job active" by the CP 143.

Depending on the service, the communications processor requires data from the PLC or must transfer data to the PLC. To allow for this data transfer, the jobs are processed by the background communication between the PLC and CP. This background communication is implemented by calling the SEND-ALL or RECEIVE-ALL handling block.

When the job is completed, the status is set by the communications processor to "job complete with/without errors".

Note:

The exact sequence of a service is described in detail in the individual chapters. Segmentation of the data which may be necessary with the "all" calls is not discussed in these chapters, but is supported by the communications processor whenever necessary.

Description of the HDB Call Parameters

The following parameter descriptions are based on Figure 3.3.

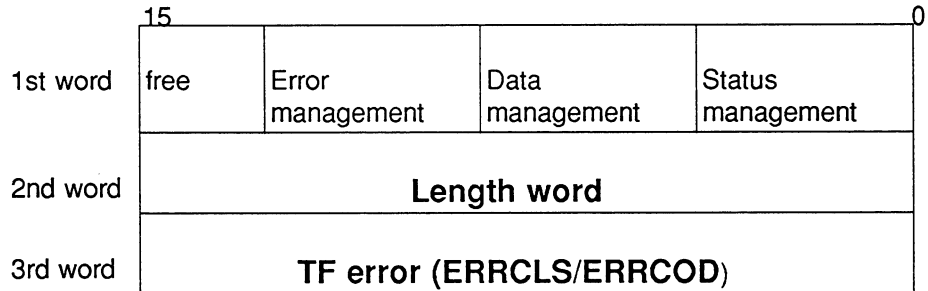
SSNR: 0 ... 255, the correct base interface number must be set on the hardware of the CP

ANR: 1 ... Max_ANR
 Max_ANR : 127
 Only odd job numbers are permitted for TF jobs (see "Configuring, Test").



The following job numbers have a special significance:
 205: configuration of the local VMD (local job)
 203/204: reserved
 218: clock functions

ANZW: Status word as specified in the HDB description, however, extended to three words::
 1st and 2nd word: same significance as in the description of the CP handling blocks
 3rd word: if the status "job completed with error" is entered in the 1st word and the error number "remote error occurred" is indicated, the 3rd word is valid and specifies the TF error in greater detail. Error numbers are normally taken from the reply (ERRCLS/ERRCOD). Additional unique codes were also defined for several errors.



ANZW: The TF error number is valid when
(continued) Status = Job completed with error

The following are distinguished in the error codes

- a) Error in the local protocol handler
- b) Error in the remote protocol handler
- c) Error for all function classes
- d) Error for special function classes (e.g. serial transfer)

In addition to these errors defined in TF, there are also the following:

- e) Errors on the client side (e.g. after analysis of the job buffer)
- f) Errors that are specified in more detail for the variable services in the "access result" parameter .
(errcls = 82H, errcod = 4xH)

The possible error codes and the causes of errors are described in the Appendix in Volume 2 of this manual.

Selecting the status word

The status word specified in the SEND-direct HDBs should match the status word selected during the configuration of an application association. If these status words do not match, the entry in the ANZW of the HDB is incomplete after the job has been processed. The information about data management, the transmitted length and the SINEC TF error word are missing (this information is then in the ANZW of the configured link).

Instead of configuring the status word, this can be stipulated using a configuration job (refer to the section on supplementary services and the section describing the request editor).

S5-ADD.: Source address of the job buffer:
Source identifier: DB/DX
DB/DX no.: 1...255
DW no.: 0 ... 2042
Length: max. 256 bytes (128 words)

PAFE Parameter assignment error byte
The possible error codes and the causes of errors are described in Appendix C in Volume 1 of this manual.

3

In the following descriptions, the term "data block" means a DB or if permitted DX.

3.3 General Server Interface

The Principle of Activating the Server Function

The TF server functions are handled on the CP 143 largely without support of the CPU on the programmable logic controller. TF jobs are interpreted by the CP 143 and executed with the PLC via the background communication, so that only a "SEND-ALL" and "RECEIVE-ALL" handling block call is required.

This applies to the following:

- > all TF variable services
- > domain services
- > program invocation services (PI services)
- > general services for virtual manufacturing devices.

Configuring Local Variables

Local variables (i.e. variables managed on the CP 143 in the server role and which can be read or written) must be configured with the configuration tool COM 143 TF:

- > for link-specific variables in the application associations screen
- > for VMD-specific variables using the VMD variable editor

or with the COM 143 tool PGLOAD

- > for domain-specific variables

Service-specific selections

The server interface is explained for the specific service. Special features when using domain and PI services are described in the section "TF Services for Implementing a CIM Network".

Special features with serial transfer

In contrast to the services mentioned above, with the non-open service "transparent data exchange", the service request must be transferred to the programmable logic controller on the server side, since the data can only be interpreted there. This "general interface" is explained in the section "Serial Data Transfer".

For this reason, a configuration job is necessary on the server side for the non-open service "Read/write byte string".

3

Notes

4 TF Variable Services

This chapter contains the information you require to use the variable services.

You can use this chapter in two ways:

- as a tutorial for configuring and programming variables and variable services (Section 4.1 Basics with checklist)
- as a source of reference when programming variable service jobs (Section 4.2 Service Description)

You can decide which services you require for your task based on the description in Chapter 2 "The TF Model and the TF Services."

4

Overview of the services:

The following services are described for the **initiator**:

- Read variable
and how to configure remote variable definitions
- Write variable
and how to configure remote variable definitions
- Information report
and how to configure local variables
- Read and write variable with free format addresses

For the **server**, the scope-specific configuration of variables (local variables) is described.

4.1 Basics of the Variable Services

4.1.1 Description and Management of Variables

Definition

Variables are unstructured or freely structured data objects of the application system which can be read or written with the variable services.

Describing variables

The structure of these data objects must be formulated in a type description. The type description is required both on a client and on the server in a network.

➤ PLC is server:

The type description is stored on the CP by configuring with the COM 143 TF tool. Configuring involves the local variables since the source of the variable to be read or the destination of the variable to be written is on the local device.

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

➤ PLC is client:

The type description is either transferred to the CP in the job buffer or is stored by configuring with the COM 143 TF tool. Configuring involves remote variables, since the source of the variables to be read or the destination of the variables to be written is on the partner.

A buffer must be available for the read or written data in the user program (DB, DX block).

Special feature of the information report service

When reporting a variable, the initiator of the transfer is the device on which the variable object was configured as a local object. The initiator of the job is therefore the device with the "server" role.

To be able to receive reported variable values, the variable must be configured as a remote object on the receiver.

Client: significance of the job buffer for the variable description

In the job buffers, only simple types (basic data types) and arrays of simple types are supported. These can be coded in four words. If more complex variables are involved, the user must configure this in COM 143 TF (TF definitions). If the variable is configured, the four words in the job buffer have the value 0.

4

The following more complex data types are supported but must be configured for the client access:

- Structures with components of the basic data type
- Structures with components containing structures with components of the basic type
- Structures containing components that are arrays of a basic type
- Arrays with elements that are structures whose components are of the basic data type
- Arrays whose elements are arrays containing elements of a basic data type

Note: detailed information about the type description can be found at the end of this chapter.

4.1.2 Scope of Variables in a SIMATIC S5 Programmable Logic Controller

Meaning

Each variable is assigned a scope. This allows remote access to the variable on the PLC (via the network) to be restricted. The following scopes are possible:

a)VMD-specific:

Since in SIMATIC S5 a programmable logic controller always represents a VMD, this scope means that the variable is valid and known in the whole station. Access to the variable is permitted via every application association regardless of whether a particular program or particular program section (domain) is loaded. A VMD-specific variable is visible from every station, i.e. every station can access this variable via any application association.

Application:

Global lists or variables accessed by different programs in the PLC.

Server configuration:

VMD-specific variables are configured with COM 143 TF using the function VMD variable editor (local variables).

Client configuration:

On the client, the variable must be configured if it is more complex (data type description > 4 words). The variable is, however, configured for the application association as a remote variable.

b)Domain-specific:

Domain-specific variables are also assigned to the whole VMD and valid within it. The existence of such variables, however, depends on whether a particular program or particular program section (domain) is loaded. Domain-specific variables are always assigned to a specific program. Access to domain-specific variables is possible via all application associations.

Application:

Domains can be understood as a "cocoon" around an application program for a specific automation task. Domain-specific variables are then local variables belonging to this application program and are protected by the domain "cocoon".

Server configuration:

The type description of the domain-specific variables is only required when the domain is loaded on the PLC. For this reason, the type configuration must be made with the COM 143 TF tool PGLOAD. PGLOAD generates and loads domains.

Client configuration:

On the client, the variable must be configured if it is more complex (data type description > 4 words). The variable is, however, configured for the application association as a remote variable.

4**c) Link-specific:**

Link-specific (or application association specific) variables are assigned to a particular application association. The variable is only visible via this link, i.e. it is only possible to access the variable via this link. Access does not depend on a particular program being loaded.

Application:

The application association is capable of allowing access to one of several tasks on a VMD. The use and access to data areas can therefore be restricted to a specific task by means of a link-specific assignment.

Server configuration:

Link-specific variables are configured using the sub-function local variables during link configuration with COM 143 TF.

Client configuration:

If the variable is more complex (data description > 4 words) a variable configuration is necessary on the client.

Fig. 4 .1 provides an example of a complete virtual device, two domains, a program invocation and assigned variables.

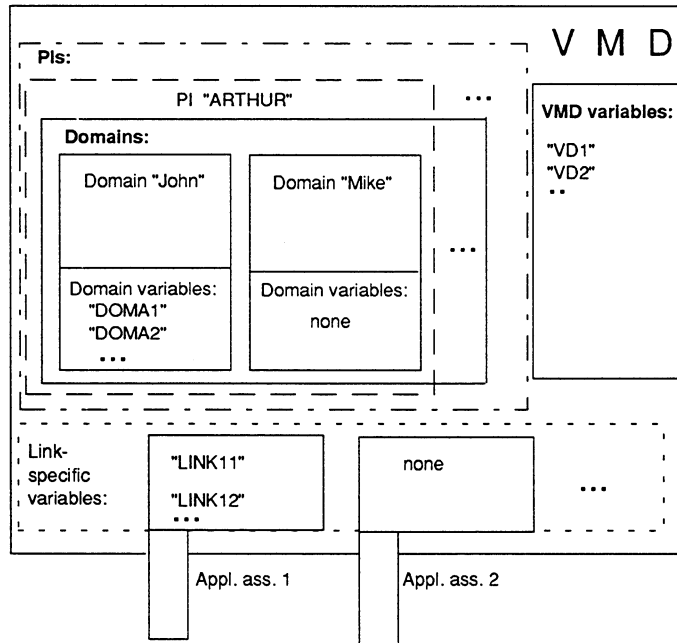


Fig. 4.1 Example of VMD Structure

There are two application associations to the virtual device: application association 1, application association 2.

The variables "LINK 11" and "LINK 12" have a link-specific scope and are assigned to application association 1. Access to these variables is only possible via application association 1.

The variables "VD1" and "VD2" have the VMD-specific scope. Access to these variables is possible via application association 1 and application association 2.

Within the virtual device, there are two domains, domain "John" and domain "Mike". The two variables "DOMA1" and "DOMA2" have the domain-specific scope and are assigned to the domain "John". Access to these variables is possible both via application association 1 and application association 2 by

specifying the domain name "John" and the variable name "DOMA1" or "DOMA2".

The domains "John" and "Mike" are used by the program invocation "ARTHUR". The program invocation "ARTHUR" can be addressed both via application association 1 and application association 2.

The services address the object of the server since the object really exists there.

Access to variable objects

The key to accessing an object and object description is the name for logical addressing (possibly also access via address without scope). These names are listed on the VMD in the object description. Access is controlled and restricted by the scope and the link.

Figure 4.2 illustrates the ways of accessing a variable using a name.

Scope	Identified by	Access/report via
Virtual device	Variable name	One or more application associations
Domain	Domain name and variable name	One or more application associations
Link	Variable name	Exactly one application association

Fig. 4.2 Access to Variables Dependent on Scope

For reporting variable values, domain-specific or VMD-specific variables can also have more than one application association available. Which of these application associations is used is specified by the SSNR/ANR of the SEND job.

4.1.3 Checklist for the Application

This checklist specifies the steps you must take when using the variable services.

Initiating jobs for variable services means the following:

- Configuring the job buffer (request editor or direct programming of the blocks in STEP 5 LAD/CSF/STL).
- Specifying the variable definition (data type description):
 - with simple variables (max. 4 data words) in the job buffer
 - with complex variables by configuring on the CP.
- Referencing the job buffer in the HDB program call.
- Integrating the HDBs and the status evaluation in the PLC program.

Using variable services with the PLC in the server role means the following:

- Specifying the variable definition (data type description) by configuring on the CP.
- Integrating the HDBs (SEND ALL and RECEIVE ALL) and the status evaluation in the PLC program.

4.2 Service Description

4.2.1 Read Variable (Client)

Meaning

With the read variable service, a client application can read a variable on a server. The data of the variable are transferred from the server to the client in the acknowledgment and entered under the local S5 address of the client (data block).

If the description of the data type requires more than 4 data words, the data description of the variable must be configured as a remote variable on the CP of the client.

Job buffer "read"

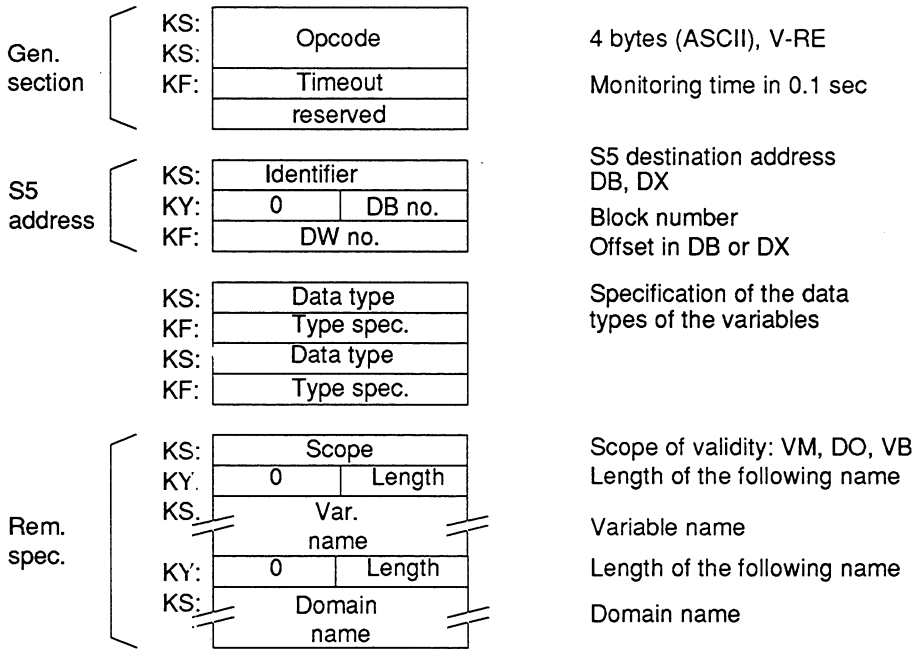


Fig. 4.3 Structure of the Job Buffer for TF Variable Service "Read"

Call description

General section:

Opcode V-RE

Timeout 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.

For more information about timeout, see page 3 - 9.

4

S5 address:

Description of the local destination address for the service.

Identifier 1 word, format: KS
Range of values:DB, DX
DB for data block
DX for extended data block

DB no. 1 word, format:KY
Range of values:high byte: 0, low byte: 0-255
Meaning:DB or DX number

If the identifier is DB or DX, the DB no. can also be 0. This means that there is no address specified in the job buffer. The addressed variable (see below) must then be completely programmed



As S5 address, you can only use data blocks not reserved by the CPU as system DBs (e.g. DB1). Otherwise the job is aborted with an error.

DW no. 1 word, format:KF
 Range of values:0 - 2042
 This is an offset within the data block or extended data block. The "length" parameter required for the complete definition of an S5 address is not specified. This is calculated in the PC from the type information for the variable.

Data Type Description:

This defines the data type of the addressed TF object. In the job buffers, only simple types (basic data types) and arrays of simple types are supported. These can be coded in four words. If the variable is more complex, the user must program this completely in COM 143 TF. If the variable is programmed, the four words in the job buffer have the value 0 or blank.

Data type 1 word, format:KS
 permitted values: see Table 4-2, page 4-40 .

Type
specification 1 word, format:KF
 permitted values: see Table 4-2, 4-40 .

Remote Object Description:

A network object is defined by the name and scope, i.e. a variable name, for example, must be specified here in the job buffer. In one job, the SIMATIC PLC acting as client can only ever access one variable.

Scope 1 word, format:KS
 Range of values:VM, DO, VB

Meaning:

VM:VMD-specific

The variable is known and valid throughout the whole destination station.

DO:Domain-specific

The variable is only valid in a particular domain in the destination station. In this case, the name of the domain must also be specified.

Read Variable, Client Interface (continued)

VB:Link-specific

The specified variable is only valid for the link identified by the interface number/job number.

- Length** 1 Byte, format:KY (low byte)
Range of values:1 to 32
Specifies the number of following valid bytes (length of the variable name).
- Variable name:** n bytes, format:KS
If the length of the variable name is odd, the last byte has no significance.
- Length** 1 word, format:KY
Range of values:
high byte:0
low byte:1 to 32
Specifies the number of following valid bytes (length of the domain name; only with scope=DO).
- Domain name:** m bytes, format KS
If the length of the domain name is odd, the last byte has no significance.

Sequence description (Read, positive acknowledgment)

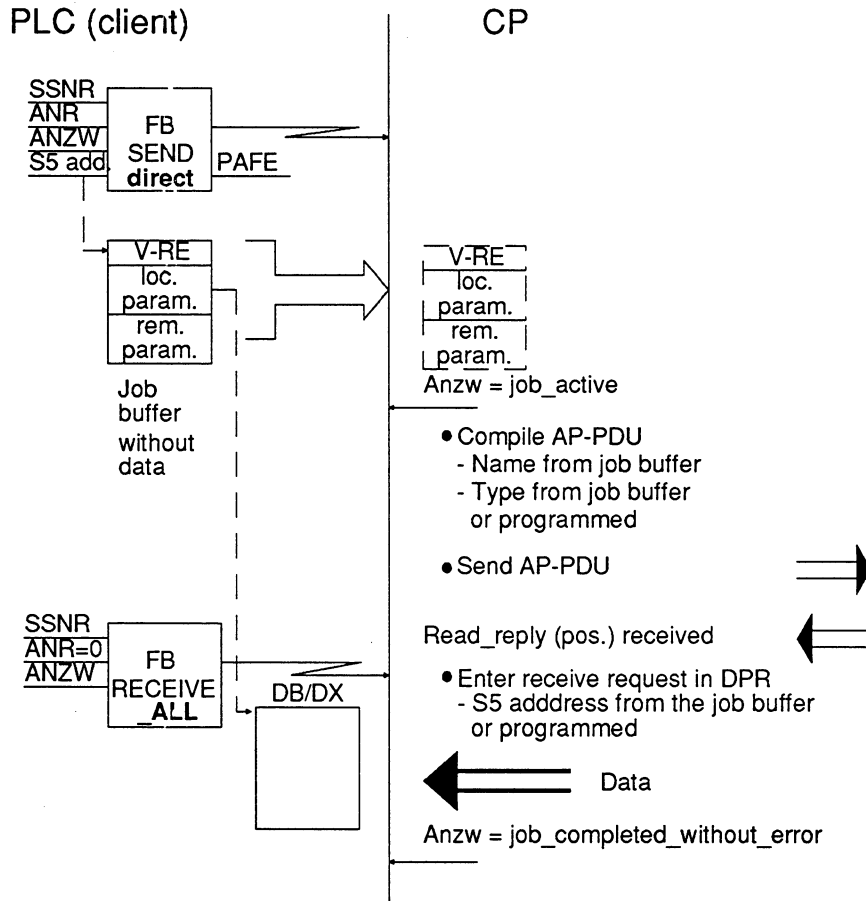
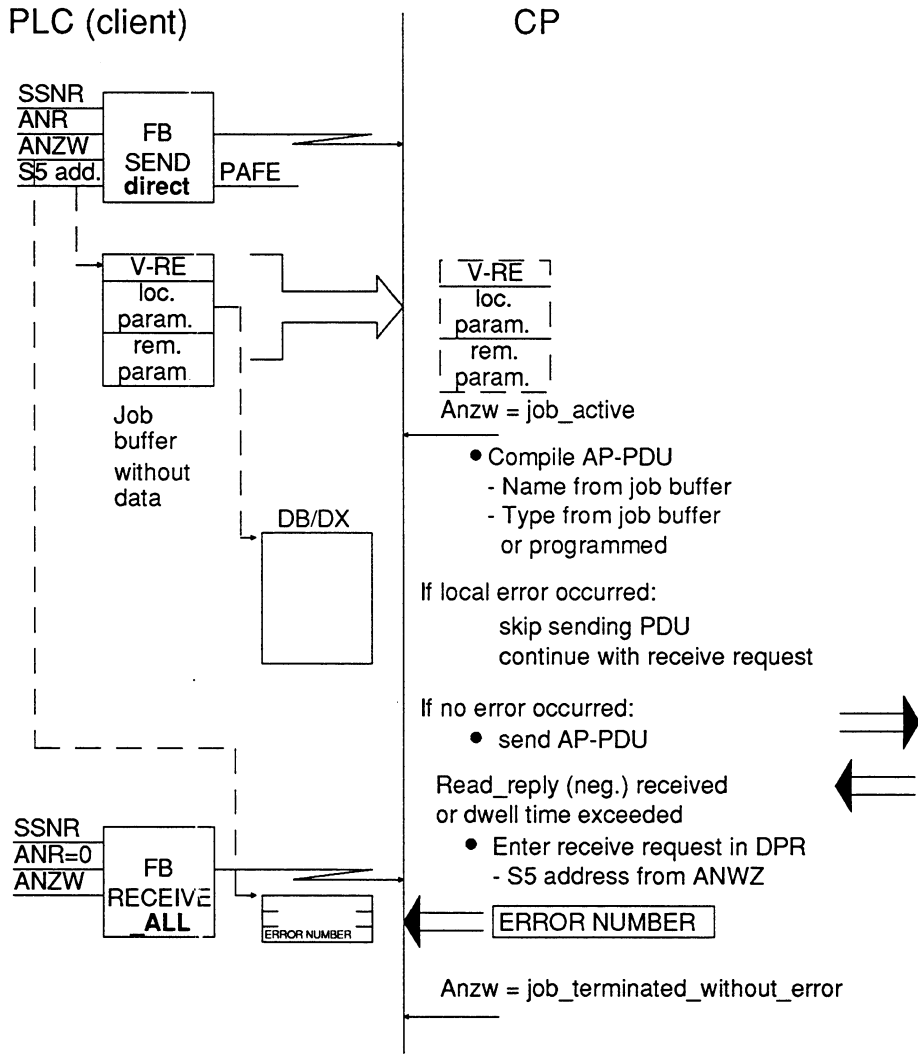


Fig. 4.4 Sequence Description (Read, Positive Acknowledgment)

Sequence description (Read, negative acknowledgment)



4

Fig. 4.5 Sequence Description (Read, Negative Acknowledgment)

4.2.2 Read Variable (Server)

The TF variable service read variable is interpreted and executed in the server communications processor largely without the support of the CPU of the programmable logic controller. In the PLC program, only the CP handling blocks "SEND ALL" and "RECEIVE ALL" must be called.

The variables to be read, must be configured as local variables specifying the scope.

If the requested variable type does not match the configured variable type, the CP 143 generates a negative acknowledgment.

4.2.3 Write Variable (Client)

Meaning

With the service write variable, a client application can write to a variable on a server. The data of the variable are transferred from the client to the server and entered in the local variable on the server by overwriting the existing content.

If the description of the data type requires more than 4 data words, the description of the data type of the variable must be configured on the CP as a remote variable.

"Write" Job Buffer

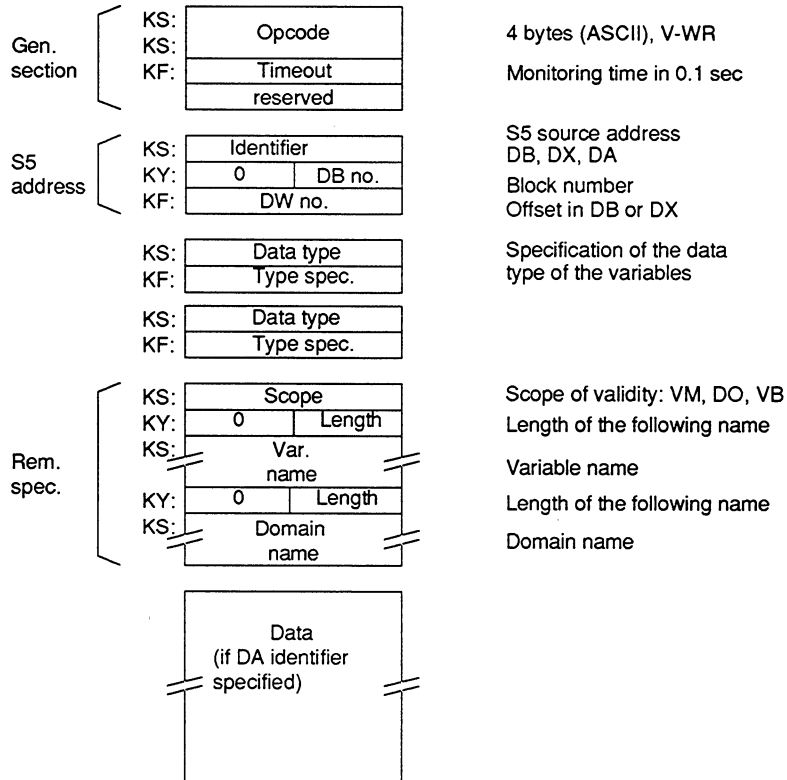


Fig. 4.6 Job Buffer for TF Variable Service "Write"

Call description

General section:

Opcode: V-WR

Timeout: 1 word, format:KF
Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.

For more information about timeout, see page 3 - 9.

S5 address:

Description of the local source address or indicates that the data are in the buffer.

Identifier: 1 word, format:KS
Range of values:DB, DX, DA
DB for data block
DX for extended data block
DA for data in the job buffer

Note on the "DA" identifier:

Apart from the TF service specification and the required parameters, the S5 user program can also transfer the data completely to the CP. This is possible when the specified S5 address is a data source. This allows a considerable increase in the data throughput, as can be seen in the description of the sequence. When using this facility, remember that a job buffer must not exceed 256 bytes. The data must follow on immediately after the last valid parameter of the job buffer.

If the DA identifier is specified the next two words are invalid.

DB no.: 1 word, format:KY
Range of values:high byte: 0, low byte: 0-255
Meaning:DB or DX number

If the identifier is DB or DX, the DB no. can also be 0. This means that there is no address specified in the job buffer. The addressed variable (see below) must then be completely configured.



As S5 address, you can only use data blocks not reserved by the CPU as system DBs (e.g. DB1). Otherwise the job is aborted with an error.

DW no.: 1 word, format:KF
 Range of values:0 - 2042
 This is an offset within the data block or extended data block. The "length" parameter required for the complete specification of an S5 address is not specified. This is calculated in the CP from the type information for the variable.

4

Data Type Description:

This defines the data type of the addressed TF object. In the job buffers, only simple types (basic data types) and arrays of simple types are supported. These can be coded in four words. If the variable is more complex, you must configure this as a remote variable. If the variable is configured, the four words in the job buffer have the value 0.

Data type: 1 word, format:KS
 permitted values:see "Table 4-2, page 4-40,

Type 1 word, format:KF
 specification: permitted values:see Table 4-2, page 4-40,

Remote Object Description:

A network object is defined by the name, i.e. a variable name must be specified in the job buffer. In one job, the SIMATIC PLC acting as client can only ever access one variable. According to the conventions of TF, you must also specify the scope.

Write Variable, Client Interface (continued)

Scope:	1 word, format:KS Range of values:VM, DO, VB
Meaning:	VM: VMD-specific The variable is known and valid throughout the whole destination station.
	DO: Domain-specific The variable is only valid in a particular domain in the destination station. In this case, the name of the domain must also be specified.
	VB:Link-specific The specified variable is only valid for the link identified by the interface number/job number.
Length	1 word, format:KY (low byte) Range of values:1 to 32 Specifies the number of following valid bytes (length of the variable name) Box names are restricted to a maximum of 8 characters.
Variable name:	n bytes, format:KS If the length of the variable name is odd, the last byte has no significance.
Length	1 word, format:KY Range of values: high byte: 0 low byte: 1 to 32 Specifies the number of following valid bytes (length of the domain name).
Domain name:	m bytes, format:KS If the length of the domain name is odd, the last byte has no significance.

Data with identifier DA:

Data If the source identifier is DA, the CP expects the current values of the data to be transmitted according to the data type description.

Sequence description (write, positive acknowledgment)

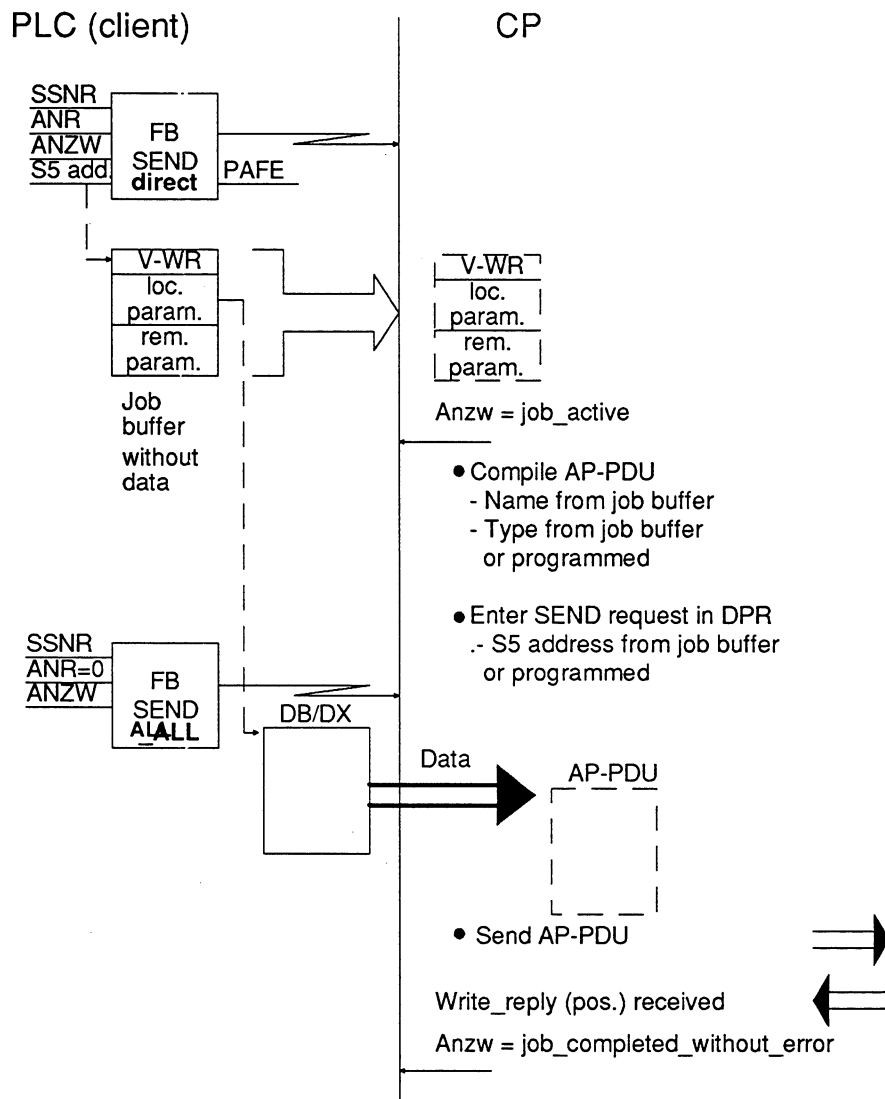
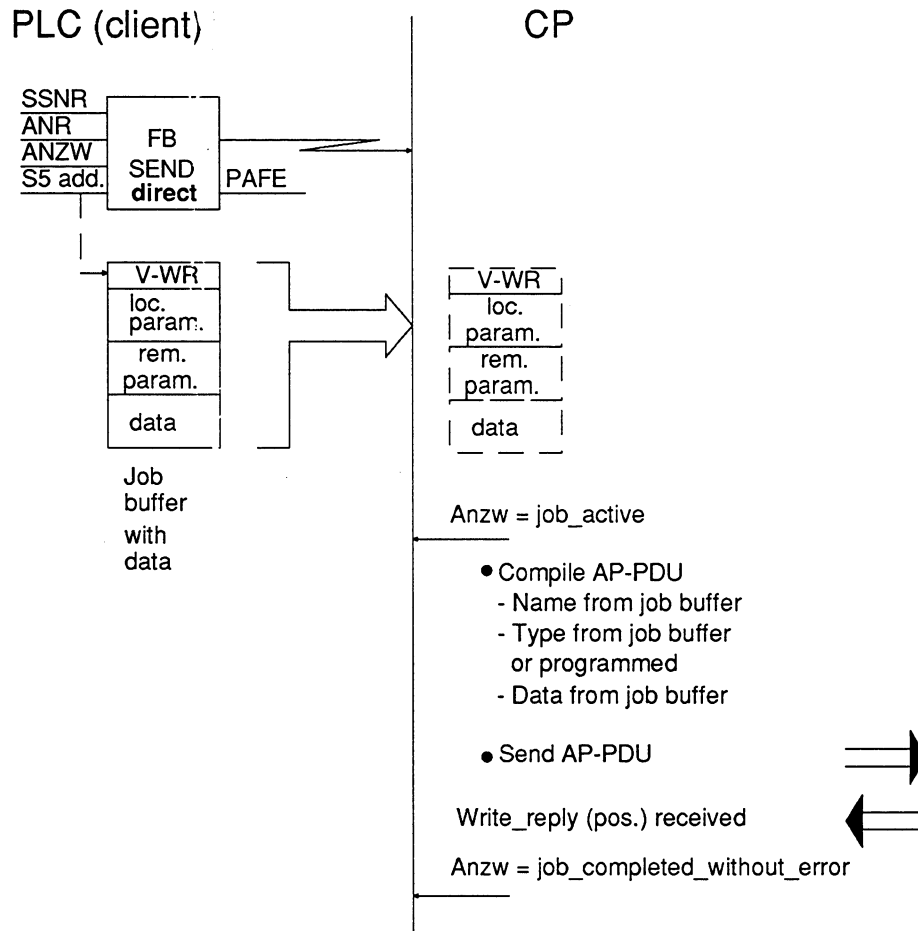


Fig. 4.7 Write with Source ID "DB" or "DX"



4

Fig. 4.8 Write with Source ID "DA"

Sequence description (write, negative acknowledgment)

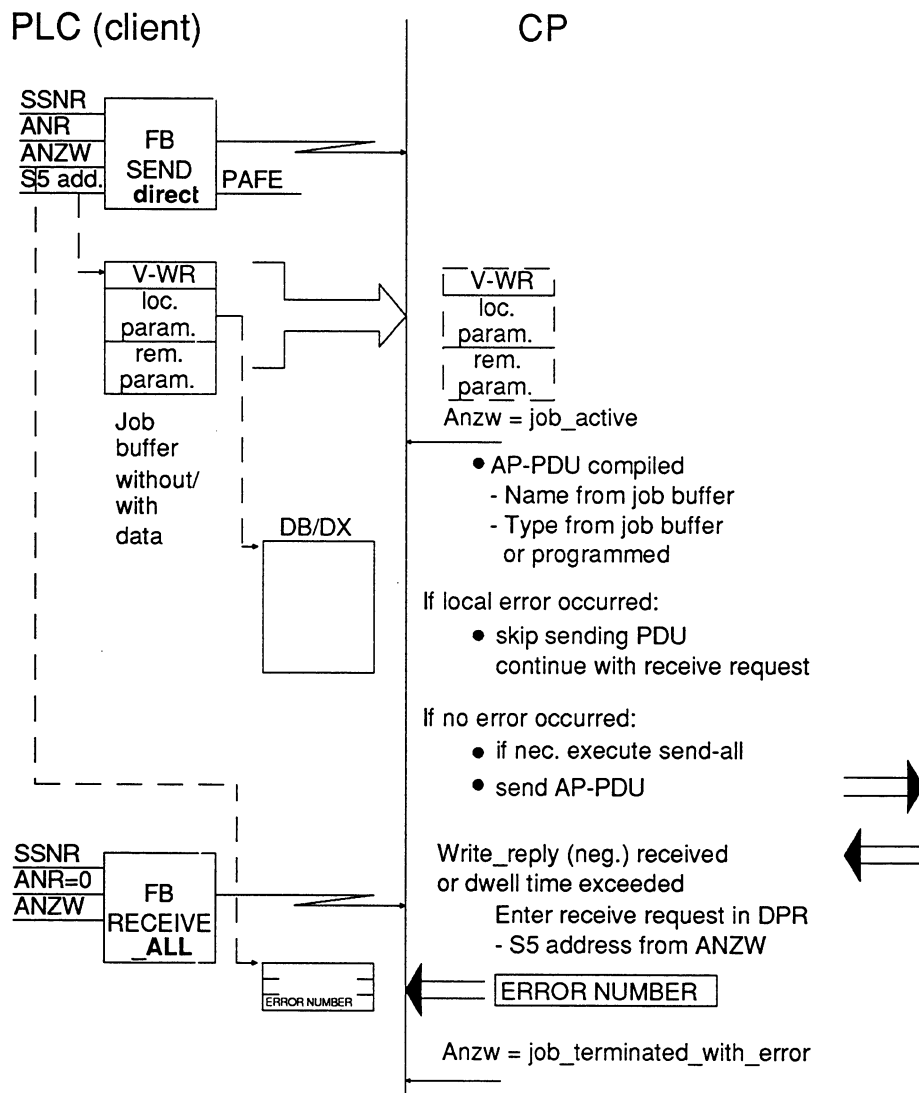


Fig. 4.9 Sequence Description (Write, Negative Acknowledgment)

4.2.4 Write variable (server)

The TF variable service write variable-is interpreted and executed in the server communications processor largely without the support of the CPU of the programmable logic controller. In the PLC program, only the CP handling blocks "SEND ALL" and "RECEIVE ALL" must be called.

The variables to be written, must be configured as local variables specifying the scope.

If the requested variable type does not match the configured variable type, the CP 143 generates a negative acknowledgment.

4.2.5 Information Report (Client)

Meaning

With the information report service, an application can send a variable unsolicited to a another application. The data of the variables is transferred in the job and entered in the buffer made available by the receiver.

With this service, the data description of the variable must be configured as a local variable (with scope) on the CP of the client. On the receiver, the variable must be configured as a remote variable.

It is also possible to group variables together during configuration and to specify such a group in the job. This makes it possible to transfer several variables in one job.

"Information report" job buffer

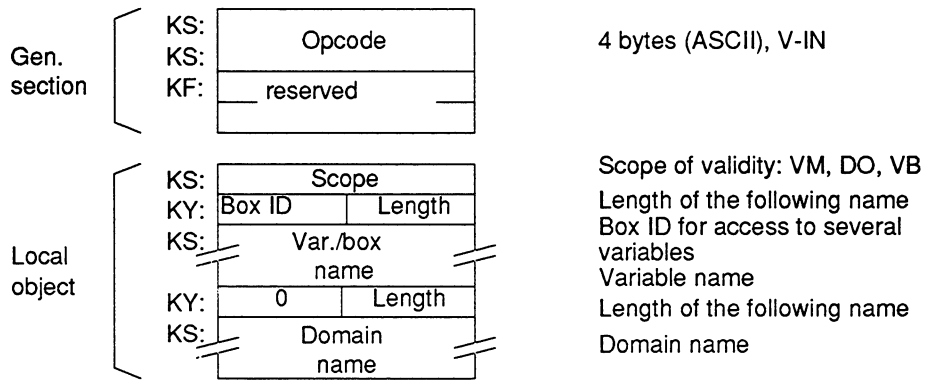


Fig. 4.10 Structure of the Job Buffer for TF Variable Service "Information Report"

Call description

General section:

Opcode V-IN

Local object:

For the "information report" service, with which a local object (or its value) is transferred to another station, the local object is described by the name.

Scope: 1 word, format:KS
Range of values:VM, DO, VB

Meaning:

VM:VMD-specific

The variable is known and valid in the whole VMD

DO:Domain-specific

The variable is only valid in a particular domain in of the local VMD In this case, the name of the domain must also be specified.

VB:Link-specific

The specified variable is only valid for the link identified by the interface number/job number.

Box ID: 1 byte, format:KY (high byte)
Range of values:0 to 1

Meaning:

The term box is used to mean a collection of variables forming a group of VMD-specific variables. If you select box ID = 0, only one single variable is involved, otherwise several variables are requested with one call. The following name is then not interpreted as a variable name but as a box name. The collection of variables under a "box name" is performed using COM 143 TF. In this case, the domain name in the job buffer does not exist.

Length 1 word, format:KY (low byte)
Range of values:1 to 32
Specifies the number of following valid bytes (length of the

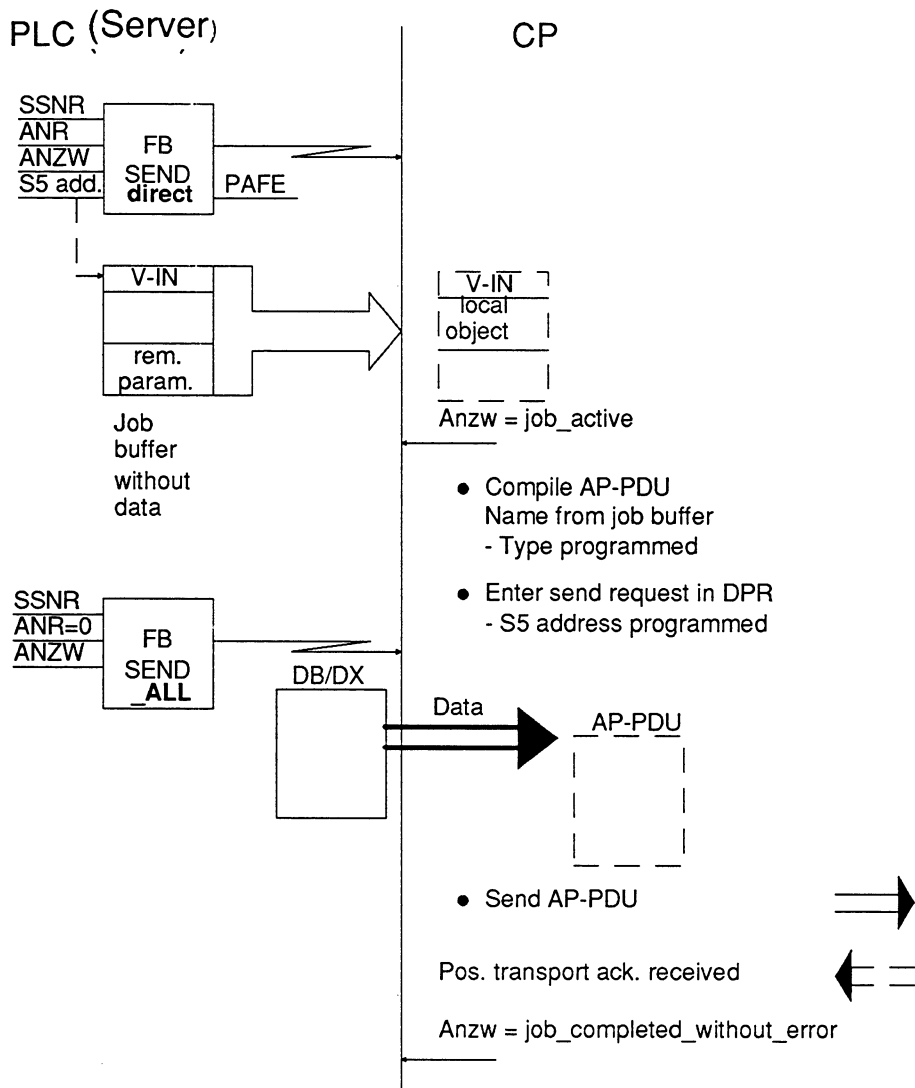
variable name or box name)
Box names are restricted to a maximum of 8 characters.

Variable name /Box name: n bytes, format:KS
If the length of the variable name is odd, a byte is appended at the end of the name.

Length: 1 word, format:KY
Range of values: high byte: 0
low byte: 1 to 32
Specifies the number of following valid bytes (length of the domain name)

Domain name: m bytes, format: KS
If the length of the domain name is odd, a byte is appended at the end of the name.

Sequence description (information report, positive acknowledgment)



4

Fig. 4.11 Information Report

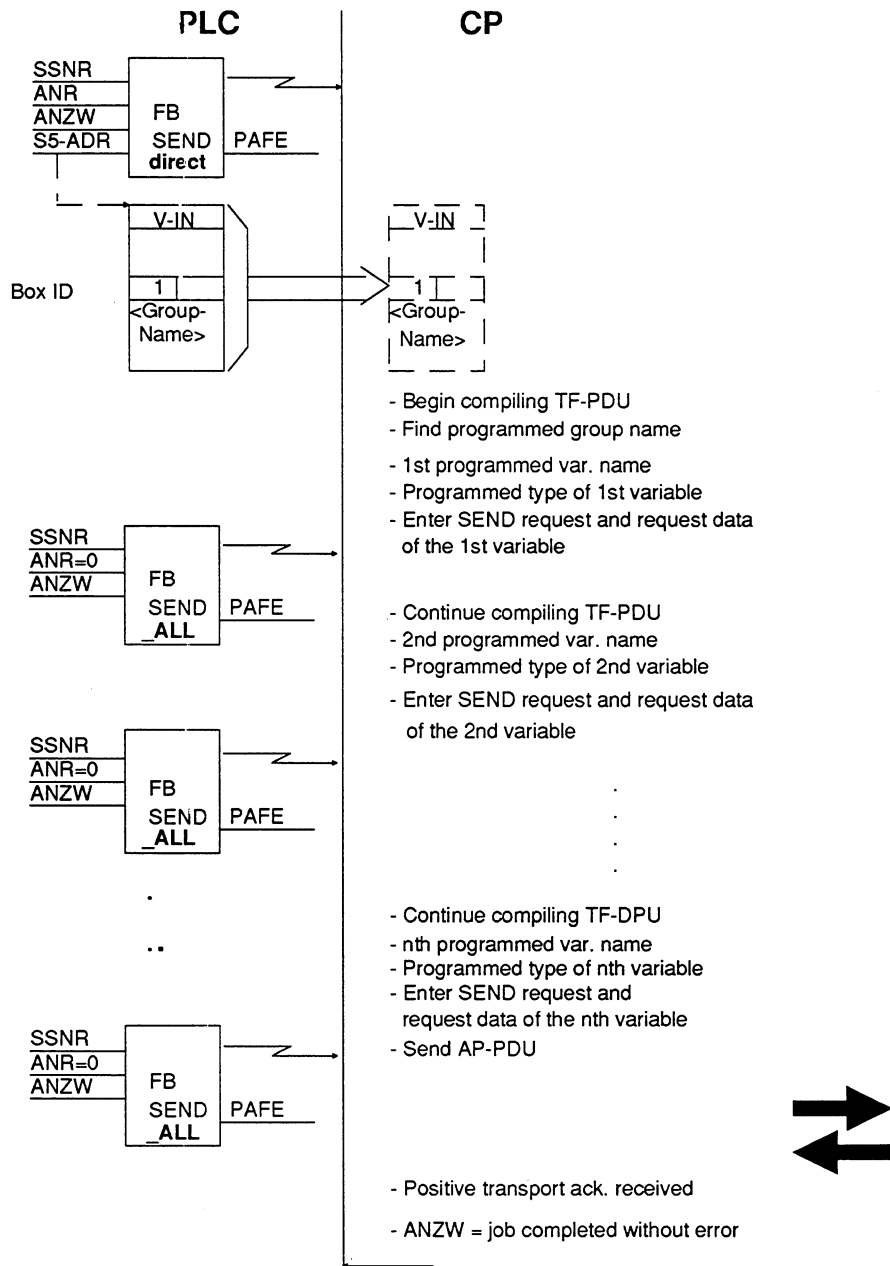


Fig. 4.12 Reporting Several Variables in One Call

Sequence description (information report, neg. acknowledgment)

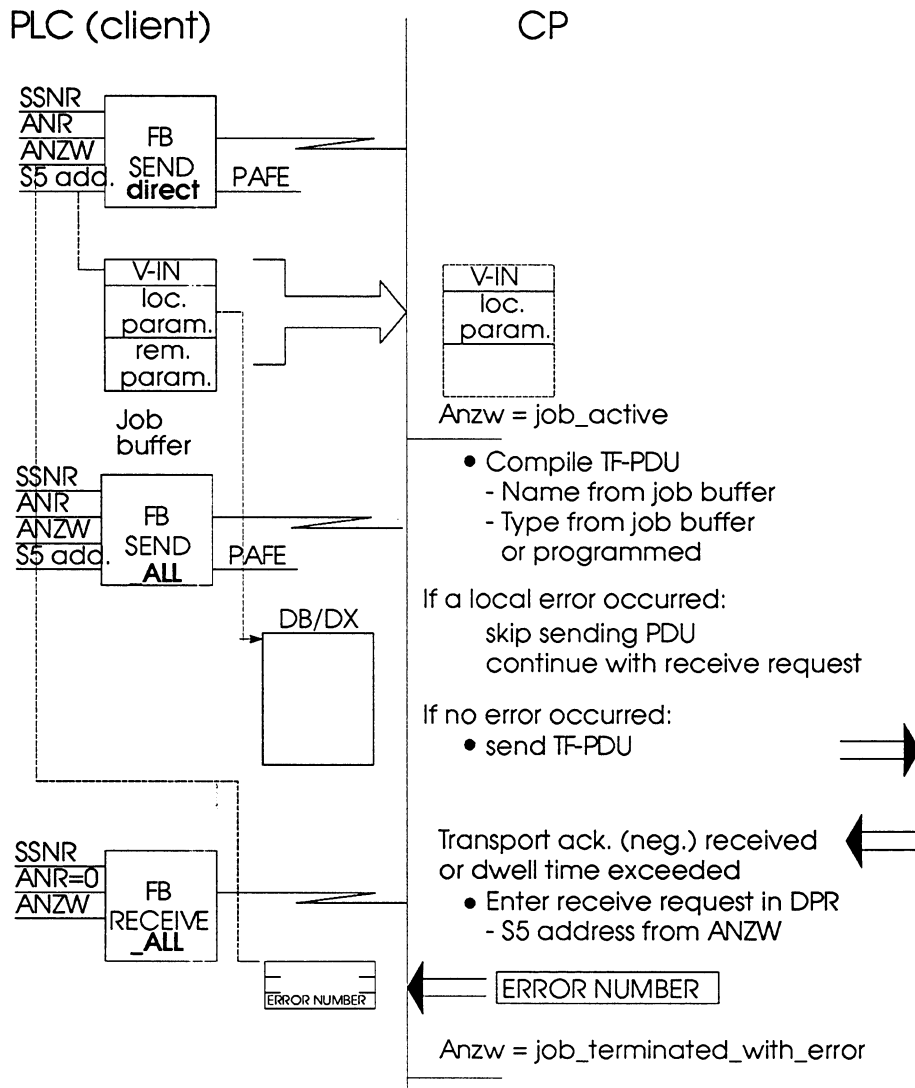


Fig. 4.13 Sequence Description (Report, neg. Acknowledgment)

4.2.6 Information Report (Receiver)

The TF variable service information report is interpreted and executed in the receiving communications processor largely without the support of the CPU of the programmable logic controller. In the PLC program, only the CP handling blocks "SEND ALL" and "RECEIVE ALL" must be called.

The variables to be reported must be configured as remote variables on the receiver specifying the scope.

4.3 Read and Write Variable with the Option of Addressing via a Free Format Address

The free format read/write service provides a more flexible form of access to variables.

This means the following:

- A direct and therefore more efficient access to the source or destination data area on the partner PLC is possible.
- Flexible adaptation to the address format of the partner device is possible.
- The variable does not need to be configured on the server.

The use of this service is, however, restricted compared with access using a name.

- Variable access cannot be differentiated and there is therefore no access protection using a scope (the scope is always VMD-specific).
- The transferred variables always have the type identifier octet string, i.e. differentiation according to a task-oriented variable structure is not possible.

4.3.1 Client Interface

The option of addressing via the free format address is permitted for variables of the octet string type. The service supports access to complete variables of up to 32 characters. Alternative access is not supported.

Write job buffer

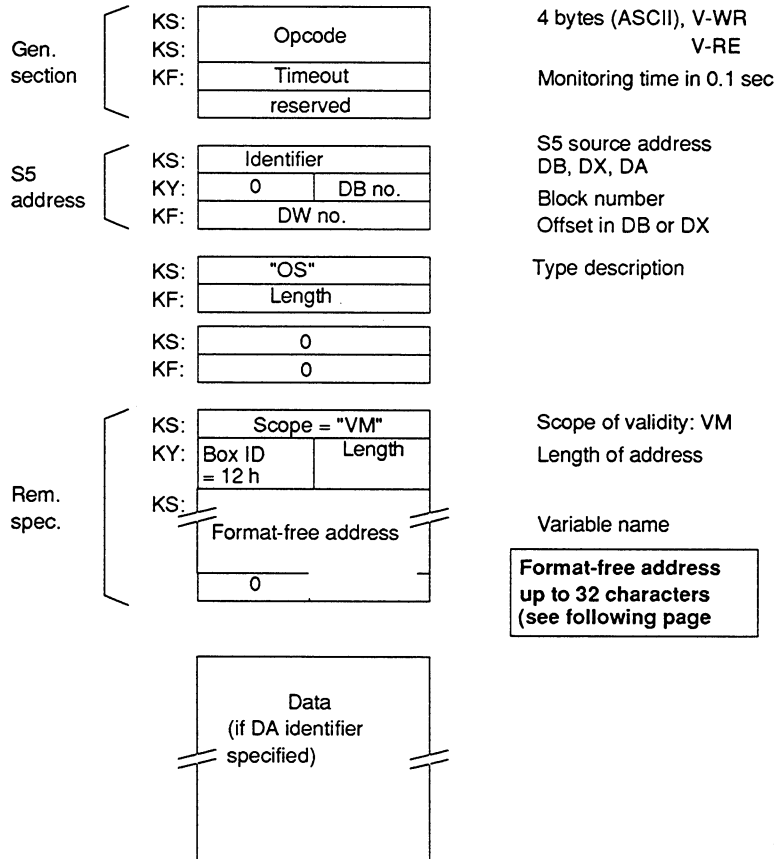


Fig. 4.14 Address Buffer for Variable Services Read and Write with Free Format Address

Call description**General section:**

Opcode: V-WR, V-RE

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.

For more information about timeout, see page 3 - 9.

S5 address:

Description of the local source address or indicates that the data is in the job buffer.

Identifier: 1 word, format:KS
 Range of values:DB, DX, DA
 DB for data block
 DX for extended data block
 DA for data in the job buffer (only with V-WR)

Note on identifier "DA":

Apart from the TF service specification and the required parameters, the S5 user program can also transfer the data completely to the CP. This is possible when the specified S5 address is a data source. This allows a considerable increase in the data throughput, as can be seen in the description of the sequence. When using this facility, remember that a job buffer must not exceed 256 bytes. The data must follow on immediately after the last valid parameter of the job buffer.

If the DA identifier is selected, the next two words are invalid.

DB no.: 1 word, format:KY
 Range of values:high byte: 0, low byte: 2-255
 Meaning: DB- or DX number

DW no.: 1 word, format:KF
 Range of values:0 - 2042

Offset within the data block or extended data block.

Data Type Description:

The data type is an octet string, the length is determined by the data block limit.

Data type: 1 word, format:KS
permitted value:"OS"

Type 1 word, format:KF
specification: String type:Number of bytes in a string

Remote Object Description:

The format-free address can be freely defined by the user. For communication between an S5 PLC and an S5 PLC, select a structure as described above.

The scope of validity is always VM.

Identifier 12_H indicates the job read/write variable with free format addressing.

Length 1 byte, format:KY (low byte)
Range of values:1 to 32
Specifies the number of following valid bytes of the free format address an.

with S5:
length = 8 for type 0 (see Figure 4.15)
length = 11 for type 1 (see Figure 4.16)

Free Format Address:

The structure of the free format address is represented for communication between SIEMENS programmable controllers. The format-free address with a status word is used by the S5 program for coordination when several PLC cycles are required for the data exchange between the CPU and CP.

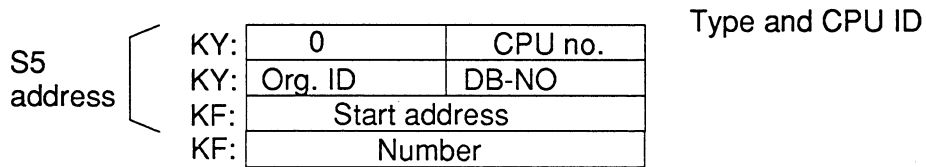


Fig. 4.15 S5-Specific Free Format Address in Job Buffer without Status Word

4

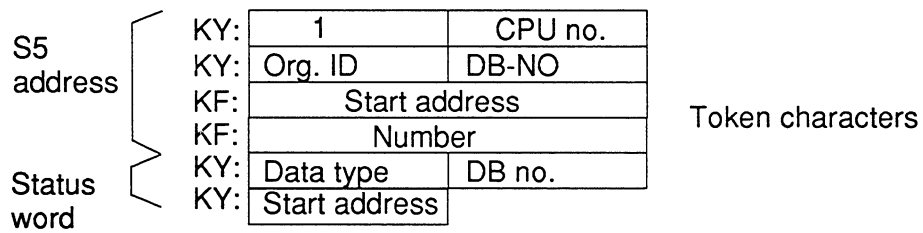


Fig. 4.16 S5-Specific Free Format Address in Job Buffer with Status Word



For communication between SIEMENS programmable controllers and a non-SIMATIC unit, the format-free address can have a different structure.

S5 address:

Description of the local source address or indication that the data are contained in the job buffer.

CPU No.: 1 byte, format:KY (low byte)
Range of values:1 to 4
Meaning: specifies the CPU no. in which the variable will be written.

Org identifier: 1 byte, format:KY (high byte)
Range of values:01,10
Meaning: address to which the variable will be written
10 for extended data block (DX)
01 for data block (DB)

DB No: 1 byte, format:KY (low byte)
Range of values:1 to 255
Meaning: if DB/DX, data block number to which the variable will be written; otherwise no significance.

Start address: 1 word, format:KF
Range of values:0 to 2047
Meaning: start address from which the variable will be stored in the data block.

Number: 1 word, format:KF
Range of values:1 to 2048
Meaning: number of data values to be transferred.

Status word:

Type: 1 byte, format:KY (high byte)
Range of values:01,02,03,255
Meaning: type of status word (see Table 1).

DB no.: 1 byte, format:KY (low byte)
Range of values:0 to 255
low byte:DB- or DX number

Start address: 1 byte, format:KY (high byte)
Range of values:0 to 255 (DW), 1 to 255 (flag area)
Meaning: area for storing the status word

Status word	Type	Values
Type	Flag area DW in DB area DW in DX area No status word	01H 02H 03H FFH
DB number	DB-Nr. DX-Nr. irrelevant with type = flag area	0 to 255 0 to 255
DW number	Flag word no. DW-Nr.	1 to 255 0 to 255

4

Tabelle 4.1:Status Word

Data with identifier DA:

If the source identifier is DA, the CP expects the current values of the data to be transmitted according to the data type description. The sequence descriptions correspond to those of the services "Read variable" and "Write variable" via object addressing.

4.3.2 Server Interface

At the server end, only the required SEND-ALL and RECEIVE-ALL calls must be incorporated.

Variables do not need to be configured.

4.4 TF Data Types in SIMATIC S5

In this section, you will find an explanation of the data descriptions used with SIMATIC S5. You will require this information to use the variable services and for configuring variables using the COM 143 TF configuring tool.

TF TYPE	TYPE description	Meaning	Corr. to S5 type
BO	no entry	Boolean	-
IN	8 16 32	Integer, 8 bits Integer, 16 bits Integer, 32 bits	- KF -
UN	8 16 32	Unsigned, 8 bits Unsigned, 16 bits Unsigned, 32 bits	- KH -
FP	32	Floating point number in MC5 format, 32 bits	KG
TI	4	Time of day, 4 bytes, format see below	-
TD	6	Time of day with date, format see below	-
BS	n	Bit string, n = number of bits in string	KM
OS	n	Octet string, n = number of bytes in string	KY
VS	n	Visible string, n= number of bytes in string	KS
AR	Array	Number of elements in an array	

Table 4.2: TF Type and Meaning

Explanation of the TF types and representation of the types in SIMATIC S5:

1. **BO:** Boolean

Boolean variables are modelled on the CP on a data word in the data block.

The following values are permitted:

0H "false"

≠0H "true"

2. **IN:** Integer 8/16/32

The TF data type integer 8 (representation 1 byte, range of values (-128 to +127)) and the TF data type integer 16 is modelled by the CP on a data word in a data block (format KF). The TF-data type integer 32 is modelled by the CP on two data words in a data block.

3. **UN:** Unsigned 8/16/32

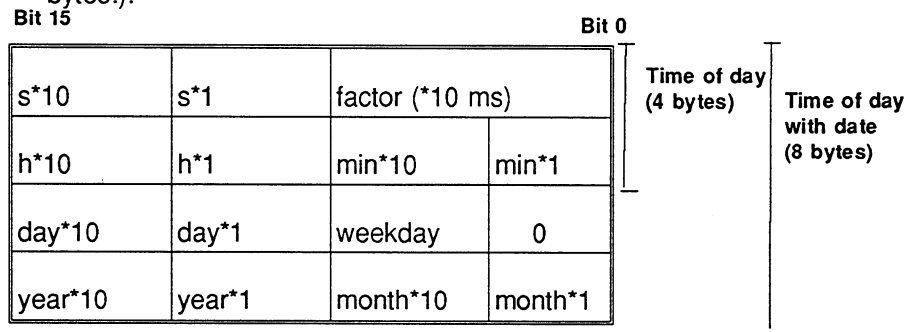
The TF data type Unsigned 8 (representation 1 byte, range of values 0 to +255) and the TF data type unsigned 16 are modelled by the CP on a data word in a data block (format KH). The TF data type unsigned 32 is modelled by the CP on two data words in a data block (format KH).

4. **FP:** Floating point number

Floating point numbers are modelled by the CP on KG in the SIMATIC PLC.

5. **TI, TD:** Time of day, time of day with date

Formats in the PLC for representing the TF formats time of day and time of day with date, are as shown below: (Note: S5=8 bytes; network=6 bytes!):



Permitted values:		Weekday 0	Monday
seconds units:	0 to 9		1Tuesday
seconds tens	0 to 5		2Wednesday
minutes units:	0 to 9		3Thursday
minutes tens	0 to 5		4Friday
hours units :	0 to 9		5Saturday
hours tens	0 to 1/0 to 2 (see below)		6Sunday
days units	0 to 9		
days tens	0 to 3		
months units:	0 to 9		
months tens	0 to 1		
years units	0 to 9		
years tens	0 to 9		
factor (*10 ms):	Factor (BCD-coded) * 10 ms		

Note on hours tens: Bit 15:

- 1: 24 hour format
- 0: 12 hour format

Bit 14:

- 0: AM
- 1: PM

Note:
when converting the TF time representation to the PLC time format used here, the CP always uses the 24 hour format



Every completed binary date is a valid date in the range 01.03.1984 to 29.02.2084.

Before a date is transmitted it is "normalized".

Example:

35.12.93	becomes	04.01.1994
35.14.93	becomes	07.03.1994
00.01.93	becomes	31.12.1992
00.00.93	becomes	30.11.1992
61.01.84	becomes	01.03.1984 but 60.01.84 becomes 29.02.1984

The individual numbers are BCD-coded, i.e. numbers between 0 and 9 are "normal". However non-normal values are also accepted. A BCD number always becomes $10 \cdot X + Y$.

Example:

F1_H becomes $15 \cdot 10 + 1 = 151$
 AB_H becomes $10 \cdot 10 + 11 = 111$
 FF_H becomes $15 \cdot 10 + 15 = 165$

Times are also normalized in a similar way to dates. This means that more than 59 minutes can be added to an hour.

A day consists of 86400 seconds. If a time and date are transmitted together and if the time is greater than or equal to the time 86400 seconds, this is converted to days and automatically added to the date.

4

Only normalized dates and times are received. The procedure allows the user to transfer a time (base time and [optional date] plus delta time in BCD arithmetic) extremely simply. The user is not concerned with the normalization, since the date is always received in a normalized form.

6. **BS:** Bit string

The "bit string" data type is modelled on the type "KM" known in the SIMATIC PLC. If the number of valid bits can be divided by 16 (number of bits in a data word, is modelled immediately on the resulting number of data words. Otherwise, the string is extended to the next word limit, and the bits appended must not be used in the PLC program.

BS types are stored as shown below:

Bit	15 ... 8							7 ... 0								
DW _n	8	7	6	5	4	3	2	1	16	15	14	13	12	11	10	9
DW _{n+1}	24	23	22	21	20	19	18	17	32	31	30	29	28	27	26	25

7. **OS:** Octet string

The octet string represents a string of bytes (with any content). It is modelled on the SIMATIC format (KY). An octet string with the length n occupies n/2 data words in the PLC, if n is an odd number, then (n-1)/2+1 data words are occupied and the last low byte is invalid.

8. **VS:** Visible string

The handling in the CP is the same as with the octet string data type, however, the range of values is restricted to representable ASCII characters (format KS).



**The VS is not checked by the CP 143 for permitted values
--> user check!!**



All string types are stored in the SIMATIC PLC in ascending memory addresses. This means with data blocks that first the high byte (this is located at the lower memory address) and then the low byte is written to.

The format and type conversions specified here are also carried out at the server interface, so that the user is unaware of differences in the data representation.

The definition of the data base types is made in the first and second data word of the type description. In this case, the third and fourth data words are irrelevant. Arrays of data base types are defined by the user entering the ASCII characters "AR" (for array) in the first data word of the type description in the job buffer and the number of elements in the array (repetition factor) in the second data word. The third and fourth data words contain the definition of the data type of the array elements.



5 TF Domain and PI Services Implementing a CIM Network

In this chapter you will find information required for using domain and PI services.

You can decide which services you require for your task based on the description in Chapter 2 "The TF Model and TF Services".

Domain services support you in task-oriented structuring of your automation task and supplying the PLC with currently required data and program code.

PI services support you in all phases of operation with the programmable logic controller.

The VMD object program invocation is closely linked with the VMD object domain. Both objects describe a specific view of the same physical object or parts of the same physical object.

In SIMATIC S5 this means the following:

- The PLC is represented by two program invocations, a system PI and a user PI. These PIs can be controlled using the PI services.
- The user PI includes the domains assigned to it when it is generated and within a PLC there can be up to eight loadable domains.
- The system PI covers the whole PLC. It is only used to start and stop a PLC using the PI functions.

5.1 Domain Services

Definition

A domain in the SIMATIC S5 programmable logic controller is always part of (or even the whole) PLC program. A domain always consists of one or more blocks.

Overview

The following domain services are available for the SIMATIC S5 PLC:

- > load domain content
- > store domain content
- > delete domain content
- > get domain attributes

Generating domains

To create a domain, two steps are necessary, as follows:

- > Creating the blocks (OB, PB, FB ...) and storing them in a block file using the STEP 5 basic package LAD, CSF, STL.
- > Grouping the blocks to form a domain using the COM 143 tool PGLOAD.

A domain is assigned to a CPU when the domain is loaded on the PLC. The additional information "dynamic" means that the domain is loaded by a host using the TF services. When domains are discussed in this chapter, they are always dynamic domains.

The function of the PGLOAD tool is described in Chapter 9 in this manual.

Defining domains

Which blocks are assigned to a domain is decided by the user. In practice, the blocks are grouped together to handle a definable automation task so that it is possible to adapt the program to the process by loading new domains as required.

Example

Domains could be distributed on a CPU for example so that one domain contains the actual PLC program (OBs, FB, PBs, SBs and DBs) while another domain contains the parameters for the PLC program (normally only DBs). If a modification is then made, normally only a parameter record must be exchanged and not the whole PLC program (e.g. changing product from red to green cars only requires a different parameter and not a different program). This division is, however, not mandatory and is not checked by the communications processor. It is also possible to split the PLC program so that not the whole program is contained in one (loadable) domain, but part is always loaded on the PLC (possibly even in an EPROM). Once again, the user can decide on the most suitable strategy.

Number of domains

The communications processor supports a maximum of eight dynamically loadable domains.

Local services

Every domain service can also be triggered locally (job number 205).



Before this function is executed locally using ANR 205, the communications processor must first be informed of the status word ANZW (of ANR 205), refer to "configuration job".

Static domain SIMATIC_S5

Under certain circumstances, you may not wish to use the domain services but nevertheless want to control the program using PI services. In this case, the static domain with the name SIMATIC_S5 is available.

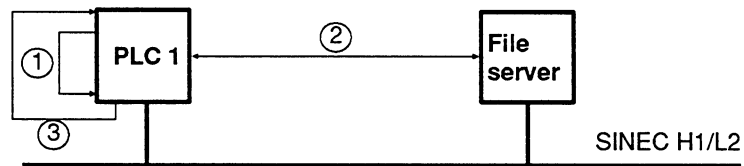
Checklist for using domain services

Using domain services means the following:

- Structuring domains, i.e. deciding which program or data sections (blocks) must be loaded or exchanged at what time.
- Defining variables belonging to a domain and configuring them as domain-specific variables using the PGLOAD tool.
- Grouping the required blocks to loadable domains using the PGLOAD tool.
- Clarifying the system configuration in which the loadable domains will be managed. Will there be a third-party association (host - PLC - fileserver)?
- Configuring application associations for transmitting the domain services. If third-party associations are to be used, fileserver application associations must also be configured.

Modes

Domain services can be activated directly or indirectly. A host computer (e.g. PG, PC or PLC) can control the loading or archiving of a station (e.g. PLC) using domain services. This situation is illustrated by the following figure.

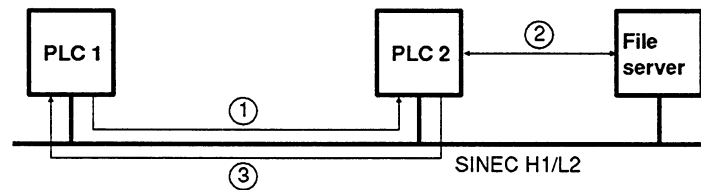


- ① "Load domain content" request
- ② Loading sequence
- ③ "Load domain content" acknowledgement

Fig. 5.1: Loading the Local PLC

5

The host computer can, however, also request the second station to load itself from a third station (e.g. a filesaver) or to save the domains on a third station. This is known as a third-party association. Before loading blocks, the station to be loaded establishes a communications link with the filesaver. This situation is illustrated by the following Figure



- ① "Load domain content" request
- ② Loading sequence
- ③ "Load domain content" acknowledgement

5.2 Loading a Further PLC (third-party association)

Third-party associations are supported by the CP.

This results in the following sequence between the devices:

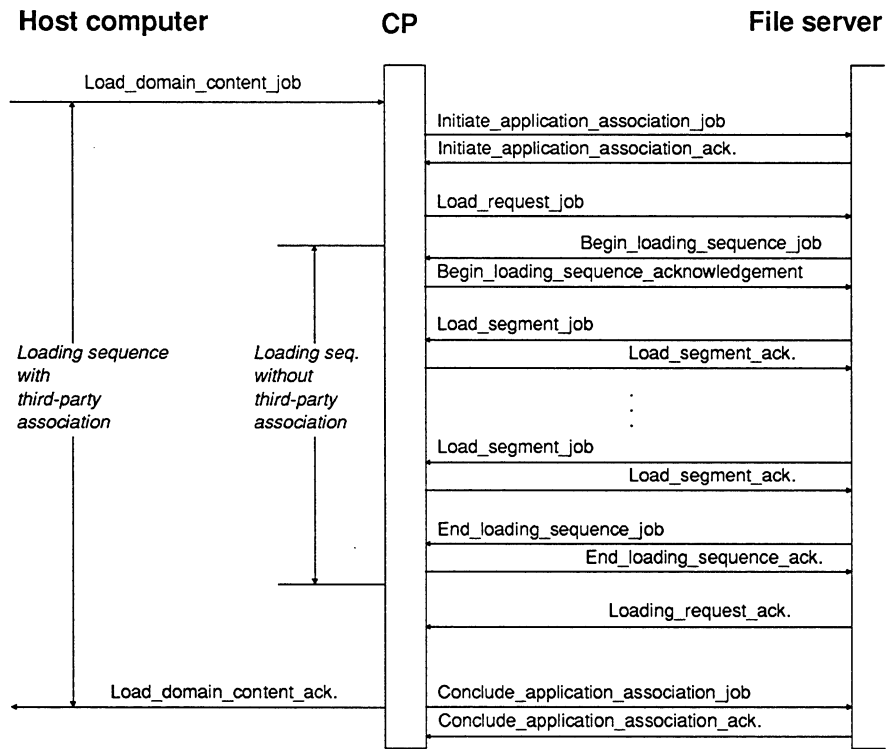
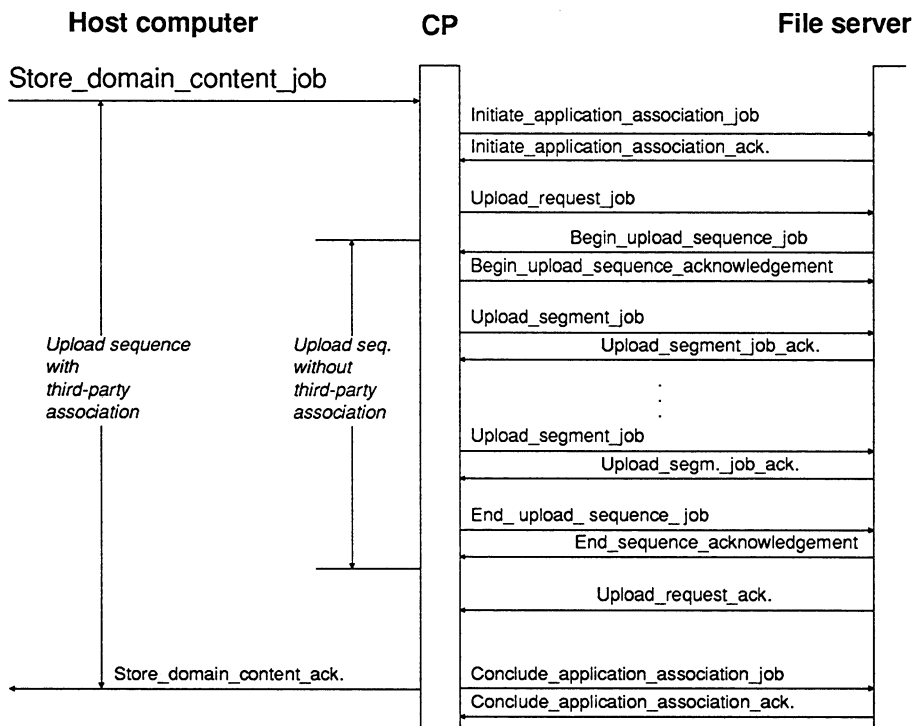


Fig. 5.3 Load Sequence with/without Third-Party Association

The sequence of the upload sequence (archiving) is as follows:



5

Fig. 5 - 4 Upload Sequence with/without Third-Party Association



For all domain services, the CP requires the "swing cable". This means that load services and the PG bus communication function cannot be used simultaneously. If this is attempted, one of the two jobs will be rejected.

5.1.1 Load Domain Content

Using the "load domain content" service, a PLC program can load a domain from a fileserver (i.e. load part of the PLC program). The destination station can either be a different network station or the PLC's own station.

The PLC loads a domain in its own station by using the send direct job with job number 205.

Loading a domain first initiates a compress memory function on the AS511 interface to the CPU.

Job buffer "load domain content"

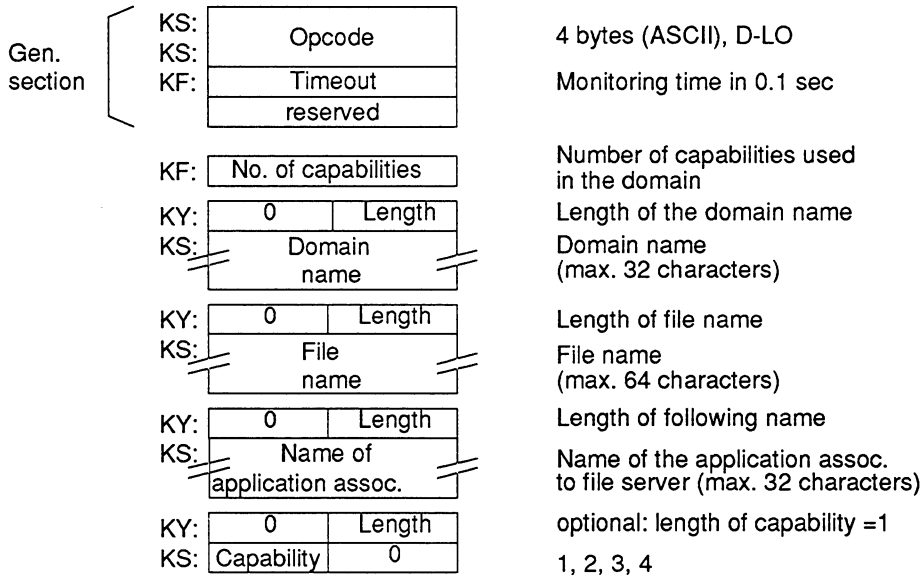


Fig. 5 - 5 Structure of the Job Buffer for TF Service "Request Download"

Call description

General section

Opcode = D-LO

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). remember that the time required for loading depends on the length of the domain.

For more information about timeout, see page 3 - 9.

Job-related section

Number of capabilities: Format: KF
 Range of values: 0, 1 (for SIMATIC applications)

5

Meaning:

0: no capabilities specified, i.e. the domain is loaded in CPU 1 (only permitted for single processor operation) (default)

1: a capability is specified that will be used by the domain. With SIMATIC PLCs, the capability is always the CPU number in which the domain will be loaded.

It is, however, possible to specify more than one capability if this is required in the destination station. If the SIMATIC S5 PLC is operating as the server, this job request is acknowledged negatively.

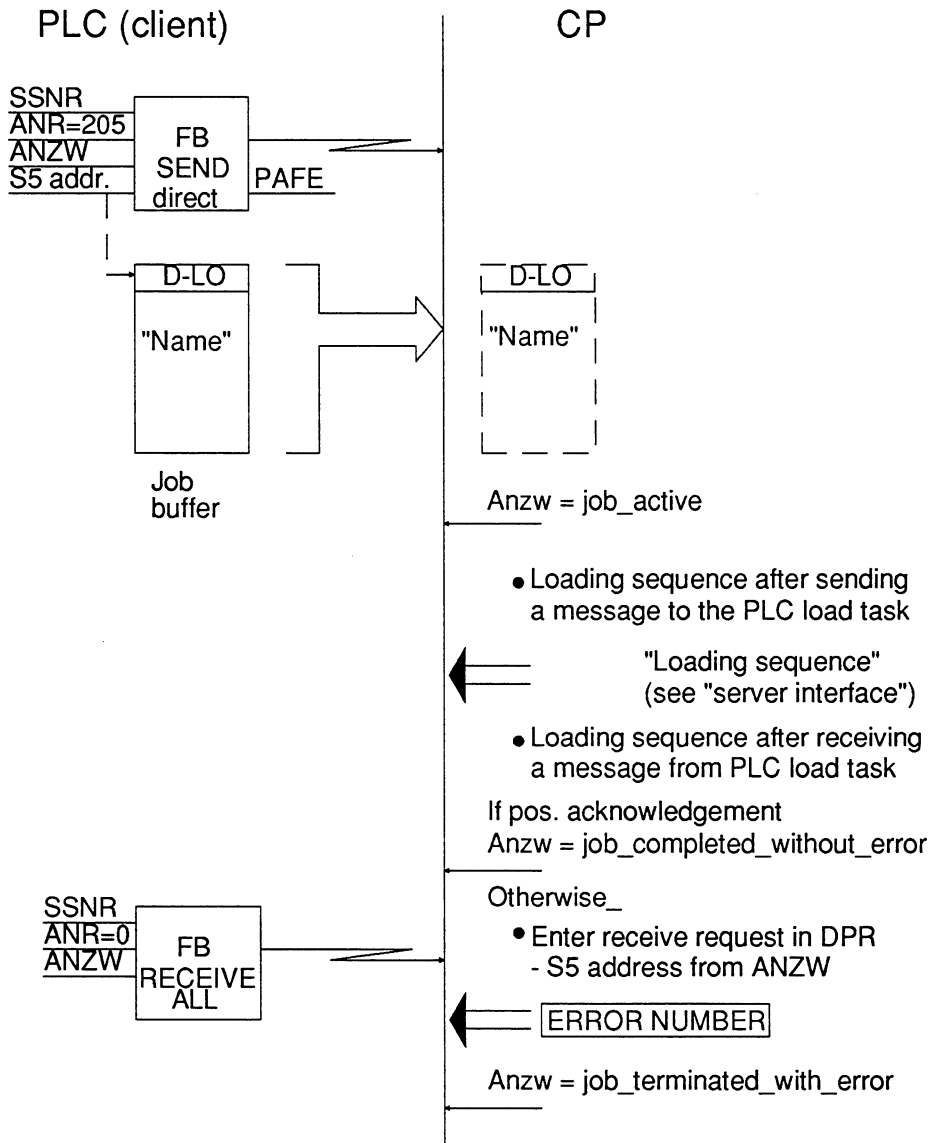
Length of domain name: Format: KY
 Range of values: high byte: 0
 low byte: 1..32
 Meaning: length of the following domain name

Domain name: Format: KS
 Name of the domain to be loaded.
 If the length of the domain name is odd, a padding byte is appended.

Length of file name.	Format: KY Range of values: high byte: 0 low byte: 1...64 Meaning: length of the file name (including path)
File name.	Format: KS Meaning: name of the file on the fileserver containing the domain.
Length of name of appl. ass. to fileserver:	Format: KY Range of values: high byte: 0 low byte: 1...32
Name of appl. ass. to fileserver:	Format: KS Meaning: name of the link via which the fileserver can be obtained by the destination station (third-party association).;
Length of capability:	Format: KY Range of values: high byte: 0 low byte: 1 (bei SIMATIC S5) Meaning: length of the first capability. The first capability describes the CPU belonging to the domain.
Capability:	Format: KS Range of values: for SIMATIC S5 feasible: 1, 2, 3, 4 Meaning: specifies the CPU number in which the domain is to be loaded.

Note:

The job buffer illustrated in the diagram only contains one capability, since only one is necessary for SIMATIC S5. When generating the buffer with the TF editor, only one capability can be specified. If more are required for links to other systems, the capabilities list and the number can be increased. The job buffer must not, however, exceed a length of 256 bytes.



5

Fig. 5 - 6 Sequence "Load-Domain-Content" (local PLC)

Sequence description "load-domain-content"

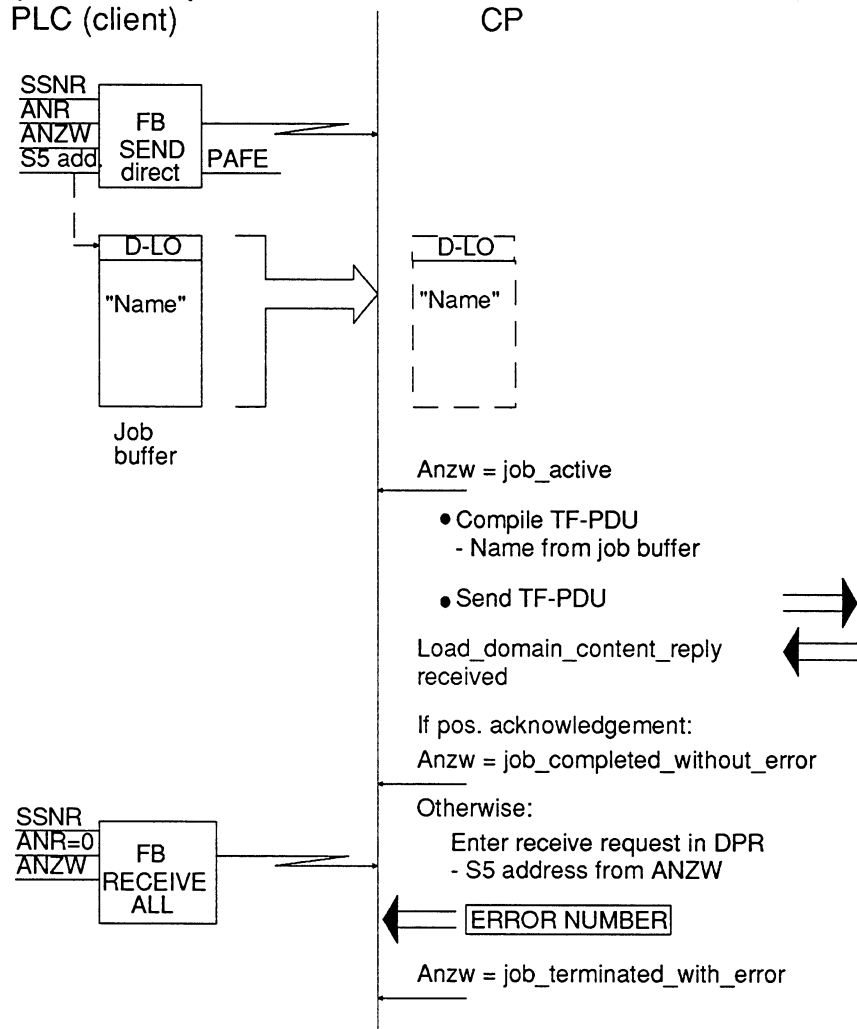


Fig. 5 - 7 Sequence "Load-Domain Content" (second PLC)

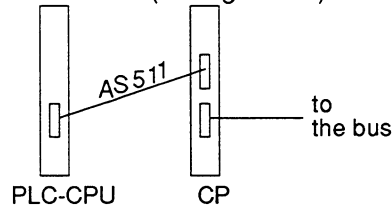
Points to note with the load domain content:

Stipulating the CPU route for loading

The "capabilities list" parameter informs the CP for which CPU the domain is intended. The CP expects the CPU number in the capabilities list. If no capabilities list is specified, CPU number 1 is assumed. There are the following possible "routes" to the PLC that the user must select:

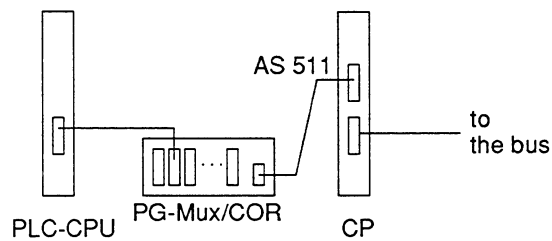
- Direct, i.e. via the AS 511 connection ("swing cable") to the PLC

Configuration:



- Route via PG Mux/COR

Configuration:



In the multiprocessor mode, the VMD configuration must be defined using the COM 143 function Edit | VMD configuration.

Insufficient memory on the PLC for loading a domain

If the error "insufficient memory" occurs when loading a domain, the sequence is aborted and the parts that have already been loaded are deleted. In this case, the error must be dealt with locally. The CP attempts to prevent this error as far as possible by compressing the memory after generating a program invocation (see "Program Invocation Services").

Assigning domains to a CPU

A maximum of eight loaded dynamic domains is supported by the CP. The distribution of the domains on the CPUs (1...4) can be decided by the user. The capabilities list informs the CP of the CPU in which the domain is to be loaded.

Block allocations must be unique

If a domain contains a block that is also part of another domain loaded in this CPU, the loading function is aborted. Blocks contained in the PLC that do not belong to another domain are overwritten.

Managing domains on the CP

The data structures required to manage the domain are stored on the CP. They are stored in battery-backed areas and are retained if there is a power failure.

Configuring and establishing the fileserver application association

The link to the fileserver specified by the "application association name" parameter is set up by the CP (see application association management - special links). This is not defined in the same way as a normal application association, but in COM 143 in the "fileserver application associations" screen form.

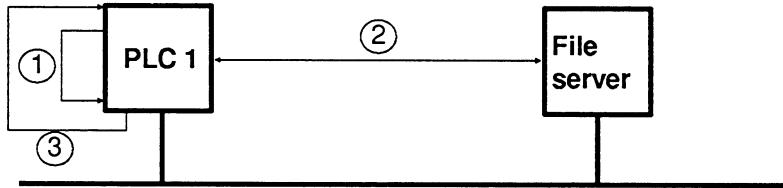
After the loading sequence is complete, the fileserver application association is terminated again and the load job acknowledged.



Domains must not contain blocks that exist in the EPROM of the destination PLC (e.g. HDBs on the S5-115U).

5.1.2 Store domain content

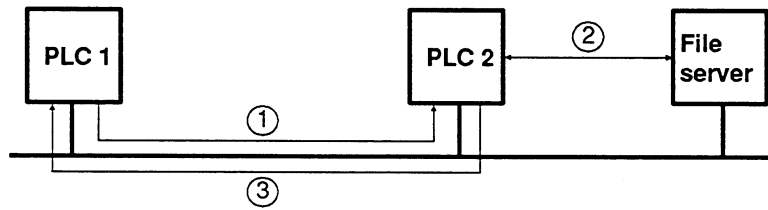
Using the "store domain content" service, the content of a (loaded) domain can be archived on a fileserver. The domain can either exist locally (Figure 5-8) or in a remote PLC (Figure 5-9).



- ① "Store domain content" request
- ② Storing sequence
- ③ "Store domain content" acknowledgement

5

Fig. 5 - 8 Archiving a Domain of the Local PLC



- ① "Store domain content" request
- ② Storing sequence
- ③ "Store domain content" acknowledgement

Fig. 5 - 9 Archiving a Domain of a Remote PLC

The domain is remote

The application association to the second PLC is identified by the call parameters "SSNR" and "ANR" of the "send-direct" call to transfer the job buffer.

The domain is local

The archiving of a local domain is achieved by using the VMD configuration job number (205).

Job buffer "store-domain-content"

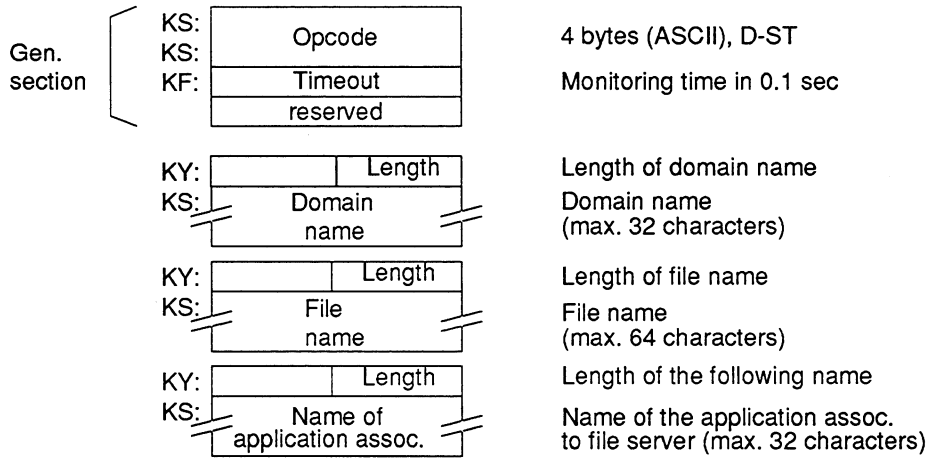


Fig. 5 - 10 Structure of the Job Buffer for TF Service "Request Upload"

Name of the application association to the fileserver:

Format: KS

Meaning: name of the appl. ass. via which the fileserver can be obtained by the dest. station (third-party association).

Sequence description "store-domain-content"

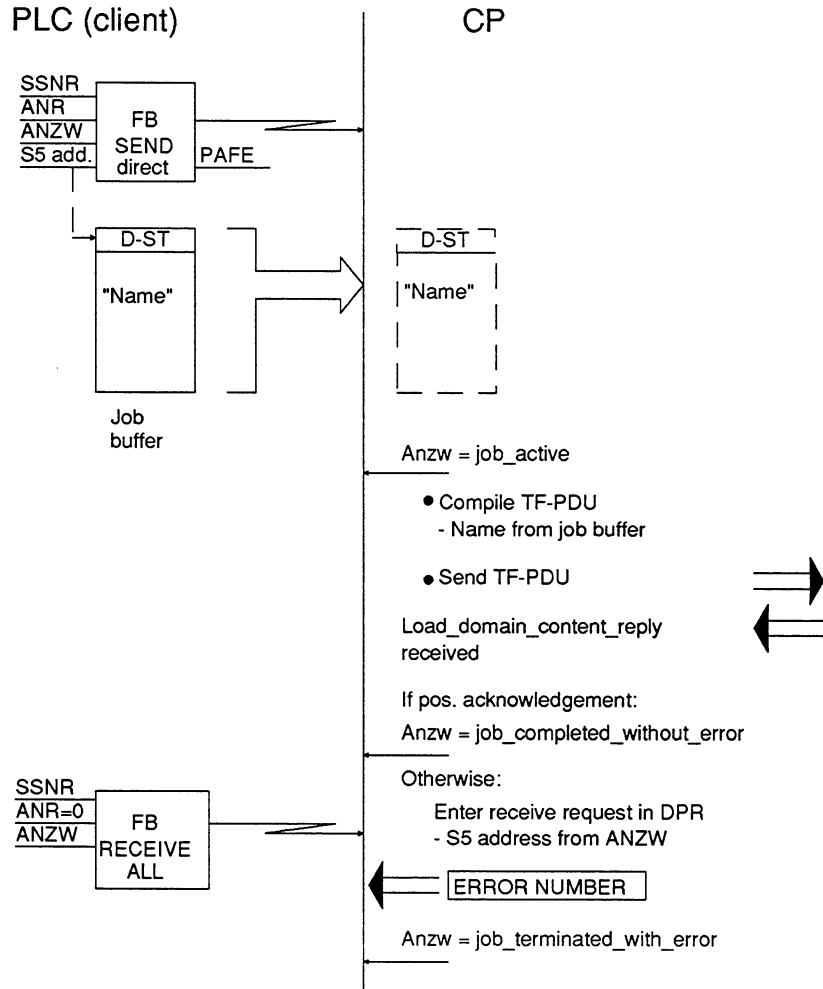
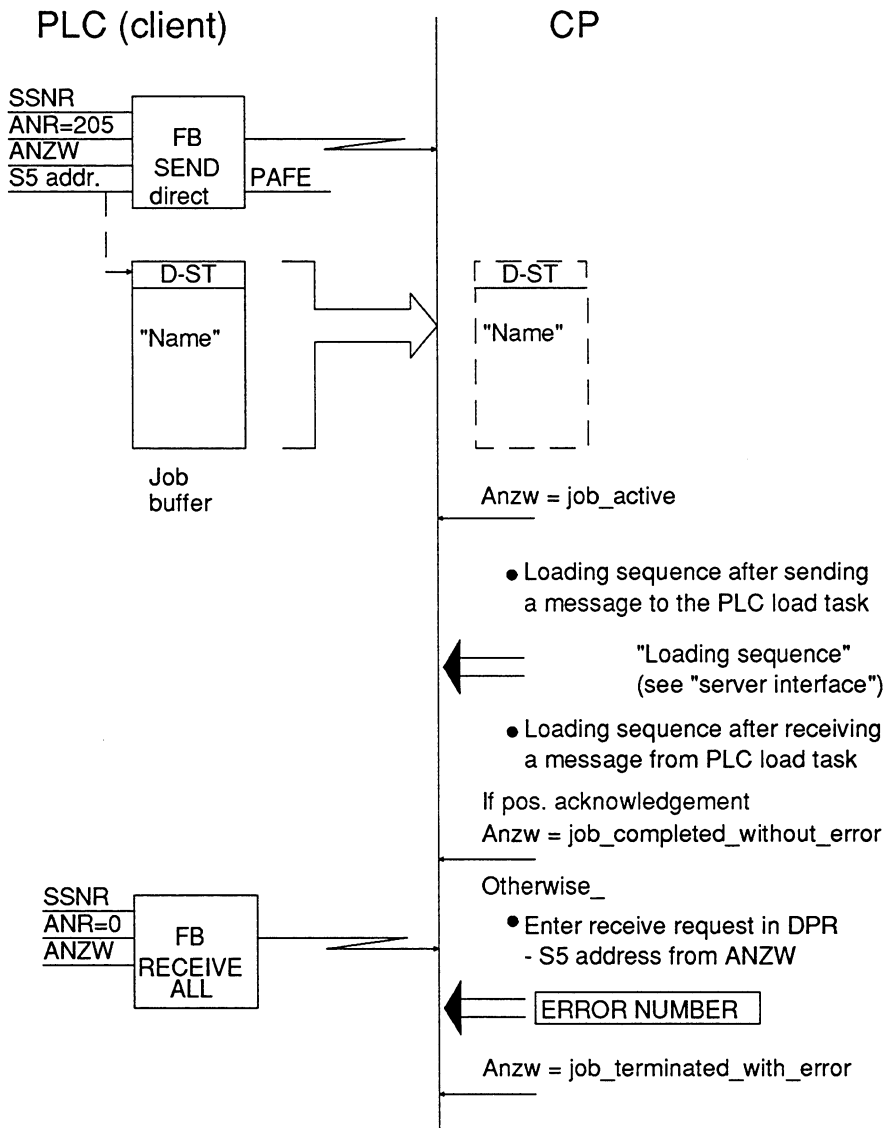


Fig. 5 - 11 Sequence 'Store Domain Content (second PLC)



5

Fig. 5 - 12 Sequence 'Store Domain Content' (local PLC)

5.1.3 Delete Domain Content (Client)

This job deletes a domain identified by the domain name. It can be loaded either locally or on a remote PLC.

Deleting a domain initiates a compress memory function on the AS511 interface to the CPU.

Job buffer "delete domain content"

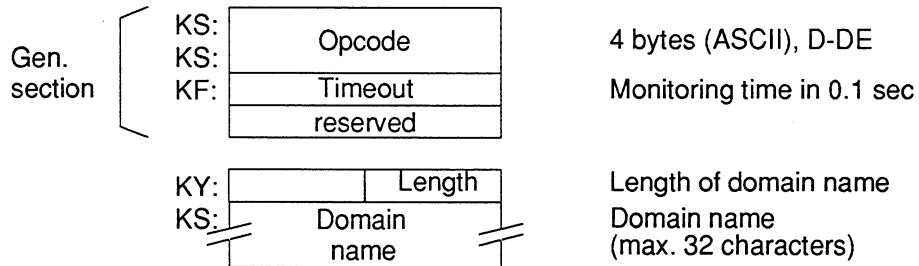


Fig. 5 - 13 Structure of the Job Buffer "Delete Domain"

Call description

General section:

Opcode D-DE

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.
 For more information about timeout, see page 3 - 9.

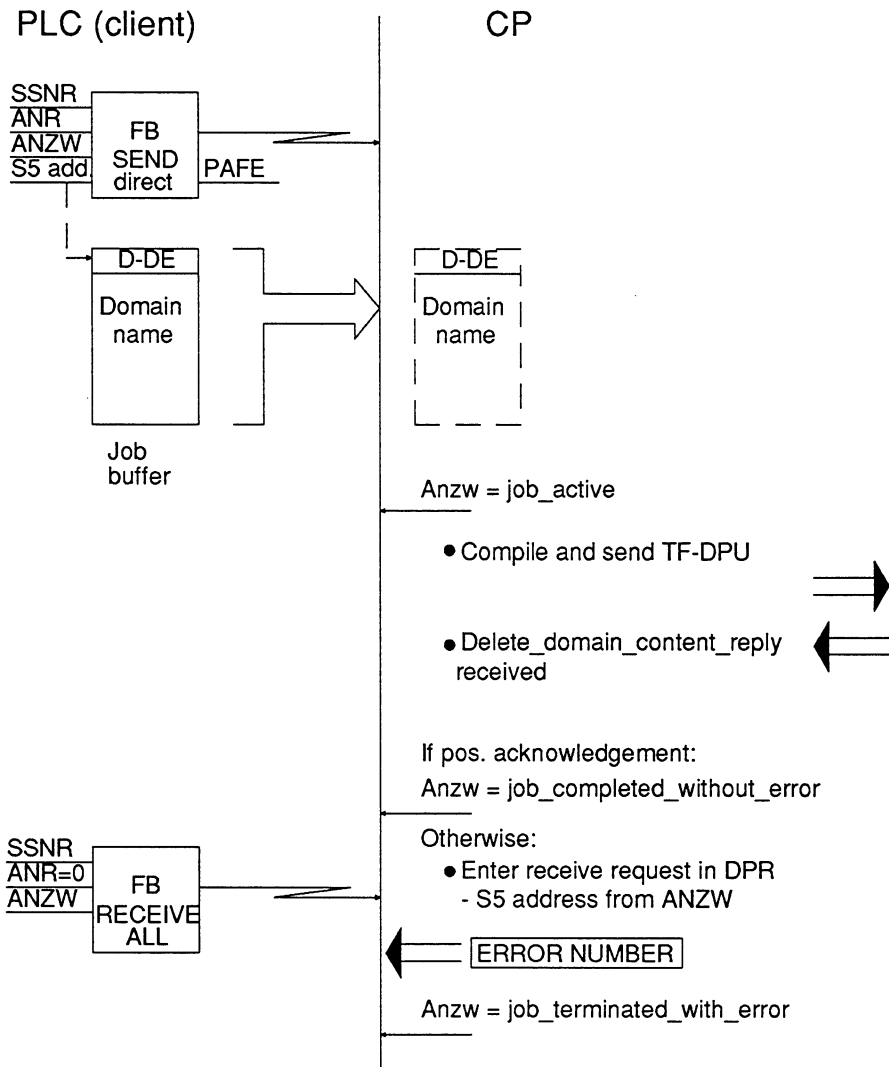
Job-related section

Length of domain name: Format: KY
 Range of values: high byte: 0, low byte: 1..32
 Meaning: length of following domain name.

Domain name: Format: KS

Meaning: name of the domain to be deleted, if the length of the domain name is odd, a padding byte is appended.

Sequence description "delete domain content"



5

Fig. 5 - 14 Sequence "Delete Domain" (other PLC)

5.1.4 Get Domain Attributes (Client)

This service requests the attributes of a particular domain. Domain attributes are information about capabilities, status information and information about the PI assignment.

On receiving the job, the server checks whether or not the domain with the domain name exists. If no domain exists with this name, a negative acknowledgment is sent.

Job buffer "get domain attributes"

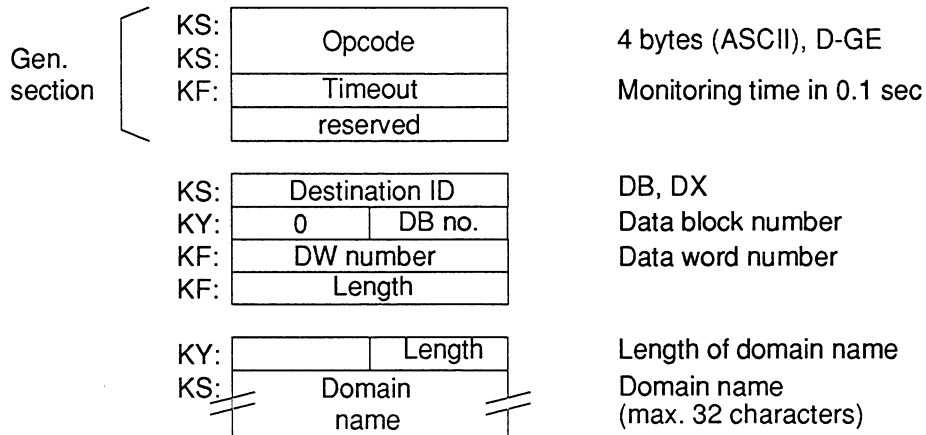


Fig. 5 - 15 Job Buffer "Get Domain Attributes"

Call description

General section:

Opcode D-GE

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified

in multiples of 0.1 sec.

For more information about timeout, see page 3 - 9.

Job-related section

Dest. ID	Format: KS Range of values: DB, DX Meaning: S5 destination address, at which the information about the attributes of the domain will be stored.
DB number	Format: KY Range of values: high byte: 0, low byte: 1..255
DW number	Format: KF Range of values: 0..2042
Length:	Format: KF Range of values: 1..2043, -1 Meaning: length of the data block area in which the domain attributes can be stored; the value -1 indicates that all the domain attributes sent in the acknowledgement from the DW number up to the end of the data block can be accepted.
Length of domain name:	Format: KY Range of values: high byte: 0, low byte: 1..32 Meaning: length of the following domain name
Domain name:	Format: KS Name of the domain if the length of the domain name is odd, a padding byte is appended.

Sequence description "get domain attributes"

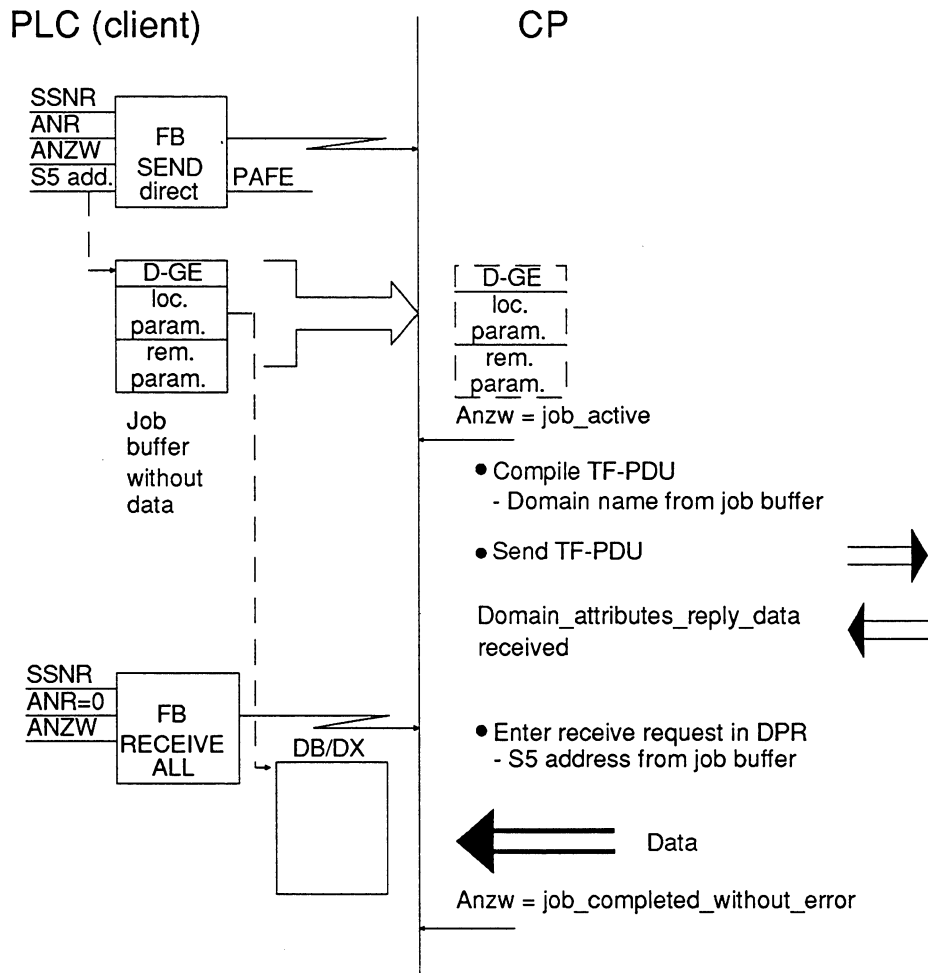
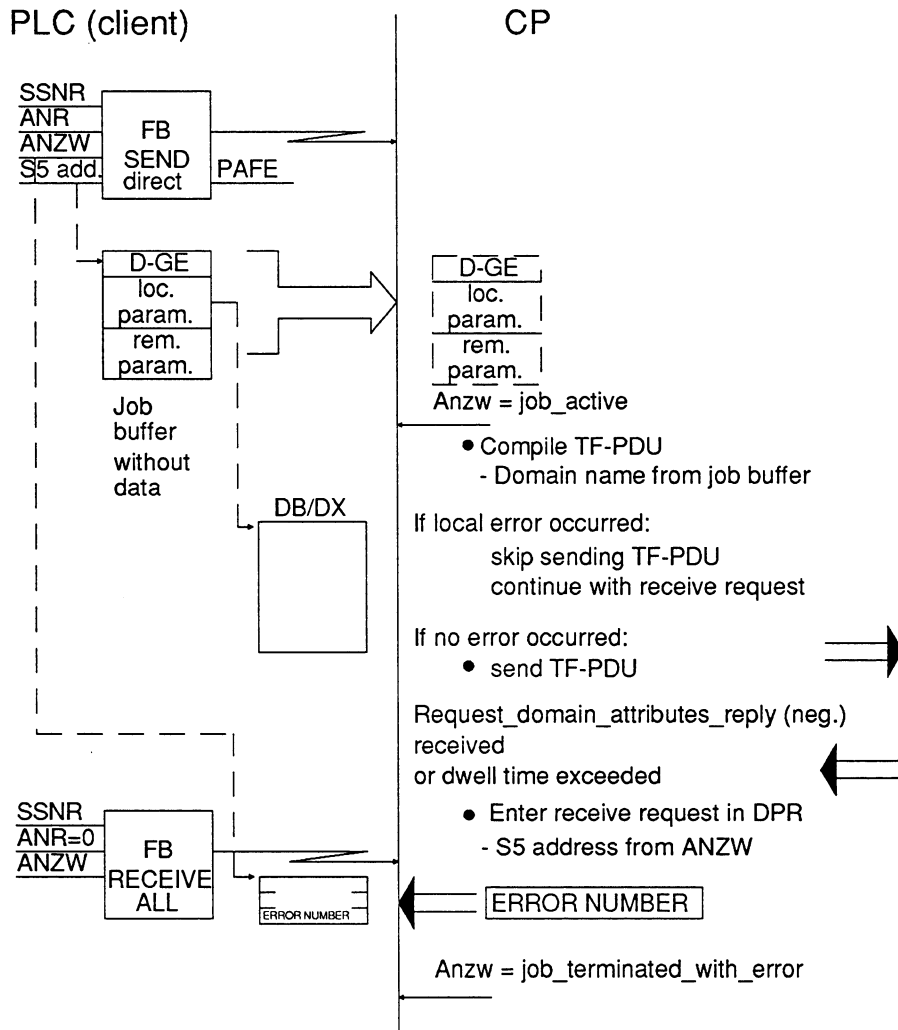


Fig. 5 - 16 Sequence Description (Get Domain Attributes, Positive Acknowledgment)



5

Fig. 5 - 17 Sequence Description (Get Domain Attributes, Negative Acknowledgment)

Structure of the reply data

The reply data stored at the S5 address specified in the job buffer have the following structure:

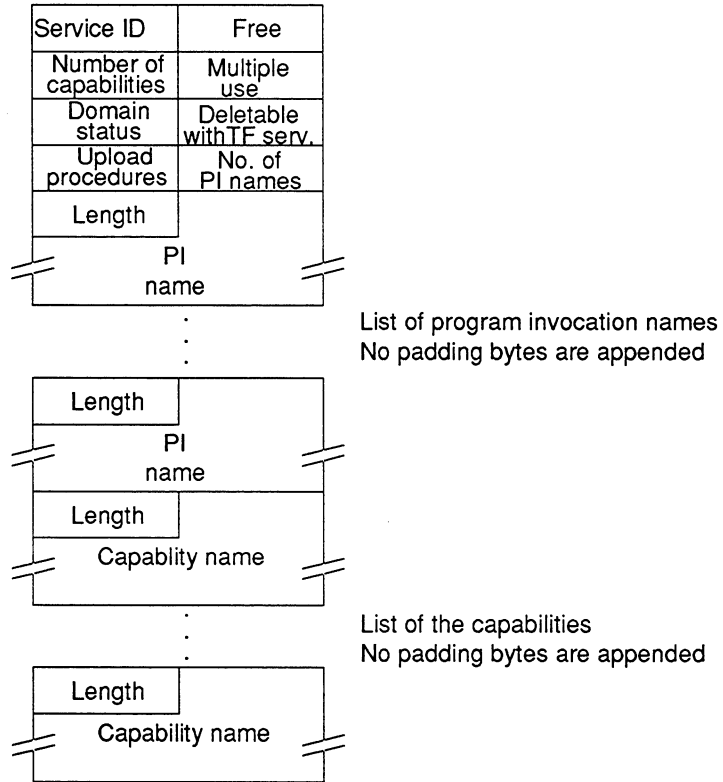


Fig. 5 - 18 Structure of the Reply Data

Meaning of the individual parameters in the reply data:**Service ID**

25h (for unique assignment of the reply to the requested service)

Number of capabilities:

number of the entries contained in the capabilities list parameter

Multiple use:

specifies whether the domain can be used by more than one program invocation at any time, for SIMATIC S5 stations always "false"

Domain status:

- 0 the domain does not exist (reserved value)
- 1 the domain is being loaded, the CP is currently processing data
- 2 the domain is loaded
- 3 the domain is loaded and assigned to a program invocation
- 4 the domain is loaded but the loading sequence is not yet completed
- 5 the loading procedure for the domain was aborted
- 6 (reserved)
- 7 the domain is being generated
- 8 the domain is being loaded, the interface module is waiting for data
- 9 the loading sequence is being terminated
- 10 the domain is being assigned to a program invocation
- 11 the domain is being assigned to a further program invocation (not with SIMATIC S5)
- 12 the domain is being released by a program invocation, however, it remains linked into another program invocation (not with SIMATIC S5)
- 13 the domain is being released by a program invocation and is changing to status 2
- 14 the domain is being deleted
- 15 The loading procedure was aborted, the domain is being deleted.

Statuses 7...15 are only temporary, i.e. between a job currently processed and the corresponding reply.

Deletable with TF services:

Specifies whether the domain can be deleted with the "delete domain service".

Upload procedures:

Specifies whether the domain is currently being archived on a fileserver. If this parameter is "true" the domain cannot be deleted at the present time.

Number of program invocation names:

Number of elements in the program invocation_list, with SIMATIC S5, a maximum of one.

Program invocation_list:

Contains the names of the program invocations currently occupying the domain, with SIMATIC S5, a maximum of one.

Capability list:

List of the capabilities used by the domain, for SIMATIC S5 the CPU number which the domain is loaded is specified (see also job description "load domain content").

5.1.5 Domain Services (Server)

On the server, the service is executed automatically in the CP without support of the PLC program. The tables or data structures addressed are only in the address area of the CP and contain the domain and PI object attributes known in the CP.

The swing cable is required for this service.

5.2 Program Invocation Services

The program invocation services support you in controlling the programs which process the automation tasks in the programmable logic controllers. With these services you can control the functions of these devices dependent on their processing status.

These services are available for PLCs in the client and server roles.

In the **client role**, the PLC functions as a host computer requesting status changes in the monitored device by means of service jobs.

In the **server role**, a PLC reacts to the instructions contained in the service job. The CP is responsible for keeping the PI statuses up to date in the PI management, interpreting the instructions and checking that they are permitted. The instructions must then be converted in the PLC user program and acknowledged. After the acknowledgment, the CP notes the status change in its PI management.

The following services are supported (in each case both on the client and server):

- > Create PI
- > Delete PI
- > Start PI
- > Stop PI
- > Resume PI
- > Reset PI
- > Kill PI
- > Local program stop (only server function -> local service)
- > Get PI attributes

5.2.1 PLC Program Structure, Status Transitions

PI in the SIMATIC S5 PLC

In the SIMATIC S5 PLC there are two program invocations defined:

➤ The system PI (PLC_START_STOP)

The system PI represents the PLC with its global start/stop procedure. The system PI ensures that the status of the whole device is independent of the statuses of the user PI. No status of a user PI can, for example, automatically cause a PLC stop.

➤ The user PI

The user PI includes all the domains defined on the PLC. Remember that apart from the blocks grouped to form domains, other blocks can exist on the PLC. The separate view of the user PI and the system PI allows non-domain blocks to be run independent of the user PI.

Up to eight domains (sets of blocks) can be stored on a SIMATIC S5 PLC. Along with any permanently loaded blocks (option) these domains perform the actual PLC program. A "program invocation" (PI) as available in the TF model for sequential control of an application process, is always formed by the domains loaded in the PLC. The modelling of possible program invocation statuses is explained in the following sections.

Dynamic and static domains

Normally, the user PI represents the domains that can be loaded using the loadable domains. These are also known as dynamic domains.

A static domain has been defined for the special situation where the user wishes to control the program using PI services but is not using the domain services. The static domain includes the PLC blocks not explicitly assigned to a domain. By generating a PI related to this static domain, PI services can be used with the blocks of this "pseudo" domain.

PI status management

The management of the TF-PI statuses (see illustration) is performed by the CP alone. The PLC program is informed of status change requests by PI jobs, for example, a request to the application to change to the stop status). The PLC user program executes the service by changing the status as requested and acknowledging the request according to the process status.

Note: Due to its coordinating role, the client is also known as a **host** computer in conjunction with the PI services.

PI status indication via FB 103 (PI-ZUSTD)

The information about the actual PI status and any status change requested by the host computer is passed on by a standard function block (FB PI-ZUSTD), that can either be loaded permanently in the PLC or be part of a domain. This block can be called in any program branch by the application program.

The standard function block FB 103 (FB PI-ZUSTD) is supplied with COM 143 TF.

5

Status transitions

The PI status changes defined in TF can be seen in the diagram. Whether or not a status transition requested by the host computer is permitted is checked in the CP.

Multiprocessor mode

In the multiprocessor PLC, the view of the PLC via the system PI and the user PI remains unchanged. The number of possible domains also remains unchanged. The information depends on the number of CPUs in operation!

In the multiprocessor mode, the only difference is that a master CPU must be specified with which the CP executes the system PI functions (START/STOP PLC). The master CPU is specified using the COM 143 TF configuration tool, with the function VMD Configuration.

Checklist for using PI services

Using PI services means the following:

In the PLC with the server role:

- Configure an application association for transferring the PI services (it may be possible to use the predefined application association S5_TF; refer to the Chapter Supplementary Services, Section Special Links).
- Assign the data and program blocks to domains (refer to domain services and the PGLOAD tool).
- Specify the reactions to PI jobs in the PLC user program, i.e. what it means to resume, reset, start or stop a PI etc.
- Call the function block for the status request (FB 103) in the PLC program.
- Evaluate the status information in the PLC program and if applicable send an acknowledgment from the PLC program.

In the PLC with the client role:

- Configure an application association for transferring the PI services (it may be possible to use the predefined application association S5_TF; refer to the Chapter Supplementary Services, Section Special Links).
- Send PI jobs via handling blocks if status changes are required in the server PLC.
- Evaluate status information, i.e. the acknowledgments of the server PLC in the PLC program.

- 15 Kill acknowledgment (positive)
- 16 Reset job
- 17 Reset acknowledgment (positive; if PI is re-usable)
- 18 Reset acknowledgment (positive; if PI not re-usable)
- 19 Reset acknowledgment (negative, non-destructive)
- 20 Reset acknowledgment (negative, destructive)
- 21 Local program stop
- 22 Create_PI job
- 23 Create_PI acknowledgment (positive)
- 24 Create_PI acknowledgment (negative)
- 25 Delete_PI job
- 26 Delete_PI acknowledgment (positive)
- 27 Delete_PI acknowledgment (negative)

The term "destructive" in this situation means that the PI can then no longer be used.

Description of the PI statuses for the System PI

The system PI controls the START/STOP response of the PLC. The following service jobs are possible:

- A resume PI job generates a PLC start on the AS511 interface, but does not influence the status of the user PI. The PLC memory is not compressed.
- A stop PI job generates a PLC stop on the AS511 interface, but does not influence the status of the user PI.

Further status changes of this system PI are not allowed and are acknowledged negatively.

Description of the PI statuses for the user PI

Some of the effects of PI jobs on the user process must be programmed in the PLC. After the application has requested a specific PI status (FB 103) the status change is triggered and after acknowledgment, it is entered in the status management of the CP.

PI non-existent	A program invocation has not been created.
PI idle	The PI has been created, the actual user process has, however, not yet started. This status follows a create PI job.
PI starting	The process controlled by the PLC program will be started. In this status, the user program can, for example, make certain preparations to allow the transition to the "running" status. The change of status (to running or unrunnable) is triggered by the user program (acknowledgement of the start PI job).
PI running	The user process has started and is being controlled by the PLC program.
PI stopping	The process controlled by the PLC program will stop. In this status the user program can for example make certain preparations to allow the transition to the "stopped" status. The change of status (to stopped or unrunnable) is triggered by the user program (acknowledgement of the PI start job)
PI stopped	The process controlled by the PLC program is (temporarily) stopped. As can be seen from the PI status diagram, the process can be completely stopped in this status (for example because another program is to be loaded) or can be resumed by a command from the host computer, i.e. can change back to the "running" status. When in the "running" status, the change to stopped can also be caused by a local event (see description of FB "PI-ZUSTD").

- PI resuming The process controlled by the PLC program will be resumed after a temporary stoppage. The change from this status (to running or stopped/unrunnable) is triggered by the user program (acknowledgement of the resume PI job).
- PI resetting The process will be returned to the idle state by the user program. Changing from this state to the idle or stopped/unrunnable status is triggered by the user program (acknowledgement of the reset PI job).
- PI unrunnable The "unrunnable" status means that an event has occurred during the control of the process that prevents further processing. This may, for example, be the result of one of the following causes:
- the host computer has sent an abort request
 - the user program acknowledges a status transition "negative, destructive".

The statuses P1 to P4 are managed by the CP. They are decision phases on the CP when the CP decides which of the alternative statuses will result from a status request. The user program has no influence on these transitions.

5.2.2 General Sequence of a Status Change

The description of the sequence of a status change illustrates the interaction of the user program on the PLC and the status processing on the CP. The steps from the arrival of the PI job on the CP until the new status is acknowledged are described.

The following steps are run through on the CP 143:

- Request received
- Requested status change checked for consistency
- If status change not allowed, negative acknowledgement and abort

Otherwise:

New status entered in the DB-RAM. The transmission of an acknowledgement is not time-monitored

- If applicable, wait for acknowledgement from the user program

The following steps are run through on the programmable logic controller:

- Check the PI status with FB "PI-ZUSTD" call. Processing of a program section dependent on the PI status
- If a status with an acknowledgement request is set in the PI status, the user program reacts as follows:

It recognizes the request and prepares the process for a new status (Running, Stopped, Idle). It acknowledges the job positively.

If the process is already in the status requested by the host computer, or it is not possible to change to this status the job must be acknowledged negatively (by calling FB "PI-ZUSTD").

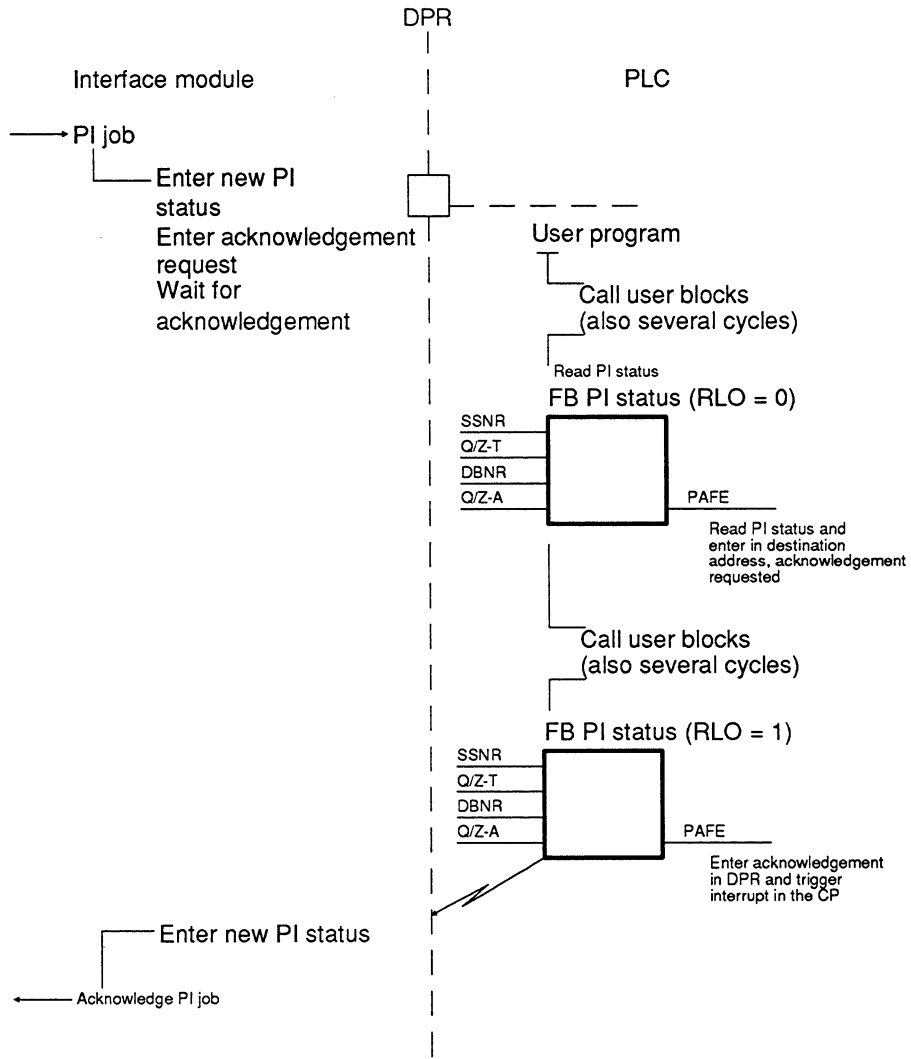


Fig. 5 - 20 General Sequence (no error) of a Status Change

5.2.3 Interface of the PLC Program to the PI Services on the Server

Significance of FB 103

The function block "PI-ZUSTD" (FB 103) is used to inform the user program of the current status of the program invocation and therefore represents the interface of the user program to the program invocation services of TF.

Block function

The block executes two different functions: read PI status or send acknowledgement. The function block is processed dependent on the result of logic operation (RLO), as follows:

- > Call when
RLO = 0:
The PI status is read and stored at the specified S5 address
- > Call when
RLO = 1:
The user wants to acknowledge a PI status, either positively or negatively. The FB reads the acknowledgement at the specified S5 address and then writes the new PI status at this address.

5

Block interface

As can be seen in the diagram illustrating the general sequence of a status change, the block requires various parameters:

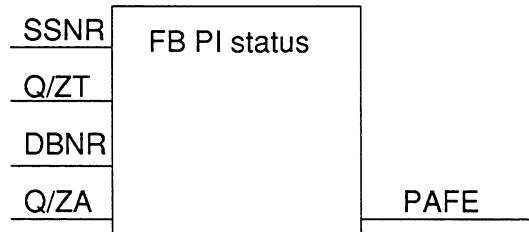


Fig. 5 - 21 Call Parameters of the Status Function Block "PI-ZUSTD"

Description of the interface parameters

1. SSNR (interface number)

The value specified here has the same significance as when calling a handling block and determines the page via which the PI status in the CP can be scanned. This value is also the value assigned to the application association on which the host computer triggers the PI services. The base interface number of the module must be set according to the value used here.

Special feature in multiprocessor PLCs: as with communication via handling blocks, the interface number is determined by the slot on the CPU. This means that the CPU in slot n (n=1..4) obtains the PI status via interface number n-1 (+ base interface number). The interface module ensures that the same PI status is read in all CPUs.

2. Q/ZT (source/destination type)

Depending on the function of the block, this parameter specifies the S5 source or S5 destination type to be used by the block.

Identifier:

DB data block area

DX extended data block area

FW flag area

3. DBNR (data block number)

If Q/ZT =DB or DX: 1...255

If Q/ZT = FW: invalid.

4. Q/ZA (source/destination start)

This parameter specifies the start address within the selected area.

If Q/ZT = DB or DX:0...2042

If Q/ZT = FW: 1...(dependent on PLC type)

Since one word of the block is always required, no length needs to be specified.

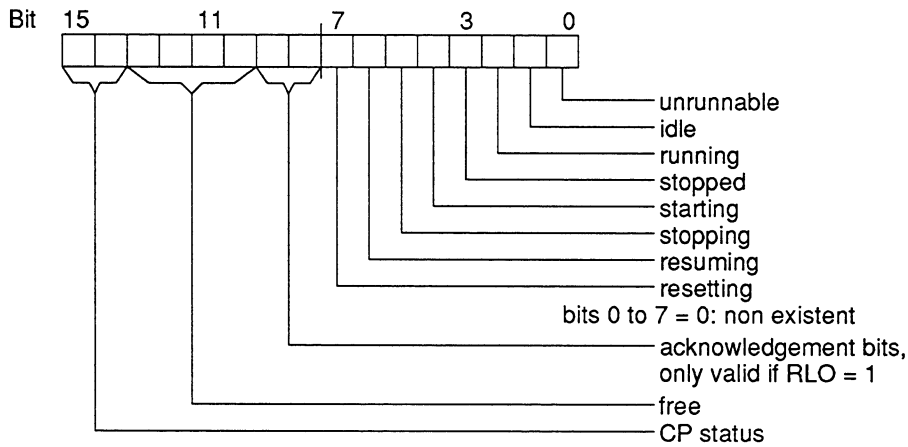
5. PAFE (parameter assignment error)
Parameter assignment error, 1 byte
The block transfers any errors that may occur to the user program at the FB output (byte parameter) specified by this address (see following table).

List of possible error messages and their causes:

00h	no error occurred
11h	parameter "Q/ZT" not correctly entered, i.e. not DB/DX/FW
21h	access to address not possible, e.g. DB does not exist
31h	area too small, i.e. DW does not exist
41h	parameter Q/ZA is greater than 255
51h	RLO = 1, but no acknowledgement requested
61h	CP/PLC not synchronized
71h	access to DPR not possible, interface does not exist
81h	interface not ready
91h	interface overloaded
A1h	free
B1h	free
C1h	interface not acknowledging
D1h	free
E1h	free
F1h	free

5.2.3.1 S5 Files of the Function Block "PI-ZUSTD" (FB 103)

PLC	CPU	S5 file	Library number
S5-115 U	CPU 942A/B CPU 943A/B CPU 944A/B	S5CI50ST.S5D	P71200-S 5103-C-2
S5-135 U	CPU 922 CPU 928A/B CPU 948	S5CI29ST.S5D	P71200-S 9103-C-1
S5-150 U		S5CI40ST.S5D	P71200-S 4103-C-1
S5-155 U	CPU 946/947 CPU 922 CPU 928 A/B CPU 948	S5CI69ST.S5D	P71200-S 6103-C-1

Format and Significance of the Supplied PI Status:

In the following statuses, the user program must generate an acknowledgement:

- Starting
- Stopping
- Resuming
- Resetting

In all other statuses the user program must not generate an acknowledgement.

Significance of the acknowledgement bits

- > Bits 8 and 9 = 0:
The status transition will be executed (pos. acknowledgement)
- > Bit 8 = 1, bit 9 = 0:
The status transition will not be executed, the PI is to remain in the old status (negative, acknowledgement, non-destructive).
- > Bit 8 = 0 or 1, bit 9 = 1:
The PI must be brought to the "unrunnable" status (negative acknowledgement, destructive).

Significance of the CP status

Bit 14 = 1, bit 15 = 0: CP stopped

Bit 14 = 0, bit 15 = 1: CP in run mode

Bit 14 = 1, bit 15 = 1: CP and PLC not synchronized

Notes on the sequence in multiprocessor PLCs:

The PI status is valid for the entire PLC

In a multiprocessor PLC, the CP ensures that the same PI status is always read out in all CPUs. The user must make sure that the "SSNR" parameter is specified correctly when FB "PI-ZUSTD" is called (= CPU number - 1).

In the statuses in which an acknowledgement is required, the user must make sure that the acknowledgement is only triggered when all the CPUs require the status change. The CP itself can only accept one acknowledgement.

Note on the system PI

Since the CP starts and stops the PLC (with the "resume PI" and "stop PI" services for the system PI) the CP must know the configuration of the PLC. This is achieved in the configuring phase with COM 143 TF using the VMD Configuration function with which the user defines a "master CPU" that is controlled via the CP. At this stage, the user also specifies how the CPU is to be started and stopped.

5.2.4 Start-up, Installation

The installation on a SIMATIC PLC that will later be controlled within a system by a host computer using TF services (particularly with the aid of domain and PI services) must be performed as follows:

- The domain must already have been generated with the COM 143 tool PGLOAD. The PG (as an aid to installation) transfers the files as "load files" to a fileserver on completion of the programming (or after changes in the program).
- The PLC should only contain blocks that are not contained in a domain (if these blocks do exist in a domain, they will be overwritten when the domain is loaded).
- Since the CP assumes the status "stopped" as the initial status of the system PI "PLC_START_STOP", the PLC must be in the "stop" mode. In the multiprocessor mode, the CPU designated as "master CPU" during configuration must be set to stop.

The use of variables services is not dependent on the existence of domains or PIs. Only the definition and access to domain-specific variables require that the domain exists.



Following power down, a PI always has the "non-existent" status.

5.2.5 Create Program Invocation (Client)

This service allocates one or more domains to a program. This can be in a remote station (identified by SSNR/ANR) or in the local station (ANR = 205).

Job buffer "create program invocation"

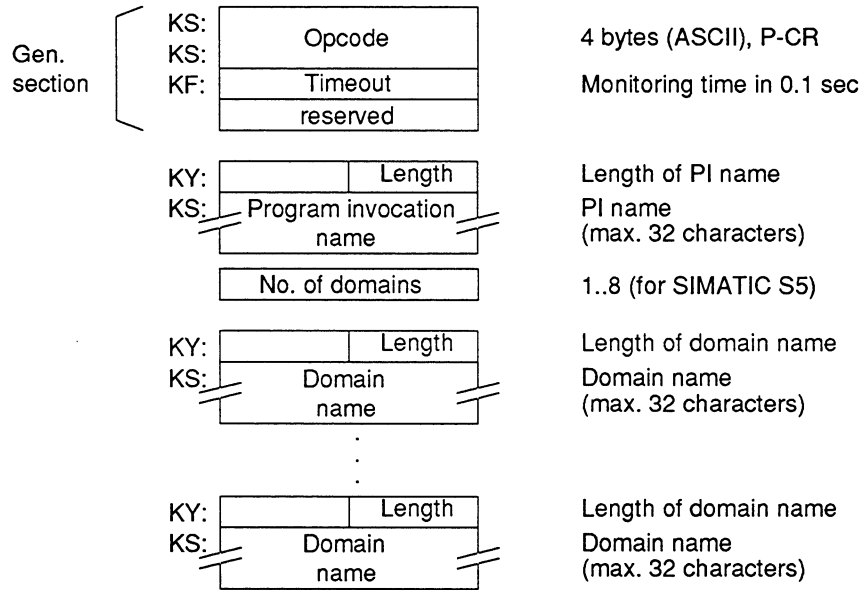


Fig. 5 - 22 Structure of the Job Buffer for TF Service "Create Program Invocation"

Call description

General section:

Opcode P-CR

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec
 For more information about timeout, see page 3 - 9.

Job-related section:

Length of PI name Format: KY
 Range of values: 1..32
 Meaning: length of the following PI name

PI name Format: KS
 Meaning: name of PI to be created. If the length of the name is odd, a padding byte is appended.

Number of domains Format: KF
 Range of values: 1..8
 Meaning: specifies the number of domains to be linked together in a PI. In the job buffer itself, the number of domain names is only limited by the maximum length of the job buffer.

Length of domain name Format: KY
 Range of values: high byte: 0, low byte: 1..32
 Meaning: length of the following domain name

domain name Format: KS
 Meaning: name of the domain, if the length is odd, a padding byte is appended.

Parameters length of domain name and domain name are repeated as often as specified by the "number of domains"

Job sequence "create program invocation"

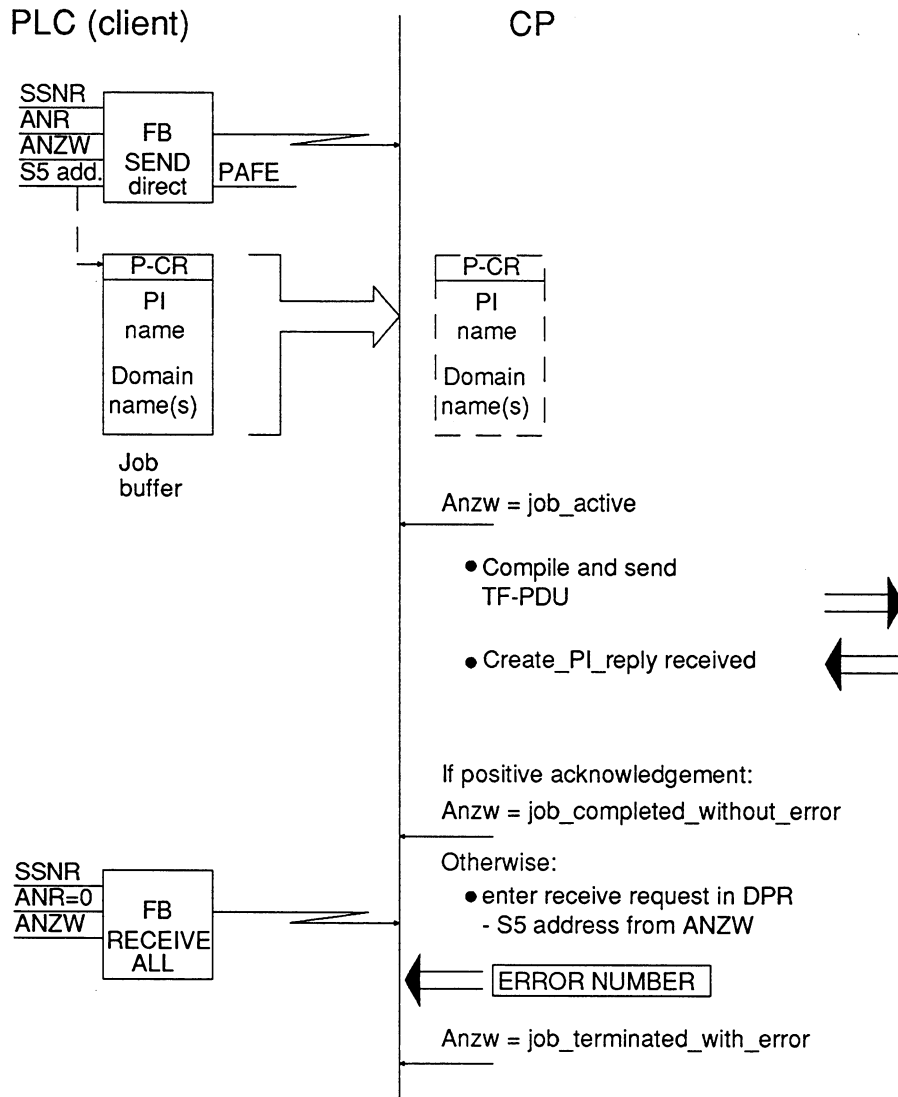
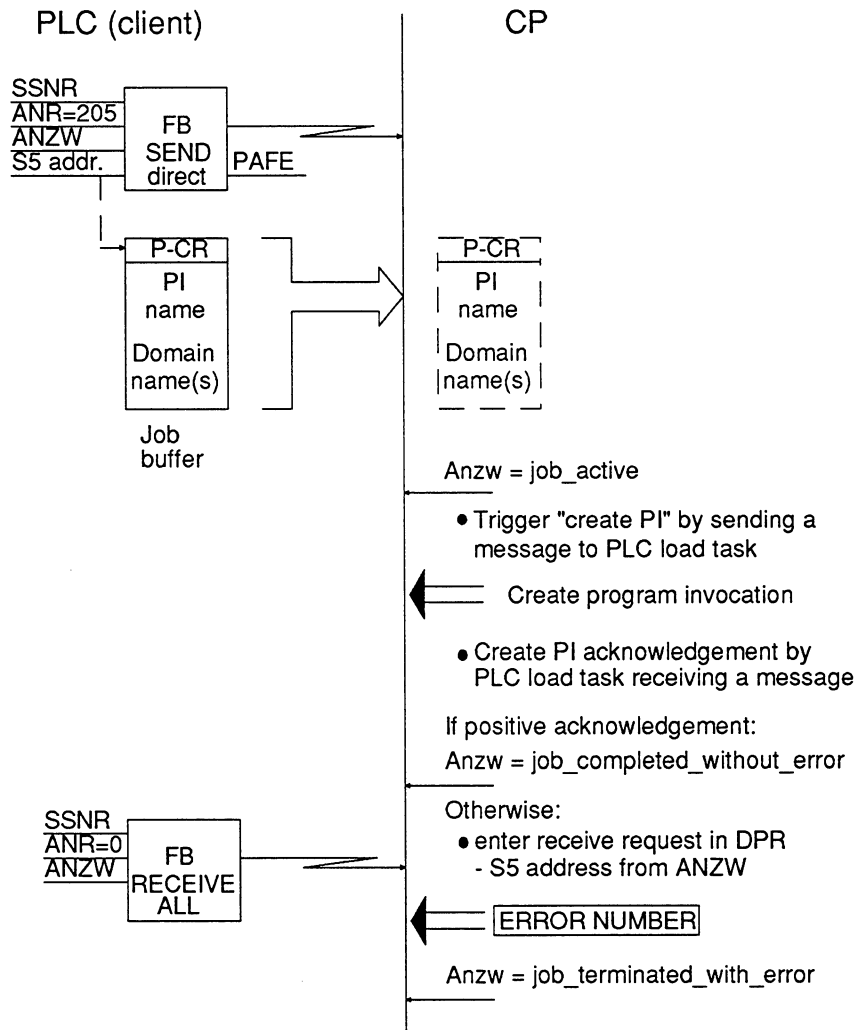


Fig. 5 - 23 Sequence "Create Program Invocation "



5

Fig. 5 - 24 Sequence "Create Program Invocation" on Local PLC

The PLC triggers the creation of a PI in itself by sending the job buffer with the VMD configuration job number (205).

5.2.6 Create Program Invocation (Server)

This section describes the handling of the service on the server and the conditions that must be met.

Preparations on the interface module:

- > Domains are loaded
- > No user PI exists (PI status = non-existent)

Sequence:

- > A data structure is set up to manage the PI. The PI status is stored in all four pages of the dual-port RAM.
- > The PLC is started as stipulated in the configuration ("swing cable", PG-MUX, dual-port RAM)
- > The PI status "idle" is entered (= 2)
- > Acknowledgement of the TF job

Note on loaded domains:

If no dynamic domains were loaded, i.e. the CP is not explicitly informed of domains, a so-called "static domain" is supported. In this case, the computer (or another client) must enter the following in the list of domains of the "create PI" TF service:

Number of domains:	1
Length of domain name:	10 (dec)
Domain name:	SIMATIC_S5

This domain name is also supplied in the reply in the "get name list" service if no other domain is loaded. It cannot be deleted with TF services

This allows a SIMATIC S5 PLC to be controlled using PI services without requiring the TF transfer services.

Note on domain statuses:

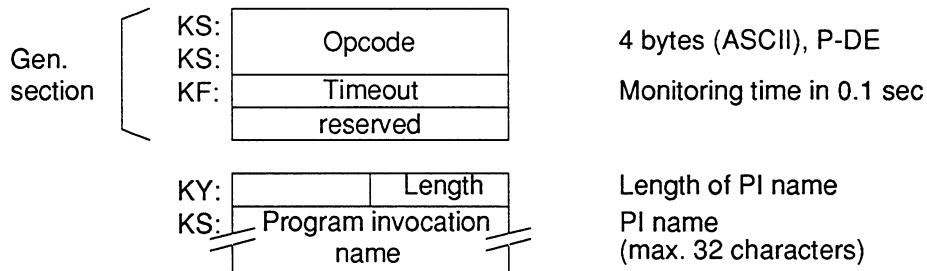
A domain specified with "create PI" changes to the "in use" status. It cannot be deleted in this status.

5.2.7 Delete Program Invocation (Client)

This service deletes a previously created PI in the local or in a remote station.

Since only one user PI can exist on a SIMATIC S5 PLC at any one time, the user PI in the SIMATIC S5 PLC must be deleted before a new PI with a different structure can be generated.

Job buffer "delete PI"



5

Fig. 5 - 25 Structure of the Job Buffer for TF Service "Delete Program Invocation"

Call description

General section:

Opcode P-DE

Timeout: 1 word, format: KF
 Meaning: specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.
 For more information about timeout, see page 3 - 9.

Job-related section:

Length of Program invocation name Format: KY
 Range of values: 1..32
 Meaning: length of the following PI name

Program invocation name Format: KS
 Meaning: Name of the PI to be deleted. If the length of the PI name is odd, a padding byte is appended.

Sequence description "delete PI"

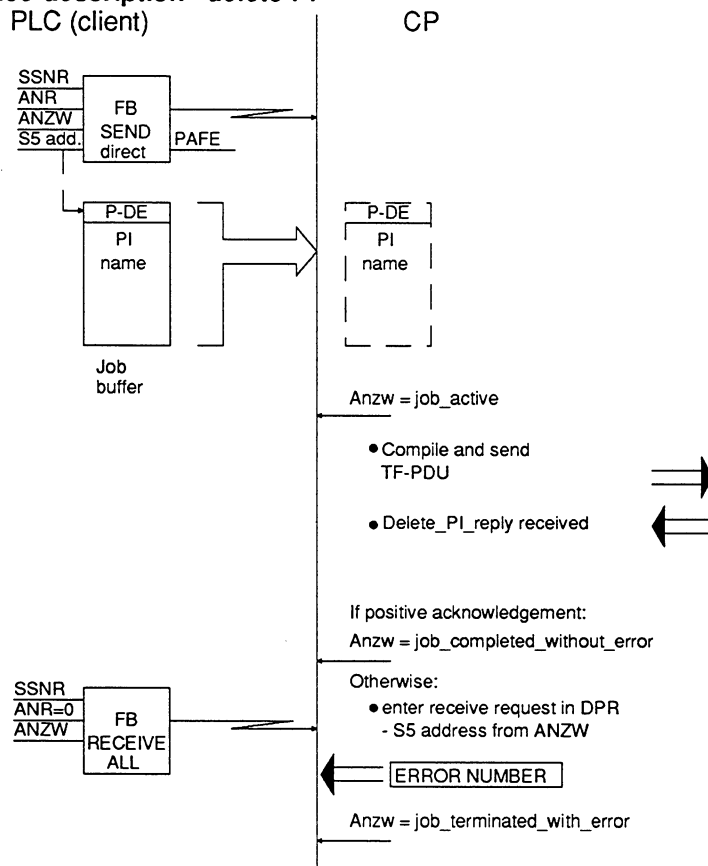
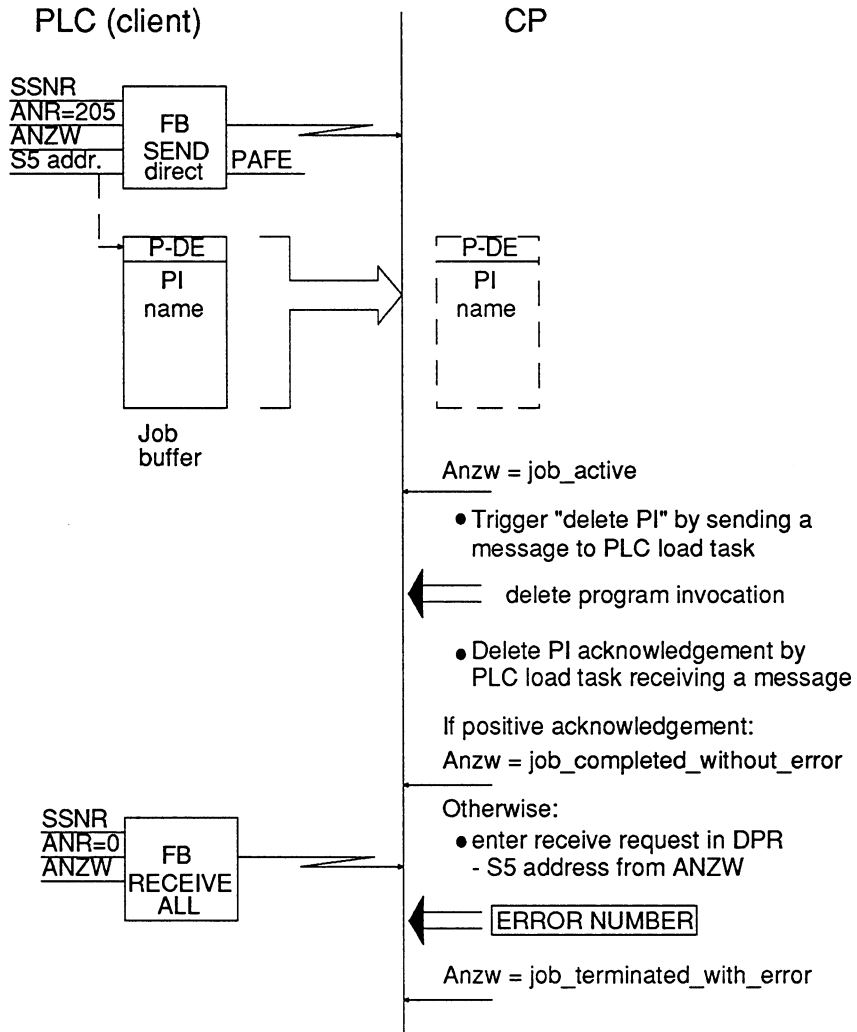


Fig. 5 - 26 Sequence 'Delete Program Invocation'



5

Fig. 5 - 27 Sequence 'Delete Program Invocation' (on Local PLC)

5.2.8 Delete Program Invocation (Server)

Effects on the user PI in the SIMATIC S5 PLC:

- The TF job is acknowledged by the PC providing the current status allows such a status change.

After this job, the PI management once again allows a create PI job for a user PI.

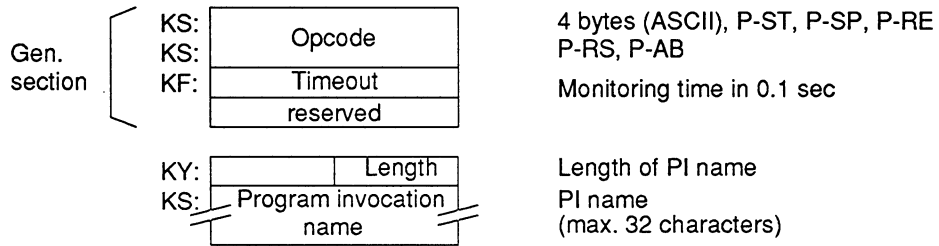
The domains used by the PI change from the "in use" status to the "ready" status; if required, they can also be deleted.

5.2.9 Start, Stop, Resume, Reset, Kill Program Invocation and Local Program Stop (Client)

Depending on the current status, the jobs cause status changes in the system or user PI.

Only the jobs resume PI and stop PI are permitted for the system PI.

Job buffer: "start_PI", "stop_PI", "resume_PI", "reset_PI", "kill_PI" and "local program stop"



5

Fig. 5 - 28 Structure of the Job Buffer for TF Services "Start PI", "Stop PI", "Resume PI", "Reset PI", "Kill PI" and "Local Program Stop".

Note: local program stop does not have a timeout parameter since it is only local.

Call description

General section:

- Opcode P-ST (start program)
- P-SP (stop program)
- P-RE (resume program)
- P-RS (reset program)
- P-AB (kill program)
- P-HL (local program stop)

Timeout: 1 word, Format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.
 For more information about timeout, see page 3 - 9.

Job-related section:

Length of Format: KY
PI name: Range of values: 1..32
 Meaning: length of the following PI name

PI name Format: KS
 Meaning: name of PI to be processed, if the length of the PI name is odd, a padding byte is appended.

Sequence description "start_PI", "stop_PI", "resume_PI", "reset_PI", "kill_PI"

The sequence for starting, stopping, resuming, resetting and killing a program invocation is the same for creating the PI.

The syntax ID execution argument and length execution argument parameters required for the "start PI" and resume PI" services are preassigned the value 0 by the CP.

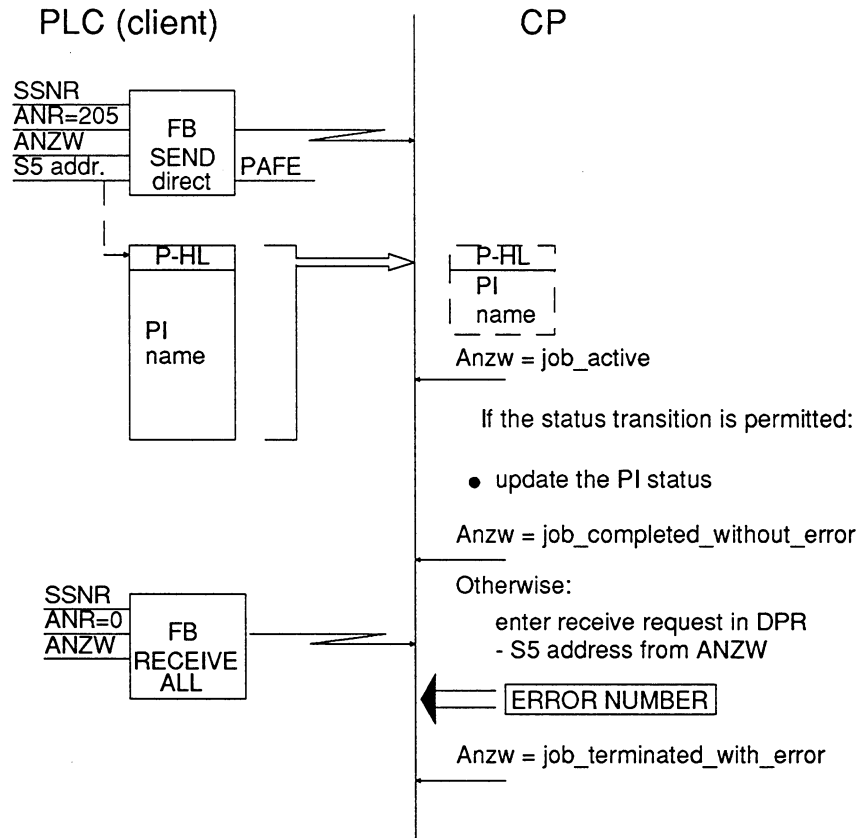
Sequence description "local program stop"

The user program can trigger the transition of the PI from running to stopped (see status diagram of a program invocation page 5- 33).

The transition takes place without a job from the network, i.e. without an explicit "stop PI" job from a client. The intermediate status "stopping" is skipped.

This can, for example, be useful when the user process must be retriggered by the host computer each time it has worked through its task (e.g. manufacturing a part).

The status transition is triggered by the user program sending a job buffer as a local job with the job number 205.



5

Fig. 5 - 29 Sequence "Local Program Stop"

5.2.10 Start, Stop, Resume, Reset, Kill a Program Invocation (Server)

The services are executed in the interface module as described in the Section "General Sequence of a Status Change", Section 5.2.2

Note the information about handling the user program in Section 5.2.3 'Interface of the PLC Program to the PI Services on the Server'.

With the following results as the current status, the user program must generate an acknowledgment:

- Starting
- Stopping
- Resuming
- Resetting

Based on this acknowledgment, status changes are entered into the status management. The actual effects in the PLC user program must be programmed.

5.2.11 Points to Note when Starting and Stopping the PLC using the System PI

The stop system PI service brings about a transition from RUN to STOP on the PLC. The resume system PI service brings about a transition from STOP to RUN on the PLC. The type of startup selected by the CP 143 is always a cold restart.

The "synchron" handling block call required to synchronize the CP with the CPU must only be called in the cold restart branch when the CP status "not synchronized" is indicated. You can find out about this status by calling FB "PI ZUSTD".

If you nevertheless call the synchron block although the PLC and CP are synchronized, a cold restart is executed on the CP which means that the MMS/TF service "Create PI" which triggered the cold restart can no longer be acknowledged.

Note on capabilities

To execute this service, the CP requires the "swing cable" (or backplane bus). If this is occupied (by a "PG on the bus") the service is acknowledged negatively. The PLC cannot be addressed by a PG on the bus while the service is being executed.

5.2.12 Get Program Invocation Attributes (Client)

This service allows the PI attributes such as PI status or domain assignment to be requested (both local and remote).

Job buffer "get PI attributes"

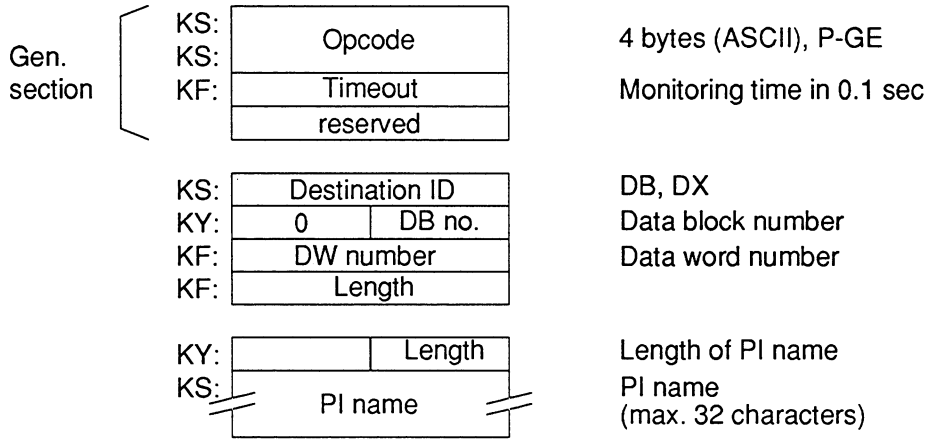


Fig. 5 - 30 Structure of the Job Buffer "Get PI Attributes"

Call description

General section:

Opcode P-GE

Timeout: 1 word, format: KF
 Meaning: specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.
 For more information about timeout, see page 3 - 9.

Job-related section:

Zielkennung:	Format: KS Range of values: DB, DX Meaning: S5 destination address at which the information about the attributes of the PI will be stored.
DB number:	Format: KY Range of values: high byte: 0, low byte: 1..255
DW number:	Format: KF Range of values: 0..2042
Length:	Format: KF Range of values: 1..2043, -1 Meaning: Length of the data block area in which the PI attributes can be stored; the value -1 means that all the data in the acknowledgement from the DW number to the end of the data block can be accepted.
Length of PI name:	Format: KY Range of values: high byte:0 low byte: 1..32 Meaning: length of the following PI name
PI name:	Format: KS Meaning: name of PI, if the length of the PI name is odd, a padding byte is appended.

Sequence description " get PI attributes"

The sequence of the "get PI attributes" service is analogous to the sequence of the "get domain attributes" TF service.

On the server, the service is executed automatically in the CP without support of the PLC program. The tables or data structures addressed are only in the address area of the CP and contain the domain and PI object attributes known in the CP.

Structure of the reply data

The reply data stored by the CP at the S5 address specified in the job buffer has the following structure:

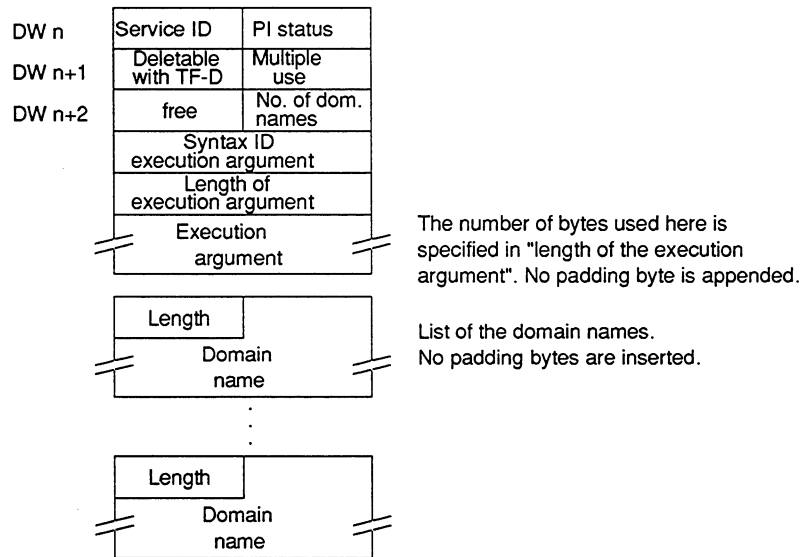


Fig. 5 - 31 Structure of the Job Buffer "Get PI Attributes"

Significance of the individual parameters in the reply data:

Service ID:

2Dh (to assign the reply to the requested service unequivocally)

PI status:

Value	MMS status	Status description
1	unrunnable	the program invocation was aborted or is in a status caused locally by an exceptional situation in which it can no longer be started.
2	idle:	the PI has been created
3	running	the user process has started
4	stopped	the user process was stopped
5	starting	the user process is in the start-up routine
6	stopping	the user process is in the stopping phase
7	resuming	the user process had stopped and is now starting up again.
8	resetting	the user process was stopped and is being reset so that it can then be deleted
-	non-existent	if no PI with this name exists a negative acknowledgment is returned with ERRCLS and ERRCOD.

Deletable with TF services

If this parameter = TRUE, it means that the program invocation can be deleted using the "delete program invocation" service. Otherwise this is not possible.

Multiple use

If this parameter = TRUE, it means that the program invocation can be started again following the reset carried out with the "reset program invocation" service. Otherwise this is not possible.

Number of domain names

Specifies the number of the domains belonging to the PI.

Execution argument:

the execution argument can be used to start an action on the server. The definition and execution are not specified by TF.

The following
applies to
SIMATIC S5:

- The "syntax_ID_execution_argument" has the value 252. This defines a visible string as the type for the execution argument.
- "Length of the execution argument" for SIMATIC applications is always 0.
- The execution argument itself is not used in SIMATIC S5;

Domain list

List of the domain names belonging to the program invocation.

5.2.13 Get Program Invocation Attributes (Server)

The service is executed on the server without support of the PLC program. The tables and data structures addressed are in the address area of the CP and include the domain and PI object attributes known in the CP.

□

6 Supplementary Services

This chapter provides you with the information required for handling the following services:

- application association management
- VMD services
- configuration jobs
- clock services

In a network with SIMATIC S5 PLCs, the CP relieves you of much of the handling of application associations during the operational phase. The CP establishes the application associations based on the configuration information during the start-up or as necessary. Information in this chapter about application association management is only required in exceptional cases when connecting two systems of other manufacturers.

Using the VMD services you can clarify the availability and capabilities of the communicating devices.

Configuration jobs are used to supply application associations with certain parameters during the operating phase independent of the configuration.

The clock function of the CP 143 is implemented by a clock chip and clock software which uses the clock chip. With this clock function, you can set and synchronize the time in the SINEC H1 network.

6.1 Application Association Management

Overview

The TF services for managing application associations provide the infrastructure required for communication on the application layer (layer 7) of the OSI reference model. The management of the corresponding transport connections (connections on layer 4) are also handled by the application link management. These services are executed automatically by the CP 143 (with the exception of the "abort application association" service based on the configuration information).

Service jobs for non-SIMATIC connections

Due to the implicit execution of the services for application association management, S5 applications do not normally require access via a program interface. The "application association management services" described in this section are only required when linking up with non-SIMATIC systems.

6.1.1 Definition of Application Associations

Application associations are defined in the COM 143 TF "definitions" screen and stored in data link blocks in the memory module of the CP.

The global and local parameters of the data link block are essential for link management. (See the introduction in Volume 1, Chapter 2 of this manual).

Global parameters:

definition of the layer 4 connection with the parameters

- local TSAP
- remote TSAP
- remote Ethernet address (MAC address)

Local parameters

Definition of the application association (layer 7 link) and modelling of the virtual circuit on the SIMATIC S5 PLC

- Multiplex address
- application association name
- interface number
- job number
- status word (can be modified later using a configuration job)
-> see also the REQUEST-EDITOR tool)
- type of connection establishment
- number of jobs (or number of application associations)

Number of jobs

The parameter "number of jobs" specifies how many application associations can use the same layer 4 connection. These application associations are distinguished based on the multiplex address.

Type of connection establishment

Using the "type of connection establishment" the system planner can decide whether the connection is established only as far as layer 4 or up to layer 7. The conditions for establishment are also stipulated, i.e. how and when the connection should be established

In concrete terms this means that if a link is required to a system in which application association management functions are not implemented, the connection is established only as far as layer 4. All communications partners must be capable of this connection establishment. This is, for example, the case when a link is required between the CP 143 and CP 535 with the AP protocol handler SINEC AV/S5.

Normally, establishment up to layer 7 should be selected.

The three following types of establishment must be distinguished:

static active

The CP establishes a permanent connection during start up.

static passive

The CP is ready to positively confirm a connection request for the configured link.

dynamic

The CP establishes a connection as soon as there is a job for the configured link.

The recommended type of connection establishment for various services is listed in the following table:

Connection establishment for TF services

Service	Type of establishment
Non-open services (transparent data exchange, read/write byte string)	Layer 4
Variable services	Layer 7
Domain services	Layer 7
PI services	Layer 7

6.1.2 Connection Establishment

The connection is established implicitly by the CP based on the configuration information.

The establishment of connections is only completely transparent for the PLC program. The connection establishment is triggered by the CP and not by an application program on the PLC. Only the dynamic links (priority 3) are triggered indirectly by the PLC when a TF job is triggered. Nevertheless, even in this case there is no specific job for triggering connection establishment. This is always in conjunction with a client job (not configuration jobs).

During the start-up on the module, the establishment of all priority 2 links (establishment "ACTIVE" or "PASSIVE") is initiated.

Layer 4

The establishment of the transport connection is made by the transport software using the appropriate layer calls (e.g. open, connection request).

CP143 transport software: iNA960 R 1.1

6

Layer 7

The establishment of the layer 7 link is only possible when the layer 4 connection establishment was successful.

Since the "initiate application association" service contains specific implementation parameters, these are explained in detail. These parameters are sent to the partner when the connection is established. They are fixed by the CP and cannot be modified by programming or configuring (exception max_receive_buffer_calling)!

Service ID
F0H

Local_expansion_valid
always "FALSE"

Max_AmQ_calling_proposition
0FFH, i.e. any number of server jobs per application association can be managed in the CP at one time.

Max_AmQ_called_proposition
1H, i.e. the remote VMD must expect a maximum of one server job on an application association at one time.

Max_receive_buffer_calling

Value from the INIT block of the module card.
The maximum length of a TF-PDU can be selected by the user in COM 143 for all application associations when the module is initialized.

Non_open_services_calling 0000 0000 0000 0111B i.e. the following non-open TF services are supported:

- read byte string
- write byte string
- transparent data exchange

Nesting_level_proposition
2H, i.e. the CP supports variables up to the following structure:

- arrays of arrays of basic data types
- arrays of structures of basic data types
- structures of structures of basic data types

Number_syntax
1H exactly one syntax field follows

Syntax_ID
0H (assigned by TF)

Length_abstract_syntax
5H

Connection Establishment (continued)

Abstract_syntax
SINEC_TF_CORE_VERSION_1
Codierung: 28 CA 22 02 01 (hex.)

Length_Transfer_Syntax
0FH

Transfer_Syntax
SINEC_TF_CODING

Version number_proposition
1H

Parameter_CBB_proposition
0000 0000 0101 1111

:

- Data type "ARRAY" is supported
- Data type "STRUCTURE" is supported
- Access to variables via a name is supported
- Access to variables via an address is supported
- Third-party association for loading domain is supported

Supported_TF_services_calling:

All TF services of level 1 are supported (see appendix "CP 143 Product Data Sheet").

If a third-party association is involved, only the necessary services are required.

These values are proposed in the "initiate application association" TF-PDU. When the CPU sends the acknowledgement, it negotiates that the partner settles for these values.



The specific implementation parameters are supplied by the CP 143 and do not need to be set by the user.

6.1.3 Connection Termination

A TF application association is terminated by the "conclude application association" service that requires no parameters other than the service identifier.

If, following the termination of the layer 7 link, there are no further layer 7 links set up on the layer 4 connection, the layer 4 connection is also terminated.

The PLC program can trigger connection termination explicitly with a reset call (handling block) with the corresponding job numbers. This can, for example, be necessary when an TF service was triggered without timeout monitoring and was not completed.

With priority 3 jobs, the link remains terminated until it is triggered again by the PLC program. Priority 2 connections are re-established immediately by the CP.

A "reset all job" (job number 0) to terminate all connections is not permitted.

6.1.4 Special Connections

Application associations to a fileserver

To load or store domains, the "third-party association" is supported by the CP 143. This means that after a host computer has triggered a load procedure (load domain content, store domain content) the CP itself establishes the connection identified by the "application association name" parameter.

Up to 16 different fileserver application associations can be defined, of which a maximum of one can be established at any time.

The type of connection establishment for fileserver application associations is always "dynamic". Variable definitions are not possible. The assignment of the connections to the end system is not specified explicitly for these associations, but is transferred by the user in the capabilities list when the load function is triggered.

Multiplex addresses are not supported on these connections.

Predefined connection

The communications processor provides a non-configurable standard connection for loading services and for program invocation services, for which the following parameters are selected:

local TSAP: S5_STF (length 6)
remote TSAP: S5_STF (length 6)
remote Ethernet address: unspecified
MUX address: 0

6.2 VMD Services for Virtual Manufacturing Devices

The general TF services for virtual manufacturing devices allow a client to request information about the status or attributes of the virtual manufacturing device (VMD) in the server. The server can also indicate the status to a client without being requested. The information can then be further processed at the client, for example to provide a supervisory control center with an overview of the whole status of the plant.

The following services are available:

- Status of a VMD
- Unsolicited status of a VMD
- Identify VMD

6.2.1 Status of the Virtual Device (Client)

Using the "status of the virtual device" service, a client requests information about the physical and logical status of the virtual manufacturing device managed in the server. The server sends the requested information in the acknowledgement (e.g. whether the "real" manufacturing device or the communications processor of the server is in the RUN or STOP mode or whether the PLC and CP are synchronized or not).

Job buffer "VMD-Status"

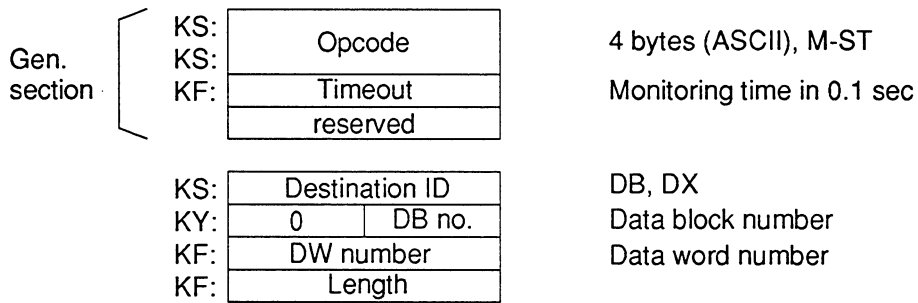


Fig. 6 - 1 Structure of the Job Buffer "VMD Status"

6

Call description

General section

Opcode M-ST

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.

For further information about timeout see page 3 - 9.

6.2.2 Status of the Virtual Device (Server)

The service is executed by the communications processor without the support of the PLC. However, the status of the PLC is also included in the data of the reply.

Structure of the reply data

The reply data stored by the CP at the STEP 5 address specified in the job buffer has the following structure:

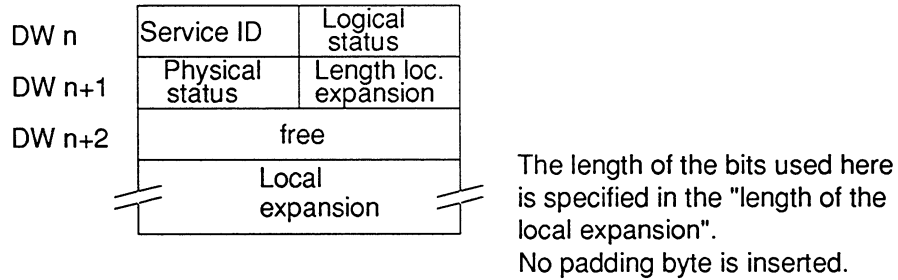


Fig. 6 - 2 Structure of the Reply Data in the Data Block

Service ID:

0h (to assign the reply to the required service unequivocally)

The range of values of the other parameters is described below.

VMD Status Indication:

1. The AS 511 master connector is plugged in or PG functions are possible via the dual-port RAM (with swing cable)

PLC status/ CP status	PLC in RUN	PLC in STOP	
CP in RUN	state changes allowed	state changes allowed	logical status
	operational	partially operational	physical status
CP in STOP	limited services permitted	limited services permitted	logical status
	needs commissioning	needs commissioning	physical status
CP/PLC not synchronized	limited services permitted	state changes allowed	logical status
	partially operational	needs commissioning	physical status

VMD Status (continued)

2. No AS 511 master connector plugged in and no PG functions via the dual-port RAM (swing cable)

PLC status/ CP status	PLC in RUN	PLC in STOP	
CP in RUN	limited services permitted	support services permitted	logical status
	operational	partially operational	physical status
CP in STOP	limited services permitted	limited services permitted	logical status
	needs commissioning	needs commissioning	physical status
CP/PLC not synchronized	limited services permitted	limited services permitted	logical status
	inoperable	inoperable	physical status

6

Explanations:

1. PLC in RUN, CP in RUN, master connector:
all TF services permitted, fully functional
2. PLC in RUN, CP in RUN, no master connector:
no domain or PI services possible
3. PLC in Stop, CP in RUN, master connector:
only domain and PI services and information services (status, name list etc.) permitted, no variable services, no non-open services
4. PLC in Stop, CP in RUN, no master connector:
only information services allowed

5. CP in stop:
only status and identify VMD permitted (regardless of PLC mode and the configuration with the master connector). The CP must first be switched to the run mode or at least brought to the "CP/PLC not synchronized" status before other TF services are permitted.
6. PLC in stop, CP/PLC not synchronized master connector:
only domain services and the services "create PI"/"delete PI" and information services (status, name list etc.) permitted, no variable services, no non-open services.
7. PLC in stop, CP/PLC not synchronized no master connector:
only information services permitted.
8. PLC in run, CP/PLC not synchronized, master connector:
only domain services and the services "create PI"/"delete PI" and information services (status, name list etc.) permitted, no variable services, no non-open services.
9. PLC in run, CP/PLC not synchronized:
no master connector only information services permitted.

Coding of the parameters "logical status" and "physical status":

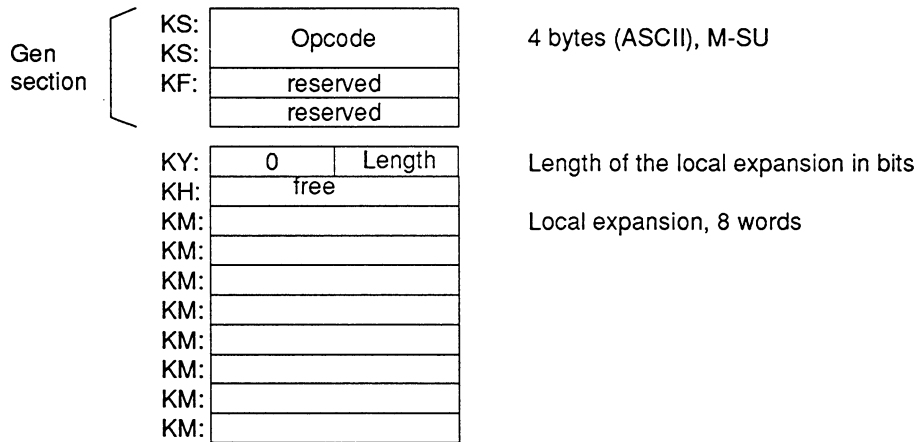
logical status:	state changes allowed	00h
	no state changes allowed	01h
	limited services permitted	02h
	supported services permitted	03h
physical status:	operational	10h
	partially operational	11h
	inoperable	12h
	needs commissioning	13h

6.2.3 Unsolicited VMD Status (Initiator)

Spontaneous indication of the status of the local VMD to another partner.

Job buffer "unsolicited VMD status"

Call description



6

Fig. 6.3 Structure of the Job Buffer "Unsolicited Status"

General section:

Opcode M-SU

Job-related section:

Length of local expansion Format: KY (low byte)
 Values: 0..128
 Meaning: the parameter length_of_local_expansion specifies the number of left-justified bits in the "local expansion" parameter.

Local Format: KM (8 words)
expansion: Meaning: the significance of the local expansion is fixed
 depending on the particular application.

The TF parameters "logical status" and "physical status" are inserted by the CP itself. The table of statuses is described in the previous section.

Sequence description

The sequence of the "unsolicited VMD status" service is analogous to the sequence of the "information report" TF variable service.

6.2.4 Unsolicited VMD Status (Receiver)

To allow the frame to be processed correctly on the receiver, the CP must be informed of the location of the received data with a configuration job.

6.2.5 Identify Virtual Manufacturing Device (Client)

With this service, a client can request information about the attributes of the virtual manufacturing device (VMD) from a server.

Job buffer "identify virtual device "

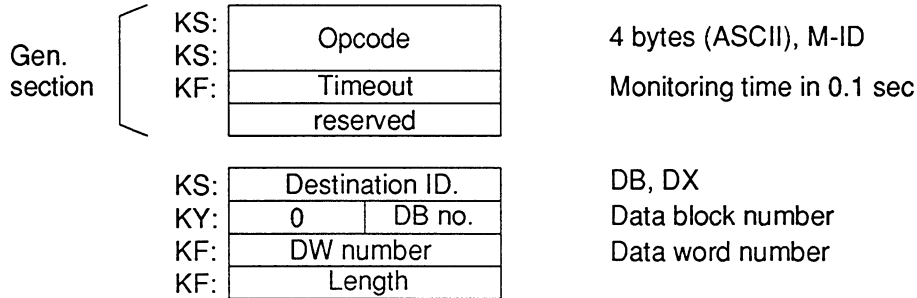


Fig. 6.4 Structure of the Job Buffer " Identify"

Call description

6

General section:

Opcode M-ID

Timeout: 1 word, format: KF
 Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.
 For further information about timeout see page 3 - 9.

Job-related section:

Dest. ID: Format: KS
 Values: DB, DX
 Meaning: address at which the information about the identity of the remote station will be stored.

Identify (continued)

DB number: Format: KY
 Values: high byte: 0, low byte: 1..255

DW number: Format: KF
 Values: 0..2042

Length: Format: KF
 Values: 1..2043, -1
 Meaning: length of the data block area in which the
 information contained in the acknowledgement
 can be stored; the value -1 means that all the
 data in the acknowledgement from the DW
 number to the end of the data block can be
 accepted.

Sequence description

The sequence of the "identify virtual device" service is analogous to the sequence of the "read" TF variable service.

The service cannot be used on the local PLC.

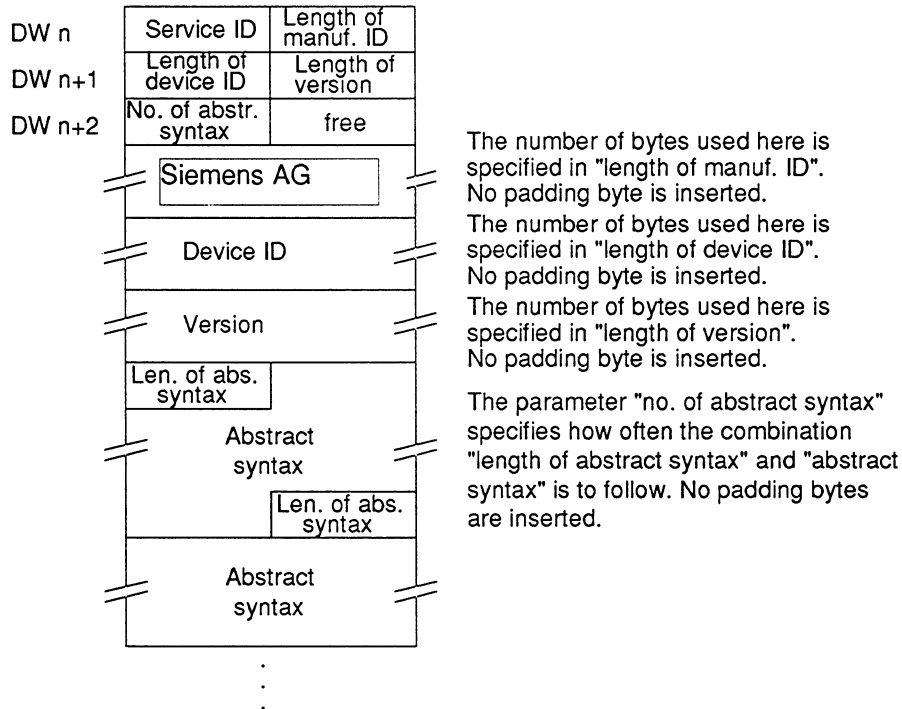
Identify (continued)

6.2.6 Identify VMD (Server)

The service is executed by the CP without support of the PLC. However, the attributes of the PLC are contained in the data of the reply.

Structure of the reply data

The reply data stored by the CP at the S5 address specified in the job buffer has the following structure:



6

Fig. 6.5 Structure of the Reply Data in the Data Block

Service ID:

2h (to assign the reply to the requested service unequivocally)

Length_of_manufacturer's_identifier:

length of the manufacturer's identifier contained in the reply data

Length_of_the_device_identifier:
length of the device identifier contained in the reply data

Length_of_the_version:
length of the version contained in the reply data

Number_of_abstract_syntax:
number of elements in the syntax list of the reply data

Syntax_list:
in the syntax list, the server enters the number of syntaxes it supports

If the remote partner is a SIMATIC PLC, the following values are contained in the reply.

Length of manufacturer's ID:	0BH
Manufacturer's ID:	SIEMENS AG
Length of device ID:	0FH
Device ID:	6GK1 143 0Ax00
Length of version:	12H
Version:	V x,y - - - / - <date>

The version consists of 18 characters (including blanks)

x,y; current version ID of the CP 143 firmware

The parameter <date> contains the data the module file was created specified by the user when entering the SYSID

Number of syntax: 0H

Protocol information:

The CP does not support any CS_parameters (Companion Standards) that may occur with the TF services (variable services, general services, domain and program services).

6.3 Configuration Jobs

Purpose

Using the configuration jobs, an S5 program can assign certain parameters to an application association. This means that the parameters do not need to be programmed, but can be passed on to the CP while it is running. The configuration only ever applies to the link specified by the HDB call parameter "SSNR/ANR". The PLC program triggers configuration jobs once again using job buffers. These can be transferred to the CP at any time after start up.

Special note:

If the status of the job (can be ascertained by calling the control HDB) is "job terminated with error" with the error number 0, this means that the connection has been re-established. In this case previously transferred configuration parameters are invalid.



Configuration parameters are valid as long as the connection is established. The parameters transferred with a configuration job always have priority over programmed parameters.

6

The following connection-specific parameters can be configured by the PLC program:

1. Status word for PLC client jobs
For client jobs (odd ANR) the communications processor requires information about the status word assigned by the user, also for calling the SEND-DIR to trigger the job. To inform the CP, the parameter type AN (status word for PLC client jobs) is used.
Note: the status word can also be programmed.
2. Source address for server job "read byte string"
To be able to execute an incoming "read byte string" job, the CP must know the S5 address of the required byte string.
Parameter type: BL

3. Destination address for server jobs "write byte string"
To be able to execute an incoming "write byte string" job, the CP must know the S5 address of the byte string to be written.
Parameter type: BS

4. Destination address for server jobs "unsolicited VMD status indication"
To be able to process an "unsolicited VMD status indication" job in the CP that was received without being triggered by the PLC, the PLC program must have an S5 destination address available at which the data contained in the job are stored. The structure of the data stored by the CP in the PLC is the same as for the reply data of the "VMD status" service. The service identifier is 1H.
Parameter type: MS

Job buffer "configure ANZW [local]"

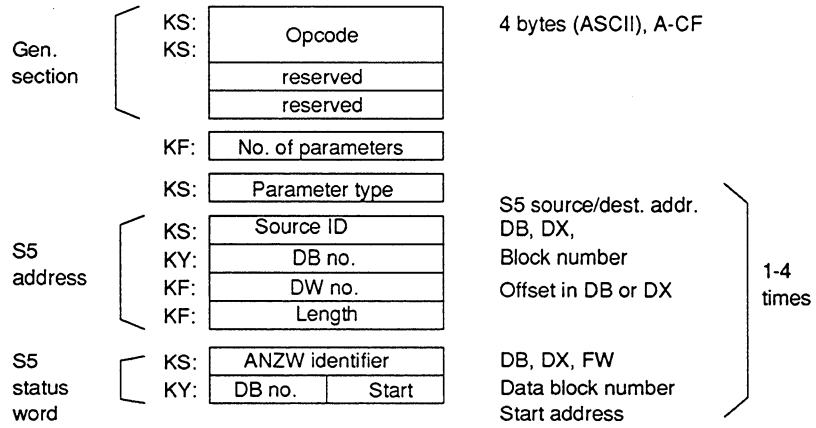


Fig. 6.7 Structure of the Job Buffer "Configure ANZW (local)"

Sequence description (configure ANZW [local])

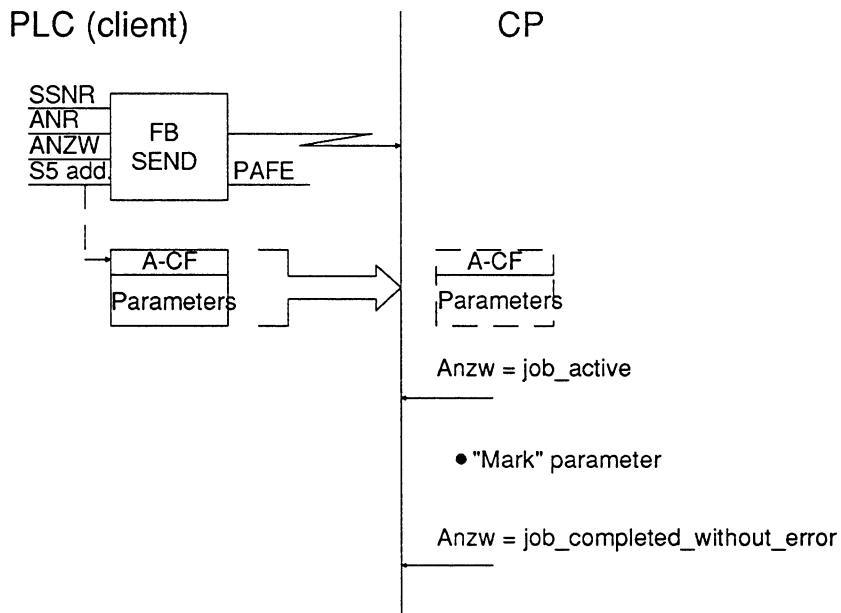


Fig. 6.6 Sequence: "Configure ANZW (local)"

Status word

ANZW Format: KS
identifier: Values: FW, DB, DX
 Meaning: status word type

ANZW Format: KY
specification Values: high byte: block number
 low byte: DW number, FW number

6

6.4 Clock Services

The CP 143 clock function is implemented by a clock chip and clock software that uses the clock chip.

There are three basic clock functions:

1. The clock keeps the time on the CP 143 within the absolute limits of accuracy described in the technical data. This clock continues to run during a power down as long as the battery voltage is present.
2. The clock can also be used to send synchronization frames so that all the CP 143s connected to the SINEC H1 network and which participate in synchronization have a relative deviation of 20 ms from each other. The transmitter of the clock message can be either a CP 143 or the SINEC time transmitter.
3. The CP can function as clock master and transmit synchronization frames.

The clock frame has a fixed format for SINEC which corresponds to the TF standard. The transmitter uses a special multicast set (Ethernet address 09 00 06 01 FF EF) for transmitting the time message. The user does not need to assign parameters to this, since the clock software executes this task.

The time is supplied to the PLC in the S5-155U data format (see page 6-40).

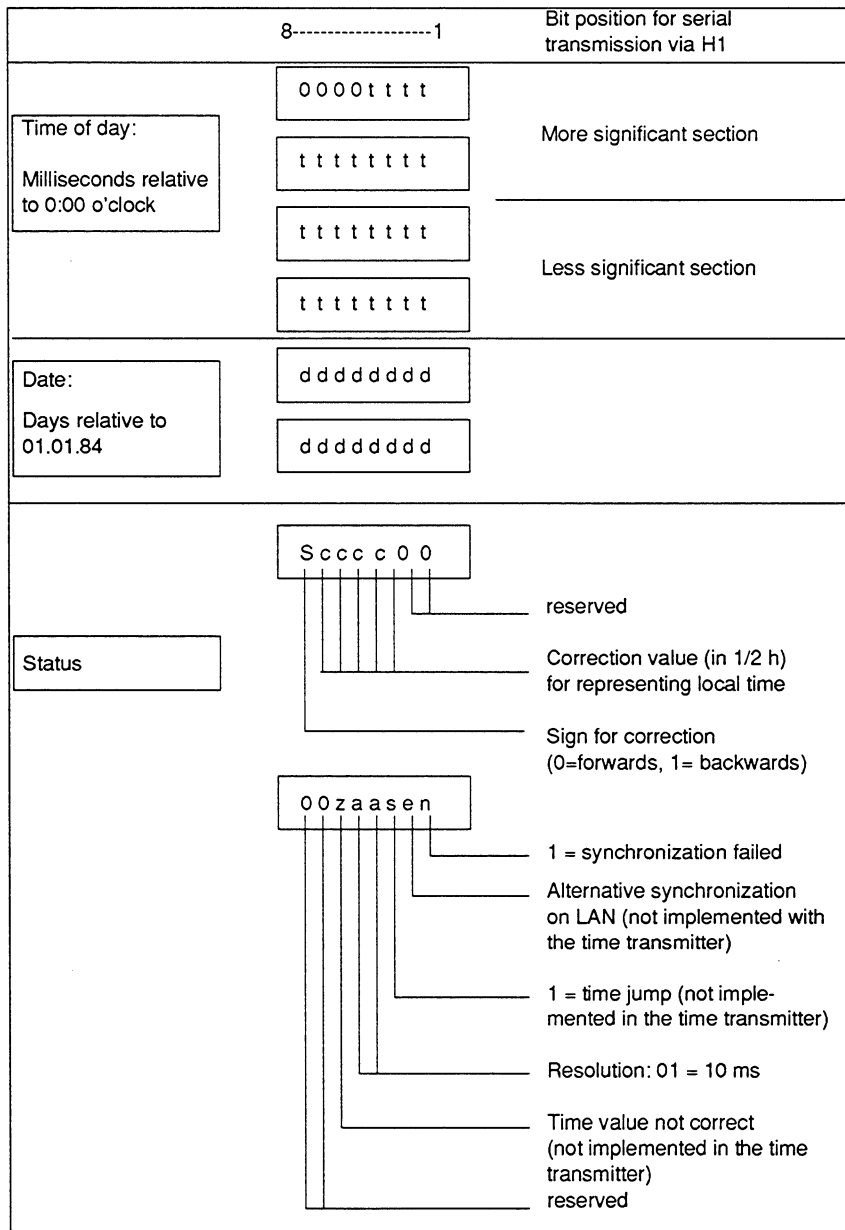
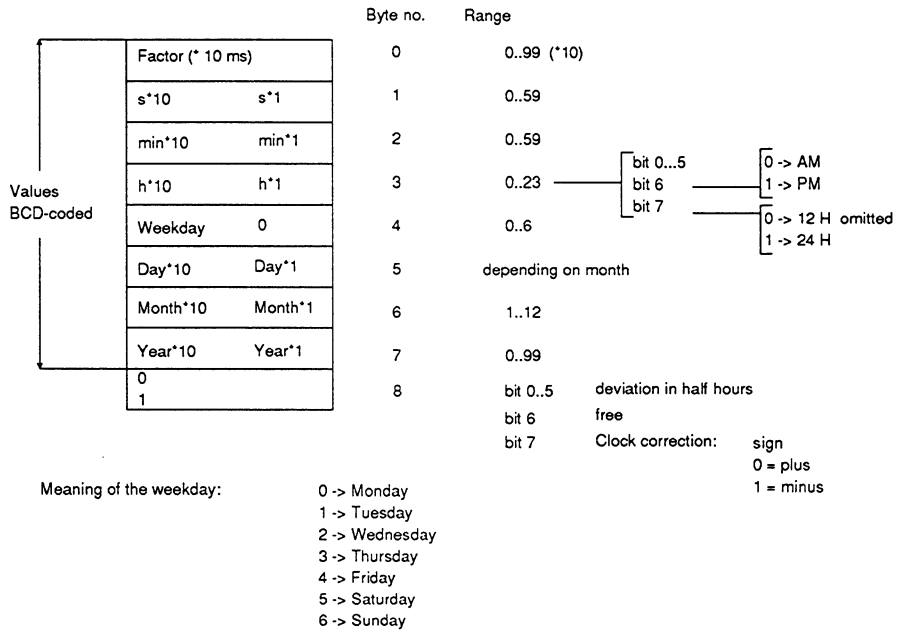


Fig. 6.8 Time and Status Representation



This format corresponds to the S5-155format (see page 6-40).

Byte 8 is an extension derived from the MMS time representation. This represents a flexible time deviation from the standard time.

6.4.1 Network Topology, Clock Master/Slave Functions

Within a SINEC H1 network, all the 143s can execute clock functions. The aim is to achieve network-wide clock synchronization.

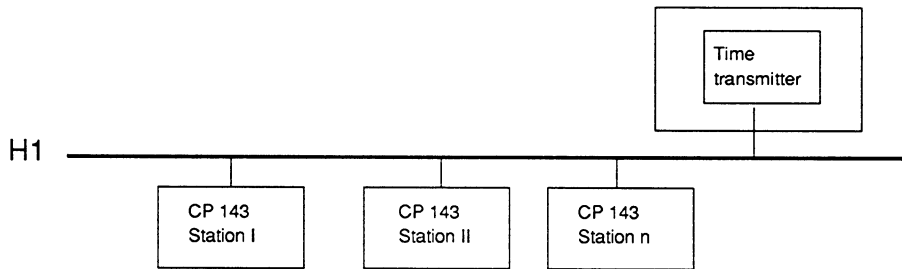


Fig. 6.9 Network Topology

The synchronization can be performed by a SINEC time transmitter or by **one** selected CP 143.

The station that transmits the clock synchronization frames is known as the "clock master".

In this case, all other stations are "clock slaves".

If the synchronization is handled by a SINEC time transmitter, all the CP 143s are clock slaves.

M 2-2

If there is no SINEC time transmitter in the network, a CP 143 can be programmed to take over the clock master functions (dynamic clock master = yes in the "CP Initialization Block" screen).

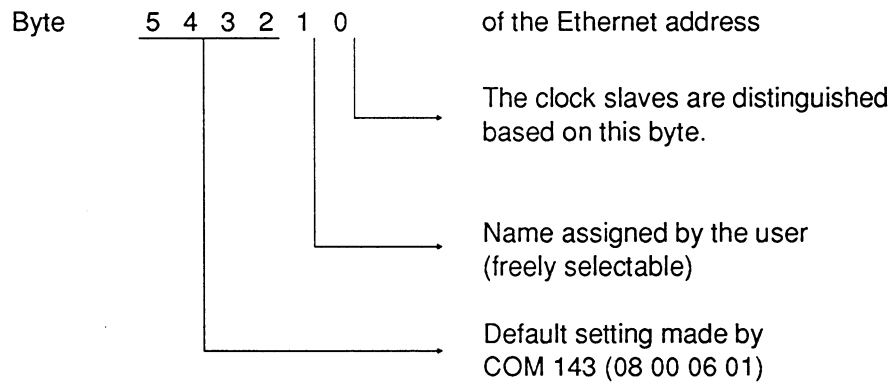
The time transmitter has time transmission intervals of 1 sec, 10 sec and 60 sec. If the clock messages are transmitted without the clock transmitter, these transmission intervals should be retained. The default value supplied by COM 143 is 10 sec. This means that all the slaves expect a synchronization frame from the clock master after a maximum of 10 sec. Otherwise, the clock slaves attempt to take over the clock master function (this is only achieved by the highest priority clock slave station, provided this is configured as the DYNAMIC CLOCK MASTER in COM 143). The other CPs then take their time from the highest priority CP.

Priority can be assigned to the stations for taking over the clock master function by assigning the Ethernet address as described below.

The Ethernet address is defined as follows:

Byte	5	4	3	2	1	0	Ethernet address (6 bytes) (default values supplied by COM 143)
	08	00	06	01	XX	XX	

All six bytes of the Ethernet address can be freely selected. To achieve a clearly defined priority, the user must keep to the following Ethernet address assignment:



Further explanations:

- a) If the first four bytes of the Ethernet address differ from the default value, the module will not attempt to take over the clock master function.
- b) The CP 143 can only take over the master function when byte 0 of the Ethernet address is within the following limits and at the same time "DYNAMIC CLOCK MASTER" (Y/N)? was answered with "Y" in COM 143.
- c) Based on byte 0 of the Ethernet address, a time is stipulated after which (if no clock message has been received) the station attempts to become clock master.

The following terms are important:

- Delay Time
corresponds to byte 0 of the Ethernet address in seconds.
- Update Time
selected time interval for transmitting clock synchronization frames.
- Undefined Time
sum of the delay time and update time

By evaluating byte 0 of the Ethernet address as the delay time, a priority is established for the attempt to take over the clock master functions.

Example:

the following table shows which station takes over the clock master function and if this fails, which station will replace it.

	Status	Dyn. master	Delay time	
Can be master	Master	Y	03	Assigning the priority based on the delay time
	Slave	Y	07	
	Slave	Y	08	
	Slave	Y	10	
	Slave	Y	12	
	Slave	Y	13	
Cannot be master	Slave	N	18	
	Slave	N	21	
	Slave	N	01	
	etc.		!=!	



The delay time must be different for each station. If this is not the case, the stations with the same delay time will never attempt to take over the clock master functions.

This concept ensures that there is always clock synchronization within the network.

6.4.2 Accuracy

The hardware clock of the CP 143 has a maximum deviation of 11.94 s/day or 8.3 ms/min. This deviation is based on a calculation involving the quartz inaccuracy and temperature fluctuation.

To perform accuracy calculations, twice the clock deviation must be assumed, since a clock can be both fast and slow. This results in a time of 23.88 s/day or 16.6 ms/min.

For this reason, it is necessary to compensate for this deviation in the CP 143 hardware clock by receiving synchronization messages.

The highest accuracy can be achieved by using the SINEC time transmitter for transmitting synchronization frames.

The time is kept on the hardware clock of the CP 143 with a resolution of 10 ms.

To achieve a system-wide clock accuracy in the programmable controllers, a time difference of 20 ms must not be exceeded. This is achieved by clock synchronization.

6.4.3 How the Clock Functions

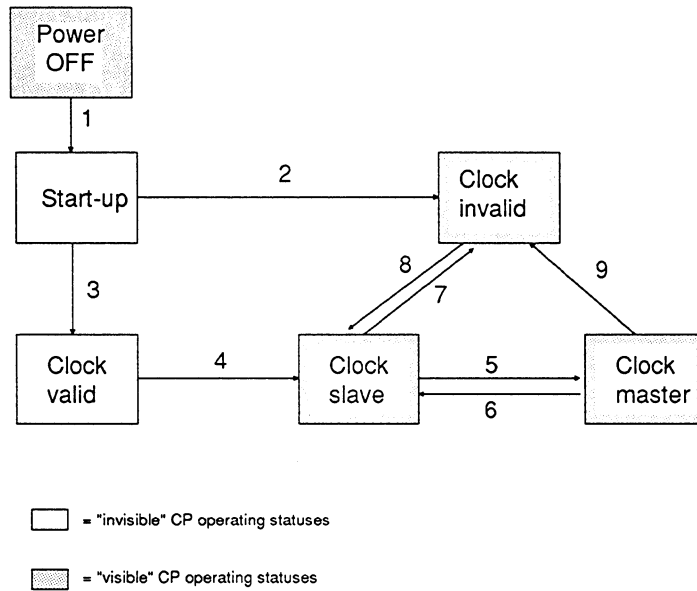


Fig. 6.10 Operating Statuses of the Clock

The clock can have the following statuses:

Description of the status transitions:

1. While the CP is starting up, the hardware clock of the CP 143 is checked.
2. The status of the hardware clock was recognized as invalid. The clock must be reset.
3. The status of the hardware clock was recognized as valid, i.e. the hardware clock has already been set.
4. If during the CP start-up the clock is recognized as valid, the CP automatically assumes the "clock_slave" status.

5. During the undefined time, no synchronization frame was received. The CP therefore attempts to take over the clock master function.
6. The current clock master has received a synchronization frame from a SINEC time transmitter or a higher priority CP 143. The station once again assumes the status of clock slave.
7. The CP with the clock slave status recognizes an invalid time (e.g. defect hardware clock).
8. This status change is possible after receiving a valid clock message from the clock master from the PG or from the PLC.
9. The CP with the clock master status recognizes an invalid time (e.g. defect hardware clock).

6.4.4 Several CP 143 TFs on One SINEC H1 Bus without a SINEC Time Transmitter

With a bus structure, dynamic clock masters can be configured. Byte 0 of the Ethernet address determines which CP 143 executes the clock master function. A double definition is not possible.

ff 2-2

Configuring the clock

The clock is configured in the screen for configuration of the initialization block of the CP. The values entered in the screen correspond to the defaults.

DYNAMIC CLOCK MASTER (Y/N): Y
The CP 143 can become the clock master if it has the highest priority and there is no time transmitter in the SINEC H1 network.

N
The CP 143 receives synchronization frames if they exist in the H1 network.

CYCLE TIMES FOR SYNC FRAMES: 10 (sec) default
If the CP 143 is the clock master, it sends clock synchronization frames to the SINEC H1 network at the time intervals specified above.

Possible values: 1 sec, 10 sec or 60 sec.

ETHERNET ADDRESS: 080006010000

The Ethernet address has a double significance for the CP 143 clock function.

1. Only CPs with the address 08000601 at the start can become clock masters.

2. The last byte (ID) of the address decides which CP will become clock master. Only the CP whose ID is the lowest on the SINEC H1 network will become clock master, i.e. the CP with the highest priority.

6.4.5 CP 143 TF on a SINEC H1 Bus with SINEC Time Transmitter

The SINEC time transmitter has the highest priority as clock master. If a SINEC time transmitter is integrated in an existing bus structure, the CP 143s connected to the SINEC H1 bus that receive a clock message from the SINEC time transmitter assume the status of clock slaves.

If a SINEC time transmitter is taken out of the existing bus structure, the CP 143 with the lowest delay time (byte 0 of the Ethernet address) assumes the clock master function.

6.4.6 Setting and Reading the Time in the Programmable Logic Controller

The SIMATIC S5 CP has the job number 218 available for processing the time.

A SEND with this job number writes the CP's time, a RECEIVE, reads the CP's time.

These services are possible on the synchronized CP interfaces using the appropriate standard HDBs for the PLC.

Data format of the time in a PLC data block (S5-155 U-format)

	15	121	87	43	0
DW n:	tens sec	units sec	1/10 sec	1/100s sec	
DW n+1:	tens hour	units hour	tens min	units min	
DW n+2:	tens day	units day	weekday	0	
DW n+3:	tens year	units year	tens mon	units mon	
DW n+4:			↑ Sign of the time difference	time adjustment	

Setting and Getting the Time from the PLC (continued)

Possible values (hexadecimal):

100ths	seconds:	irrelevant for "set time"
tenths	seconds	irrelevant for "set time"
units	seconds	0...9
tens	seconds	0...5
units	minutes	0...9
tens	minutes	0...5
units	hours	0...9
tens	hours	0...1 / 0...2
		Bit 15 = 1: 24 hour format
		Bit 15 = 0: 12 hour format
		Bit 14 = 0: AM
		Bit 15 = 1: PM

Weekday	Mon...Sun = 0...6
---------	-------------------

units	day	0...9
tens	day	0...3
units	month	0...9
tens	month	0...1
units	year	0...9
tens	year	0...9

6

Time adjustment: +/- 0 to 24 in 1/2 hours.

Sign of the time adjustment: 0=positive; 1=negative

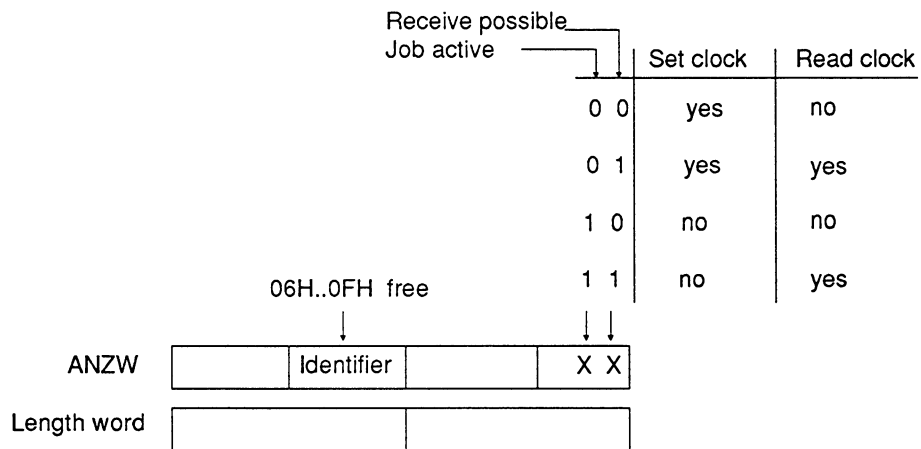
The following identifiers are possible replies to a "set time" job from the PLC.

Reply (decoded IDs)	ID	Meaning
OK, no error	00H	Command could be executed without errors
Protocol error	01H	Time is invalid (was not set etc.)
System error	0EH	System error (e.g. invalid command)
Hardware clock	0FH	Hardware clock failure

The following identifiers are possible as the reply to a "read time" job of the PLC.

Reply	ID	Meaning
System error	0EH	System error (e.g. invalid command)
Hardware clock	0FH	Hardware clock failure
Clock_master	06H	CP is clock master and is operating as master
Clock_slave	07H	CP is clock slave
Clock_slave, + invalid	08H	Station has an invalid clock chip
Clock_slave, + asynchronous	09H	Station does not receive a clock message
Slave, >master	0AH	CP is clock slave; prepare for master function
Master,>slave		CP is clock master; prepare for slave function
Transmitter, synchronous	0BH	Transmitter itself does not receive synchronization frames
Replacement synchronization	0CH 0DH	Replacement synchronization free

Identifiers of the status word of the handling blocks (HDBs)



When the CP 143 is starting up, the lower two bits of the status word are set to "set clock" and "read clock not possible". During normal operation, these bits are set according to the CP clock status.

6.4.7 Setting and Reading the Time with COM 143

Using COM 143, it is possible to both set the hardware clock of the CP 143 as well as to read the current time (including cyclically).

COM 143 TF provides the CLOCK FUNCTIONS screen. When this is called, a read time message is sent to the currently selected CP 143.

The screen form is filled with the received data and the possible functions available with the softkeys correspond to the CP clock status.

When reading the time, an identifier byte is also sent to the PG protocol, the value of which provides information about the current status of the clock chip. The decoded identifiers are entered in the "CP clock status" field.

CLOCK FUNCTIONS	SINEC NCM CP 143 (EXIT)
WEEKDAY: ##### CURRENT DATE: ##.##.#### CURRENT TIME: ##:##:## TIME DIFFERENCE (1/2 H): # ## CLOCK MASTER : # CP CLOCK STATUS: #####	
F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/>	
1 2 READ 3 SET 4 5 UPDATE 6 7 OK 8 SELECT	

F2
READ

The time is requested once by the PG

F3
SET

The time can only be set when the CP status is "master clock" or when the clock chip of the CP 143 is marked as invalid.

F5
UPDATE

The PG requests the time cyclically. The CP clock status is also updated.

F7
OK

The data edited in the screen form are entered as the current data.

F8
RETURN

Branches back to the previous screen form.

Displays in the COM 143 screen

WEEKDAY:	MONDAY - SUNDAY
CURRENT DATE:	e.g. 15. 11. 1991 The data can be set within the limits 01.03.1984 to 31.12 2083.
CURRENT TIME:	e.g. 15:23:43
TIME DIFFERENCE (1/2 H):	"+" or "-" and range between 0 and 24
CLOCK MASTER:	Indicates whether the current CP 143 is the clock master or is a clock slave
CP CLOCK STATUS:	CLOCK MASTER the clock sends synchronization frames CLOCK SLAVE the clock receives synchronization frames CLOCK SLAVE,+ INVALID the clock must be set CLOCK SLAVE,+ ASYNC clock does not receive synchronization frames SLAVE <--> MASTER clock status change TRANSMITTER ASYNC time transmitter is itself asynchronous REPLACEMENT SYNC the clock is synchronized from a CP 143 SYSTEM ERROR an internal error has occurred HW CLOCK FAILURE hardware clock has failed

6.4.8 Restrictions / Tips

The time should be read or set by the programmable controller (with RECEIVE) at a time interval > 10 ms.

- a) The hardware clock of the CP 143 itself only has a resolution of 10 ms.
- b) With an extremely fast handshake with the handling blocks of the PLCs, the CP 143 can be influenced so that the module is disabled for other activities.

To avoid loading the SINEC H1 bus with unnecessary time messages, a synchronization time greater than 10 sec should be selected.

To ensure perfect functionality when there is no SINEC time transmitter on the SINEC H1 bus, the following points must be taken into account:

- > CPs must be distinguished uniquely with byte 0 of the Ethernet address.
- > The cycle time for synchronization frames on every CP 143 is the same. The preset cycle time is 10 sec (can be modified in the INIT screen form).
- > At least one dynamic clock master must be programmed.

6.4.9 Accuracy

➤ Absolute accuracy

The absolute accuracy of the clock chip on the CP 143 is in the worst case +/- 11.94 sec per day.

➤ Relative accuracy

If the times on the SINEC H1 network relative to each other should not deviate by more than 20 ms, the relationship between the Ethernet address (ID) and cycle time of the synchronization frame must be borne in mind.

Deviations of the hardware clock dependent on the transmission of the synchronization frame:

ID	Cycle time of the synchronization frame and resulting deviation		
	1 sec	10 sec	60 sec
0	0.28 ms	2.77 ms	16.6 ms
1	0.55 ms	3.04	16.88 ms
10			19.37 ms
15			20.75 ms
20			
25			
30		- !! -	
35	- !! -		
40			
45			
50			
55			
60		19.37 ms	
65		20.75 ms	
70	19.64 ms		
75	21.03 ms		

□

6

Notes

7 Non-Open Services for Serial Transfer

This chapter will familiarize you with the so-called non-open services.

The serial transfer services are known as "non-open services". They are not included in the scope of functions of the international standard (MMS standard) and can therefore not be modelled on MMS services.

You should only consider using these services when you do not require the conformity with devices of other manufacturers that can be achieved with the open services.

Above all, you should check whether the characteristics and concept of the variable services or the domain services would not in fact be better for your communication task. The non-open services are used primarily with existing systems.

The serial transfer function class is distinguished by the following characteristics:

Data are exchanged between the client and server without address information or parameters relating to the meaning of the data.

The following non-open services are supported by the CP:

Read byte string	(client + server)
Write byte string	(client + server)
Transparent data exchange	(client + server)

7.1 Overview of the Functions and Services

General

Applications making use of serial transfer have already negotiated the structure and content of the data. In the frame itself, no further information is transmitted apart from the length specification and the data itself.

Since no address information is transferred, only one data area (source or destination) can be identified via an application association.

Read/write byte string

The byte string services are used for "unidirectional" data transfer, i.e. data are transmitted in only one direction: with the "read byte string" service in the acknowledgement message, and in the "write byte string" service with the job message.

By calling the "read byte string" service, the client requests data from the server. The job frame itself must not contain data. The client obtains the data from the server in the acknowledgment.

With the "write byte string" service, the client transfers data to a server. The client can decide whether or not an acknowledgment is required.

Transparent data exchange

With the "transparent data exchange" service, data exchange can be bi-directional. Data can be transmitted both in the job message and in the acknowledgement. The client can decide whether or not an acknowledgement is required.

Segmented jobs are not supported on either the client or server.

Advantage and restrictions

The serial transfer services provide the greatest degree of freedom when configuring communication associations. The user (programmer) must, however, negotiate the meaning and processing of the data.

Compared with direct use of layer 4, the user can make use of the TF infrastructure with the serial transfer services. This ensures for example increased reliability with the logical acknowledgment of messages, the time and logical monitoring of TF jobs and the possibility of better use of a transport connection by multiplexing at the application layer.

TF service	Job PDU	Acknowledgment PDU
Read byte string	Data request frame without data	Acknowledgment frame with requested data
Write byte string	Data frame	Acknowledgment frame without data or no acknowledgment
Transparent data exchange	Data frame	Acknowledgment frame with or without data or no acknowledgment

7.2 Read Byte String (Client)

Request transfer of a data area.

Job buffer "read byte string"

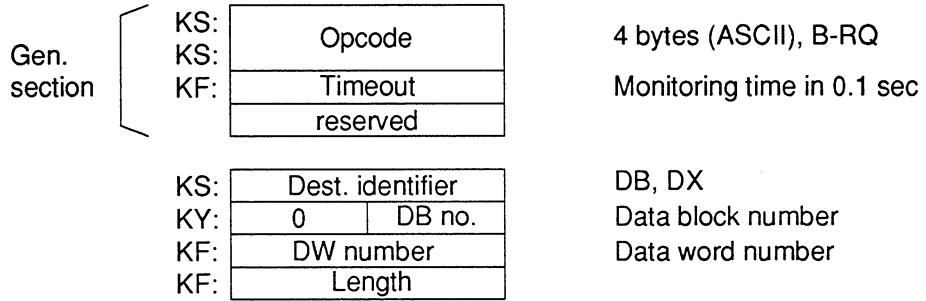


Fig. 7.1 Structure of the Job Buffer "Read Byte String"

Job buffer

General section:

Opcode: B-RQ

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec.
For further information about timeout, see page 3 - 9.

Job-related section:

Dest. identifier: Format: KS
Values: DB, DX
Meaning: S5 address at which the byte string read out will be stored.

DB number: Format: KY
Values: high byte: 0, low byte: 1..255

DW number: Format: KF
Values: 0..2042

Length: Format: KF
Values: 1..2043
Meaning: length of the data block area in which the byte string can be written.

7

Sequence of "read byte string"

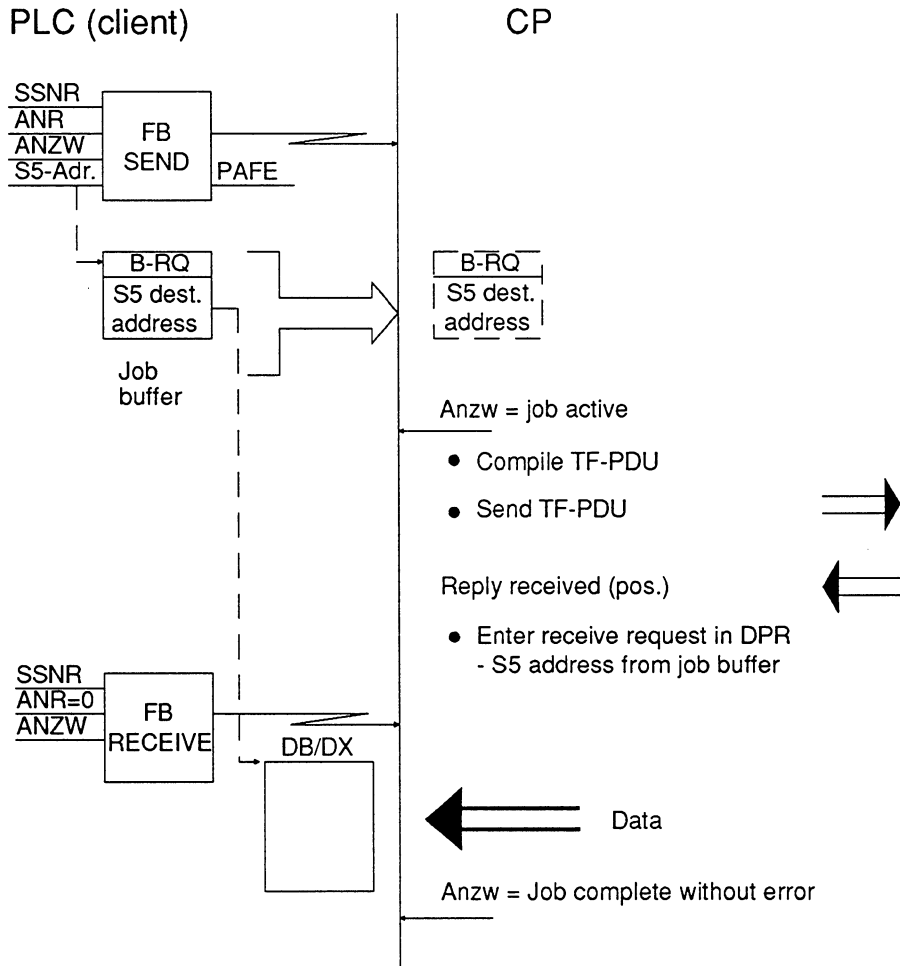


Fig. 7.2 Sequence "Read Byte String"

If the job cannot be processed without errors (not illustrated here) this is indicated to the user in the status word (see description "write byte string").

7.3 Write Byte String (Client)

Transfer a data area to the partner

Job buffer "write byte string "

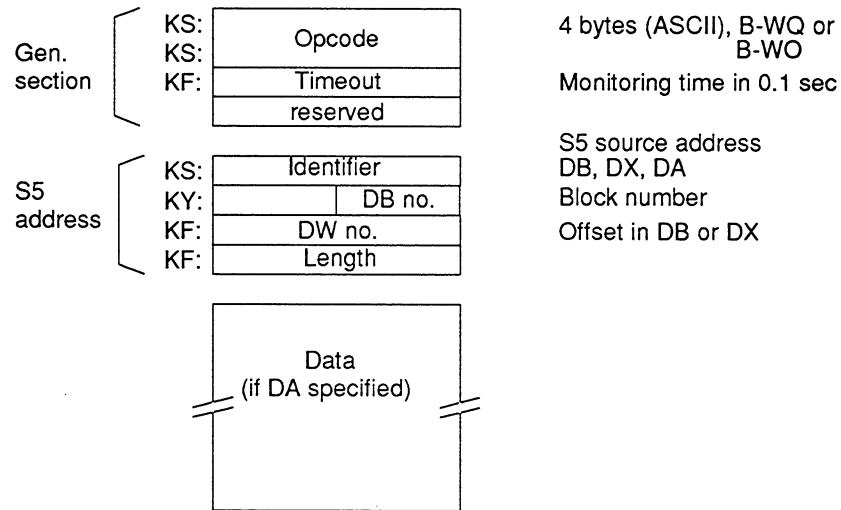


Fig. 7.3 Structure of the Job Buffer "Write Byte String"

Job buffer

General section

Opcode: B-WQ ("write byte string with acknowledgement")

or

Opcode: B-WO ("write byte string without acknowledgement")

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified

in multiples of 0.1 sec.

For further information about timeout, see page 3 - 9.

Job-related section

Source ID: Format: KS
 Values: DB, DX, DA
 Meaning: S5 source address at which the byte string to be
 written is stored.
 DB for data block
 DX for extended data block
 DA for data in job buffer

Note on the "DA" identifier

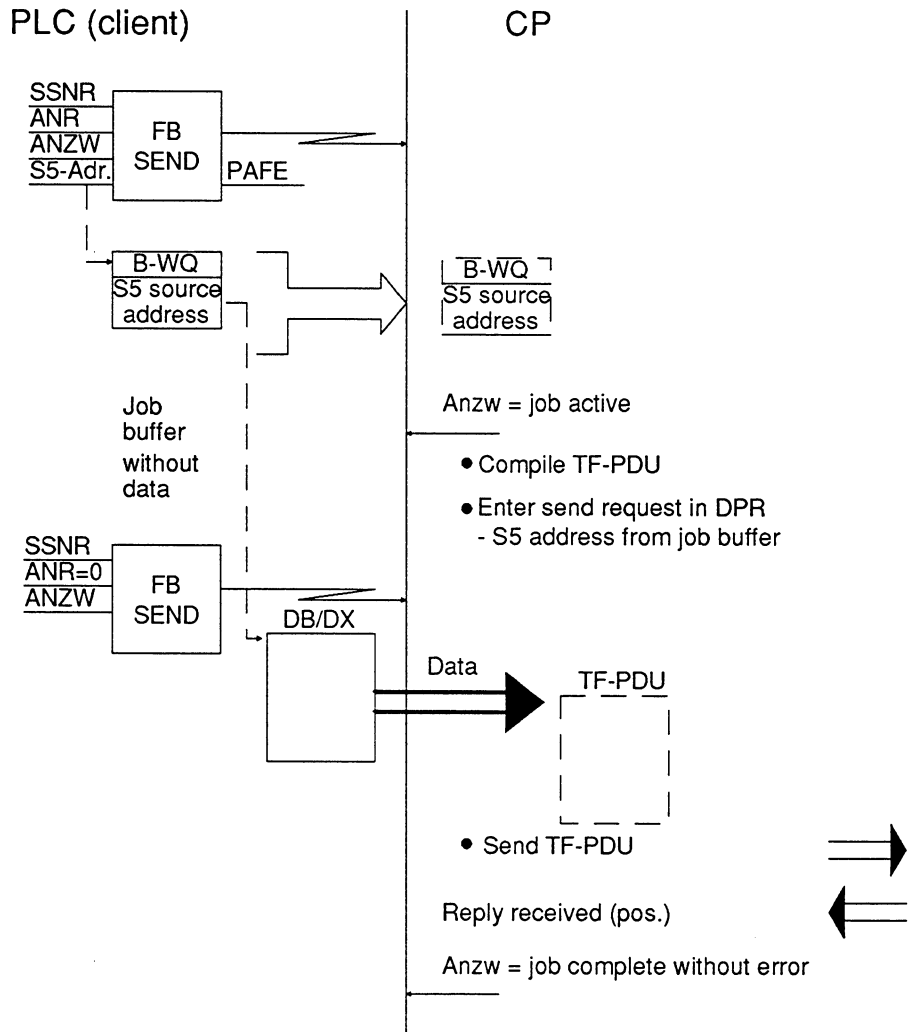
Apart from the TF service specification and the required parameters, the S5 user program can also transfer the data simultaneously to the CP. This is possible when the specified S5 address is a data source. This allows a considerable increase in the data throughput, as can be seen in the description of the sequence (see detailed description of the services). When using this facility, remember that a job buffer must not exceed 256 bytes. The data must follow on immediately after the last valid parameter of the job buffer.

DB number: Format: KY
 Values: high byte: 0, low byte: 1..255
 invalid with source ID DA

DW number: Format: KF
 Values: 0..2042
 invalid with source ID DA

Length: Format: KF
 Values: 1..2043
 Meaning: length of the data block area to be transferred
 with the service.

Sequence of "write byte string"



7

Fig. 7.4 Write Byte String with Acknowledgment (Example: Source ID = DB/DX)

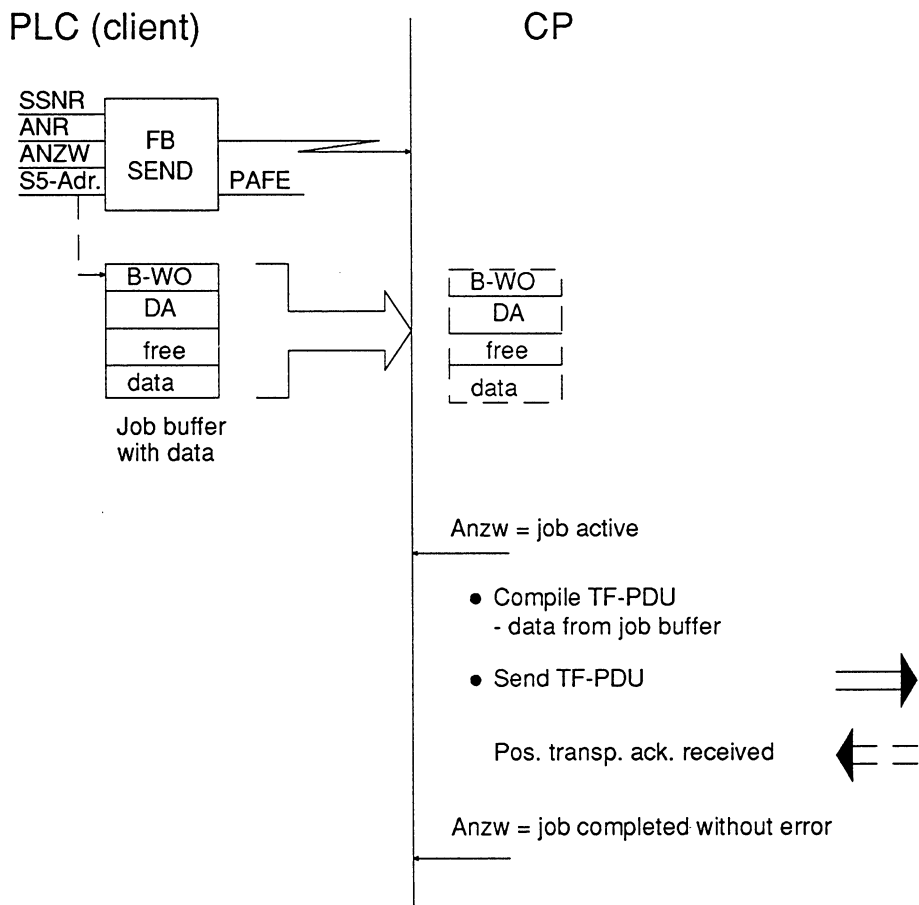


Fig. 7.5 Write Byte String without Acknowledgment (Example: Source ID = DA)

Errors

The situation in which the job could not be completed without errors is not represented here, but is indicated to the user as with the other services by setting the error identifier (TF error occurred) in the status word and entering the parameters "ERRCLS" and "ERRCOD" of the reply in the third word after the status word address.

The situation in which the server is unable to accept the data completely (ERRCLS = 2AH, ERRCOD = 1H) is a special case. To allow the client to determine how many bytes have been accepted in the server, it can trigger a (local) "request byte string length" job (see below), by specifying an S5 destination address at which the parameter "number of accepted data" in the server is to be stored. This job must always be triggered when the parameter mentioned above is to be transferred to the PLC and is then always valid for the last "write byte string" job triggered by the client.

The service is only executed locally and is only useful when a "write byte string" with acknowledgement (B-WQ) was started immediately before.

The "request byte string length" is triggered as with all client jobs by a job buffer, whose structure is illustrated in the following diagram. Following this, the sequence is illustrated based on an example.

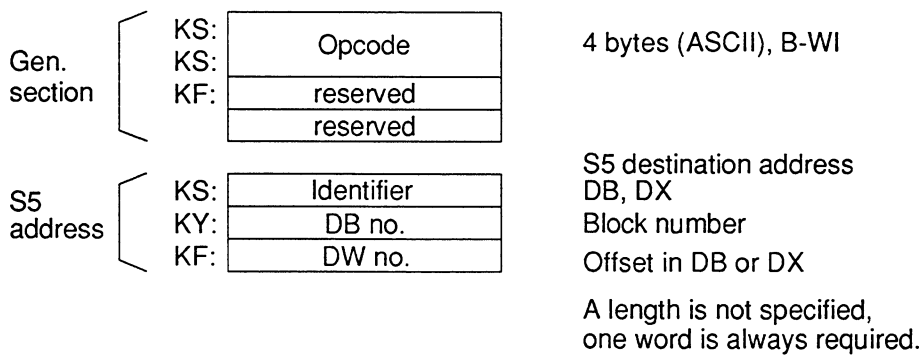


Fig. 7.6 Structure of the Job Buffer "Request Byte String Length"

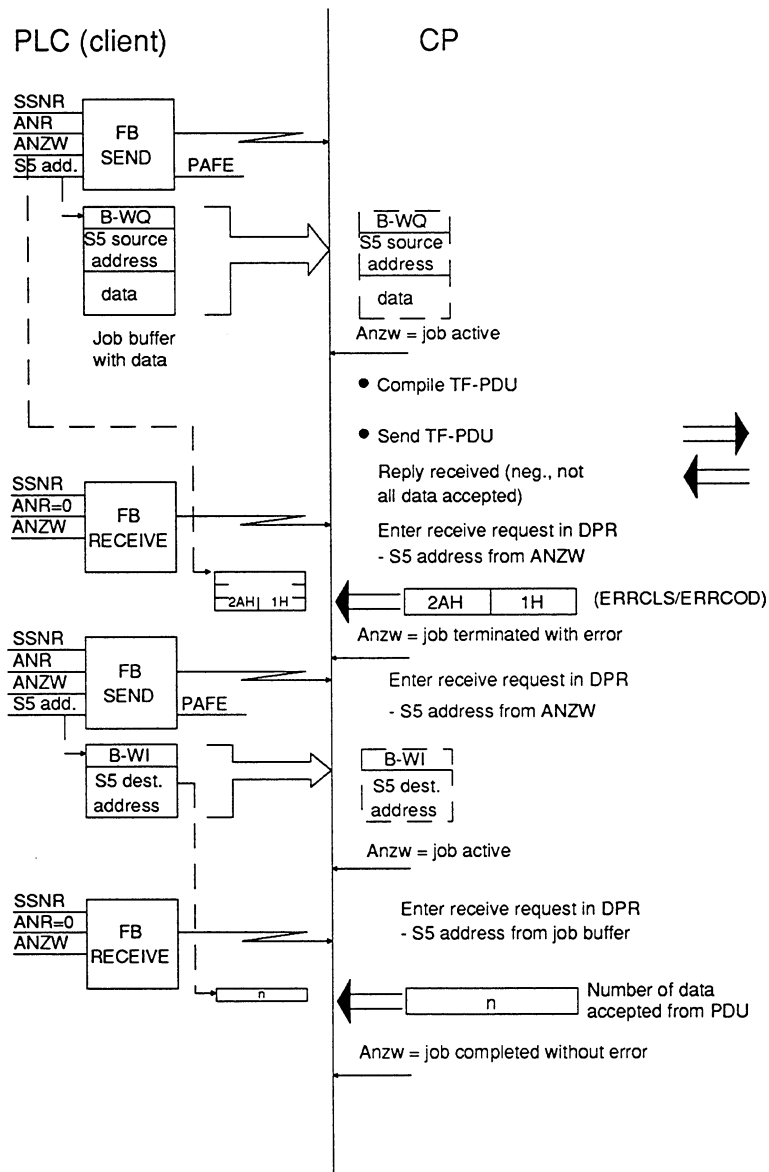


Fig. 7.7 Write Byte String with Acknowledgment if Error Occurs, when the Server cannot accept all the data of the job (Example: Source ID = DA)

7.4 Read/Write Byte String (Server)

Both these services are interpreted and executed in the CP on the server side, largely without support of the PLC CPU. All the CP handling blocks "SEND-ALL" and "RECEIVE-ALL" required as DMA substitutes must be called in the PLC program.

The assignment of the data to an S5 address must be configured for these services (see also "configuration jobs"). This means that before the CP can process this type of service, an appropriate configuration job must be triggered locally on the link.

To allow server processing of a "read byte string" job, the source of the data is specified in the configuration job. For the server processing of a "write byte string" job, the destination of the data is specified in the configuration job. The configuration job must be transferred to the communications processor as a client job with a "SEND DIRECT" job.

Job buffer "configure ANZW" for the server handling of "write/read byte string" jobs

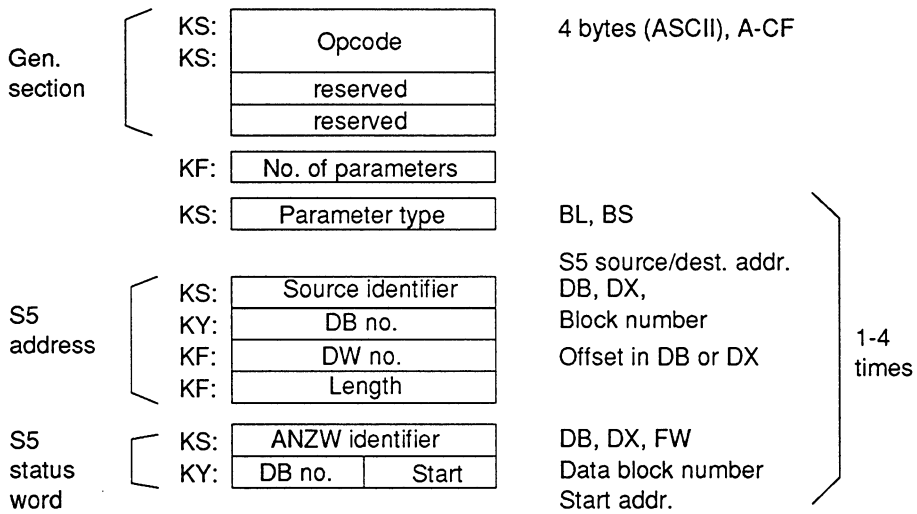


Fig. 7.8 Structure of the Job Buffer "Configure ANZW"

Call description

General section

Opcode: A-CF

Job-related section

Number of parameters: Format: KF
 Values: 1..4
 Meaning: up to four parameters can be transferred simultaneously.

Parameter type: Format: KS
 Values: BL, BS
 Meaning: specifies the parameter to be configured.
 BL: source address for "read byte string" server job
 BS: source address for "read byte string" server job
 S5 address

Source/destination identifier: Format: KS
 Values: DB, DX
 Meaning: source/destination address identifier for configuring the address for the read/write byte string services.

DB no.: Format: KY
 Values: high byte: 0, low byte: 1..255
 Meaning: data block number for the parameter types "BL", "BS".

DW no.: Format: KF
 Values: 0..2042

Length: Format: KF
 Values: 1..2043, -1
 Meaning: length of the data block area to be transferred with the "read byte string" service or to be made available for the "write byte string" service. The value -1 means that all the data can be accepted (only permitted for parameter ("BS").

Status word ANZW identifier
 Format: KS
 Values: FW, DB, DX
 Meaning: status word type

ANZW specification
 Format: KY
 Values: high byte: block number
 low byte: DW number, FW number

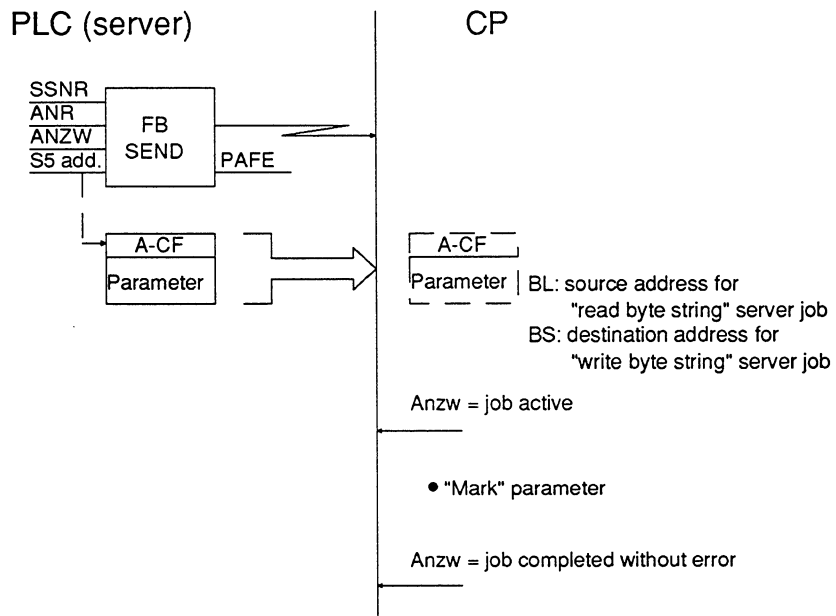


Fig. 7.9 Sequence: "Configure ANZW" for handling "Read/Write Byte String" jobs on the server

7.5 Transparent Data Exchange (Client)

A data area is transferred to the partner with the implicit request to send a reply with data.

Job buffer "transparent data exchange"

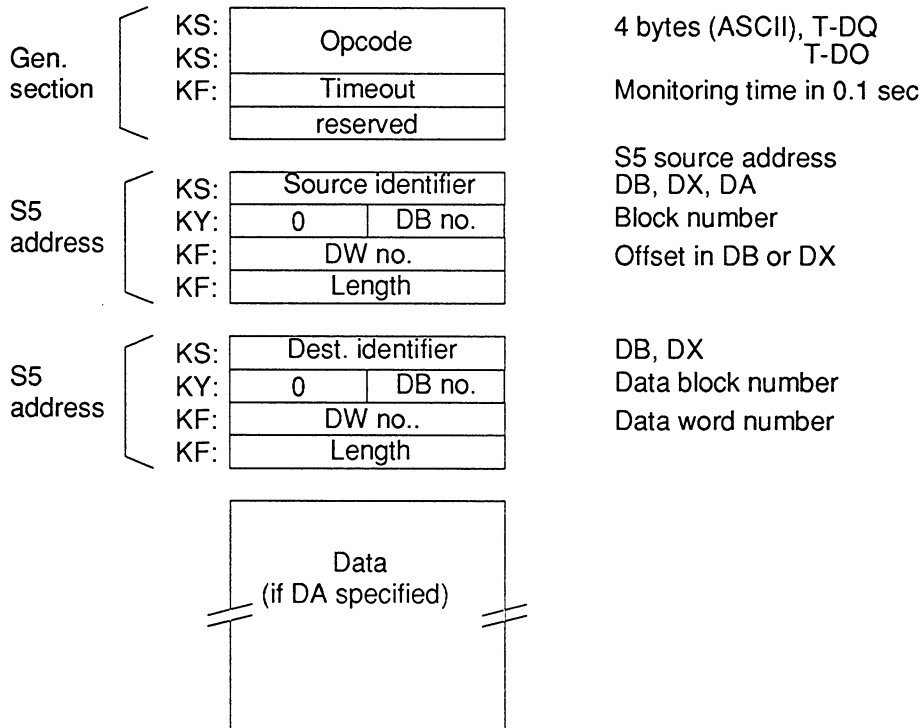


Fig. 7.10 Structure of the Job Buffer "Transparent Data Exchange"

Job buffer

General section

Opcode: T-DQ ("transparent data exchange with acknowledgement")
or
Opcode = T-DO ("transparent data exchange without acknowledgement")

Note: If the client receives an acknowledgment with data length zero after a job with acknowledgment, this is indicated by the SINEC TF error number 3028.

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgement for the service (i.e. the maximum dwell time of the job in the CP). This is specified in multiples of 0.1 sec. If the job is completed within the specified time, the parameter is irrelevant.
For further information about timeout, see page 3 - 9.

S5 source address

Source ID: Format: KS
Values: DB, DX, DA
Meaning: Address at which the data to be transmitted are stored.

DB for data block
DX for extended data block
DA for data in job buffer

Note on the "DA" identifier

Apart from the TF service specification and the required parameters, the S5 user program can also transfer the data simultaneously to the CP. This is possible when the specified S5 address is a data source. This allows a considerable increase in the data throughput, as can be seen in the description of the sequence (see detailed description of the services). When using this facility,

remember that a job buffer must not exceed 256 bytes.
The data must follow on immediately after the last valid
parameter of the job buffer.

DB number: Format: KY
Values: high byte: 0, low byte: 1..255
invalid for source ID DA

DW number: Format: KF
Values: 0..2042
invalid for source ID DA

Length: Format: KF
Values: 1..2043
Meaning: length of the data block area to be transferred
with the service.

S5 destination address

Dest. identifier: Format: KS
Values: DB, DX
Meaning: Address at which the data in the
acknowledgement will be stored. This address can also be
invalid (DB no. = 0). In this case, no data can be expected
in the acknowledgement.

DB number: Format: KY
Values: high byte: 0, low byte: 0..255

DW number: Format: KF
Values: 0..2042

Length: Format: KF
Values: ..2043, -1
Meaning: length of the data block area in which the data in
the acknowledgement will be stored; the value -1 indicates
that all the data in the acknowledgement from the DW
number to the end of the data block can be accepted.

7

Sequence of "transparent data exchange"

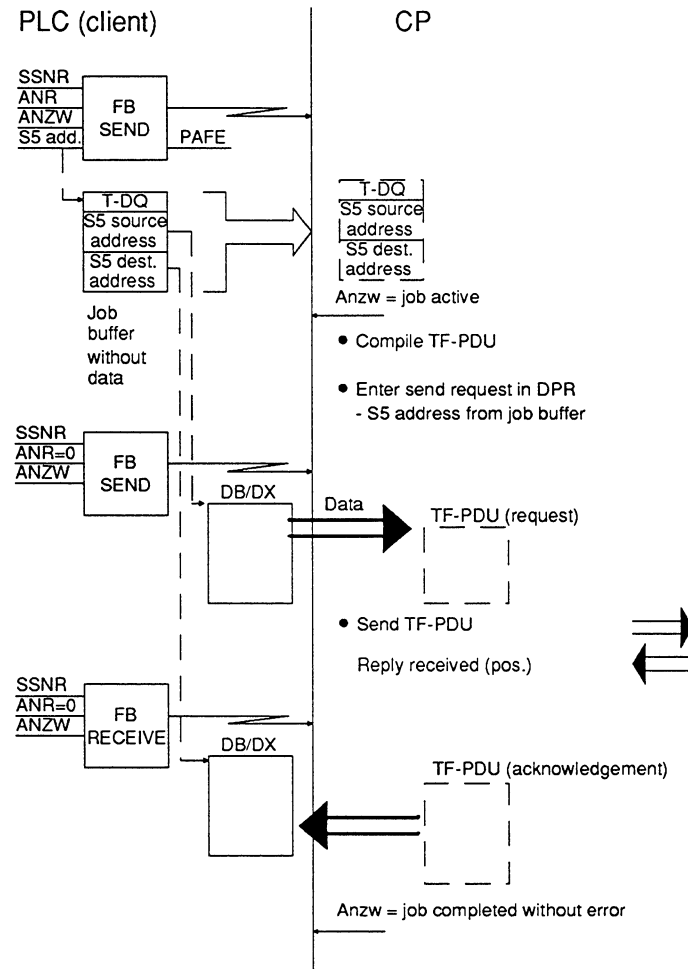


Fig. 7.11 Transparent Data Exchange with Source ID "DB" or "DX"

If the source identifier in the job buffer is "DA", then the SEND-ALL block call is omitted in the sequence illustrated here.

For the service without an acknowledgement request (T-DO), or if the acknowledgement does not contain data or the destination identifier in the job buffer is invalid, then the RECEIVE-ALL block call is omitted in this sequence.

7.6 Transparent Data Exchange (Server)

For the non-open STF service "transparent data exchange" the service request must be passed on to the PLC program, since the data can only be interpreted there. To allow for this, the parameter and data section of the TF-PDU is preceded by a "job header" in which the service is described and passed on to the PLC.

Structure of the job header (three words)

KY:	Opcode
KY:	reserved
KY:	length

In the opcode, the CP informs the PLC CPU of the required service as follows:

High byte: 0B0H: non-open STF services

Low byte: 0 transparent data exchange without acknowledgement
1 transparent data exchange with acknowledgement

The 2nd word in the job header is reserved (value 0 is entered) it contains a response code in the reply.

In the "length" parameter, the interface module informs the PLC how many valid bytes (without header) were transferred to the PLC.

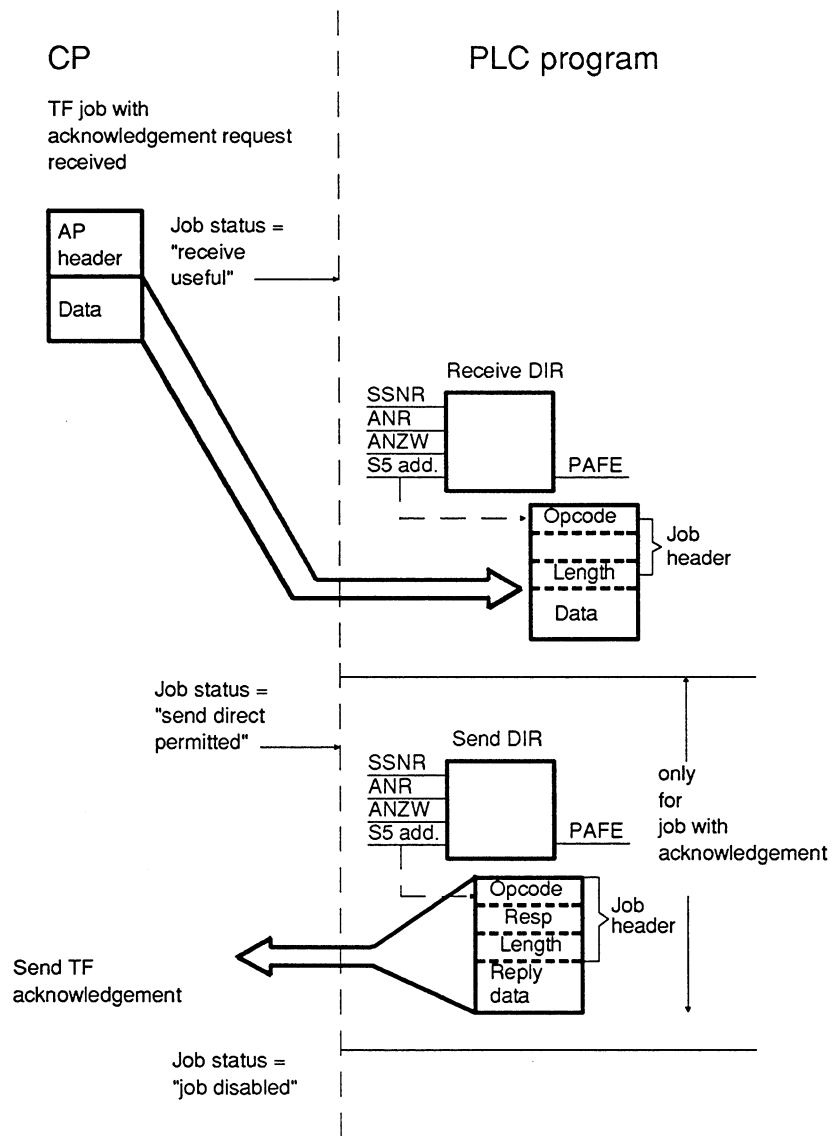


Fig. 7.12 Sequence "General Server Interface"

If the job does not require an acknowledgement, the status word is set to "job disabled" on completion of the receive job.

For a job with an acknowledgement, the PLC program must generate a reply that is transferred to the CP with a send direct call. The sending of the acknowledgement by the PLC program is not time-monitored by the CP. The first three words of the PLC acknowledgement must once again be contained in a job header, in which the second word contains a response code. The first word of the job header is taken from the job.

If the PLC wants to acknowledge the job positively, the response code is 0. Otherwise, error codes according to the AP protocol (ERRCLS, ERRCOD) are entered in this word. The response code is entered in the AP header by the CP. If data bytes are to be transmitted in the acknowledgement, these must be located immediately after the job header, the length of the data (in bytes) must be located in the third word of the job header.

If the acknowledgement does not contain data, the send direct call by the PLC program must have a length = 3. In the third word of the job buffer, the length = 0 must be entered (no data in acknowledgement).

If the length of the frame to be transmitted exceeds the frame length negotiated between the PLC and interface module during the course of a handling block call (send-dir, receive-dir), the segmentation is automatically carried out by the CP and entered in the status word to inform the PLC program.

If the length of the data to be transmitted exceeds the frame length set in the synchron HDB, the handling blocks SEND or RECEIVE-ALL must continue to be called in the cyclic PLC program. The PDU length must not be less than the number of data bytes to be transmitted. If all the data are accepted with a RECEIVE-DIRECT, remember that the maximum number of data to be received is not limited by the length specified for the handling block.□

7.7 Addendum to Transparent Data Exchange

7.7.1 Status Word of the TRADA on the Server

Status	ANZW without ackn.	ANZW with ackn.
After cold restart	0A0A not defined at present	0A0A not defined at present
After connection establishment	0E0A	0E0A
After connection termination	0A0A	0A0A
After trigger from client	00x3	00x3
Data reception triggered	00x2	00x2
Data all received	0ExA	0Ex4
Ackn. triggered by server	omitted	00x2
Ackn. transmitted	omitted	0E0A

The table illustrates the time sequence of the transparent data exchange service with/without acknowledgment and the status word on the server can be seen in greater detail.

When configuring the server, remember the header required for the receive and transmit buffers.

7.7.2 Example of a Program for Evaluating the Bits of the ANWZ on the TRADA

Coordination of the status word corresponds to the normal status word evaluation. The bits are scanned as described in the CP 535 and CP 143 manuals. The basic state can, however, cause confusion. This basic state (0E0A) is a self-protection mechanism of the CP so that jobs cannot be connected to the passive ANR without the CP being informed.

The following bit evaluation is suitable both for the TRADA with and without acknowledgment. The parameters for the direct blocks must be adapted to the situation.

The example was written for an S5 115U programmable logic controller but could easily be ported to other PLC types.

FB20

Segment 1 0000
NAME: TRADA

```

0005      :          Bit evaluation of the server
0006      :          ANZW with the non-open
0007      :          SINEC TF service TRADA
0008      :          -   ANZW for RCV-D and SNDD-D
0009      :          selected the same (not absolutely
000A      :          necessary)
000B      :          -   Run through CONTROL for this ANR
000C      :          at the beginning
000D      :
000E      :          =====
000F      :          System status bytes and control
0010      :          for the user:
0011      :          FY 80.0 = 1 => trigger RCV-D
0012      :          FY 80.1 = 1 => trigger SND-D
0013      :          FY 80.2 = 1 => run RCV
0014      :          FY 80.3 = 1 => run SND
0015      :          FY 80.7 = 1 => user acknowledgment
0016      :
0017      :
```

```

0018      :          FY 81 internal edge ANZB
0019      :
001A      :          FW 82 ANZW of the HDB
001B      :          FW 84 length word of the HDB
001C      :          FW 86 SINEC TF error word if
001D      :          link status word is the same
001E      :
001F      :
0020      :          O F 1.0
0021      :          ON F 1.0
0022      :          JU FB 247
0023      NAME      :          Update control status word
0024      :          KY 0,0
0025      :          KY 0,2
0026      ANZW      FW 82
0027      PAFE      :          FY 5
0028      :
0029      :
002A      :A F 83.0  Handshake useful &
002B      :A F 83.1  Job active &
002C      :AN F 83.3 Job complete without error
002D      :S F 80.0  -> RCV-D can be triggered
002E      :S F 81.0
002F      :
0030      :A F 80.7  Acknowledgment for the trigger
0031      :          must be provided by the user
0032      :A F 81.0
0033      :R F 80.7
0034      :R F 80.0
0035      :R F 81.0
0036      :JC FB 245
0037      NAME      :RECEIVE  RCV-D to trigger data reception
0038      SSNR      :KY 0,0
0039      A-NR      :KY 0,2
003A      ANZW      :FW 82
003B      ZTYP      :KS DB
003C      DBNR      :KY 0,3
003D      ZANF      :KF +0
003E      ZLAE      :KF + 103
003F      PAFE      :FY 5

```



```

0040      :
0041      :
0042      :AN F 83.0  Handshake not useful and &
0043      :AN F 83.1  Job not active &
0044      :AN F 83.3  Job complete without error
0045      :S  F 80.1  ->SND-D can be triggered
0046      :S  F 81.1
0047      :
0048      :A  F 80.7  Acknowledgment for trigger
0049      :                must be provided by user
004A      A  F 81.1
004B      :R  F 80.7
004C      :R  F 80.1
004D      :R  F 81.1
004E      :JC FB 244
004F      NAME   :SEND    SND-D to trigger acknowledgment
0050      SSNR    :    KY 0,0
0051      A-NR    :    KY 0,2
0052      ANZW    :    FW 82
0053      ZTYP    :    KS DB
0054      DBNR    :    KY 0,2
0055      ZANF :    KF +0
0056      PAFE    :    FY 5
0058      :
0059      :
005A      :JU FB 25  Call ALL blocks since
005B NAME :ALL  bit evaluation follows
005C      :
005D      :A  F 83.6  Data accepted
005E      :R  F 83.6
005F      :R  F83.4
0060      :S  F 80.2  Identifier for the user that the data
0061      :                have been accepted completely
0062      :
0063      : A  F 83.5  Data transfer successful
0064      :R  F 83.5
0065      :R  F 83.4
0066      :S    F 80.3  Identifier for the user that the data
0067      :                were transferred completely
0068      :
0069      :BE

```

FB 25

SEGMENT 1 0000
NAME: ALL

```
0005                   :JU FB 244
0006      NAME        :SEND      SEND-ALL for segmenting
0007      SSNR        :     KY 0,0
0008      A-NR        :     KY 0,0
0009      ANZW        :     FW 100
000A      ZTYP        :     KS
000B      DBNR        :     KY 0,0
000C      ZANF        :     KF +0
000D      ZLAE        :     KF +0
000E      PAFE        :     FY 5
000F                   :
0010                   :JC FB 244
0011      NAME        :RECEIVE   RCV-ALL for segmenting
0012      SSNR        :     KY 0,0
0013      A-NR        :     KY 0,0
0014      ANZW        :     FW 102
0015      ZTYP        :     KS
0016      DBNR        :     KY 0,0
0017      ZANF        :     KF +0
0018      ZLAE        :     KF +0
0019      PAFE        :     FY 5
001A                   :
001B                   :BE
```

The data blocks must have the appropriate headers.□

8 Configuring and Testing the TF Interface

If you have decided to use the TF interface, you will find the descriptions of the required configuration functions of the COM 143 TF tool in this chapter.

The main point of interest is the description of the link in the data link block. Apart from this, the definition and use of variables are important when configuring the TF interface.

Along with the configuration functions for the data link block, you are also provided with the tools necessary for documentation, starting-up and testing the link.

The possibilities range from start/stop of the CP to data-dependent monitoring of the communication.

This chapter explains the use of the following functions:

- configuring application associations
- configuring variables
- test functions for links
- documentation functions

During configuration, it is advisable to perform the steps in the order shown in the following overview.

The margin entries shown on a gray background are references to the "COM 143 TF Configuration Tool" supplement accompanying this manual.

8.1 Overview

The diagram below is an overview of the steps and functions available for configuring and operating a TF interface.

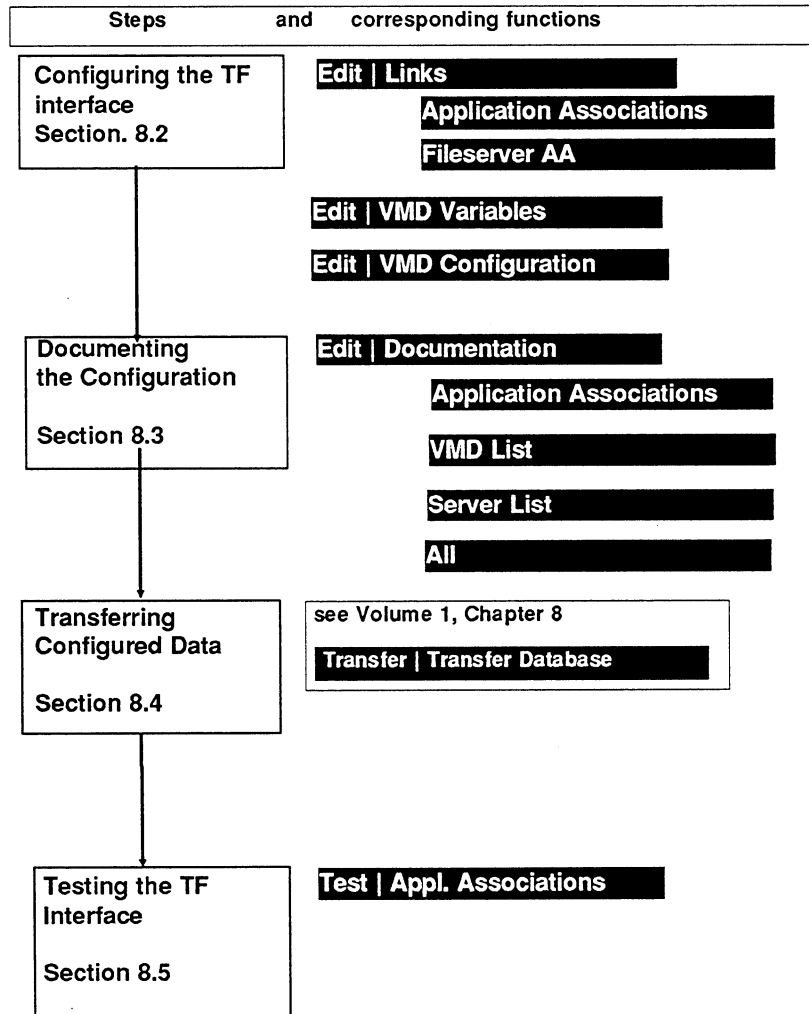


Fig. 8.1: Configuration Steps for TF Services

8.2 Configuring the TF interface

8.2.1 Edit | ... | Application Associations

Aims

M2-4-2.1

This screen allows the definition of data link blocks that can be used for TF services. Starting with this link configuration, link-specific variables can then be configured by changing from this screen to the variables editor.

Link-specific configuration of variables

When configuring variables, a distinction is made between the local definition for server functions and the remote definition for client functions:

Local definition : Variables are managed in the PLC whose application associations are being configured. Other communications partners (PLCs) write or read these variables.

Remote definition : The PLC whose application associations are being configured writes, reads or reports variables.

Essential basic configuration

A data link block can only be defined when the module file has been completely initialized. Refer to the procedure for basic configuration described in Volume 1, Chapter 7.

A maximum of 64 SINEC application associations each with a TF-PDU size of 1024 bytes can be configured.

Note:

Some of the information in this screen is identical to the parameters for calling the handling blocks in the STEP 5 user programs.

M2-4-2.1

The parameters of the screen for data link blocks

SSNR

The parameter specifies the interface number or page number via which the communications channel is addressed. The number is formed from the base interface number and the selected page. The PG checks that the interface number is within the permitted range of values and rejects illegal interface numbers.

The number specified here must also be specified in the handling block for link identification.

Range of values: base SSNR + page number (0..3)

ANR:

In conjunction with the local interface number, the job number specifies the data link block uniquely. This means that the ANR cannot be assigned more than once per SSNR of a CP 143 module.

Only odd job numbers can be assigned.

In the control program, the job number and the interface number must be transferred to the handling block to identify the link and job.

Range of values: 1..127

ANZW:

Status word for the defined job (see: specification of handling blocks)

Form: FW 0.....,199 or
 DB' < 0,....,255> '< 0,....,32...>
 " "

DB number word number

With client jobs, this status word is transferred by the CP to the HDBs in the SEND/RECEIVE-ALL jobs.

M2-4-2.1**Note:**

The link status word is used for the CLIENT station, i.e. is the service initiator.

Three words are required for the status word. It is advisable to make this status word identical to the handling block status word.

Structure of the ANZW:

		M
Length word		M + 1
ERRCLS/ERRCOD		M + 2

If the address of the ANZW for the link is different from that of the ANZW for the HDBs, the address M and M+1 are used exclusively for the SEND/RECEIVE ALL block.

Local TSAP: Transport service access point of the local device.

Representation and input of the TSAP:

LENGTH: This has the default "8". With connections to non-SIMATIC S5 stations, it may be necessary to specify shorter lengths.
Range of values: 2..8

HEX: The individual bytes of the TSAP-ID must be entered in groups of two hexadecimal numbers (values 00 to FF). To make the ID easier to read, the groups of two numbers should be separated by blanks. With this type of input, the TSAP-ID can also have numerical values. If you only enter zeros here, the TSAP counts as being unspecified.
Range of values: 8 Bytes

M2-4-2.1

ASC: The TSAP-ID entered in the HEX field is displayed here as an ASCII string. Blanks and non-interpretable characters are displayed as underscores. An ASCII character entered here is displayed in the "HEX" field in hexadecimal notation.
Range of values: 8 characters

Example.
Length: 5 HEX: 31 32 33 34 35
ASCII: 12345

Note:
This distinction between hex and ASCII has the following advantages:
- specifying TSAPs as an ASCII string
- TSAP selection not only restricted to ASCII characters.

Est. type A4 = active connection establishment only layer 4
A7 = active connection establishment layers 4 and 7
P4 = passive connection establishment only layer 4
P7 = passive connection establishment layers 4 + 7
D4 = dynamic connection establishment only layer 4
D7 = dynamic connection establishment layers 4 + 7

Connection establishment using only layer 4 should be selected when there is no application association management implemented on the partner system. For more detailed information, refer to Chapter 6 (Application associations).

The establishment of a link is handled implicitly by the CPs, i.e. there is no explicit job from the PLC. Connections are established as follows:

active: the CP initiates connection establishment during start-up.
passive: the CP expects a connection request from the partner.

M2-4-2.1M

dynamic: the CP establishes the connection as soon as a job is pending for the configured link.

Multiplex address of the layer 7 link
 Range of values: 0,...,255
 where 0 : no multiplexing

Note:

The multiplex addresses of layer 7 links must be unique for the current layer 4 connection.

**Remote
 TSAP:**

Analogous to local TSAP

For correct connection establishment, the local TSAP must match the remote TSAP of the partner and vice-versa.

**Appi. Ass.
 Name:**

Name of the defined application association
 Range of values: 32 ASCII characters
 (only for information)

**Remote
 Ethernet-
 Address:**

Ethernet address of the partner station
 Range of values: 12 characters from {0,...,F}
 Default: 080060010000

**Number of
 appl. ass.
 for cur. TSAP**

Number of defined jobs <SSNR, ANR> for this layer 4 connection, defined by TSAPs and Ethernet address in the case of SINEC H1.
 Range of values: 1,...,255
 Default: 1

**Current
 job:**

This indicates which of the jobs included in the number of application associations field is currently being processed or displayed.
 Range of values: 1,...,255

8

Note: The link-specific part described here for COM 143 refers to SINEC H1 connections and is used to supply the transport software. The global parameter field of a data link block is generated from the TSAPs and Ethernet address. The corresponding local parameter fields (1... number of jobs) are made up from the establishment type, MUX address, logical partner name, local variables and remote variables.

M2-4-2.1**Function keys** (additional keys or keys with a special significance):

F1 +1

Display the next data link block or next field of local parameters for a layer 4 connection. Here, there are two possibilities:

a) Number of jobs > 1 and current job < number: only the next local parameter field of the already displayed link is displayed. The global parameters can then no longer be modified.

b) Next data link block is read and displayed or if the number of jobs is 1, the first local parameter field is displayed.

If there is no further block, the display begins from the beginning again.

F2 -1

Analogous to F1, paging backwards

F3 INPUT

Input new data link block:

F4 LOCAL

Submenus for variable definitions, to be used for server services (read, write) (-> M 2-4-2.2)

F5 REMOTE

Submenus for remote variable definition to be used in client calls (-> M 2-4-2.3)

F6 DELETE

The current job (currently displayed job) is deleted (confirmation is required: delete yes/no). If there is only one job for the connection (layer 4), the whole data link block is deleted.

F7 OK

All the changes are accepted. If you are working offline, the module file is modified.

M2-4-2.2 Configuring local variables

After selecting the function keys F4 - LOCAL in the M 2-4-2.1 screen, the LOCAL DEFINITION screen is displayed.

Here, you define or configure local, link-specific variables and their structures belonging to values in local data blocks. These variables can only be accessed via the link currently being configured.

Name: Maximum 32 character long identifier for a variable.

Type: Specifies the type of the variables: the first input field (2 characters long) identifies the actual variable type and the 2nd input field (4 characters long) the variable length.

M2-4-2.2

The following types of variable are permitted:

BO	Boolean	-
BS	Bit string	number of valid bits
IN	Integer	8, 16, 32 bits
UN	Unsigned Integer	8, 16, 32 bits
FP	Floating point	32 bits
OS	Octet string	length in bytes
VS	Visible string	length in bytes
TI	Time of day	4 bytes generated by COM
TD	Time and date	6 bytes generated by COM
{	Start of structure	number of components (calculated by COM)
}	End of structure	
AR	Field	number of elements in an array

Table 8.1 Variable Types

ACC: Access type:
R only reading possible, no specification means access not restricted

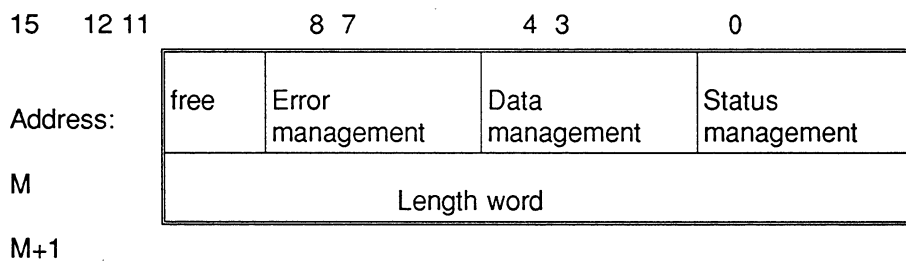
SS ADDRESS: Address of the memory area for modelling the value of the variable on the PLC.
FORMAT: <ORG_identifier> <EXT_identifier> <start addr.>
 ORG_identifier = DB or DX for data block or extended data block
 EXT_identifier = 0..225 data block number

M2-4-2.2

Start addr. = data word number, at which value of the variable starts.

ANZW: Configured status word for server functions with these variables
 Format: <TYPE> <DBNO> <DWNO> or <FWNO>
 (Range of values: TYPE = FW, DB, DX
 DBNO = 0..255, if TYPE = DB or DX (data block number)
 FWNO = 0..255, if TYPE = FW (flag word number)
 DWNO = 0..255, if TYPE = DB, DX (data word number)
 DWNO = empty, if TYPE = FW).

Note:
 The variable status word is required for the SERVER station. Two words are required for this status word and overlapping must be avoided.



Apart from configuring the name and type of the variables and the S5 address at which the variable is stored by the PLC program, it is also possible during configuration to specify an address for a status word. Here, the PLC program can then obtain information about access to the variable. The PLC program can, for example, recognize whether the value of a variable has been updated by another station or that there has been no access. Using the status word, the PLC program can also block access to a variable for another station (temporarily). How to use the status word and the meaning of the individual bits can be found in the descriptions of the CP handling blocks (see Chapter 3).

M2-4-2.2

The status word as presented here is identical in its structure to the status words of the handling blocks.

SSNR: This specifies the CPU in which the variable is physically located. In the single processor mode the SSNR specified for the link must be entered (default). In the multiprocessor mode, ((CPU no.) -1) must be entered.

Since in the SIMATIC PLC in a station (e.g.S5-155 U, S5-135 U) up to four CPUs can operate in the multiprocessor mode, not only the S5 address but also the interface via which the CP accesses the variable (i.e. the physical CPU on which the variable is located) must be specified when configuring VMD-specific and link-specific variables.



The nesting level for variables is restricted to 2.

Function keys (additional keys or keys with a special significance):

F5
DELETE

The variable marked by the cursor is deleted. If this variable is a structure or an array identifier, the whole structure or array is deleted and the variables definitions are shifted together to close the gap.

F6
INSERT

A line is inserted for a new definition at the cursor position.

F7
CONTINUE

With this key, you can change the softkey menu. Your input can then be entered with the OK function key (F6).

8

Sofkeys displayed after changing over with F7:

F6
OK

The data entered is converted to internal structures. You quit the screen, but only save the data when you quit the link screen.

N2-4-2.2 Example:

LOCAL DEFINITION		SOURCE:C:XXXXXXXX					CP 143 (EXIT)
NAME	TYPE	ACC	S5 ADDRESS	ANZW	SSNR		
OTTO	{	3	R	DB 100 200	FW 30	1	
KARL	AR	10					
	IN	16					
THEO	{	2					
CHRIS	IN	16					
ANDREW	UN	32					
	}						
ULLI	VS	16					
	}						

F F F F F F F

1 PAGE +1 2 PAGE -1 3 LINE +1 4 LINE -1 5 DELETE 6 INSERT 7 OK 8 SELECT

Fig. 8.2: Local Variables

In plain language, this screen has the following meaning:

- OTTO is a structure with the following components:
- KARL = an array with 10 elements of the type integer 16
 - THEO = a further structure with the components:
 - CHRIS = integer 16
 - ANDREW = unsigned integer 32
 - ULLI = visible string with a length of 16 bytes

The structure OTTO can only be read owing to ACC = R. The structure is mapped on data block 100 starting at data word 200. Flag word 30 is used as the status word for the variable.

Note the following points:

For structures, you only need to specify the S5 address of the start of the structure. Where the components of the structure (within a data block) are

actually located is calculated by COM 143 and displayed the next time you page through the variables definitions.

M2-4-2.2

The length of structures is also calculated by COM 143 based on the components entered and is displayed the next time the variable definitions are called.

The calculation of S5 addresses and structure lengths is only started when you exit the edit mode with "OK" (F7) since only then are the external ASCII structures converted to the internal representation.

If an error is detected, an error message is generated and displayed. e.g.:
"Error in group: xyz variable: OTTO not defined" "Parenthesis error"
(the cursor is then positioned on the line containing the error).

The definitions are inset automatically to make the screen clearer, but this is only visible the next time the variables are edited.

8.2.1.1 Configuring Remote Variables

M2-4-2.3

Analogous to the link-specific local variables accessed by the CP as server, remote variables can also be specified which can be accessed via the link acting as the client (known as "remote variables").

These variable definitions are used by the CP when the variable is not completely specified in the job buffer or cannot be completely specified (e.g. no type specification in the job buffer) during a client call (read/write a variable).

The scope must also be specified for such variables.

Input fields:

SCOPE: scope of the remote variables

VB = link
VM = virtual machine
DO = domain

NAME: analogous to definition of local variables

TYPE: analogous to definition of local variables

SS ADDRESS: analogous to definition of local variables

ANZW.: analogous to definition of local variables

SSNR does not need to be specified, since in client jobs, the addressed variable must always be located in the CPU from which the job was triggered.

8.2.2 Edit | VMD Variables

The VMD variables menu item provides you with functions for defining VMD-specific variables.

VMD-specific variables are stored in a special organization block within the configured module.

Meaning of the scope

VMD-specific variables can be accessed via any application association.

Creating groups for the information report service

VMD-specific variables can be collected together into groups. Groups are purely local objects which allow simpler access to all the grouped variables for the information report service.

M2-5.1

The "Local Definition" screen is structured identically to the screen for variable definitions when configuring links (see page 8- 10).



The nesting level for variables is restricted to 2.

Function keys (additional keys or keys with a special significance):

F5 DELETE

The variable marked by the cursor is deleted. If this variable is a structure or an array identifier, the whole structure or array is deleted and the variables definitions are shifted together to close the gap.

F6 INSERT

A line is inserted for a new definition at the cursor position.

F7 CONTINUE

With this key, you can change to a new softkey menu.

M 2-5.1

The following function keys are available in the second softkey menu. (additional keys or keys with a special significance):

F2
GROUP

This function allows variables to be grouped together. The screen M2-5.3 'Definition of Grouped Variables' is displayed. (see Section 6.2.2.1)

F4
DEL BLOCK

All the VMD-specific variables can be deleted, however, you will be prompted to confirm your intention DELETE YES/NO.
After deleting the variables, you return to the TF overlay initial screen form.

F5
DELETE

The variable marked by the cursor is deleted. If this variable is a structure or an array identifier, the whole structure or array is deleted and the variables definitions are shifted together to close the gap.

F6
CONTINUE

With this key, you can change to the first softkey menu.

F7
OK

The variable definitions are converted to corresponding address directory entries and checked to make sure they are syntactically correct (parenthesis error, double declarations). The defined groups are also converted to an internal structure and checked (unique group names, existence of all the specified variables names).

If an error is detected, an error message is displayed, for example, as follows:
"error in group: xyz variable: OTTO not defined"
"parenthesis error" (the cursor is then positioned on the incorrect line.

M2-5.2**Defining Groups**

You display the "Definition of grouped variables" screen by pressing function key F2 in the VMD variables screen, screen M2-5.1.

By defining a group, you can transfer variables collected in groups using the information report service.

Input fields:

Group name: 8 characters long ASCII string, must be unique per VMD.



Note that the variable names must be entered line by line and not one column after the other.

The free 32 character long fields are for variable names. Only complete structures or arrays can be accessed using the structure names or array names.

Function keys (additional keys or keys with a special significance):

F3 INPUT

A screen form with empty fields is displayed in which a new group can be defined.

F4 DELETE

The currently displayed group is deleted.

F7 OK

Any changes you have made are accepted and the screen form is exited. The data are only converted to the internal representation when you exit the VMD screen form, since you can still change the variables definitions.

8.2.3 Edit | Fileserver AA

M2-4-3

In this screen form you can define so-called "third-party associations". These are links that are established between the CP and a fileserver after a load operation is triggered by a host computer.

Screen structure

The screen is identical to screen M2-4-2 'LINK', however, the input fields for SSNR, ANR, ANZW, EST TYPE, MUX ADDRESS and NO OF JOBS are omitted.

Neither local nor remote definitions are possible (function keys F4 and F5).

The establishment type for the server application association is always "dynamic". The connection establishment request (in order to load a new domain), is always started by the CP 143.

8.2.4 Edit | VMD Configuration

M2-6

The VMD Configuration function is used to specify the master CPU for domain and PI services in the multiprocessor mode.

The master CPU is the CPU via which the PG functions "PLC start" and "PLC stop" are executed.

The function allows up to four CPUs to be selected and assigned to the VMD. If more than one CPU is selected, the field COR is automatically activated.

The currently selected CPU is displayed inversely on the screen. CPUs that have been clicked on (selected) are displayed with a heavier margin. You select a CPU using the function keys.

Default: CPU 1 and CP are selected.
(The CP cannot be deleted and is only included to illustrate the complete configuration.)

Note: COR and CP are not input fields.

If you press F7 = OK, at least one CPU must be selected.

Meaning of the input fields:

Master - CPU: An x must only be entered in one of these four fields. If you enter a second x, the first one is automatically deleted.

COR/MUX-ADDRESS: This specifies the path via which the CP can reach the CPU. If you do not make an entry, this means that the CPU can be reached via the "swing cable" without the MUX.

The entry can be made for each selected CPU (even if only one is selected). Values between 1 and 38 are possible. An entry is only obligatory if more than one CPU is selected.

M2-6

SSNR: Only if no COR/MUX address is entered, otherwise as for
 COR/MUX
 Default : 0,
 In the multiprocessor mode : CPU-NO - 1.

The information about the configuration is only generated after it has been selected by the user. If the information does not exist, a CP uses the default setting.

Function keys (additional keys or keys with a special significance):

F1
INC/DEL

Include or remove a CPU

F3
←-----

Select the previous CPU

F4
----->

Select the next CPU

F6
PRESETS

Create presets

8.3 Documenting the Configuration

To document the configured communication parameters, there are functions available for outputting lists. The lists are structured based on the configuration steps. All the lists are displayed on the screen and can be printed out if required (how to control output is described on the following page).

8.3.1 Application association list

Output of all application association objects

8.3.2 VMD list

Output of all VMD objects

8.3.3 Server list

Output of the server list

8.3.4 All

With the menu item "ALL", all the lists are generated one after the other. The screens correspond to screens M 2-7-5 to M 2-7-7.

Both the application association objects and the VMD objects are output.

Additional function keys for all documentation functions

Output can be stopped or resumed with the function key 'STOP/CONTINUE':

M 1-1**Controlling output**

In the 'Init | Edit' screen, you can control the printout and footer. In the PRINTER field, you can switch the printout on or off. In the FOOTER (ON/OFF) field, you can decide whether to include a footer at the bottom of each page. If you want the printout to have a footer, the footer file must be specified. The footer is created with the "Footer editor" utility.

8.4 Transferring Configured Data

8.4.1 Transfer | Transfer Database

M 3-1 ...

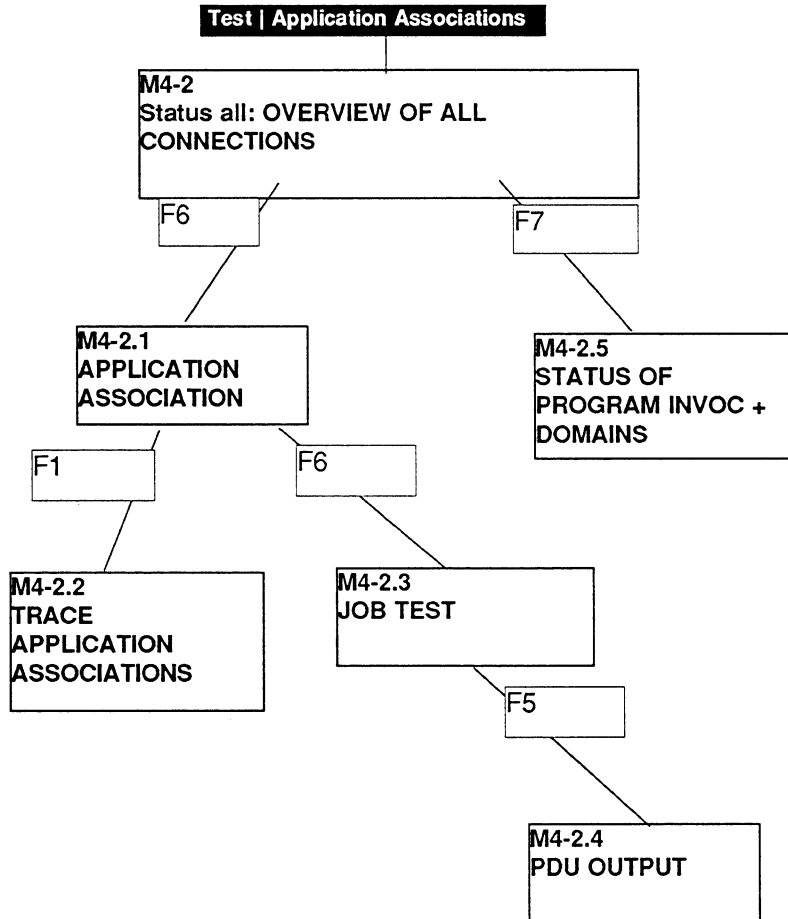
M 3-6

The transfer functions for the configured parameter records are used in exactly the same way for the blocks of the transport interface and the TF interface. The transfer functions are explained in Volume 1 of the COM 143 TF description.

8.5 Testing the TF interface

The test functions allow the statuses of individual parts of the system to be recognized at the PG during communication and therefore allow errors to be localized.

The diagram below is an overview of the screen hierarchy of the test functions. The screens can be displayed by pressing function keys in the basic screen "Overview of all connections".



M4-2

8.5.1 Outputting Status All

If you select the function 'Test | Application associations', the function "TF status all" is executed between the PG and CP 143. The PG obtains the current status of all programmed application associations and displays this in the "overview of all application associations" screen form.

Output fields and their meaning:

POS:	Position indicator, links are numbered in ascending order (0-255).
SSNR :	interface number.
ANR :	Job number.
PRIO:	Priority of the link.
LINK :	Status of the layer 4 connection (see table 'Link status').
LINK E:	Link flags to indicate error in the layer 4 connection (see Figure 'Link flags').
Job :	TF job (see table TF jobs).
ERROR	Error in TF job
CHA	When the information is updated, the '**' indicates which link has changed.

The following links always exist:

ANR 203:	Job number for the third-party-fileserver application association
ANR 204	Job number for loading with the S5-DOS package PGLOAD
ANR 205:	This job number is used for local jobs.

Note: with the test function, it is not possible to record the sequence of a load job.

M4-2

Overview of the link statuses (LINK) of the layer 4 connection

'0'	The link has a defined basic status. The link is defined but not established.
'2'	The preparations for connection establishment have been made, the link itself has not yet been established.
'4'	Status '2' has been completed successfully. The ACTIVE partner sent a connection request or the PASSIVE partner sent a connection await.
'17'	The layer 4 connection is completely established. The transport parameters may still have to be set. The transport is, however, executed automatically.
'1'	The layer 4 connection is completely established. From now on, certain TF services of the CP 143 can be executed via this connection.
'8'	The layer 4 connection has been terminated. Since, however, buffers are still occupied
'36'	The layer 4 connection establishment is completed, the layer 7 link establishment has been initiated. Either an application association initiation message is sent or the partner is waiting for such a message.
'33'	The layer 7 link is established. All TF services of the CP 143 can be executed via this link.
'24'	The layer 7 link conclusion is requested.
'40'	The layer 7 link has been concluded, aborted or has broken down.

Table 8.2 Link Statuses

M4-2

Link flags provide information about dynamic events affecting the connection.

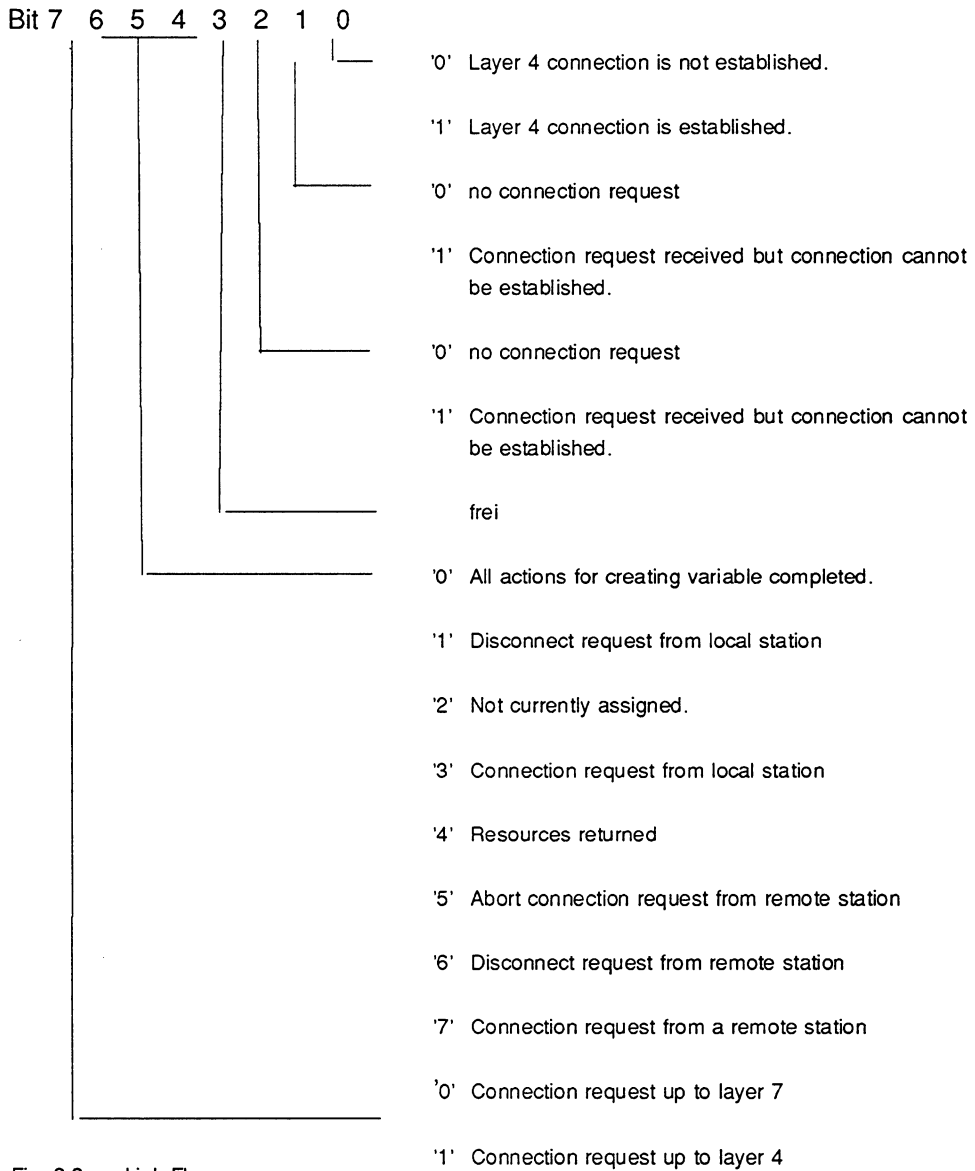


Fig. 8.3 Link Flags

M4-2

The TF jobs with the corresponding opcodes and texts are as follows:

Opcode	Job
M-ST	Status
M-GN	Get name list
M-ID	Identify
V-RE	Read variable
V-WR	Write variable
V-GE	Get variable attributes
V-IN	Information report
P-ST	Start PI
P-SP	Stop PI
P-RE	Resume PI
P-RS	Reset PI
P-AB	Kill PI
B-RQ	Read byte string
B-WQ	Write byte string with ack.
B-WO	Write byte string without ack.
T-DQ	Transp. data exch. with ack.
T-DO	Transp. data exch. without ack.
D-LI	Initiate up/download
D-LS	Up/download segment
D-LT	Terminate up/download
D-UI	Request upload sequence
D-US	Upload segment
D-UT	Request upload sequence response
D-LR	Request download sequence
D-UR	Request upload sequence
D-LO	Load domain content
D-ST	Store domain content
D-DE	Delete domain content
D-GE	Get domain attributes
P-CR	Create PI
P-DE	Delete PI
P-GE	Get PI attributes
D-CA	Get capability list
M-SU	Unsolicited status
A-IN	Initiate
A-TE	Conclude
A-AB	Abort
A-CF	Configure ANZW (local)
B-WI	Request byte string length
P-HL	Local program stop

Table 8.3

M4-2

The job statuses for the status display and the corresponding texts for the PG are as follows:

Job status	PG text
0	Disconnect
1	Connect
2	No job
3	Job
4	Request
5	Confirmation
6	Response
7	Indication
8	Send All
9	Receive All
10	Error output
11	Receive Dir

M4-2

Function keys (additional keys or keys with a special significance)

F3
LINE + 1

When the links are displayed, the first line is displayed inversely (first link is selected).
With F3, you can move on to the next line which is then displayed inversely.

The cursor key "arrow down" has the same function. The link selected in this way is then the selected link for single status.

F4
LINE - 1

Analogous to F3

F5
UPDATE

If you press the "UPDATE ON/OFF" function key, the status is constantly updated. Only status changes are indicated and marked in a special field.

F6
SEL LINK

The selected link is valid for the next functions you select. The screen form is exited and the single status of the link is displayed (screen M4-2.1).

F7
PI-DOM

By pressing this key, you can obtain the PI/domain status (screen M4-2.5).

8.5.2 Output Single Status

M4-2.1

When you select a link in the "OVERVIEW OF ALL APPLICATION ASSOCIATIONS" display, the "TF single status" function is executed between the PG and CP 143 and the "APPLICATION ASSOCIATION" display appears on the PG with the status and link information for the selected link.

Updating the status

Pressing the "UPDATE" function key activates or deactivates the update status function. When it is activated, the status is continuously updated. Only status changes are indicated. After activating the update function, the "start TF single trace" function is run once between the PG and CP 143 and following this, the "enable TF single trace" function is called cyclically. The "stop test" function terminates the updating.

Function keys (additional keys or keys with a special significance)

F1 TRACE

Start trace functions

F5 UPDATE

Activate/deactivate cyclic updating of the data

F6 JOB TEST

Trigger job processing

8.5.3 Trace Functions

M4-4.3

Using the trace functions, status changes on a selected application association can be written to a trace buffer and displayed on the PG in chronological order. The function is activated with the TRACE function key in the APPLICATION ASSOCIATION screen (screen M4-2.1) .

Selecting the application association

The application association is selected with the "single status" function (screen M4-2.1).

Trace buffer overflow

If the CP 143 is not able to enter all the status changes in the trace buffer (overflow), this is indicated in the "trace application associations" screen.

8.5.4 Job Test

M4-2.3

The job test function or single step mode is started by pressing the "JOB TEST" function key in the APPLICATION ASSOCIATION screen.

The "job test" screen form appears on the PG as soon as data arrive from the CP.

While the PG is waiting for a reply, the message "waiting for data from CP" is displayed.

Meaning of the output fields:

SSNR,ANR:	Interface and job number.
ENABLE	The ENABLE field contains the default text AUTOMATIC. This status can be changed over with function key F1.
LINK STATUS:	Status of the layer 4 connection (see table in Section 6.5.1).
TF JOB	TF job (see table in Section 6.5.1).
JOB STATUS:	(see table in Section 6.5.1.)

Function keys (additional keys or keys with a special significance):

F1 AUT/MAN

This changes the type of enable as displayed in the ENABLE

F2 START

In the AUTOMATIC enable mode, an "enable TF job test" job is sent to the CP 143, the message "waiting for data from CP" is displayed and when the reply is received with the data, the display including the JOB STATUS is updated. Any additional information is displayed below JOB STATUS. This sequence is repeated until either F3 or F1 is pressed. F4, F5 and F6 are not permitted during this time.

If the MANUAL enable mode is selected, after pressing F2 the message "waiting for enable" appears. The "enable TF job test" is only sent to the CP 143 after you press this key. On reception of the reply, the display is updated and the PG then waits for the next enable. While the PG is waiting for the next enable, you can also use the functions of F4, F5 and F6.

F3 STOP

You can stop the job test with F3, particularly in the AUTOMATIC enable mode to request more information with F4, F5 or F6. The job test is continued with F2.

F5 TF-PDU

The "output TF-PDU" job is sent to the CP 143. The content of the current TF-PDU is displayed in a special screen form (screen M4-4.5). If no PDU exists, the message "no TF-PDU exists" is displayed.□

9 PGLOAD

The PGLOAD tool provides useful and user-friendly functions with which you can address programmable logic controllers (PLCs) via the TF interface conforming with the MMS standard.

PGLOAD is also required to structure a PLC with domain and program invocation objects in keeping with the TF services.

This chapter tells you how to use the PGLOAD tool for the following purposes

- To supply PLCs with programs either directly using the TF domain services or dynamically via file servers and so keep up to date with the current tasks in the process.
- To monitor and control PLC using the TF program invocation services.

The tool is integrated in COM 143 TF. You can activate it with the menu item "Utilities".

9.1 Overview

9.1.1 Adapting Programmable Logic Controllers to the Process with PGLOAD

Domain and PI services

The SINEC technological functions include not only variable services but also domain and program invocation services, as follows:

- > A domain corresponds to a loadable data area in a PLC.
- > Single domains are collected together to form a program invocation (PI), that represents an executable program for an automation task.
- > Program invocations can be monitored and controlled with the PI services.

Example

If the algorithmic sequence of an automation task is always the same and only the parameters and data/variables change from time to time (red, green, blue cars), then the program can be structured as follows:

- > the algorithmic sequence in a "program" domain
- > the changing parameters in their own "parameter" domain.

Depending on the requirements, the program domain can be combined with one or more parameter domain(s) to form an executable PLC program.

Using domain and PI services without programming

With the PGLOAD tool, you use "off-the-shelf" load and control programs and with them the facilities of the domain and PI services without needing to program. With the PGLOAD user interface, you simply specify the connections for data exchange and select objects for the transfer (domains) to the controller (PIs).

The PGLOAD host computer functions allow you to intervene in the running of the PLC again using the PGLOAD user interface and so you implicitly use the TF services.

Structuring the PLC with PGLOAD for TF services

PGLOAD is responsible for forming domains from blocks and generating loadable domain files. PGLOAD is therefore needed for PLC structuring in terms of the TF services.

9.1.2 Range of Functions

TF services under PGLOAD

The PGLOAD program supports the following domain and program invocation services:

- Domain services:
 - uploading a domain from the PG into a fileserver (triggered by the PG)
 - downloading a domain from the fileserver to the PG (triggered by the PG)
 - triggering the load sequence in the PLC from a fileserver or PG (triggered by the PG)
 - triggering the uploading from the PLC into a fileserver or PG (triggered by the PG)
 - deleting domains in the PLC.
- Program invocation services:
 - start program invocation
 - stop program invocation
 - create program invocation
 - delete program invocation

TF services and the SIMATIC S5 view

PGLOAD is designed for the domain and PI model on which a SIMATIC S5 programmable logic controller is modelled.

The following rules apply to domains:

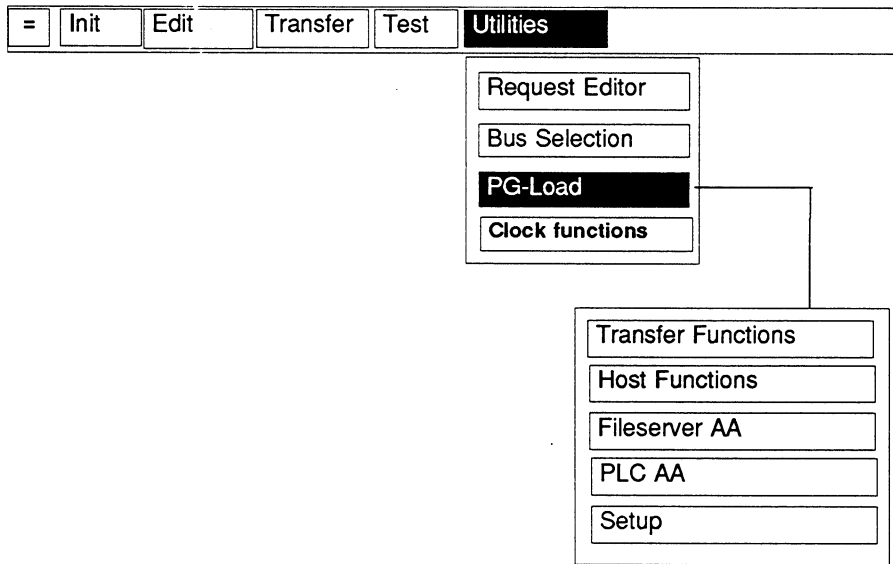
- A PLC contains up to 8 domains.
- A domain consists of any combination of blocks. It is useful to separate blocks containing program (logic) and those containing parameters (data).

The following applies to PIs:

A PLC consists of two PIs: a system PI and an application-specific PI. The application-specific PI includes all the domains loaded on the PLC.

9.2 Description of the Tool

The structure of PGLOAD is reflected in the menu:



Initialization

With the initialization function, you select the files for configuring application associations.

PLC AA

With PLC AA, you specify an application association between the PG and the selected PLC (PI services).

Fileserver AA

You specify a third-party association. This determines which PLC is supplied with data from which fileserver. The PG with PGLOAD assumes the initiative in this three-way association.

Transfer Functions

Transfer functions are used to transfer load files to a PLC or to convert and retransfer load files to the PG. Load files contain the program and data blocks put together to form domains.

Host Functions

With the host computer functions, you have direct control over a PLC addressed via an application association. You can therefore use the TF program invocation services without needing to write programs.

9.3 Functional Description

9.3.1 System Configuration and Device Functions

This section describes the use and functions of PGLOAD.

To simplify the explanation, the following network of devices is assumed.

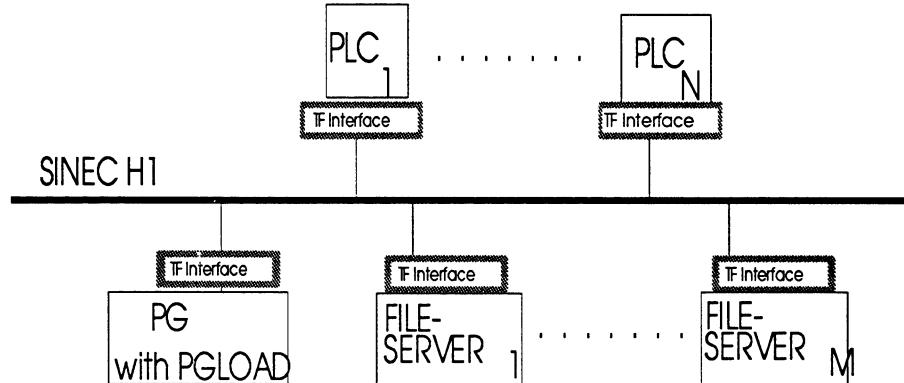


Fig. 9.1 Example of a Network when using PGLOAD

Programmer PG

The PG is integrated in the automation system via SINEC H1. With PGLOAD it is responsible for controlling the PLC program and data supply and for monitoring the PLC programs (host function).

Programmable logic controllers PLC 1 to n

Programmable logic controllers are responsible for controlling the process. PLCs are supplied with programs and data (domains) either on their own initiative or instigated externally (PGLOAD).

Fileserver 1..m

Fileservers are devices with capacity for storing the programs and data required on the PLCs. In special cases, a PG can also be used as a fileserver. PGLOAD supports the situation in which the PG is used both for the PGLOAD functions and for data storage as a fileserver.

9.3.2 Configuring Links and Selection Functions**Aims and Procedure**

The logical partners, i.e. the PLCs and fileserver can be defined using a link editor. The link definitions are written as link blocks in a file for the PLC links and fileserver application associations.

Existing application associations can be selected in the current data link file and can be stipulated as defaults for transfer and host functions.

PLC links

PLC links are links between the PG and a PLC. These links are used to transfer jobs for fileserver access and PI service jobs.

Configuring a PLC link simply involves specifying the Ethernet address of the partner a link name used locally on the PG. The local and remote TSAPs are formed implicitly.

The link blocks for PLC links are stored in a file with the name: <xxxxxxCP.LOD> (xxxxxx is any 6-character long ASCII character string).

Server application associations

Server application associations are links between a PLC and a fileserver. The data transfer from and to a fileserver is triggered by a third party, in this case the PG. This is known as a third-party association .

Configuring therefore requires the specification of the Ethernet address and the service access point on the PLC (known here as the local address) and the appropriate specification for the fileserver (known here as the remote address).

The link blocks for fileserver application associations are stored in a file with the name :<xxxxxxPG.LOD> (xxxxxx is any 6-character long ASCII character string).

9.3.3 Transfer Functions

Support of data storage

The transfer functions support the conversion and transfer of S5 program files to the fileserver. They also support retransfer and reconversion when the programs require modification with the programming tools (LAD, CSF and STL).

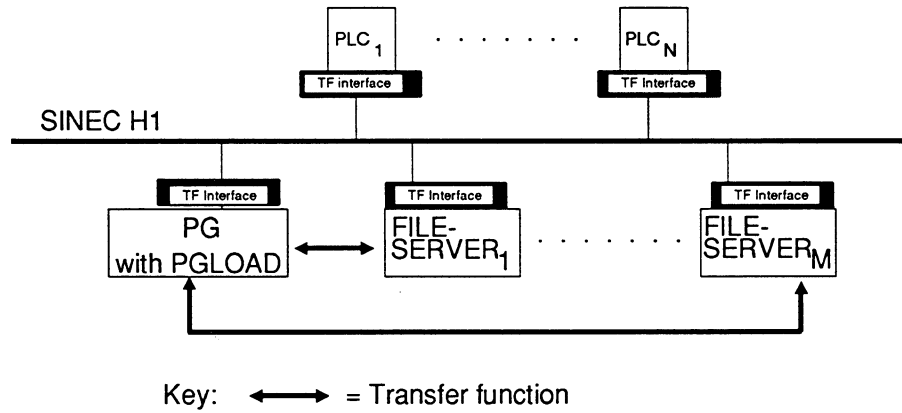


Fig. 9.2 Direction of the Transfer Functions in a Network

The following functions are available

Transferring domains from the PG to a filesaver. Domains on the PG are transferred to a filesaver for data storage.

An S5 program file is converted and transferred to the server along with local variable definitions (if they exist) as a loadable "domain file" using the upload function.

Transferring domains from the filesaver to the PG Domains on the filesaver are transferred to the PG with the load domain TF service.

The received data are converted to a program file and, if applicable, a variables file that can be processed with "S5 resources" (LAD, CSF, STL).

The name of the variables file is <xxxxxxST.VAR>. This is generated automatically by the tool from the name of the program file..

File conversion

The conversion functions can also be started separately. This means it is possible to store data on the PG (PG is filesaver) when transfer to a different filesaver is not required.

Restrictions

Dynamic loading and supply of programs and data (domains) for the PLCs is only possible with the filesaver functions. This also applies when the PG itself is used as the filesaver. This is then triggered by the host functions in PGLOAD (load PLC and save PLC).

9.3.4 Host Computer Functions

Aims

Whereas the functions introduced so far are used to prepare the system and the PG, the host functions support the system during operation.

The following functions are available:

- Load PLC:** One or a maximum of two domains in a screen can be transferred from the fileserver to a CPU of the PLC (the fileserver can also be the local PG).
- Situation a: third-party configuration
If the link name of the server is not "PG", a PLC (destination station) is instructed to load the domains from the fileserver.
- Situation b: PG = fileserver
If, however, the link name of the server is "PG", the domains are transferred directly from the PG to the PLC = destination station
- Save PLC:** A PLC domain can be saved on the fileserver. This means that the PG instructs the PLC to send the data to the server. The link name is handled in the same way as in the load PLC function.
- The distinction between a third-party configuration and PG=fileserver is analogous to the load PLC function.
- Delete PLC:** Domains in the PLC are deleted provided they are marked as deletable and their current status allows them to be deleted.
- Start/stop PLC:** The PLC is started or stopped (depending on its mode).
Implicitly: this means creating/deleting a program invocation.

Start program: The domains collected under the program invocation are started.

Stop program: If a program invocation with the specified name has been started, it will be stopped.

The following diagram illustrates the host functions:

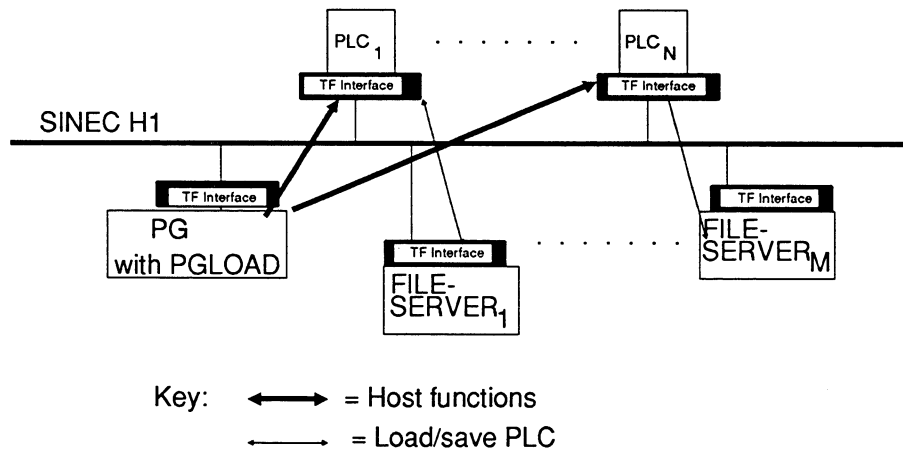


Fig. 9.3 Direction of the Host Functions in a Network

Note: The PG load program only operates via the integrated Ethernet interface, i.e. CP 141, CP 1413.

9.4 PGLOAD - Application

9.4.1 PGLOAD | Initialization

The data link files specify the destination devices that can be accessed via PGLOAD. The first task to perform with PGLOAD is therefore to specify these data link files.

After selecting PGLOAD | Initialization, PGLOAD displays the following screen:

PGLOAD		CP 143 (EXIT)	
Initialization PGLOAD for H1 interface		SINEC NCM	
DATA LINK FILE (FILESERVER)	:DR:	<input type="text"/>	NAME: <input type="text"/> PG.LOD
DATA LINK FILE (PLC)	:DR:	<input type="text"/>	NAME: <input type="text"/> CP.LOD
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7	8 SELECT

Fig. 9.4 Start screen

Input fields

DATA LINK FILE (FILESERVER):

Depending on the functions you require, you can specify a data link file for server application associations with the format:

DR := A, ..., N
NAME : <XXXXXXPG.LOD>
where <XXXXXX>: = can be freely selected.

The suffix PG.LOD is fixed and cannot be changed.

DATA LINK FILE (PLC) Here, you specify the data link file for PLC links .

Format :

DR := A, ..., N
NAME : <XXXXXX> CP.LOD
where <XXXXXX>: = can be freely selected.

The suffix CP.LOD is also fixed and cannot be changed.

9.4.2 Link Configuration / Server Selection

By selecting PGLOAD | Server Links, you can configure fileserver application associations.

Server application associations are links between a PLC and a fileserver.

PGLOAD		CP 143 (EXIT)	
Link Configuration / Server Selection		SINEC NCM	
LINK NAME :	<input type="text"/>		
LOCAL TSAP :			
LENGTH <input type="checkbox"/>	HEX <input type="text"/>	ASCII <input type="text"/>	
REMOTE TSAP :			
LENGTH <input type="checkbox"/>	HEX <input type="text"/>	ASCII <input type="text"/>	
REMOTE ETHERNET ADDRESS :	<input type="text" value="080006010000"/>		
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1 +1	2 -1	3 INPUT	4
		5 DEL LINK	6 DEL FILE
		7 OK	8 SELECT
			F <input type="text" value="HELP"/>

Fig. 9.5 Link screen: fileserver

Input fields:

LINK NAME: Name of the fileserver application association (must be unique for the file)
Range of values : 32 characters

LOCAL TSAP Service access point of the PLC on which domains will be loaded or from which they will be saved.

REMOTE TSAP Service access point of the fileserver.

Representation and input of the TSAP:

LENGTH: This field has the default "8" entered. If you require links to non-SIMATIC S5 stations it may be necessary to specify shorter lengths.
Range of values: 1 character

HEX: The individual bytes of the TSAP ID must be entered in hexadecimal format in groups of two. These groups should be separated by blanks. With this format, the TSAP ID can also be numerical. If you only enter zeros, the TSAP counts as unspecified.
Range of values: 8 bytes

ASC: The TSAP ID entered in the hex field is displayed here as an ASCII string. Blanks and non-interpretable characters are displayed as underscores. A TSAP ID entered here in ASCII characters is displayed in the HEX field in hexadecimal notation.
Range of values: 8 characters

Example:

Length: 5 HEX: 31 32 33 34 35
ASCII: 12345

Note:

The distinction between hex and ASCII has two advantages:

- TSAPs can be entered conveniently as an ASCII string
- The TSAP ID is not restricted to only ASCII characters .

REMOTE ETHERNET ADDRESS: The default Ethernet address 080006010000 is displayed, this can, however, be modified.
Range of values: 12 characters

Function keys:

F1, F2 +1, -1

With these function keys, you can load and edit the next or previous link block in the file.

F3 INPUT

You can set up a new link block. An empty LINK CONFIGURATION / SERVER SELECTION screen is displayed. The only available function key is then F7 OK and F8 SELECT/HELP. With F7 OK, the new link block is entered but not yet saved. The data are saved by pressing F7 again as described below.

F4 DEL LINK

The currently displayed link block can be deleted.

F5 DEL FILE

You can delete the whole data link file. You will first be prompted to confirm your intention: delete yes/no?

F7 OK

The currently displayed link is selected and is used as the default for subsequent functions. You exit the screen form, all changes are saved.

Possible errors and messages include the following:

File too long: i.e. already contains 30 link blocks.

No link defined:: this appears when you start with an empty file after deleting the last block.

Link name
not unique: the link name already exists

9.4.3 Link Configuration / PLC links

By selecting PGLOAD | PLC links, you can configure PLC links.

PLC links are links between the PG and a PLC. These links are used to transfer jobs for fileserver access and PI service jobs.

PGLOAD		CP 143 (EXIT)	
Link Configuration / PLC Selection		SINEC NCM	
LINK NAME :	<input type="text"/>		
REMOTE ETHERNET ADDRESS :	<input type="text" value="080006010000"/>		
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1 +1	2 -1	3 INPUT	4
		5 DEL LINK	6 DEL FILE
		7 OK	8 SELECT
			F <input type="text" value="HELP"/>

Fig. 9.6 Link Configuration PLC Selection

Input fields:

Link name: Name of the link to the PLC (must be unique for the file)..
Range of values. 32 characters

Ethernet address: The default Ethernet address 080006010000 is displayed, this can, however, be modified.
Range of values: 12 characters

Function keys:

F1, F2 +1, -1

With these function keys, you can load and edit the next or previous link block in the file.

F3 INPUT

You can set up a new link block. An empty LINK CONFIGURATION / PLC SELECTION screen is displayed. The only available function key is then F7 OK and F8 SELECT/HELP. With F7 OK, the new link block is entered but not yet saved. The data are saved by pressing F7 again as described below.

F4 DEL LINK

The currently displayed link block can be deleted.

F5 DEL FILE

You can delete the whole data link file. You will first be prompted to confirm your intention: delete yes/no?

F7 OK.

The currently displayed link is selected and is used as the default for subsequent functions. You exit the screen form, all changes are entered.

Possible errors and messages include the following::

File too long: i.e. already contains 30 link blocks.

No link defined: this appears when you start with an empty file after deleting the last block.

Link name not unique: the link name already exists

9.4.4 Using Transfer Functions

The transfer functions support the conversion and transfer of S5 program files to the fileserver. They also support retransfer and reconversion when the programs require modification with the programming tools (LAD, CSF and STL).

SEND: Transfer domains from the PG to a fileserver.

Specifying the name of the S5 program file on the PG determines the domain to be sent.

The destination is the file name specified using the conventions of the fileserver. The link to the destination is established by selecting the destination station in the data link file for the fileserver.

Domain-specific variables that can be defined with the PG load system are part of the load file. You can define variables using this screen by calling the variables editor for domain-specific variables. The variables are stored in a file whose name is derived from the name of the S5 program file: <xxxxxxST.VAR>

FETCH: Transfer domains from fileserver to PG

Specifying the server file determines the domain to be transferred with fetch function. the destination is the local S5 program file.

Domain-specific variables, which can also be defined with the PGLOAD program, are automatically transferred along with the domain and reconverted into the file <xxxxxxST.VAR>.

The input screen for the transfer functions:

PGLOAD		CP 143 (EXIT)	
Transfer Functions		SINEC NCM	
DEST STATION (FILESERVER) :			
SERVER FILE :			
LOCAL S5 PROGRAM FILE	DR:	NAME:	ST.S5D
COMMENT :			
F	F	F	F
1	2 VARIABLES	3 SEND	4 CREATE
F	F	F	F
5 FETCH	6 CR S5FILE	7 SERVER	8 SELECT
F	F	F	F
			HELP

Fig. 9.7 Transfer Functions for Load Files Screen

Input fields::

DEST. STATION (FILESERVER): In this case, the logical partner name of the last selected fileserver application association is displayed.

With F7, (server) you can page through the data link file and select a new partner = destination station.

SERVER FILE : File under which the load file will be stored (in the syntax of the destination system) on the server (destination) station.
Range of values: max. 64 characters

LOCAL S5 PROGRAM FILE: Name of the program file from which the load file will be generated.

Comment : Freely selectable character string, e.g. for documentation or for administration functions.
Range of values: max. 128 characters

Function keys:

F1
VARIABLES

Analogous to the COM function (VMD variables editor), you can define domain-specific variables, to be stored in the variables file.

M 2-5.1

The screen PGLOAD DOMAIN SPECIFIC VARIABLES is displayed. In terms of use and possible inputs, the screen corresponds to the COM screen M2-5.1 (see accompanying supplement with COM 143 TF screens) and Chapter 8 'Configuring VMD-specific variables'.

F3
SEND

Transfer load file to the specified server.

F4
CREATE

The load file is created from the specified local "S5 file" and from the variables file (if it exists).

F5
FETCH

The specified load file is fetched from the fileserver or from the PLC and stored on the PG under the name specified for the local S5 file.

F6
CR S5FILE

An S5 program file is created from a load file and any existing variables are converted to a variables file.

F7
SERVER

With this function, the servers entered in the data link file (fileserver) are displayed. The selected server is displayed in the DESTINATION STATION output field.

9.4.5 Using Host Computer Functions

Whereas the functions introduced so far (link configuration and transfer functions) are used to prepare the system and the PG, the host functions support the automation system during operation.

From the initial screen form you call the host computer functions by

PGLOAD		CP 143 (EXIT)	
Host Computer Functions		SINEC NCM	
STATION (PLC): <input type="text"/>			
F1 : TRANSFER DOMAIN FROM FILESERVER TO PLC			
F2 : SAVE PLC DOMAIN ON FILESERVER			
F3 : DELETE DOMAIN IN PLC			
F4 : START/STOP PLC (CREATE/DELETE PROGRAM INVOCATION)			
F5 : START APPLICATION			
F6 : STOP APPLICATION			
F7 : SELECT DEST STATION (PLC)			
NAME OF APPLICATION (PROGRAM INVOCATION NAME):			
<input type="text"/>			
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7	8
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
			HELP
			SELECT

Fig. 9.8 Input Screen - Host Functions

selecting PGLOAD | Host computer and the screen below is displayed

Input fields

STATION (PLC): The destination station (PLC) to which the following services refer. You can select a station with F7.
Range of values: max. 32 characters

PI name: Name to which services selected with the keys F4 to F7 refer:

- create program invocation,
- delete program invocation,
- start program

Range of values: max. 32 characters

Along with user PI on the PLC, which may or may not exist, there is also a system PI. This system PI can be controlled with the PI name PLC_START_STOP.

Function keys:

F1
LOAD

Load a PLC (-> next screen PGLOAD Load PLC)

F2
SAVE

Save the domains of a PLC (-> next screen PGLOAD Save PLC)

F3
DELETE

Delete all the domains in the PLC (confirmation prompted)

F4
PLC START/
RESET

Create/delete a program invocation. If you select the system PI this means start/stop the PLC.

F5
PRG START

The program invocation with the specified name is started on the PLC from the existing domains.

F6
PRG STOP

Providing a program invocation with the specified name exists in the PLC, it is stopped (if possible).

Note:

Only one (application-specific) program invocation can exist at any one time on the PLC.

F7
STATION

With this function key you can page through the PLC data link file to find a required destination station.

Errors and messages:

> "Program invocation does not exist":

the PI specified in the PLC does not exist.

> "A program invocation exists already":

when starting a program, there is already a PI in the PLC.

> "No program invocation exists":

when stopping a program, there is no PI.

> "No domain(s) loaded":

when starting a program there are no domains loaded in the PLC.

> "Function impossible: domain status = ?":

when starting a program or deleting the PLC, the domains are not in the correct status.

9.4.5.1 Load PLC

One or a maximum of two domains in a screen can be transferred from the fileserver to a CPU of the PLC (the fileserver can also be the local PG).

PGLOAD Load PLC									
PLC LINK	<input type="text"/>								
SERVER AA	<input type="text"/>								
PROGRAM (DOMAIN)	<input type="text"/>								
STORED IN FILE :	<input type="text"/>								
PARAMETERS (DOMAIN)	<input type="text"/>								
STORED IN FILE :	<input type="text"/>								
CPU NO	<input type="text" value="1"/>								
F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> F <input type="text"/> HELP									
1	2	3	4	5	6	7	OK	8	SELECT

Fig. 9.9 Load PLC Screen

Output fields:

PLC LINK: Name of the link to the PLC to be loaded with domains. The displayed PLC was selected in the basic screen form for host computer functions).
Range of values: (32 characters)

Input fields:

SERVER AA: Application association name of the server from which the domains will be loaded, (must be configured on the CP).
Range of values: 32 characters

Situation a: third-party configuration

If the link name of the server is not "PG", a PLC (destination station) is instructed to load the domains from the fileserver.

Situation b: PG = fileserver

If, however, the link name of the server is "PG", the domains are transferred directly from the PG to the PLC = destination station.

Note:

(The name of the file is then <xxxxxx>ST.S5D).

PROGRAM (DOMAIN):	Name of the program domain to be loaded. Range of values: 32 characters:
STORED IN FILE:	Name of the server file in the syntax of the server system. Range of values: 64 characters:
PARAMETER (DOMAIN):	Name of the data domain to be loaded. Range of values: 32 characters:
STORED IN FILE:	Name of the server file in the syntax of the server system. Range of values: 64 characters:
CPU NO:	With the CPU number, you specify the CPU in the programmable controller in which the domains will be loaded.

In single processor operation: the value is always 1

In the multiprocessor mode this is the slot number of the CPU (1 to 4).



The load file must be generated explicitly.

If a loadable file (xxxxST.DOM) was created from an S5 file (xxxxST.S5D) then this loadable file is loaded on the PLC. If you make changes in the S5 file, and if no loadable file is regenerated, these changes are ignored when the PLC is loaded.



The standard function blocks integrated in the operating system (e.g. send and receive blocks in the S5-115U) must not be included in the load file.

Softkey:



The specified domains are loaded from the server into the destination station via the specified PLC configuration. The PLC configuration is entered in the capabilities list of the "load domain content" job and is interpreted there by the CP. During a job, a message is displayed to indicate which job is currently being processed.

9.4.5.2 Save PLC

A PLC domain can be saved on the fileserver. This means that the PG instructs the PLC to send the data to the server. The link name is handled in the same way as in the load PLC function.

PGLOAD		CP 143 (EXIT)	
Save PLC		SINEC NCM	
PLC LINK	<input type="text"/>		
SERVER AA	<input type="text"/>		
FOLLOWING DOMAINS EXIST IN THE DEST STATION :			
DOMAIN NAME	<input type="text"/>		
SAVE IN FILE :	<input type="text"/>		
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1 +1	2 -1	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7 OK	8 SELECT

Fig. 9.10 Save PLC Screen

Output fields:

PLC LINK: Name of the link to the PLC from which the domains will be saved. The displayed PLC was selected in the basic screen form for host computer functions).
Range of values: (32 characters)

DOMAIN NAME: Name of a domain existing on the destination station. The domain displayed here can be saved. You select a on the destination station with function keys F1 or F2.

Input fields:

SERVER AA : Name of the application association to the server on which the domain will be saved.
(The link must be configured on the CP)
Range of values: 32 characters

Situation a: third-party configuration

If the application association name of the server is not "PG", a PLC (destination station) is instructed to load the domains from the fileserver.

Situation b: PG = fileserver

If, however, the link name of the server is "PG", the domains are transferred directly from the PG to the PLC = destination station.

(The name of the file is then <xxxxxx>ST.S5D).

Note:

The PLC configuration (CPU number) is omitted, since the CP records the configuration when the PLC is loaded. The PLC's domain must then be saved using the same capability.

SAVE IN FILE : Name of the file in the syntax of the server system. If the file is to be stored on the PG, the name is <xxxxxxST.S5D> and an S5-DOS file is automatically generated.
Range of values: (64 characters)



If a domain was loaded on the CPU via the CP 143 using Load PLC, and then further blocks added there using LAD/CSF/STL, these additional blocks are not saved with the Save PLC function.

Function keys:

F1 +1

Selects the next domain found in the destination station.

F1 -1:

Selects the previous domain found in the destination station.

F7 OK

The specified domains are transferred from the source station (PLC) to the server.□

Notes

10 Request Editor User-Friendly User Interface for Generating Job Buffers

The Request Editor tool supports you when creating job buffers required for programming TF communications services on your SIMATIC programmable logic controller.

This chapter explains the range of functions of the tool and how to use it. It is intended for first-time users and as a source of reference when configuring the various TF services.

The chapter contains the following information:

- The structure of job buffers
- The structure of the tool
- The steps necessary for creating and documenting job buffers
- How to make modifications
- The layout of the screens for the job buffers of the individual TF services.

10.1 Overview

10.1.1 Mode of Operation and Requirements

Creating job buffers

With the graphics-oriented user interface of the Request Editor, (similar to a control system flowchart), you enter the communications parameters in the job buffers for the individual services. The tool enters these job buffers in the selectable data blocks. Using the transfer functions, you can then load these data blocks on the PLC. The transfer functions are part of S5DOS-KOMI.

Integration in SINEC NCM

The tool runs as a component of the COM 143 TF configuration environment under SINEC NCM. With its user interface resembling a control system flowchart, the parameters of the individual services are entered and the corresponding job buffers stored in a data block of an S5 program file.

NCM conventions

When using this tool, the general rules for handling the NCM user interface apply.

The most important of these rules are as follows:

- > The screen is divided into a screen header, dialog area, message line and function key menu.
- > The functions of the tool can be called using the COM 143 TF menu bar.
- > Help texts can be called using the help key or function key ->F8.
- > Input options can be displayed using the selection key F8.

A first impression

To gain a first impression, call the tool on your PG and select the available functions one after the other.

10.1.2 Meaning of the Job Buffer

Requesting communications services

Job buffers are used in the PLC to describe a communications service requested by a PLC program. In its communications job, the PLC program refers to the data block containing the job buffers. To specify the job, the program also refers to the data word within the data block at which the required job buffer is located.

Referencing the job buffers when programming the PLC

When configuring the job buffers, the Request Editor specifies the parameters required in the PLC program to formulate the communications job. This information can be documented and is therefore available for programming the PLC.

Explanation of the structure of the data block with job buffers

To help you understand the inputs you can make using the tool, the basic structure of a data block and the job buffers it contains is explained below.

The following aspects are important:

1. The length of each job buffer is store in words before the job buffer.
2. Each job buffer begins with a fixed structure
 - Opcode (2 words, 4 characters)
 - Timeout (1 word, 16 bit fixed point), possibly reserved
 - Reserve (1 word)
3. The structure following this is adapted dynamically to the type. The length of the structure depends both on the type and the parameters.
4. No job buffer can exceed the maximum length of 256 bytes.

Overview of the structure

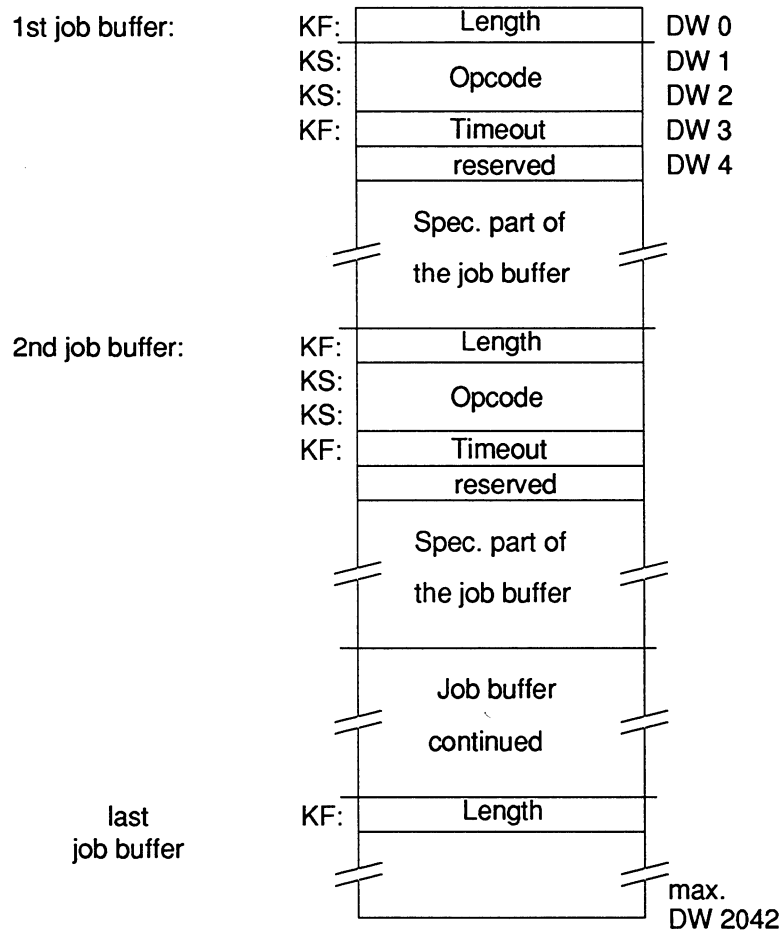


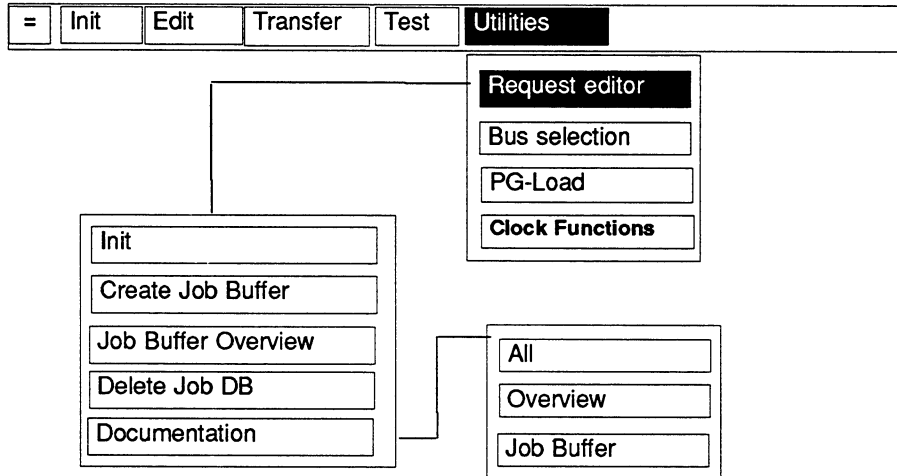
Fig. 10.1 Structure of a Data Block with Job Buffers

Number of buffers

A data block can contain several job buffers. The number of job buffers in the DB is limited by its length (2042 words). The typical length of a job buffer is 10 to 20 words.

10.2 Description of the Request Editor

The procedures supported by the Request Editor tool can be seen in the menu structure.



With the menu items, you can obtain the following functions::

Init

You select an S5 program file and a data block to which the job buffers you then edit are assigned.

Create Job Buffer

You create the job buffers using the type selection screen.

job Buffer Overview

This function provides you with a general overview of the job buffers configured in the program file.

Delete Job DB

The configured data block can be deleted.

Documentation

The configuration data are displayed on the screen or printed.

Using the Request Editor

To create job buffers with the Request Editor tool, follow the sequence outlined below:

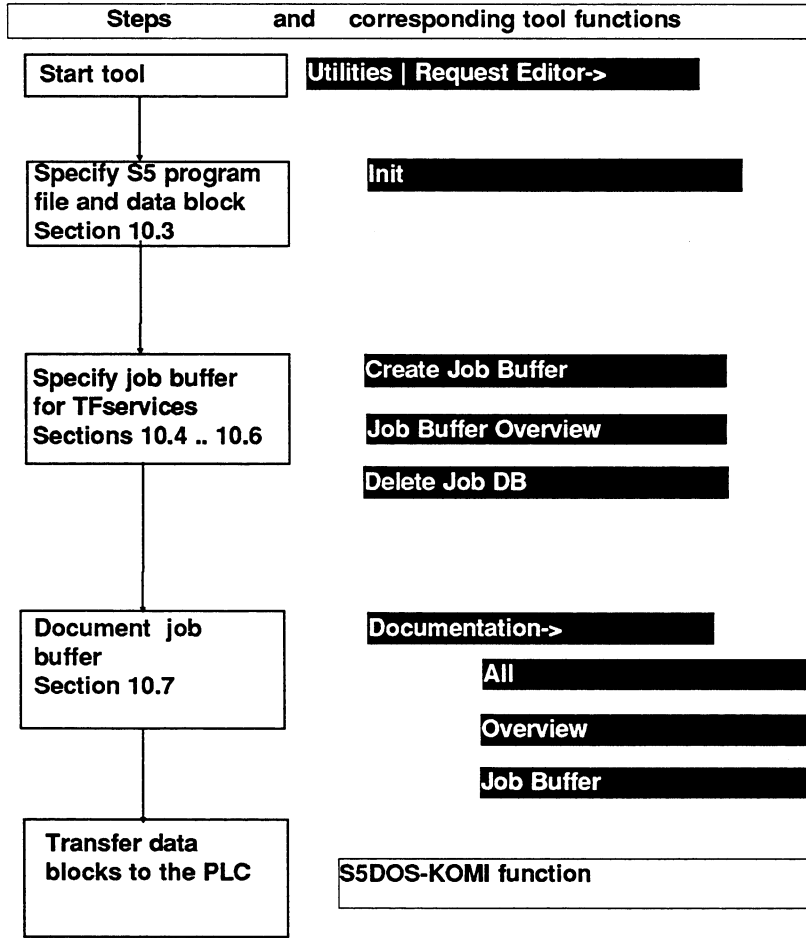


Fig. 10.2: Configuring TF job buffers

10.3 Request Editor | Init

The first step is to select an S5 program file and a data block using this function. All the job buffers you then define are assigned to this data block.

The initialization screen has the following layout:

Request Editor Initialization							CP TYPE: <input type="text"/>										
PROGRAM FILE		<input type="text"/>	ST.S5D														
BLOCK		<input type="text"/>															
F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>
1	2	3	4	5	6	7	OK	8	SELECTION								

Fig. 10.3: Request Editor initialization screen

Input fields:

PROGRAM FILE: Specifies the S5 program file to which the job buffer will be assigned. If the file does not exist it is created. If the specified file is read-only, an appropriate message is displayed in the message line. In this case, no new job buffers can be edited, but only existing buffers output.

BLOCK: 1st input field:
Specifies the type of block containing the job buffer or that will contain the job buffer.
Possible values: DB, DX
(In the following descriptions, both block types are simply described as "data blocks".
Default: DB

2nd input field:
Number of the data block containing the job buffer or that will contain the job buffer. If the data block does not yet exist in the program file, it is created. In this case the following message is displayed in the message line:
BLOCK DOES NOT EXIST

Neither a data block preheader nor comment block is created or managed.

Function keys (extra or with special meaning):

F7
OK

Enters the data you have input.

10.4 Specifying the Job Buffers for TF Services

10.4.1 Create Job buffer

Depending on the status of the program file you selected in the initialization screen, you obtain either an empty screen or the input screen of an existing job buffer.

The input screen appears as follows:

Request Editor		CP TYPE: <input type="text"/>	
		Source: <input type="text"/> ST.S5D <input type="text"/>	
<p><i>1st job buffer in selected DB is displayed if it exists</i></p> <p><i>Layout depends on the type of job buffer</i></p>			
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1 +1	2 -1	3 NEW	4 EDIT
		5 DELETE	6 COMP
		7 OK	8 SELECT

Fig. 10.4 Input Screen

If there is no job buffer in the selected DB, the following message is displayed in the message line:
NO JOB BUFFER EXISTS. In this case, select the function key F3 (NEW) to input a job buffer

If the data block exists, but does not contain a job buffer, the following message is output:
DATA BLOCK INCORRECT, DELETE?

Function keys (extra or with special meaning):

F1 +1

Finds the next job buffer in the data block and displays it.

F2 -1

Finds and displays the previous job buffer.

F3 NEW

Input of a new job buffer at the end of the current data block.

Next screen form: type selection (see next section).

If the selected data block cannot accept any more job buffers, but sufficient space would result from compressing the block (see below), the following message appears: BLOCK TOO LONG, FIRST COMPRESS

If compressing the block would still not provide sufficient space for a further job buffer, the following message is displayed: BLOCK TOO LONG, COMPRESSING NO HELP

F4 EDIT

Allows you to modify an existing job buffer. The original job buffer is deleted automatically and a new buffer appended to the end of the block.



This changes the call parameters of the "SEND DIR" for triggering the service.

F5 DELETE

Deletes the current job buffer from the data block. To prevent the remaining job buffers in the data block from being shifted together, the job buffer is not deleted but declared invalid, it can nevertheless no longer be restored.

To prevent you accidentally deleting a job buffer, you must confirm the prompt: delete (YES/NO).

You remain in the input screen form.

F6 COMP

Compresses the selected data block. This means that all invalid job buffers are removed and the valid job buffers are shifted together. The following message then appears in the message line:

CAUTION: X-REF (start address of job buffer) WILL CHANGE, PLEASE CONFIRM (xxx BYTES FREE)

xxx indicates the number of free bytes in the data block. To prevent accidental compressing of the data block, you must confirm your intention. On completion of the function, the following text appears in the message line:

COMPRESSING ENDED, xxx BYTES FREE

If the data block does not contain any invalid job buffers, the following text appears in the message line:

NO INVALID JOB BUFFER EXISTS, xxx BYTES FREE

If there are one or more invalid job buffers in the data block, but compressing does not mean that further job buffers can be accepted, the following text appears in the message line:

COMPRESSING NO HELP, GO AHEAD OR NOT?

Following this, you can decide whether you want to compress the data block or abort the function.

F7 OK

Completes the entry of new job buffers and writes the data block back to the program file.

10.4.2 Type Selection Screen Form for TF and Other Services

The type selection screen provides you with an overview of the job types available for communication.

This screen is displayed by selecting Request Editor | Create job buffer and pressing the NEW function key.

Request Editor Service Selection		CP TYPE: <input type="text"/>
		Source: <input type="text"/> ST.S5D <input type="text"/>
READ VARIABLE		STATUS
WRITE VARIABLE		UNSOL. STATUS
INFORMATION REPORT		IDENTIFY VMD
LOAD DOMAIN CONTENT		READ BYTE STRING
STORE DOMAIN CONTENT		WRITE BYTE STRING
DELETE DOMAIN CONTENT		REQUEST BYTE STRING LENGTH
GET DOMAIN ATTRIBUTES		TRANSPARENT DATA EXCHANGE
CREATE PI		CONFIGURE ANZW (local)
START PI		
RESUME PI		
STOP PI		
RESET PI		FREE FORMAT WRITE
KILL PI		FREE FORMAT READ
DELETE PI		
LOCAL PROGRAM STOP		
GET PI ATTRIBUTES		

F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4	5	6	7	8

Fig. 10.5: Type Selection Screen

Select the TF service you require or the type of function by positioning the cursor on the appropriate field and pressing return or by clicking on the field with the mouse. The selected field is displayed on a gray background. Press the OK key to confirm your selection and you then move on to the specific screen for entering a job buffer.

Function keys (extra or with special meaning):

F7	generates a job buffer for the currently selected function
OK	Next screen: depends on the function selected

The following TF services and job types are available

Variable services

READ VARIABLE	Read the current value of a variable from another station.
WRITE VARIABLE	Transfer the current value of a variable to another station.
INFORMATION REPORT	Spontaneous transmission of the current value of a variable to another station without being triggered by the other station and without being acknowledged by the other station.

At the end of the list in the screen:

FREE FORMAT WRITE	Transfer the current value of a variable to another station identified by the free format address.
FREE FORMAT READ	Read the current value of a variable identified by the free format address.

Domain services

LOAD DOMAIN CONTENT	Load a program or parameters of a program in the local CPU or in a remote CPU from the fileserver.
STORE DOMAIN CONTENT	Save a program or parameters of a program of the local CPU or a remote CPU on a fileserver (also known as upload domain)
DELETE DOMAIN CONTENT	Delete a program or parameters of a program in the local or remote CPU.
GET DOMAIN-ATTRIBUTES	Request the attributes of a program or of parameters of a program from another CPU.

Program invocation services

CREATE PI	Assignment of one or more domains to form an executable program invocation in the local or remote station.
START PI	The previously created program invocation is changed to the "RUN" status, i.e. the program now runs in the local or remote CPU
STOP PI	The previously started program invocation is stopped again in the local or remote CPU.
RESUME PI	The previously stopped program invocation is started again in the local or remote CPU.
RESET PI	The previously stopped program invocation is changed to the deletable status in the local or remote CPU. a) if PI re-usable -> to IDLE state b) if PI not re-usable -> to unrunnable state
KILL PI	Instant termination of a program invocation in the local or remote CPU.
DELETE PI	A previously reset program invocation is deleted in the local or remote CPU.
LOCAL PROGRAM STOP	The local program invocation is stopped by the user program (transition from running to stopped).
GET PI ATTRIBUTES	Request the attributes of a program invocation from a different CPU.

Selectable TF Services (continued)

VMD services

STATUS	Request the status (physical and logical) of another CPU.
UNSOLICITED STATUS	Send the local station information (physical and logical) spontaneously to another station without being triggered externally and without requesting acknowledgement.
IDENTIFY VMD	Request information about the type and characteristics of a remote station.

Transparent data exchange (non-open services)

READ BYTE STRING	Read a byte string from another station.
WRITE BYTE STRING	Transfer a byte string to another station.
REQUEST BYTE STRING LENGTH	Request the number of bytes accepted by the partner during the last write byte string job to be transferred to the PLC (local job).
TRANSPARENT DATA EXCHANGE	Trigger the non-open TF service "transparent data exchange".

Other jobs

CONFIGURE ANZW (local)	Transfers the configuration parameters for an application association to the local network interface.
------------------------	---

Select a function with the cursor keys, the currently selected function is displayed inversely on the screen.

Default: READ VARIABLE.

10.4.3 Variable Services

Read Variable

Request Editor Job Buffer		CP TYPE:	<input type="text"/>
		Source:	<input type="text"/> ST.SSD <input type="text"/>
TIMEOUT	<input type="text" value="100"/>	READ	
S5 DEST ADD	<input type="text"/>		
SCOPE	<input type="text" value="VM"/>		
VAR ID	<input type="text"/>		
DOM ID	<input type="text"/>		
VAR TYPE	<input type="text" value="IN"/> <input type="text" value="16"/>		
NUMBER	<input type="text" value="1"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP:	DB-NR:	Q-ANF:	Q-LAE:
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
5	6	7 OK	8 SELECT

Fig. 10.6: Read Variable Screen

Input fields

TIMEOUT

Acknowledgement monitoring time for the job in units of 0.1 sec. Default: 10 sec. If the job is not completed within this time, the CP aborts the job. If you do not enter a value in the field, the CP assumes that no time monitoring is required for the job.

S5 DESTINATION ADDRESS

The address in the S5 system at which the value of the requested variable will be stored by the CP.

Dest. type: DB, DX
 DB no.: 1..255
 Start: 0..2042

Example: DB 12 0

If you specify DB no. = 0, this means that the variable must be completely configured. This is mandatory for complex and structured variables.

The configuration is then specified during the link or VMD configuration in the REMOTE DEFINITIONS screen.

When you press F7, the Request Editor indicates that variable configuration is necessary.

<VAR NAME MUST BE CONFIGURED!>

It is not possible to specify the length, this is determined implicitly by the type of variable.

The address is input as normal in COM programming by separating the individual parameters of the S5 address with blanks.

SCOPE

Specifies the validity of the requested variable in the other system.

Permitted values:

VM: VMD-specific

The requested variable is valid throughout the other station (no restrictions).

DO: Domain-specific

The requested variable is only valid in the other station within the area specified by the "domain name".

VB: Link-specific

The requested variable is only accessible in the other station via a particular link. The link is identified by the SEND-DIR call parameters "SSNR" and "ANR" when the job buffer is transferred.

Default: VM

VAR ID:

The name of the requested variable in the other system.

DOM ID: This parameter is only specified if the "SCOPE = DO". It identifies the domain to which the variable is assigned via scope.

VARIABLE TYPE: Specifies the type of the requested variable.

1st input field:

Input of the basic type, see table.

2nd input field:

Length of the type specified in the first input field (specification as in COM 143).

Default: IN 16

Note:

It is not possible to access complex or structured variables (structures or arrays) using the Request Editor. In this case, you must use the "remote definitions" of variables.

NUMBER: Number of elements in arrays
Default: 1 (no array)

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

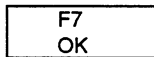
Read Variable (continued)

The table lists the types that can be specified under variable type:

TF TYPE	TYPE Description	Meaning	Corr. to S5Type
BO	No entry	Boolean	-
IN	8 16 32	Integer, 8 bits Integer, 16 bits Integer, 32 bits	- KF -
UN	8 16 32	Unsigned, 8 bits Unsigned, 16 bits Unsigned, 32 bits	- KH -
FP	32	Floating point number in MC5 format, 32 bits	KG
TI	4	Time, 4 bytes format see below	-
TD	6	Time with date format see below	-

Table 10.4: TF Type and Meaning

Function keys (extra or with special meaning):



Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Example of using a "HELP" menu

This section is intended as a reminder of the help and selection functions which make configuration easier and ensure the correctness of entries.

To illustrate these functions, it is assumed that you want a job buffer for the read variable service.

You want to make an entry in the "VAR TYPE" input field and you cannot remember the abbreviations for the types (press the SELECT key on the PG). By moving the inversely displayed line in the help window with the cursor up and cursor down keys, you can select the required type and enter it in the field by pressing the enter key or carriage return key. The help window then disappears.

For the second input field specifying the type, a selection window is displayed to help you make the input, only the options allowed for the selection made in the first window are displayed.

Write Variable

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
		WRITE	
TIMEOUT	<input type="text" value="100"/>		
S5 SOURCE ADD	<input type="text"/>		
SCOPE	<input type="text" value="VM"/>		
VAR ID	<input type="text"/>		
DOM ID	<input type="text"/>		
VAR TYPE	<input type="text" value="IN"/> <input type="text" value="16"/>		
NUMBER	<input type="text" value="1"/>		
S5 ADDRESS OF THE VARIABLE			
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7 OK	8 SELECT

Fig. 10.7: Write Variable Screen

Input fields:

TIMEOUT: Same as for read variable.

S5 SOURCE ADDRESS Address in the S5 system at which the user program has stored the value of the variable to be sent.
 Source type: DB, DX, DA
 DB no.: 1..255
 Start: 0..2042

Example: DB 12 0

If you specify DB no. = 0, this means that the variable must be completely configured. This is mandatory for complex and structured variables.

The configuration is then specified during the link or VMD configuration in the REMOTE DEFINITIONS screen.

When you press F7, the Request Editor indicates that variable configuration is necessary.

<VAR NAME MUST BE CONFIGURED!>

It is not possible to specify the length, this is determined implicitly by the type of variable.

SCOPE

Specifies the validity of the requested variable in the other system.

Permitted values:

VM: VMD-specific

The requested variable is valid throughout the other station (no restrictions).

DO: Domain-specific

The requested variable is only valid in the other station within the area specified by the "domain name".

VB: Link-specific

The requested variable is only accessible in the other station via a particular link. The link is identified by the SEND-DIR call parameters "SSNR" and "ANR" when the job buffer is transferred.

Default: VM

VAR ID:

The name of the requested variable in the other system.

DOM ID:

This parameter is only specified if the "SCOPE = DO". It identifies the domain to which the variable is assigned via scope.

VARIABLE TYPE: Specifies the type of the requested variable.

1st input field:

Input of the basic type, see table.

2nd input field:

Length of the type specified in the first input field (specification as in COM 143).

Default: IN 16

Note:

It is not possible to access complex or structured variables (structures or arrays) using the Request Editor. In this case, you must use the "remote definitions" of variables.

NUMBER: Number of elements in arrays
Default: 1 (no array)

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Note on the source type "DA":

This means that the user program stores the value of the variables after the parameters. In this case, the parameters "DB no" and "start" are invalid. Since the job buffers have different lengths depending on the length of the variables and domain names, the complete S5 address of the variables is output after you press F7 (OK). (Output field "S5 ADDRESS OF VARIABLES").

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen form: input.

Information Report

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
SCOPE	<input type="text" value="VM"/>	REPORT	
VAR ID	<input type="text"/>		
DOM ID	<input type="text"/>		
MULTIPLE ACCESS	<input type="text" value="N"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7 OK	8 SELECT

Fig. 10.8: Information Report Screen

Input fields:

SCOPE Specifies the validity of the requested variable in the local system.

Permitted values:

VM: VMD-specific

The requested variable is valid throughout the local station (no restrictions).

DO: Domain-specific

The requested variable is only valid in the local station within the area specified by the "domain name".

VB: Link-specific

The requested variable is only accessible in the local station via a particular link. The link is identified by

the SEND-DIR call parameters "SSNR" and "ANR" when the job buffer is transferred.

Default: VM

VAR ID Name of the variable to be reported in the local system.
If multiple access = YES, the group name is displayed here.

DOM ID The parameter is only specified when "SCOPE = DO". It indicates the domain to which the variable is assigned via the scope.

Multiple access If you specify "yes" here, several variables are sent in a message. In this case the variable group must be configured in the COM in the "Group Definitions" screen (VMD variables editor).

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7 OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Free Format Write

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
		WRITE	
TIMEOUT	<input type="text" value="100"/>		
S5 SOURCE ADDRESS	<input type="text"/>		
SCOPE	<input type="text" value="VM"/>		
VAR TYPE	<input type="text" value="OS 16"/>		
ADDRESS			
S5 ADDRESS OF THE VARIABLE			
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP : DB	DB-NR:	Q-ANF:	Q-LAE:
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7	8
		OK	SELECT

Fig. 10.9: Write Free Format Screen

Input fields:

TIMEOUT Same as for read variable.

S5 SOURCE ADDRESS Address in the S5 system at which the user program has stored the value of the variable to be sent.

Source type:DB, DX, DA
 DB no.:1-255
 Start 0...2042

Write Free Format (continued)

SCOPE Specifies the area in which the variable to be written is valid in the other system.

Permitted value:

VM:

The requested variable is valid throughout the whole station (no restrictions).

VARIABLE TYPE Specifies the type of the requested variable.

1st input field
specifies the basic type
always "OS"(octet string)

2nd input field
length of the type specified in the 1st input field
"OS" 1...4086 octet string, length = number of bytes
in user data string

ADDRESS You can define a maximum of 32 characters for a free format address. The entry is made in hexadecimal (see also function description of free-format write).

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Free Format Read

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>
		READ
TIMEOUT	<input type="text" value="100"/>	
S5 DEST ADD	<input type="text"/>	
SCOPE	<input type="text" value="VM"/>	
VAR TYPE	<input type="text" value="OS 8"/>	
ADDRESS		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP : DB	DB-NR:	Q-ANF:
		Q-LAE:
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3
4	5	6
7	OK	8 SELECT

Fig. 10.10: Read Free Format Variable Screen

Input fields:

TIMEOUT	Same as for read variable.
S5 SOURCE ADDRESS	Address in the S5 system at which the value of the variable will be stored.
	Source type:DB, DX DB no.: 1-255 Start 0...2042.
SCOPE	Specifies the area in which the requested variable is valid in the other system.
	Permitted value: VM: The requested variable is valid throughout the station (no restriction).

Read Free Format (continued)

VARIABLE TYPE	Specifies the type of the requested variable. 1st input field specifies the basic type always "OS"(octet string) 2 input field length of the type specified in the 1st input field "OS" 1...4086 octet string, length = number of bytes in string
ADDRESS	You can define a maximum of 32 characters for a format-free address. The entry is made in hexadecimal (see also function description free format write).

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

10.4.4 Domain Services

Load Domain

Request Editor Job Buffer
CP TYPE:

Source:
ST.SSD

TIMEOUT

DOM ID

FILE SERVER

FILE NAME

CPU NO
IN THE OTHER STATION

PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE

Q-TYP	DB-NR	Q-ANF	Q-LAE
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
1	2	3	4

LOAD

F
F
F
F
F
F
F

7 OK
8 SELECT

Fig. 10.11: Load Domain Content Screen

Input fields:

TIMEOUT Acknowledgement monitoring time for the job in units of 0.1 sec. Default: 10 sec. If the job is not completed within this time, the CP aborts the job. If you do not enter a value in the field, the CP assumes that no time monitoring is required for the job.

DOM ID The domain (group of blocks) is managed on the destination system using this name.

FILE SERVER Specifies the name of the application association to be established to the fileserver.

Load Domain Content (continued)

FILE NAME The file specified here contains the domain to be loaded.

**CPU NO IN THE
OTHER STATION:** Specifies the number of the CPU in which the domain will be loaded in the other station (SIMATIC S5)
Permitted values: 1 to 4.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7 OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Store Domain Content

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	STORE
DOM ID	<input type="text"/>	
FILE SERVICE	<input type="text"/>	
FILE NAME	<input type="text"/>	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF
		Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3
4	5	6
7	OK	8
		SELECT

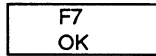
Fig. 10.12: Store Domain Content Screen

Input fields:

TIMEOUT	Same as for load domain
DOM ID	The domain specified here will be saved on the fileserver.
FILE SERVER	Specifies the name of the application association to be established to the fileserver.
FILE NAME	The domain will be stored on the fileserver under the name specified here.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Delete Domain

Request Editor Job Buffer		CP TYPE: <input type="text"/>
		Source: <input type="text"/> ST.S5D <input type="text"/>
TIMEOUT	<input type="text" value="100"/>	DELETE
DOM ID	<input type="text"/>	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3
4	5	6
7	OK	8 SELECT

Fig. 10.13: Delete Domain Content-Screen

Input fields:

TIMEOUT Same as for load domain

DOM ID The domain specified here will be deleted.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Get Domain Attributes

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	DOM-ATTR
DOM ID	<input type="text"/>	
S5 DEST ADD	<input type="text"/>	
LENGTH	<input type="text"/>	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF
		Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3
4	5	6
7	OK	8
		SELECT

Fig. 10.14: Get Domain Attributes

Input fields:

- TIMEOUT** Same as for load domain
- DOM ID** The attributes of the domain specified here are requested.
- S5 DEST ADDRESS** Address in the S5 system at which the requested domain attributes will be stored by the CP.
Dest. type: DB, DX
DB no.: 1..255
Start: 0..2042.
- Length** The "length" parameter specifies how many data words can be written by the CP into the data block. The value -1 means that all data of the acknowledgement can be entered.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

10.4.5 Program invocation services

Create PI

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	CREATE PI
PI NAME	<input type="text"/>	
DOM ID 1	<input type="text"/>	
DOM ID 2	<input type="text"/>	
DOM ID 3	<input type="text"/>	
DOM ID 4	<input type="text"/>	
DOM ID 5	<input type="text"/>	
DOM ID 6	<input type="text"/>	
DOM ID 7	<input type="text"/>	
DOM ID 8	<input type="text"/>	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF
Q-LAE		
F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>
F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>
F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>
F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>
1	2	3
4	5	6
7	OK	8
		SELECT

Fig. 10.15: Create PI Screen

Input fields:

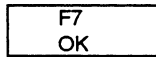
- TIMEOUT** Acknowledgement monitoring time for the job in units of 0.1 sec. Default: 10 sec. If the job is not completed within this time, the CP aborts the job. If you do not enter a value in the field, the CP assumes that no time monitoring is required for the job.
- PI NAME** The created program invocation is assigned the name specified here.
- DOM ID 1 - 8** Name of the domain belonging to the program invocation. For SIMATIC S5, up to eight domain names are possible. Input fields can, however, also remain empty. Remember that if you use longer names, the maximum length of a job buffer (256

bytes) must not be exceeded. The Request Editor monitors this and displays the message "JOB BUFFER TOO LONG!" if this limit is exceeded.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:



Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Start PI

Request Editor Job Buffer		CP TYPE: <input type="text"/>
		Source: <input type="text"/> ST.S5D <input type="text"/>
TIMEOUT	<input style="width: 50px;" type="text" value="100"/>	START PI <input style="width: 60px; height: 40px;" type="button"/>
PI NAME	<input style="width: 250px;" type="text"/>	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF
		Q-LAE
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
1	2	3
4	5	6
7	OK	8
		SELECT

Fig. 10.16: Start PI Screen

Input fields:

- TIMEOUT Same as for create PI.
- PI NAME The program invocation created with this name will be started.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

- | | |
|----------|---|
| F7
OK | Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input. |
|----------|---|

Stop PI

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.SSD <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	STOP PI	
PI NAME	<input type="text"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
1	2	3	4
5	6	7 OK	8 SELECT

Fig. 10.17: Stop PI Screen

Input fields:

TIMEOUT Same as for create PI.

PI NAME The program invocation created with this name will be stopped.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input..

Resume PI

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	RESUME PI	
PI NAME	<input type="text"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
F	<input type="text"/>	F	<input type="text"/>
1	2	3	4
5	6	7 OK	8 SELECT

Fig. 10.18: Resume PI Screen

Input fields:

TIMEOUT Same as for create PI.

PI NAME The program invocation with this name will be resumed.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Reset PI

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	RESET PI	
PI NAME	<input type="text"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7 OK	8 SELECT

Fig. 10.19: Reset PI Screen

Input fields:

TIMEOUT Same as for create PI.

PI NAME The program invocation with this name will be reset.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Kill PI

Request Editor Job Buffer

CP TYPE:
 Source: ST.S5D

TIMEOUT

PI NAME

KILL PI

PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE

Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
1	2	3	4

F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
1	2	3	4	5	6	7 OK	8 SELECT

Fig. 10.20: Kill PI Screen

Input fields:

- TIMEOUT Same as for create PI.
- PI NAME The program invocation with this name will be aborted.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

- | | |
|--|---|
| <div style="display: flex; flex-direction: column; align-items: center;"> F7 OK </div> | <p>Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.</p> |
|--|---|

Delete PI

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	DELETE PI
PI NAME	<input type="text"/>	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF
		Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3
4	5	6
7	OK	8
		SELECT

Fig. 10.21: Delete PI Screen

Input fields:

TIMEOUT Same as for create PI.

PI NAME The program invocation with this name will be deleted.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Local Program Stop

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>
PI NAME <input type="text"/>		LOC. STOP <input type="checkbox"/>
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP	DB-NR:	Q-ANF
		Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3
4	5	6
7	OK	8
		SELECT

Fig. 10.22: Local Program Stop Screen

Input fields:

PI NAME The program invocation with this name will be stopped.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Get PI Attributes

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	PI-ATTR	
PI NAME	<input type="text"/>		
S5 DEST ADD	<input type="text"/>		
LENGTH	<input type="text"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7	8
		OK	SELECT

Fig. 10.23: PI Attributes Screen

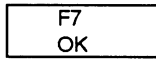
Input fields:

TIMEOUT	Same as for create PI.
PI NAME	The attributes of the program invocation specified here are requested.
S5 DESTINATION ADDRESS	Address in the S5 system at which the requested PI attributes will be stored by the CP. Dest. type: DB, DX DB no.: 1..255 Start: 0..2042
Length	The "length" parameter specifies how many data words can be written by the CP into the data block. The value -1 means that all data of the acknowledgement can be entered.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:



Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

10.4.6 VMD Services

Status

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>	
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>	
		STATUS	
TIMEOUT	<input type="text" value="100"/>	<input type="checkbox"/>	
S5 DEST ADD	<input type="text"/>		
LENGTH	<input type="text"/>		
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7	8
		OK	SELECT

Fig. 10.24: Status Screen

Input fields:

TIMEOUT

Acknowledgement monitoring time for the job in units of 0.1 sec. Default: 10 sec. If the job is not completed within this time, the CP aborts the job. If you do not enter a value in the field, the CP assumes that no time monitoring is required for the job.

S5 DESTINATION ADDRESS

Address in the S5 system at which the requested status information will be stored by the CP.

Dest. type: DB, DX

DB no.: 1..255

Start: 0..2042

Length: The "length" parameter specifies how many data words can be written by the CP into the data block. The value -1 means that all data of the acknowledgement can be entered.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.

Next screen: input.

Unsolicited Status

Request Editor Job Buffer		CP TYPE: <input type="text"/>	
		Source: <input type="text"/> ST.S5D <input type="text"/>	
UNSOL STATUS			
<input type="text"/>			
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP: DB		DB-NR: 100	
Q-ANF		Q-LAE	
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7 OK	8 SELECT

Fig. 10.25: Unsolicited Status Screen

For the "unsolicited status" service, no entry is required in the screen. The Request Editor simply generates the required buffer in the job data block and displays the address of the job buffer

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

F7 OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Identify VMD

Request Editor		CP TYPE: <input type="checkbox"/>									
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>									
TIMEOUT S5 DEST ADD LENGTH	<table style="margin: auto;"> <tr> <td style="border: 1px solid black; width: 40px; height: 15px; text-align: center;">100</td> <td style="border: 1px solid black; width: 200px; height: 15px;"></td> <td style="border: 1px solid black; width: 40px; height: 15px;"></td> </tr> <tr> <td style="border: 1px solid black; width: 40px; height: 15px;"></td> <td style="border: 1px solid black; width: 200px; height: 15px;"></td> <td style="border: 1px solid black; width: 40px; height: 15px;"></td> </tr> <tr> <td style="border: 1px solid black; width: 40px; height: 15px;"></td> <td style="border: 1px solid black; width: 200px; height: 15px;"></td> <td style="border: 1px solid black; width: 40px; height: 15px;"></td> </tr> </table>	100									IDENTIFY <div style="border: 1px solid black; width: 40px; height: 40px; margin: auto;"></div>
100											
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE											
Q-TYP	DB-NR:	Q-ANF									
Q-LAE											
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>									
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>									
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>									
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>									
1	2	3									
4	5	6									
7	OK	8 SELECT									

Fig. 10.26: Identify VMD Screen

Input fields:

- TIMEOUT** Same as for status
- S5 DESTINATION ADDRESS** Address in the S5 system at which the requested information will be stored by the CP..
 Dest. type: DB, DX
 DB no.: 1..255
 Start: 0..2042
- Length:** The "length" parameter specifies how many data words can be written by the CP into the data block. The value -1 means that all data of the acknowledgement can be entered.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:



Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

10.4.7 Transparent Data Exchange (non-open services)

Read Byte String

Request Editor Job Buffer CP TYPE:
Source: ST.SSD

TIMEOUT

S5 DEST ADD

LENGTH

100

READ BS

PARAMETERS FOR "SEND-DIR" CALL TO ACTIVATE THE SERVICE

Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
1	2	3	4

F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
5	6	7	OK	8	SELECT				

Fig. 10.27 Read BS Screen

Input fields:

TIMEOUT Acknowledgement monitoring time for the job in units of 0.1 sec. Default: 10 sec. If the job is not completed within this time, the CP aborts the job. If you do not enter a value in the field, the CP assumes that no time monitoring is required for the job.

S5 DESTINATION ADDRESS Address in the S5 system at which the requested byte string will be stored by the CP.
 Dest. type: DB, DX
 DB no.: 1..255
 Start: 0..2042

10

10 - 53

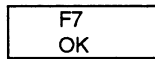
Volume 2

Length: The "length" parameter specifies how many data words can be written by the CP into the data block.
Range of values: 1 .. 2043

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:



Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

Write Byte String

Request Editor Job Buffer		CP TYPE:	<input type="checkbox"/>
		Source:	<input type="checkbox"/> ST.S5D <input type="checkbox"/>
TIMEOUT	<input type="text" value="100"/>	WRITE BS	
S5 SOURC ADD	<input type="text"/>		
LENGTH	<input type="text"/>		
ACKNOWLEDGE	<input type="text" value="YES"/>		
S5 ADDRESS OF BYTE STRING:			
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE			
Q-TYP	DB-NR:	Q-ANF	Q-LAE
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
1	2	3	4
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text"/>
5	6	7	8
F <input type="text"/>	F <input type="text"/>	F <input type="text"/>	F <input type="text" value="HELP"/>
	OK	SELECT	

Fig. 10.28 Write BS Screen

Input fields:

TIMEOUT Same as for Read byte string

S5 SOURCE ADDRESS Address in the S5 system at which the user program stored the byte string to be sent.
 Source type: DB, DX, DA
 DB no.: 1..255
 Start: 0..2042

Note on the source type "DA":
 This means that the user program stored the byte after the parameters in the job buffer. In this case, the parameters "DB no" and "Start" are invalid. If you press the softkey F7 (OK) the complete S5 address of the byte string is displayed (output field "S5 ADDRESS OF BYTE STRING").

Write Byte String (continued)

Length: The "length" parameter specifies the number of data words contained in the byte string.

ACKNOWLEDGE This input field is used to specify the service to be triggered in more detail.

Permitted values:

YES:

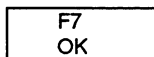
The "write byte string with acknowledgement" service is triggered.

NO:

The "write byte string without acknowledgement" service is triggered

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

Completes the input and stores the newly edited job buffer in the main memory of the programmer.

Next screen: input.

Request Byte String Length

Request Editor Job Buffer		CP TYPE: <input type="text"/>
		Source: <input type="text"/> ST.S5D <input type="text"/>
TIMEOUT	<input type="text" value="100"/>	BS LENGTH <input style="width: 100px; height: 30px;" type="text"/>
S5 DEST ADD	<input style="width: 100px;" type="text"/>	
<small>PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE</small> <small>Q-TYP DB-NR: Q-ANF Q-LAE</small>		
1	2	3
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>	F <input style="width: 40px;" type="text"/>
F		

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:



Completes the input and stores the newly edited job buffer in the main memory of the programmer.
Next screen: input.

invalid. After pressing F7 (OK), the complete S5 address of the useful data is displayed (output field "S5 ADDRESS OF SOURCE DATA").

S5 DESTINATION ADDRESS

Address in the S5 system at which the data in the acknowledgement will be stored by the CP.

Dest. type: DB, DX

DB no.: 1..255

Start: 0..2042

D length:

The "D length" parameter specifies how many data words can be written to the data block by the CP. The value -1 means that all the data of the acknowledgement can be accepted.

ACKNOWLEDGE

This input field is used to define the service to be triggered in more detail.

Permitted values:

YES:

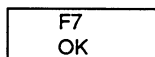
The "TRADA with acknowledgement" service is triggered.

NO:

The "TRADA without acknowledgement" service is triggered.

Output fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the screen. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Function keys:

Completes the input and stores the newly edited job buffer in the main memory of the programmer.

Next screen: input.

Function keys:

F1
ANZW

You can input the address of the status word to be addressed in client jobs on the link used with the SEND direct. The address transferred with a configuration job has priority over the address configured in the COM-application association configuration.

Next screen form: see subfunction F1

F2
S ADD R

You can input the address in the S5 system at which the data for the "read byte string" job are stored

Next screen : see subfunction F2

F3
D ADD W

You can input the address in the S5 system at which the CP stores the byte string contained in the TF-PDU in a "write byte string indication".

Next screen: see subfunction F3

F4
STATUS

You can input the address in the S5 system at which the CP will store the data for an "unsolicited VMD status indication"

Next screen: see subfunction F4

F7
OK

Completes the input and stores the newly edited job buffer in the main memory of the programmer.

Next screen: input..

Configure ANZW (continued)

Configure ANZW (local) Subfunction F1: status word for client jobs

You select the address of the status word to be addressed on the link for SEND direct in client jobs. The address transferred with a configuration job has priority over the address configured in the COM-application association configuration.

Request Editor Job Buffer CP TYPE: [] Source: [] ST.S5D []

STATUS WORD FOR CLIENT JOBS

ANZW ADD [] CONFIG

F [] F [] F [] F [] F [] F [] F [] F [] F [] HELP

1 2 3 4 5 6 7 OK 8 SELECT

Fig. 10.32: Status Word for Client Jobs Screen

Input fields:

ANZW ADD

Address of the status word to be addressed on the link for SEND direct in client jobs.

This consist of the following:

Anzw type: FW, DB, DX

Anzw no.: FW number or DB/DX number

DW no.: data word number for ANZW type = DB or DX

Example:

FW 100, if flag word 100 to 102 is to be used as the status word.

DB 20 10, if data block 20, data words 10-12 are to be used as the status word.

Function keys:

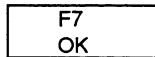
F7
OK

The parameters are entered in the job buffer. If you do not make an entry in the input field and the parameter already exists, it is deleted from the job buffer.

Next screen: configure ANZW (local).

ANZW ADD Address of the status word for such jobs. The entry is the same as for the ANZW parameter.

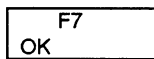
Function keys:



The parameter is entered in the job buffer. If you do not make an entry in the input field and the parameter already exists, it is deleted from the job buffer.
Next screen: configure ANZW.

ANZW ADD Address of the status word for such jobs. The entry is the same as for the ANZW parameter.

Function keys:



The parameters are entered in the job buffer. If you do not make an entry in the input field and the parameter already exists, it is deleted from the job buffer.
Next screen: configure ANZW.

Configure ANZW (local)

Subfunction F4: Destination address for unsolicited VMD status

The function is used to input the address in the S5 system at which the CP will store the data for an "unsolicited VMD status indication" You also specify the status word to be used with such jobs.

Fig. 10.35: Unsolicited VMD Status

Input fields:

S5 DESTINATION Dest. type: DB, DX
 ADDRESS DB no.: 1..255
 Start: 0..2042

ANZW ADD Address of the status word for such jobs.

Function keys:

F7
OK

The parameters are entered in the job buffer. If you do not make an entry in the input field and the parameter already exists, it is deleted from the job buffer.

10.5 Displaying and Evaluating the Job Buffer Overview

With the Request Editor, you can display a list of the job buffers contained in the selected data block. Each output line corresponds to a job buffer in the data block.

Request Editor Job Buffer		CP TYPE: <input type="checkbox"/>	
		Source: <input type="checkbox"/> ST.S5D <input type="checkbox"/>	
OPCD	S5 ADDR.JOB B.	NAME / INDEX	S5 ADDRESS
V-RE	DB 10 1 29	PRESSURE_IN_STEAM_CHAMBER	DB 31 10 2
V-WR	DB 10 31 24	SETPOINT_FOR_PRESSURE	CONFIGURED
P-ST	DB 10 56 17	PRESSURE_CONTROL_PROGRAM	

F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>	F <input type="checkbox"/>
1	+1	2	-1	3	4 SEARCH	5	6
						7 OK	8 SELECT

Fig. 10.36 Job Buffers in a Data Block

Output fields:

- OPCD** Displays the selected service contained in the job buffer. The abbreviations for the service are explained in the following table.
- S5 ADDR. JOB B.** Displays the data block number and data word number in the data block.
- NAME/INDEX** Displays a job-specific name such as variable name or program invocation name.

S5 ADDRESS Displays the source address for calling "SEND DIR" to trigger the service

The S5 address contained in the job buffer is displayed. With variables services, this is the source or destination address of the variables. Since there is no S5 address contained in certain job buffer types, this display may be omitted.

Overview of the OPCD abbreviations

OPCD	Meaning
V-RE	Read Variable
V-WR	Write Variable
V-IN	Information Report
D-LO	Load Domain Content
D-ST	Store Domain Content
D-DE	Delete Domain Content
D-GE	Get Domain Attributes
P-CR	Create Program Invocation
P-ST	Start Program Invocation
P-RE	Resume Program Invocation
P-SP	Stop Program Invocation
P-RS	Reset Program Invocation
P-AB	Kill Program Invocation
P-DE	Delete Program Invocation
P-HL	Local Program Stop
P-GE	Get Program Invocation Attributes
M-ST	Status
M-SU	Unsolicited Status
M-ID	Identify VMD
B-RQ	Read Byte String
B-WQ	Write Byte String with Acknowledgment
B-WO	Write Byte String without Acknowledgment
B-WI	Request Byte String Length
T-DQ	Transparent Data Exchange with Acknowledgment
T-DO	Transparent Data Exchange without Acknowledgment
A-CF	Configure ANZW [local]

Function keys (extra or with special meaning):

F4 SEARCH

Using the cursor keys (up/down) you can mark a job buffer in the screen (line displayed inversely) and select it with the function key F4 or F7.

F7 OK

With this function, you can select the marked job buffer for display or processing.

Explanation of the example::

1. Read the variable "PRESSURE_IN_STEAM_CHAMBER"
The variable is defined in data block 31 from data word 10 and occupies two words (the type in the job buffer is floating point). The job buffer was created in data block 10 beginning at data word 1. When this service is triggered, 29 words must be transferred to the CP.
2. Write the variable "SETPOINT_FOR_PRESSURE"
This variable must be programmed with the COM. The job buffer was created in data block 10 beginning at data word 31. When the service is triggered, 24 words must be sent to the CP.
3. Start the program "PRESSURE_CONTROL_PROGRAM"
The job buffer was created in data block 10 beginning at data word 56. When the service is triggered, 17 words must be sent to the CP. There is no S5 address in the job buffer.

10.6 Delete Data Block

You can delete a data block completely. When you use this function, all the information about the job buffers in the selected data block is lost.

10.7 Documenting Job Buffers

Using the documentation functions, you can output the configuration data either on the screen or on a printer. The output depends on the selections you make in the screen in which you specify the configuration environment.

The functions always relate to the currently selected program file and selected data block.

10.7.1 Documentation | All

Both the job buffers and the overview list are output.

10.7.2 Documentation | Overview

You obtain a printout of the overview list. The explanation of the individual fields can be found in the function description overview of job buffers.

10.7.3 Documentation | Job Buffers

The job buffers for the selected program file and the selected data block are output based on the representation on the screen..q

Notes

11 Example Programs

11.1 Overview and Requirements

Aims

This chapter is intended to familiarize you with the TF interface on the SINEC H1 bus system for SIMATIC S5. Emphasis is on the services and the parameter assignment for the CP 143 using the software package COM 143.

The aim of this chapter is to provide an overview of the services by creating a small communications system. The example leads step by step to the TF services:

- Example 1: using variable services to transfer process values to a host computer and to supply a programmable logic controller (VMD) with current control information. The function of the monitoring computer is handled by a second PLC.
- Example 2: using domain and program invocation services to adapt PLC programs to process requirements dynamically and to control the programs.
- Example 3: transparent data exchange for simple transfer of data without structural information between S5 PLCs.

Requirements

It is assumed that you are familiar with the CP handling blocks. The CP handling blocks are standard function blocks allowing the use of the communications functions of the PLC programs. The following minimum hardware must be available:

- 2 programmable logic controllers (e.g. S5 155 U) with memory and additional 15 V modules in the power supply.
- 2 CP 143 communications processors
- 2 EPROMs each with 16 K words (=32 Kbytes)
- a transmission line consisting of:
 - 2 transceivers,
 - 2 transceiver cables,
 - 2 terminators
 - 1 bus cable with coaxial connectors
- 2 programmers (e.g. PG 730/PG 750)

The following software packages are also required:

- COM 143 TF
- PG software for the STEP 5 programming language,
- handling blocks for your PLCs.
- the example files supplied with COM 143 TF.



Note that the lists of function and data blocks shown in this chapter are intended to illustrate the text. The actual values are in the example files on the diskette. Use these files to assign parameters to the PLC!

11.2 Example 1: Using Variable Services

11.2.1 Task

In this example, the TF variable services are used to exchange structured data between two stations a programmable logic controller and a monitoring computer. Two SIMATIC PLCs are used as the stations. The host operates as the client and the PLC as server.

Server PLC

The following simulation exists on the server PLC.:

There is an array of five process values, each of which is represented as a whole number (integer 16). The following process is simulated in an FB:

Each of the analog values is incremented by a fixed value at a certain time interval. When the preset upper limit is reached, each process value once again assumes the preset default value (lower limit). The graph of the process values is therefore a sawtooth function.

The variables that can be influenced by the process control include the time interval and the upper and lower limits of the analog values.

Client PLC

The client PLC, here the monitoring computer, has the following tasks:

You want to monitor and influence the process in the server from the client PLC.

The values transferred by the server are stored in a data block and can be monitored at a PG.

In addition to this, the client PLC program must be able to preset the process parameters of the server PLC (write a variable) at the request of the user (setting a bit in the flag).

In the same way, the client must be able to read the title of the simulation in the server.

These requirements produce the following sequence of events for storing

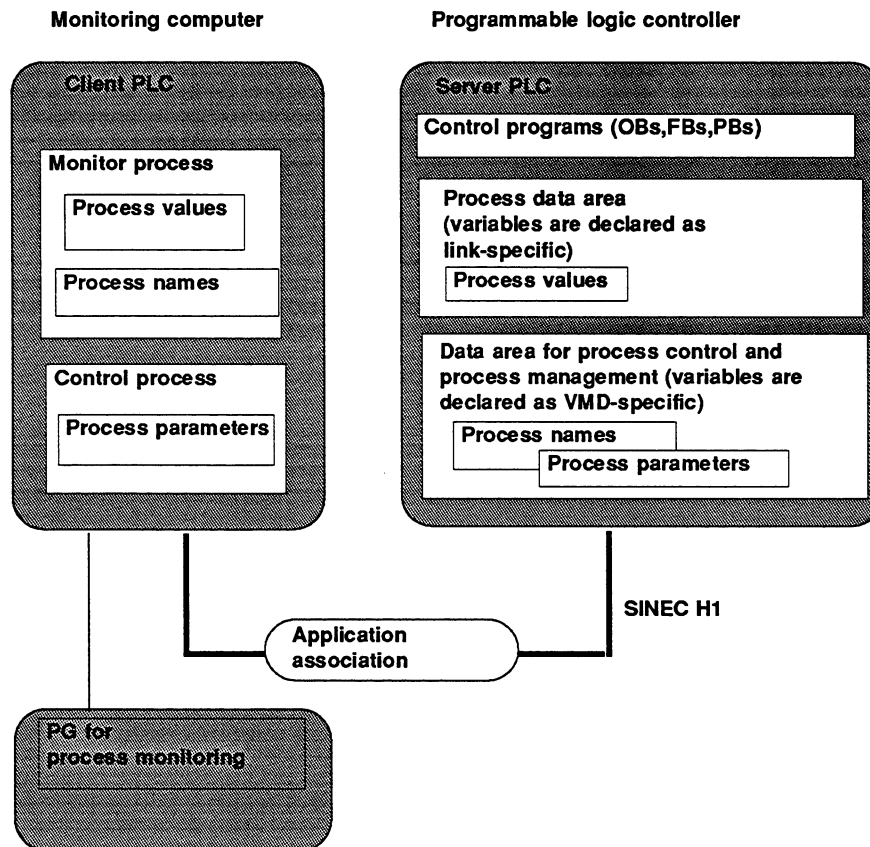


Fig. 11.1 Example Configuration for Stage 1 - Variable Services

the data and using the variable services:

Tasks of the client PLC :

- Read process name (TF service: read variable)

Task of the server PLC:

- Report process values (TF service: information report)

11.2.2 Defining Variables

Based on the tasks described above, the data definitions discussed below are necessary and must be defined and in some cases configured for TF access.

The following objects are stored on the server PLC:

- 5 PROCESS VALUES

These are of the type "integer 16", and are therefore whole numbers and are stored in the data block in 16-bit representation. The 5 process values are stored as an array of 5 whole numbers.

The scope of the process values is link-specific, so that the values can be only accessed via one link.

- PROCESS PARAMETERS

The following parameters are stored to control the process:

Speed of change (type: integer 16). This parameter specifies the rate at which the process values will change. This is the same rate for all process values. To simplify programming, the value specified for the rate of change is decremented once per PC cycle until it reaches the value zero. Following this, the process values are updated and the process cycle begins again.

Upper limits (type: array with 5 elements of type integer 16):

For each process value, an upper limit is fixed in an array of 5 elements, and must not be exceeded.

Lower limits (type: array with 5 elements of type integer 16):

An array of 5 whole numbers specifies the values at which the process values should start (default values).

This produces the following data structure:

```
PROCESS PARAMETERS      {  
RATE OF CHANGE          IN 16  
UPPER LIMITS            AR 5  
                        IN16  
LOWER LIMITS            AR 5  
                        IN16  
                        }
```

The scope of the process parameters is selected as VMD-specific, since it is assumed that the data areas for managing and controlling the PLC process are maintained as global data (VMD) on the PLC.

➤ **PROCESS NAME**

This object contains a title for the current process. The object is of the type "visible string" (i.e. "KS" for SIMATIC S5) and consists of 32 characters.

The scope of this object is also selected as VMD-specific, since the process title and the process parameters belong to the process control and process monitoring data area.

Data transfer

The current process values are to be reported to the client by the server cyclically as often as possible, so that the client always has an up-to-date process image of the server.

On the other hand, the PROCESS PARAMETERS and the PROCESS NAME are transferred on the initiative of the host.

11.2.3 TF Services Required

To transfer the data required for the tasks between the two devices, the following TF services are required:

➤ Read variable

The process name must be read by the client.

Configuration on the server:

The process name is VMD-specific and must be configured in COM 143.

Name: PROCESS NAME

Configuration on the client:

Since the process name is a simple data type (string with 32 ASCII characters), no configuration is necessary. A corresponding job buffer must simply be created for the PLC program

➤ Write variable

The process parameters must be written to the server PLC by the client.

Configuration on the server:

The process parameters are VMD-specific and must be configured in COM 143.

Name: PROCESS PARAMETERS

Configuration on the Client:

Since this variable is of a complex type (structure) and a complete description of the variables in the job buffer is not possible, this variable must be configured as a "remote object" (i.e. an object that is not defined on the local machine, but on a remote machine). This definition is made in COM 143 when defining the link via which the variable will be written.

Name: PROCESS PARAMETERS

In addition to this, a job buffer containing the name of the variable must also be defined to trigger the service in the client PLC program.

➤ Information report

The program in the server PLC reports the process values to the client.

Configuration on the Server:

Since the process values are local objects in the server with the "link-specific" scope of validity, they must be configured in COM 143 as link-specific, local objects.

To trigger the service, a job buffer must be created in the PLC program.

Name: PROCESS VALUE

Configuration on the Client:

To process the indication in the client, the process values must be configured as remote objects. These are defined when configuring the link via which the values are indicated.

Name: PROCESS VALUE

The listed services must be executed via an application association. This is established actively by the client. This means that the establishment type on the client must be "A7" and "P7" on the server.

11.2.4 Creating the Client Configuration File

Start COM 143 as described in the Section 'Introduction to the Configuration Software NCM/COM 143'. If you have worked through the example of configuring the transport interface in Volume 1, Chapter 5, you are already familiar with the next two steps in basic configuration.

Specifying the configuration environment

M 1-1

Select the function Init | Edit, to specify the configuration environment. Make the following entries or accept the proposed values if they are suitable:

CP TYPE: CP143

STATUS: OFFLINE FD

DATABASE FILENAME: PBU1PLC1.CP1

Enter your selections.

The PG automatically recognizes whether or not a file with the name "PBU1PLC1.CP1" already exists and if it does not, generates the file on the hard disk (after initializing the SYSID block, see below.).

CP basic configuration

M 2-1

The next step is to create the SYSID block. Select the function Edit | CP Basic Initialization. In the screen displayed, you will see the CP type and the name of the selected database file.

Some of the fields already have defaults entered or are purely display fields. The field "VERSION" is simply a display field. The "PASSWORD" is intended to prevent unauthorized access to a CP 143 via the SINEC H1 bus. Since the SYSID block of the module is being called for the first time, no password is entered. To simplify matters, the password field can remain empty for this example. In the field "DATE", enter the current date (free format). The base interface number "BASE SSNR" of module 1 is set to the default value "0" (set the hardware accordingly!). "MODULE SIZE" is already set to 32K BYTES. If you are using 64 Kbyte modules, this entry must be changed. The Ethernet address can be entered either in this screen or in the "CP INITIALIZATION BLOCK" screen as in this example. You can call the "CP INITIALIZATION BLOCK" screen by pressing F2 (Init).

M 2-2

In the "CP INITIALIZATION BLOCK" screen, only change the Ethernet address to 08000601B010. All the other parameters remain unchanged or empty.

Complete your entries with F7 (OK). The file PBU1PLC1.CP1 is then set up on the hard disk. This completes input of the data specific to the CP 143.

For the next step, select the Edit | Links | Appl. Associations in the selection menu.

M2-4-2.1**11****Configuring the application association**

You now specify the application association and its assignment to the transport connection. Make your entries according to the section of screen shown below.

Application association configuration	...		
SSNR : 0	ANR : 1	ANZW : FW 100	
LOCAL TSAP:-	LENGTH : 6	HEX : 43 4C 49 45 4E 54	ASC : CLIENT
EST TYPE (A4/A7/D4/D7/P4/P7)	:	A7	MUX ADDRESS: 0
APPL. ASS. NAME	:	PLC-PLC LINK	
REMOTE TSAP:-	LENGTH : 6	HEX : 53 45 52 56 45 52	ASC : SERVER
REMOTE ETHERNET ADDRESS	:	08000601B030	

Fig. 11.2 Application Association

Now press F5 (Remote) to call the screen for configuring remote variables.

M2-4-2.3**Configuring the link-specific variables**

You now configure the variables managed on the partner device which are to be accessed by the PLC configured here using write or read jobs. The variables which report information on the initiative of the server PLC must also be specified here.

Make the entries according to the section of screen shown below.

Remote Definition		...			
SCOPE	NAME	TYPE	S5 ADDRESS	ANZW	
VB	PROCESS VALUE	AR	5	DB 10 0	FW 50
		IN	16		
VM	PROCESS PARAMETERS	{	3	DB 0 0	FW 0
	RATE OF CHANGE	IN	16	DB 0 0	FW 0
	UPPER LIMITS	AR	5	DB 0 1	FW 0
		IN	16		
	LOWER LIMITS	AR	5	DB 0 6	FW 0
		IN	16		
		}			

Fig. 11.3 Remote Definition

Complete your entries with F7 (OK). You return to the application association configuration screen.

Complete your input by pressing F7 (OK). Confirm the prompt about entering the data with YES.

11.2.5 Creating the Server Configuration File

Carry out the basic configuration the same way as for the client PLC. The name of the file in this case is "PBSPL.SRV".

The Ethernet address is 08000601B030.

For this link, a local link-specific object will be defined in the next step (PROCESS VALUE).

Following this, the VMD-specific variable structure PROCESS PARAMETERS must be configured.

M2-4-2.1**Configuring an application association**

You now specify the application association and its assignment to the transport connection. Make your entries based on the section of screen shown below.

Application association configuration				...
	SSNR : 0	ANR : 1	ANZW : FW 100	
LOCAL TSAP :	LENGTH : 6	HEX : 53 45 52 56 45 52	ASC : SERVER	
EST TYPE (A4/A7/D4/D7/P4/P7)		: P7	MUX-ADDRESS: 0	
APPL. ASS. NAME		: PLC-PLC LINK		
REMOTE TSAP :	LENGTH : 6	HEX : 43 4C 49 45 4E 54	ASC : CLIENT	
REMOTE ETHERNET ADDRESS		: 0800601B010		
NUMBER OF APPL. ASS. FOR CURRENT TSAP : 1 CURRENT JOB : 1				

Fig. 11.4 Application Association

Now press F4 (LOCAL) to call the screen for configuring local variables.

M2-4-2.3**Configuring link-specific variables**

You now configure the variables managed on the server and accessed by the client with read or write jobs. The variables reported on the initiative of the server PLC must also be specified.

Make your entries based on the section of screen shown below; first for PROCESS VALUE.

Local Definition		...				
NAME	TYPE	ACC	S5 ADDRESS	ANZW	SSNR	
PROCESS VALUE	AR 5 IN 16		DB 10 0	FW 50	0	

Fig. 11.5 Local Variable Definition (VMD)

Complete your input with F7 (OK). You return to the application association configuration screen.

Complete your input with F7 (OK). Confirm the prompt about entering data with YES.

M2-5.1

Configuring VMD-specific variables

You now define the structure PROCESS PARAMETERS. This is a VMD-specific variable. You call this configuration screen using the menu bar Edit | VMD Variables. After inputting the data, complete the entry by pressing F7 (OK). Confirm the prompt about entering the data with YES.

Local Definitions						
NAME	TYPE	ACC	SS ADDRESS	ANZW	SSNR	
PROCESS PARAMETERS	{	3				
RATE OF CHANGE	IN	16	DB 10 5	FW 52	0	
UPPER LIMITS	AR	5	DB 10 5	FW 52	0	
LOWER LIMITS	IN	16	DB 10 6	FW 52	0	
	AR	5	DB 10 11	FW 52	0	
PROCESS NAME	R		DB 11 0	FW 54	0	

Automatic, after adopting inserted entries!

Fig. 11.6 Local Variable Definition - Process Parameters

11.2.6 Creating the Job Buffers with the Request Editor

Client PLC

To create the job buffers, you must start the Request Editor | Request Editor | Init.

You should first create the job buffers for the PLC program of the client. The program file in our example is BSPC@@ST.S5D. The job buffers are stored in DB 20.

Request Editor Initialization		...
PROGRAM FILE	C:	BSPC@@ST.S5D
BLOCK	DB 20	

Fig. 11.7 Request Editor - Client PLC

Creating the job buffer

First you must select the TF service to be defined in the job buffer. To do this, select Utilities | Request Editor | Create Job Buffer.

the following selection list is then displayed:

Request editor service selection		...
READ VARIABLE :	STATUS :	
WRITE VARIABLE :	UNSOLICITED STATUS :	
INFORMATION REPORT :	IDENTIFY VMD :	
LOAD DOMAIN CONTENT :	READ BYTE STRING :	
STORE DOMAIN CONTENT :	WRITE BYTE STRING :	
DELETE DOMAIN CONTENT :	REQUEST BYTE STRING LENGTH :	
GET DOMAIN ATTRIBUTES :	TRANSPARENT DATA EXCH.:	
CREATE PI :	CONFIGURE ANZW (local):	
START PI :		
STOP PI :		
RESUME PI :	FREE FORMAT WRITE	
RESET PI :	FREE FORMAT READ	
KILL PI :		
DELETE PI :		
LOCAL PROGRAM STOP :		
GET PI ATTRIBUTES :		

Fig. 11.8 Request Editor - Selection list

Select the job type 'read variable'. The appropriate screen is then displayed.

Job buffer for the TF service 'read variable'

Enter the variables as shown in the section of screen below for the variable PROCESS NAME.

Request editor		...
		READ
TIMEOUT	100	
SS DEST ADD	DB 11 0	
SCOPE	VM	
VAR NAME	PROCESS NAME	
DOM NAME		
VAR TYPE	VS 32	
NUMBER	1	
PARAMETERS FOR "SEND DIR" CALL TO ACTIVATE THE SERVICE		
Q-TYP : DB	DB-NR : 20	Q-ANF : Q-LAE:

Fig. 11.9 Job Buffer- Read Variable

After entering the data, you return to the overview by pressing F7 (OK) and F3 (NEW). Then select the job type 'write variable'.

Job buffer for the TF service 'write variable'

Enter the values according to the section of screen shown below for the variable PROCESS PARAMETERS.

Request Editor		...
		WRITE
TIMEOUT	100	_____
SS DEST ADD	DB 10 5	_____
SCOPE	VM	_____
VAR NAME	PROCESS PARAMETERS	_____
DOM NAME		_____
VAR TYPE		_____
NUMBER	1	_____
PARAMETERS FOR "SEND DIR" CALL TO TRIGGER THE SERVICE		
Q-TYP : DB	DB-NR : 20	Q-ANF : Q-LAE:

Fig. 11.10 Job Buffer - Write Variable

Printout of the job buffer

Once you have created the job buffers, you can use the menu function Utilities | Request Editor | Job Buffer Overview to print out the job buffers and addresses.

Request Editor		...	
OPCD	S5 ADDR.JOB.B.	NAME	S5 ADDRESS
V-RE	DB 20 1 19	PROCESS NAME	DB 11 0 16
V-WR	DB 20 21 21	PROCESS PARAMETERS	DB 10 5 11

Fig. 11.11 Job Buffer - Printout

Server PLC

The next step is to create a job buffer for reporting the process values in the program file of the server.

The program file will be called BSP1S@@ST.S5D. The job buffer will be entered in DB 20.

Request Editor Initialization		...
PROGRAM FILE	C:	BSP1S@@ST.S5D
BLOCK	DB	20

Fig. 11.12 Request Editor - Initialization of Server PLC

Job buffer for the TF service 'information report'

Enter the values according to the section of screen shown below for the variable PROCESS VALUE.

Request Editor ...

SCOPE	VB	REPORT
VAR NAME	PROCESS VALUE	
DOM NAME		
MULTIPLE ACCESS	N	

PARAMETERS FOR "SEND DIR" CALL TO TRIGGER THE SERVICE

Q-TYP : DB DB-NR : 20 Q-ANF : Q-LAE:

Fig. 11.13 Job Buffer - Information Report

Printout of the job buffer

Once you have created the job buffers, you can use the menu function Utilities | Request Editor | Job Buffer Overview to print out the job buffers and addresses.

Request Editor		...	
OPCD	S5 ADDR.JOB.B.	NAME	S5 ADDRESS
V-IN	DB 20 1 12	PROCESS VALUE	PROJECT

Fig. 11.14 Job Buffer - Printout

11.2.7 PLC Programs

The following pages contain printouts of the PLC blocks that must be created for the simulation.

The program structure is illustrated in the two following diagrams.

Following this, the data and function blocks are listed.

In both files, DB 20 contains job buffers created with the TF DB editor. The preheader of these blocks is not created automatically. You must create this yourself to make the blocks easier to read. The preheader is, however, not necessary.

The other data blocks contain the data structures recognizable from the configuring of the modules.

The handling blocks called in these listings must be those belonging to the particular PLC.



Note that the lists of function and data blocks shown in this chapter are intended to illustrate the text. The actual values are in the example files on the diskette. Use these files to assign parameters to the PLC!

Client PLC (host)

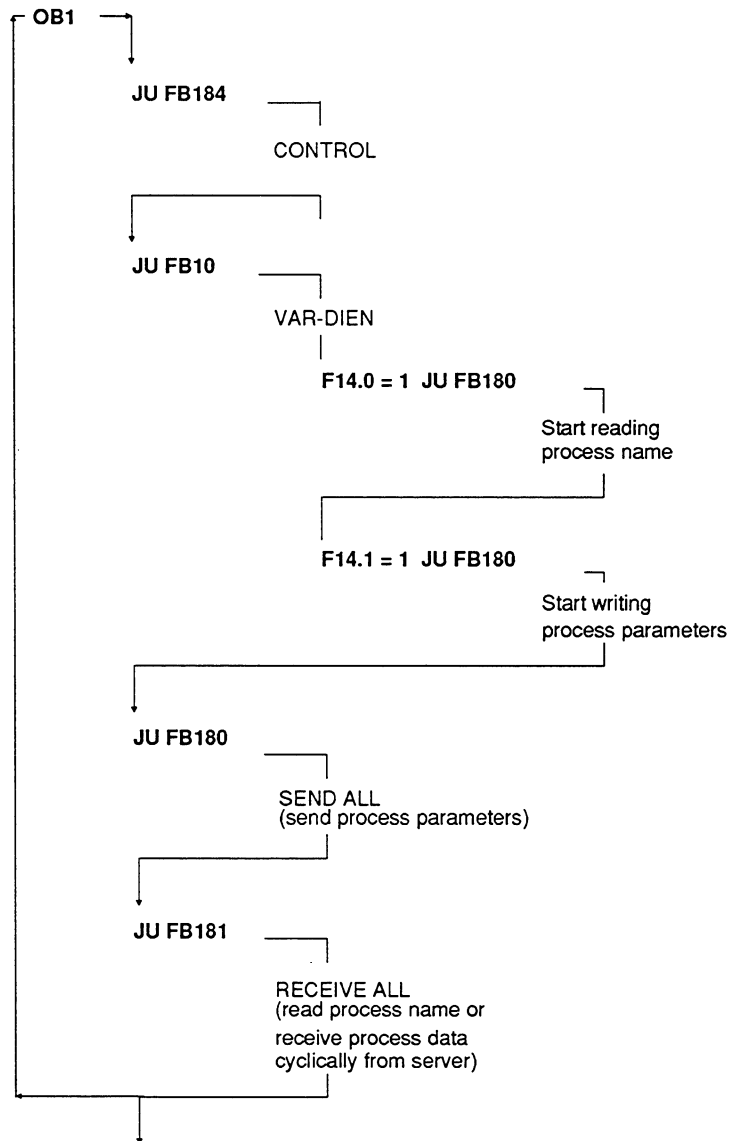


Fig. 11.15 Program Structure of the PLC Programs on the Client PLC

Server PLC (programmable logic controller)

11

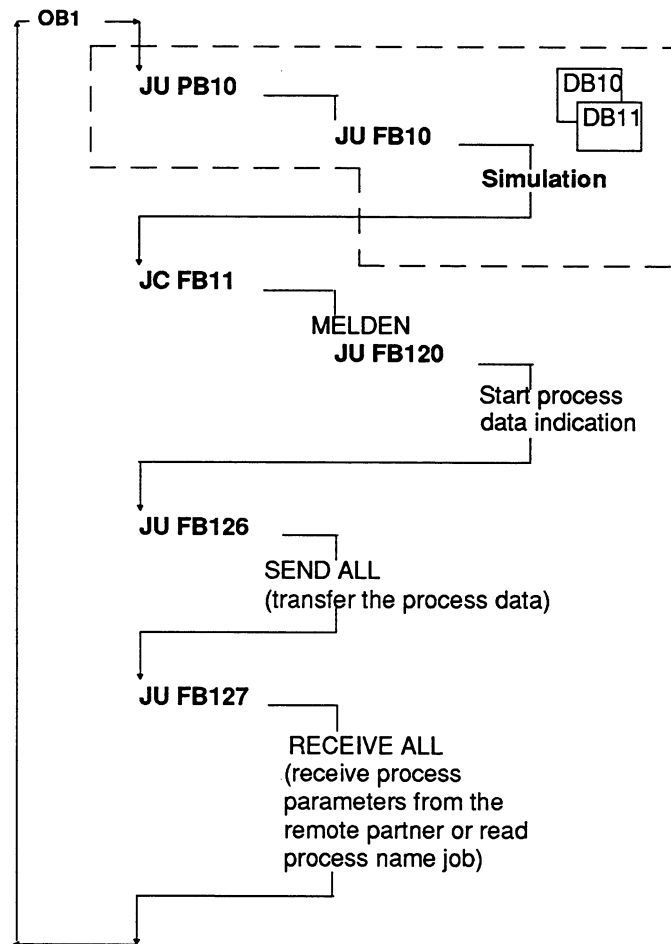


Fig. 11.16 Program Structure of the PLC Programs on the Server PLC

DB20

B:BSPC@@ST.S5D

LEN = 52 / 62
PAGE 1

```
0:      KH = 0014;
1:      KS = ' V-RE';
3:      KF = + 00100;
4:      KH = 0000;
5:      KS = ' DB';
6:      KY = 000,011;
7:      KF = + 00000;
8:      KS = ' VS';
9:      KF = + 00032;
10:     KS = ' ';
11:     KF = + 00000;
12:     KS = ' VM';
13:     KY = 000,011;
14:     KS = ' PROCESS NAME ';
20:     KY = 000,022;
21:     KS = ' V-WR';
23:     KF = + 00100;
24:     KH = 0000;
25:     KS = ' DB';
26:     KY = 000,010;
27:     KF = + 00005;
28:     KS = ' ';
29:     KF = + 00000;
30:     KS = ' ';
31:     KF = + 00000;
32:     KS = ' VM';
33:     KY = 000,016;
34:     KS = ' PROCESS PARAMETERS';
42:     KY = 255,255;
43:     KS = 'TF_EDIT';
47:
```

FB 10 B:BSPC@@ST.S5D

LEN= 50
PAGE 1

SEGMENT 1 0000

NAME	:VAR-DIEN		Variables services of client
0005	:		see also:
0006	:		TF instructions for S5;
0007	:		1.1.4 Variables Services
0008	:		
0009	:A F 14.0		The current process name can be
000A	:R F 14.0		read using flag bit 14.0;
000B	:		the bit is set to 1 here;
000C	:		DB11 from DW0 onwards
000D	:		Start the READ VAR service
000E	:		
000F	:JU FB 180		Read process name from server
0010	NAME :SEND		
0011	SSNR : KY 0,0		
0012	A-NR : KY 0,1		
0013	ANZW : FW 100	Specify in configuration	
0014	QTYP : KS DB		
0015	DBNR : KY 0,20		
0016	QANF : KF +1		Taken from configuration
0017	QLAE : KF +19		Taken from configuration
0018	PAFE : FY 106		
0019	:		
001A	:		
001B	:A F 14.1		The current process parameters
001C	:R F 14.1		can be written using flag bit 14.1.
001D	:		
001E	:		
001F	:		Start the WRITE VAR service
0020	:		
0021	:JU FB 180		Write parameters in server
0022	NAME :SEND		
0023	SSNR : KY 0,0		
0024	A-NR : KY 0,1		
0025	ANZW : FW 100		
0026	QTYP : KS DB		
0027	DBNR : KY 0,20		
0028	QANF : KF +21		Taken from configuration
0029	QLAE : KF +21		Taken from configuration
002A	PAFE : FY 106		
002B	:		
002C	:BE		

OB 1 B:SPC@@ST.S5D

LEN= 52
PAGE 1

```

SEGMENT 1 0000
0000      :                               Check the PLC-PLC link
0001      :JU FB 184
0002 NAME :CONTROL
0003 SSNR :      KY 0,0
0004 A-NR :      KY 0,1
0005 ANZW :      FW 100
0006 PAFE :      FY 106
0007      :
0008      :
0009      :                               Transfer the job buffer for
000A      :                               receiving the process name
000B      :                               or for sending the process
000C      :                               parameters to the CP;
000D      :                               depends on FY14
000D      :JU FB 10
000E NAME:VAR-DIEN
000F      :
0010      :O  F  0.0                               SEND ALL for the
0011      :ON F  0.0                               background communication
0012      :
0013      :JU FB 180
0014 NAME:SEND
0015 SSNR :      KY 0,0
0016 A-NR :      KY 0,0
0017 ANZW:  FW 110
0018 QTYP :      KS NN
0019 DBNR :  KY 0,0
001A QANF :  KF +0
001B QLAE :  KF +0
001C PAFE :  FY 114
001D      :
001E      :O  F  0.0                               RECEIVE ALL for the
001F      :ON F  0.0                               background communication
0020      :
0021      :
0022      :
0023      :JU FB 181
0024 NAME:RECEIVE
0025 SSNR :      KY 0,0
0026 A-NR :      KY 0,0
0027 ANZW :      FW 115
0028 ZTYP :      KS NN
0029 DBNR :      KY 0,0
002A ZANF :      KF +0
002B ZLAE :      KF +0
002C PAFE :  FY 119
002D      :
002E      :BE

```

OB 20 B:BSPC@@ST.S5D

LEN= 13
PAGE 1

SEGMENT 1 0000
0000 :
0001 :JU FB 185
0002 NAME :SYNCHRON
0003 SSNR : KY 0,0
0004 BLGR : KY 0,0
0005 PAFE : FY 2
0006 :
0007 :BE

OB 21 B:BSPC@@ST.S5D

LEN= 13
PAGE 1

SEGMENT 1 0000
0000 :
0001 :JU FB 185
0002 NAME :SYNCHRON
0003 SSNR : KY 0,0
0004 BLGR : KY 0,0
0005 PAFE : FY 2
0006 :
0007 :BE

OB 22 B:BSPC@@ST.S5D

LEN= 13
PAGE 1

11

SEGMENT 1 0000
0000 :
0001 :JU FB 185
0002 NAME :SYNCHRON
0003 SSNR : KY 0,0
0004 BLGR : KY 0,0
0005 PAFE : FY 2
0006 :
0007 :BE

PLC program on the server

DB10 B:BSP1S@ST.S5D

LEN= 21 /4

PAGE 1

0:	KF = + 00000;	Process value	1 (default values)
1:	KF = + 00000;	\=	2
2:	KF = + 00000;	\=	3
3:	KF = + 00000;	\=	4
4:	KF = + 00000;	\=	5
5:	KF = + 00500;	Update factor	
6:	KF = + 00100;	Process upper limit	1
7:	KF = + 00110;	\=	2
8:	KF = + 00120;	\=	3
9:	KF = + 00130;	\=	4
10:	KF = + 00140;	\=	5
11:	KF = -00140;	Process lower limit	1
12:	KF = -00130;	\=	2
13:	KF = -00120;	\=	3
14:	KF = -00110;	\=	4
15:	KF = -00100;	\=	5
16:			

```

DB11          B:BSP1S@ST.S5D                      LEN=21 /4
                                                    PAGE 1
0:           KS = 'sawtooth function '; 32 ASCII characters
12:          KS = ' ';                               are available for the process name
16:

```

```

DB20          B:BSP1S@ST.S5D                      LEN=23 /20
                                                    PAGE 1
0:           KH = 000D;
1:           KS = 'V-IN';
3:           KF = +00100;
4:           KH = 0000;
5:           KS = 'VB';
6:           KY = 000,011;
7:           KS = 'PROCESS VALUE ';
13:          KH = FFFF;
14:          KS = 'TF_EDIT';
18:

```

```

PB 10         B:BSP1S@ST.S5D                      LEN= 11
                                                    PAGE 1
SEGMENT 1    0000
0000         :
0001         :
0002         :JU FB 10
0003 NAME    :SIMUL1
0004         :
0005         :BE

```

Jump block that calls the simulation blocks.

```

FB 10          B:BSP1S@ST.S5D                                LEN= 107
                                                         PAGE  1

SEGMENT 1    0000
NAME         :SIMUL1 Sawtooth function
0005         :
0006         :C DB 10Initialize current DB
0007         :
0008         :L KH 0005          5 processes to be simulated;
000A         :T FW 10          fixed value!
000B         :
000C         :L DW 5           Test, whether FW 12 is in
000D         :L FW 12          valid area
000E         :<F              FW12 update factor
000F         :JC =M010
0010         :L KH 0000
0012         :> = F           FW12 = 0 (not negative value)
0013         :JC =M020
0014         :
0015 M010    :L DW 5           Initialization
0016         :T FW 12          - Load update factor in FW 12
0017         :L KH 0000        - Preset process values with 0
0019         :T DW 0
001A         :T DW 1
001B         :T DW 2
001C         :T DW 3
001D         :T DW 4
001E         :
001F         :
0020 M020    :L FW 12          Scan whether update factor
0021         :L KH 0000        is already 0:
0023         :> F              = 0 -> update process values
0024         :JC =M030         != 0 -> decrement update factor
0025         :
0026         :
0027         :L DW 0           Update process 1
0028         :ADD KF +2        Add constant value
002A         :L DW 6           Process upper limit (PU)
002B         :TAK
002C         :> = F           PU reached?
002D         :JC =M040
002E         :L DW 11          Process lower limit (PL)
002F M040    :T DW 0           Update process value
0030         :
0031         :L DW 1           Update process 2
0032         :ADD KF +4        Add constant value
0034         :L DW 7           Process upper limit (PU)
0035         :TAK
0036         :> =F            PU reached?
0037         :JC =M041
0038         :L DW 12          Process lower limit (PL)
0039 M041    :T DW 1           Update process value

```

003A	:	
003B	:L DW 2	Update process 3
003C	:ADD KF +5	Add constant value
003E	:L DW 8	Process upper limit (PU)
003F	:TAK	
0040	:> = F	PU reached?
0041	:JC =M042	
0042	:L DW 13	Process lower limit (PL)
0043 M042	:T DW 2	Update process value
0044	:	
0045	:L DW 3	Update process 4
0046	:ADD KF +7	Add constant value
0048	:L DW 9	Process upper limit (PU)
0049	:TAK	

FB 10	B:BSPI@ST.S5D		LEN= 107
			PAGE 2
004A	:> = F		PU reached??
004B	:JC = M043		
004C	:L DW 14		Process lower limit (PL)
004D M043	:T DW 3		Update process value
004E	:		
004F	:L DW 4		Update process 5
0050	:ADD KF +9		Add constant value
0052	:L DW 10		Process upper limit (PU)
0053	:TAK		
0054	:>=F		PU reached?
0055	:JC = M044		
0056	:L DW 15		Process lower limit (PL)
0057 M044	:T DW 4		Update process value
0058	:		
0059	:		
005A	:L DW 5		Update the update factor
005B	:T FW 12		
005C	:JU = ENDE		
005D	:		
005E	:		
005F M030	:L FW 12		Decrement update factor
0060	:ADD KF -1		
0062	:T FW 12		
0063	:		
0064	:		
0065 ENDE	:BE		

FB 11

B:BSP1S@ST.S5D

LEN= 23

PAGE 1

11

```

SEGMENT 1 0000
NAME :MELDEN
0005 :
0006 :JU FB 120
0007 NAME :SEND
0008 SSNR : KY 0,0
0009 A-NR : KY 0,1 Send "INDICATE" job buffer
000A ANZW : FW 100
000B QTYP : KS DB Current PLC link
000C DBNR : KY 0,20
000D QANF : KF +1
000E QLEN : KF +12
000F PAFE : FY 106
0010 : 5 process values are
0011 :BE transferred in two words

```

OB 1

B:BSP1S@ST.S5D

LEN= 36

PAGE 1

```

SEGMENT 1 0000
0000 : Jump file to call the
0001 : simulation
0002 :JU PB 10
0003 :
0004 :O F 0.0 Cyclic indication of process
0005 :ON F 0.0 data to the PLC remote partner
0006 : (RLO = 1)
0007 :JC FB 11
0008 NAME :INDIC
0009 :
000A :O F 0.0 SEND ALL for transferring
000B :ON F 0.0 the process data
000C :
000D :JU FB 126
000E NAME :SEND-A
000F SSNR : KY 0,0
0010 A-NR : KY 0,0
0011 ANZW : FW 110
0012 PAFE : FY 114
0013 :
0014 :O F 0.0 RECEIVE ALL for receiving
0015 :ON F 0.0 process parameters from
0016 : the remote partner
0017 :JU FB 127
0018 NAME :REC-A
0019 SSNR : KY 0,0
001A A-NR : KY 0,0
001B ANZW : FW 115

```

001C PAFE : FY 119
001D :
001E :BE

OB 20 B:BSP1S@ST.S5D

LEN= 13
PAGE 1

SEGMENT 1 0000
0000 :
0001 :JU FB 125 Synchronization PLC-CP
0002 NAME :SYNCHRON
0003 SSNR : KY 0,0
0004 BLGR : KY 0,0
0005 PAFE : FY 2
0006 :
0007 :BE

OB 21 B:BSP1S@ST.S5D

LEN= 13
PAGE 1

SEGMENT 1 0000
0000 :
0001 :JU FB 125
0002 NAME :SYNCHRON
0003 SSNR : KY 0,0
0004 BLGR : KY 0,0
0005 PAFE : FY 2
0006 :
0007 :BE

OB 22 B:BSP1S@ST.S5D

LEN=13
PAGE 1

SEGMENT 1 0000
0000 :
0001 :JU FB 125
0002 NAME :SYNCHRON
0003 SSNR : KY 0,0
0004 BLGR : KY 0,0
0005 PAFE : FY 2
0006 :
0007 :BE

11.2.8 Starting Up

To start up, the configuration data must be transferred to the PLCs. Use the following functions:

- > Transfer | Transfer database for the files
PBSPL.CLT -> client PLC and PBSPL.SRV -> server PLC
- > The load function under S5DOS-KOMI to transfer the PLC data base created with the REQUEST-EDITOR.

11.2.9 Monitoring the Process at the PG

To monitor the process, the PG must be online with the CPU of the client. This connection can be established by the AS 511 interface or via the SINEC H1 bus.

Using the function "FORCE VAR", you can monitor the current process values and the status of the link (status word). In addition to this, you can also read the process name and write the process parameters.

FW 100 Status word for the job (job number 1)
FW 102 Length word for this job
FW 104 TF error number, if a TF error occurs

FY 14 Bit 0 Read the process name
Bit 1 Write the process parameters

DB 10
DW 0 to 4 Current process values from the server
DW 5 to 15 Process parameters transferred to the server

DB 11
DW 0 to 15 After reading, the process name is stored here

11.3 Example 2: Using the Domain and Program Invocation Services

The server PLC program written for the first example can be used in modified form in the second example. First, the additional tasks are explained.

11.3.1 Task for the Domain Services

The task

The server PLC simulates a certain process response by incrementing process values on the PLC with a selectable rate of rise until a certain limit value is reached (sawtooth function). It is now assumed that a different manufacturing process with a different process response is to be monitored. This process response is a continuous rise and fall in the process values (delta function). The PLC program must therefore be able to change its simulated process response when requested to.

The solution

The solution to this task is provided by the domain services. They permit loadable program sections to be defined.

Implementation

The program developed in the first example for the server PLC is extended by a further simulation in the second example. The program and data sections responsible for the sawtooth are taken out and copied. The copy is changed so that a delta function is implemented.

Loading and activating the simulation

To be able to load and activate the required process simulation on the PLC, domain services are used. These are available under the program package PGLOAD on the programmer and can be activated in various screens. A PG is also connected to the system via a SINEC-H1 link and used as the host computer in the example.

11.3.2 Tasks for the Program Invocation Services

The task

It must be possible to influence the running of the simulation program using the host. Certain status requests must be transferred to the PLC to the simulation program reacts with certain actions.

The following program invocation (PI) statuses are requested and the PLC must respond with the specified actions:

- IDLE
The simulation is not intended to run. This means that PB 10 must not be called
- STARTING
In this status, all process values must be set to the default (0).
Following this, the transition is acknowledged positively.
- RUNNING
The simulation process is run without any restrictions.
- STOPPING
The simulation is to be stopped, the transition must be acknowledged.
- STOPPED
As IDLE
- RESUMING
As STARTING
- RESETTING
As Stopping

The solution

The program invocation services allow the process to be combined with a status. With these services, the status information can be transferred and the required actions triggered on the PLC.

Implementation

The program invocation services can be called directly using the PGLOAD package in various screens. The PG connected to the system via the

SINEC H1 link is used to control the PLC simulation with the PGLOAD functions.

Overview of the device configuration

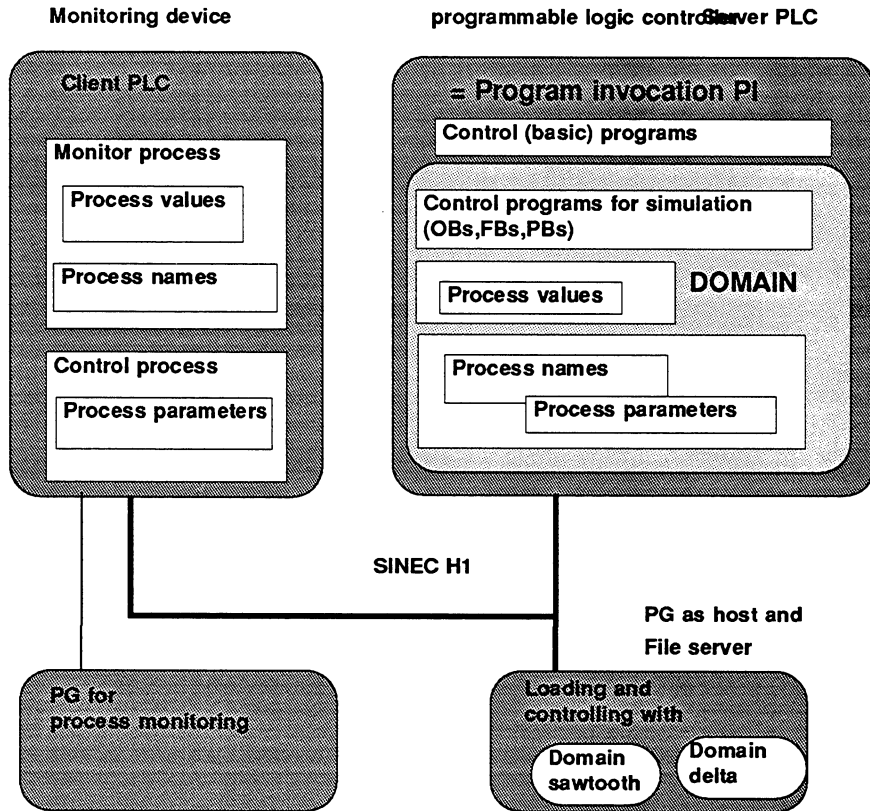


Fig. 11.17 Device Configuration in Example 2

11.3.3 Preparing Programs and Data

The sections of the program responsible for the simulation must be removed from the program file "BSP1S@ST.S5D" and stored in the file "SIMUL1ST.S5D". These involve the blocks:

PB 10 call block
FB 10 actual simulation
DB 10 process values/process parameters
DB 11 process name.

These four blocks must once again be contained in the program file "SIMUL2ST.S5D" that will contain the delta function, and FB 10 modified accordingly. The name is replaced by "DELTA FUNCTION".

The program files "SIMUL1ST.S5D" and "SIMUL2ST.S5D" can be loaded alternatively as domains in the server station.

All other blocks from the program file "BSP1S@ST.S5D" are transferred to the new program "BSP2S@ST.S5D".

To integrate the program invocation services, a control sequence is programmed in FB 1 that allows a specific program section to be called depending on the status of the program invocation generated by the host computer. The status of the program invocation is scanned using FB 103 (FB PI-ZUSTD). If the invocation is to change to a different status, this can also be achieved with this block (acknowledgement).

The following pages contain the blocks of the three program files that exist for the server in the second example.

Notes on the restart blocks (OB 20, 21, 22).

In contrast to the first example, when using the program invocation services in the cold restart branch (OB 20, 21), the block for synchronization (FB SYNCHRON) must not necessarily be called. A new synchronization must only be carried out if the CP 143 is in the non-synchronized status. The CP status can also be scanned using FB PI-ZUSTD.

In the warm restart branch, following power down, synchronization must always be carried out.

The following diagram illustrates how the domain is taken from the original program file of the server PLC.

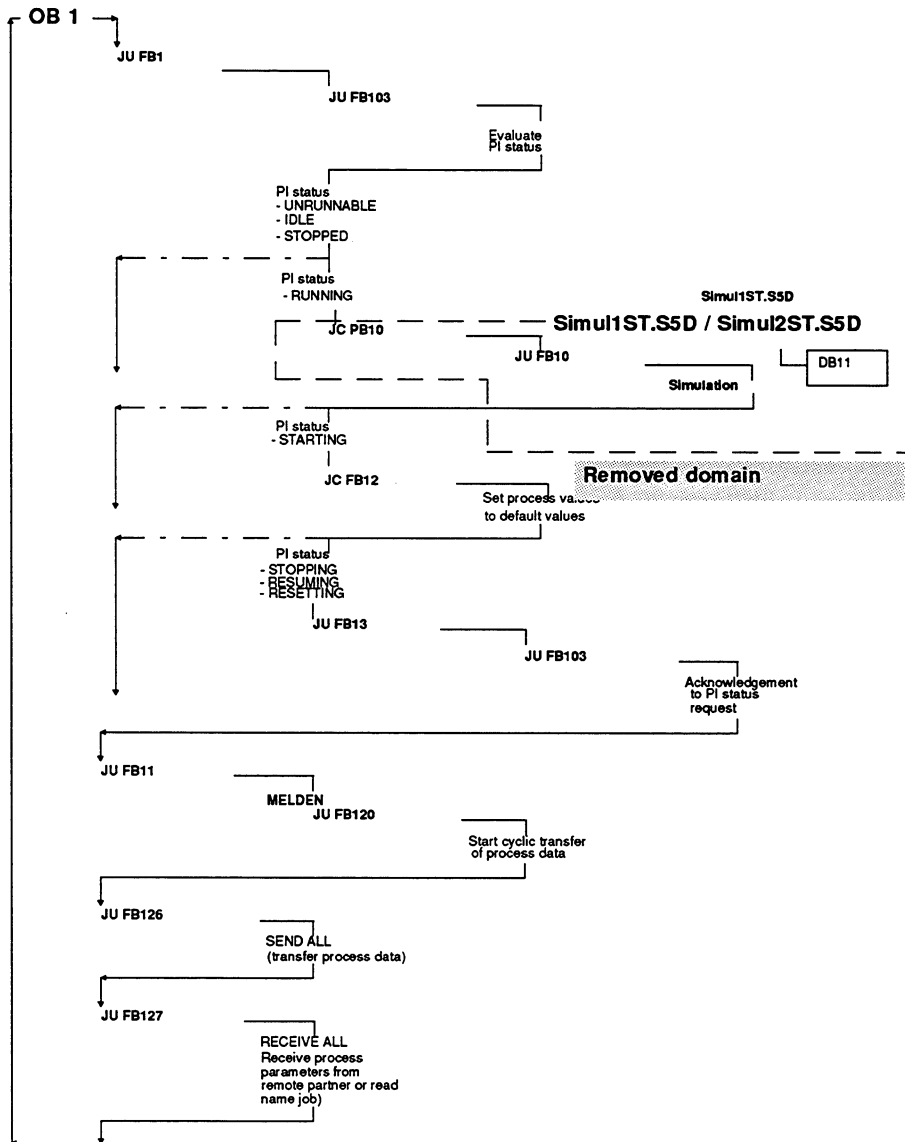


Fig. 11.18 Extracting the Domain from the Original Program File of the Server PLC

```

DB20      B:BSP2S@ST.S5D
0:        KH =    000D;
1:        KS =    'V-IN';
3:        KF =    +00100;
4:        KH =    0000;
5:        KS =    'VB';
6:        KY =    000,011;
7:        KS =    'PROCESS VALUE';
13:       KH =    FFFF;
14:       KS =    'TF_EDIT';
18:

```

```

LEN= 23 /20
PAGE  1

```

```

FB 1      B:BSP2S@ST.S5D

```

```

LEN= 77
PAGE  1

```

```

SEGMENT 1  0000
NAME       :CONTROL
0005      :
0006      :A F  0.0           RLO = 0; PI status scan
0007      :AN F  0.0
0008      :
0009      :
000A      :
000B      :
000C      :JU FB 103
000D NAME :PI-ZUSTD
000E SSNR :      KY 0,0
000F Q/ZT :      KS FW
0010 DBNR : KY 0,0
0011 Q/ZA :      KF + 60     Status in FW60
0012 PAFE :      FY 62
0013      :
0014      :A F  61.0         PI status = UNRUNNABLE
0015      :JC = ENDE
0016      :
0017      :A F  61.1         PI status = IDLE
0018      :JC = ENDE
0019      :
001A      :A F  61.2         PI status = RUNNING
001B      :JC PB 10         Jump block for calling
001C      :                 the simulation
001D      :A F  61.2
001E      :JC = ENDE
001F      :
0020      :A F  61.3         PI status = STOPPED
0021      :JC = ENDE
0022      :
0023      :A F  61.4         PI status = STARTING
0024      :                 If the STARTING status
0025      :                 is activated, the process
0026      :                 data are preset with a

```

```
0027      :                               fixed value.
0028      :JC FB 12
0029 NAME :STARTING
002A      :A F 61.4                       There is an acknowledgement
002B      :                               to the PI status scan
002C      :JC FB 13
002D NAME:ACKNOWL
002E      :A F 61.4
002F      :JC = ENDE
0030      :
0031      :A F 61.5                       PI status = STOPPING
0032      :                               There is an acknowledgement
0033      :                               to the PI status scan
0034      :JC FB 13
0035 NAME:ACKNOWL
0036      :A F 61.5
0037      :JC = ENDE
0038      :
0039      :A F 61.6                       PI status = RESUMING
003A      :                               There is an acknowledgement
003B      :                               to the PI status scan
003C      :JC FB 13
003D NAME:ACKNOWL
003E      :A F 61.6
003F      :JC = ENDE
0040      :
0041      :A F 61.7                       PI status = RESETTING
```



```

FB 1          B:BSP2S@ST.S5D          LEN= 77          PAGE 2
0042          :
0043          :
0044          :JC FB 13
0045 NAME     :ACKNOWL
0046          :
0047 ENDE     :BE

```

There is an acknowledgement to the PI status scan

```

FB 11         B:BSP2S@ST.S5D          LEN= 23          PAGE 1
SEGMENT 1     0000
NAME          :INDIC
0005          :
0006          :JU FB 120
0007 NAME     :SEND
0008 SSNR     :          KY 0,0
0009 A-NR     :          KY 0,1
000A ANZW     :          FW 100      Specified in the configuration
000B QTYP     : KS DB
000C DBNR     : KY 0,20
000D QANF     : KF + 1   Taken from configuration
000E QLAE     : KF + 12  Taken from configuration
000F PAFE     :          FY 106
0010          :
0011          :BE

```

Send "INDICATE" job buffer

```

FB 12         B:BSP2S@ST.S5D          LEN=24          PAGE 1
SEGMENT 1     0000
NAME          :STARTING
0005          :
0006          :C  DB 10
0007          :L  KH 0000
0009          :T  DW  0
000A          :T  DW  1
000B          :T  DW  2
000C          :T  DW  3
000D          :T  DW  4
000E          :L  KF +500
0010          :T  DW  5
0011          :
0012          :BE

```

Set process values to default

FB 13 **B:BSP2S@ST.S5D** **LEN= 23**
PAGE 1

```

SEGMENT 1  0000
NAME       :ACKNOWL
0005      :
0006      :O F  0.0           RLO = 1
0007      :ON F  0.0
0008      :
0009      :JU FB 103         Acknowledge PI status
000A NAME :PI-ZUSTD
000B SSNR : KY 0,0
000C Q/ZT :           KS FW
000D DBNR : KY 0,0
000E Q/ZA :           KF + 60
000F PAFE :           FY 62
0010      :
0011      :BE

```

OB 1 **B:BSP2S@ST.S5D** **LEN= 38**
PAGE 1

```

SEGMENT 1  0000
0000      :
0001      :
0002      :
0003      :JU FB 1
0004 NAME :CONTROL
0005      :
0006      :O F  0.0           There is no cyclic indication
0007      :ON F  0.0           of process data to the PLC
0008      :                   remote partner (RLO = 1)
0009      :JU FB 11
000A NAME :INDIC
000B      :
000C      :O F  0.0           SEND ALL for transferring the
000D      :ON F  0.0           process data
000E      :
000F      :JU FB 126
0010 NAME :SEND-A
0011 SSNR :           KY 0,0
0012 A-NR :           KY 0,0
0013 ANZW : FW 115
0014 PAFE : FY 119
0015      :
0016      :O F  0.0           RECEIVE ALL for transferring
0017      :ON F  0.0           process parameters of
0018      :                   the remote partner
0019      :JU FB 127
001A NAME :REC-A
001B SSNR :           KY 0,0

```

```

001C A-NR :      KY 0,0
001D ANZW: FW 120
001E PAFE :      FY 124
001F      :
0020      :BE

```

OB 20 B:BSP2S@ST.S5D

LEN= 27
PAGE 1

```

SEGMENT 1  0000
0000      :
0001      :A F  0.0           RLO = 0; PI status scan
0002      :AN F  0.0
0003      :
0004      :JU FB 103
0005 NAME :PI-ZUSTD
0006 SSNR :      KY 0,0
0007 Q/ZT :      KS FW
0008 DBNR :      KY 0,0
0009 Q/ZA :      KF + 60
000A PAFE :      FY 62
000B      :
000C      :A F  60.6         Scan CP status
000D      :A F  60.7         - not synchronized
000E      :
000F      :JC FB 125
0010 NAME :SYNCHRON
0011 SSNR :      KY 0,0
0012 BLGR :      KY 0,0
0013 PAFE :      FY 2
0014      :
0015      :BE

```

OB 21 B:BSP2S@ST.S5D

LEN= 27
PAGE 1

```

SEGMENT 1  0000
0000      :
0001      :A F  0.0           RLO = 0; PI status scan
0002      :AN F  0.0
0003      :
0004      :JU FB 103
0005 NAME :PI-ZUSTD
0006 SSNR :      KY 0,0
0007 Q/ZT :      KS FW
0008 DBNR : KY 0,0
0009 Q/ZA :      KF + 60
000A PAFE :      FY 62

```

```
000B      :
000C      :A F 60.6          Scan CP status
000D      :A F 60.7          - not synchronized
000E      :
000F      :JC FB 125
0010 NAME :SYNCHRON
0011 SSNR  :          KY 0,0
0012 BLGR  :          KY 0,0
0013 PAFE  :          FY 2
0014      :
0015      :BE
```

OB 22 B:BSP2S@ST.S5D

LEN=13
PAGE 1

```

SEGMENT 1 0000
0000      :Synchronization PLC-CP
0001      :JU FB 125
0002 NAME :SYNCHRON
0003 SSNR :      KY 0,0
0004 BLGR :      KY 0,0
0005 PAFE :      FY 2
0006      :
0007      :BE

```

Simulation of the sawtooth function:

DB10 B:SIMUL1ST.S5D

LEN= 21 /4
PAGE 1

0:	KF = + 00000;	Process value	1 (Default values)
1:	KF = + 00000;	=	2
2:	KF = + 00000;	=	3
3:	KF = + 00000;	=	4
4:	KF = + 00000;	=	5
5:	KF = + 00500;	Update factor	
6:	KF = + 00100;	Process upper limit	1
7:	KF = + 00110;	=	2
8:	KF = + 00120;	=	3
9:	KF = + 00130;	=	4
10:	KF = + 00140;	=	5
11:	KF = - 00140;	Process lower limit	1
12:	KF = - 00130;	=	2
13:	KF = - 00120;	=	3
14:	KF = - 00110;	=	4
15:	KF = - 00100;	=	5
16:			

DB11 B:SIMUL1ST.S5D

LEN= 21 /4
PAGE 1

```

0      : KS = 'Sawtooth function'; 32 ASCII characters are
12     : KS = '      ';          available for the process name
16:

```

PB 10 **B:SIMUL1ST.S5D** **LEN=11** **PAGE 1**

SEGMENT 1 0000
0000 : Jump block that calls the
0001 : simulation blocks.
0002 : JU FB 10
0003 NAME :SIMUL1
0004 : :
0005 :BE

FB 10 **B:SIMUL1ST.S5D** **LEN=107** **PAGE 1**

SEGMENT 1 0000
NAME :SIMUL1 Sawtooth function
0005 :
0006 :C DB 10 Initialize current DB
0007 :
0008 :L KH 0005 5 processes to be simulated;
000A :T FW 10 fixed value!
000B :
000C :L DW 5 Test whether FW12 is in
000D :L FW 12 valid area
000E :<=F FW12 update factor
000F :JC =M010
0010 :L KH 0000
0012 :>=F FW12 = 0 (not negative value)
0013 :JC =M020
0014 :
0015 M010 :L DW 5 Initialization
0016 :T FW 12 - Load update factor in FW12
0017 :L KH 0000 - Preset process values with 0
0019 :T DW 0
001A :T DW 1
001B :T DW 2
001C :T DW 3
001D :T DW 4
001E :
001F :
0020 M020 :L FW 12 Scan whether the update factor
0021 :L KH 0000 is already 0:
0023 :>=F = 0 - update process values
0024 :JC = M030 != 0 - decrement update factor
0025 :
0026 :
0027 :L DW 0 Update process 1
0028 :ADD KF +2 Add constant value
002A :L DW 6 Process upper limit (PU)
002B :TAK
002C :>=F PU reached?

002D	:JC =M040	
002E	:L DW 11	Process lower limit (PL)
002F M040	:T DW 0	Update process value
0030	:	
0031	:L DW 1	Update process 2
0032	:ADD KF +4	Add constant value
0034	:L DW 7	Process upper limit (PU)
0035	:TAK	
0036	:> =F	PU reached?
0037	:JC =M041	
0038	:L DW 12	Process lower limit (PL)
0039 M041	:T DW 1	Update process value
003A	:	
003B	:L DW 2	Update process 3
003C	:ADD KF +5	Add constant value
003E	:L DW 8	Process upper limit (PU)
003F	:TAK	
0040	:> =F	PU reached?
0041	:JC =M042	
0042	:L DW 13	Process lower limit (PL)
0043 M042	:T DW 2	Update process value
0044	:	
0045	:L DW 3	Update process 4
0046	:ADD KF +7	Add constant value
0048	:L DW 9	Process upper limit (PU)
0049	:TAK	

FB 10	B:SIMUL1ST.S5D		LEN= 107
			PAGE 2
004A	:> = F	PU reached?	
004B	:JC = M043		
004C	:L DW 14	Process lower limit (PL)	
004D M043	:T DW 3	Update process value	
004E	:		
004F	:L DW 4	Update process 5	
0050	:ADD KF + 9	Add constant value	
0052	:L DW 10	Process upper limit (PU)	
0053	:TAK		
0054	:> = F	PU reached?	
0055	:JC = M044		
0056	:L DW 15	Process lower limit (PL)	
0057 M044	:T DW 4	Update process value	
0058	:		
0059	:		
005A	:L DW 5	Update update factor	
005B	:T FW 12		
005C	:JU = ENDE		
005D	:		
005E	:		
005F M030	:L FW 12	Decrement update factor	
0060	:ADD KF -1		
0062	:T FW 12		
0063	:		
0064	:		
0065 ENDE	:BE		

Simulation of the delta function

```

DB10          B:SIMUL2ST.S5D                               LEN= 21 /4
                                                         PAGE 1
0:            KF = + 00000;           Process value      1 (Default values)
1:            KF = + 00000;           |=\                2
2:            KF = + 00000;           |=\                3
3:            KF = + 00000;           |=\                4
4:            KF = + 00000;           |=\                5
5:            KF = + 00500;           Update factor
6:            KF = + 00100;           Process upper limit 1
7:            KF = + 00110;           |=\                2
8:            KF = + 00120;           |=\                3
9:            KF = + 00130;           |=\                4
10:           KF = + 00140;           |=\                5
11:           KF = -00140;           Process lower limit 1
12:           KF = -00130;           |=\                2
13:           KF = -00120;           |=\                3
14:           KF = -00110;           |=\                4
15:           KF = -00100;           |=\                5
16:

```

```

DB11 B:SIMUL2ST.S5D                               LEN= 21 /4
                                                         PAGE 1
0:            KS = 'Delta function'; 32 ASCII characters are available
12:           KS = '      ';           for the process name
16:

```

```

PB 10         B:BSIMUL2ST.S5D                               LEN= 11
                                                         PAGE 1
SEGMENT 1    0000
0000         :
0001         :           Jump block that caused the
0002         :           simulation blocks
0003 NAME    :JU FB 10
0004         :
0005         :BE

```

```

FB 10         B:SIMUL2ST.S5D                               LEN=166
                                                         PAGE 1
SEGMENT 1    0000
NAME         :SIMUL2 Delta function
0005         :
0006         :C DB 10           Initialize current DB
0007         :
0008         :L KH 0005        5 processes to be simulated;

```

000A	:T FW 10	fixed value!
000B	:	
000C	:L DW 5	Test whether FW12 is in
000D	:L FW 12	valid area
000E	:< F	FW12 > update factor
000F	:JC =M010	
0010	:L KH 0000	
0012	:>=F	FW12 = 0 (not negative value)
0013	:JC =M020	
0014	:	
0015 M010	:L DW 5	Initialization
0016	:T FW 12	- Load update factor in FW 12
0017	:L KH 0000	- Preset process values with 0
0019	:T DW 0	
001A	:T DW 1	
001B	:T DW 2	
001C	:T DW 3	
001D	:T DW 4	
001E	:	
001F	:	
0020 M020	:L FW 12	Scan whether the update factor
0021	:L KH 0000	is already 0:
0023	:> F	= 0 - update process values
0024	:JC =M030	!= 0 - decrement update factor
0025	:	
0026	:	
0027 M021	:L DW 0	Update process 1
0028	:AN F 14.0	Rising or falling edge?
0029	:JC = M040	= 0 - process value falling
002A	:	= 1 - process value rising
002B	:ADD KF +2	Add constant value
002D	:L DW 6	Process upper limit (PU)
002E	:<=F	PU reached?
002F	:JC =M041	
0030	:R F 14.0	From here, falling edge
0031	:L DW 0	
0032 M040	:ADD KF -2	Subtract constant value
0034	:L DW 11	Process lower limit (PL)
0035	:> F	PL reached?
0036	:JC =M041	
0037	:S F 14.0	From here, rising edge
0038	:JU =M021	
0039 M041	:TAK	Update process value
003A	:T DW 0	
003B	:	
003C	:	
003D M022	:L DW 1	Update process 2
003E	:AN F 14.1	Edge rising or falling?
003F	:JC =M042	= 0 - process value falling
0040	:	= 1 - process value rising
0041	:ADD KF +4	Add constant value

0043	:L DW 7	Process upper limit (PU)
0044	:<=F	PU reached?
0045	:JC =M043	
0046	:R F 14.1	From here, falling edge
0047	:L DW 1	
0048 M042	:ADD KF -4	Subtract constant value

FB 10 B:SIMUL2ST.S5D

LEN= 166

PAGE 2

004A	:L DW 12	Process lower limit (PL)
004B	:>F	PL reached?
004C	:JC = M043	
004D	:S F 14.1	From here, rising edge
004E	:JU = M022	
004F M043	:TAK	Update process value
0050	:T DW 1	
0051	:	
0052	:	
0053 M023	:L DW 2	Update process 3
0054	:AN F 14.2	Edge rising or falling?
0055	:JC = M044	= 0 - process value falling
0056	:	= 1 - process value rising
0057	:ADD KF +5	Add constant value
0059	:L DW 8	Process upper limit (PU)
005A	:<=F	PU reached?
005B	:JC = M045	
005C	:R F 14.2	From here, falling edge
005D	:L DW 2	
005E M044	:ADD KF -5	Subtract constant value
0060	:L DW 13	Process lower limit (PL)
0061	:> F	PL reached?
0062	:JC =M045	
0063	:S F 14.2	From here, rising edge
0064	:JU = M023	
0065 M045	:TAK	Update process value
0066	:T DW 2	
0067	:	
0068	:	
0069 M024	:L DW 3	Update process 4
006A	:AN F 14.3	Edge rising or falling?
006B	:JC =M046	= 0 - process value falling
006C	:	= 1 - process value rising
006D	:ADD KF +7	Add constant value
006F	:L DW 9	Process upper limit (PU)
0070	:< = F	PU reached?
0071	:JC = M047	
0072	:R F 14.3	From here, falling edge
0073	:L DW 3	
0074 M046	:ADD KF -7	Subtract constant value

0076	:L DW 14	Process lower limit (PL)
0077	:>F	PL reached?
0078	:JC = M047	
0079	:S F 14.3	From here, rising edge
007A	:JU = M024	
007B M047	:TAK	Update process value
007C	:T DW 3	
007D	:	
007E	:	
007F M025	:L DW 4	Update process 5
0080	:AN F 14.4	Edge rising or falling?
0081	:JC = M048	= 0 - process value falling
0082	:	= 1 - process value rising
0083	:ADD KF +9	Add constant value
0085	:L DW 10	Process upper limit (PU)
0086	:<=F	PU reached?
0087	:JC = M049	
0088	:R F 14.4	From here, falling edge
0089	:L DW 4	
008A M048	:ADD KF -9	Subtract constant value
008C	:L DW 15	Process lower limit (PL)
008D	:> F	PL reached?
008E	:JC = M049	
008F	:S F 14.4	From here, rising edge

FB 10 B:SIMUL2ST.S5D

LEN= 166

PAGE 3

0090	:JU = M025	
0091 M049	:TAK	Update process value
0092	:T DW 4	
0093	:	
0094	:	
0095	:L DW 5	Update update factor
0096	:T FW 12	
0097	:JU =ENDE	
0098	:	
0099	:	
009A M030	:L FW 12	Decrement update factor
009B	:ADD KF -1	
009D	:T FW 12	
009E	:	
009F	:	
00A0 ENDE	:BE	

11.3.4 Executing Domain and PI Services

The server PLC is set to STOP at the PG and an overall reset carried out.

The file "BSP2S@ST.S5D" is transferred to the server PLC.
In this status, no domain is loaded and no program invocation created.

Monitoring the processes

As described in the first example, the process is monitored at a PG connected ONLINE to the client station.

From the process values (e.g. whether or not they change) you can recognize the current status of the process in the server.

By reading the process name, you can also determine which process is currently loaded. As in the first example, you can preset the process parameters.

Loading domains, handling the program invocation

As already explained, a PG serves as the host to be able to manipulate domains and PIs.

In the server PLC, an AS 511 cable connection between the CP 143 and PLC-CPU must be established ("swing cable").

The "PGLOAD" package is started on the host PG equipped with a SINEC H1 interface module

In the initialization screen form, you specify the name of a file containing the information about the PG link to the required PLC (in this case "SERVERCP.LOD")

After selecting the function PGLOAD | PLC Links, you can input the Ethernet address of the server PLC. The additional name of the link allows you to display the currently selected PLC link in the PGLOAD package.

After storing the information with "OK" (F7) and pressing F1, you return to the basic screen.

PGLOAD		CP 143 (END)													
LINK CONFIGURATION / PLC SELECTION															
LINK NAME :		SERVER													
REMOTE ETHERNET ADDRESS:		08000601B030													
F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>
1		2		3		4		5		6		7	OK	8	SELECT

Fig. 11.19 PGLOAD Link Configuration / Select PLC

Generating loadable domains

Before an S5-DOS program file can be transferred to the PLC using the domain services, a load file must first be generated. To do this, select the function PLOAD | Transfer Functions.

In the next screen form, you enter the name of the local S5-DOS program file (SIMUL1ST.S5D) from which the load file is generated with "CREATE" (F4).

PGLOAD		SINEC NCM		CP 143 (END)	
TRANSFER FUNCTIONS					
DEST STATION (FILE SERVER):				SINEC NCM	
SERVER FILE :					
LOCAL S5 PROGRAM FILE :		DR: B	NAME: SIMUL1 ST.S5D		
COMMENT :					
F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>
1		2	VARIABLES	3	SEND
		4	CREATE	5	FETCH
		6	S5 FIL CR	7	SERVER
		8	SELECT		
				F	HELP

Fig. 11.20 PLOAD Transfer Functions

Follow the same procedure with the file SIMUL2ST.S5D.

Generating load files produces files with the names "SIMUL1ST.DOM" or "SIMUL2ST.DOM" on the local drive. The program files remain unchanged.

When you exit the screen form with "RETURN" you return to the main menu.

Loading domains

Now select the function PGLOAD | Host computer functions.

By pressing PLC LOAD in the currently displayed screen, call the screen PGLOAD LOAD PLC. Firstly, the file "SIMUL1ST.S5D" should be loaded as a domain in the server station.

Enter the following parameters:

"PG" as the server link: this means that the load file exists on the PG and will be loaded from the PG to the PLC. The name of the domain on the server station will be "SAWTOOTH".

As file name, use the name of the local program file (not the name of the load file), i.e. "SIMUL1ST.S5D". The other input fields remain unchanged or empty.

PGLOAD		SINEC NCM		CP 143 (END)											
LOAD PLC															
PLC LINK:	SERVER														
SERVER LINK:	PG														
PROGRAM (DOMAIN) :	SAWTOOTH														
STORED IN FILE :	B: SIMUL1ST.S5D														
PARAMETERS (DOMAIN) :															
STORED IN FILE :															
CPU NO : 1															
F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>	F	<input type="text"/>
1	2	3	4	5	6	7	OK	8	SELECT						

Fig. 11.21 PGLOAD Load PLC

If you press the "OK" key (F7), the file is transferred to the PLC as a domain. If you press "RETURN" (F8) you exit the screen form and return to the screen PGLOAD HOST COMPUTER FUNCTIONS:

With this screen displayed, you can control the system by selecting the required program statuses and transferring them to the PLC. The reaction of the process simulated on the PLC can be monitored using the displays on the monitoring device.

The function keys and their significance in controlling the example process are described below.

In this screen (PGLOAD HOST COMPUTER FUNCTIONS), the function keys have the following effects:

F3 PLC DELETE

Domains on the PLC are deleted.

F4 PLC START/ RESET

Depending on the PLC mode, a program invocation (PI) is created or deleted.

If you delete a PI, the PLC changes to the STOP mode.

If you create a PI, the PLC changes to the RUN mode, the PI is in the IDLE status.

After creating a PI, process parameters can be set by the client PLC or the process name can be read. The process values are indicated by the server to the client, however, since the process has not yet started, the process values do not change.

F5 PROGRAM START

The PI is changed to the STARTING status. The PLC program in the server sets the process values to the default, acknowledges the STARTING or RESUMING status and the process begins. This can be monitored at the PG of the CLIENT station.

The PI is then in the RUNNING status.

F6 PROGRAM STOP

The PI is changed to the STOPPING status. The PLC program in the server acknowledges this transition immediately and the PI changes to the STOPPED status.

This can be monitored at the client, since the process values no longer change from this point onwards.

F7 STATION SELECT

Using this key, you can select the next link in the link file (SERVER CP.LOD). Since there is only one link in this example, the key does not have any effect.

You can press keys F5 and F6 alternately. The process is then started and stopped alternately.

When the process is stopped, you can delete the PI with F4. After this, the domain can then be deleted with F3 and the other simulation (e.g. "DELTA") can be loaded.

11.4 Example 3: Transparent Data Exchange with Acknowledgement (T-DQ)

The task

You want to exchange data without checking the transfer and without transmitting structural information as simply as possible within a SIMATIC network

In concrete terms, in this example 18 data words are transferred from PLC 1 DX 10 to PLC2 DB 20 and in the opposite direction (acknowledgment) 7 data words from PLC 2 DB 21 to PLC 1 DX 11.

The solution

The 'Transparent data exchange' services allow data transfer under such conditions.

Implementation: "Transparent data exchange with acknowledgment" (TDQ)

This communications service transfers data from the client to a server and from the server to the client in the acknowledgement.

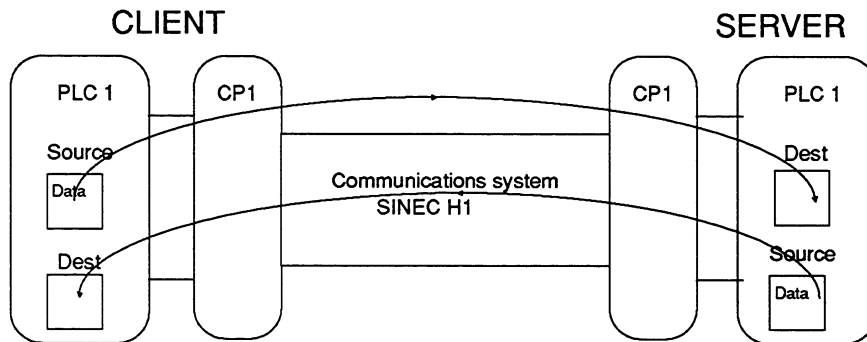


Fig. 11.22 Transparent Data Exchange with Acknowledgment

Using the configuration tool

The following example illustrates the configuration steps required.

To configure this service, you require the two software packages COM 143, TF editor and "LAD, CSF, STL" which are started on the PG 685 or PG 675 under S5-DOS. Within COM 143 TF, the Request Editor is also used when creating job buffers.

COM 143 TF

Using COM 143, you assign the station address to the CPs and set up an application association (logical link layer 7) between CP 1 and CP 2 (layer 4 is also possible).

Request Editor

With the Request Editor, you create the job buffer T-DQ for the client.

S5DOS-KOMI

Using the transfer functions of S5DOS-KOMI you transfer the job buffer from the PG to the client (PC 1) and write a user program with handling blocks in the client (PLC1) and in the server (PLC2) that triggers the job and transfers the data between the CP and PLC.

The values used for this example can be found in Fig. 11.33.

Configuration procedure

To configure the "transparent data exchange service", proceed as follows:

1. With the COM 143 package

- Set up station address (6 bytes long Ethernet address) in both CP 1 and CP 2 if this does not yet exist. (Here you require the online mode)
Menu: Edit -> CP basic initialization (CP in STOP status)
- Set up application association (logical link layer 7a) from CP 1 to CP 2.
Menu: Edit-> Links-> Appl. Associations

Link block for client (CP1)

Enter the following parameters in the screen form:

- SSNR (same as SSNR of SYNCHRON in PLC 1)
- ANR. (only use odd job numbers)
- ANZW (FW, occupies three words)
- LOCAL TSAP (CLIE T-DQ)
- EST TYPE (A7) CP 1 activates the connection establishment
- MUX ADDRESS (0 = no multiplexing)
- REMOTE TSAP (SERV T-DQ)
- APPL. ASS. NAME (only for information)
- REMOTE ETHERNET ADDRESS (address of CP 2)
- NUMBER OF APPL. ASS. FOR CURRENT TSAP
(see also VOLUME 2 "Configuring Application Associations")

Link block for server (CP2)

Enter the following parameters in this screen form:

- SSNR (same as SSNR of SYNCHRON in PLC 2)
- ANR. (only use odd job numbers)
- ANZW (FW, occupies three words)
- LOCAL TSAP (SERV T-DQ)
- EST TYPE (P7)
- MUX ADDRESS (0 =no multiplexing)
- REMOTE TSAP (CLIE T-DQ)

- APPL. ASS. NAME (only for information)

- REMOTE ETHERNET ADDRESS (address of CP 1)
- NUMBER OF APPL. ASS. FOR CURRENT TSAP
(see also VOLUME 2 "Programming Application Associations")

2. With the TF editor package

Set up the job buffer T-DQ for the CLIENT (PLC 1) as follows:

- Load the Request Editor and specify the program file in the initial screen form and select the data block.
- Menu: Tools -> Request editor -> Init -> Create job buffer
- Select transparent data exchange and enter the following parameters:
 - source address (data block)
 - source length (length of the useful data)
 - dest. address (data block)
 - dest. length (length of the useful data)
- After you have input the data, press OK.
A job buffer is set up, that you must load in PLC 1 (the CLIENT).
Transfer this data block with LAD, CSF, STL.

3. With LAD, CSF, STL

- > For PLC 1 (CLIENT)
 - Program OB 20, OB 21, OB 22, SYNCHRON block if they do not yet exist (see VOLUME 1)
 - Transfer the job buffer you created with the TF editor to PLC 1.
 - Program SEND DIRECT, that transfers the job buffer to the CP. Use the job number (ANR) you selected in the link block for CP 1.
 - Program RECEIVE ALL and SEND ALL that transfer the data between the CP and PLC.

Note:

The data block for the data source and for the data destination must exist.

- For PLC 2 (SERVER)
 - Program OB 20, OB 21, OB 22, SYNCHRON block if they do not yet exist (see VOLUME 1).
 - Program RECEIVE DIRECT that transfers the receive job to the CP. Select job number ANR+1, as selected in the link block for CP 2. In this standard function block, you specify the S5 address of the destination data block. In the destination data block, you must enter a job header consisting of three words.

DEST DATA BLOCK

Word 0	Service ID: B 0 0 1 h
Word 1	Reserved
Word 2	Length of the useful data
Word 3	Data

- Program SEND DIRECT that transfers the acknowledgement to the CP. Select job number ANR+1 as selected in the link block for CP 2. In this standard function block, you specify the S5 address of the source data block. In the source data block, you must enter a job header consisting of three words.

SOURCE DATA BLOCK

Word 0	Service ID: B 0 0 1 h
Word 1	Reserved
Word 2	Length of the useful data
Word 3	Data

- Program SEND ALL and RECEIVE ALL

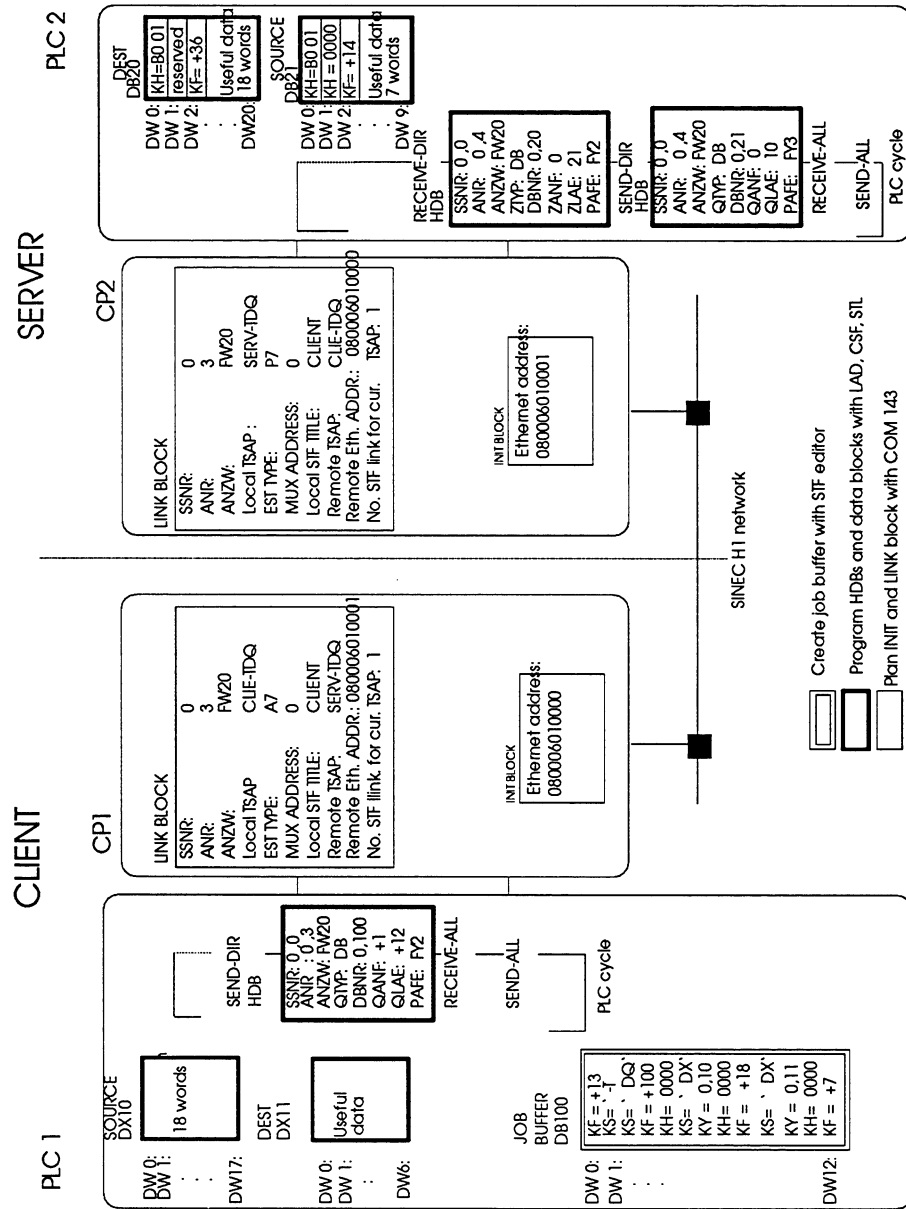


Fig. 11.23 Transparent Data Exchange with Acknowledgment (T-DQ) between two S5 Programmable Logic Controllers □

A TF Error Numbers used by the CP 143

A.1 Preface

A

This appendix describes the error numbers and the causes of errors that can occur when operating the CP 143. The error numbers consist of the parameters ERRCLS and ERRCOD of the AP protocol. There are two possible reasons for errors occurring:

- The CP 143 receives a job from the PLC (S5 as client) that is either incorrect or cannot be executed at the present time. The job is acknowledged negatively to the PLC (job terminated with error) without a TF-PDU being transferred.
- The CP 143 receives a TF indication (S5 as server) that is incorrect or cannot be executed. The CP 143 sends a reply with the corresponding error number back to the client.

The error number is made available to the user either via the test functions in COM 143 or at the client end by transferring it to the PLC. When the error number is transferred to the PLC, it is written to the second word after the programmed or configured condition codeword. It remains entered here until it is overwritten by the application or by a second error number. An error number is transferred to the PLC in the following situations:

- When an error occurs in a job from the PLC.
- When a TF reply is received with ERRCLS and ERRCOD.
- When the access result is negative in a read or write reply. In this case, the value 8240 H (response) or 3040 H (request) is added to the access result to create the error number.

A.2 Errors Arranged According to Service Groups

A.2.1 Non-Service Dependent Errors (NONS)

Errors occurring at the client end:

0000	No error
0881	Error message from the transport connection (iNA) or connection establishment was aborted
1041	There are no resources available for the current job (request field)
2041	The monitoring time between issuing the request and the reply has elapsed
2843	The negotiated PDU length is too short for the current job
3001	The job buffer is not long enough for all the parameters belonging to the job
3002	The job buffer contains an illegal opcode
9041	Protocol error: the received reply does not belong to the request
9044	TF service unknown
9045	TF service not negotiated when the application association was set up
A041	The application association is being terminated so that this request can no longer be processed
A042	The application association is no longer established

Errors occurring at the server end:

09C2	TF service not implemented or unknown
29C2	Protocol error: 1. The received indication in the AP header is PDULG22+PARLG+DATLG 2. The PDULG is too short for this type
29C3	Protocol error: 1. The transport buffer (iNA) is shorter than the AP header 2. The PDULG is shorter than the AP header. 3. The DATLG has an illegal value for this service
29C4	Protocol error: the parameter length in the AP header of the indication is inconsistent with the lengths in the PDU (e.g. the parameter length is shorter than the total of the name lengths)
51C1	TF service is not permitted in the current CP status (stop, unsynchronized)
61C1	Protocol error: in the AP header of the received indication, ROSCTR has an illegal value
61C2	Protocol error: in the AP header of the received indication, MODFR1 has an illegal value (0)
61C3	Protocol error: in the AP header of the received indication, MODFR2 has an illegal value (0)
69C1	Protocol error: in the AP header of the received indication, MPXADR has an illegal value
69C2	Protocol error: in the AP header of the received indication, PROTID has a value that is not supported
69C3	Protocol error: in the TF header of the received indication, COMCLS has a value that is not supported
69C4	Protocol error: in the TF header of the received indication, COMCOD has a value that is not supported

A

69C5	Protocol error: in the AP header of the received indication, TACTID has an illegal value (0)
69C6	Protocol error: in the AP header of the received indications, TASQNR has an illegal value (0)
69C7	The received indication was not negotiated when the application association was set up
9041	Protocol errors <ol style="list-style-type: none">1. DATLG is not equal to the length segment +4 in the AP header2. PARLG is not equal to 0 in the AP header, so that no service ID exists3. The field SPARE in the AP header has an illegal value4. In the AP header, the contents of ROSCTR and COMCLS are incompatible
91C1	Protocol error: in the AP header of the received indication, SGSQNR has an illegal value (0)
A042	The application association is no longer established

A.2.2 Errors in the General Services (GEN)

Get capabilities list:

Errors occurring at the server end:

- | | |
|------|--|
| 8305 | Start capability unknown
Identify virtual device:
Errors that occur at the client end |
| 3003 | The S5 address specified in the job buffer contains illegal parameters:
1. The code is not DB, DX or DA,
2. The data block number is 0 |
| 3026 | Error in the data transfer from the CP to the PLC (e.g. data block does not exist) |
| 3027 | The data from the acknowledgement transferred from the PLC are longer than the data area available in the PLC based on the specification in the job buffer

Errors that occur at the server end: |
| 29C5 | The negotiated PDU length is too short for the reply |

Status:

Errors occurring at the client end:

- | | |
|------|--|
| 3003 | The S5 address specified in the job buffer contains illegal parameters: 1. The code is not DB, DX or DA, 2. The data block number is 0 |
| 3026 | Error transferring data from the CP to the PLC (e.g. data block does not exist) |
| 3027 | The data transferred to the PLC in the acknowledgement are longer than the data area available in the PLC based on the specification in the job buffer |

Unsolicited status:

Errors occurring at the client end

3018 The local expansion in the job buffer is too long

Errors occurring at the server end:

8701 1. The S5 address for storing the data in the PLC is not configured

 2. The configured area for storing the data is not long enough

 3. Error in the data transfer from the CP to the PLC (e.g. data block does not exist)

Get name list

Errors occurring at the server end:

8201 The start object specified in the indication is unknown

8206 Protocol error: the specifications for class or for scope are illegal

8701 Access to expanded object class is not supported

8702 When accessing a list of domain-specific variables, the domain is unknown

A.2.3 Errors in Application Association Management (APPL)

Errors occurring at the client end:

9046 Connection establishment already active when a request to set up an application association is received

A

Errors occurring at the server end:

8000 Negative acknowledgement of terminate application association request

8001 Set up application association is not possible in the current connection status

8100 Application association already exists when "set up application link" job is sent

8800 1. A local expansion is not supported when setting up the link

2. The version number when setting up the link is 0

8801 Termination cannot be executed, since acknowledgements are still expected

8810 1. The number syntax = 0 when setting up

2. When setting up, no syntax was accepted

A.2.4 Errors in the Variable Services(VAR)

Errors occurring at the client end

- | | |
|------|--|
| 3003 | The S5 address specified in the job buffer contains illegal parameters: 1. The code is not DB, DX or DA, 2. The data block number is 0 |
| 3005 | The data are contained in the job buffer and the job buffer is not long enough |
| 3006 | The coding of the data type in the job buffer is incorrect |
| 3007 | The coding of the scope in the job buffer is incorrect |
| 3008 | The variables or group name in the job buffer is incorrect (e.g. too long) |
| 3010 | The variable in the job buffer is not programmed in the remote definitions |
| 3011 | Nothing is programmed in the remote definitions under the scope specified in the job buffer |
| 3012 | The variable in the job buffer is not programmed locally |
| 3013 | 1. Nothing is programmed locally under the scope specified in the job buffer
2. With local domain-specific variables, the domain is in an illegal status, since a load or delete function has not yet been completed with this domain (permitted status: ready, in use, D4, D5, D6, D7) |
| 3014 | Multiple variable access: the group specified in the job buffer does not exist or is empty |
| 3019 | The value of the "box identifier" specified in the job buffer does not correspond to the TF identifier for addressing via the format-free address |

- 3021 Conversion error converting floating point from TF format: the TF number is larger than the range that can be represented in MC 5
- 3022 Conversion error converting time or time and date from MC 5 format to TF format: the date in the PLC has illegal values (e.g.: 33.1.89 or 17:62).
- 3023 Conversion error converting time or time and date from TF format to MC 5 format: the time is greater than 24 hours or the date is after 31.12.2083.
- 3024 Conversion error with integer 8 or unsigned 8. The data word in the PLC contains a value for integer 8 less than -128 or greater than +127 or with unsigned 8 greater than 255
- 3025 Error in the data transfer from the PLC to the CP (e.g. data block does not exist)
- 3026 Error in the data transfer from the CP to the PLC (e.g. data block does not exist)
- 3030 Protocol: the parameter header of the received acknowledgement has illegal values
- 3032 Access to a variable is currently disabled (e.g. because it is being accessed via a different link)
- 3037 Data type inconsistent: the data type received with the acknowledgement does not correspond to the data type expected for the variable
- 3038 Protocol error in a received acknowledgement. The attributes belonging to the variables are inconsistent
- 3039 The interface via which the variable is to be accessed is not synchronized
- 303F Error in memory submodule

A

Errors occurring at the server end: (note: error numbers 824x are calculated at the client end from the access result)

- 29C5 The negotiated PDU length is too short for the reply
- 8201 The variable requested with request variable attributes does not exist
- 8241 There are not enough resources available on the CP or in the PLC (e.g. a data block for storing the variables content is too short or does not exist)
- 8242
1. The access to a variable is currently disabled (e.g. because it is being accessed via another link)
 2. With domain-specific variables, the domain is in an illegal state, since a load or delete function on this domain is not yet completed (permitted statuses: ready, in use, D4, D5, D6, D7)
- 8243 The variable can only be read and not written to
- 8244 Variable access via name: the variable does not exist (has not been programmed)
- 8245 Variable access via address: the variable does not exist (has not been programmed)
- 8247 Inconsistent data types:
1. The data type programmed for a variable does not match the type description received in the indication
 2. An error occurred converting from TF to MC 5 format or vice-versa
 3. When addressing using the format-free address, the type is not octet string ("OS") or the string length in the data field of the TF-PDU is inconsistent with the parameter "number" in the job buffer described in the S5 address.

- 8248
1. Protocol error: the received indication contains inconsistent attributes (e.g. the length of the variables data block is shorter than the number of type descriptions)
 2. When addressing using the format-free address, the scope is not VM or an illegal partial access has been attempted
- 8249
1. The type of variables access is not supported
 2. The interface via which the variable is to be accessed is not synchronized
 3. The CPU number of the format-free address is greater than 4 (CPU number 4)
- 9041
- Protocol error: the scope or the number of variables is incorrect

A

A.2.5 Errors in the Domain Services (DOM)

Errors occurring at the client end:

- | | |
|------|---|
| 3003 | The S5 address specified in the job buffer contains illegal parameters: 1. The code is not DB or DX, 2. The data block number is 0, the sum of the start address and length is greater than the maximum data block length |
| 3009 | The domain name in the job buffer is incorrect (e.g. too long) |
| 300A | The file name in the job buffer is incorrect (e.g. too long) |
| 300B | The logical partner name in the job buffer is incorrect (e.g. too long) |
| 300D | The resources length in the job buffer is incorrect (e.g. too long) |
| 3026 | Error transferring data from the CPU to the PLC (e.g. data block does not exist) |
| 3027 | The data transferred to the PLC from the acknowledgement are longer than the data area available in the PLC based on the specification in the job buffer |

Errors occurring at the server end:

- | | |
|------|--|
| 8000 | Negative acknowledgement accessing the memory submodule |
| 8102 | 1. The link to the file server was aborted
2. The link on which the load function was requested has broken down |
| 8205 | With load request: a domain with this name already exists |
| 8210 | 1. The file server indicated the end of the file, although data are still expected |

2. The domain to be loaded contains PLC blocks that already exist in another domain
3. The content of the domain is inconsistent (e.g. less variables than specified)
- 8301 1. The CP has insufficient buffer space (request fields) for the load function
2. There is not enough memory in the PLC for the blocks (compress)
- 8304 1. The serial 511 link to the PLC cannot be switched through
2. Physical error accessing the PLC via the serial link
3. Logical or protocol error accessing the PLC via the serial link
- 8304 There is not enough space left in the background memory to store the file
- 8305 The upload/load path does not exist or has incorrect syntax
- 8400 1. Protocol error handling the upload/load function with the file server
2. No load function was started for the domain specified in the indication
3. The received indication contains an HLM ID for which no upload sequence was started
- 8405 1. Access to the domain is not possible at present, since the memory module is already being accessed
2. The domain cannot be deleted because a variable is currently being accessed
- 8702 The domain specified in the indication does not exist

A

- 8703 The domain cannot be deleted
- 8710 The indication does not contain a domain name
- 8811 The syntax ID for the load data is not known to the file server
- A041 1. The link to the file server cannot be established
2. There is no server path specified in the indication (length of logical partner name = 0)

A.2.6 Errors in the Program Invocation Services

Errors occurring at the client end:

- | | |
|------|--|
| 3003 | The S5 address specified in the job buffer contains illegal parameters: 1. The code is not DB or DX, 2. The data block number is 0, 3. The sum of the start address and length is greater than the maximum data block length |
| 3009 | The domain name in the job buffer is incorrect (e.g. too long) |
| 300C | The program invocation name in the job buffer is incorrect (e.g. too long) |
| 3015 | The domain name is missing in the job buffer |
| 301F | The local stop (P-HL) is not permitted with this job number |
| 3026 | Error in the data transfer from the CP to the PLC (e.g. data block does not exist) |
| 3027 | The data transferred to the PLC in the acknowledgement are longer than the data area available in the PLC based on the specification in the job buffer |

Errors occurring at the server end:

- | | |
|------|--|
| 29C5 | The negotiated PDU length is too short for the reply |
| 8205 | Attempt to create a program invocation when a program invocation already exists |
| 8206 | Protocol error: the specifications for the PI name length or number of domains are illegal |
| 8300 | The PLC type is not supported (e.g. S5 100) |

A

- 8304 1. Error in the serial transfer via the 511 interface to the PLC
2. The serial interface is no longer plugged into the PLC-CPU programmed as master
- 8402 The service is not permissible in the current program invocation status (e.g. starting in the "unrunnable" status)
- 8405 Access to the program invocation is not possible at present, since it is already occupied by a different link
- 8702 1. The program invocation specified in the indication does not exist
2. A domain specified in the "create program invocation" job does not exist
- 8703 1. The PI status change was negatively acknowledged by the application in the PLC
2. The PI status change was acknowledged negatively (destructive) by the application in the PLC

A.2.7 Errors during Serial Transfer(SER)

Errors occurring at the client end:

- | | |
|------|---|
| 3003 | The S5 address specified in the job buffer contains illegal parameters: 1. The job is not DB or DX, 2. The data block number is 0, 3. The sum of the start address and length is greater than the maximum data block length |
| 3005 | The data are contained in the job buffer and the job buffer is not long enough |
| 3025 | Error transferring data from the PLC to the CP (e.g. data block does not exist) |
| 3026 | Error transferring data from the CP to the PLC (e.g. data block does not exist) |
| 3027 | The data transferred to the PLC from the acknowledgement are longer than the data area available in the PLC based on the specification in the job buffer |

Errors occurring at the server end:

- | | |
|------|---|
| 29C5 | The negotiated PDU length is too short for the reply |
| 2A01 | More data are to be read or written than permitted by the configured length or current length of the data block |
| 2A02 | Less data are to be read or written than permitted by the configured length |
| 2A04 | Access to the PLC is not possible, since no S5 address is configured |

A

A.2.8 Errors during Configuration (CONF)

- 3003 The S5 address specified in the job buffer contains illegal parameters: 1. The code is not DB or DX, 2. The data block number is 0, 3. The sum of the start address and length is greater than the maximum data block length
- 3004 The condition codeword specified in the job buffer contains illegal parameters: 1. The code is not FW, FY, DB or DX, 2. The address of the flag word or data word is outside the permitted area
- 3017 The job buffer contains an unknown configuration parameter (incorrect coding)

2. Error numbers in ascending order

The following pages list the error numbers in ascending order. The error numbers are followed by a cross reference to the service group in which the error can occur. The abbreviations are as follows:

NONS	Non-service dependent errors
GEN	Errors in general services
APPL	Errors in application association management
VAR	Errors in the variables services
DOM	Errors in the domain services
PI	Errors in the program invocation services
SER	Errors in serial transfer
CONF	Errors in configuration

Cross reference list

0000	NONS
0881	NONS
09C2	NONS
1041	NONS
2041	NONS
2843	NONS
29C2	NONS
29C3	NONS
29C4	NONS
29C5	GEN, VAR, PI, SER
2A01	SER
2A02	SER
2A04	SER
3001	NONS
3002	NONS
3003	GEN, VAR, DOM, PI, SER, CONF
3004	CONF
3005	VAR, SER
3006	VAR
3007	VAR

A

3008	VAR
3009	DOM, PI
300A	DOM
300B	DOM
300C	PI
300D	DOM
3010	VAR
3011	VAR
3012	VAR
3013	VAR
3014	VAR
3015	PI
3017	CONF
3018	GEN
301F	PI
3021	VAR
3022	VAR
3023	VAR
3024	VAR
3025	VAR, SER
3026	GEN, VAR, DOM, PI, SER

3027	GEN, DOM, PI, SER
3030	VAR
3032	VAR
3037	VAR
3038	VAR
3039	VAR
303F	VAR
51C1	NONS
61C1	NONS
61C2	NONS
61C3	NONS
69C1	NONS
69C2	NONS
69C3	NONS
69C4	NONS
69C5	NONS
69C6	NONS
69C7	NONS
8000	APPL, DOM
8001	APPL
8100	APPL

A

8102	DOM
8201	GEN, VAR
8205	DOM, PI
8210	DOM
8241	VAR
8242	VAR
8243	VAR
8244	VAR
8245	VAR
8247	VAR
8248	VAR
8249	VAR
8300	PI
8301	DOM
8304	DOM, PI
8305	GEN, DOM
8310	DOM, PI
8311	DOM
8400	DOM
8402	PI
8405	DOM, PI

8701	GEN
8702	GEN, DOM, PI
8703	DOM, PI
8710	DOM
8800	APPL
8801	APPL
8810	APPL
8811	DOM
9041	NONS, VAR
9044	NONS
9045	NONS
9046	APPL
91C1	NONS
A041	NONS, DOM
A042	NONS

□

A

Notes

B Protocol Implementation Conformance Statements (PICS)

The protocol implementation conformance statements (PICS) provide you with further information about implementing TF (scope and complexity) with the CP 143. This information is necessary if you want to implement a link to a non-SIMATIC system. Using the protocol implementation conformance statement, you can determine the following:

- which services the CP 143 supports and
- at which degree of complexity the supported services are available.

B

The CP 143 implements the following TF functions:

Explanation of the notation:

X means service implemented as client and server

S means service implemented as server

C means service implemented as client.

PICS 1: scope of services	CP 143 TF
Serial transfer read byte string write byte string transparent data exchange	X X X
Time functions request time set time transfer time	X
Variables services read write indicate request variables attributes	X X X S
Domain services initiate up/download up/download segment terminate up/download request upload sequence upload segment request upload sequence response	S S S S S S S
request download sequence request upload sequence load domain content store domain content	S S X X
delete domain content get domain attributes	X X

PICS 1: scope of services	CP 143
Program invocation services	
create program invocation	X
delete program invocation	X
start	X
stop	X
resume	X
reset	X
kill	X
get program invocation attributes	X
General services for virtual devices	
status of virtual device	X
information report	X
get name list	S
identify virtual device	X
get capability list	S
Application association management	
initiate application association	X
conclude application association	S
abort application association	X


B

PICS 3a: complexity		
Basic data types	a) Boolean b) bit string c) integer d) unsigned e) floating point f) octet string g) visible string h) time of day i) time and date	X
Arrays	Arrays of basic data types	X
Structures	Structures of basic data types	X
Named variable	All areas defined by TF	
Unnamed variable		
Transitions between hierarchical levels (= nesting)		2
Number of alternate accesses	Number of alternate access definitions in a job only as server	≥ 1
List of variables	Number of variables in a job	≥ 1
Relation: object description to access description in job	The object descriptions in the server have as maximum the complexity of the access definitions in the protocol	

C Abbreviations

A

ALI	Application Layer Interface
ANR	Job number (for handling blocks)
ANZW	Status word
AP	Automation protocol layers 5 to 7 of the ISO/OSI reference model
AS	Active star coupler
AS 511	511 interface, protocol for the communication between PLC and PG
ASCII	American Standard Code of Information Interchange

B

B	Block
BCD	Binary coded decimal
BE	Block end

C

CC	Central controller
CI	Cyclic interface
CIM	Computer Integrated Manufacturing
COM	Abbreviation for programming software for SIMATIC S5 CPs
COR	Coordination module
CP	Communications Processor
CPU	Central Processing Unit

C

CSF	Control System Flowchart, graphical representation of automation tasks with symbols
CSMA/CD	Carrier sense multiple access with collision detect
D	
DA	Destination Address
DB	Data block
DCE	Data Communication Equipment
DIN	Deutsches Institut für Normung (German Standards Institute)
DIR	Directory of data medium and files
DMA	Direct Memory Access
DOS	Operating system
DP	Distributed peripherals
DPR	Dual Port RAM
DTE	Data Terminal Equipment
DW	Data word (16 bits)
DX	Extended data block
E	
EG/EU	Expansion unit
EIA	Electronic Industries Association
EPROM	Erasable Programmable Read Only Memory
ET 200	Electronic Terminal 200

F	
F	Flag bit
FB	Function block
FD	Floppy Disk (data medium)
FD	Flag double word
FDDI	Fiber Distributed Data Interface
FDL	Fieldbus Data Link (subfunction of layer 2)
FDL2	Free layer 2 communications
FlexOs	Multitasking operating system
FMA	Fieldbus Management Layer
FMS	Fieldbus Message Specification (complying with PROFIBUS)
FO	Fibre Optic
FW	Flag word
FY	Flag byte
G	
GO	Global Object
GP	Global Peripherals
GPW	Global Peripheral Word
GRAPH 5	Software package for planning and programming sequence controllers
H	
HDB	Handling blocks

HSA	Highest Station Address
I	
IB	Input byte
IEC	International Electronics Commission
IEEE	Institution of Electrical and Electronic Engineers
IP	Intelligent peripheral module
ISO	International Standardization Organization
IW	Input word
K	
KOMI	Command interpreter
L	
LAD	Ladder Diagram, graphical representation of the automation task with symbols of a circuit diagram
LAN	Local Area Network
LB	Link block
LED	Light Emitting Diode
LEN	Length of a block
LLC	Logical Link Control
LLI	Lower Layer Interface
LSB	Least Significant Bit
M	
MAC	Medium Access Control
MAP	Manufacturing Automation Protocol

MMS	Manufacturing Message Specification
N	
NCM	Network and Communication Management
O	
OB	Organization block
OSI	Open System Interconnection
OW	Word from the extended peripherals
OY	Byte from the extended peripherals
P	
PAFE	Parameter assignment error
PB	Program block
PC	Personal Computer
PCI	Protocol Control Information (for coordinating a protocol)
PCP/M-86	Operating system Personal CP/M-86
PDU	Protocol Data Unit (frames consisting of PCI and SDU)
PG	Programmer
PI	Program invocation
PI	Process image
PII	Process image of the inputs
PIQ	Process image of the outputs
PLC	Programmable controller
PNO	PROFIBUS user organization

PRIO	Priority
PROFIBUS	PROcess Field BUS
PW	Peripheral word
PY	Peripheral byte
Q	
QB	Output byte
QW	Output word
R	
RAM	Random Access Memory
RLO	Result of logic operation (code bits)
RS	Recommended Standard
RS 485	EIA standard (multipoint capability) standard for electrical data transmission
S	
S5S5	Special type of communication PLC with PLC
SA	Source Address
SAP	Service Access Point. Logical interface points on the interface between the layers via which the PDUs are exchanged between service users.
SB	Sequence block
SDA	Send Data with Acknowledge
SDN	Send Data with No Acknowledge
SDU	Service Data Unit. Information about the service used and the user data contained within it.

SINEC	Siemens network architecture for coordination and engineering
SINEC AP	SINEC automation protocol
SINEC H1	SINEC bus system for industrial applications based on CSMA/CD
SINEC H1FO	SINEC bus system for industrial applications based on CSMA/CD with fiber optics
SINEC H3	SINEC bus system for industrial applications based on FDDI
SINEC L2	SINEC bus system for industrial applications based on PROFIBUS
SINEC L2-FO	SINEC bus system for industrial applications based on PROFIBUS with fiber optics
SINEC L2-FMS	SINEC bus system for industrial applications based on PROFIBUS with the FMS protocol
SINEC L2-DP	SINEC bus system for industrial applications based on PROFIBUS with the DP protocol
SINEC L2TF	SINEC bus system for industrial applications based on PROFIBUS with the TF protocol
SINEC TF	SINEC technological functions
SRD	Send and Request Data
SSNR	Interface number
STEP 5	Programming language for programming programmable controllers of the SIMATIC S5 range
STL	Statement List, STEP 5 method of representation as a series of mnemonics of PLC commands (complying with DIN 19239)

Sub-D	Subminiature D (connector)
SYM	Symbolic addressing
SYSID	Block for system identification
S5-KOMI	S5 command interpreter
S5-DOS/MT	S5 operating system based on FlexOS
T	
TF	Technological functions
TSAP	Transport Service Access Point
TSAP-ID	Transport Service Access Point Identifier
TSET	Set-up time
TSDR	Station delay
TSL	Slot-time
TTR	Target rotation time
TPDU	Transport Protocol Data Unit (size of the block of data transferred by the transport system)
TSDU	Transport Service Data Unit (size of the block of data transferred to the transport system with a job for transportation via a transport relation)
TSEL	Transport selector, term used as an alternative for TSAP-ID
V	
VFD	Virtual Field Device
VMD	Virtual Manufacturing Device
Z	
ZP	Cyclic peripherals□

D Index

A

Access	
to variables	4-8
Access rights	2-12
Addressing	
free format	4-36
variables	4-8
ANR	8-27
configuring	8-4
ANZW	8-12
configuring	8-4
extended for TF	3-3
specifying /selecting	3-12
Application	
See application program	
Application association	6-3
configuring	8-3
configuring name	8-7
configuring the type of establishment	8-6
definition	6-2
establishment/termination	2-15
handling with S5 PLC	2-18
management	2-34
management (overview)	2-3
object	2-7
object description	2-15
with S5 PLC	2-15
Application association list	8-23
Application association management	6-2
overview	2-3
Application associations	
number of jobs	6-3
type of connection establishment	6-3
Application program	
TF architecture	2-5



B

Basic configuration	8-3
Box ID	
See group name	

C

Capability	5-10, 5-59
Capability list	5-28
CIM	
support by TF	1-3
Client	
See also client/server association	
Client interface	
calling TF services	3-6
messages	3-10
sequence	3-10
Client-/Server association	
functions in S5	2-20
Client/server association	
example	2-19
principle	2-19
Clock	
accuracy	6-49
configuration	6-38
how it functions	6-36
reading/setting time	6-40
restrictions	6-48
setting/reading	6-45
status transitions	6-36
tips for application	6-48
Clock frame	6-28
Clock master	6-31
Clock services	6-28
accuracy	6-35
delay time	6-33
overview	2-4
synchronization	6-31
undefined time	6-33
update time	6-33

Clock slave	6-31
Communication	
message-oriented	2-2
open	2-2
Compress	
See also job buffer	
See also PLC	
Compress memory	5-8
Configuration	
documenting	8-23
Configuration job	6-23
Configuring	
local variables	3-14
Connection	
special	6-9
Connection establishment	6-5
dynamic	6-4
layer 4	6-5
layer 7	6-5
static passive	6-3
static,active	6-3
Connection termination	6-8

D

Data link blocks	6-2
configuring	8-4
Data type	
bit string	4-43
Boolean	4-41
floating point	4-41
integer	4-41
octet string	4-43
time and date	4-41
unsigned	4-41
visible string	4-44
Database transfer	8-25
Documentation	8-23
Domain	
assignment to CPU	5-14
attributes	5-22

D

block allocation	5-14
definition in S5	5-2
dynamic	5-30
generating	5-2
in S5 PLC	2-10
managing on the CP	5-14
object	2-7
object description	2-10
services	2-3, 2-23, 2-27, 5-1 - 5-64
static	5-30
Domain list	5-64
Domain services	
check list for application	5-4
delete domain services	5-20
See also domain	
example	11-42
get domain attributes	5-22
load domain content	5-8
modes	5-5
store domain content	5-15
third party association	5-5
Dual-port RAM	3-6
E	
Ethernet address	
remote (configuring)	8-7
Example programs	11-1
F	
Fileserver appl. ass.	
configuring and establishing	5-14, 8-20
type of establishment	8-20
Free format address	4-33
G	
Group name	4-27, 8-17, 8-19
Groups	
defining	8-19

H

Handling block

See HDB

HDB

address information	3-2
application	3-4
call parameter description	3-11
example of application	3-4
overview	3-2
parameter assignment	3-10
tool to support parameter assignment	3-10

I

Installation 5-45

Insufficient memory

loading domain 5-14

Interface number

See SSNR

J

Job buffer	3-3, 3-6
compressing	10-11
creating with supporting tools	3-6
creating with the editor	10-3
general section	3-8
structure	3-6, 10-3

Job number

See ANR

Job status 8-35

L

Language standardization

aims 1-3

Link establishment 6-3

Link flag 8-27

Load sequence 5-6

D

M

MAC	2-6
MAC address	
See Ethernet address	
MAP	
SINEC integration	2-2
MMS	
basis of definition for TF	1-3
Model	
See TF model	
Multiplex address	
See MUX address	
Multiprocessor mode	5-44
PI view	5-31
VMD configuration	8-21
MUX address	
configuring	8-7
meaning	2-15

N

NCM	10-2
Nesting level	8-17
Non-open services	7-1

O

Open communication	
See communication	

P

PGLOAD	3-7, 9-1
function	9-2
host computer functions	9-6, 9-11
system configuration	9-7
TF architecture	2-5
transfer functions	9-6, 9-9
PI	
in an S5 PLC	2-11

object	2-7
object description	2-11
service description	2-29
services	2-3, 2-28, 5-1 - 5-64
status coding	5-43
status indication	5-31
status management	5-31
system PI	2-11
user PI	2-11
PI services	5-29
check list for application	5-32
create PI (server)	5-50
example	11-42
Get PI attributes	5-60
kill PI (client)	5-55
local program stop (client)	5-55
See also PI	
reset (client)	5-55
resume PI (client)	5-55
server interface	5-39
server function	5-29
standard function block	5-42
start PI (client)	5-55
stop PI (client)	5-55
PLC	
start program	9-12
delete	9-11
in client/server association	2-19
load	9-11
save	9-11
start/stop	9-11
stop program	9-12
virtual	2-9
PLC-CP link	
principle	3-2
Predefined connection	6-9
Printout	
controlling	8-24
documentation	8-23
Program invocation	
See PI	

D

Programmable logic controller	2-19
Protocol	2-2
R	
RECEIVE-ALL	3-2, 3-4, 3-14
REQUEST EDITOR	10-1
functions and mode of operation	10-2
S	
Scope	2-12
configuring remote variables	8-16
domain-specific	4-4
example	4-6
link-specific	4-5
See also variable	
variable access	4-8
VMD specific	4-4
with free format read/write	4-36
SEND-ALL	3-2, 3-4, 3-14
SEND-direct	3-4
Serial transfer	2-37, 7-1
example	11-68
job header	7-21
overview	2-4
read byte string (client)	7-4
read/write byte string (server)	7-13
service descriptions	2-38
transparent data exchange (Client)	7-17
transparent data exchange (server)	7-21
write byte string (client)	7-7
Server	
See also client/server association	
Server function	3-14
activating	3-14
Server interface	3-14
Server list	8-23
Services	
See also TF services	
SINEC H1/H1FO	

overview	1-3
SINEC TF	
advantages	1-3
architecture	2-5
communications model	2-5
Single status	
See test	
SSNR	8-13, 8-22, 8-27
configuring	8-4
Standard function block	
for PI services	5-31
PI services	5-42
Start-up	5-45
START/STOP	5-34
Status all	
See test	
Status diagram	
PI	5-33
Status of a link	8-27, 8-35
Status transitions	5-30
Structures	4-3
Supplementary services	6-1
System PI	2-11, 5-34, 5-44


D
T

Test	8-26
JOB TEST	8-35
single status	8-33
status all	8-27
TF interface	3-1 - 3-16
configuring and testing	8-1
testing	8-26
TF job	3-3
TF link	3-2
establishment/termination	2-15
handling with S5 PLC	2-18
with S5 PLC	2-15
TF model	
introduction	2-1 - 2-38
TF object	

dynamic objects	2-8
static object	2-8
types of object	2-8
TF objects and TF services	2-7
TF services	
call on the client interface	3-6
example programs	11-1
overview	2-3
selecting	2-21
under PGLoad	9-4
Third party association	2-25 - 2-26, 5-5
Timeout	3-9
Tools	
PGLoad	9-1
REQUEST EDITOR	10-1
Trace buffer	8-34
Trace functions	8-34
Transfer functions	8-25
Transparent data exchange	
evaluating status bits	7-24
TSAP	
local (configuring)	8-5
remote (configuring)	8-7
Type description	4-40
complexity	4-3
free format read/write	4-36
PLC as client	4-2
PLC as server	4-2
read variable	4-12
See variable	
write variable	4-19
Type selection screen	10-12
U	
Upload sequence	5-7
User groups	1-10
User memory module	
See also EPROM	
User PI	2-11, 5-35
statuses	5-35

V**Variable**

access protection	2-14
access right	2-12
basics of the services	4-2
characteristics	2-12
condition code word address	2-13
configuring with scope	4-4 - 4-5
example of defining	11-5
example of local configuration	8-14
in an S5 PLC	2-13
interface number	2-13
Local	2-13, 4-2, 8-3
local (configuring)	8-10, 8-17
name	2-12
object	2-7
object description	2-12
remote	2-13, 4-2, 8-3
remote (configuring)	8-16
S5 address	2-13
scope	2-12, 4-4
service description	2-31
services	2-3, 2-30, 4-1 - 4-44
See also type description	
Variable description	2-12
Variable name	2-12
Variable services	
check list for application	4-9
example	11-3
free format read/write	4-33
information report	4-8, 4-26
See variable	
write (client)	4-17
Variable type	
array	2-14
record	2-14
standard data type	2-14
VMD	
configuring	8-21
in an S5 PLC	2-9

D

object	2-7
object description	2-9
services	2-3, 2-21
variables editor	8-17
VMD list	8-23
VMD services	6-10
identify VMD	6-19
unsolicited VMD status	6-17
See also VMD	
VMD status	6-11

□

E Further Reading

- /1/ Wege zur offenen Kommunikation
Das ISO-Referenzmodell im Umfeld der Kommunikation
Siemens AG DÖA PM Order no.: U 1474-J-Z72-11984
(only available in German)
- /2/ [ISO/IEC 9506-1] Information Processing Systems Open Systems Interconnection- Manufacturing Message specification, Part 1: Service Definition
- /3/ Kerner H.
Rechnernetze nach OSI
ADDISON-WESLEY 1992
ISBN 3-89319-408-8 (only available in German)
- /4/ Instructions
Installation of SINEC H1 Local Area Network
SIEMENS AG, Order no.: AR 463-220
- /5/ Instructions
Installation of SINEC H1FO Local Area Network
SIEMENS AG, Order no.: AR 464-220
- /6/ SINEC TF User Interface
User Interface for the SINEC Technological Functions
SIEMENS AG, Order no.: 6GK1971-1AB00-0AA1 Release 02
- /7/ Handling blocks are described in the following:
- for S5-115 as part of the device manual
Order no.: 6 ES 5998-3-UFX 1 for CPU 945
Order no.: 6 ES 5998-0-UFX 3 for CPU 941 - CPU 944
- for S5-135 can be ordered as HDB software + description
Order no.: 6 ES 5842-7-CB 01 for CPU 928A/B - CPU 948
- for S5-155 can be ordered as HDB software + description
Order no.: 6 ES 5846-7-CA 01 for CPU 946 / 947

E

NOTIZEN

SIEMENS AG
 AUT 933
 Siemensallee 84
 76187 Karlsruhe
 Federal Republic of Germany

Vorschläge Korrekturen
 Suggetions Corrections

Für Druckschrift bzw. Handbuch
 For instruction or manual

Titel/title

SINEC CP 143 TF mit NCM COM 143 TF

Absender/From - Name/Name

Bestell-Nr./Order No.

6GK1970-1AC43-0AA1

Firma/Dienststelle - Company/Department

Anschrift/Address

Telefon/Telephone

Vorschläge und/oder Korrekturen
 Suggestions/Corrections

Sollten Sie beim Lesen dieser Unterlage auf Druckfehler gestoßen sein, so bitten wir Sie, uns diese mitzuteilen. Ebenso sind wir für Anregungen, Hinweise und Verbesserungsvorschläge dankbar.

Should you come across any printing errors when reading this publication, we would ask you to inform us using this form. We would also welcome any suggestions you may have for improvement.

Bitte die Bestell-Nr. der betreffenden Druckschrift oder des Handbuches oben eintragen!

Please fill in the order no. of the affected document!

SIEMENS AG
AUT 933
Siemensallee 84
76187 Karlsruhe
Federal Republic of Germany

Corrections
Correcciones

Pour l'imprimé ou le manuel
Para folleto o manual

Titre/título

SINEC CP 143 TF mit NCM COM 143 TF

Expéditeur/Expeditor - Nom/Nombre y apellido

N° de réf./N° de ped.

6GK1970-1AC43-0AA1

Société/Service - Empresa/Seccion

Adresse/Direction

Téléphone/Téléfono

Propositions ou corrections
Propuestas y/o correcciones

Si, lors de la lecture de ce document, vous trouvez des fautes d'imprimerie, nous vous prions de nous en faire part dans ce formulaire. Nous recevrons aussi avec reconnaissance vos suggestions, remarques et propositions d'amélioration.

Si encuentra Usted erratas de imprenta,, por favor, infómenos utilizando este formulario. Le rogamos que nos communique también las reclamaciones, indicaciones, y propuestas de mejoramiento.

Indiquez s.v.p. le n° de référence de l'imprimé ou de manuel concerné !

Indique por favor el N° de pedido del folleto o del manual respectivo!