SIEMENS

SINEC

CP 1430 TF with COM 1430 TF

Volume 2 of 2

6GK1970-1TA43-0AA1 C79000-G8976-C056 Release 01

Contents in Volume 1 of 2

SIEMENS

Siemens Aktiengesellschaft

Elektronikwerk Karlsruhe
Printed in the Federal Republic of Germany

Contents

I Basic Information

II COM 1430 TF Functions

III Reference Section TF Services

IV Appendix

❑

Notes

1 Introduction

1.1 How to Use the Manual

1.1.1 Recommendations

The manual consists of two volumes and a supplement.

This second volume of the manual "CP 1430 TF with COM 1430 TF" deals with the protocol and the services for open, heterogeneous communication with the CP 1430 TF communications processor.

Read the following chapters if...

...you want an overview of the areas of application of the CP 1430 TF and how it functions.	➡	- Chapters 1, 2 and 4: Performance and Technical Data in Volume 1 - Chapters 1 and 2 in Volume 2
...you want to create PLC programs and require communication services.	➡	- Chapter 2/Volume 1: What Types of Communication are Available? - Chapter 3 in Volume 1 and Volume 2: Principles of the HDB Interface - Reference Section TF Services in Volume 2
...you want to configure the CP for transport services.	➡	- Volume 1
...you want to configure the CP for TF services.	➡	in Volume 2: - Chapter 4: Job Configuration and Test - Chapter 5 Tools PG Load and Chapter 6 Request Editor - Appendix A: Example
...you want to install and start up the CP.	➡	Chapter 4/Volume 1: Installation, Start-Stop, Connecting a PG, Addressing.

1.1.2 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.**

M x-y This note in the margin indicates the number of a dialog which you can refer to in the 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 of the HDBs for the programmable logic controllers.

Training courses
Siemens provides comprehensive training opportunities for SINEC users.

For more detailed information contact your Siemens office.

Order numbers of the products mentioned in this manual can be found in the appropriate catalogs.

1.2 Significance of the SINEC Technological Functions

Overview

The SINEC technological functions (TF) form the application interface for communication in a heterogeneous automation network. They provide the user with services to allow problem-free interaction between different automation components (for example PLCs, 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, minimizing 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 SINEC 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.

- Networks can interconnected be implemented without problems.

- By using SINEC TF, the time and expense of software development can be greatly reduced.❑

Notes

I Basic Information

2 The TF Model and the TF Services

Topics in this Chapter

To allow you to use the TF services of the CP 1430 TF, this chapter explains the communications model and the range of services of the SINEC technological functions.

If you are familiar with the TF communications 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 1430 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 functions actually implemented on the CP 1430 TF 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 subnets 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 the non-open services. The latter are only available in the SINEC TF protocol architecture described here.

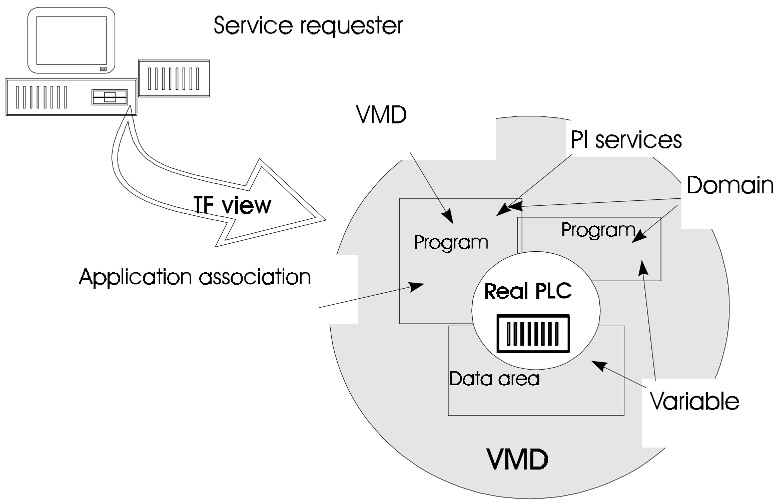


Fig. 2.1: The TF/VMD View of the Real PLC

TF services

➢ VMD services

With the services for the **V**irtual **M**anufacturing **D**evice (VMD), information about the characteristics and the status of a VMD can be requested (which services the device can perform, which objects exist etc.).

➢ Application association management

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

➢ 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).

➢ 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 file server 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

Additional Services only Available for SINEC H1 (non-open services)

➢ Serial transfer

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

2.2 SINEC TF Communication Model

2.2.1 Overview of the Architecture

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 services via the TF interface (PG Load).

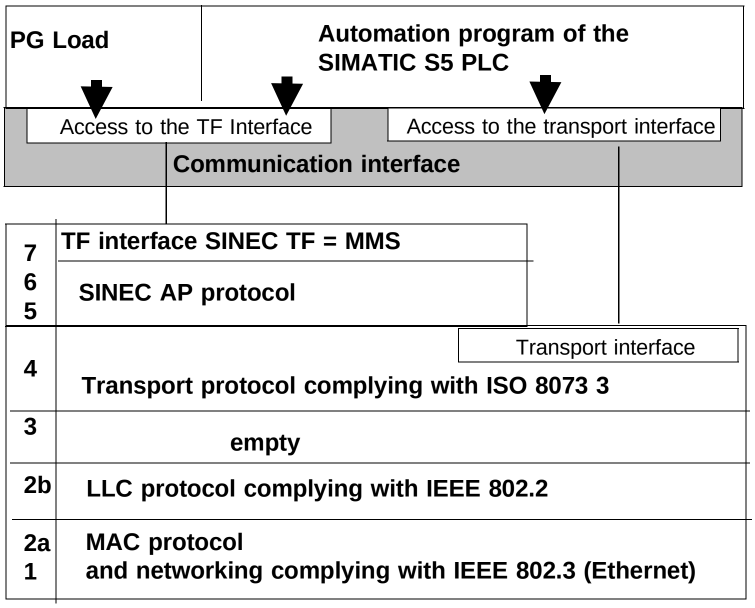


Fig. 2.2: Interface and Protocol Profile of the CP 1430 TF Communications Processor

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 modeled on the MAP 3.0/MMS application protocol. This means that the user software also runs identically on the MAP protocol stack providing that conformity rules are adhered to.

PG Load for

- Loading/saving/deleting domains
- Host 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

LLC: Logical Link Control

2.2.2 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 using application associations.

Prior to the communication, the user must specify which automation task is to be addressed via which application association, for example, the application association and the name of the variable to be read must be specified.

Initiation, Use and Termination of Application Associations

During the establishment phase, an initiate request is sent to the remote application process (initiate service). The initiate 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 application association as well as any other options required.

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

An application association is terminated by a conclude function. Following termination, data exchange can only be resumed after the application association has been initiated (established) again.

SIMATIC S5 and Application Associations

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

Access to communication in SIMATIC S5 is achieved using the interface number and job number.

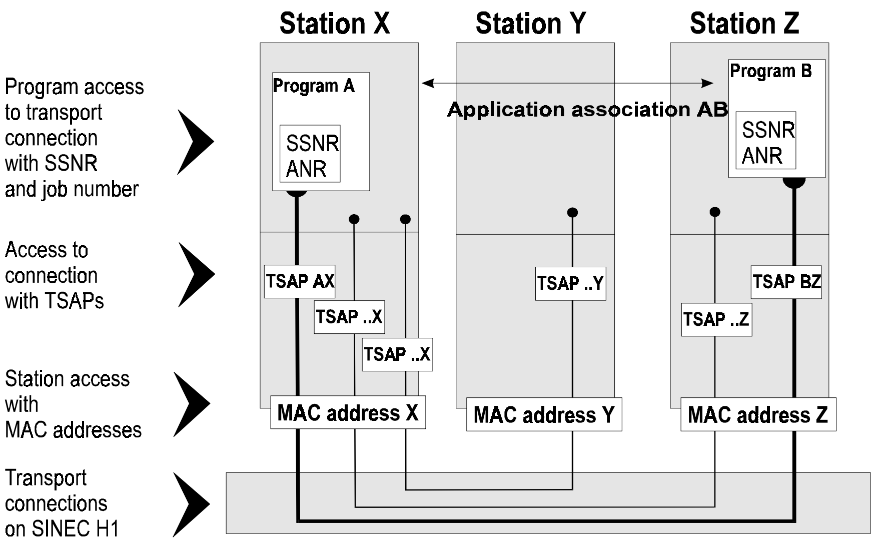


Fig. 2.3: Access by Application Associations to the Transport Services

Just as the with transport connections, the parameters for an application association are stored in connection blocks and in the basic initialization data on stations X and Z. Taking the situation as illustrated in Figure 2.3 , parameters would be stored as shown below:

Connection block

Connection parameters	**Local parameters**
Application assoc. name	SSNR
Rem. station address=Z	A-NR
Local TSAP=AX	
Remote TSAP=BZ	

SYSID block

Local station address=X

Fig. 2.4: Connection Block and SYSID Block for Application Associations

Handling Application Associations with SIMATIC S5

In the client and server role, the CP 1430 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 by the CP 1430 in the main without being triggered by the user program. The required parameters for executing the services are entered with COM 1430 when defining the application associations and saved on the communications processor. Depending on the type of connection (static or dynamic), the application associations are initiated when the modules are started or when a job is initiated for productive communication.

An application association can only be initiated after the corresponding "pneumatic post (transport) connection" 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.

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".

2.2.3 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:

➢ 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.

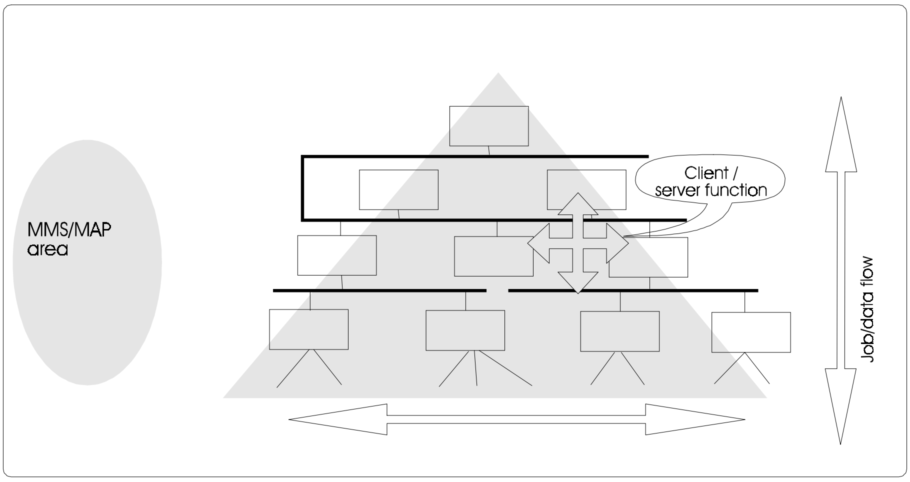


Fig. 2.5: Hierarchic Structure of the CIM Network; Devices with Client and Server Functions

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.

2.2.4 VMD Services

Meaning

The virtual manufacturing device (VMD) represents a standardized image of a programmable logic controller in the form of a model. It is described by the objects it contains and the characteristics of the physical device, that it models with a standard view.

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 (unsolicited). The information can be processed further on the client, for example to provide an overview of the whole system status in a control room.

SIMATIC S5 and VMD

The essential function of the communications processor (CP) is to model the VMD on the programmable logic controller (PLC) and to execute the VMD services.

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.

Overview

➢ 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 IDLE) 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 1430 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 1430 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 1430 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 1430 without support of the S5 program. The reply information sent by the CP 1430 might include, for example, a list of all communications objects (PI, domain, variable) defined on the server and managed by it.

2.2.5 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

➢ 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 acknowledgment.

➢ 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 acknowledgment tells the client whether the service was successful or not.

➢ 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 1430 package.

➢ Get Variable Attributes
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 acknowledgment.

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 1430) in the acknowledgment.

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 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 1430 TF configuration tool ("VMD variables editor") and with the application association editor as local application association-specific variables. 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 PG Load tool (component of COM 1430 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 1430 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.

2.2.6 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) and that a file server can also be included as a "third party" communications partner.

Domains can be the following:

➢ Logical management units for variables
A domain as a logical management unit defines a scope 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. there meaning is only known to the applications which use the domain (e.g. not to a file server).

SIMATIC S5 and Domains

A domain normally consists of "STEP 5" blocks, saved 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 Variables). In addition to these "loadable" domains, there is also a "static" domain SIMATIC_S5.

Up to eight domains can be loaded on a SIMATIC S5 programmable logic controller in addition to the static SIMATIC_S5 domain.

The static domain can be used as follows:

1. Archiving
   The first time that the "SIMATIC_S5" domain is uploaded, the entire PLC program is saved, in other words, all blocks are grouped together in this domain and uploaded. This domain is then available on the PG as a loadable domain. If a user domain exists, "SIMATIC_S5" is then a dummy domain without data or program.

2. Using PI services **without** domain services
   You can use the "SIMATIC_S5" dummy domain. This means that you do **not** first have to load a domain before you can use the PI services.

In a SIMATIC S5 PLC, you can load up to eight domains in addition to the SIMATIC S5 static domain.

Overview

The following TF domain services are supported by the CP 1430 TF:

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

➢ Upload (server function)
  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 (client and server function)
  With the TF "load domain content" service, a client requests a server to load a domain via the application association. The domain is located in a file stored on the TF file server.

➢ Store domain content (client and server function)
  With the TF "store domain content" service, a client requests a server to save a loaded domain via the application association in a file on the TF file server.

➢ Delete domain content (client and server function)
  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 (client and server function)
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 acknowledgment (e.g. whether the domain can be deleted or whether it is used by a program invocation).

Initiative

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

In this case the SIMATIC S5 PLC acting as the client triggers the loading of domains from a TF file server 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 a host

In the simplest case, the domain services are triggered by a device known as the host device. The S5-DOS program file (name: xxxxxxST.S5D) contains the program and data blocks to be loaded. TF handles these blocks as a domain.

The domains must first be defined using the COM 1430 tool PG Load (transfer function ’create load file’).

In this configuration, the host can be a PG but not a SIMATIC PLC.

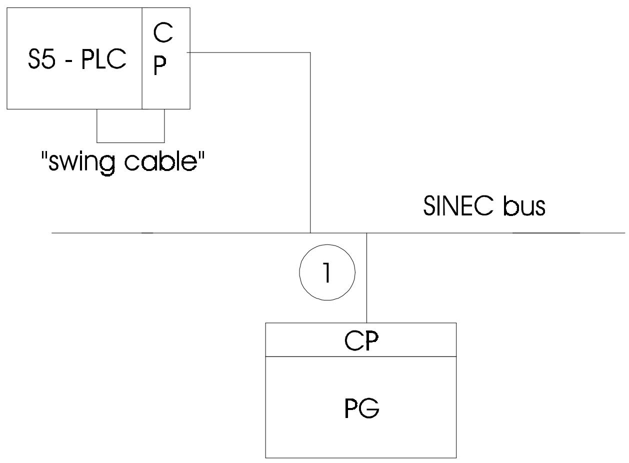


Fig. 2.6: Domain Services with the PG - ’Download/Upload’

The “download” domain service transfers the domain to the PLC. The “upload” domain service transfers the domain from the PLC to the host for archiving. A domain on the PLC can be deleted with the “delete domain” service.

2.) Third party association with PLC, host and file server

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 "file server" stores the files that will be loaded later as domains.

In this case, the host can be a PG or a SIMATIC PLC.

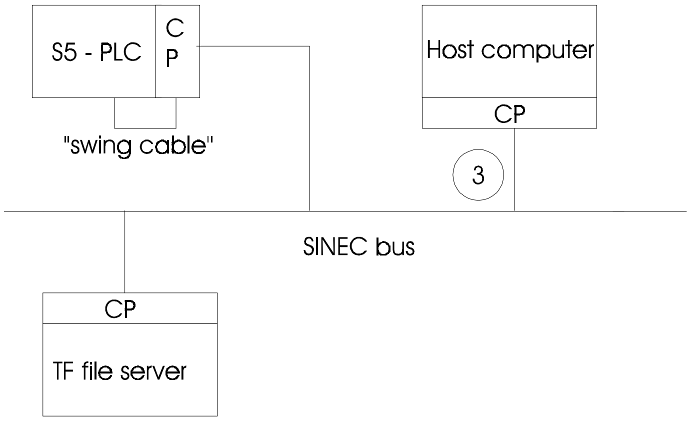


Fig. 2.7: Third-Party Association with PG and File Server

The host can now request the CP 1430 to load a particular domain from the TF file server into the programmable controller. The host can also request the CP 1430 TF to save a loaded domain on a file server or to delete the domain on the PLC.

This configuration is known as a "third party association" since the host requests the manufacturing device (PLC) to load data from or on a third station.

2.2.7 Program Invocation (PI)

Purpose

A program invocation is the grouping of all domains into an executable program. The PI services allow these programs to be controlled. PIs can, for example, be started or stopped.

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 user program loaded on the programmable controller.

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

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.

PI services can be executed by a local programmable logic controller both on itself and on a remote partner. 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 1430 TF

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

Overview

➢ 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 acknowledgment (e.g. whether the program invocation can be deleted or the current status of the program invocation).

2.2.8 Serial Transfer

Meaning

For simple data transfer, the CP 1430 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 modeled on MMS services.

Advantages and Restrictions

The serial transfer services allow the greatest freedom in configuring application associations. The users (programmers) 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 and the timed and logical monitoring of TF jobs.

Overview

➢ 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 acknowledgment 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 acknowledgment. The client can decide whether or not an acknowledgment is required.

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

Fig. 2.8: Overview of the Services of Serial Transfer ❑

3 The TF Interface on the CP 1430 TF

Topics in this Chapter

Volume 1 (Configuring and Programming CP 1430 TF Communication) explains the basic principles of the PLC-CP connection. This chapter looks at the PLC-CP connection in terms of the TF interface.

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

Further Information

For more information about related topics:

➢ Read Chapter 4 in this Volume about configuring application associations.

➢ Read /7/ about parameter assignment and the use of handling blocks.

3.1 The Principle of the TF Connection PLC - CP

3.1.1 Differences in Handling the Transport and TF Interface

On the TF interface, just as on the transport interface, you use handling blocks (HDBs) for transferring jobs and controlling the CP 1430 TF.

Compared with the transport interface described in Volume 1, note the following differences in the handling of the TF interface:

➢ Job buffers
The TF service associated with a send job is specified in a job buffer. The job buffers are referenced in the standard handling blocks and transferred to the CP 1430 TF via the dual-port RAM.

➢ Structured data
Data are not transferred as byte sequences but in a structured form. The structure information is located either in the configuration in the CP database or, with simple variables, in the job buffer.

➢ Extended status word
The status word for the TF interface is extended by a third word for TF error IDs. 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 3-15).

3.1.2 Overview of the HDB Blocks when Using the TF Interface

The following system calls (HDBs) are available:

➢ SEND
The SEND block is used to transfer a job (with or without user data) to the CP 1430 TF and to trigger an MMS/TF service.

➢ SEND_ALL HDB
The SEND_ALL HDB is used to trigger data transfer from the PLC and CP.

Exception with the S5-115U:

This PLC does not have its own SEND_ALL HDB. If the SEND block is assigned the job number 0, it acts as a SEND_ALL block.

☞ **The SEND_ALL must be triggered cyclically on the client and server, otherwise there is no connection established if there is a problem on an application association.**

➢ RECEIVE
The RECEIVE block is used on the CP 1430 TF for the RECEIVE ALL function, i.e. for transfer of data from the CP 1430 TF to the PLC.

➢ RECEIVE_ALL HDB
The RECEIVE_ALL HDB is used to trigger data acceptance between the PLC and CP.

➢ Exception with the S5-115U:

This PLC does not have its own RECEIVE_ALL HDB. If the RECEIVE block is assigned the value 0, it acts as a RECEIVE_ALL block.

☞ **The RECEIVE_ALL must be triggered cyclically on the client and server, otherwise there is no connection established if there is a problem on an application association.**

➢ RESET
The RESET block resets the application association assigned to the ANR (job number).

➢ CONTROL
The CONTROL block is used to query the status of a job.

➢ SYNCHRON
The SYNCHRON block establishes the synchronization between the PLC and CP 1430 TF.

The handling blocks are incorporated in special function blocks on the SIMATIC S5 PLCs. Refer to the following table to see which block contains which HDB.

HDB	S5-115U/H	S5-135 and S5-155U/H	
	CPU 942 CPU 942R CPU 943 CPU 944 CPU 945	CPU 922 CPU 928 CPU 948 CPU 948R	CPU 946/947 CPU 946/947R
SEND	FB244	FB120	FB120
RECEIVE	FB245	FB121	FB121
CONTROL	FB247	FB123	FB123
RESET	FB248	FB124	FB124
SYNCHRON	FB249	FB125	FB125
SEND_ALL	FB244 (ANR=0)	FB126	FB126
RECEIVE_ALL	FB245 (ANR=0)	FB127	FB127

Table 3.1: HDB Numbers in the Various CPU Types

☞ **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. and in /7/.**

3.1.3 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 1430 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 1430, the TF-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=$00X2_H$).

Read Variable Job (compare Fig. 3.1)

With the read job, the TF-PDU for the specific job is transmitted to the partner (PLC 2). The CP 1430 TF in PLC 2 evaluates the TF-PDU and requests the data from the CPU via the background communication (SEND-ALL).

Once the TF-PDU is filled with the required data by the CP 1430 TF, it is transmitted with a positive acknowledgment to the caller (PLC 1 = service initiator). The data is then transferred to the CPU using the background communication (RECEIVE-ALL) and is available on the call interface.

- ➢ Positive acknowledgment:
  After successful transfer of the data, "job completed without error" (=$00X4_H$) is entered in the SEND-direct ANZW and the job can be triggered again.

- ➢ Negative acknowledgment:
  If a negative acknowledgment is received from the remote partner or if the data cannot be transferred to the PLC, "job completed with error" is entered in the SEND-direct ANZW and "remote error" (= $09X8_H$) 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.

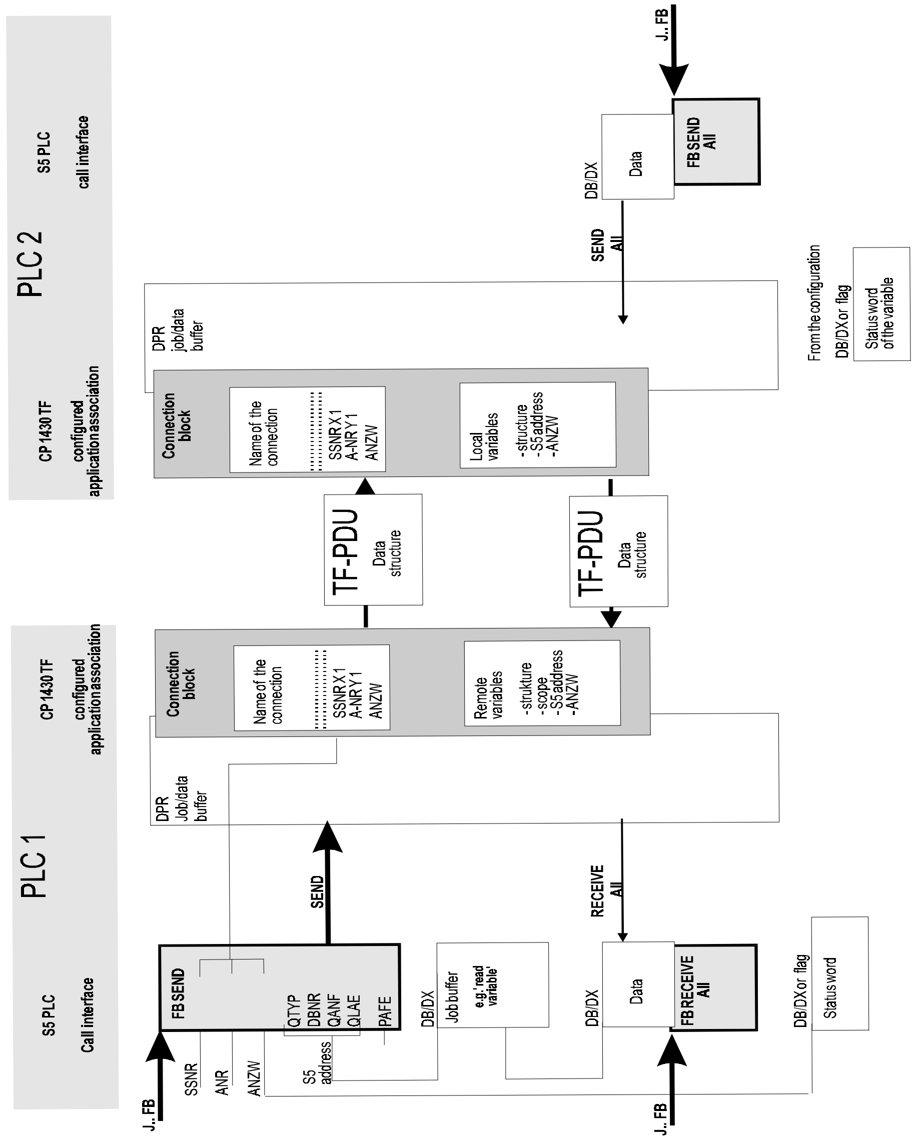


Fig. 3.1: TF Principle of Communication - Exampe Read Variable

Write Variable Job

With a write job, the data for transmission is requested by the CP 1430 TF from PLC 1 using the background communication (SEND-ALL), entered in the TF-PDU and sent to PLC 2

When the data arrives in PLC 2, it is compared with the configured data description. Following this, the data is made available to the CPU by the background communication (RECEIVE-ALL) and transferred to the appropriate data block in the PLC.

The job is then acknowledged on the call interface. The initiator of the service evaluates the acknowledgment and terminates the job accordingly:

➢ Positive acknowledgment:
the job was completed successfully and "job completed without error" is entered in the SEND-direct ANZW= $00X4_H$, allowing the job to be triggered again.

➢ Negative acknowledgment:
the job could not be executed. "Job completed with error" is entered in the SEND-direct ANZW and "remote error" =$09X8_H$, can be found in the TF error status word (ERRCLS/ERRCOD). The job can then 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.

User 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 TF services).

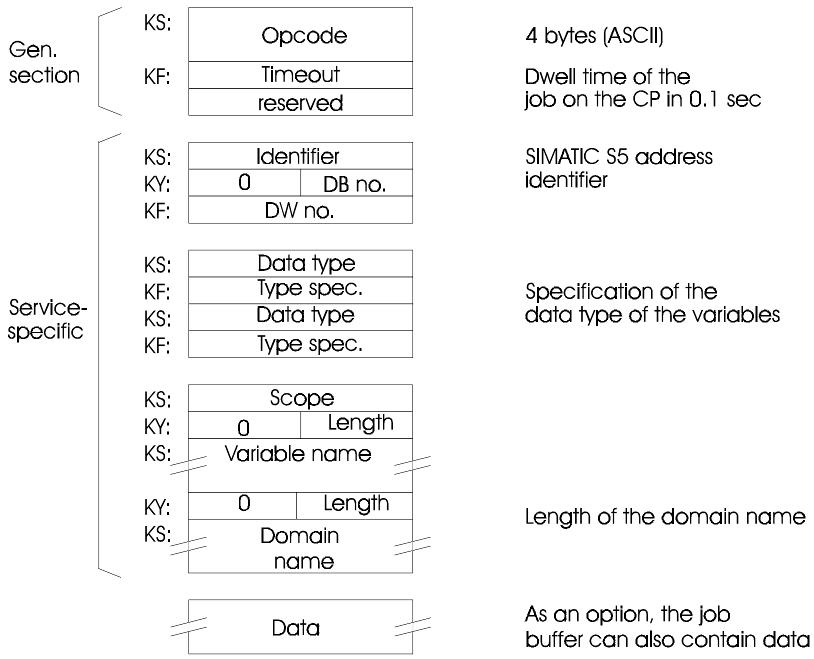


Fig. 3.2: Structure of the Job Buffer - Example Variable Services

Special Features

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

It is also possible to configure certain parameters on the CP 1430 TF (local and remote definition, see Chapter 4 Configuring and Testing or 4.3.2 Edit | Application Associations) 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 1430 TF or in the COM 1430 tool PG Load.**

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.

Opcode	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 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 without acknowledgment
B-WI	Query byte string length
T-DQ	Transparent data exchange with acknowledgment
T-DO	Transparent data exchange without acknowledgment
A-CF	Configure ANZW [local]

The meaning of the individual services and how they are modeled on the SIMATIC PLC is described in the sections of this manual dealing with the specific services.

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgment for the service (i.e. the maximum dwell time of the job in the CP 1430 TF). This is specified in multiples of 0.1 sec. If the job is completed within the specified time, the parameter is irrelevant.

The value of the acknowledgment monitoring time at the application level (layer 7) should be adapted to the "data retransmission time" of the transport layer (layer 4) so that there are at least two repetitions before the connection is terminated on the transport layer (for example if data is lost due to a disturbance).

If the dwell time of the job elapses without an acknowledgment arriving, the following actions are initiated by the CP 1430 TF:

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

Error number in status word: D

2. If the acknowledgment arrives later it is ignored.

3. Application association terminated and static association automatically re-established.

Default:
Timeout = 10s
Data retransmission time = 0.6 s

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 HDB in the S5 user program. The call on the interface to the user program and the relationship to the configured connection block on the CP 1430 TF are illustrated in Fig. 3.1:

Support by the Request Editor

After the job buffers have been configured, the COM 1430 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 1430.

Depending on the service, the CP 1430 TF 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 (see also Fig. 3.1).

When the job is completed, the status is set by the communications processor to "job completed 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 CP 1430 TF whenever necessary. Segmentation is the transfer of the data field by field when the data length is greater than the field length specified in the SYNCHRON HDB for data transfer on the dual-port RAM.

Description of the SEND HDB Call Parameters

The following parameter descriptions are based on Fig. 3.1.

SSNR: 0 ... 255, the correct base interface number must be configured on the CP 1430 TF

ANR: 1 ... 199
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)
218: clock functions

ANZW: Status word as specified in the HDB description /7/, however, extended to three words:
1st and 2nd word
For a detailed description, refer to Volume 1, Appendix C.2 "Content of the status word"
3rd word
If the status "job completed with error" is entered in the 1st word, the 3rd word is valid and specifies the TF error in greater detail.

	15			0
1st word	free	Error management	Data management	Status management
2nd word	**Length word**			
3rd word	**TF error (ERRCLS/ERRCOD**)			

The TF error number is valid when
Status = Job completed with error

For more information about the status word, refer to Volume 1, Appendix C1/C2.

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 TF services

d) Error for special TF services (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 Appendix C 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 connection).

Instead of configuring the status word, this can be stipulated using a configuration job (refer to Chapter 9 “Supplementary Services" and Chapter 6 ”Request Editor”).

S5 address: Source address of the job buffer:
QTYP: Type of data block (DB, DX)
DB/DX-Nr.: Number of the data block or extended data block (1...255)
QANF: Start address of the job buffer
(0...(2043 - length))
QLAE: Length of the job buffer in words
(max. 128 words)

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

PAFE Parameter assignment error byte
The possible error codes and the causes of errors are described in Appendix C.3 'Parameter Assignment Error Byte PAFE', Volume 1.

3.3 General Server Interface

The Principle of Activating the Server Function

The TF server functions are handled on the CP 1430 largely without support of the CPU on the programmable logic controller. TF jobs are interpreted by the CP 1430 and executed with the PLC via the background communication, so that only a handling block call is required.

SEND-ALL
To transfer current process values to the CP 1430 TF.

RECEIVE-ALL
To receive transmitted data in the data areas of the user program.

This applies to:

➢ all TF variable services

➢ the general services for virtual manufacturing devices

Configuring Local Variables

Local variables (i.e. variables managed on the CP 1430 in the server role and which can be read or written) must be configured with the configuration tool COM 1430 TF:

➢ for application association-specific variables in the application associations dialog

➢ for VMD-specific variables using the VMD variables editor

or with the COM 1430 tool PG Load

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

This "general interface" is explained in the Chapter "Serial Data Transfer".

❑

NOTES

II COM 1430 TF Functions

4 Configuring and Testing the TF Interface

Topics in this Chapter

This chapter describes the functions, dialogs and parameters of the NCM COM 1430 TF tool for configuring and testing application associations and file server application associations.

You can refer to this chapter during configuration and testing.

During configuration, it is advisable to perform the steps in the order shown in the following overview.

Further Information

For further information, you should read the following:

- ➢ Chapter 3 'The TF Interface on the CP 1430 TF' in this volume about the functions and modes of the transfer services on the TF interface.
- ➢ Chapter 6 'Basic Configuration' in Volume 1 tells you about the general steps to be taken to create a database file.
- ➢ The online help function of NCM COM 1430 TF informs you about the meaning of input fields during configuration.

The margin entries shown on a gray background are references to the dialogs in the "COM 1430 TF Configuration Tool' supplement accompanying this manual.

4.1 Overview

The diagram below is an overview of the steps and functions available for configuring and operating a TF interface.

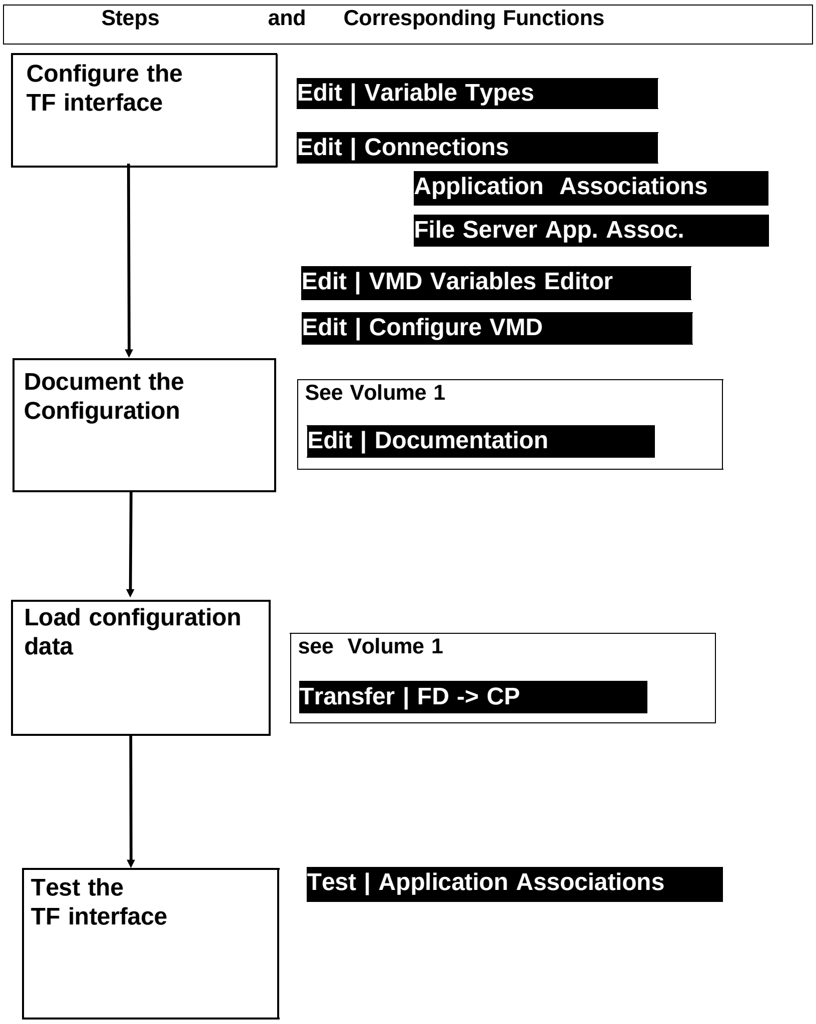


Fig. 4.1: Configuration Steps for TF Services

4.2 Defining TF Variable Types

The Meaning of the Defined Type

Automation systems often have variables with identical structures that describe the process variables and process statuses. Defining and entering the structure descriptions often represents a considerable part of the time and effort taken for configuring.

Specifying structures in the form of a defined type reduces the time and effort required for defining the variables managed on the CP.

The following example illustrates this advantage: Since the technological processes ’tank_1’ and ’tank_2’ are the same, the process variables ’status _tank’ have the identical structure ’tank_status’, that needs to be defined only once and assigned to the variables.

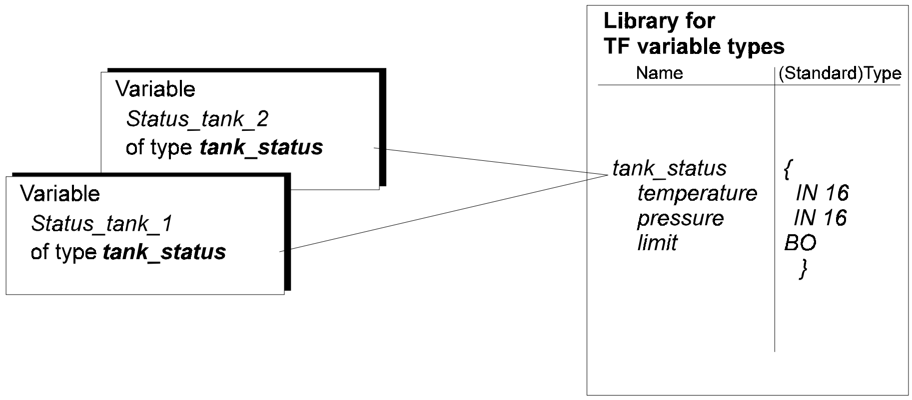


Fig. 4.2: Relationship Between TF Variables and TF Variable Types

Defining TF Variable Types with the TF Variables Editor
You can create a library in the CP database file containing the TF-variable types you require for your application.

NCM COM 1430 TF provides you with the TF variables editor with which you can define variable types. The library created by this procedure is stored in the CP block OB14.

Defining Types for Several Devices
The common type definition can, for example, be used both on the PLC at the process level and on a device at the management level.

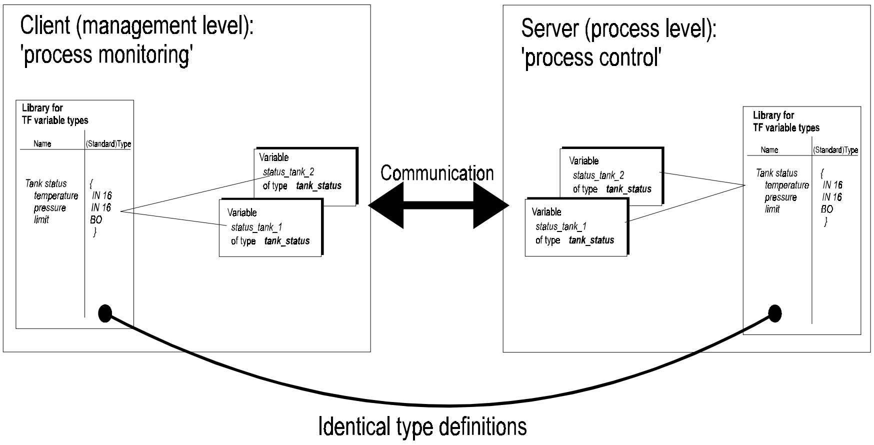


Fig. 4.3: Sharing the Type Library in Several Devices

You can copy the database block with the type library (OB14) to a different CP database file using the **File | Copy** function. This means you can define types once and use them for the entire system.

Limits
The library for TF variable types has capacity for a maximum of 100 type definitions (0..99) or 800 definition lines. If the variable types contain complex structures, the capacity of the CP block OB14 can be exhausted with less than 99 type definitions.

4.2.1 Edit | Variable Types | Editor

M 2-3-1

Meaning

You define user-defined variable types and save them in a type library (database block OB 14) of the selected database file or in the CP database.

When you enter variables, the ’Select’ and ’Help’ functions display not only the standard variable types but also variable types defined here, along with the variable type name and variable type number (see below).

Variable type name	Type	Offset	Variable type number
Tank status	*{*		
temperature	*IN 16*		0
pressure	*IN 16*		1
limit	*BO*		2
	}		
Temp setpoint	*IN 16*		

Fig. 4.4: Example of User-Defined Variable Types

Dialog/Input Fields

Variable type name: Maximum 32 character long name for a variable type.

Type: You specify the structure of a variable type. In the simplest case, you assign a simple standard type such as BO (Boolean) or IN 16 (integer 16 bit) to the variable type. The type definition of a variable type is analogous to the type definition of a variable (see Chapter 4.3). Example: definition of a structured variable type (tank status) and a simple standard type (temp setpoint).

The following table shows the possible types.

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 the COM
TD	Time and date	6 bytes generated by the COM
{	Start of structure	Number of components (calculated by the COM)
}	End of structure	
AR	Field	Number of elements in an array

Table 4.1: Variable Types in the TF Variables Editor

☞ **Note: it is not possible to use a second user-defined type within a user-defined type.**

Offset: The offset field is only an output field. With structured type definitions, it indicates the relative address of the data element within the structure. The value corresponds to the ’S5 address’ output field of the variable definitions. Since the variable types cannot be specifically assigned, no block addresses but only relative word addresses can be specified.

Variable type number: The automatically generated number is displayed here. You use this number as a reference when entering TF variables.

Note:
If you have entered a new type definition, the variable type number is only generated after pressing the F7 key (OK).

Function Keys (with an additional or context-specific meaning)

F3 FIND	With the "FIND" key, you can search for a user-defined variable type using the variable type.
F5 DELETE	With the "DELETE" softkey, you can delete the currently selected variable type. The variable type is deleted from the displayed list immediately. All other entries including generated variable type numbers, remain unchanged. Deleted variable types are simply marked internally as deleted and continue to take up space in the type library! The storage space is only deleted physically with the **Edit | Variable Types | Compress** function. For the effects of the variable definitions see dialogs M 2-4-4.2, M 2-4-4.3 and M 2-5.1): Variables that used the deleted variable type are now displayed in the Type column as ’undefined’.
F6 INSERT	This key inserts an empty line at the current cursor position. You can enter a new variable type in this line. To be able to insert structured variables, you must insert the required number of empty lines.

4.2.2 Edit | Variable Types | Compress

Meaning
The type library (database block OB 14) has a restricted capacity. Using the **Edit | Variable Types | Compress** function you can remove variable types marked as deleted from the type library and make more space for new type definitions.

All variables with the undefined status due to the type being deleted previously in the type library are removed from the database file or CP database.

After compressing, all the other entries in the type library, including the generated variable type numbers, are unchanged.

Procedure
To prevent you accidentally deleting type definitions, you should follow the steps outlined below:

✓ Delete the type definitions you no longer require in the TF Variable Types Editor dialog.

✓ Check the variable definitions in dialogs M 2-4-4.2, M 2-4-4.3 and M 2-5.1. If you want to retain variables marked as undefined (UN) in the Type column, assign a new variable type to them or delete the variables if they are no longer required. The variables you left as undefined are then deleted during the subsequent compression.

✓ Then select the **Edit | Variable Types | Compress** function. If variable definitions with undefined variable type assignments still exist, you will be prompted to confirm that you want to compress.

4.2.3 Multiple Use of the Type Library

Meaning

You can transfer the database block OB14 to a different database file or CP database with the **File | Copy** function. This allows you to make valid type definitions throughout the entire system although you only need to enter the definitions once.

This is an option with which you can be sure that data structures are valid and consistent throughout the entire system. You can, of course, make separate type definitions in every single database.

Procedure

✓ Select the **File | Copy** function.

✓ As the source file, specify the file in which you previously entered the globally valid type definitions. Select the CP database file containing no definitions or whose definitions you want to overwrite as the destination file.

✓ Select the SINGLE function with the F1 key.

✓ Confirm the data transfer only for the CP block OB 14 with YES.

☞ **Note: if you overwrite an existing type library with the copy function, the following situations can arise:**

1. Variable definitions with undefined type assignments: assigned variable type numbers no longer exist in the new type library.

2. Variable definitions have other structures:
The variable type was changed.

4.3 Editing Connection Blocks

4.3.1 Edit | ...| Overview

M2-4-1 You can obtain information about existing connection blocks with the **Edit | Connections | Overview** function. The following types are displayed:

➢ Transport connections

➢ Datagram services

➢ Application associations

➢ File server application associations

To display or modify the configuration data of a connection block displayed in the list:

✓ Select the entry with the cursor.

✓ Press the F7 key (OK)

To edit a connection block, you can also select the input dialogs for the types directly using the NCM COM 1430 TF menu.

4.3.2 Edit | ... | Application Associations

M2-4-2.1 **Aims**

With the **Edit | Connections | Application Associations** function or by selecting a connection block as described in the previous section, you display the **'Application Associations'** dialog. Here, you assign parameters for the transport connections.

If you have already created connection blocks for the database file, when you select the **Edit | Connections | Application Associations** function, the first block of the database file is displayed.

The configuration of variables to define the services with which the variables are transferred is part of the connection configuration.

Essential Basic Configuration

A connection block can only be defined when the database file has been completely initialized. Refer to the procedure for basic configuration described in Volume 1, Chapter 6.

Number of Application Associations

A maximum of 16/100 SINEC application associations each with a TF-PDU size of 1024 bytes can be configured. This number is reduced if you select a higher value for the TF-PDU size (see Volume 1, Section 4.6).

Note:
Some of the entries in this dialog are identical to the parameters for calling the handling blocks in the STEP 5 user programs.

Configuring Variables

When configuring variables, a distinction is made between the local definition for server functions (the PLC provides data = variables) and the remote definition for client functions: (the PLC accesses data = variables):

Local application association-specific variables:	Variables are managed in the PLC whose application associations are being configured. Other communications partners (PLCs) write or read these variables (the PLC is server).
Remote variables :	The PLC whose application associations are being configured writes, reads or reports variables (the PLC is client).

Input Fields for Assigning Parameters to the Interface to the PLC

Application association name:	Name of the defined application association (only for information) Possible values: 32 ASCII characters
SSNR offset:	The parameter specifies the interface number offset 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 (SSNR = base SSNR + SSNR offset) must also be specified in the handling block for connection identification. Possible values: 0..3
ANR:	In conjunction with the local interface number, the job number specifies the connection block uniquely. In the control program, the job number and the interface number must be transferred to the handling block to identify the connection and the job. Possible values: 1...199 Only odd job numbers can be assigned for application associations!

Input Fields for Assigning Parameters to the Interface to the PLC (continued)

Status word:

Status word for the defined job (see: specification of handling blocks)
Format:
FW 0...250 or
DB’ < 0...255> ’< 0...2040>
" "
DB number word number

With client jobs, this status word is transferred by the CP to the HDBs in the SEND/RECEIVE-ALL jobs.

Note:
The connection 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:

	15			0
1st word	free	Error management	Data management	Status management
2nd word	**Length word**			
3rd word	**TF error (ERRCLS/ERRCOD)**			

If the address of the ANZW for the connection is different from that of the ANZW for the HDB, the 2nd and 3rd words are used exclusively for the SEND/RECEIVE ALL block.

How to use the status word and the meaning of the individual bits can be found in the descriptions of the handling blocks (see Chapter 3 and /7/).

Input Fields for Assigning Parameters for Job Processing

TF-PDU size

With this parameter, you specify the maximum size of the buffer used for transfer on the application association.

Possible values 128...65535

Select the TF-PDU size
- if possible < 1025, if you want to operate the maximum number of application associations
- however not less than necessary for transferring the service (header + data information).

Possible values: 128...65535

For more detailed information, refer to Volume 1, Chapter 4 'Technical Description and Guide to Installing the CP 1430 TF'.

Establishment 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 9 'Application Associations'.

The establishment of a connection is handled implicitly by the CPs, i.e. there is no explicit job from the PLC. Connections are established as follows:

active:
The CP 1430 initiates connection establishment during start-up. The resources (buffers) are therefore always in use.

Input Fields for Assigning Parameters for Job Processing (continued)

passive:
The CP 1430 expects a connection request from the partner.

dynamic:
The CP establishes the connection as soon as a job is pending for the configured connection. Resources are only used as they are needed.

Multiplex address: Multiplex address of the layer 7 connection
Possible values: 0,...,255

Here, you enter the multiplex address “negotiated” with the partner application. The multiplex address is an additional address to identify the peer application.
Since each application association is modeled on its own transport connection on the CP 1430 TF, the multiplex address has no meaning in a homogeneous environment (only CP 1430 TF modules). It is therefore advisable to leave the multiplex address setting at 0 in this situation.

☞ **The same multiplex address must be configured on the local and remote station.**

☞ **Note the following when converting a COM 143 database to COM 1430:**
If the multiplex function was used in the COM 143 database, the application associations multiplexed on one TSAP will be modeled on different transport connections when the database is converted. After the conversion, the same TSAPs are assigned to these transport connections!
Modify the TSAPs so that the assignments are unique!

Transport Addresses - Local Parameters

In these input fields, you address the communications path in the local PLC.

Entries for the TSAP:

Default:
When you call the dialog, the TSAP from the previous dialog is entered.

Note:
Displaying the TSAP in both hex and ASCII has the following advantages
- it is easier to enter the TSAPs as an ASCII string
- the TSAPs are not restricted to only ASCII characters.

TSAP (ASC): The TSAP for the local PLC can be entered here in ASCII characters.

Possible values: max. 8 ASCII characters

TSAP (HEX): The individual bytes of the TSAP ID can be entered here in hexadecimal notation in groups of two (values from 00 to FF).

Possible values: max. 8 hex characters

TSAP length: This specifies the number of TSAP characters and has the default "8". With connections to non-SIMATIC devices, it may be necessary to specify a shorter length.

Transport Addresses - Remote Parameters

In these input fields, you address the communications partner.

MAC address (HEX): This is the physical module address of the remote PLC.
If MAC address 00 00 00 00 00 00_H is entered for the RECEIVE job, the address counts as being unspecified. When the connection is established, any partner is accepted regardless of its address.

Entries for the TSAP:

Default:
When you call the dialog, the TSAP from the previous dialog is entered.

Note:
Displaying the TSAP in both hex and ASCII has the following advantages
- it is easier to enter the TSAPs as an ASCII string
- the TSAPs are not restricted to only ASCII characters.

TSAP (ASC): The TSAP for the remote PLC can be entered here in ASCII characters.
Possible values: max. 8 ASCII characters

TSAP (HEX): The individual bytes of the TSAP ID can be entered here in hexadecimal notation in groups of two (values from 00 to FF).
Possible values: max. 8 hex characters

TSAP length: This specifies the number of TSAP characters and has the default "8". With connections to other types of devices, it may be necessary to specify a shorter length.

If you specify TSAP length = 0, the TSAP counts as being unspecified.

An unspecified connection end point must be configured as passive.
The MAC address must also be unspecified (address 00 00 00 00 00 00).

M2-4-2.1 **Function Keys** (with an additional or context-specific meaning)

F1/F2 +1 /-1	With the F1 key "+1"; you display the next connection block, with F2 "-1", the previous.
F3 INPUT	With the F3 key "INPUT", you can enter a new block. An empty dialog is displayed.
F4 LOC VAR	Submenus for variable definitions used for server services (Read, Write) (-> M 2-4-4.2).
F5 REM VAR	Submenus for remote variable definitions used for client calls (-> M 2-4-4.3).
F6 DELETE	With the F6 key "DELETE", you delete a connection block. This command must be confirmed again after the PG prompt.
F7 OK	The F7 key "OK", saves all the parameter settings in the database file or (in the online mode) directly on the CP 1430. In the offline mode, it is advisable to save the parameter settings with this from time to time while editing.
^F7 TR PARA	You select the follow-on dialog **'Transport Parameters'** M 2-4-2.3.

M2-4-4.2 Configuring Local Variables

Meaning

To define or configure local application association-specific variables, you press the F4 key (LOC VAR) in dialog M 2-4-4.1 to branch to the “Local AA-specific Variables” dialog.

Here, you define variables and their structures belonging to values in local data blocks. These variables can only be accessed via the application association currently being configured.

Dialogs/Input Fields

Name: Maximum 32 character long identifier for a variable.

Type: This parameter specifies the type of variable. You can specify user-specific variable types created with the TF variable type editor or standard variable types.

a) User-defined variable types

You specify the number of the variable type you defined for this variable. Use the selection key F8! Apart from the standard types, all user-defined variable types with their numbers and type identifiers are displayed.

b) Standard variable types

The first input field (2 characters long) identifies the actual variable type and the 2nd input field (4 characters long) the variable length.

Example: IN 16

M2-4-4.2 Configuring Local Variables (continued)

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 4.2: Standard Variable Types

ACC: Access type:
R only reading possible, no specification means access not restricted.

M2-4-4.2 Configuring Local Variables (continued)

S5 ADDRESS: Address of the memory area for modeling the value of the variable on the PLC.
FORMAT: <ORG_identifier> <ERW_identifier> <start addr.>
ORG_identifier = DB or DX for data block
ERW_identifier = 0..255 data block number

start addr.= 0..2042 data word number, at which value of the variable starts.

ANZW: Variable status word: Configured status word for server functions with these variables .

When configuring, you can not only configure the name and type of the variables and the S5 address at which the variable is stored by the PLC program, but also specify an address for a status word. Here, the PLC program can check information about access to the variable.

It is, for example, possible for the PLC program to recognize whether the value of a variable has been updated by another station or whether there has been no access.

Using this status word, the PLC program can also (temporarily) block access to the variable for another station.

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 and /7/).

M2-4-4.2 Configuring Local Variables (continued)

Format:
<TYPE> <DBNR> <DWNR> or <FWNR>
(Possible values: where TYPE = FW, DB, DX
DBNR = 0..255, if TYPE = DB or DX (data block number)
FWNR = 0..250, if TYPE = FW (flag word number)
DWNR = 0..255, if TYPE = DB, DX (data word number)
DWNR = empty, if TYPE = FW)

The status word as presented here is identical in its structure to the status words of the handling blocks.

15 0

free	Error management	Data management	Status management
Length word			

SSNR offset: This specifies the CPU in which the variable is physically located. In the single processor mode the SSNR offset specified for the connection must be entered, 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 connection-specific variables.

M2-4-4.2 **Function Keys** (with an additional or context-specific meaning)

F3 INSERT	This key inserts an empty line at the current cursor position. You can enter a new variable type in this line. To be able to insert structured variables, you must insert the required number of empty lines.
F4 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.
F5 FIND	With the "FIND" key, you can search for a variable using the variable name.
F6 INFO	With the "INFO" key, you obtain further information about the type of the variable currently marked by the cursor (e.g. the variable type name of self-defined types).
F7 OK	The entered data is converted to internal structures. You exit the dialog, but only after the connection dialog has been saved.

M2-4-4.2 Configuring Local Variables - Example

The “Local AA-specific Variables” dialog in the supplement, illustrates the following example of a variable definition.

The variable Status_heating is a structured variable consisting of three components

- ➢ Water_temperature: standard type IN 16
- ➢ Setpoint: standard type IN 16
- ➢ Indicators: structure, consisting of three components of the type BO

The setting ACC = R means that the structure Status_heating can only be read.

The structure is mapped on data block 7 starting at data word 8.

Flag word 10 is used as the status word for the variable.

M 2-4-4.2 Configuring Local Variables - Entering a Structure

Entry and Calculation of the S5 Address
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 1430 and displayed the next time you page through the variables definitions.

The length of structures is also calculated based on the components entered and displayed next time you call the variable definitions.

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.

Indented Display
Structure definitions are indented automatically by NCM COM 1430 TF. You should therefore edit starting on the left without indenting.

The text is displayed indented the next time you enter your data with the F7 key 'OK' and then select the dialog again.

☞ **The nesting level for variables is restricted to 2.**

Errors
If an error is detected, an error message is generated and displayed.

M2-4-4.3 Configuring Remote Variables

Meaning

Analogous to the application association-specific local variables accessed by the PLC as server, remote variables can also be configured in COM 1430 TF. These variables can be read or written in the server via the application association with the PLC in the client role.

The remote variables are configured on the device which

➢ triggers the variable services read and write variable

➢ receives the reported variables

These variable definitions are only necessary and only used by the CP 1430 TF 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).

Dialogs/Input Fields

Scope:	Scope of the remote variable AA = application association VM = virtual machine DO = domain
Name:	The meaning is analogous to the definition of local variables. For a variable with the scope DO = domain, the domain name must also be entered in the first line.
Type:	The meaning is analogous to definition of local variables.
S5 address:	The meaning is analogous to definition of local variables.
ANZW.:	The meaning is 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.

4.3.3 Edit | Connections | File Server App. Assoc.

M2-4-3

Meaning

In the “Application Associations to File Server” dialog you can define so-called "third-party associations". These are connections that are established implicitly between the CP and a file server after a load operation is triggered by a host computer.

Dialog Structure

M 2-4-4.1

The dialog is identical to dialog M2-4-4.1 ’Application Associations’, however, the input fields for SSNR offset, status word, establishment type and multiplex address 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 1430 TF.

4.3.4 Edit | VMD Variables Editor

M 2-5.1 **Meaning**

The VMD Variables Editor option provides functions with which you can define VMD-specific variables.

VMD-specific variables can be accessed via any application association.

VMD-specific variables are stored in a special organization block within the configured module.

M 2-4-4.2 The structure of the dialog 'VMD-Specific Variables' is identical to the dialog 'Local AA-specific Variables (see Page 4-20).

☞ **The nesting level for variables is restricted to 2!**

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. You obtain the dialog for groups with the F4 key.

Function Keys (with an additional or context-specific meaning):

Shift+F4 GROUPS	This function allows variables to be grouped together. The dialog M 2-5.2 'Group Definitions' is displayed. (see next section)
Shift+F5 DEL BLK	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 dialog.

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 variable names).

M 2-5.2 Edit | VMD Variables Editor - Defining Groups

You display the 'Group Definitions' dialog by pressing function key ^F4 in the VMD-Specific Variables dialog, dialog M2-5.1.

By defining a group, you can transfer variables collected in groups using the information report service.

Input Fields:

Group name:	8 character long ASCII string, must be unique per VMD.
Subfields (left and right column)	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.

☞ **Note that the variable names must be entered line by line alternating columns and not one complete column and then the other.**

Function Keys (with an additional or context-specific meaning)

F3 INPUT	A dialog with empty fields is displayed in which a new group can be defined.
F4 DEL GRP	The currently displayed group is deleted.
F5 DEL VAR	The variable entry at the current cursor position is deleted.
F7 OK	Any changes you have made are entered and the dialog is exited. The data are only converted to the internal representation when you exit the VMD dialog, since you can still change the variables definitions.

4.3.5 Edit | Configure VMD

M 2-6

The ’Configure VMD’ 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 (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.

M 2-6 Edit | Configure VMD (continued)

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 (with an additional or context-specific meaning):

F1 INCL/DEL	Include or remove a CPU
F3 <----------	Select the previous CPU
F4 ---------->	Select the next CPU
F6 PRESETS	Create presets

4.4 Testing the TF Interface

The following dialogs are available for testing the TF interface:

Total Status Application Associations

M 5-2.1 This display provides you with an overview of the current status of all the configured connection blocks of the TF layer. These include application associations and file server application associations.

Single Status Application Associations

M 5-2.2 You can run diagnostics on an application association you selected in the overview display Total Status Application Associations.
(To select an application association, move the inverse bar with the cursor keys.)

You obtain the following extra information:

➢ Display of a connection error
➢ Hex codes are decoded in plain text
➢ Complete address information

Single Trace Application Association

M 5-2.3 You can run diagnostics on an application association you selected in the overview display Total Status Application Associations. Each status change leads to a new entry in the trace list.

You obtain the following extra information compared with Total Status:

➢ Display of a connection error
➢ Registration of the job history

You can check that the data and job transfer on the CPU-CP interface and to SINEC H1 is functioning correctly.

☞ **The ONLINE test functions can be run both on the AS511 and the SINEC H1 interface. The test functions affect the processing of the communications services. Due to its low transmission rate of 9.6 Kbps, the AS511 interface delays the communication on the TF interface.**

4.4.1 Test | Application Associations

M 5-2.1

Meaning and Functions

With the **Test | Application Associations** function, you select the **’Total Status Application Associations’** dialog.

The PG fetches information from the CP 1430 about all configured application associations and file server application associations and displays them with their current status in a table. A maximum of 13 single statuses can be displayed at one time.

Updating the Display

To update the display, follow the steps below:

✓ Select a job you want to have updated the cursor.

✓ Press the F4 key SELECT. The job is then marked in the Sel column with ’x’.

✓ Repeat this for all other jobs you want to have updated.

✓ Activate the update function with the F1 key. Only the selected jobs are then displayed. All jobs whose jobs status changes while the update function is active are marked with ’x’ in the Chge column.

Output Fields

Operating mode:	Information about the CP status (RUN, STOP, IDLE)
Local MAC address:	Physical module address of the local PLC.
Sel.:	This column indicates the entries selected for continuous updating. You can toggle the updating (ON/OFF) with the F1 key.
POS :	Consecutive number of the job in the table.
SSNR offset:	Interface number offset of the job.
ANR :	Job number

AA-ID: Displays an application association identifier that is transferred by the underlying transport layer when an attempt is made to open a transport connection.

If the displayed value changes continuously, you can see that no connection is established.

AA status: Displays the status of an application association. (see Table 4.3)

TF job: Displays the type of job specified in the job buffer.

Opcode as shown in Table 4.4

J status Displays the status of the job processing.

see Table 4.5

J error TF errors that occurred during job processing.

See Appendix C 'TF Error Numbers used by the CP 1430"

Chge: This column indicates the jobs whose status has changed since the last status query. The status query begins at the point in time when the updating of the job display was activated.

The following application associations are predefined and always displayed:

ANR 205: Job number for local jobs sent to the CP 1430 TF by the local PLC program.

ANR 206: Job number for the predefined application associations (server side - see Section. 9.1.4 "Special Application Associations"), for example, used by PG Load for load functions.

Function Keys (with an additional or context-specific meaning**)**:

F1 UPD ON	Toggles the updating of jobs selected with the SELECT key on and off.
F2 SING STAT	The follow-on dialog **'Single Status Application Association'** M 5-2.2 is selected.
F3 TRACE	The follow-on dialog **'Single Trace Application Association'** M 5-2.3 is selected.
F4 SELECT	You select a job marked by the cursor.
F5 DESELECT	You deselect a ‘selected’ job marked by the cursor.
F7 RETURN	You terminate the test session.

4.4.2 Follow-on Dialog ’Single Status Application Association’

M 5-2.2

Meaning and Functions

The "F2" key in the "Total Status Application Associations" dialog displays the follow-on dialog "Single Status Application Association".

The PG fetches information from the CP 1430 about a configured application association and displays detailed information about it along with its current status.

In contrast to the overview, you can identify the application association uniquely using the address information. Status information is also displayed in plain text.

Updating the Display

Just as in the Total Status you can continuously update the status display.

✓ Press the F1 key to toggle between a static and an updated display.

Output Fields:

AA type.:	The job is one of the following types: - App. Ass.: application association - File Server: dynamic load connection to the file server - Predefined: predefined static connection, for example for PG Load services.
Operating mode:	Information about the CP status (RUN, STOP, IDLE)
Local MAC address:	Physical module address of the local PLC.
SSNR offset:	Interface number offset of the job.
ANR :	Job number.
J type :	Job type: Client (with odd ANR) Server (with even ANR)

TF job:	Displays the type of job specified in the job buffer. Opcode as shown in Table 4.4
J status	Displays the status of the job processing. see Table 4.5
J error	TF errors that occurred during job processing. See Appendix C ’TF Error Numbers used by the CP 1430”.
Appl. ass. name:	Displays the application association name if configured.
AA status:	Displays the status of the application association. see Table 4.3
AA ID:	Displays an application association identifier that is transferred by the underlying transport layer when an attempt is made to open a transport connection. If the displayed value changes continuously, you can see that no connection is established.
Est type	Displays the establishment type selected for the application association: - active - passive
TF PDU size:	The TF PDU size selected for the application association.
Job watchdog time:	Displays the watchdog time set for the current job. This time is known as the timeout on the job buffer interface (see also Chapter 6 Request Editor). When this time expires without the communication partner responding, the job is terminated and a message is generated.

Local parameters: Configured local TSAP address of this application association.

Remote parameters: Configured remote MAC address and TSAP address of this application association.

Function Keys (additional or with context-specific meaning):

F1 UPD ON	Toggles the updating of jobs selected with the SELECT key on and off.
F4 MANUAL	With this key, you can stop the automatic processing of a TF job on the CP. Pressing the key again moves the job on one step. To reactivate the automatic processing, press the ESC key.
F7 RETURN	You terminate the test session.

4.4.3 Follow-on Dialog ’Single Trace Application Association’

M 5-2.3

Meaning and Functions

The "F3" key in the "Total Status Application Associations" dialog displays the follow-on dialog "Single Trace Application Associations".

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. This function allows you to follow the history of the selected application association.

Updating the Display

Just as with Total Status, you can also continuously update the status display.

✓ Press the F1 key to continue or stop the recording of status information.

☞ **Disabling the recording or pressing the UPD OFF key only produces a reaction after 20 seconds if no job is active.**

Trace Buffer Overflow

If the CP 1430 is not able to enter all the status changes in the trace buffer (overflow), this is indicated on the PG.

Output Fields:

Operating mode:	Information about the CP status (RUN, STOP, IDLE)
Local MAC address:	This is the physical module address of the local PLC.
SSNR offset:	Interface number offset of the job.
ANR:	Job number
Pos:	Consecutive number

AA ID:	Displays an application association identifier that is transferred by the underlying transport layer when an attempt is made to open a transport connection. If the displayed value changes continuously, you can see that no connection is established.
AA status:	Displays the status of an application association. (see Table 4.3)
TF job:	Displays the type of job specified in the job buffer. see Table 4.4
J status	Displays the status of the job processing. see Table 4.5
J error	TF errors that occurred during job processing. See Appendix C 'TF Error Numbers used by the CP 1430".

Function Keys (with an additional or context-specific meaning**)**:

F1 UPD ON	Toggles the updating of jobs selected with the SELECT key on and off.
F7 RETURN	You terminate the test session.

4.4.4 PI/Domain Status

M 5-3

With the **Test | PI and Domains** function, you obtain a display of the current statuses in the program invocation (PLC) and the domains in the "PI/Domain Status" dialog.

➢ PI status:
Using the PI services, a communications partner, for example a host computer, can trigger start/stop functions on the PLC. The effects of such control functions can be seen using this test function.

➢ Domain status:
Using the domain services (see also PG Load), programs and data are loaded on the PLC and managed as domains. The status of these domains is displayed here.

Output Fields:

Program invocation	Predefined (system-PI = PLC_START_STOP) or user-defined name of the current program invocation.
PI status	Displays the status of the PI named as the program invocation. You can update the status continuously with the F1 key.
No.	Consecutive number of the displayed domain information.
Domain name	Predefined (static domain = SIMATIC_S5) or user-defined name of the domain displayed in the line.
Domain status	Displays the status of the domain. You can update the status continuously with the F1 key.

Function Keys (with an additional or context-specific meaning**)**:

F1 UPD ON	You toggle between a static display and a display updated when the status changes.

4.4.5 Status Displays of the Test Functions

AA status

Coded display	Meaning
0000_H	Initialization phase active
0100_H	Transport connection establishment active
0101_H	Transport connection establishment active (again)
0200_H	Awaiting AA establishment
0201_H	Awaiting AA establishment (again)
0300_H	Connection established
0500_H	Transport connection abort after timeout
0501_H	Transport connection abort after bus problem
0502_H	Transport connection abort after protocol error
0503_H	Transport connection abort from partner
$0F00_H$	Database defective or no memory
1000_H	Initialization phase active
1100_H	Transport connection establishment active
1101_H	Transport connection establishment active (again)
1200_H	Activating AA establishment
1201_H	Activating AA establishment (again)
1300_H	Connection established
1500_H	Transport connection abort after timeout
1501_H	Transport connection abort after bus problem
1502_H	Transport connection abort after protocol error
1503_H	Transport connection abort from partner
$1F00_H$	Database defective or no memory
2000_H	Initialization phase active
2100_H	Transport connection establishment active dyn
2101_H	Transport connection establishment active (again)
2102_H	Transport connection establishment active dyn
2200_H	Awaiting AA establishment
2201_H	Awaiting AA establishment (again)

Coded display	Meaning
2202_H	Activating AA establishment dyn
2300_H	Connection established
2400_H	Deactivating AA establishment dyn
2500_H	Transport connection abort after timeout
2501_H	Transport connection abort after bus problem
2502_H	Transport connection abort after protocol error
2503_H	Transport connection abort from partner
$2F00_H$	Database defective or no memory
$F000_H$	Unknown initialization
$FF00_H$	Type of connection not defined

Table 4.3 Status Displays AA status

TF jobs

Opcode	Job
A-IA	Initialize application association
A-AA	Abort application association
A-CF	Configure ANZW
ABOR	Abort
B-RQ	Read byte string
B-WQ	Write byte string with acknowledgment
B-WO	Write byte string without acknowledgment
B-WI	Query byte string length
CONN	Connect
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	Terminate upload sequence
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 program invocation
P-DE	Delete program invocation
P-GE	Get program invocation attributes
D-CA	Get capability list
IDLE	Job not active
M-ST	Status (of virtual device)
M-GN	Get name list
M-ID	Identify (virtual device)
M-SU	Unsolicited status
NOID	Unknown TF function
P-ST	Start PI
P-SP	Stop PI
P-RE	Resume PI
P-RS	Reset PI
P-AB	Kill PI
P-HL	Local program stop
T-DQ	Transparent data exchange with acknowledgment
T-DO	Transparent data exchange without acknowledgment
V-RE	Read variable
V-WR	Write variable
V-IN	Information report

Table 4.4: Opcodes for TF Jobs

J status

Coded display	Meaning
0000_H	No job exists
0001_H	No job processing at present time
0008_H	Job to remote station
0009_H	Reply to job from remote station
0010_H	Awaiting job from remote station
0011_H	Job received from remote station
0012_H	Acknowledgment received
0040_H	Send direct received from PLC
0048_H	Awaiting send-all trigger
0049_H	Send-all active
$004A_H$	Data from send-all received
0050_H	Awaiting receive direct trigger
0051_H	Receive-direct active
0052_H	Ack. received from receive-direct
0058_H	Awaiting receive-all trigger
0059_H	Receive-all active
$005A_H$	Ack. received from receive-all
0060_H	Initiate error output
0061_H	Error output active
0062_H	Ack. of error output received

Table 4.5 J Status Displays

❑

NOTES

5 PG Load

Topics in this Chapter

The PG Load 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.

PG Load 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 PG Load 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 PLCs using the TF program invocation services.

The tool is integrated in COM 1430 TF. You can activate it with the menu item "Utilities".

5.1 Overview

5.1.1 Adapting Programmable Logic Controllers to the Process with PG Load

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 PG Load 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 PG Load user interface, you simply specify the connections for data exchange and select objects for the transfer (domains) to the controller (PIs).

The PG Load host functions allow you to intervene in the running of the PLC using the PG Load user interface and so you implicitly use the TF services.

Structuring the PLC with PG Load for TF Services

PG Load is responsible for forming domains from blocks and generating loadable domain files. PG Load is therefore needed for PLC structuring in the sense of the TF services.

5.1.2 Range of Functions

TF Services under PG Load

The PG Load program supports the following domain and program invocation services:

➢ Domain services:
 - uploading a domain from the PG into a file server (triggered by the PG)
 - downloading a domain from the file server to the PG (triggered by the PG)
 - triggering the load sequence in the PLC from a file server or PG (triggered by the PG)
 - triggering the uploading from the PLC into a file server 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

PG Load is designed for the domain and PI model on which a SIMATIC S5 programmable logic controller is modeled.

The following rules apply to domains and PI services

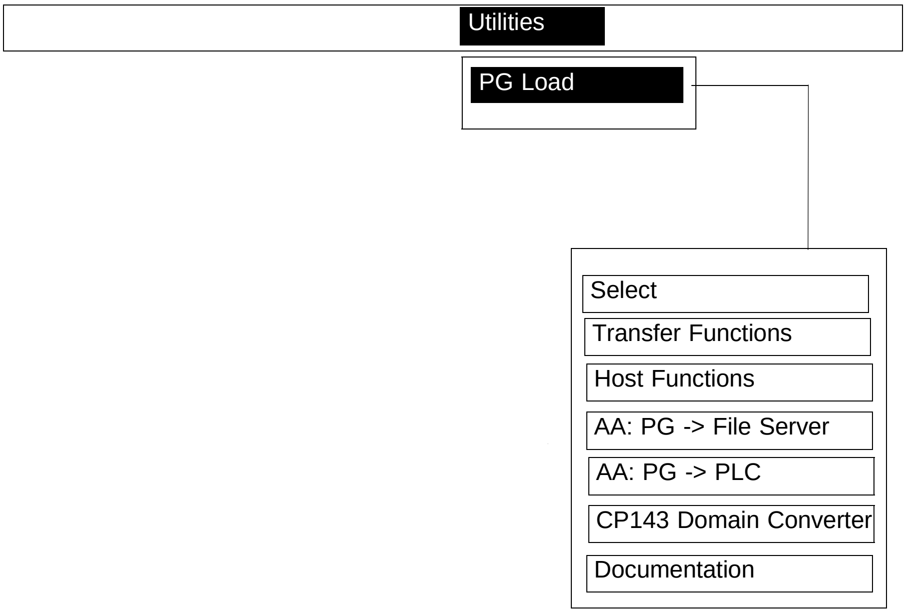
➢ A PLC contains up to 8 dynamic and 1 static domain.

➢ A domain consists of any combination of blocks. It is useful to separate blocks containing program (logic) and those containing parameters (data).

➢ 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.

5.2 Description of the Tool

The structure of PG Load is reflected in the menu:



Select
With the **Select** function, you select the files for configuring application associations.

Transfer Functions
Transfer functions are used to transfer load files between the PG and file server. Load files contain the program and data blocks put together to form domains.

Host Functions
With the host 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.

AA: PG -> File Server
You specify the application association between the PG and TF file server. You can transfer load files to the TF file server on these application associations.

AA: PG -> PLC
With PLC application associations you specify a connection between the PG and the specified PLC. You use the application associations to transfer host commands (PI services).

CP 143 Domain Converter
Domain files created with COM 143 can be converted to the COM 1430 format.

Documentation
You can display or print the application associations configured with PG Load.

5.3 Functional Description

5.3.1 System Configuration and Device Functions

This section describes the use and functions of PG Load.

To simplify the explanation, the following network of devices is assumed.

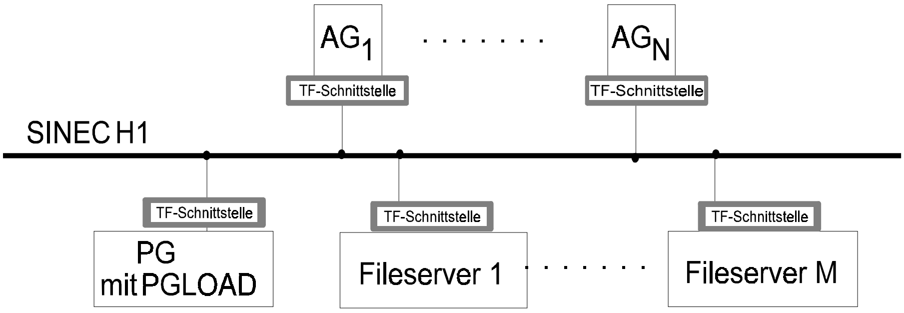


Fig. 5.1: Example of a Network when using PG Load

Programming Device PG
The PG is integrated in the automation system via SINEC H1. With PG Load 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 (PG Load).

TF File Server 1..M
TF file servers 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 file server. PG Load supports the situation in which the PG is used both for the PG Load functions and for data storage as a file server.

5.3.2 Configuring Application Associations and Selection Functions

Aims and Procedure
The logical partners, i.e. the PLCs and file servers can be defined using a connection editor. The connection definitions are written as connection blocks in a file for the PLC application associations and file server application associations.

Existing application associations can be selected in the current connection file and can be stipulated as defaults for transfer and host functions.

PLC Application Associations (Application Association PG->PLC)
PLC application associations are connections between the PG and a PLC. These connections are used to transfer jobs for file server access and PI service jobs.

Configuring a PLC application association simply involves specifying the MAC address of the partner a connection name used locally on the PG. The local and remote TSAPs are formed implicitly. (the local and remote TSAP of the predefined application association is: S5_STF -> see Section 9.1.4). Jobs on this application association can be displayed with the test functions of COM 1430 TF.

The connection blocks for PLC application associations are stored in a file with the name: <xxxxxxCP.LOD> (xxxxxx is any 6-character long ASCII character string).

TF File Server Application Associations
TF file server application associations are connections between a PG and a file server. The PG and TF file server can exchange load files using transfer functions.

The connection blocks for PLC application associations are stored in a file with the name: <xxxxxxPG.LOD> (xxxxxx is any 6-character long ASCII character string).

5.3.3 Transfer Functions

Support of Data Storage
The transfer functions support the conversion and transfer of S5 program files to the TF file server. They also support retransfer and reconversion when the programs require modification with the programming tools (LAD, CSF and STL).

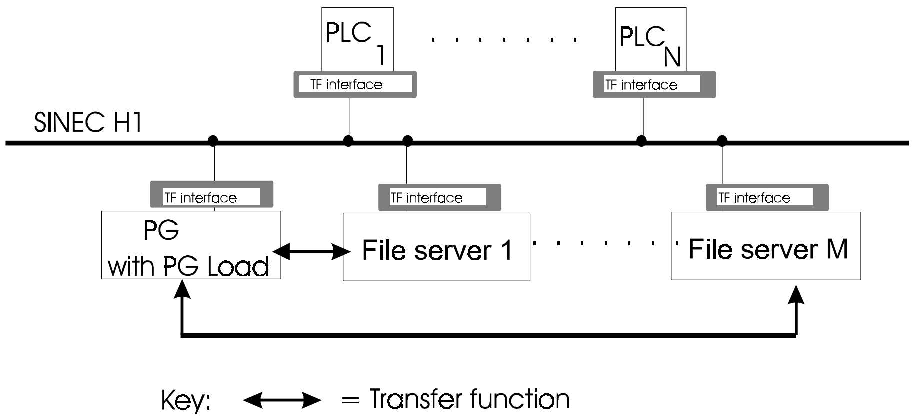


Fig. 5.2: Direction of the Transfer Functions in a Network

The Following Functions are Available

Transferring domains from the PG to a file server.	Domains on the PG are transferred to a file server 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 file server to the PG	Domains on the file server 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.

Note

The conversion functions can also be started separately. This means it is possible to store data on the PG (PG is file server) when transfer to a different file server is not required.

5.3.4 Host 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 dialog can be transferred from the file server to a CPU of the PLC (the file server can also be the local PG). Situation a: third-party configuration If the connection name of the server is not "PG", a PLC (destination station) is instructed to load the domains from the file server. Situation b: PG = file server If, however, the connection 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 file server. This means that the PG instructs the PLC to send the data to the server. The connection name is handled in the same way as in the load PLC function. The distinction between a third-party configuration and PG=file server 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.
Create/delete PI:	A program invocation is created from the domains on the CP or it is deleted.
Start Program:	The domains grouped to form a program invocation are started.

Stop Program: If a program invocation with the specified name is running, it will be stopped.

The following diagram illustrates the host functions:

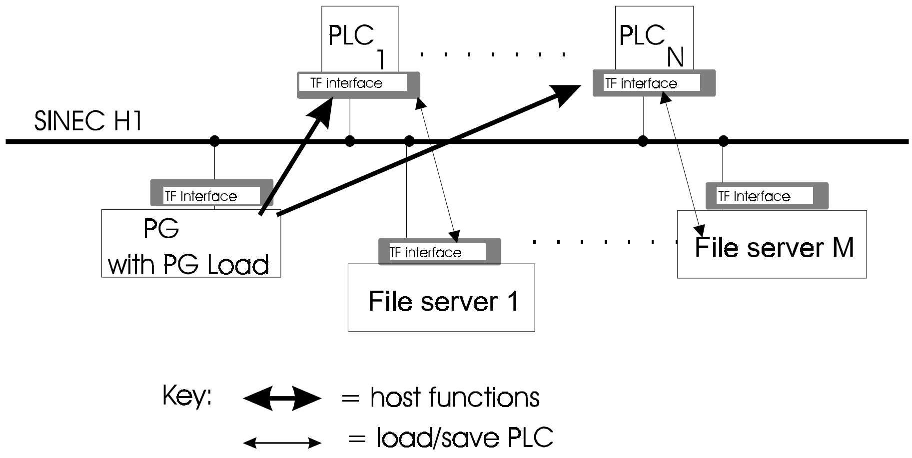


Fig. 5.3: Direction of the Host Functions in a Network

Note: The PG load program only operates via the integrated Ethernet interface, i.e. CP 141, CP 1413.

5.4 PG Load - Application

5.4.1 PG Load | Select

The connection files specify the destination devices that can be accessed with PG Load. The first task to perform with PG Load is therefore to specify these connection files.

After selecting **PG Load | Select** the following dialog appears.

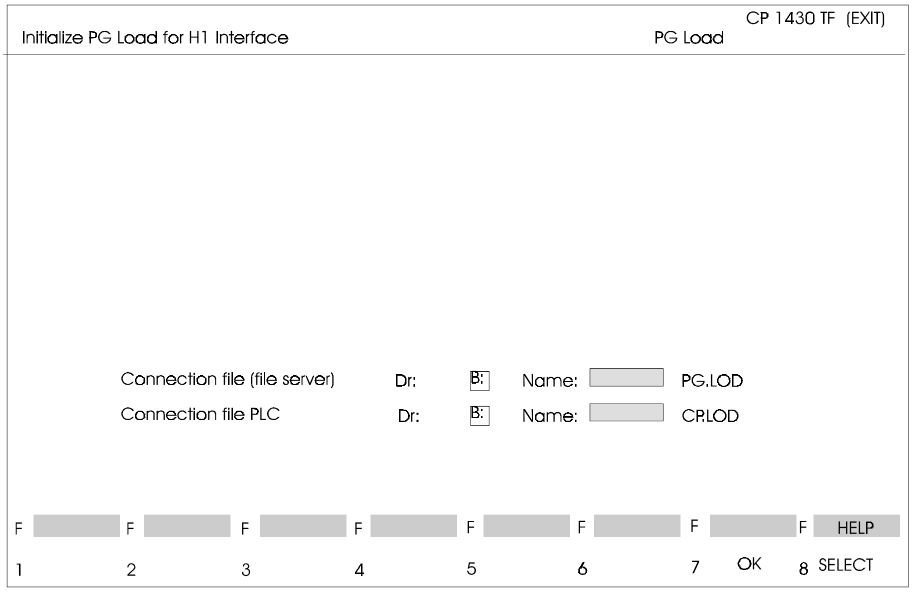


Fig. 5.4: Initialization Dialog

Input Fields:

Connection file (file server):	Depending on the functions you require, you can specify a connection file for TF 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.
Connection file (PLC)	Here, you specify the connection file for PLC application associations. Format : Dr: = A ...N Name : <XXXXXXCP.LOD> where <XXXXXX>: = can be freely selected. The suffix CP.LOD is also fixed and cannot be changed.

5.4.2 Configure AA to File Server / Select File Server

With **PG Load | AA: PG -> File Server** you can configure file server application associations.

File server application associations, in this case, are connections between a PG and a file server.

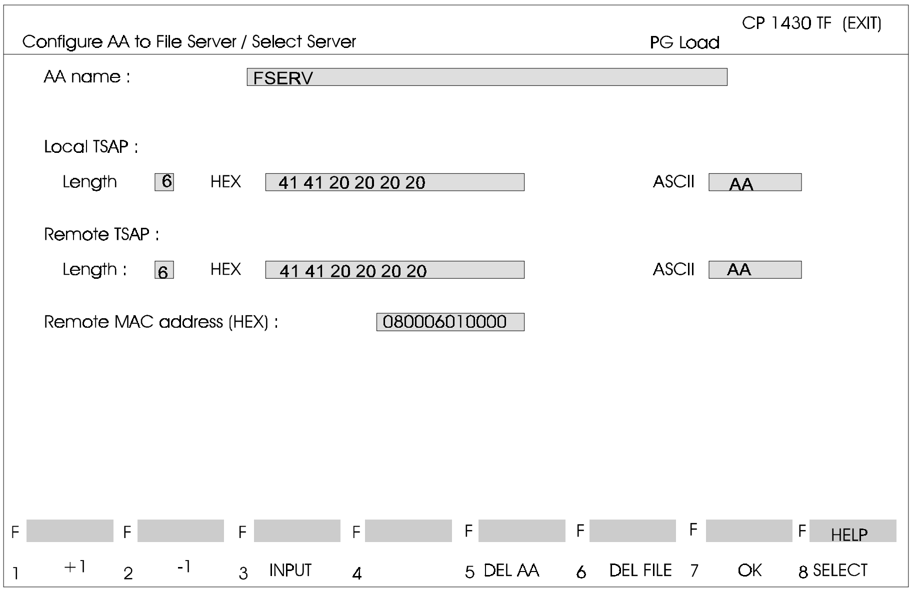


Fig. 5.5: Application Association Dialog TF File Server

Input Fields:

AA name: Name of the file server application association (must be unique for the file) This is used to select the file servers in the "Transfer Functions" dialog.
Possible values : 32 characters

Local TSAP: Service access point of the PG on which domains will be loaded or from which they will be saved.

Remote TSAP: Service access point of the file server.

Representation and input of the TSAP:

Length: This field has the default "8" entered. If you require connections to non-SIMATIC S5 stations it may be necessary to specify shorter lengths.
Possible values: 1 character

HEX: The individual bytes of the TSAP ID must be entered in hexadecimal format in groups of two (values from 00 to FF). The groups of two are easier to read if they are separated by blanks.
Possible values: 8 bytes

ASCII: 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.
Possible 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 MAC address: The default Ethernet address 080006010000 is displayed, this can, however, be modified.
Possible values: 12 characters

Function Keys (with an additional or context-specific meaning)**:**

F1, F2 +1, -1	With these function keys, you can load and edit the next or previous connection block in the file.
F3 INPUT	You can set up a new connection block. An empty “Configure AA to File Server / Select Server” dialog is displayed. The only available function keys are then F7 OK and F8 SELECT/HELP. With F7 OK, the new connection block is entered but not yet saved. The data are saved by pressing F7 again as described below.
F4 DEL AA	The currently displayed connection block can be deleted.
F5 DEL FILE	You can delete the whole connection file. You will first be prompted to confirm your intention: delete yes/no?
F7 OK	The currently displayed application association is selected and is used as the default for subsequent functions. You exit the dialog, all changes are saved.

5.4.3 Configure AA to PLC / Select PLC

With **PG Load | AA: PG -> PLC**, you can configure PLC application associations.

PLC application associations are connections between the PG and a PLC. These connections are used to transfer jobs for file server access and PI service jobs.

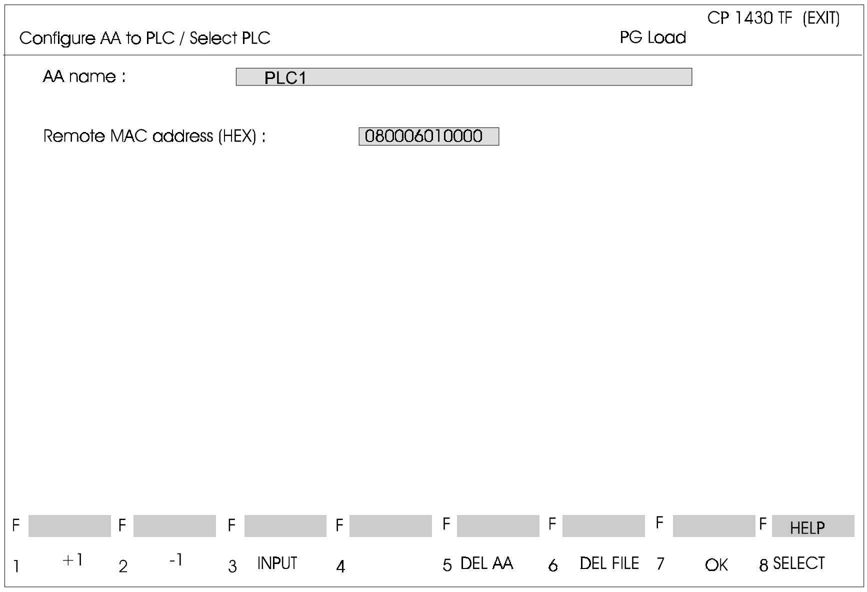


Fig. 5.6: Application Association Configuration / PLC Selection

Input Fields:

AA name:	Name of the application association to the PLC (must be unique for the file). Possible values. 32 characters
MAC address:	The default MAC address 080006010000 is displayed, this can, however, be modified. Possible values: 12 characters

Function Keys (with an additional or context-specific meaning)**:**

F1, F2 +1, -1	With these function keys, you can load and edit the next or previous connection block in the file.
F3 INPUT	You can set up a new connection block. An empty “Configure AA to PLC / Select PLC” dialog is displayed. The only available function keys are then F7 OK and F8 SELECT/HELP. With F7 OK, the new connection block is entered but not yet saved. The data are saved by pressing F7 again as described below.
F4 DEL AA	The currently displayed connection block can be deleted.
F5 DEL FILE	You can delete the whole connection file. You will first be prompted to confirm your intention: delete yes/no?
F7 OK	The currently displayed application association is selected and is used as the default for subsequent functions. You exit the dialog, all changes are saved.

5.4.4 Using Transfer Functions

The transfer functions support the conversion and transfer of S5 program and data files to the TF file server. 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 File Server.
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 file server. The connection to the destination is established by selecting the destination station in the connection file for the file server.

Domain-specific variables that can be defined with the PG load system are part of the load file. You can define variables using this dialog 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>

If you want to load a domain that was created for the CP 143 with COM 143, and it contains variables, the domain must be converted to the CP 1430 format using the PG LOAD | CP 143 Domain Converter function (see Section 5.4.6)

FETCH: Transfer Domains from the TF File Server to the PG
Specifying the server file determines the domain to be transferred with the fetch function. The destination is the local S5 program file.

Domain-specific variables, which can also be defined with the PG Load program, are automatically transferred along with the domain and reconverted into the file <xxxxxxST.VAR>.

The input dialog for the transfer functions:

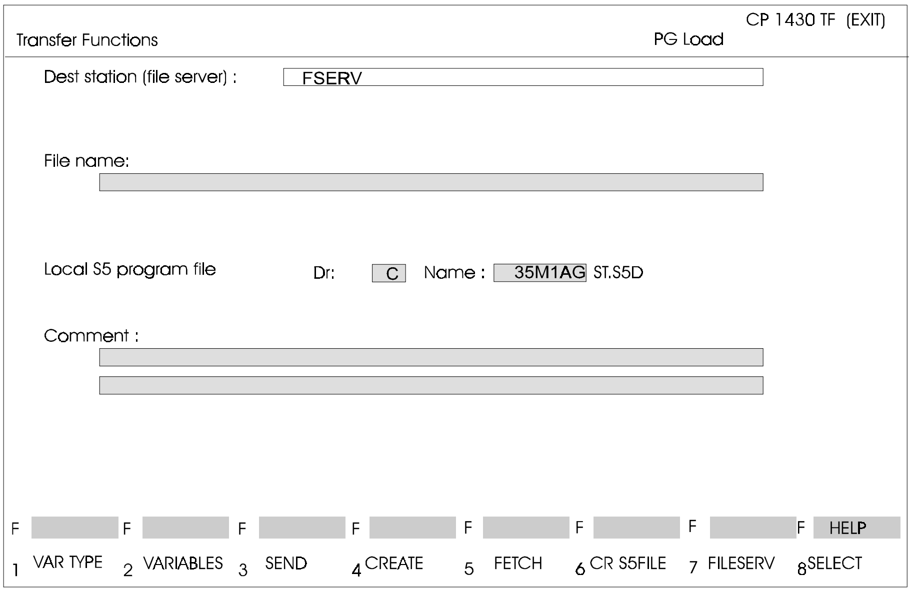


Fig. 5.7: Transfer Functions for Load Files

Input Fields::

Dest station (file server):	In this case, the logical partner name of the last selected TF file server application association is displayed. With F7, (FILESERV) you can page through the connection file and select a new partner = destination station.
File name :	File under which the load file will be stored (in the syntax of the destination system) on the server (destination) station. Possible 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.
Possible values: max. 128 characters

Function Keys (with an additional or context-specific meaning):

F1 VAR TYPE	Analogous to the COM function, you can edit variable types.
M 2-3-1	This is described in Chapter 4
F2 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 dialog PG Load DOMAIN SPECIFIC VARIABLES is displayed. In terms of use and possible inputs, the dialog corresponds to the COM dialog M2-5.1 (see accompanying supplement with COM 1430 TF dialogs) and Chapter 4 '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 file server 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 FILESERV	With this function, the servers entered in the connection file (file server) are displayed. The selected server is displayed in the 'Dest station' output field.

5.4.5 Using Host Functions

Whereas the functions introduced so far (connection configuration and transfer functions) are used to prepare the system and the PG, the host functions support the automation system during operation.

You call the host functions by selecting PG Load | Host Functions and the dialog below is displayed

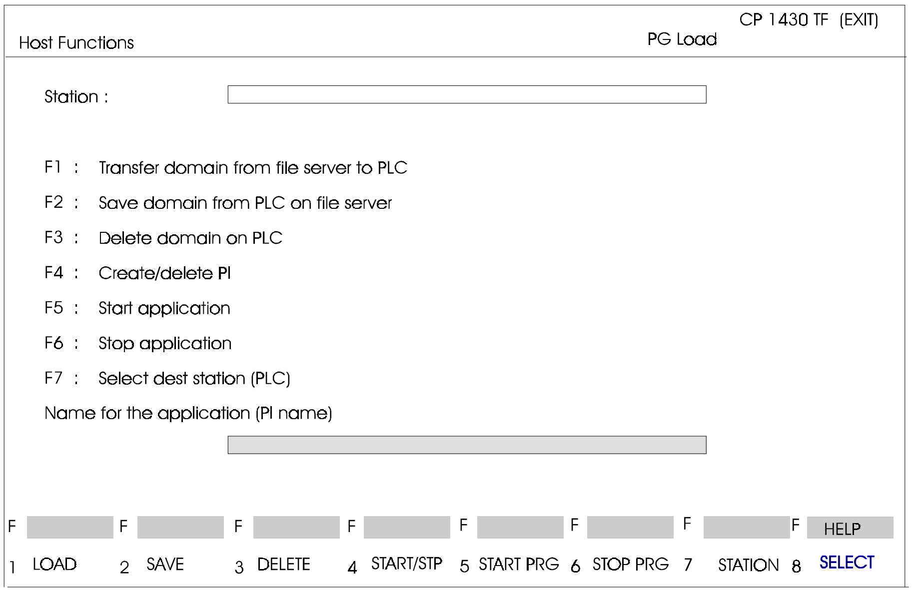


Fig. 5.8 Input Dialog - Host Functions

Input Fields:

Station (PLC): The destination station (PLC) to which the following services refer. You can select a station with F7.
Possible values: max. 32 characters

Program invocation name: Name to which PI services for program control selected with the keys F4 to F6 refer:

- create program invocation,
- delete program invocation,
- start program
- stop program

Possible values: max. 32 characters

Along with THE 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 (with an additional or context-specific meaning)**:**

F1 LOAD	Load a PLC (-> next dialog PG Load Load PLC)
F2 SAVE	Save the domains of a PLC (-> next dialog PG Load Save PLC)
F3 DELETE	Delete all the domains in the PLC (confirmation prompted)
F4 START/STP	Create/delete a program invocation. If you select the system PI this means start/stop the PLC.
F5 START PRG	The program invocation with the specified name is started on the PLC from the existing domains.
F6 STOP PRG	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 connection 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.

5.4.5.1 Load PLC

One or a maximum of two domains in a dialog can be transferred from the file server to a CPU of the PLC (the file server can also be the local PG).
You display the "PG Load - Load PLC" dialog with the F1 key in the "PG Load Host Functions" dialog.

Fig. 5.9: Load PLC Dialog

Output Fields:

PLC appl. assoc.: Name of the application association to the PLC to be loaded with domains. The displayed PLC application association was selected in the initial dialog for host functions.
Possible values: (32 characters)

Input Fields:

File server appl. assoc.:	Application association name of the server from which the domains will be loaded, (must be configured on the CP). Possible 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 file server. Situation b: PG = file server If, however, the application association 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. Possible values: 32 characters:
Saved in file:	Name of the server file in the syntax of the server system. Possible values: 64 characters:
Parameters (DOMAIN):	Name of the data domain to be loaded. Possible values: 32 characters:
Saved in file:	Name of the server file in the syntax of the server system. Possible values: 64 characters:
CPU no.	With the CPU number, you specify the CPU in the programmable logic 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.**

Function Keys (with an additional or context-specific meaning)**:**

F7 OK	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.

5.4.5.2 Save PLC

A PLC domain can be saved on the file server. This means that the PG instructs the PLC to send the data to the server. The application association name is handled in the same way as in the load PLC function.
You display the "PG Load - Save PLC" dialog with the F2 key in the "PG Load - Host Functions" dialog.

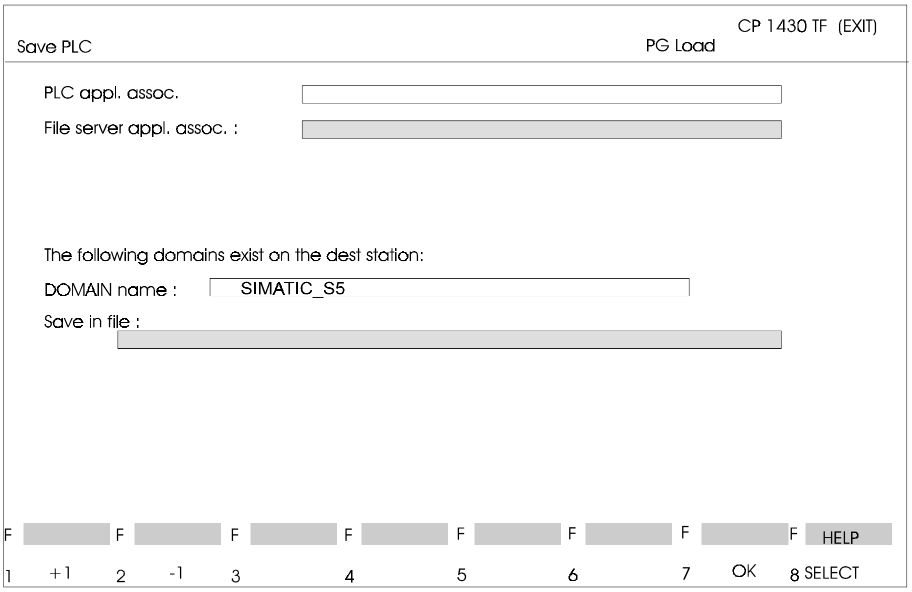


Fig. 5.10: Save PLC Dialog

Output Fields:

PLC appl. assoc. Name of the application association to the PLC from which the domains will be saved. The displayed PLC application association was selected in the initial dialog for host functions.
Possible values: (32 characters)

DOMAIN name: Name of a domain existing on the destination station. The domain displayed here can be saved. You select a domain on the destination station with function keys F1 or F2.

Input Fields:

File server-appl. assoc.: Name of the application association to the server on which the domain will be saved.
(The application association must be configured on the CP)
Possible 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 file server.

Situation b: PG = file server
If, however, the connection name of the server is "PG", the domains are transferred directly from the PLC to the PG = 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 resource.

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.
Possible values: (64 characters)

☞ **If a domain was loaded on the CPU via the CP 1430 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 (with an additional or context-specific meaning)**:**

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.

5.4.6 Converting Files from COM 143 to COM 1430

If you want to load a domain that was created for the CP 143 with COM 143, and it contains variables, the domain must be converted to the CP 1430 format using the **Utilities | PG Load | FD(CP143)->CP(FD1430)** function. The following dialog is then displayed:

Fig. 5.11: Converting Domain Files

Input Fields

Source: Here, you specify the drive and file name of the COM 143 domain file.

Dest: Here, you specify the drive and file name of the COM 1430 domain file and the variable file. The domain file can then be used directly as a loadable file. The variable file can be processed with the variables editor that can be called in the Transfer Functions dialog.

❑

Notes

6 The Request Editor

Topics in this Chapter

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 dialogs for the job buffers of the individual TF services.

6.1 Overview

6.1.1 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 the STEP5 basic package (LAD, CSF, STL).

Integration in SINEC NCM
The tool runs as a component of the COM 1430 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 features are as follows:

➢ The screen is divided into a dialog header, dialog area, message line and function key menu.

➢ The functions of the tool can be called using the COM 1430 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, start the tool on your PG and select the available functions one after the other.

6.1.2 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 stored 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

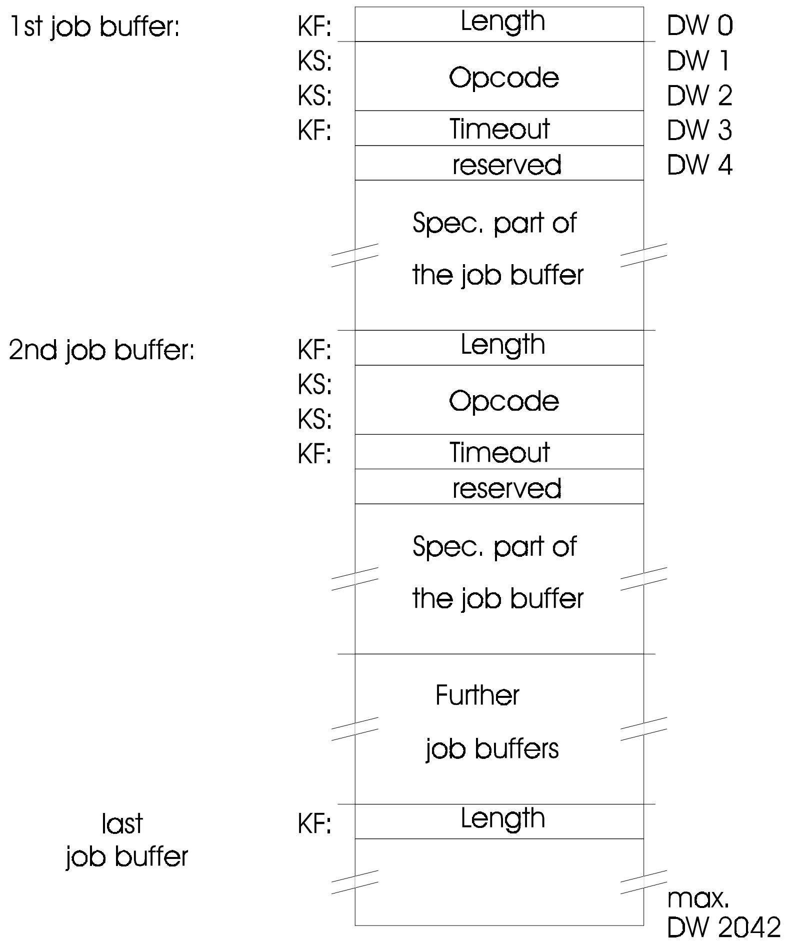


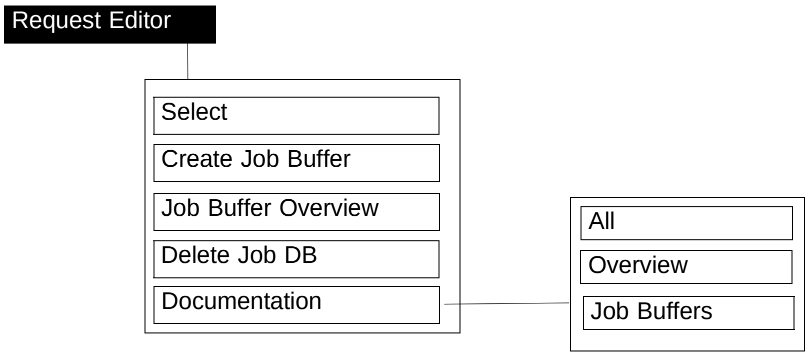
Fig. 6.1: Structure of a Data Block with Job Buffers

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.

6.2 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:

Select
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 dialog.

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 or printed out.

Using the Request Editor

To create job buffers with the Request Editor tool, follow the sequence outlined below:

Steps	and Corresponding Functions
Start the tool	**Utilities | Request Editor->**
Specify the S5 program file and data block Section 6.3	**Select**
Specify job buffers for TF services Section 6.4 .. 6.6	**Create Job Buffer** **Job Buffer Overview** **Delete Job DB**
Document the job buffer Section 6.7	**Documentation->** **All** **Overview** **Job Buffers**
Transfer data blocks to the PLC	**Function of S5DOS-KOMI**

Fig. 6.2: Configuring TF Job Buffers

6.3 Request-Editor | Select

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 setup dialog has the following layout:

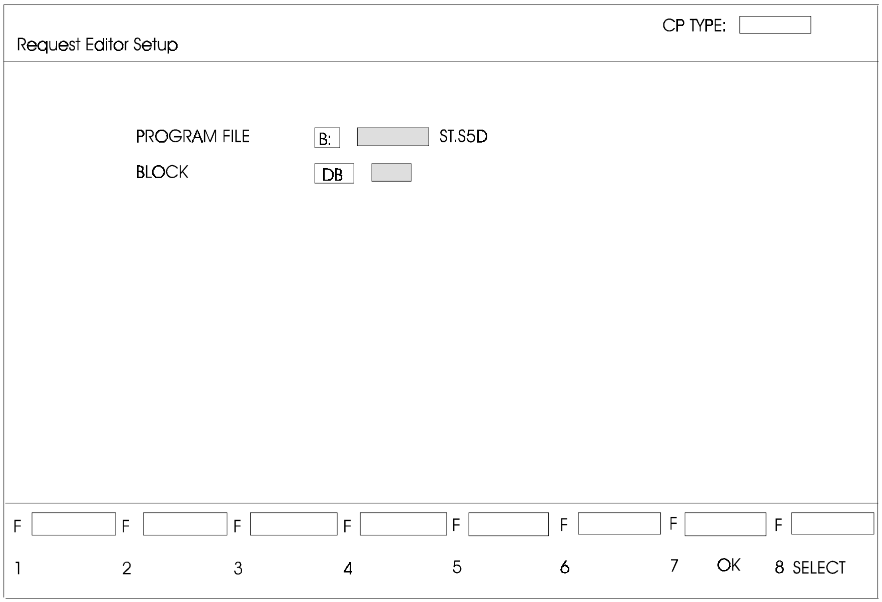


Fig. 6.3: Request Editor Initialization

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 (with an additional or context-specific meaning)**:**

F7 OK	Enters the data you have input

6.4 Specifying the Job Buffers for TF Services

6.4.1 Create Job Buffer

Depending on the status of the program file you selected in the initialization dialog, you obtain either an empty dialog or the input dialog of an existing job buffer.

The input dialog appears as follows:

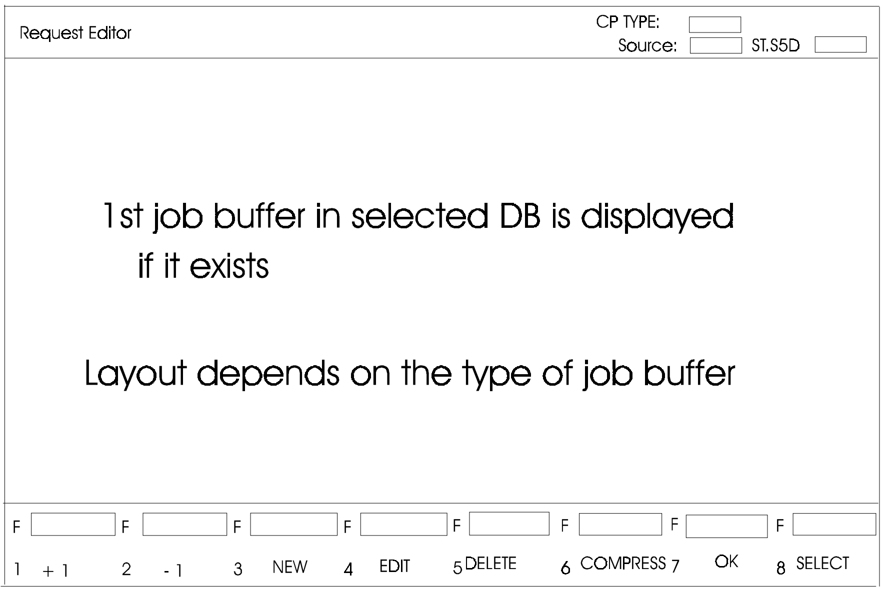


Fig. 6.4: Input Dialog

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:
ERROR IN DATA BLOCK, DELETE?

Function Keys (with an additional or context-specific 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 dialog: 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 dialog.

F6 COMPRESS	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 DONE, 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.

6.4.2 Type Selection Dialog for TF and Other Services

The type selection dialog provides you with an overview of the job types available for communication.

This dialog is displayed by selecting Request Editor | Create Job Buffer and pressing the NEW function key.

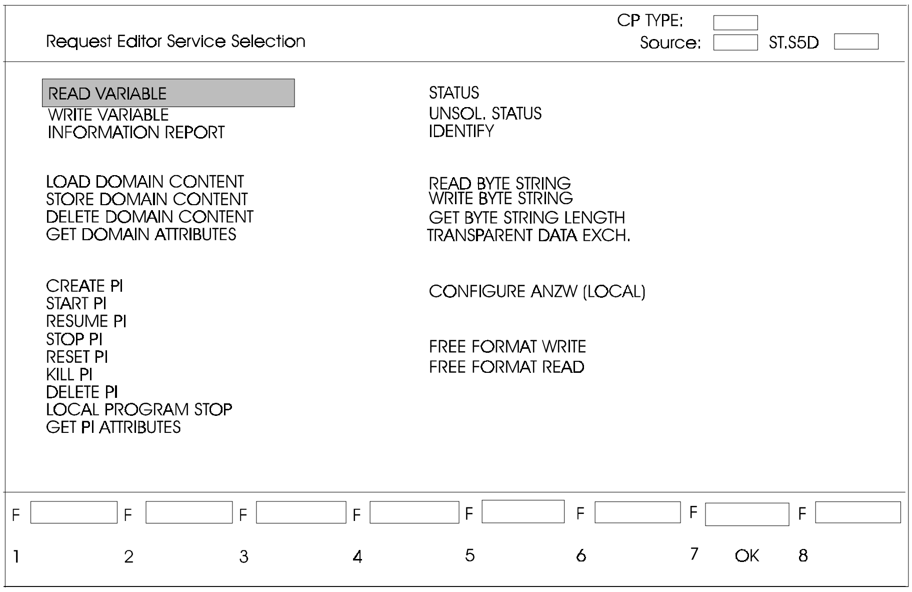


Fig. 6.5: Type Selection Dialog

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 dialog for entering a job buffer.

Function Keys (with an additional or context-specific meaning):

F7 OK	Generates a job buffer for the currently selected function Next dialog: 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 selection dialog:

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 file server.
STORE DOMAIN CONTENT	Save a program or parameters of a program of the local CPU or a remote CPU on a file server (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.

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 acknowledgment.
IDENTIFY	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.
GET 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 CP 1430.

Select a function with the cursor keys, the currently selected function is displayed inversely on the dialog.
default: READ VARIABLE.

6.4.3 Variable Services

Read Variable

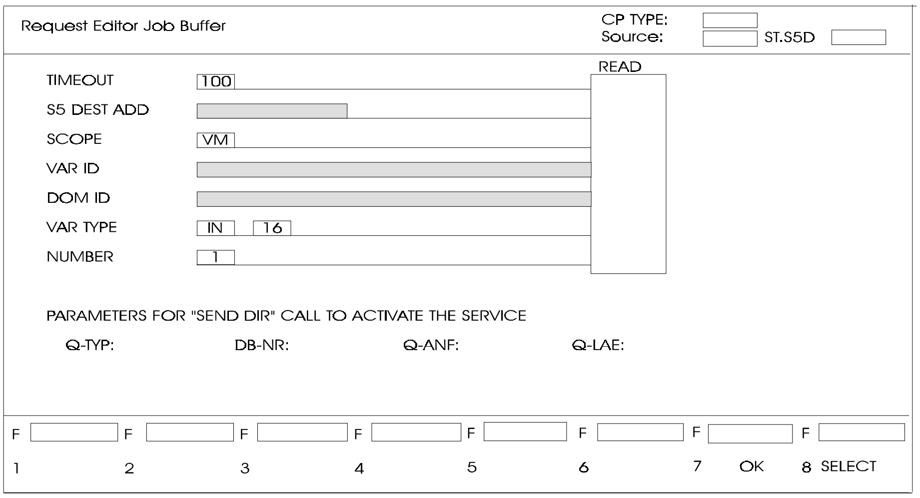


Fig. 6.6: Read Variable Dialog

Input Fields:

TIMEOUT:

Acknowledgment 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.

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042

Example: DB 12 0

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 configuration by separating the individual parameters of the S5 address with blanks.

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

SCOPE: Specifies the validity of the requested variable in the other system.

Permitted entries (default: VM):

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".

AA: application association-specific
The requested variable is only accessible in the other station via a particular application association. The application association is identified by the SEND-DIR call parameters "SSNR" and "ANR" when the job buffer is transferred.

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:
Specifies the data format type
(specification as in COM 1430 TF).

default: IN 16

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 dialog. In addition to this, call parameters for the "SEND DIR" call to trigger the service are also displayed.

Note:
Complex or structured variables (structures or arrays) and user-defined variable types must be configured with COM variable definitions. Use the "remote definitions" of variables. In the Request Editor, you must do the following:

- enter blanks in the variable type field.

- specify DB no. = 0 to indicate that the destination address is configured on the CP.

The variable is then configured in the appropriate application association in the REMOTE DEFINITIONS dialog.

When you press F7 to enter your data, the Request Editor will, if applicable, display a message to indicate that a variable must be configured.

<VARNAME MUST BE CONFIGURED!>

Data type	Data format	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, 4 bytes format see below	-
TD	6	Time 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

Table 6.1: TF Type and Meaning

The table lists the types that can be specified under variable type:

Function Keys (with an additional or context-specific meaning):

F7 OK	Completes the input and stores the newly edited job buffer in the main memory of the programming device.

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

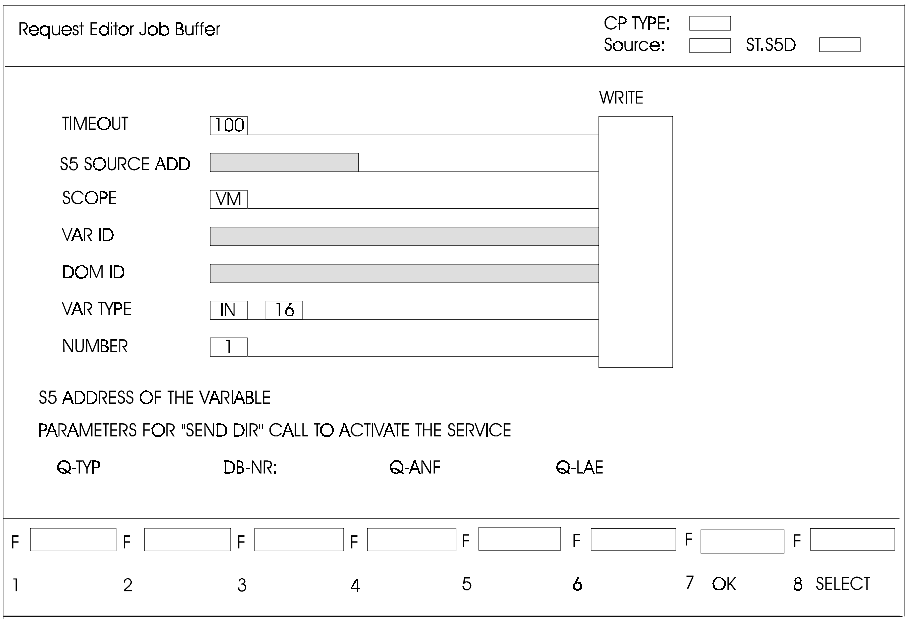


Fig. 6.7: Write Variable Dialog

Input Fields:

TIMEOUT: Same as for read variable.

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042

Example: DB 12 0

It is not possible to specify the length, this is determined implicitly by the type of variable.

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

SCOPE: Specifies the validity of the requested variable in the other system.

Permitted entries VM, DO, AA (default: VM)

VM: VMD-specific
The variable to be written is valid throughout the other station (no restrictions).

DO: Domain-specific
The variable to be written is only valid in the other station within the area specified by the "domain name".

AA: application association-specific
The variable to be written is only accessible in the other station via a particular application association. The application association is identified by the SEND-DIR call parameters "SSNR" and "ANR" when the job buffer is transferred.

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 6.4.
2nd input field:
Specifies the data format
(specification as in COM 1430 TF)
default: IN 16

NUMBER: Number of elements in arrays
default: 1 (no array)

☞ **Note:**
Complex or structured variables (structures or arrays) and user-defined variable types must be configured with COM variable definitions. Use the "remote definitions" of variables. In the Request Editor, you must do the following:

- enter blanks in the variable type field.

- specify DB no. = 0 to indicate that the destination address is configured on the CP.

The variable is then configured in the appropriate application association in the REMOTE DEFINITIONS dialog.

When you press F7 to enter your data, the Request Editor will, if applicable, display a message to indicate that a variable must be configured.

<VARNAME MUST BE CONFIGURED!>

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 THE VARIABLE").

F7 OK — Completes the input and stores the newly edited job buffer in the main memory of the programming device. Next dialog: Input

Information Report

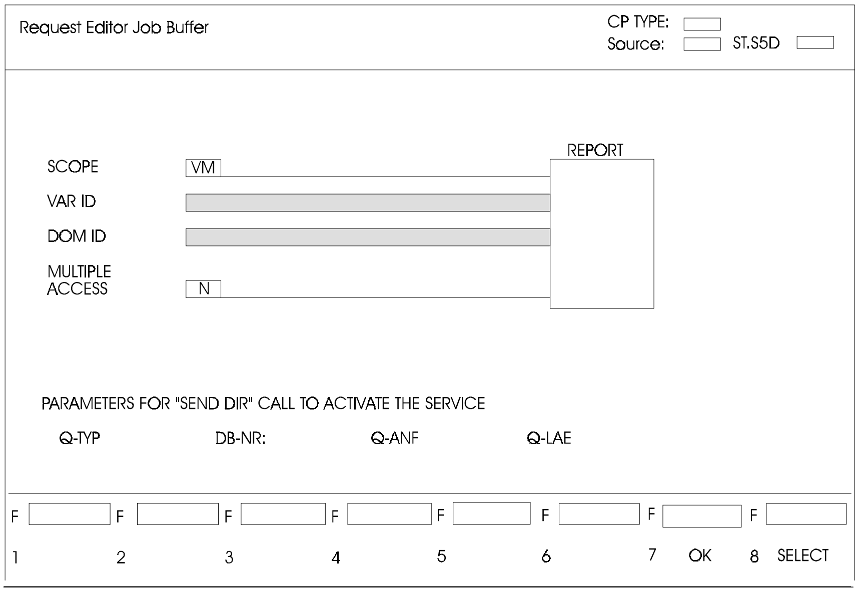


Fig. 6.8: Information Report Dialog

Input Fields:

SCOPE: Specifies the validity of the requested variable in the local system.

Permitted entries VM, DO, AA (default: VM)

VM: VMD-specific
The variable to be reported is valid throughout the local station (no restrictions).

DO: domain-specific
The variable to be reported is only valid in the local station within the area specified by the "domain name".

AA: application association-specific
The variable to be reported is only accessible in the local station via a particular application association. The application association is identified by the SEND-DIR call parameters "SSNR" and "ANR" when the job buffer is transferred.

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" dialog (VMD variables editor).

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Free Format Write

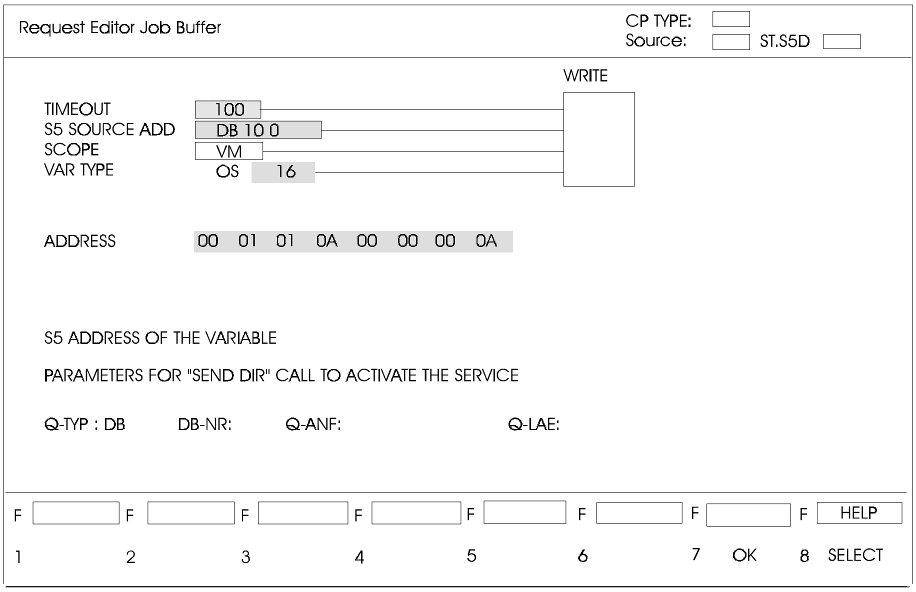


Fig. 6.9: Write Free Format Dialog

Input Fields:

TIMEOUT

Same as for read variable.

For more information about TIMEOUT see page 3-13.

S5 SOURCE ADDRESS:

Address in the local 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
Start0...2042

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

SCOPE

Specifies the area in which the variable to be written is valid in the other system.

Permitted entry: VM

The requested variable is valid throughout the whole station (no restrictions).

VARIABLE TYPE

Specifies the type of the requested variable.

Permitted entry: OS 1...4086

1st input field
specifies the basic type
always "OS"(octet string)

2nd input field
length of the type specified in the 1st input field
Octets = 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 in Section 4.3.1).

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Follow-on dialog: Input

Free Format Read

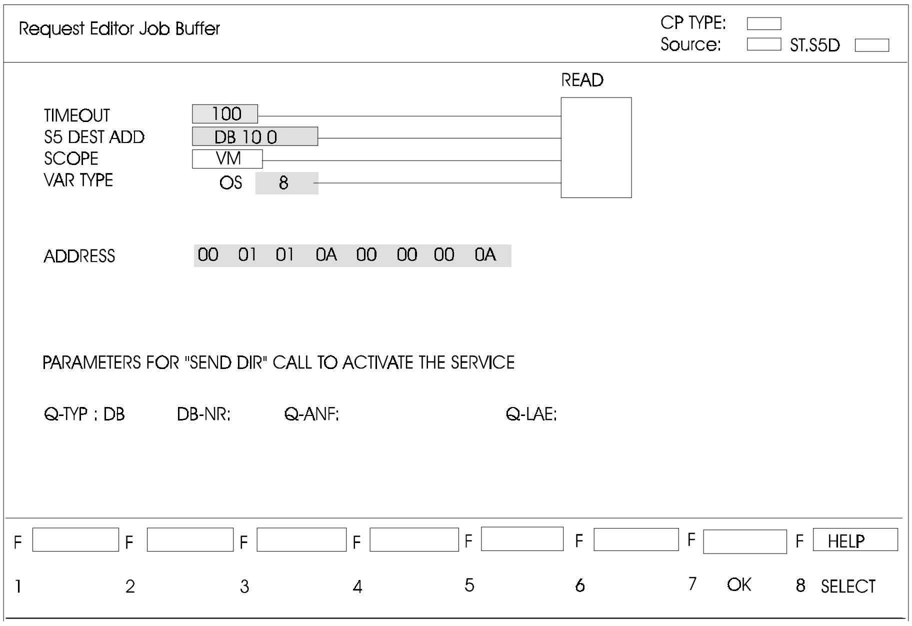


Fig. 6.10: Read Free Format Variable Dialog

Input Fields:

TIMEOUT

Same as for read variable.

For more information about TIMEOUT see page 3-13.

S5 SOURCE ADDRESS

Address in the S5 system at which the value of the read variable will be stored.

Source type:DB, DX
DB no.:0,1-255
Start0...2042.

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

SCOPE Specifies the area in which the requested variable is valid in the other system.

Permitted entry: VM

The requested variable is valid throughout the station (no restriction).

VARIABLE TYPE Specifies the type of the requested variable.

Permitted entry: OS 1...4086

1st input field
specifies the basic type
always "OS"(octet string)

2nd input field
length of the type specified in the 1st input field
Octets = 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 dialog. 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 programming device. Next dialog: Input

6.4.4 Domain Services

Load Domain Content

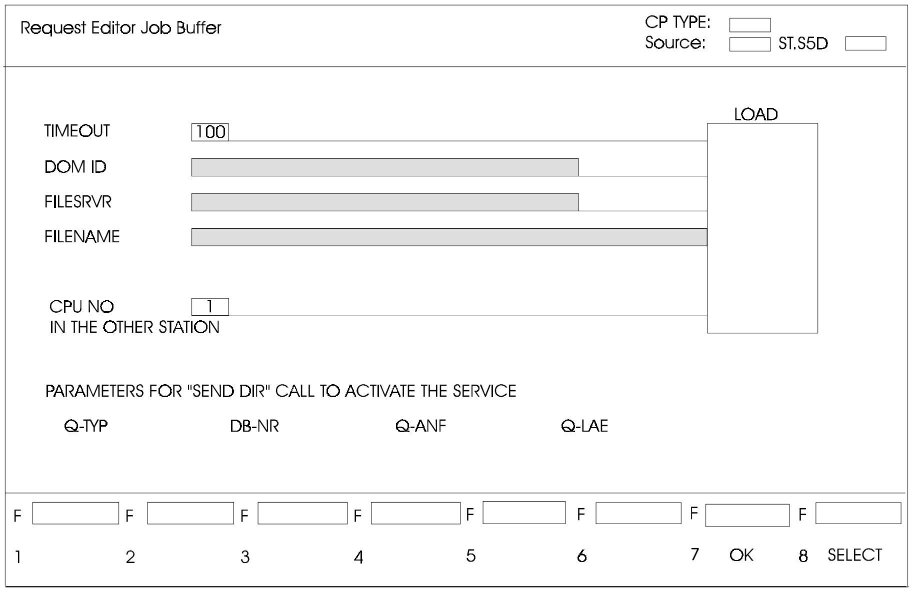


Fig. 6.11: Load Domain Content Dialog

Input Fields:

TIMEOUT: Acknowledgment 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.

For more information about TIMEOUT see page 3-13.

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 file server.

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 entries: 1..4

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Store Domain Content

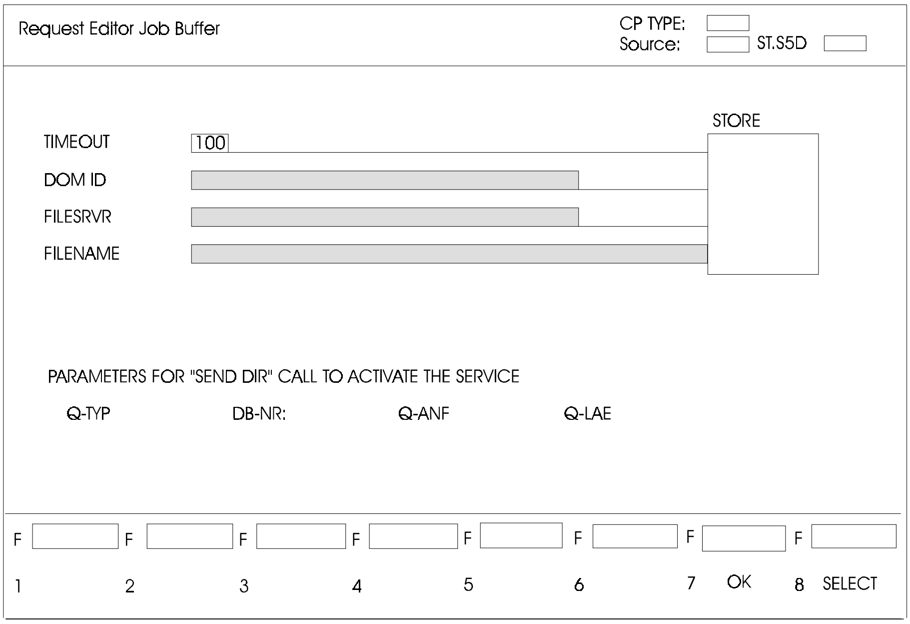


Fig. 6.12: Store Domain Content Dialog

Input Fields:

TIMEOUT: Same as for load domain

For more information about TIMEOUT see page 3-13.

DOM ID: The domain specified here will be stored on the file server.

FILE SERVER: Specifies the name of the application association to be established to the file server.

FILE NAME: The domain will be stored on the file server 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 dialog. 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 programming device. Next dialog: Input

Delete Domain

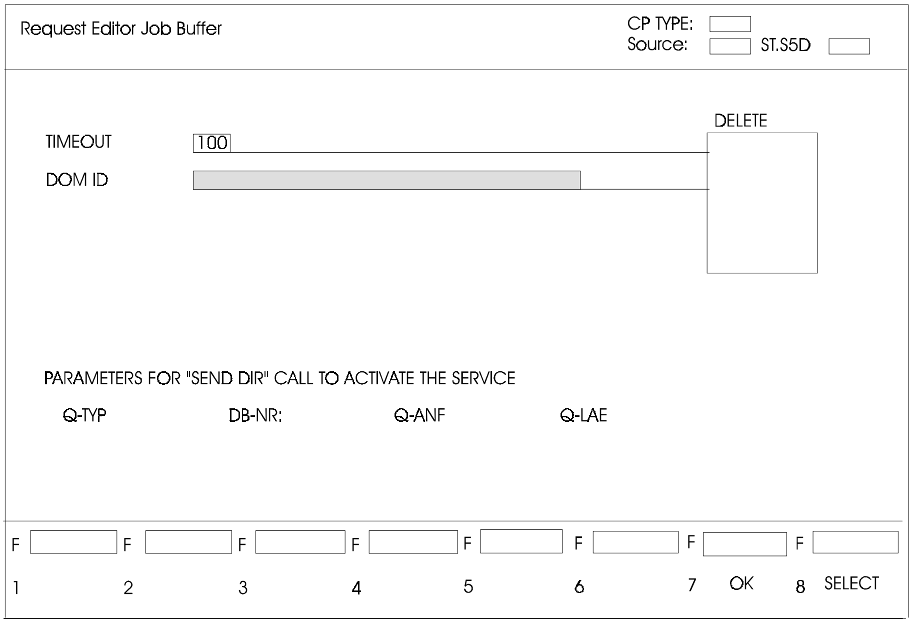


Fig. 6.13: Delete Domain Content Dialog

Input Fields:

TIMEOUT: Same as for load domain

For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Get Domain Attributes

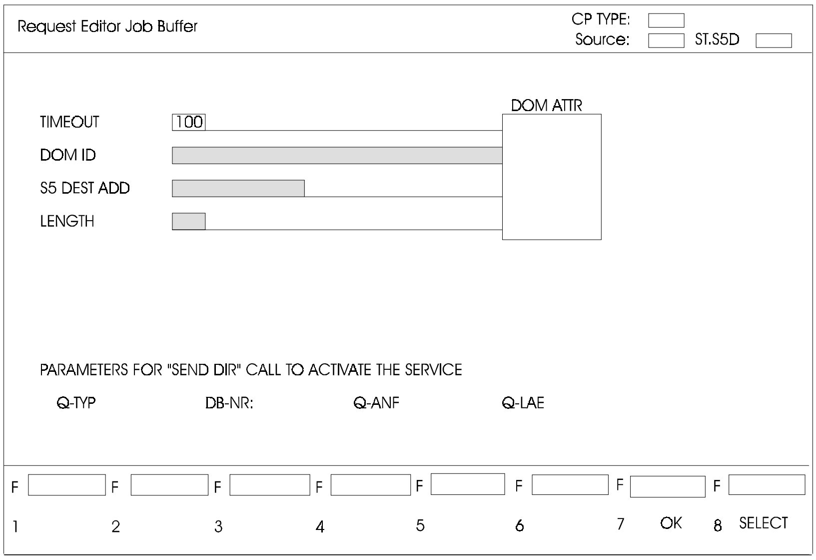


Fig. 6.14: Get Domain Attributes

Input Fields:

TIMEOUT: Same as for load domain.

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042.

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

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 acknowledgment can be entered.

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

6.4.5 Program Invocation Services

Create PI

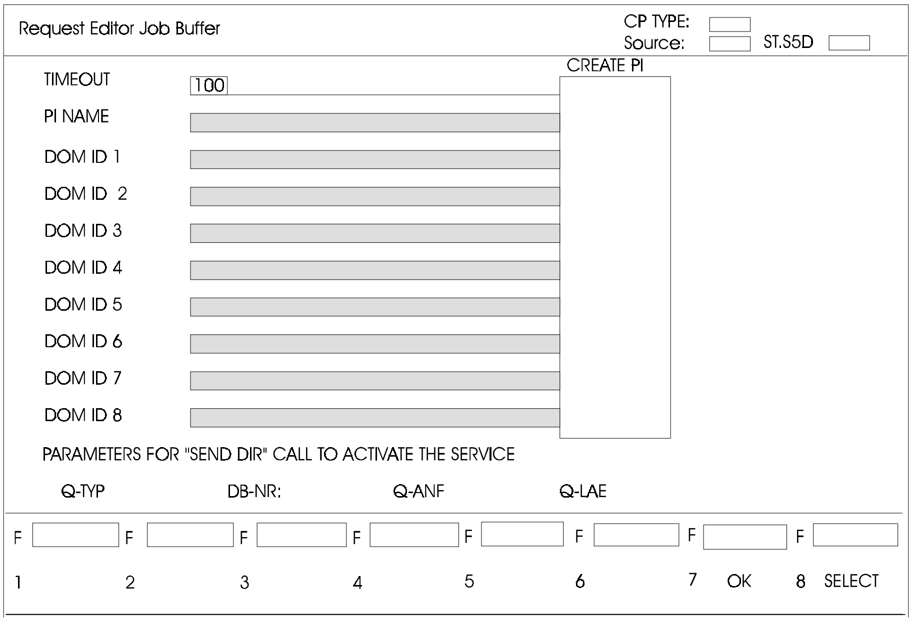


Fig. 6.15: Create PI Dialog

Input Fields:

TIMEOUT: Acknowledgment 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.

For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Start PI

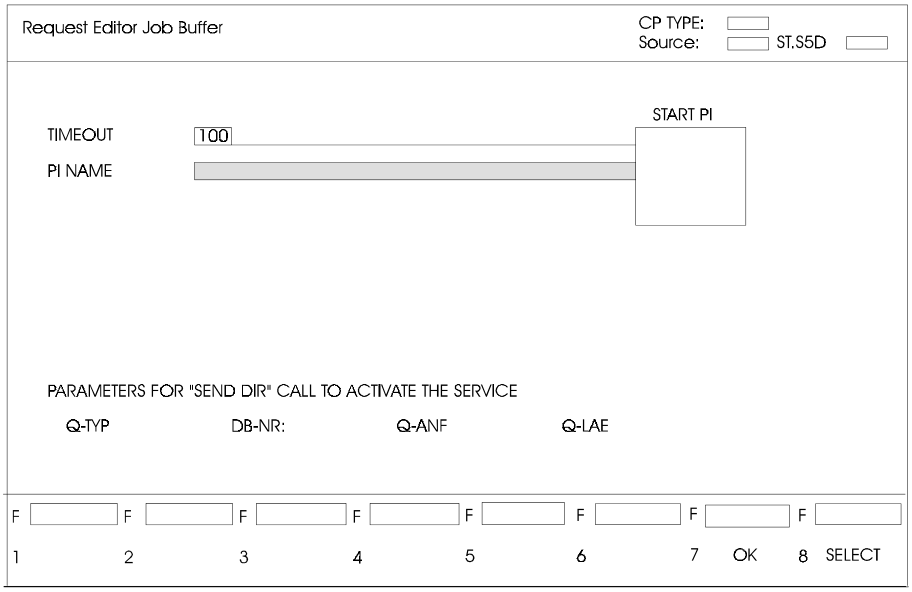


Fig. 6.16: Start PI Dialog

Input Fields:

TIMEOUT: Same as for create PI.

For more information about TIMEOUT see page 3-13.

PI NAME: The program invocation 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 dialog. 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 programming device. Next dialog: Input

Stop PI

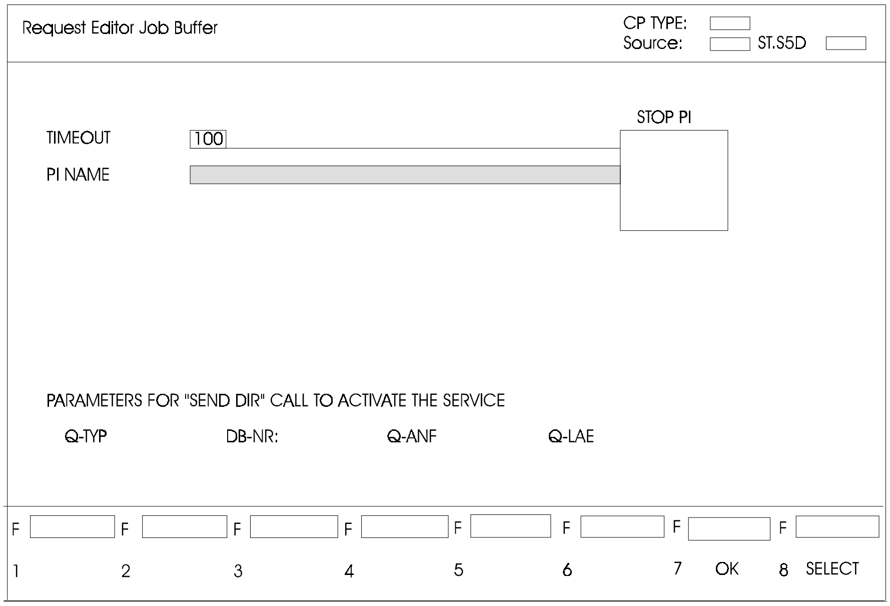


Fig. 6.17: Stop PI Dialog

Input Fields:

TIMEOUT: Same as for create PI.

For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Resume PI

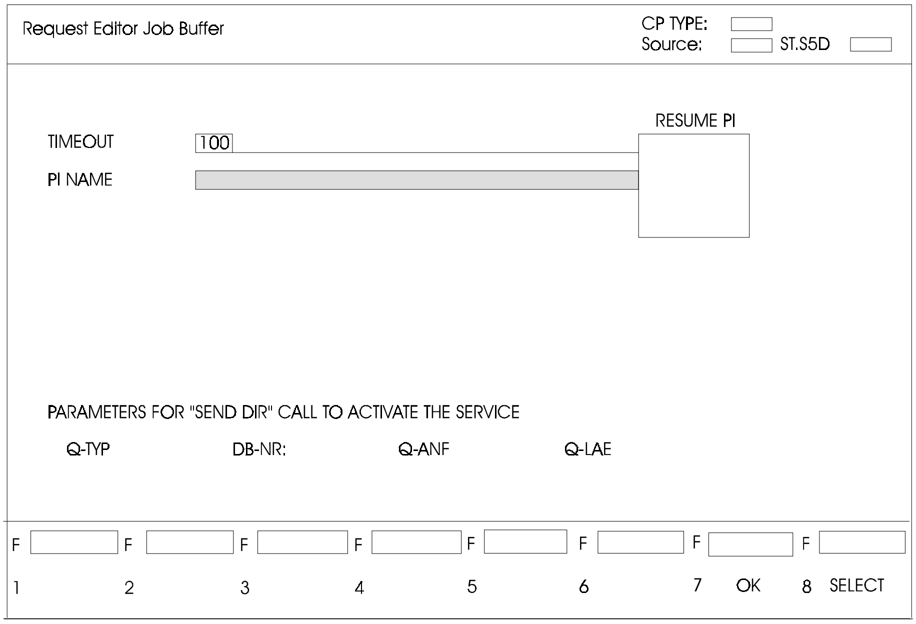


Fig. 6.18: Resume PI Dialog

Input Fields:

TIMEOUT: Same as for create PI.
For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Reset PI

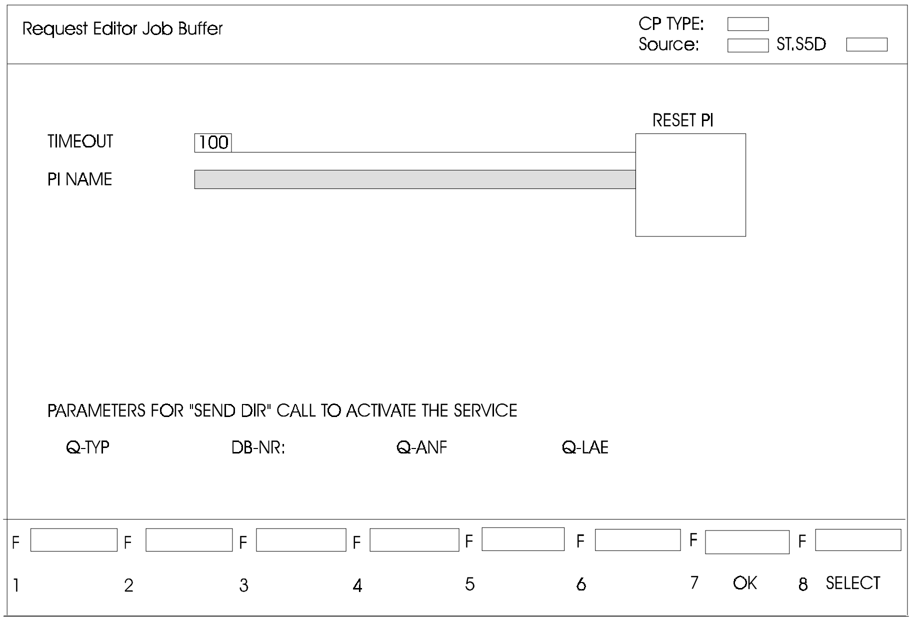


Fig. 6.19: Reset PI Dialog

Input Fields:

TIMEOUT: Same as for create PI.
For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Kill PI

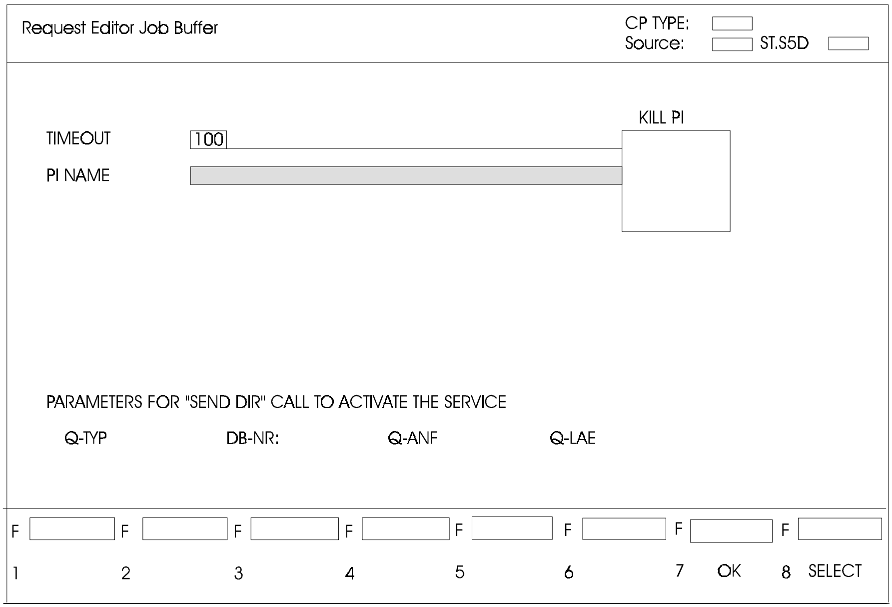


Fig. 6.20: Kill PI Dialog

Input Fields:

TIMEOUT: Same as for create PI.
For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Delete PI

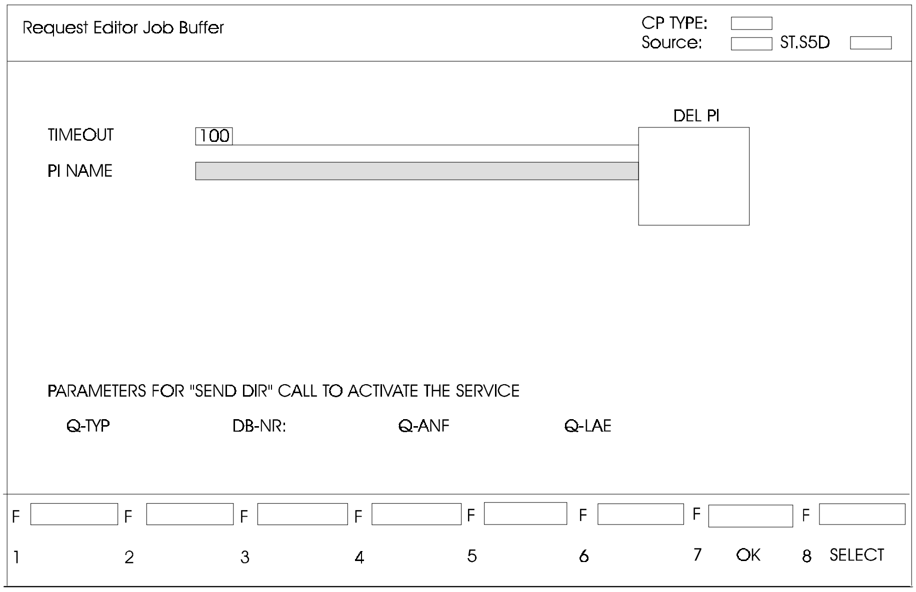


Fig. 6.21: Delete PI Dialog

Input Fields:

TIMEOUT: Same as for create PI.

For more information about TIMEOUT see page 3-13.

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 dialog. 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 programming device. Next dialog: Input

Local Program Stop

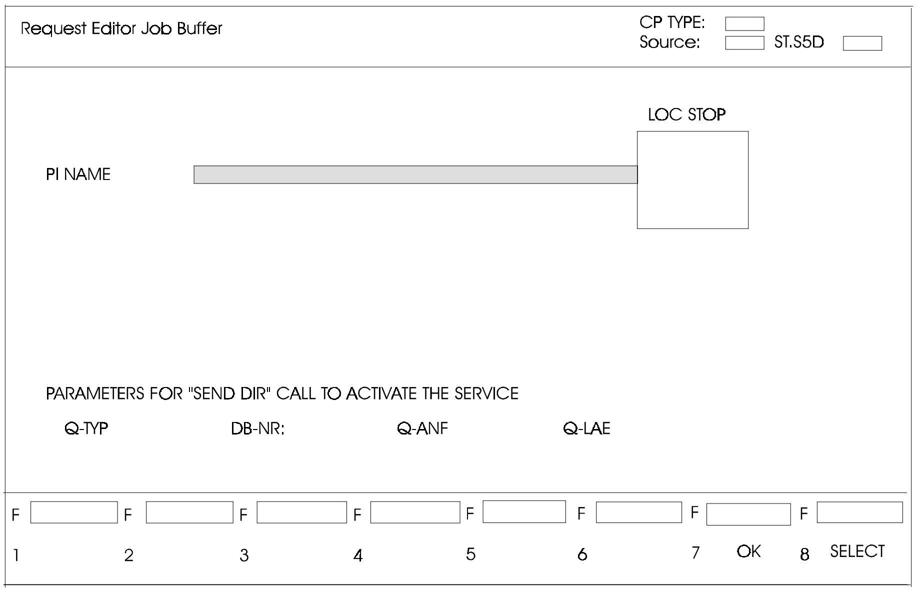


Fig. 6.22: Local Program Stop Dialog

Input Fields:

PI NAME: The program invocation with this name will be stopped. (local job; for more information see page 8-59).

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Get PI Attributes

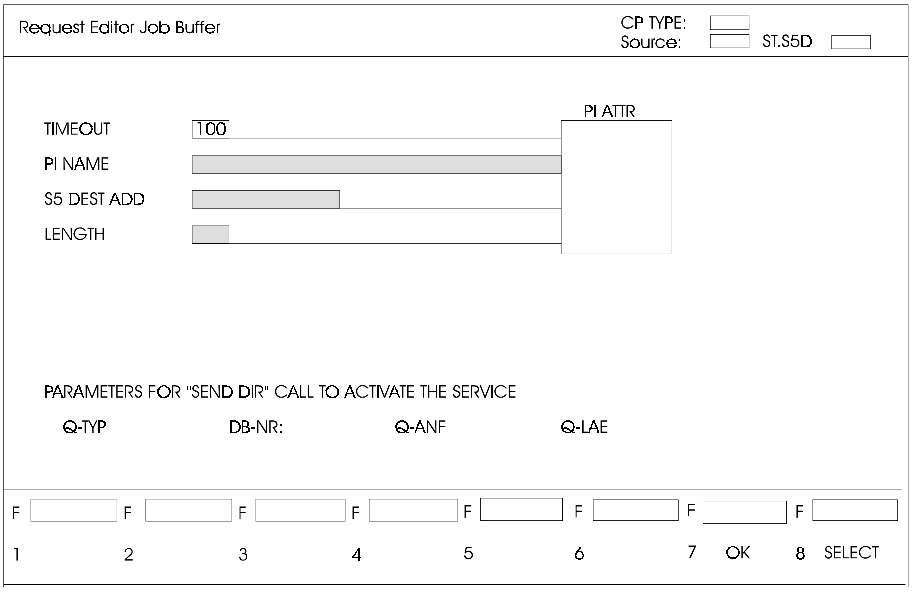


Fig. 6.23: Get PI Attributes Dialog

Input Fields:

TIMEOUT: Same as for create PI.

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042

Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.

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 acknowledgment can be entered.

Output Fields:
After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

6.4.6 VMD Services

Status

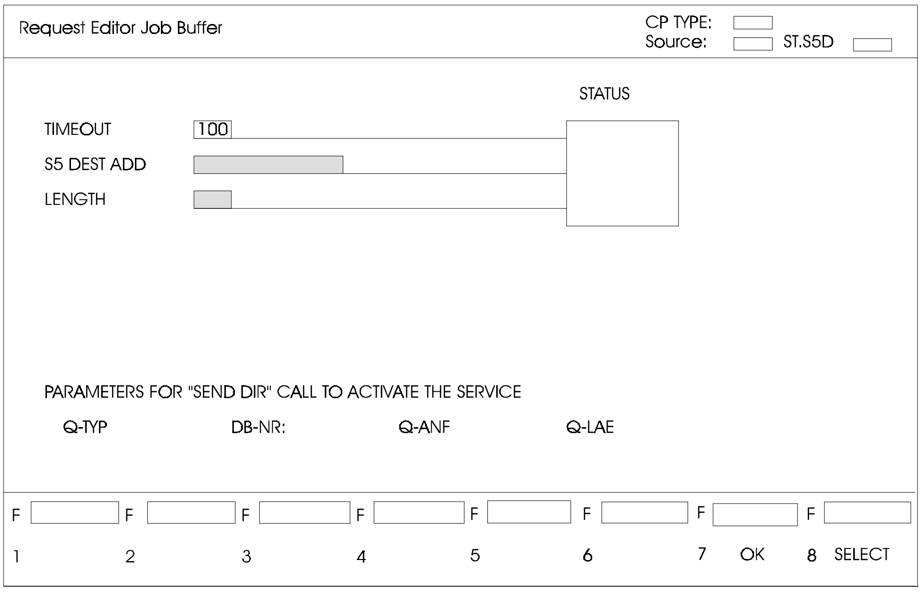


Fig. 6.24: Status Dialog

Input Fields:

TIMEOUT: Acknowledgment 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.

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

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 acknowledgment can be entered.

Output Fields:
After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Unsolicited Status

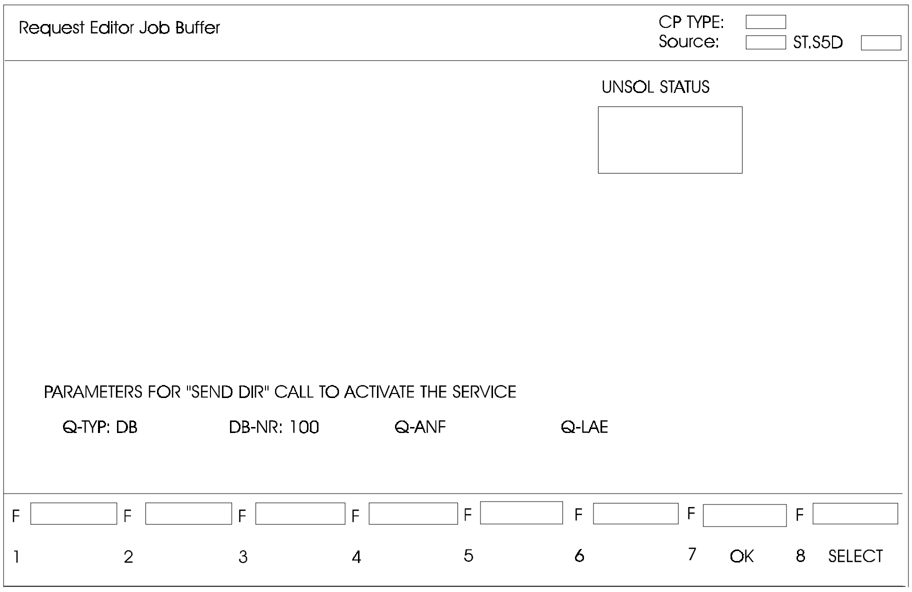


Fig. 6.25: Unsolicited Status Dialog

For the "unsolicited status" service, no entry is required in the dialog. 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 dialog. 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 programming device. Next dialog: Input

Identify VMD

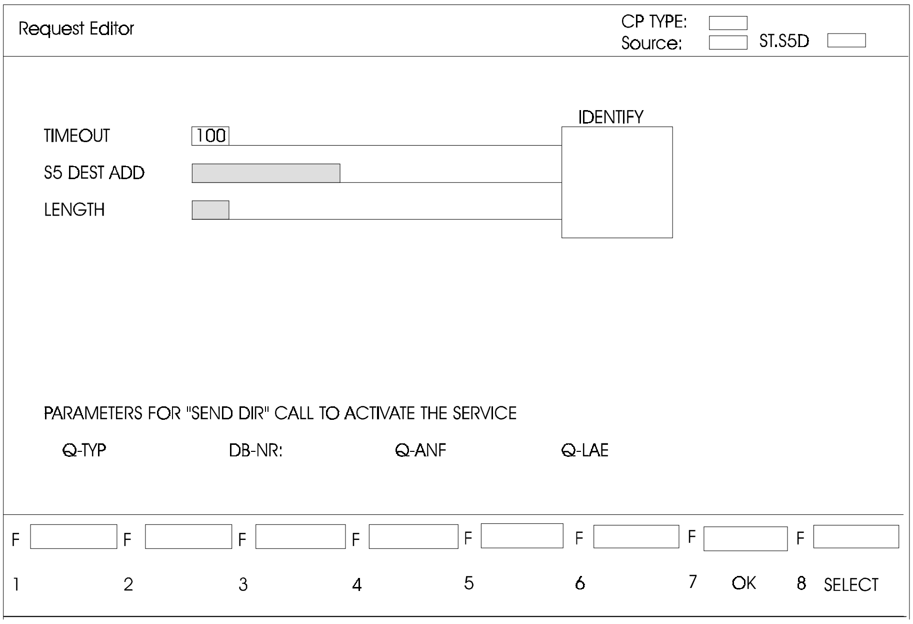


Fig. 6.26: Identify VMD Dialog

Input Fields:

TIMEOUT: Same as for status

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

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 acknowledgment can be entered.

Output Fields:
After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

6.4.7 Transparent Data Exchange (non-open services)

Read Byte String

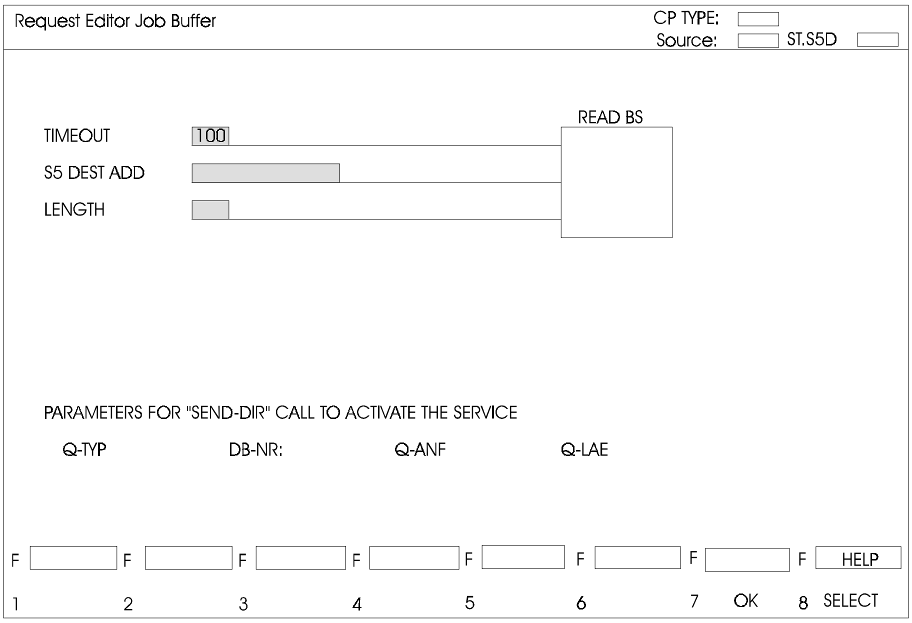


Fig. 6.27: Read BS Dialog

Input Fields:

TIMEOUT: Acknowledgment 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.

For more information about TIMEOUT see page 3-13.

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

LENGTH:

The "length" parameter specifies how many data words can be written by the CP into the data block.
Possible values: 1 .. 2043

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Write Byte String

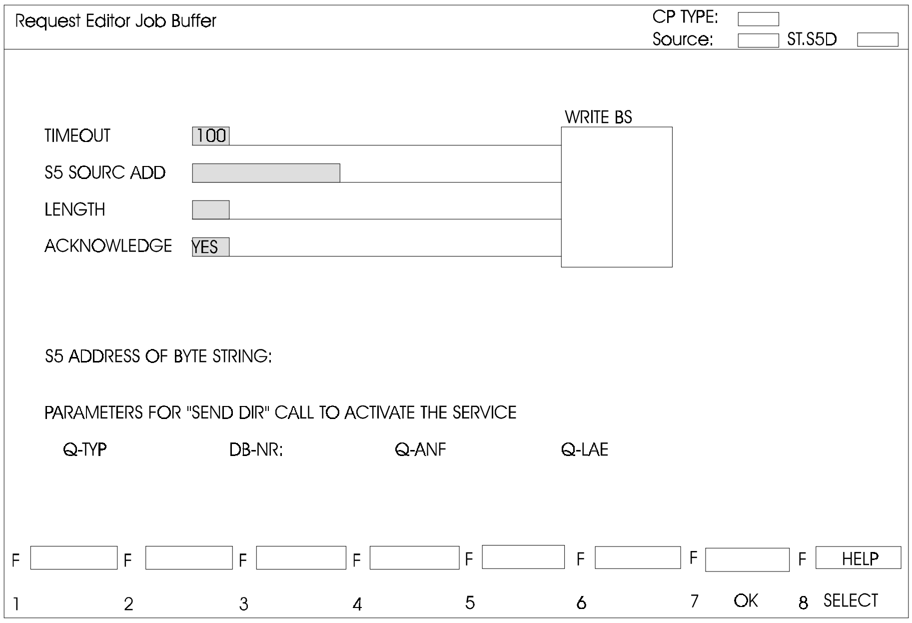


Fig. 6.28: Write BS Dialog

Input Fields:

TIMEOUT: Same as for read byte string.

For more information about TIMEOUT see page 3-13.

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.:0,1..255
Start:0..2042

Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.

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 function key F7 (OK) the complete S5 address of the byte string is displayed (output field "S5 ADDRESS OF BYTE STRING").

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 entries:

YES:
The "write byte string with acknowledgment" service is triggered.

NO:
The "write byte string without acknowledgment" service is triggered.

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Get Byte String Length

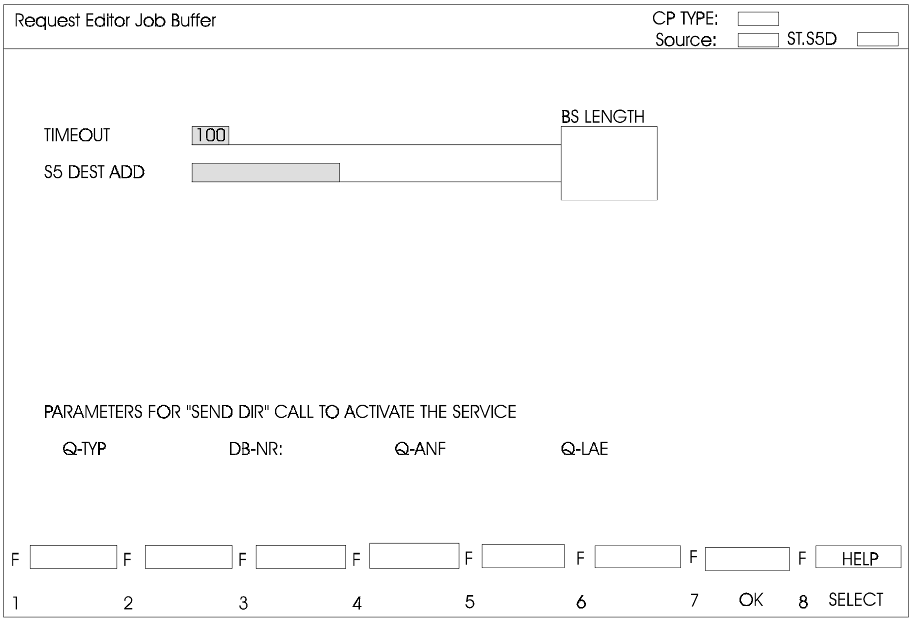


Fig. 6.29: BS Length Dialog

Input Fields:

TIMEOUT: Same as for read byte string

For more information about TIMEOUT see page 3-13.

S5 DESTINATION ADDRESS Address in the S5 system at which the length of the byte string that could be accepted by the other station during the last "write byte string" job will be stored. One word is always required.

Dest. type:DB, DX
DB no.:0,1..255
Start:0..2042

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

Output Fields:
After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

Transparent Data Exchange

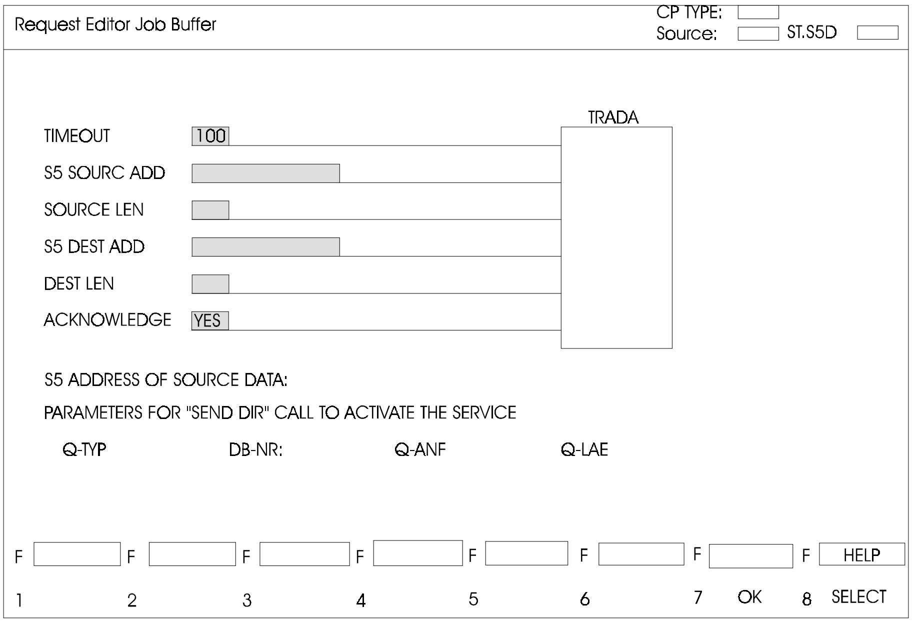


Fig. 6.30: Transparent Data Exchange

Input Fields:

TIMEOUT: Same as for read byte string

For more information about TIMEOUT see page 3-13.

S5 SOURCE ADDRESS Address in the S5 system at which the user program stored the user data to be sent.

Source type:DB, DX, DA
DB no.:0,1..255
Start:0..2042

SOURCE LEN The "SOURCE LEN" parameter specifies how many user data bytes the job contains.

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 function key F7 (OK) the complete S5 address of the byte string is displayed (output field "S5 ADDRESS OF SOURCE DATA").

S5 DESTINATION ADDRESS
Address in the S5 system at which the data in the acknowledgment will be stored by the CP.

Dest. type:DB, DX
DB no.:1..255
Start: 0..2042

DEST LEN:
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 acknowledgment can be accepted.

ACKNOWLEDGE
This input field is used to define the service to be triggered in more detail.
Permitted entries:

YES:
The "TRADA with acknowledgment" service is triggered.

NO:
The "TRADA without acknowledgment" service is triggered.

Output Fields:

After entering the data with F7, the parameters of the last edited job buffer are displayed on the dialog. 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 programming device. Next dialog: Input

6.4.8 Other Jobs

Configure ANZW (local)

With this function, a job buffer for transferring configuration parameters to the local network interface is generated. This job buffer is also transferred to the CP with a SEND-DIR job.

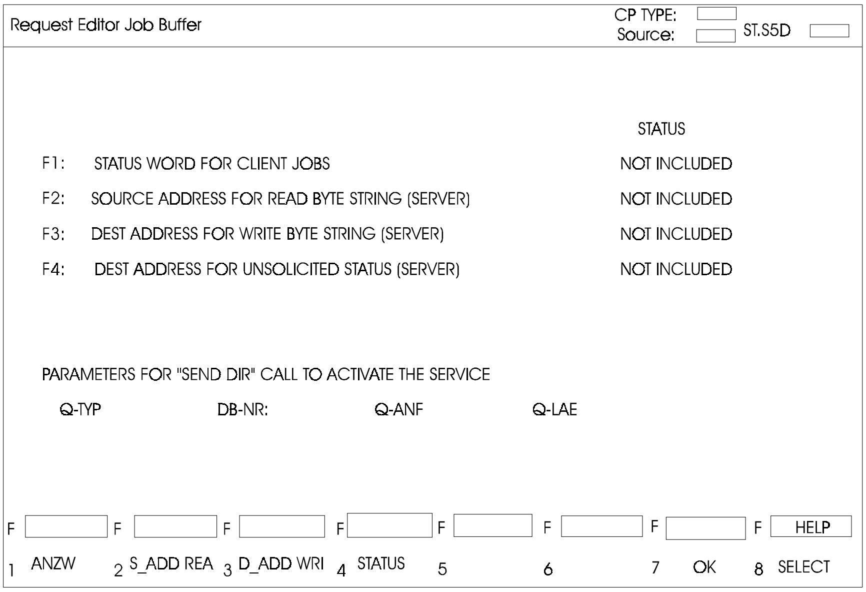


Fig. 6.31: Configure ANZW Dialog

Parameters are entered in the job buffer depending on the selections you make from the four available parameter sets. In the status column, the Request Editor indicates whether or not the parameter is contained in the job buffer.

Function Keys:

F1 ANZW	You can input the address of the status word to be addressed in client jobs on the application association used with the SEND direct. The address transferred with a configuration job has priority over the address configured in the COM 1430application association configuration. Next dialog: see subfunction F1
F2 S_ADD REA	You can input the address in the S5 system at which the data for the "read byte string" job are stored Next dialog : see subfunction F2
F3 D_ADD WRI	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 dialog: 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 dialog: see subfunction F4
F7 OK	Completes the input and stores the newly edited job buffer in the main memory of the programming device. Next dialog: Input

Configure ANZW (local)
Subfunction F1: status word for client jobs

You select the address of the status word to be addressed on the application association for SEND direct in client jobs. The address transferred with a configuration job has priority over the address configured in the COM 1430 application association configuration.

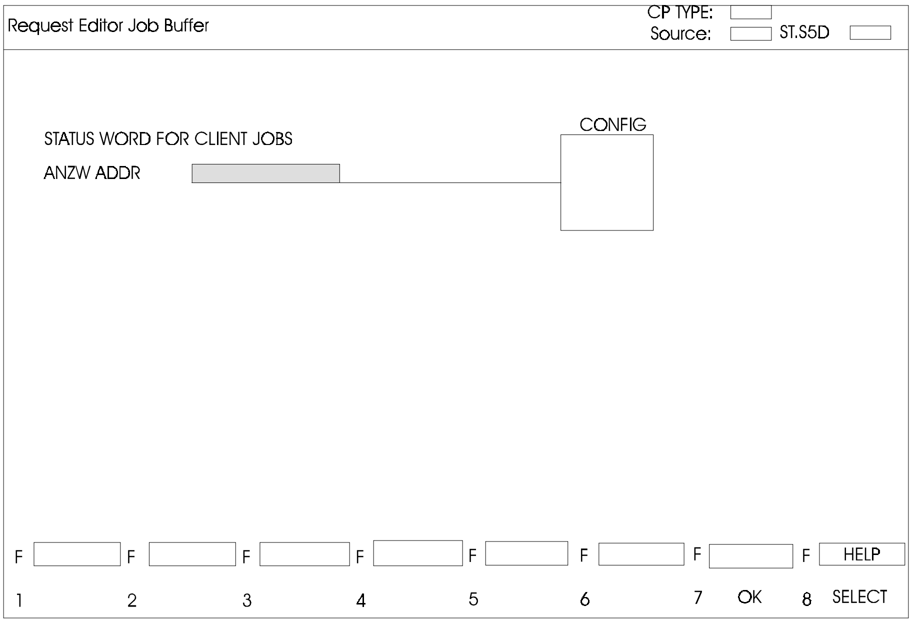


Fig. 6.32: Status Word for Client Jobs Dialog

Input Fields:

ANZW ADDR Address of the status word to be addressed on the application association 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 dialog: configure ANZW (local).

Configure ANZW (local)
Subfunction F2:Source address for read byte string (server):

The function is used to input the address in the S5 system at which the data for the "read byte string indication" are stored You also specify the status word to be used with such jobs.

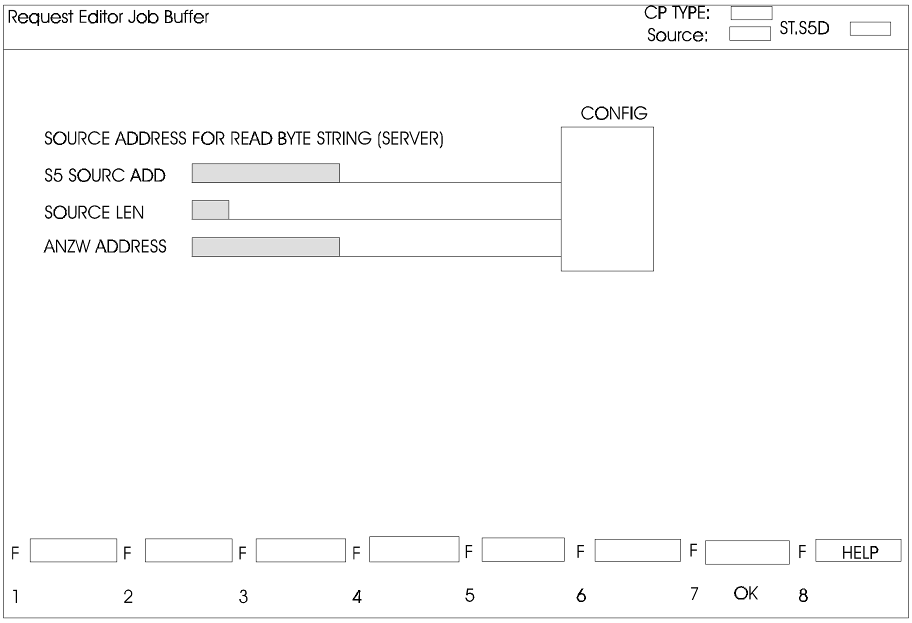


Fig. 6.33: Source Address Read BS Dialog

Input Fields:

S5 SOURCE ADDRESS: Source type:DB, DX
DB no.:1..255
Start:0..2042

SOURCE LEN The "SOURCE LEN" parameter specifies how many data words will be entered in the job acknowledgment by the CP.

ANZW ADDRESS Address of the status word for such jobs. The entry is the same as for the ANZW parameter.

Function Keys:

F7 OK	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 dialog: configure ANZW.

Configure ANZW (local)
Subfunction F3:Destination address for write byte string

The function is used to 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". You also specify the status word to be used with such jobs.

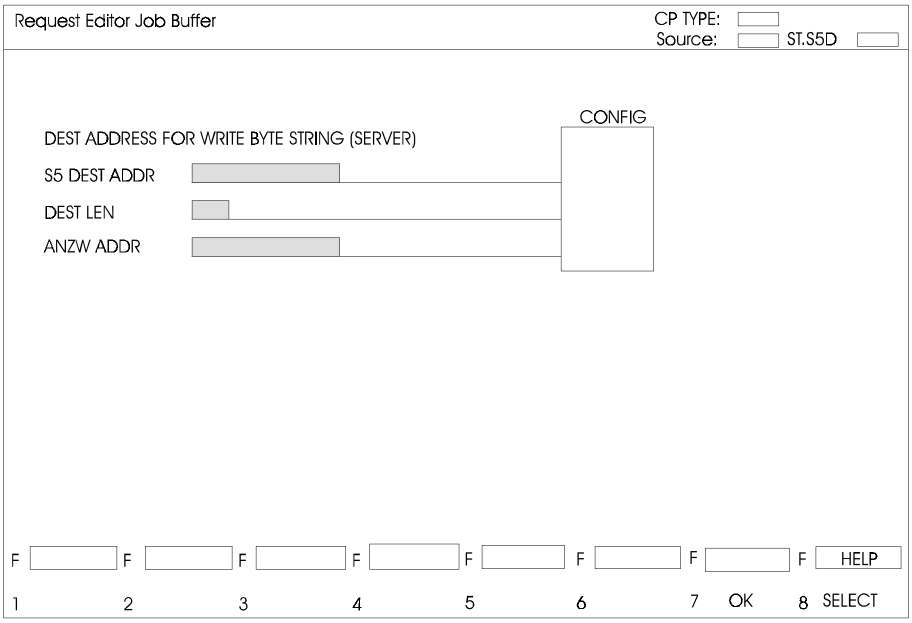


Fig. 6.34: Dest. Address Write BS Dialog

Input Fields:

S5 DESTINATION ADDRESS	Dest. type:DB, DX DB no.:1..255 Start:0..2042
DEST LEN:	The "DEST LEN" parameter specifies the maximum number of words the byte string can contain
ANZW ADDR	Address of the status word for such jobs. The entry is the same as for the ANZW parameter.

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 dialog: 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.

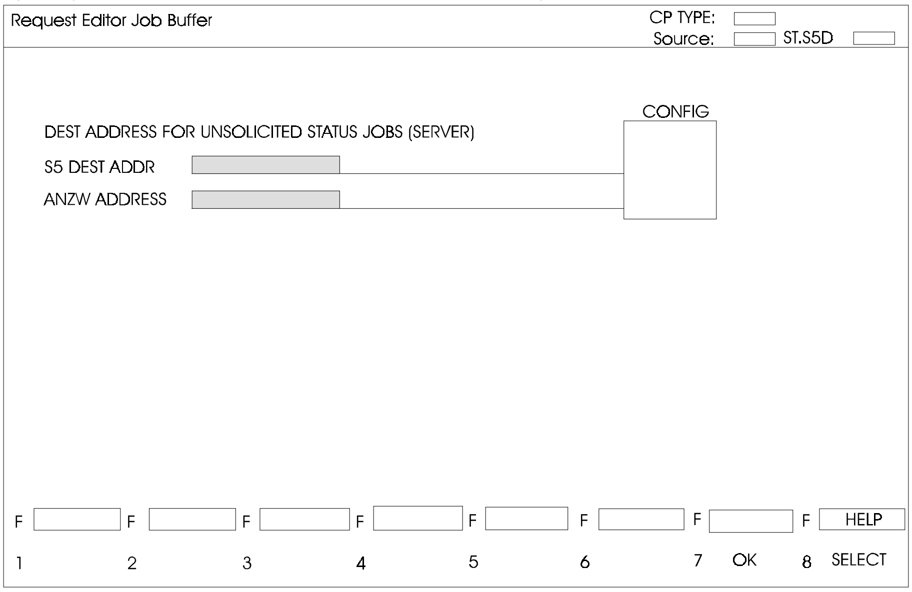


Fig. 6.35: Unsolicited VMD Status

Input Fields:

S5 DESTINATION ADDRESS	Dest. type:DB, DX DB no.:1..255 Start:0..2042
ANZW ADDRESS	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.

6.5 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.

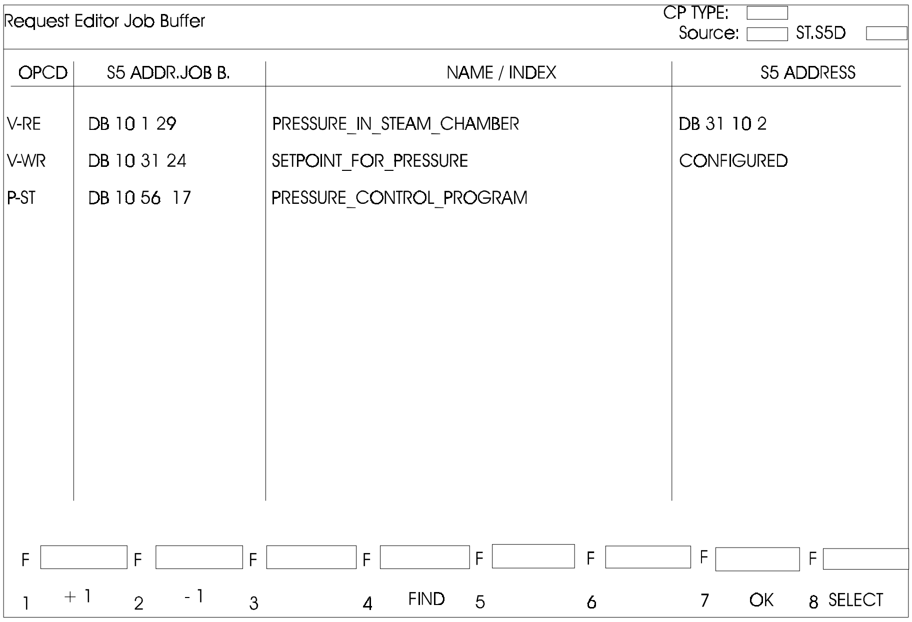


Fig. 6.36: Overview Display

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 (with an additional or context-specific meaning):

F4 FIND	Using the cursor keys (up/down) you can mark a job buffer in the dialog (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.

6.6 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.

6.7 Documenting Job Buffers

Using the documentation functions, you can display or print the configuration data. The output depends on the selections you make in the dialog in which you specify the configuration environment.

The functions always relate to the currently selected program file and selected data block.

6.7.1 Documentation | All

Both the job buffers and the overview list are output.

6.7.2 Documentation | Overview

You obtain a printout of the overview list. The explanation of the individual fields can be found in the function description "Job Buffer Overview".

6.7.3 Documentation | Job Buffers

The job buffers for the selected program file and the selected data block are output in a format based on the representation on the dialog.

❏

III Reference Section

7 TF Variable Services

Topics in this Chapter

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 7.1 Basics with Checklist)

➢ as a source of reference when programming variable service jobs (Section 7.2 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".

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.

7.1 Basics of the Variable Services

7.1.1 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 1430 by configuring with the COM 1430 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 1430 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 1430 TF (TF definitions). If the variable is configured, the four words in the job buffer have the value 0.

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.

7.1.2 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 1430 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 1430 TF tool PG Load. PG Load 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.

c) Application association-specific:
Application association-specific variables are assigned to a particular application association. The variable is only visible via this application association, i.e. it is only possible to access the variable via this application association. 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 application association-specific assignment.

Server configuration:
Application association-specific variables are configured using the subfunction local variables during application association configuration with COM 1430 TF.

Client configuration:
If the variable is more complex (data description > 4 words) a variable configuration is necessary on the client. The variable is configured for the application association as a remote variable.

Fig.7.1 provides an example of a complete virtual device, two domains, a program invocation and assigned variables.

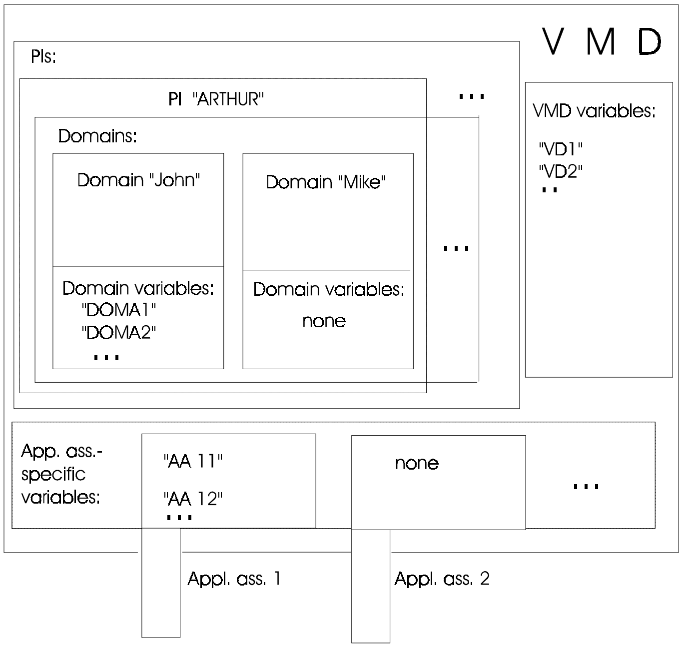


Fig. 7.1: Example of VMD Structure

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

The variables "AA 11" and "AA 12" have an application association-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 application association.

Figure 7.2 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
Application association	Variable name	Exactly one application association

Fig. 7.2: 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.

7.1.3 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 1430.

➢ 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 1430.

➢ Integrating the HDBs (SEND ALL and RECEIVE ALL) and the status evaluation in the PLC program.

7.2 Service Description

7.2.1 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.

"Read" Job Buffer

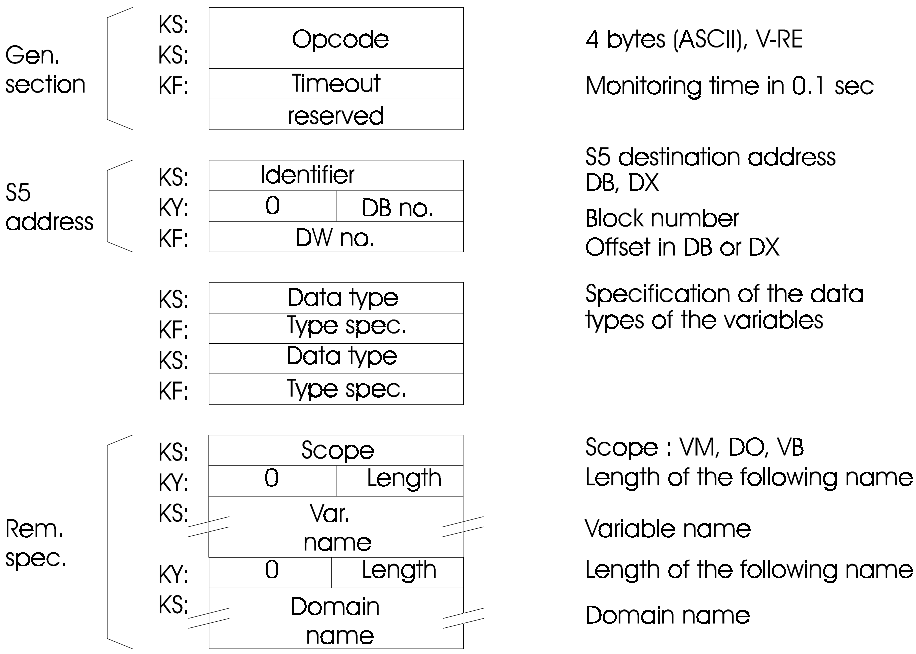


Fig. 7.3: Structure of the Job Buffer for TF Variable Service "Read"

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 acknowledgment for the service (i.e. the maximum dwell time of the job in the CP 1430). This is specified in multiples of 0.1 sec.

For more information about timeout, see page 3 - 13.

S5 Address:
Description of the local destination address for the service.

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

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

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

DW no. 1 word, format: KF
Possible 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 CP 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 configure it completely in COM 1430 TF.

Data type:	1 word, format:	KS
	Permitted values:	see Table 7 -3, page 7 -41 .
Data format:	1 word, format:	KF
	Permitted values:	see Table 7 -3, page 7 -41 .

Note on configured variables:
Enter the following for configured variables:
DB no. = 0
DW no. = 0
Specification of the data type = irrelevant

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
	Possible values:	VM, DO, AA

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.

	AA: Application association-specific The specified variable is only valid for the application association identified by the interface number/job number.	
Length:	1 word, format:	KY
	Possible values:	
	High byte:	0
	Low byte:	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
	Possible 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)

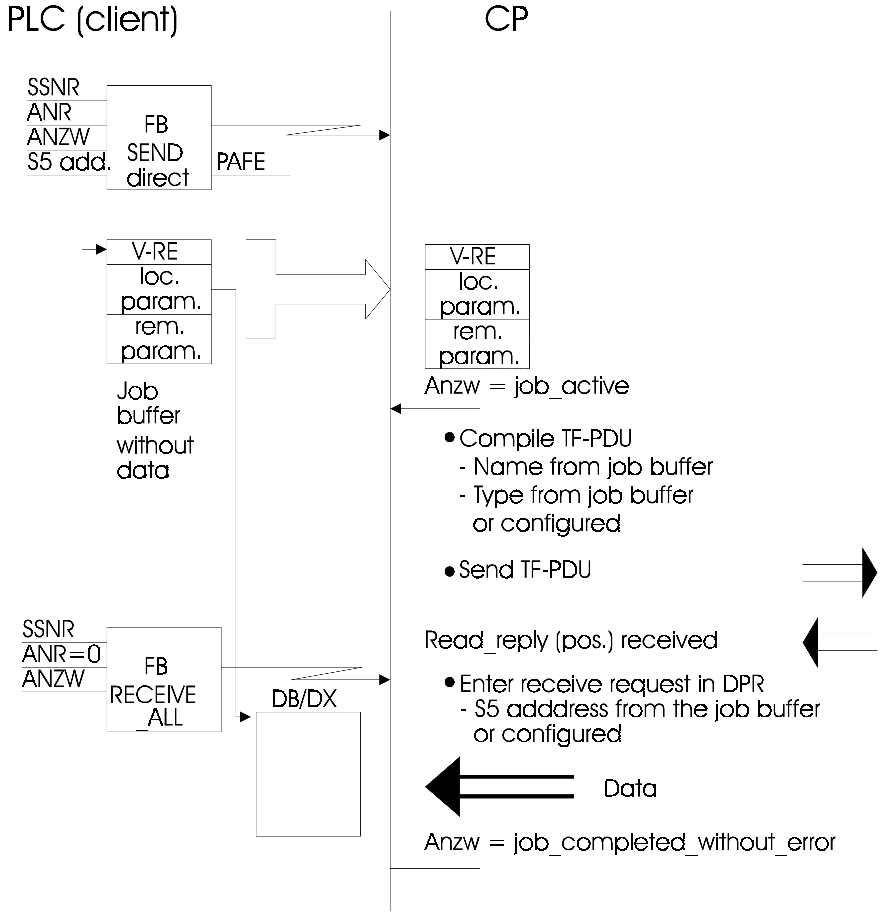


Fig. 7.4: Sequence Description (Read, Positive Acknowledgment)

Sequence Description (Read, Negative Acknowledgment)

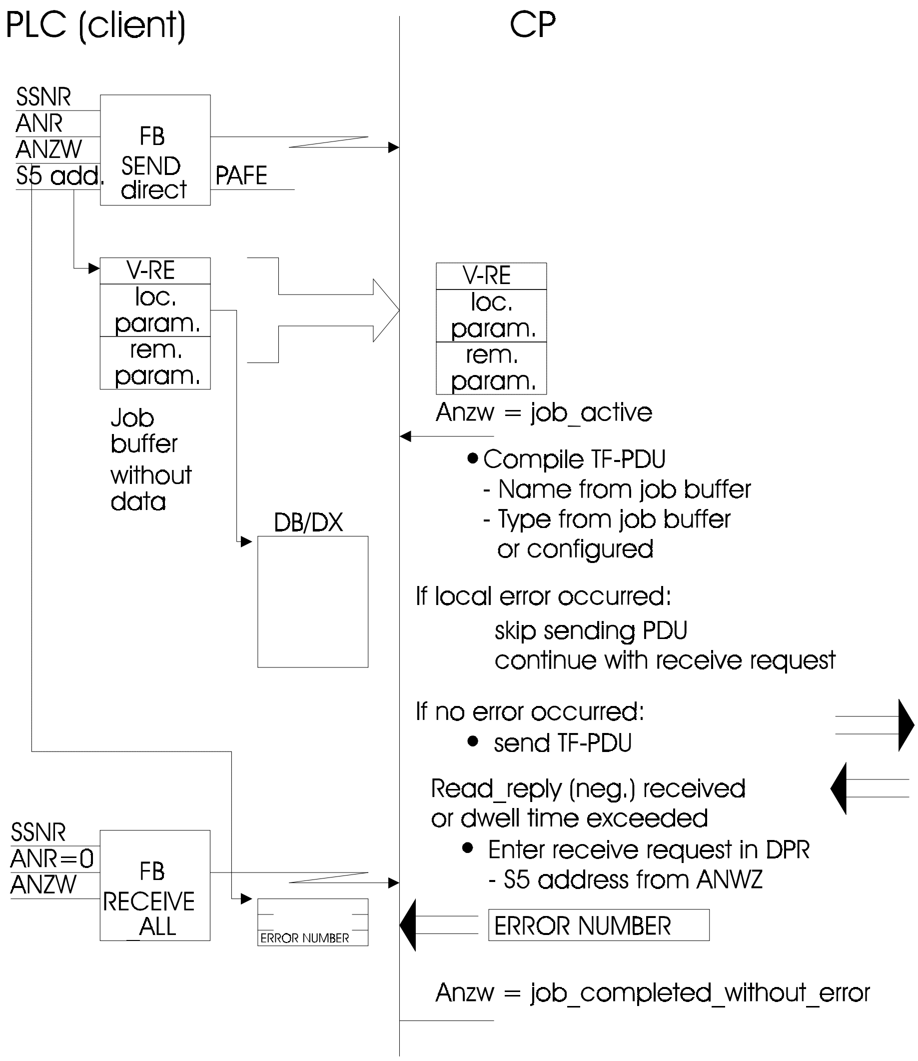


Fig. 7.5: Sequence Description (Read, Negative Acknowledgment)

7.2.2 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 1430 generates a negative acknowledgment.

7.2.3 Write Variable (Client)

Meaning

With the "write variable" service, 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 1430 as a remote variable.

"Write" Job Buffer

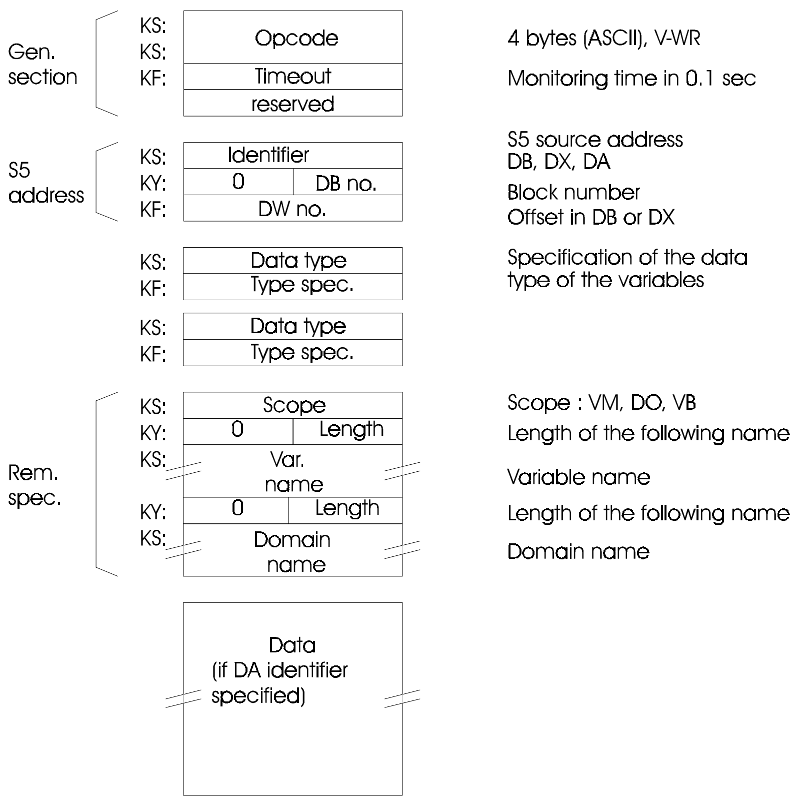


Fig. 7.6: Job Buffer for TF Variable Service "Write"

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 acknowledgment 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 - 13.

S5 Address:
Description of the local source address or indicates that the data are in the buffer.

Identifier: 1 word, format: KS
Possible 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
Possible values: high byte: 0, low byte: 0-255
Meaning: DB or DX number

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

DW no.:	1 word, format:	KF
	Possible 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 1430 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, you must configure this as a remote variable.

Data type:	1 word, format:	KS
	Permitted values:	see Table 7 -3, page 7 -41,
Type specification:	1 word, format:	KF
	Permitted values:	see Table 7 -3, page 7 -41,

Note on configured variables:
Enter the following for configured variables:
DB no. = 0
DW no. = 0
Specification of the data type = irrelevant

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.

Scope:	1 word, format:	KS
	Possible values:	VM, DO, AA

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.

AA: Application association-specific
The specified variable is only valid for the application association identified by the interface number/job number.

Length:	1 word, format:	KY
	Possible values:	
	High byte:	0
	Low byte:	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
	Possible values:	
	High byte:	0
	Low byte:	1 to 32

Specifies the number of following valid bytes (length of the domain name; only when scope = DO).

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 1430 expects the current values of the data to be transmitted according to the data type description.

Sequence Description (Write, Positive Acknowledgment)

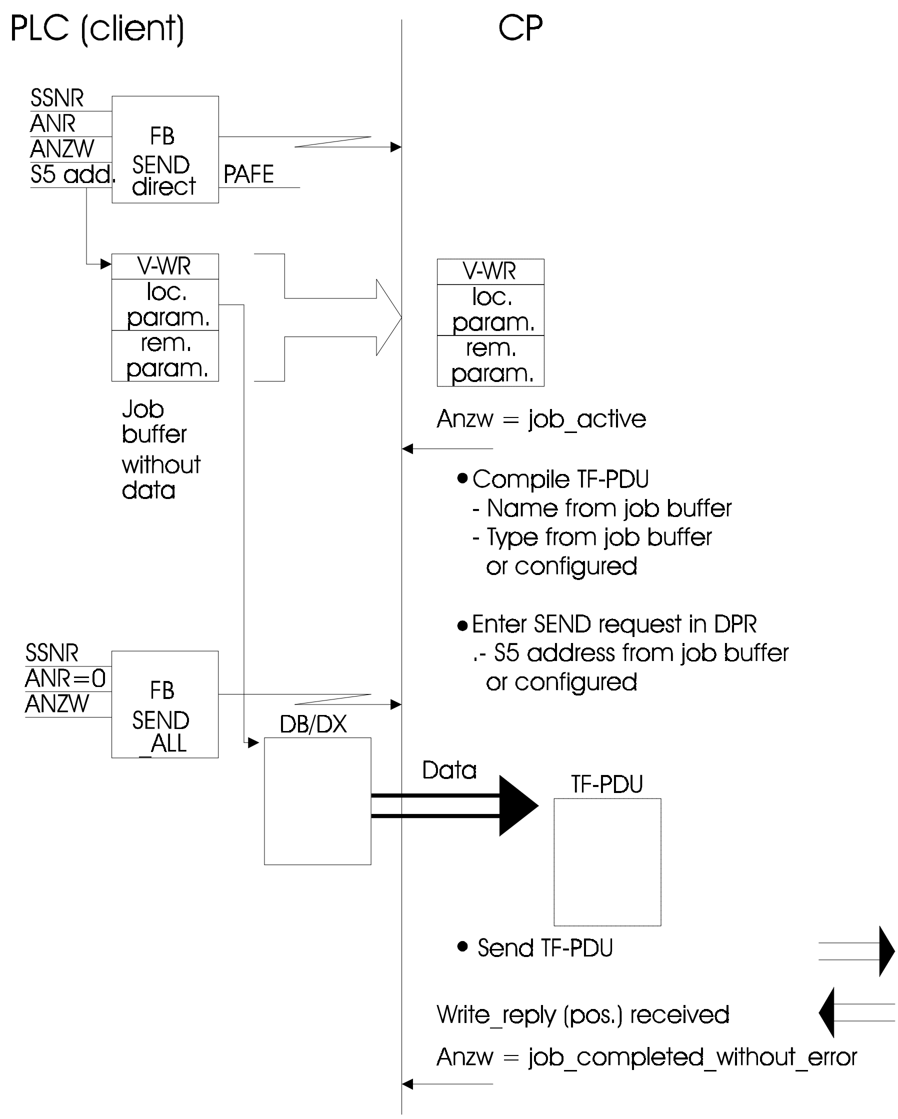


Fig. 7.7: Sequence Description Write with Source ID "DB" or "DX"

Sequence Description (Write, Positive Acknowledgment)

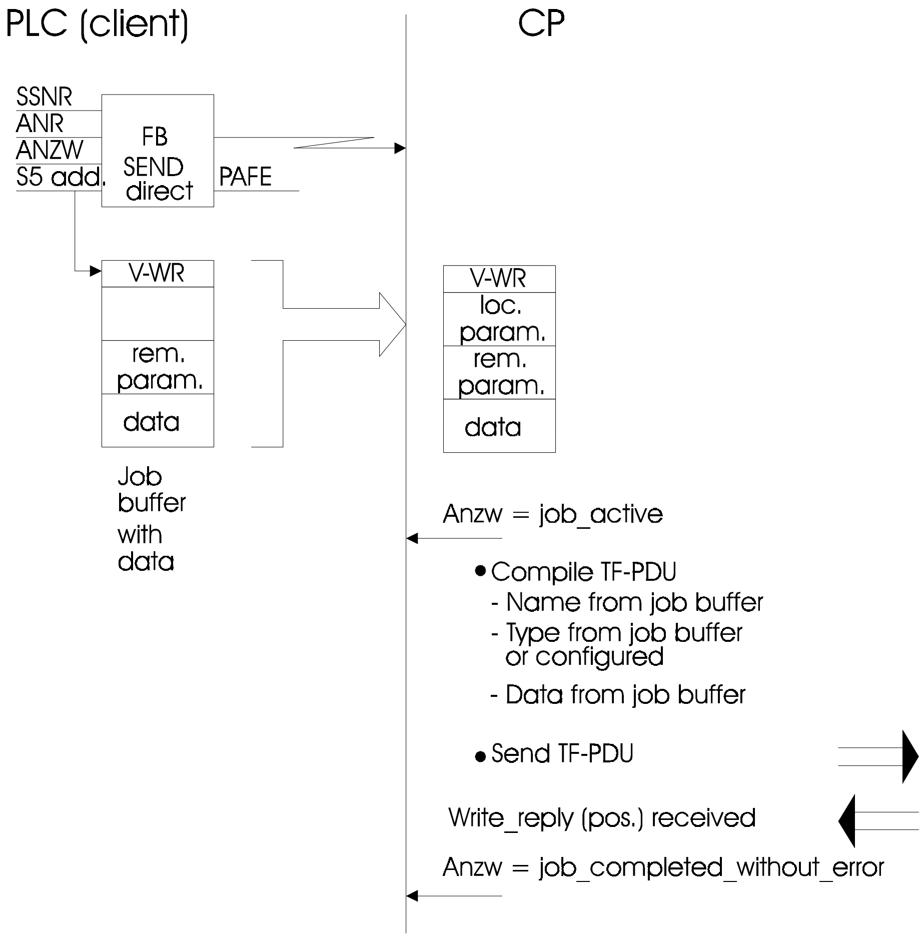


Fig. 7.8: Sequence Description Write with Source ID "DA"

Sequence Description (Write, Negative Acknowledgment)

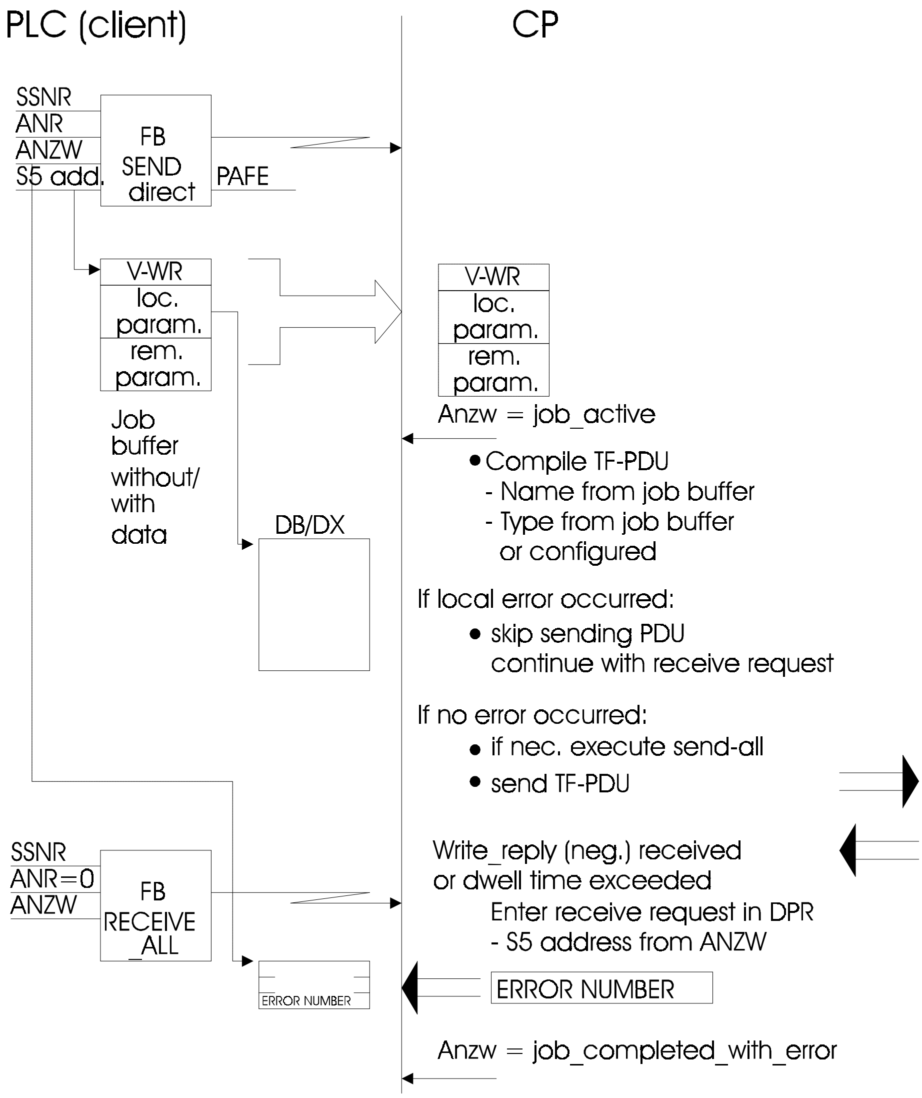


Fig. 7.9: Sequence Description (Write, Negative Acknowledgment)

7.2.4 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 1430 generates a negative acknowledgment.

7.2.5 Information Report (Client)

Meaning

With the information report service, an application can send a variable unsolicited to a another application. The data of the variable 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

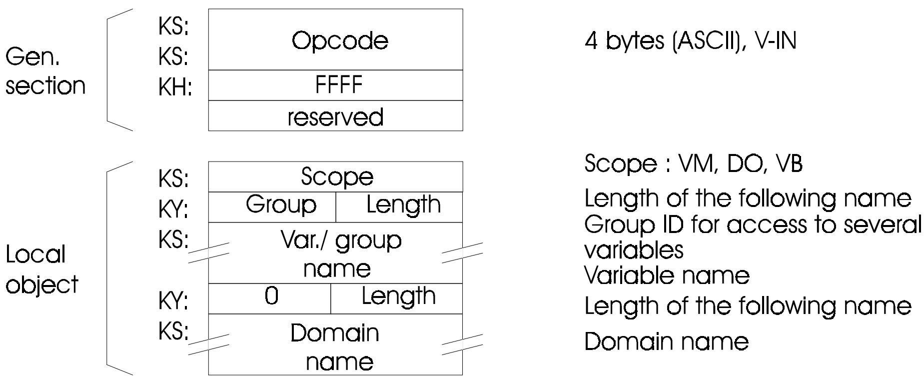


Fig. 7.10: Structure of the Job Buffer for TF Variable Service "Information Report"

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
	Possible values:	VM, DO, AA

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.

AA: Application association-specific
The specified variable is only valid for the application association identified by the interface number/job number.

Group ID/ Length:	1 word, format:	KY

High byte = group ID
Possible values: 0 to 1
Meaning:

If you select group ID = 0, only one single variable is involved, otherwise several variables are reported with one call. The following name is then not interpreted as a variable name but as a group name. You group variables under a "group name" using COM 1430 TF. In this case, the domain name in the job buffer does not exist.

	Low byte = length Possible values: 1 to 32 Specifies the number of following valid bytes (length of the variable name or group name) Group names are restricted to a maximum of 8 characters.
Variable name: /group name:	n bytes, format: KS For the “Information Report” service, with which a local object (or its value) is transferred to another station, the local object is described by its name. If the length of the variable name is odd, the last byte has no significance.
Length:	1 word, format: KY Possible 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.

Sequence Description (Information Report, Positive Acknowledgment)

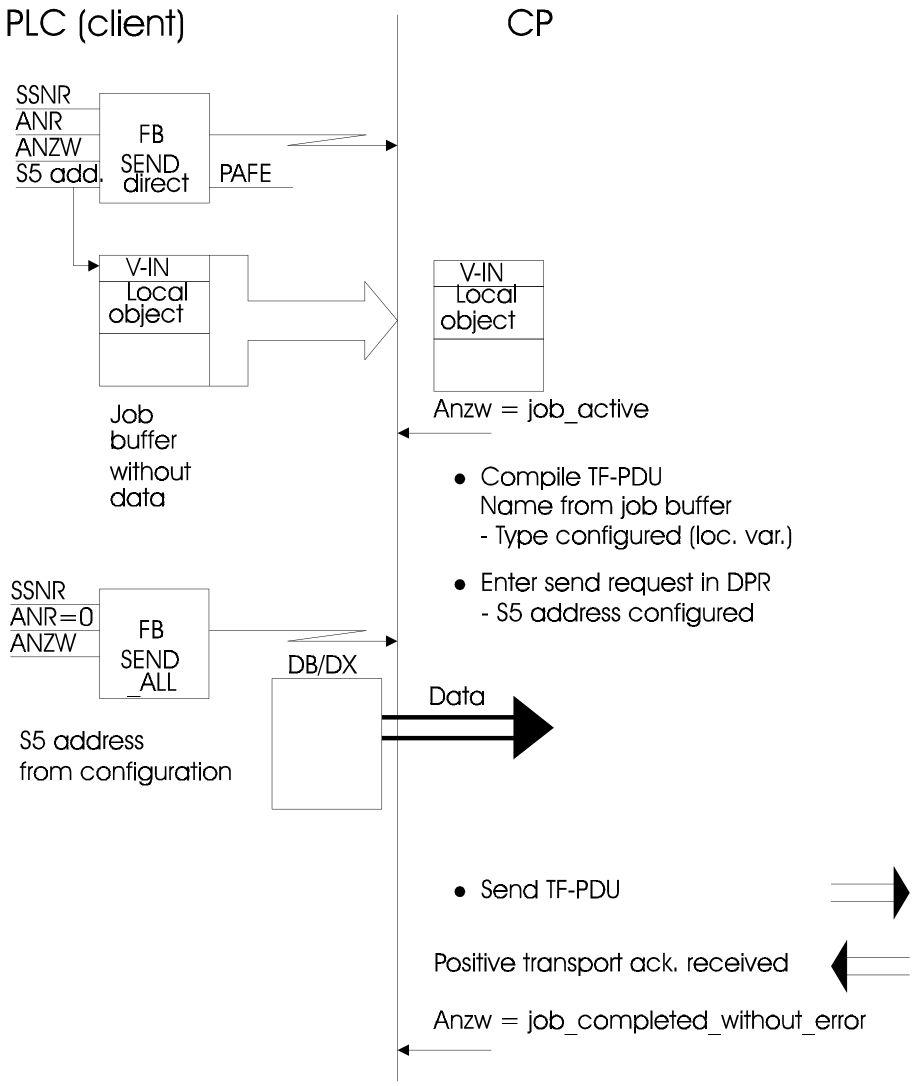


Fig. 7.11: Sequence Description Information Report (Positive Acknowledgment)

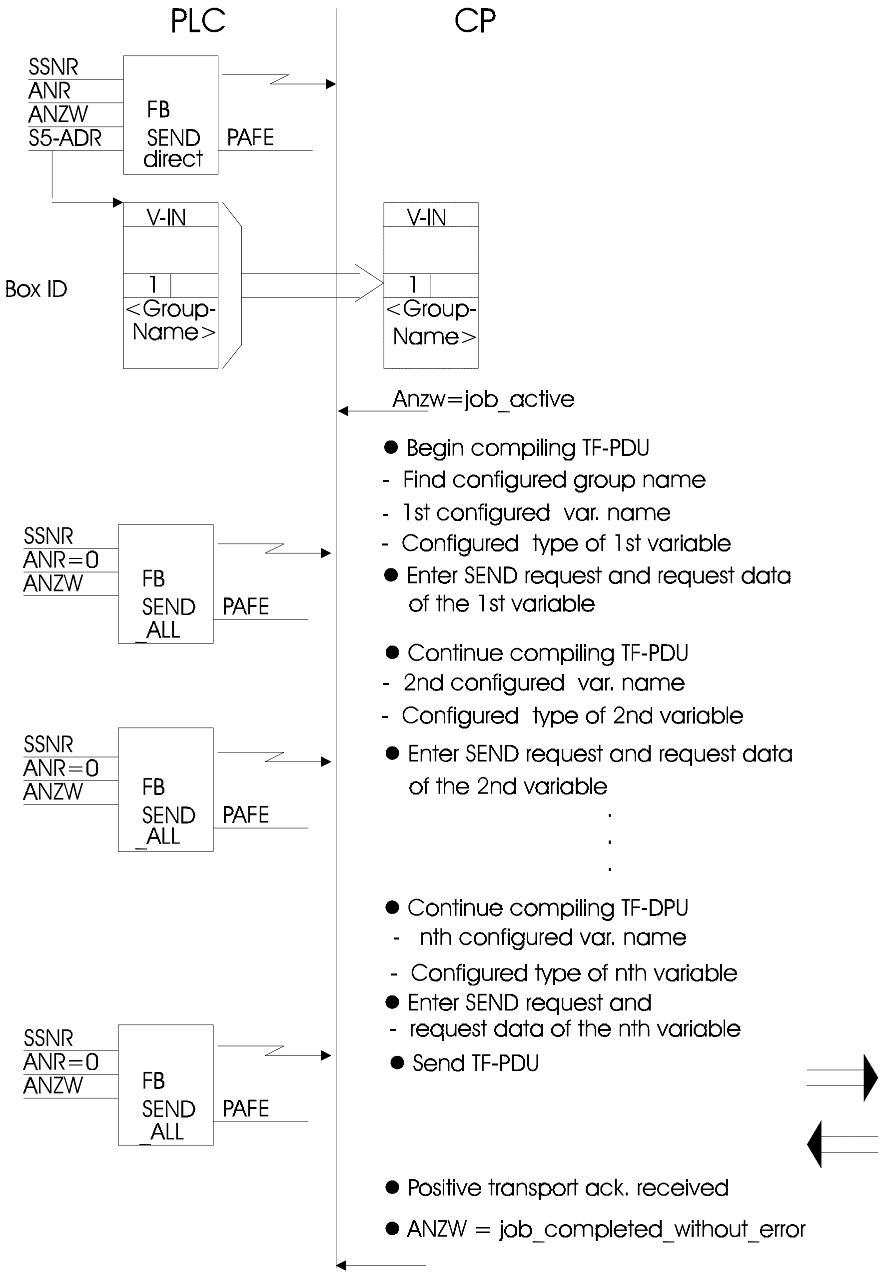


Fig. 7.12: Reporting Several Variables in One Call

Sequence Description (Information Report, Neg. Acknowledgment)

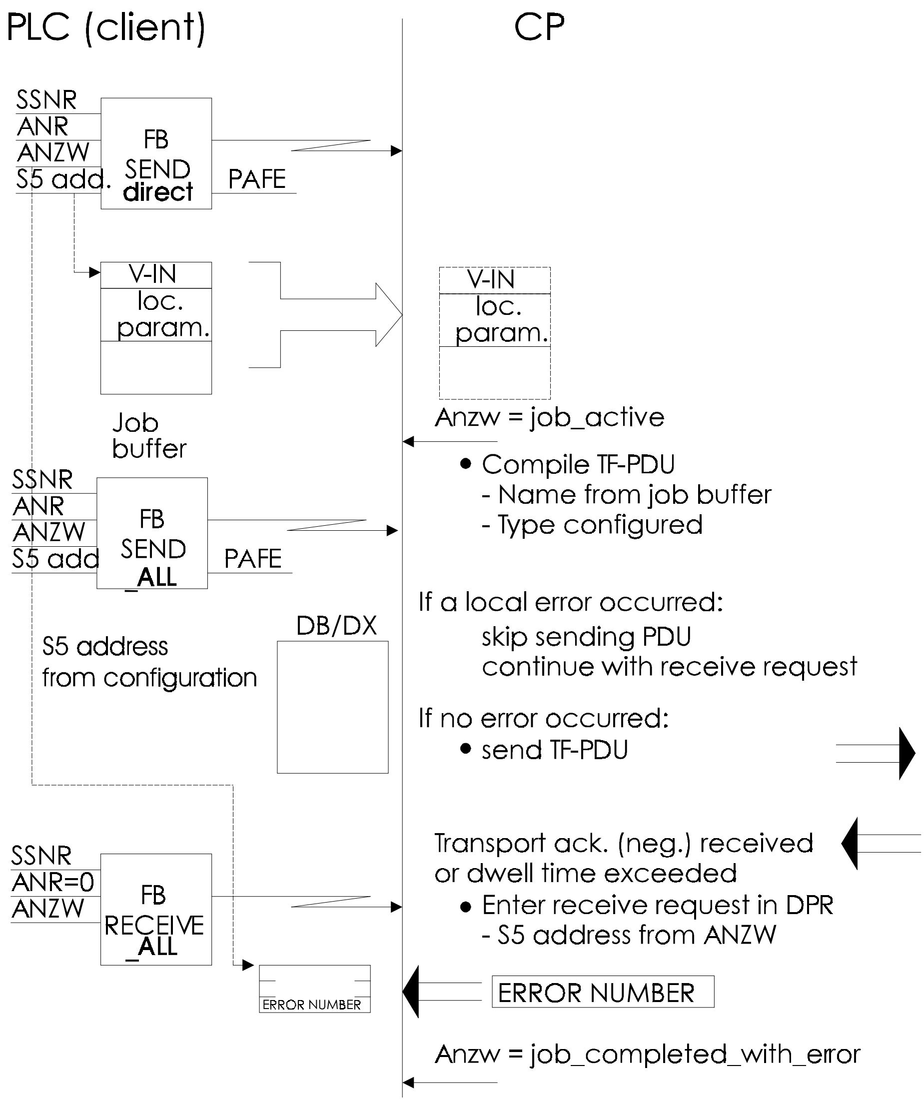


Fig. 7.13: Sequence Description Report (Negative Acknowledgment)

7.2.6 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.

7.3 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.

7.3.1 Client Interface

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.

Write job buffer

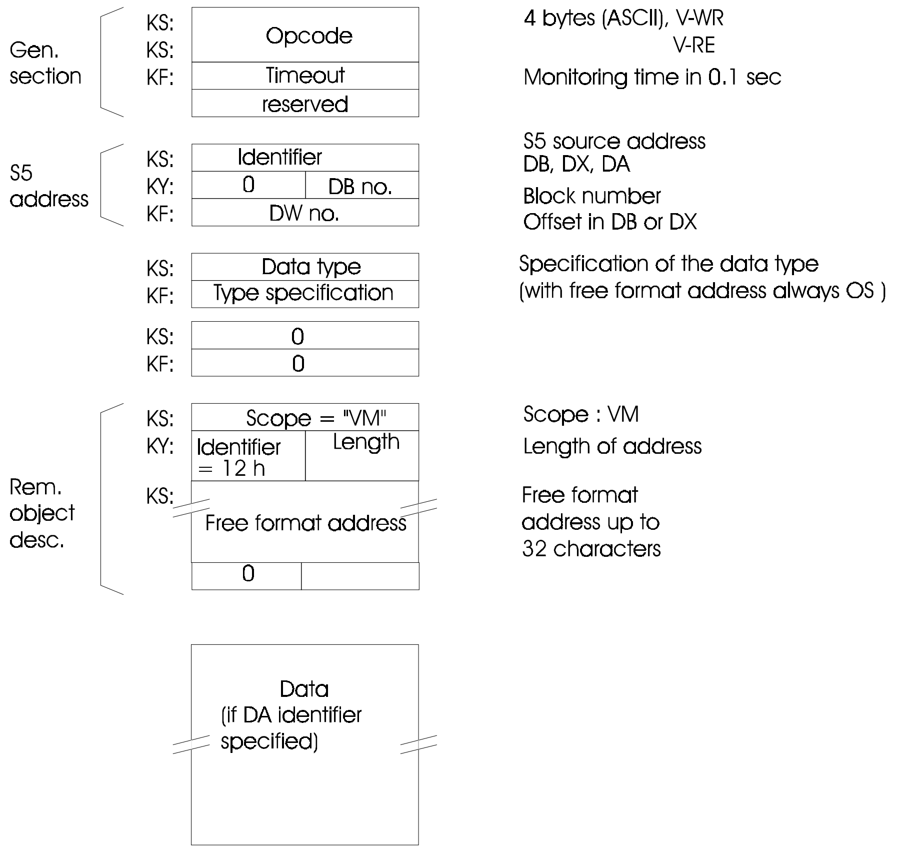


Fig. 7.14: Address Buffer for Variable Services Read and Write with Free Format Address

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 acknowledgment 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 - 13.

S5 address:
Description of the local source address or indicates that the data is in the job buffer (for V-WR) or description of the local destination address (for V-RE).

Identifier: 1 word, format: KS
Possible 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 1430. 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
Possible values: high byte: 0, low byte: 1-255
Meaning: DB or DX number

☞ **Only data blocks not reserved as system DBs by the CPU (such as DB1) can be used as the S5 address.**

DW no.:	1 word, format:	KF
	Possible 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, in other words a maximum of 2043 words.

Data type:	1 word, format:	KS
	Permitted value:	"OS"
Type specification:	1 word, format:	KF
	Possible values:	1 to 4086
	Length of the octet string in words.	

Remote Object Description:
The free format address can be freely defined by the user. For communication between an S5 PLC and an S5 PLC, select a structure as described on page 7-38.

Scope: The scope is always VM.

Identifier/ length: 1 word, format: KY

High byte = identifier
Identifier 12_H indicates the job read/write variable with free format addressing.

Low byte = length
Possible 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 Fig. 7.15)
length = 11 for type 1 (see Fig. 7.16)

Free format address: n bytes, format: KH
A free format, that can be freely defined by the user, depends on the destination device. If the length of the free format address is odd, the last byte has no significance.

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" with object addressing.

Free Format Address:
The structure of the free format address is represented in the diagrams for communication between SIEMENS programmable controllers. The free format 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 1430.

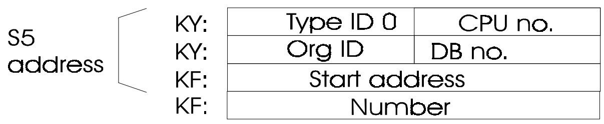


Fig. 7.15: S5-Specific Free Format Address in Job Buffer without Status Word

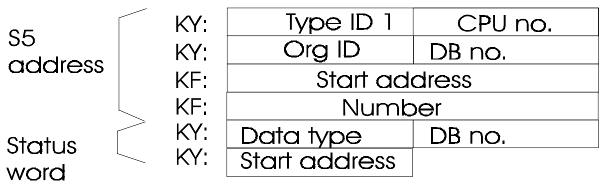


Fig. 7.16: S5-Specific Free Format Address in Job Buffer with Status Word

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

Type ID:	1 byte, format:	KY (high byte)
	Possible values:	0/1
	Identifier to indicated whether or not the S5 address is used with ANZW.	
CPU No.:	1 byte, format:	KY (low byte)
	Possible values:	1 to 4
	Meaning: specifies the CPU no. in which the variable will be written.	
Org identifier:	1 byte, format:	KY (high byte)
	Possible 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)
	Possible 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
	Possible values:	0 to 2047
	Meaning: start address from which the variable will be stored in the data block.	
Number:	1 word, format:	KF
	Possible values:	1 to 2048
	Meaning: number of data values to be transferred.	

Status word:

Type:	1 byte, format:	KY (high byte)
	Possible values:	01,02,03,255
	Meaning: type of status word (see Table 7.5).	

DB no.:	1 byte, format:	KY (low byte)
	Possible values:	0 to 255
	Low byte:	DB or DX number
Start address:	1 byte, format:	KY (high byte)
	Possible values:	0 to 255 (DW), 1 to 255 (flag area)
	Area for storing the status word	

Status word	Type	Values
Type	Flag area DW in DB area DW in DX area No status word	01_H 02_H 03_H FF_H
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

Table 7.2: Status Word

7.3.2 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.

7.4 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 1430 TF configuring tool.

Daten type	Data format	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 7.3: TF Type and Meaning

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

1. **BO:** Boolean
   Boolean variables are modeled 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 modeled by the CP on a data word in a data block (format KF). The TF-data type integer 32 is modeled 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 is modeled by the CP on a data word in a data block (format KH). The TF data type unsigned 32 is modeled by the CP on two data words in a data block (format KH).

4. **FP:** Floating point number
   Floating point numbers are modeled 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!):

Bit 15 … **Bit 0**

s*10	s*1	factor (*10 ms)	
h*10	h*1	min*10	min*1
day*10	day*1	weekday	0
year*10	year*1	month*10	month*1

Time of day (4 bytes): first two rows

Time of day with date (8 bytes): all four rows

Permitted values:		Weekday	
seconds units:	0 to 9	0	Monday
seconds tens	0 to 5	1	Tuesday
minutes units:	0 to 9	2	Wednesday
minutes tens	0 to 5	3	Thursday
hours units :	0 to 9	4	Friday
hours tens	0 to 1/0 to 2 (see note)	5	Saturday
days units	0 to 9	6	Sunday
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*X+Y.

Example:

$F1_H$	becomes	15*10 + 1	= 151
AB_H	becomes	10*10 + 11	= 111
FF_H	becomes	15*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.

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 modeled 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, it is modeled 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
DWn	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 modeled 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 complying with ISO 646 (format KS).

☞ **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.❑

8 TF Domain and PI Services Implementing a CIM Network

Topics in this Chapter

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.

8.1 Domain Services

Definition

A domain in the SIMATIC S5 programmable logic controller is always part of a (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 1430 tool PG Load.

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 PG Load tool is described in Chapter 5 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 the 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).

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.

➢ This static domain can be used as follows:

1. Archiving
   The first time the "SIMATIC_S5" domain is uploaded, the complete PLC program is saved, in other words, all the blocks are grouped together in this domain and uploaded. This domain can then be handled on the PG as a loadable domain. If a user domain exists, "SIMATIC_S5" is a dummy domain without data or program.

2. Using PI services **without** domain services
   You can use the dummy domain “SIMATIC_S5” to create a user PI, in other words you do **not** have to load a domain to use the PI services.

In a SIMATIC S5 PLC, up to eight domains can be loaded in addition to the static SIMATIC S5 domain.

☞ **Note that PI and domain services are not permitted with SSNR 232, 236 and 244.**

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 PG Load tool.

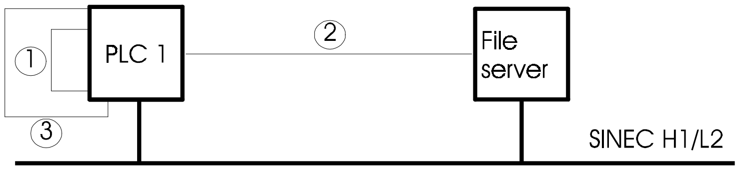
➢ Grouping the required blocks to loadable domains using the PG Load tool.

➢ Clarifying the system configuration in which the loadable domains will be managed. Will there be a third-party association (host - PLC - file server)?

➢ Configuring application associations for transmitting the domain services. If third-party associations are to be used, file server 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.



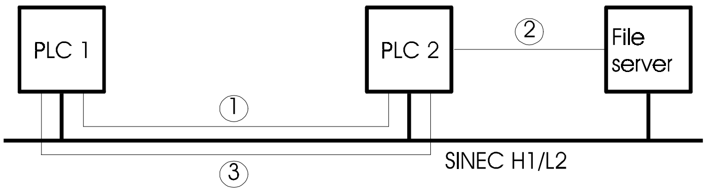
① "Load domain content" request

② Loading sequence

③ "Load domain content" acknowledgment

Fig. 8.1: Loading the Local PLC

The host computer can, however, also request the second station to load itself from a third station (e.g. a file server) 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 file server. This situation is illustrated by the following figure



① "Load domain content" request

② Loading sequence

③ "Load domain content" acknowledgment

Fig. 8.2 Loading a Further PLC (third-party association)

Third-party associations are supported by the CP.

This results in the following sequence between the devices:

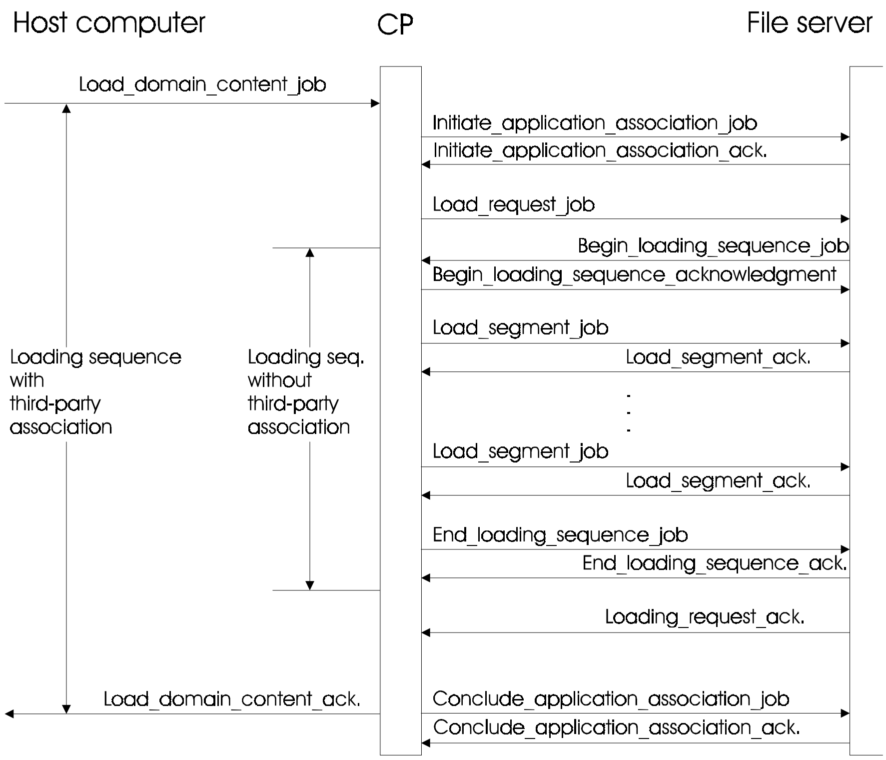


Fig. 8.3 Load Sequence with/without Third-Party Association

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

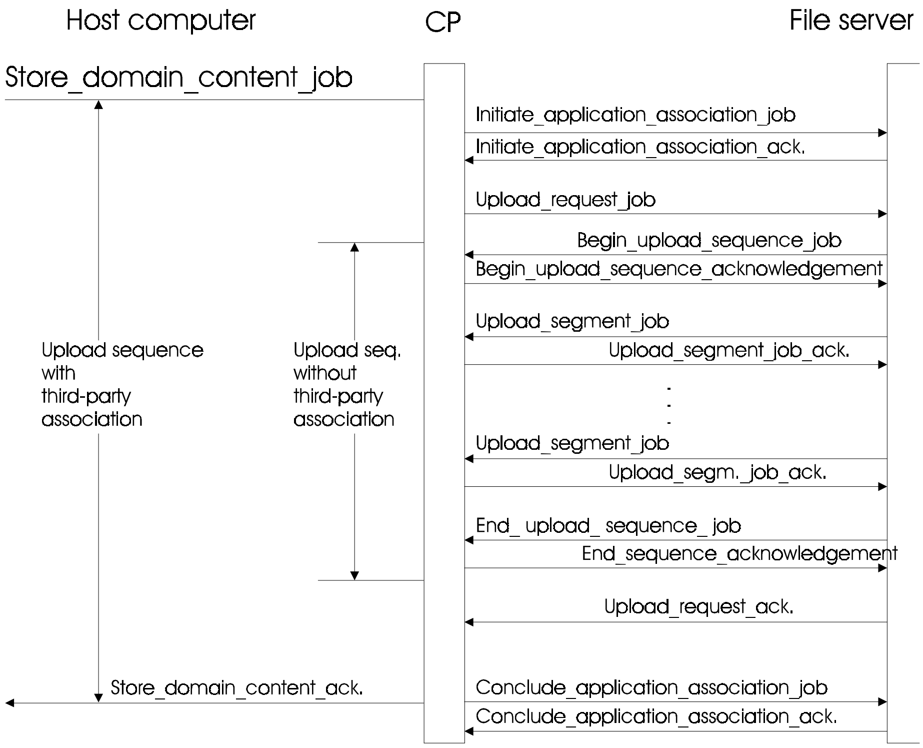


Fig. 8.4 Upload Sequence with/without Third-Party Association

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

8.1.1 Load Domain Content

Using the "load domain content" service, a PLC program can load a domain from a file server (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.

"Load Domain Content" Job Buffer

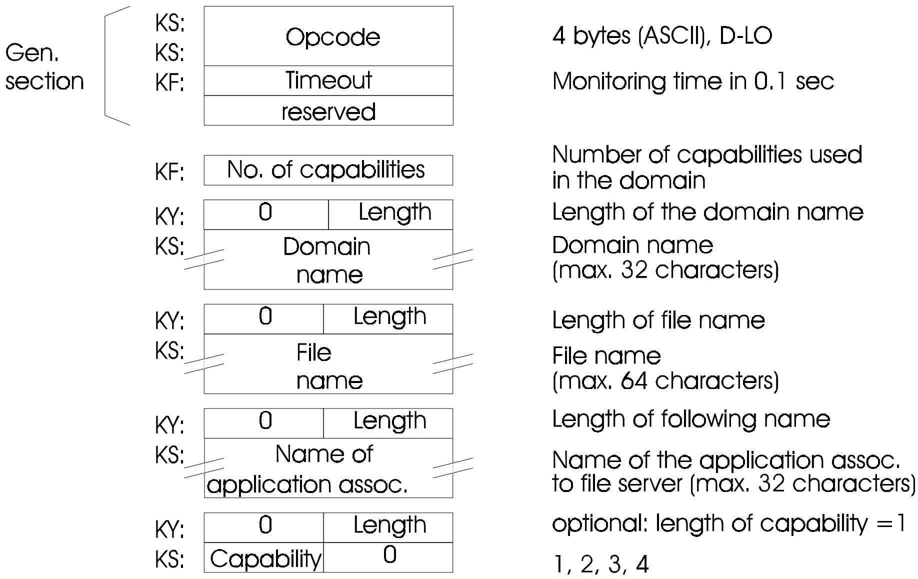


Fig. 8.5 Structure of the Job Buffer for TF Service "Load Domain Content"

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 acknowledgment 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 - 13.

Job-related section

Number of capabilities:

Format:	KF	
Possible values:	0...4, 1	
Default:	0	

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	
Possible 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, the last byte has no significance.

Length of file name.	Format:	KY
	Possible 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 file server containing the domain. If the length of the file name is odd, a "padding"byte must be added.	
Length of name of appl. ass.	Format:	KY
	Possible values:	high byte: 0 low byte: 1...32
	Meaning: length of the application association to the file server.	
Name of appl. ass.	Format:	KS
	Meaning: name of the application association via which the file server can be obtained by the destination station (third-party association). If the length of the file name is odd, a "padding"byte must be added.	
Length of capability:	Format:	KY
	Possible values:	high byte: 0 low byte: 1...?
	Meaning:length of the first capability. For SIMATIC S5: low byte =1 The first capability describes the CPU belonging to the domain.	
Capability:	Format:	KS
	Meaning: specifies the capability For SIMATIC S5: specifies the CPU number (1, 2, 3, 4) in which the domain is to be loaded.	

Note:
The job buffer 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 application

associations to other systems, the capabilities list and the number can be increased. The job buffer must not, however, exceed a length of 256 bytes.

Sequence description "load-domain-content"

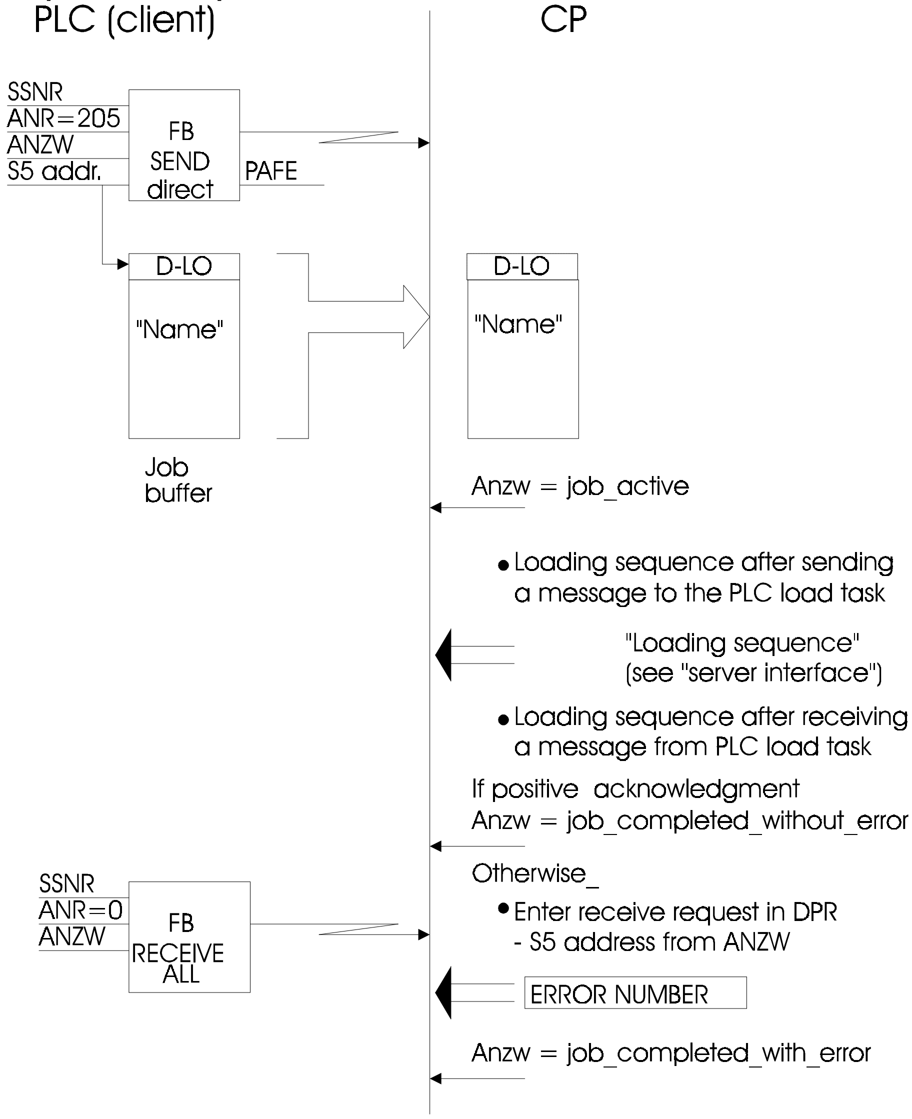


Fig. 8.6 Sequence "Load-Domain-Content" (local PLC)

Sequence description "load domain content"

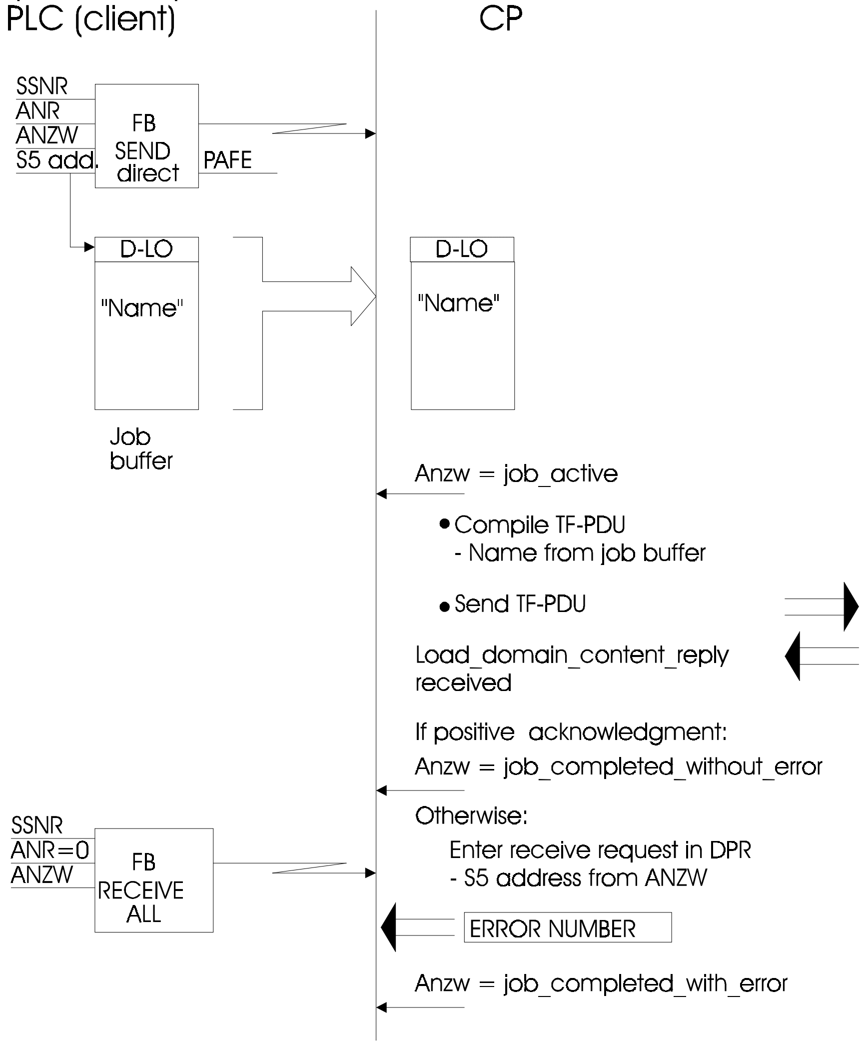


Fig. 8.7 Sequence "Load-Domain Content" (second PLC)

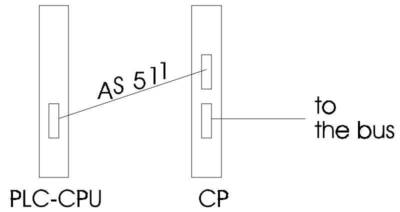
Points to Note with Load Domain Content:

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:

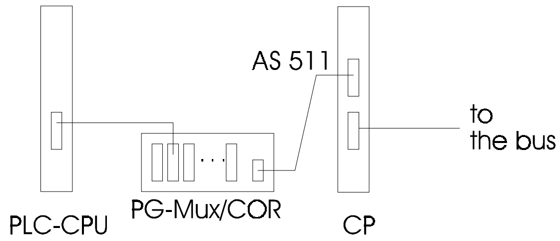
➢ Direct, i.e. via the AS 511 connection ("swing cable") to the PLC

Configuration:



➢ Route via PG Mux/COR

Configuration:



In the multiprocessor mode, the VMD configuration must be defined using the COM 1430 function Edit | VMD Variables Editor.

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 file server application association

M2-4-5 The application association to the file server specified by the "application association name" parameter is set up by the CP (see Section 9.1.4 Special connections). This is not defined in the same way as a normal application association, but in COM 1430 in the "file server application associations" dialog.

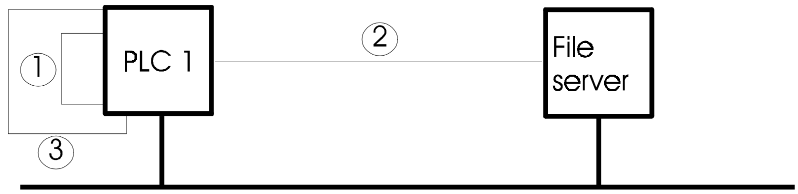
After the loading sequence is complete, the file server application association is terminated again and the load job acknowledged.

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

The loading of a domain is initiated by compressing the PLC on the AS511 interface to the CPU.

8.1.2 Store Domain Content

Using the "store domain content" service, the content of a (loaded) domain can be archived on a file server. The domain can either exist locally (Figure 8.8) or in a remote PLC (Figure 8.9).

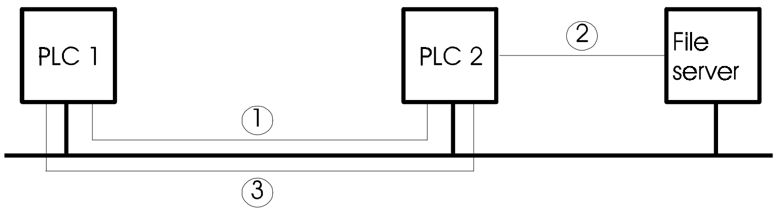


1 "Store domain content" request

2 Storing sequence

3 "Store domain content" acknowledgment

Fig. 8.8 Archiving a Domain of the Local PLC



1 "Store domain content" request

2 Storing sequence

3 "Store domain content" acknowledgment

Fig. 8.9 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).

You can archive the entire PLC content using the SIMATIC_S5 standard domain.

"Store Domain Content" Job Buffer

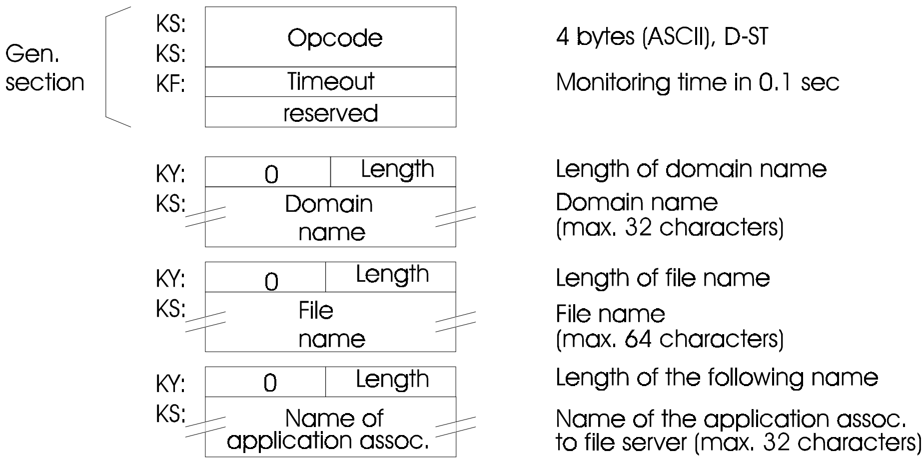


Fig. 8.10 Structure of the Job Buffer for TF Service "Store Domain Content"

Call Description:

General section:

Opcode	D-ST
Timeout:	1 word, format: KF Specifies the maximum length of time the user program will wait for an acknowledgment 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 - 13.

Job-related section

Length of dom. Name:	Format: Possible values:	KY high byte: 0 low byte: 1..32
	Meaning: length of the following domain name	
Domain name:	Format:	KS
	Meaning: Name of the domain to be archived, if the length of the domain name is odd, the last byte has no significace.	
Length of file name.	Format: Possible values:	KY high byte: 0, low byte: 1...64
	Meaning: length of the following file name (including path)	
File name:	Format:	KS
	Meaning: name of the file on the file server that will contain the domain	
Length of app. ass. name	Format: Possible values:	KY high byte: 0 low byte: 1...32
	Meaning: length of the name of the application association to the file server	

Name of appl. ass.: Format: KS
Meaning:name of the appl. ass. via which the file server can be obtained by the dest. station (third-party association).

Sequence Description "Store Domain Content"

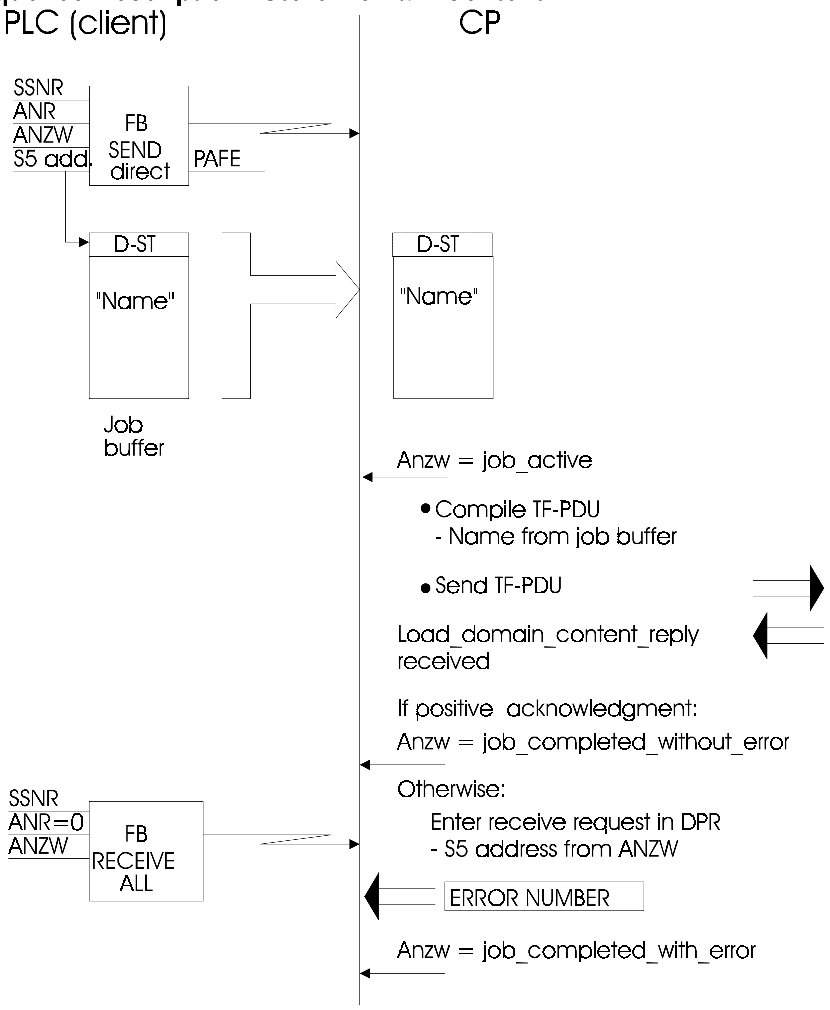


Fig. 8.11 Sequence 'Store Domain Content (second PLC)

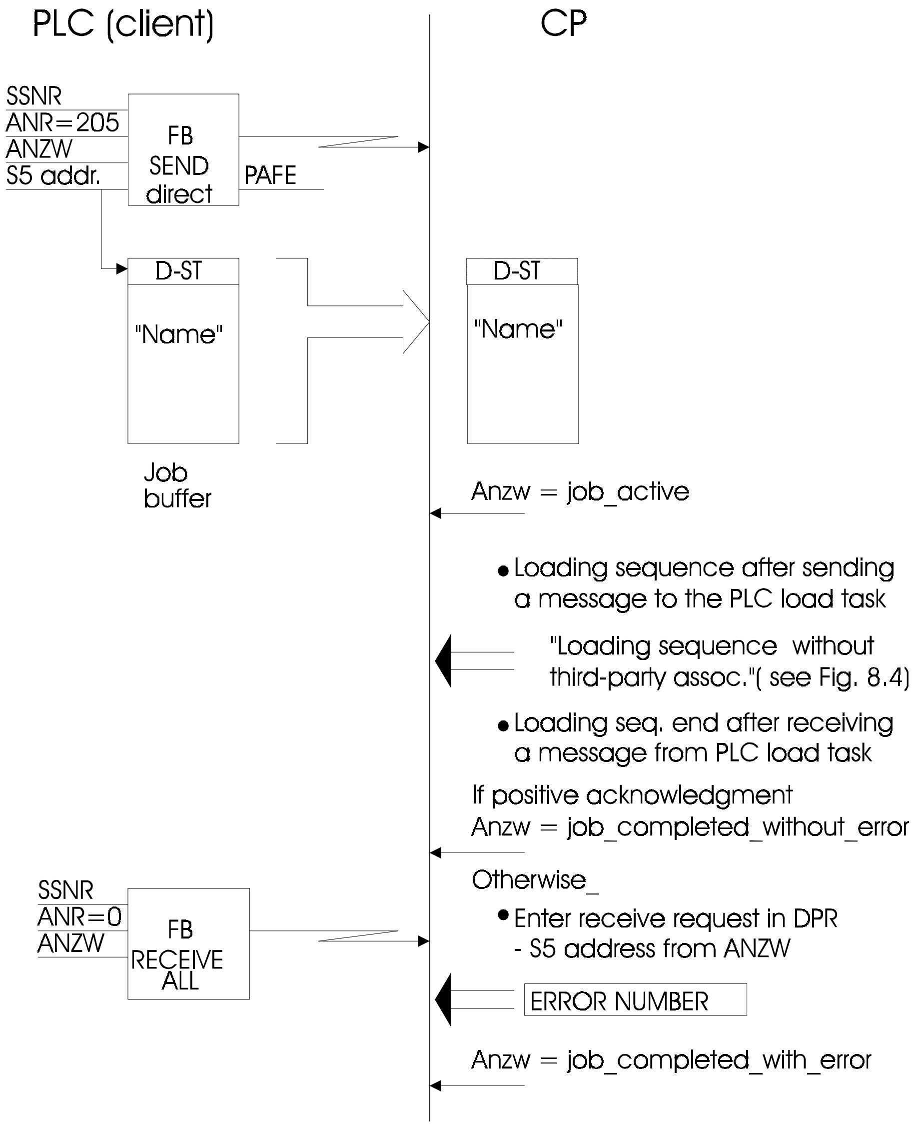


Fig. 8.12 Sequence 'Store Domain Content' (local PLC)

8.1.3 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.

"Delete Domain Content" Job Buffer

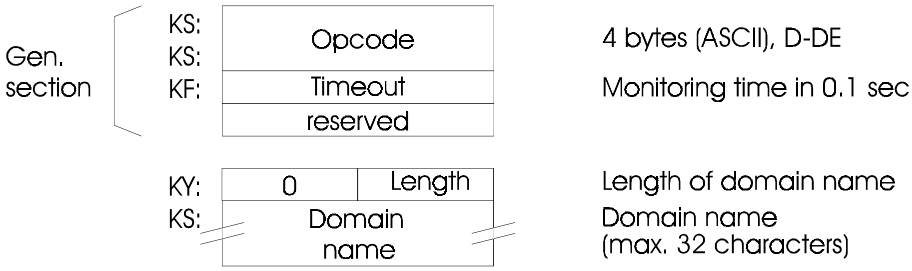


Fig. 8.13 Structure of the Job Buffer "Delete Domain"

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 acknowledgment 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 - 13.

Job-related section

Length of domain name: 1 word, format: KY
Possible 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, the last byte has no significance.

Sequence Description "Delete Domain Content"

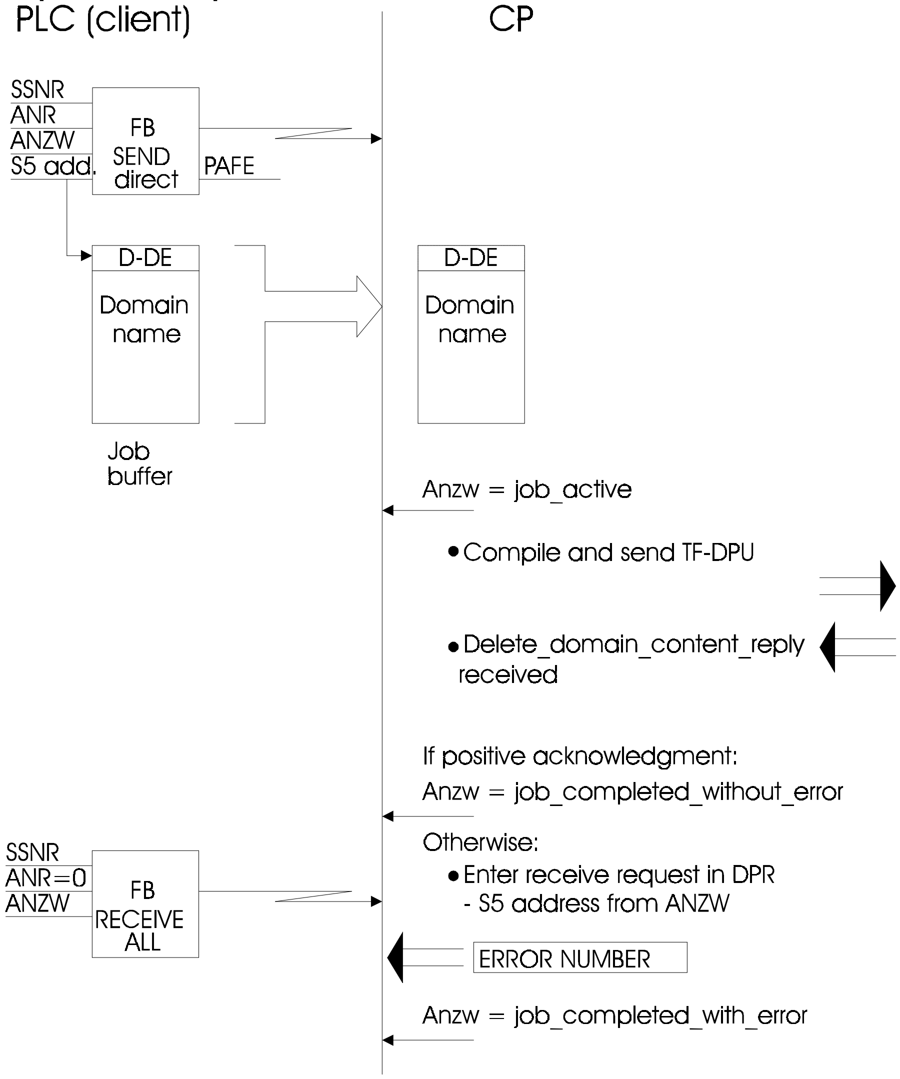


Fig. 8.14 Sequence "Delete Domain" (other PLC)

Note:

While the "delete domain" job is being processed, it is possible that domain-specific variables of other domains (loaded after the domains being deleted) are briefly not recognized. If the user program accesses these domain-specific variables during this time, access is denied.

The reason for this is that when domains are deleted, the memory for storing domain-specific variables is optimized

8.1.4 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"

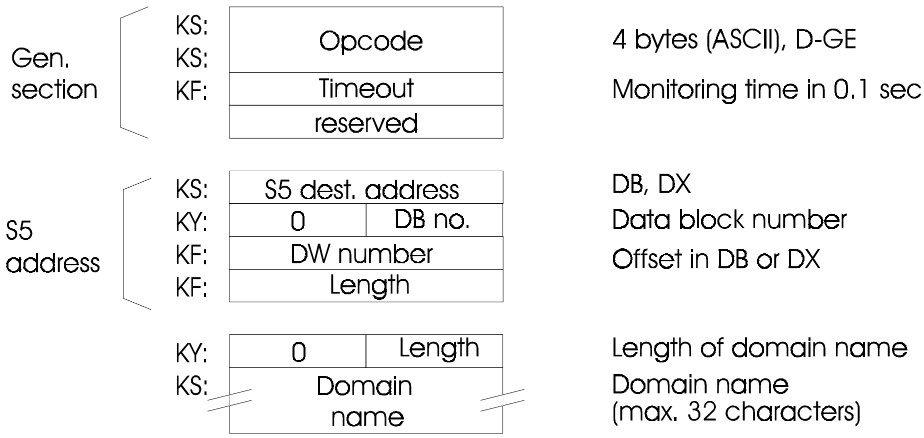


Fig. 8.15 Job Buffer "Get Domain Attributes"

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 acknowledgment 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 - 13.

Job-related section

Dest. ID	1 word, format:	KS
	Possible values:	DB, DX
	Meaning: S5 destination address, at which the information about the attributes of the domain will be stored.	
DB number	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..255
	Meaning: DB or DX number	
DW number	1 word, format:	KF
	Possible values:	0..2042
	Meaning: offset within the data block or extended data block.	
Length:	1 word, format:	KF
	Possible 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 acknowledgment from the DW number up to the end of the data block can be accepted.	
Length of domain name:	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..32
	Meaning: length of the following domain name	
Domain name:	n bytes, format:	KS
	Name of the domain. If the length of the domain name is odd, the last byte has no significance.	

Sequence Description "Get Domain Attributes"

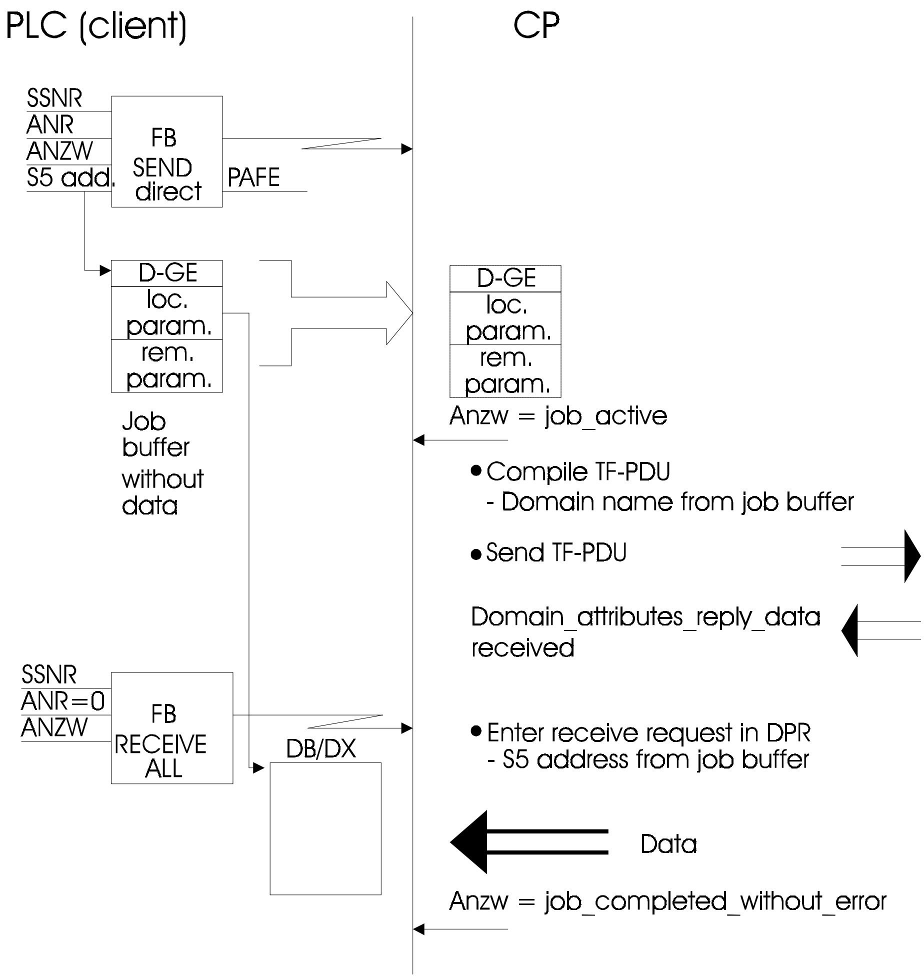


Fig. 8.16 Sequence Description (Get Domain Attributes, Positive Acknowledgment)

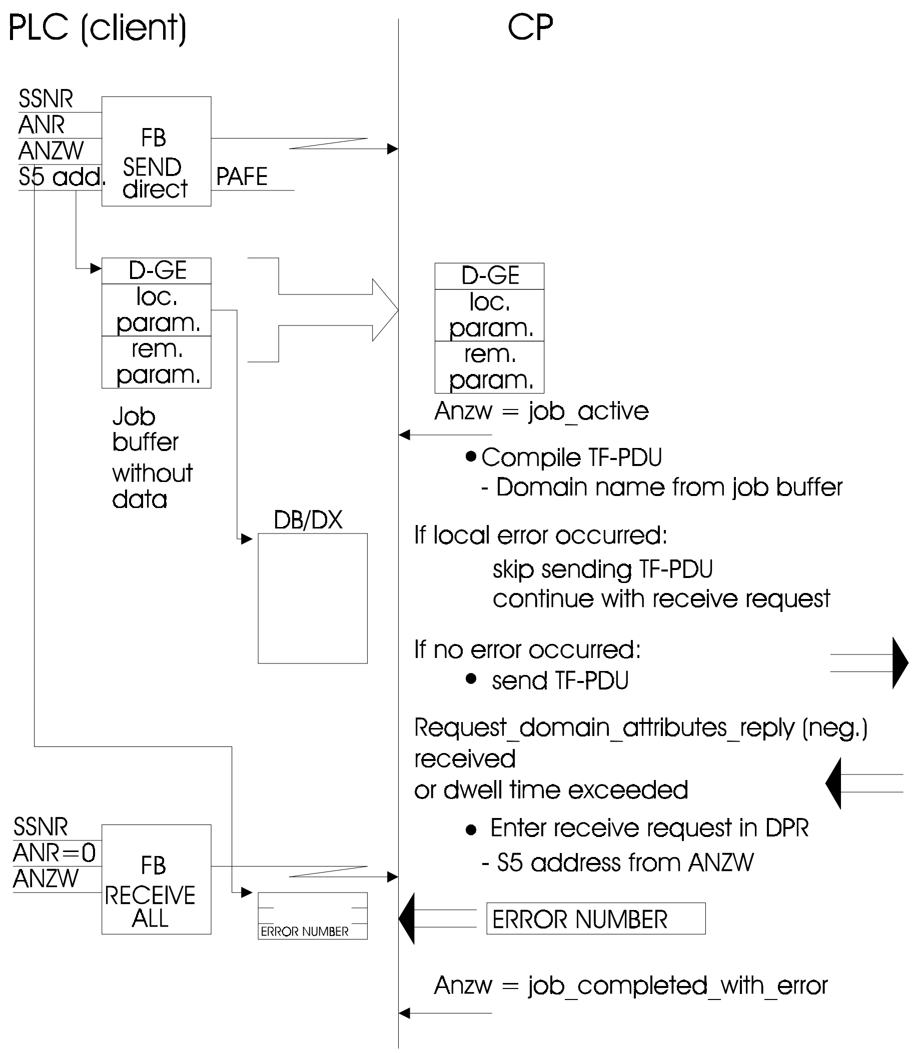


Fig. 8.17 Sequence Description (Get Domain Attributes, Negative Acknowledgment)

Structure of the reply data

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

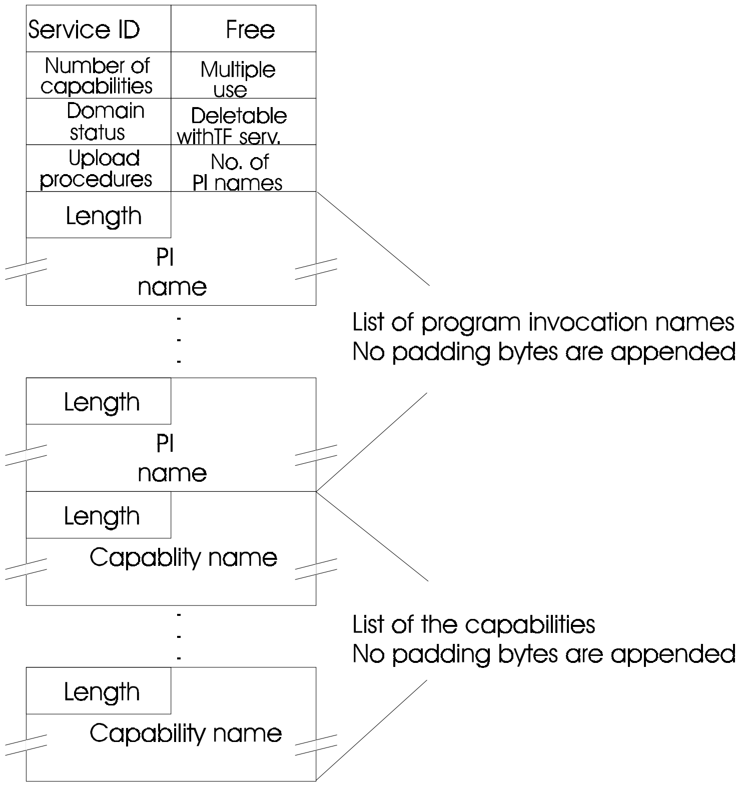


Fig. 8.18 Structure of the Reply Data

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
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 (in use)
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 file server. 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").

8.1.5 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.

8.2 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

8.2.1 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.

Number of Domains

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 modeling 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 using 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 1430 TF.

Status Transitions

The PI status changes defined in TF can be seen in the Fig. 8.19. 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 1430 TF configuration tool, with the function VMD Configuration.

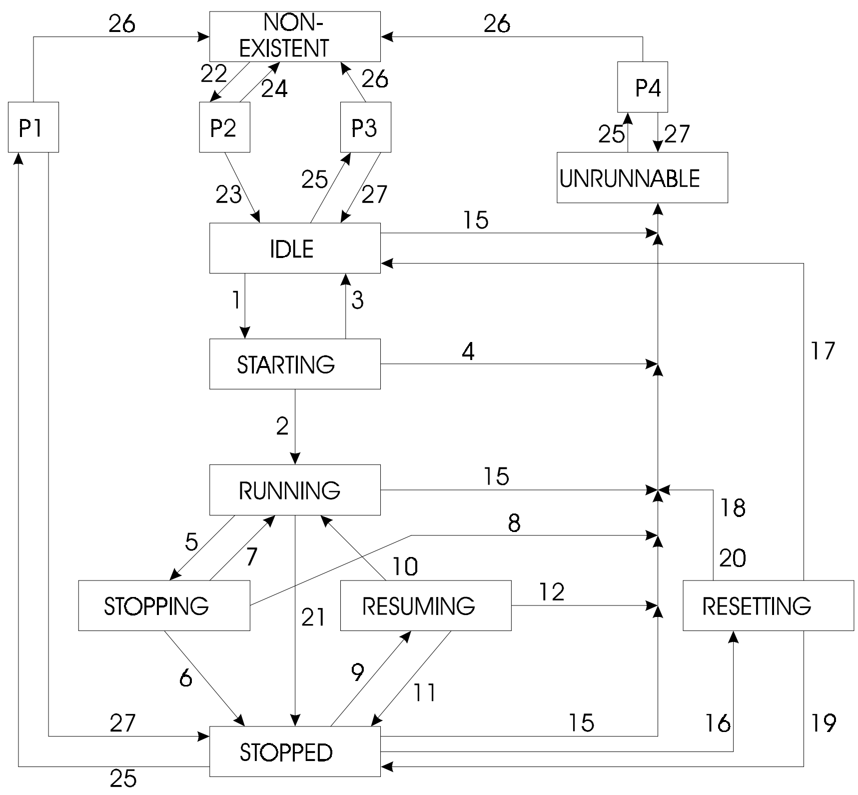


Fig. 8.19 Status Diagram of a Program Invocation

Explanation of the status transitions::

1 Start job
2 Start acknowledgment (positive)
3 Start acknowledgment (negative, non-destructive)
4 Start acknowledgment (negative, destructive)
5 Stop job
6 Stop acknowledgment (positive)
7 Stop acknowledgment (negative, non-destructive)
8 Stop acknowledgment (negative, destructive)
9 Resume Job
10 Resume acknowledgment (positive)

11 Resume acknowledgment (negative, non-destructive)
12 Resume acknowledgment (negative, destructive)
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.

➢ 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 (acknowledgment 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 (acknowledgment of the PI start job)
PI stopped	The process controlled by the PLC program is (temporarily) stopped. As can been 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 stop. The change from this status (to running or stopped/unrunnable) is triggered by the user program (acknowledgment 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 (acknowledgment 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".

P1 - P4 — 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.

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 Section 9.1.4 Special Connections.

➢ Assign the data and program blocks to domains (refer to domain services and the PG Load 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 Section 9.1.4 Special Connections.

➢ 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.

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

☞ **Note that PI and domain services are not permitted with SSNR 232, 236 and 244.**

8.2.2 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 1430:

➢ Request received

➢ Requested status change checked for consistency

➢ If status change not allowed, negative acknowledgment and abort

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

➢ If applicable, wait for acknowledgment 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 acknowledgment 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").

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

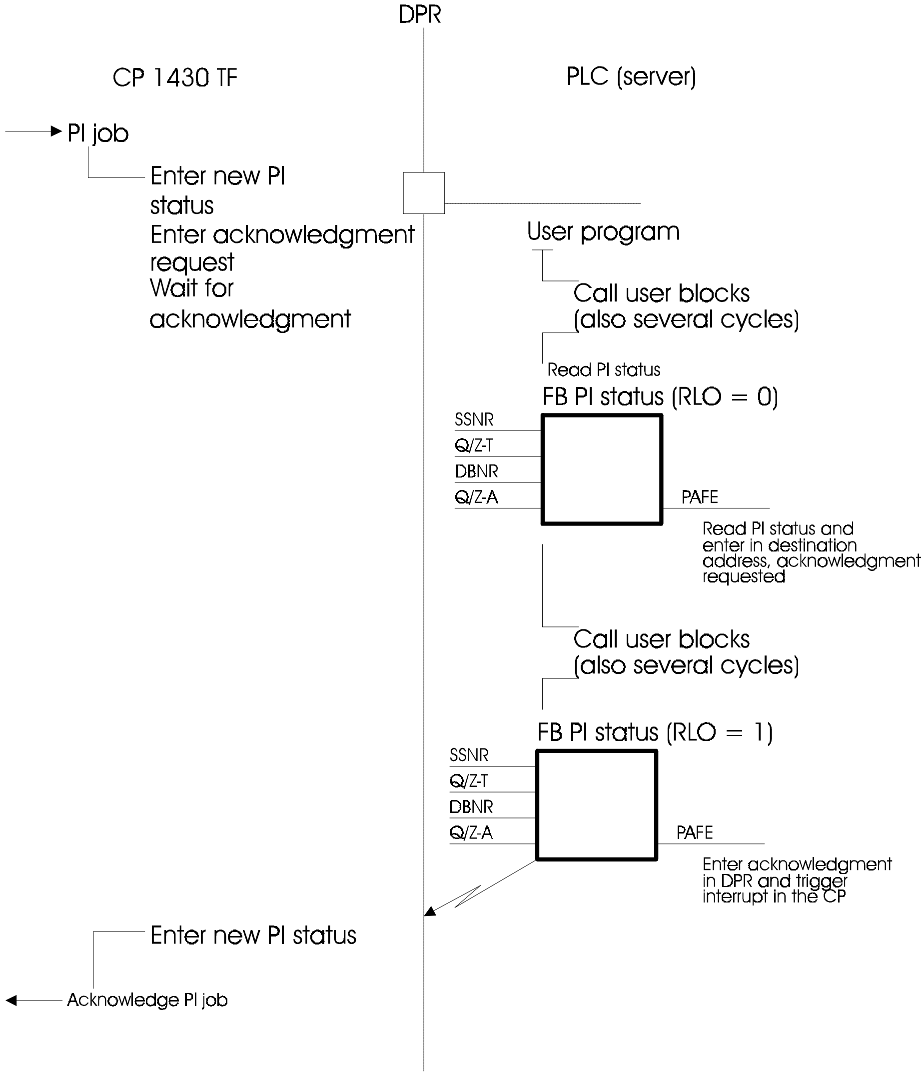


Fig. 8.20 General Sequence (no error) of a Status Change

8.2.3 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 SINEC TF.

Block Function

The block executes two different functions: read PI status or send acknowledgment. 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 acknowledgment at the specified S5 address and then writes the new PI status at this address.

Block Interface

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

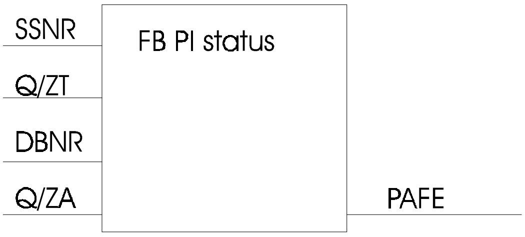


Fig. 8.21 Call Parameters of the Status Function Block "PI-ZUSTD"

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...255

   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).

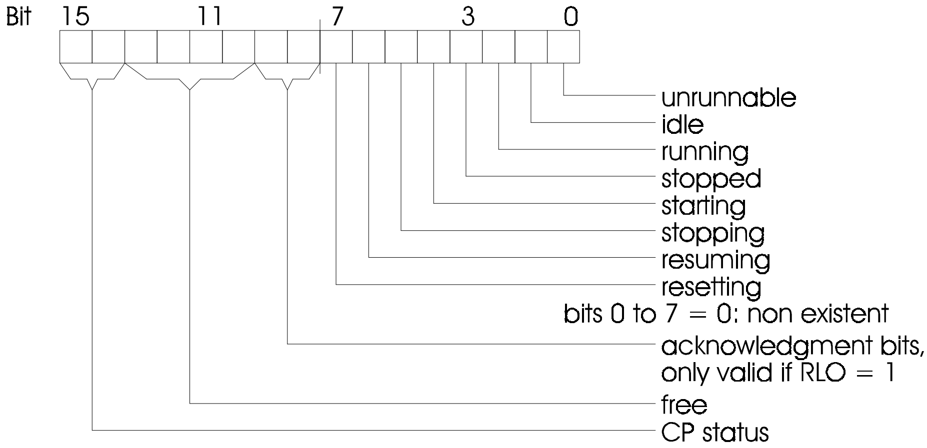
List of possible error messages and their causes:

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 acknowledgment 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

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

PLC	CPU	S5 file	Library number
S5-115 U/H	CPU 942A/B CPU 942 R CPU 943A/B CPU 944A/B	S5CI50ST.S5D	P71200-S 5103-C-2
	CPU 945	---------------	---------------
S5-135 U S5-155 U/H	CPU 922 CPU 928A/B	S5CI29ST.S5D	P71200-S 9103-C-1
	CPU 946/947 CPU 946/947 R CPU 948 CPU 948 R	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 acknowledgment:

- Starting
- Stopping
- Resuming
- Resetting

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

Significance of the acknowledgment bits

➢ Bits 8 and 9 = 0:
The status transition will be executed (pos. acknowledgment).

➢ Bit 8 = 1, bit 9 = 0:
The status transition will not be executed, the PI is to remain in the old status (negative, acknowledgment, non-destructive).

➢ Bit 8 = 0 or 1, bit 9 = 1:
The PI must be brought to the "unrunnable" status (negative acknowledgment, 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 acknowledgment is required, the user must make sure that the acknowledgment is only triggered when all the CPUs require the status change. The CP itself can only accept one acknowledgment.

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 1430 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.

8.2.4 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 1430 tool PG Load. The PG (as an aid to installation) transfers the files as "load files" to a file server 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).

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.

8.2.5 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).

"Create Program Invocation" Job Buffer

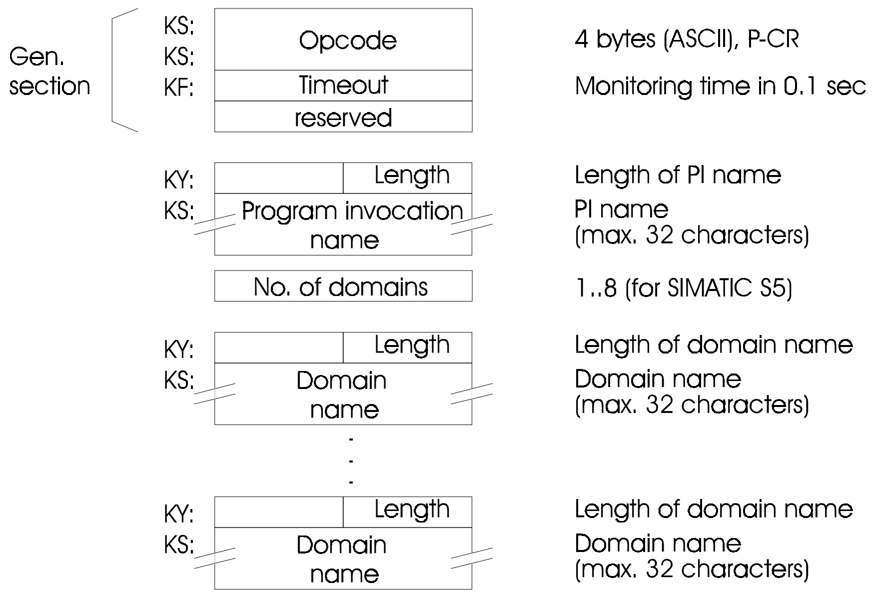


Fig. 8.22 Structure of the Job Buffer for TF Service "Create Program Invocation"

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 acknowledgment 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 - 13.

Job-related section:

Length of PI name	1 word, format: KY Possible values: 1..32 Meaning: length of the following PI name
PI name	1 word, 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	1 word, format: KF Possible values: 1..8 Meaning: specifies the number of domains to be collected 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	1 word, format: KY Possible values: high byte: 0, low byte: 1..32 Meaning: length of the following domain name
domain name	1 word, format: KS Meaning: name of the domain, if the length is odd, a padding byte is appended.

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

Job sequence "create program invocation"

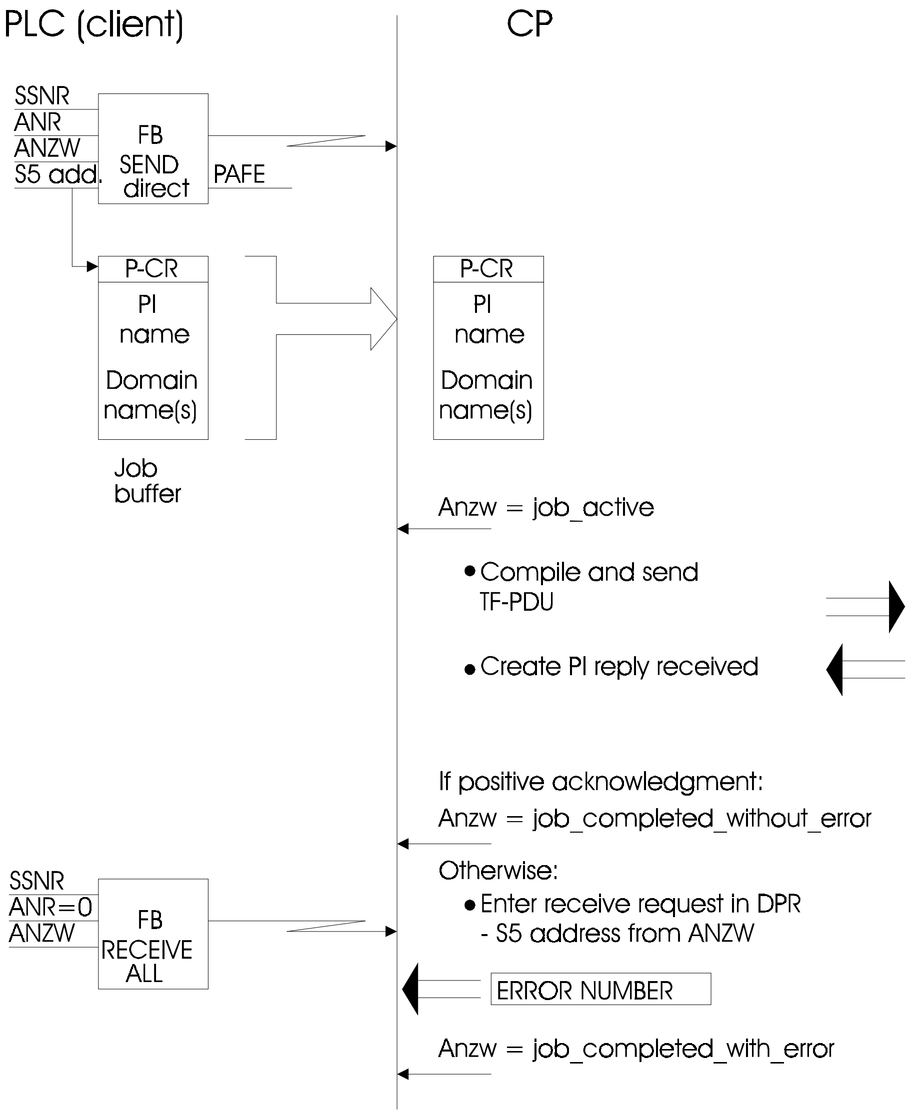


Fig. 8.23 Sequence "Create Program Invocation "

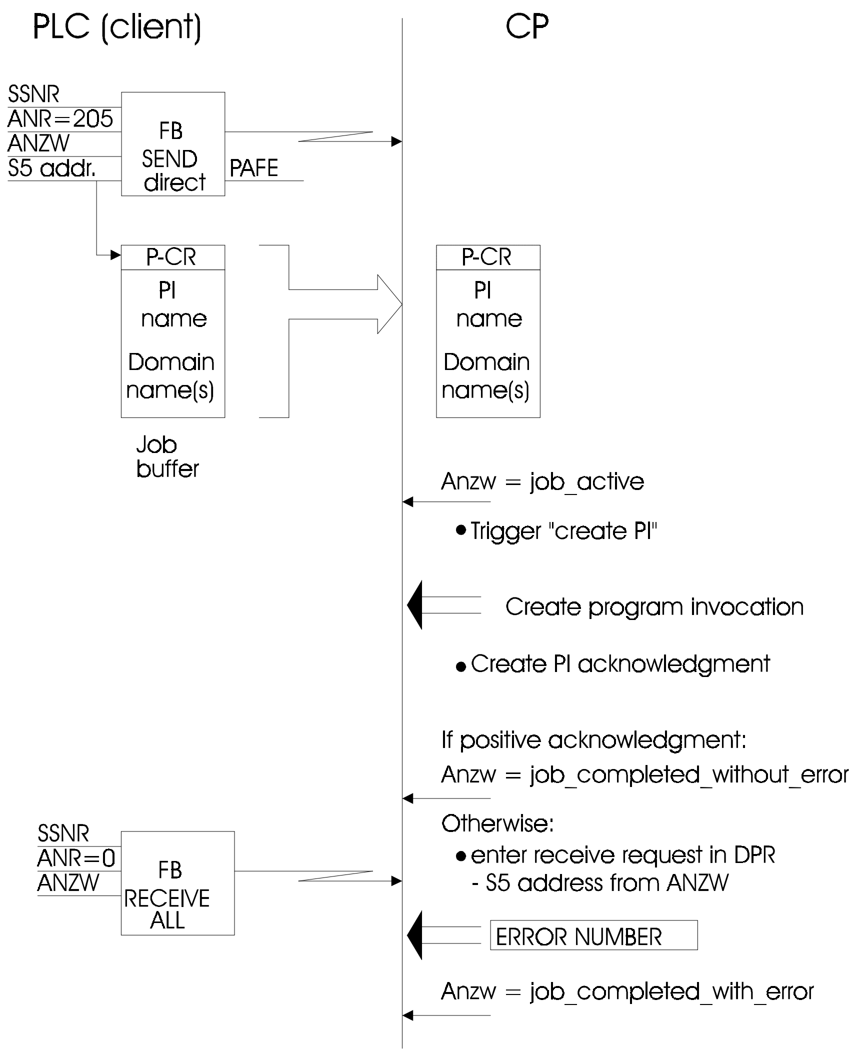


Fig. 8.24 Sequence "Create Program Invocation" on Local PLC

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

8.2.6 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)

➢ Acknowledgment 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
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.

8.2.7 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.

"Delete PI" Job Buffer

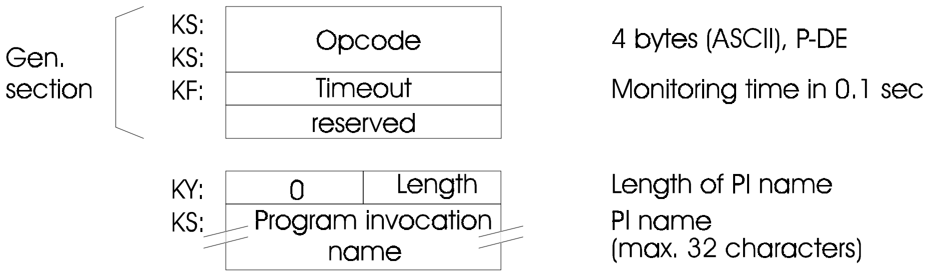


Fig. 8.25 Structure of the Job Buffer for TF Service "Delete Program Invocation"

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 acknowledgment 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 - 13.

Job-related section:

Length of PI name	1 word, format: KY Possible values: 1..32 Meaning: length of the following PI name
Program invocation name	1 word, 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"

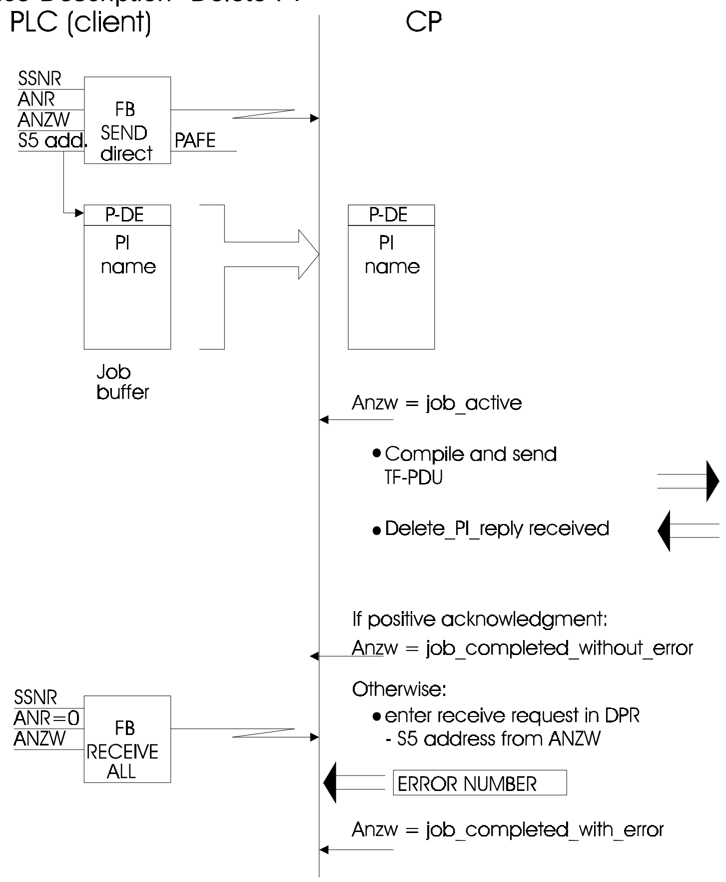


Fig. 8.26 Sequence 'Delete Program Invocation'

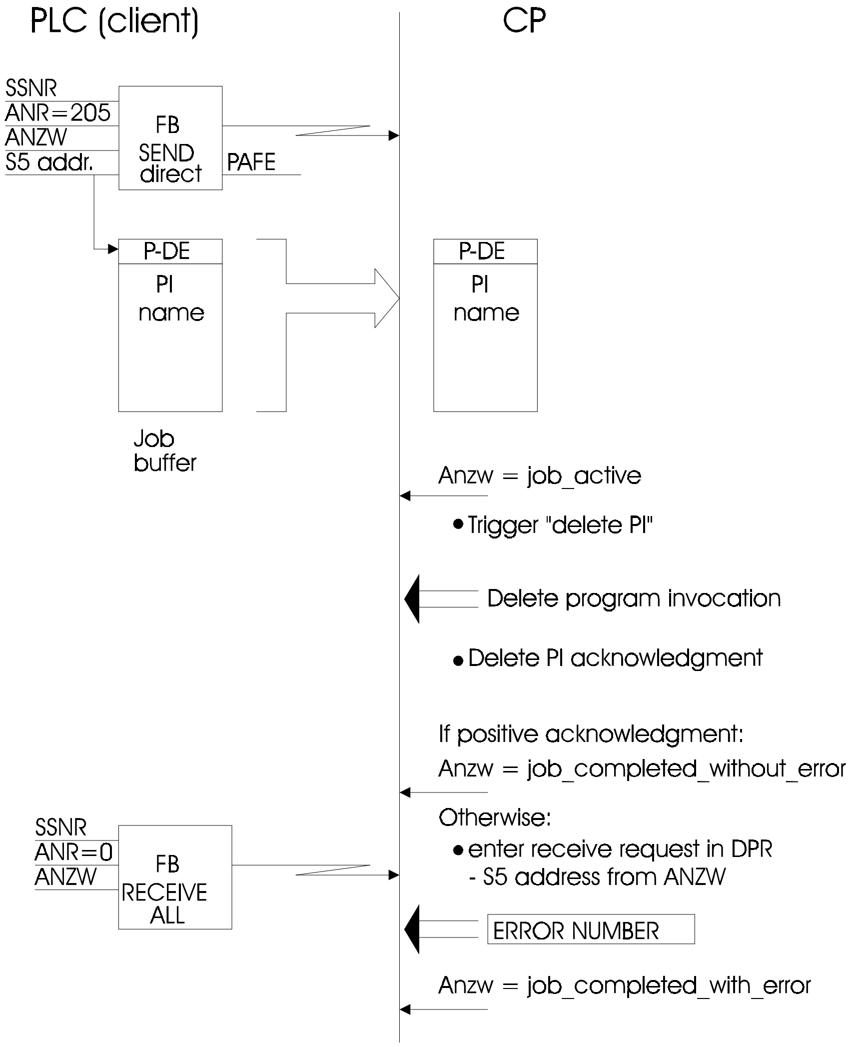


Fig. 8.27 Sequence 'Delete Program Invocation' (on Local PLC)

8.2.8 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.

8.2.9 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"

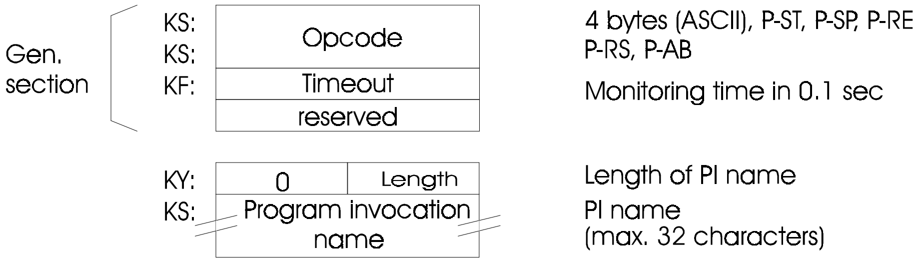


Fig. 8.28 Structure of the Job Buffer

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 acknowledgment 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 - 13.

Job-related section:

Length of PI name: Format: KY
Possible 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 8-35).

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.

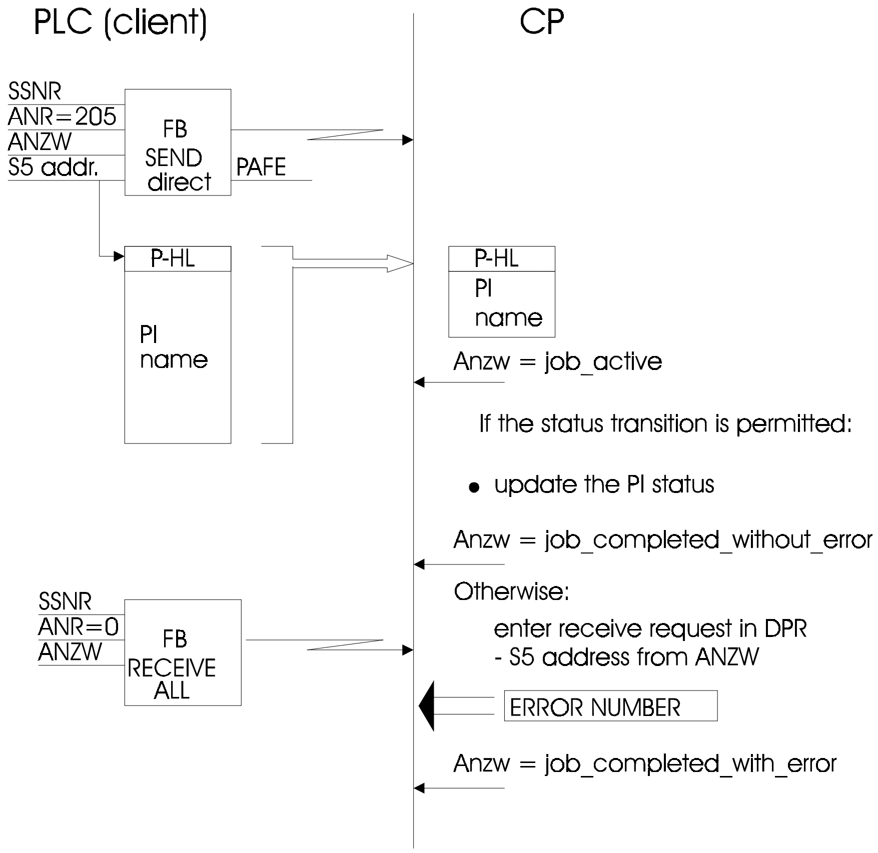


Fig. 8.29 Sequence "Local Program Stop"

8.2.10 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 8.2.2

Note the information about handling the user program in Section 8.2.3 ’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.

8.2.11 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 1430 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:**
If the system PI is changed to the "PI stopped" status with the stop PI service, and if this service does not operate on the predefined application association, the CP 1430 TF does not permit the association to be initiated again after termination until the CP 1430 TF returns to the RUN mode.

It is therefore advisable to use only the predefined application association for the stop system PI and start system PI services (see Section 9.1.4 "Special Connections").

8.2.12 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"

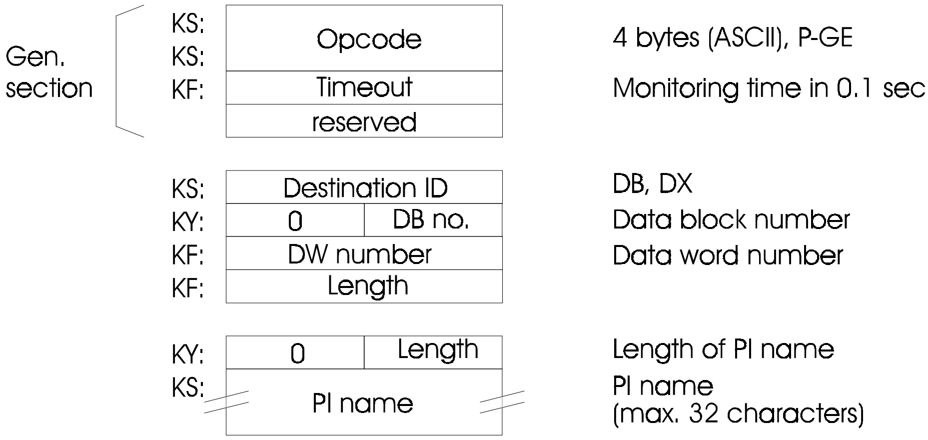


Fig. 8.30 Structure of the Job Buffer "Get PI Attributes"

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 acknowledgment 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 - 13.

Job-related section:

Dest ID:	1 word, format:	KS
	Possible values:	DB, DX
	Meaning: S5 destination address at which the information about the attributes of the PI will be stored.	
DB number:	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..255
	Meaning: DB or DX number	
DW number:	1 word, format:	KF
	Possible values:	0..2042
	Meaning: offset within the data block or extended data block.	
Length:	1 word, format:	KF
	Possible 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 acknowledgment from the DW number to the end of the data block can be accepted.	
Length of PI name:	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..32
	Meaning: length of the following PI name	
PI name:	1 word, format:	KS
	Meaning: name of PI, if the length of the PI name is odd, the last byte has no significance.	

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.

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:

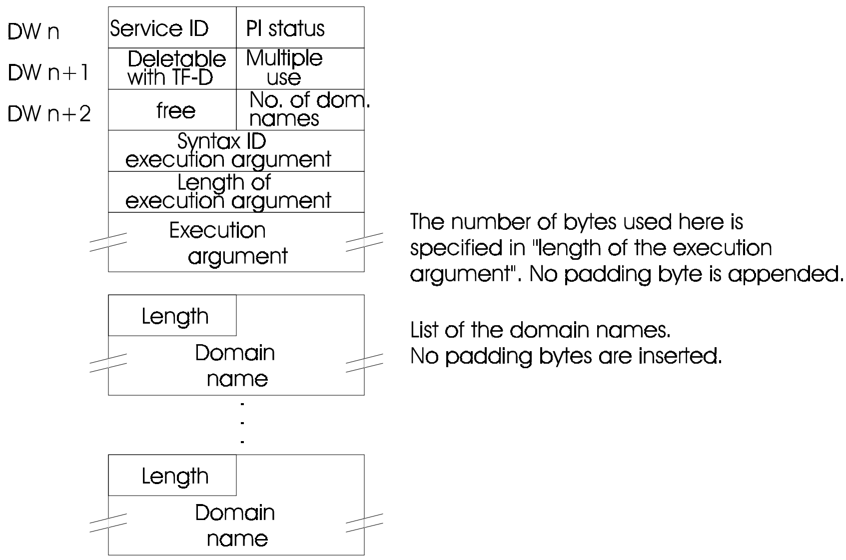


Fig. 8.31 Structure of the Job Buffer "Get PI Attributes"

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 or 0. This defines a visible sting 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.

Special feature with the job "get attributes of a system PI"
Before this service is executed, the current PLC status (RUN/STOP) is queried.

8.2.13 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.

❑

Notes

9 Supplementary Services

Topics in this Chapter

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

➢ Application association management

➢ VMD services

➢ Configuration jobs

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.

9.1 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 association management. These services are executed automatically by the CP 1430 (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.

9.1.1 Definition of Application Associations

Application associations are defined in the COM 1430 TF "Fefinitions" dialog and stored in the connection blocks of the CP.

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

Global Parameters

Definition of the layer 4 connection with the parameters

- local TSAP
- remote TSAP
- remote MAC address

Local parameters

Definition of the application association (layer 7) and modeling 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

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 connection 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 connection is required between the CP 1430 and CP 535 with the AP protocol handler SINEC AV/S5.

Normally, establishment up to layer 7 should be selected.

The 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 application association.

dynamic
The CP establishes a connection as soon as there is a job for the configured application association.

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

9.1.2 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 connections (priority 3 and 4) 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 connections (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).

CP 1430 transport software: RTS

Layer 7

The establishment of the layer 7 connection 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
Value: 0FFH

Meaning: any number of server jobs per application association can be managed in the CP at one time.

Max_AmQ_called_proposition
Value: 1H
Meaning: the remote VMD must expect a maximum of one server job on an application association at one time.

Max_receive_buffer_calling
Meaning: the maximum length of a TF-PDU can be selected by the user in COM 1430 TF for all application associations when the module is initialized. The value is entered in the connection block.

Non_open_services_calling
Value: 0000 0000 0000 0111B
Meaning: the following non-open TF services are supported:

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

Nesting_level_proposition
Value: 2H
Meaning: the CP supports variables up to the following complexity:

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

Number_syntax
>Value: 1H
Meaning: exactly one syntax field follows

Syntax_ID
Value: 0H (assigned by TF)

Length_abstract_syntax
Value: 5H

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

Length_Transfer_Syntax
Value: 0FH

Transfer_Syntax
Value: SINEC_TF_CODING

Version number_proposition
Value: 1H

Parameter_CBB_proposition
Value: 0000 0000 0101 1111
Meaning:
- Data type "ARRAY" is supported
- Data type "STRUCTURE" is supported
- Access to variables using a name is supported
- Access to variables using an address is supported
- Third-party association for loading domain is supported

Supported_TF_services_calling:
Meaning: all TF services of level 1 are supported (see appendix "CP 1430 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 acknowledgment, it negotiates that the partner settles for these values.

☞ **The specific implementation parameters are supplied by the CP 1430 and do not need to be set by the user.**

9.1.3 Connection Termination

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

Following the termination of the layer 7 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 connection 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.

9.1.4 Special Connections

Application Associations to a File Server

To load or store domains, the "third-party association" is supported by the CP 1430 TF. 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 file server application associations can be defined, of which a maximum of one can be established at any time.

The type of connection establishment for file server 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.

The TF file server application association does not have an ANR assigned to it. It is displayed by the test functions of COM 1430 TF under application associations (without ANR) and can be tested with single status or trace.

Predefined Application Association

The CP 1430 TF 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 MAC address:	unspecified
MUX address:	0

With this connection endpoint (ANR 206), the CP 1430 TF waits for the application association to be initiated by any partner (establishment type P7 static), such as the PG with PG Load for load services or host functions (application association PG-PLC when configuring PG Load).

This predefined application association is always displayed in the test functions under application association with ANR 206 (server) and for local client jobs (ANR 205).

9.2 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)

9.2.1 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 acknowledgment (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"

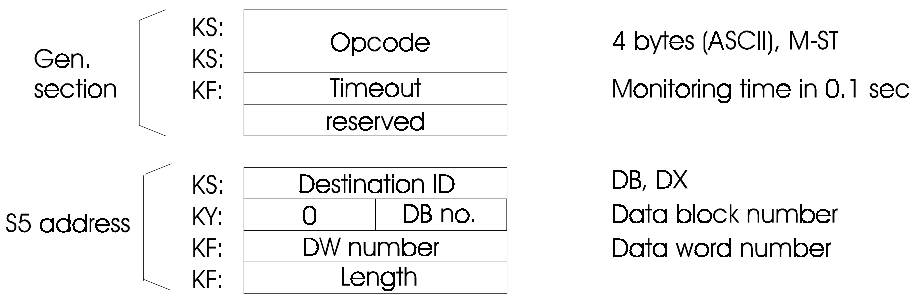


Fig. 9 - 1 Structure of the Job Buffer "VMD Status"

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 acknowledgment 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 - 13.

Job-related section

Dest. ID	1 word, format:	KS
	Possible values:	DB, DX
	Meaning: destination address at which the information about the status will be stored.	
DB number	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..255
	Meaning: DB or DX number	
DW number	1 word, format:	KF
	Possible values:	0..2042
	Meaning: offset within the data block or extended data block.	
Length	1 word, format:	KF
	Possible values:	1..2043, -1
	Meaning: length of the data block area in which the VMD status can be stored; the value -1 means that all the data in the acknowledgment from the DW number to the end of the data block can be accepted.	

Sequence Description

The sequence of the "VMD status" service is analogous to the sequence of the "read" TF variable service.

Requesting the status of the VMD in the local PLC (job number = 205) is not possible.

9.2.2 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:

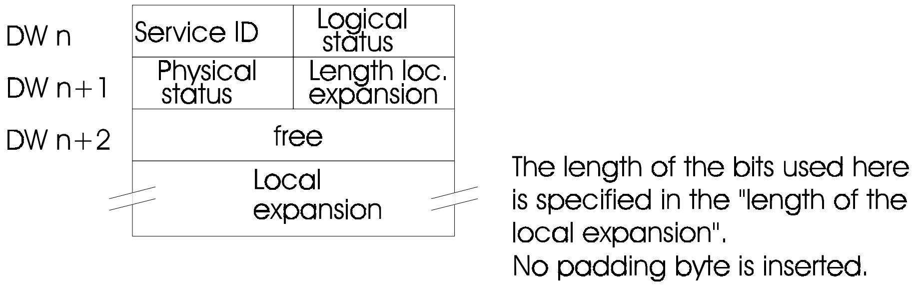


Fig. 9 - 2 Structure of the Reply Data in the Data Block

Service ID:
0h (to assign the reply to the required service unequivocally)

Logical status	1 byte, for IDs see page 9 - 17
Physical status	1 byte, for IDs see page 9 - 17
Length loc. expansion	1 byte, range of values 0 .. 128 Meaning: length of the local expansion in bits
Local expansion	Format: KM (8 words) Meaning: The meaning of the local expansion depends on the particular application association.

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

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

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

9.2.3 Unsolicited VMD Status (Initiator)

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

Job buffer "unsolicited VMD status"

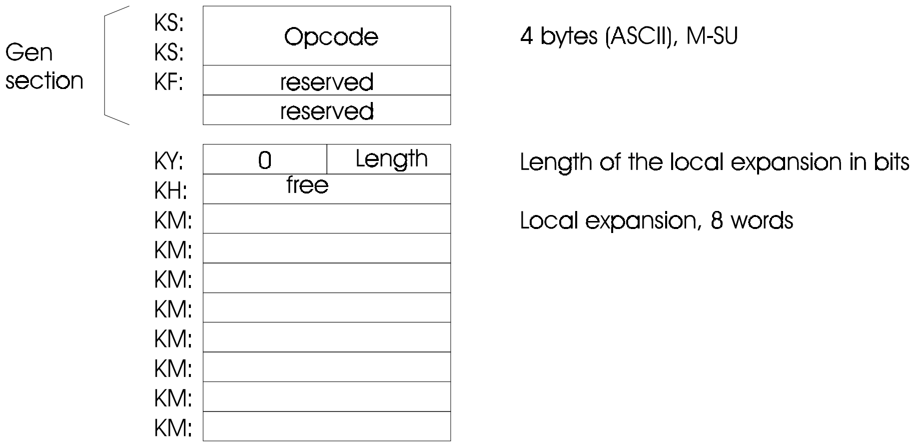


Fig. 9.3 Structure of the Job Buffer "Unsolicited Status"

Call description

General section:

Opcode M-SU

Job-related section:

Length of local expansion	1 word, format:	KY (low byte)
	Possible values:	0..128
	Meaning: the parameter length_of_local_expansion specifies the number of left-justified bits in the "local expansion" parameter.	

Local expansion:	1 word, format:	KM (8 words)
	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.

9.2.4 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.

9.2.5 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.

"Identify Virtual Device " Job Buffer

Gen. section	KS: KS:	Opcode		4 bytes (ASCII), M-ID
	KF:	Timeout		Monitoring time in 0.1 sec
		reserved		
S5 address	KS:	Destination ID.		DB, DX
	KY:	0	DB no.	Data block number
	KF:	DW number		Data word number
	KF:	Length		

Fig. 9.4 Structure of the Job Buffer " Identify"

Call Description

General section:

Opcode M-ID

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgment 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 - 13.

Job-related section:

Dest. ID: 1 word, format: KS
Possible values: DB, DX
Meaning: address at which the information about the identity of the remote station will be stored.

DB number:	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..255
DW number:	1 word, format:	KF
	Possible values:	0..2042
Length:	1 word, format:	KF
	Possible values:	1..2043, -1

Meaning: length of the data block area in which the information contained in the acknowledgment can be stored; the value -1 means that all the data in the acknowledgment 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.

9.2.6 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 1430 TF at the S5 address specified in the job buffer has the following structure:

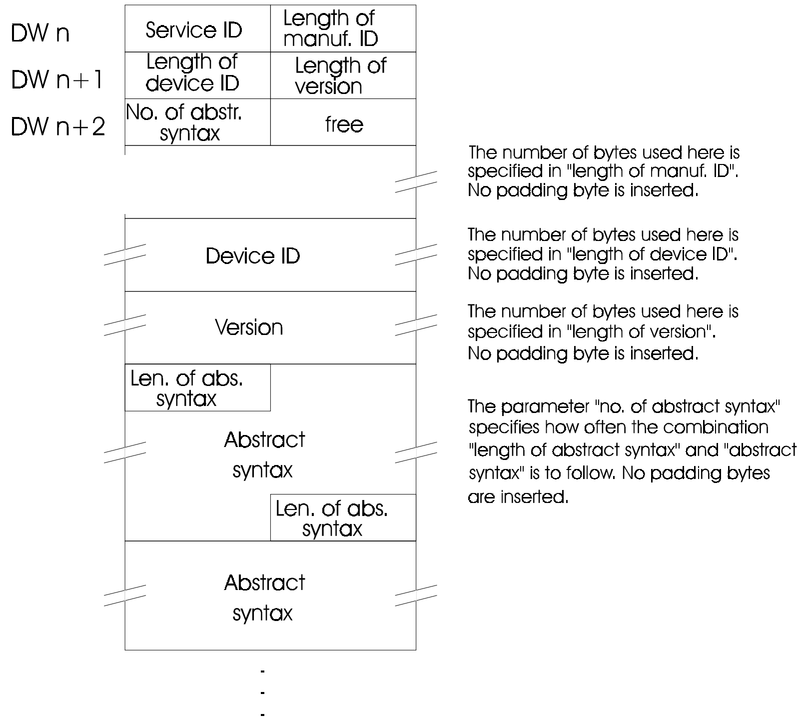


Fig. 9.5 Structure of the Reply Data in the Data Block

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 1430 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 1430 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 1430 TF does not support any CS_parameters (Companion Standards) that may occur with the TF services (variable services, general services, domain and program services).

9.3 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 1430 TF while it is running. The configuration only ever applies to the connection 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 1430 TF 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.

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 configured.

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 1430 TF 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]"

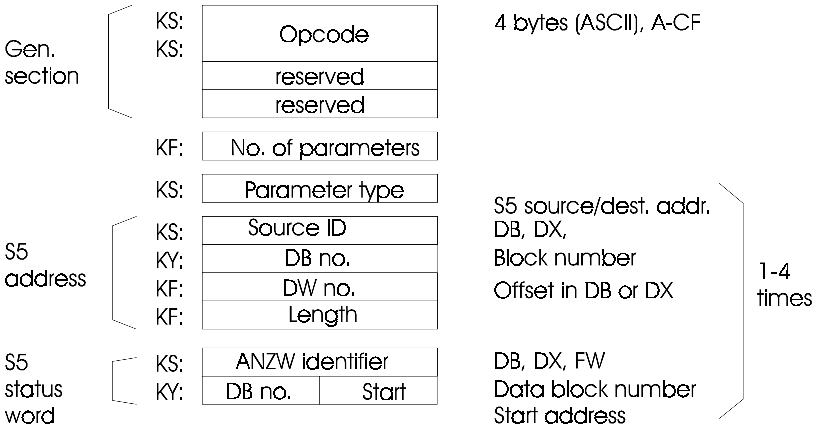


Fig. 9.6 Structure of the Job Buffer "Configure ANZW (local)"

Call Description

General section:

Opcode	A-CF	
No. of parameters:	1 word, format:	KF
	Possible values:	1..4
	Meaning: up to 4 parameters can be transferred simultaneously.	
Parameter type	1 word, format:	KS
	Possible values:	AN, BL, BS, MS
	Meaning: specifies the parameter to be configured	

S5 address:

Source/ dest. ID:	1 word, format:	KS
	Possible values:	DB, DX
	Meaning: source/destination address identifier for configuring the address for the services read/_write_byte_string. Invalid for parameter type "AN"	
DB no.	1 word, format:	KY
	Possible values:	high byte: 0, low byte: 0..255
	Meaning: data block number for the parameter types "BL", "BS" and "MS", value 0 only allowed for parameter type "AN"	
DW no.:	1 word, format:	KF
	Possible values:	0..2042
Length:	1 word, format:	KF
	Possible values:	1..2043, -1
	Meaning: length of the data block area to be transferred with the "read byte string" service or that must be made available for the "write byte string" service. The value -1 means that all data can be accepted (only permitted for parameter type "BS")	

For parameter type "MS" this parameter is invalid. In this case the length is implicitly 11 words.

Status word

ANZW identifier:	1 word, format:	KS	
	Possible values:	FW, DB, DX	
	Meaning: status word type		
ANZW specification	1 word, format:	KY	
	Possible values:	high byte:	block number
		low byte:	DW number, FW number

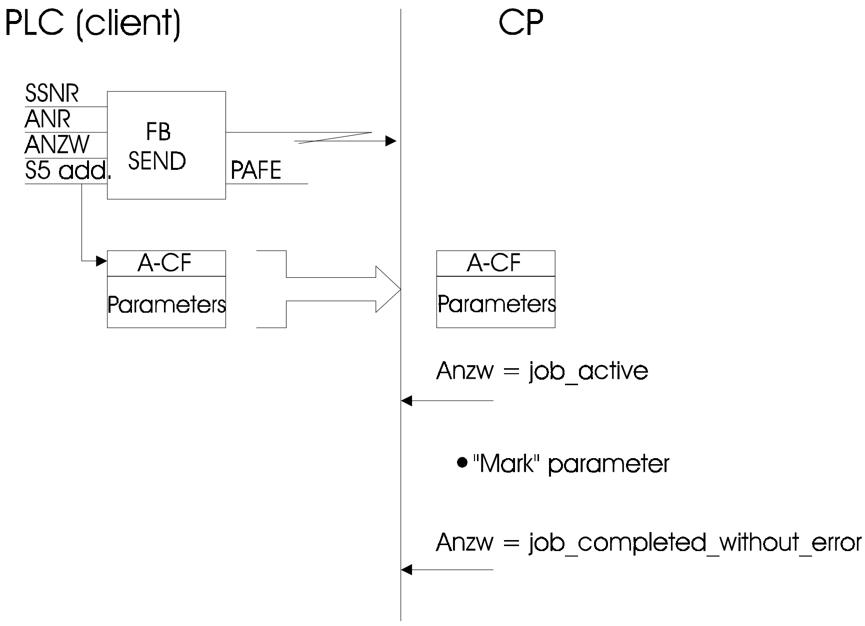


Fig. 9.7 Sequence: "Configure ANZW (local)"

❑

10 Non-Open Services for Serial Transfer

Topics in this Chapter

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 modeled 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)

10.1 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 acknowledgment 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 acknowledgment. The client can decide whether or not an acknowledgment 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.

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

10.2 Read Byte String (Client)

Request transfer of a data area.

"Read Byte String" Job Buffer

Gen. section	KS: KS:	Opcode	4 bytes (ASCII), B-RQ
	KF:	Timeout	Monitoring time in 0.1 sec
		reserved	
	KS:	Dest. identifier	DB, DX
	KY:	0 | DB no.	Data block number
	KF:	DW number	Data word number
	KF:	Length	

Fig. 10.1 Structure of the Job Buffer "Read Byte String"

Call Description

General section:

Opcode: B-RQ

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgment 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 - 13.

Job-related section:

Dest. identifier: 1 word, format: KS
Possible values: DB, DX
Meaning: S5 address at which the byte string read out will be stored.

DB number: 1 word, format: KY
Possible values: high byte: 0, low byte: 1..255
Meaning: DB or DX number

DW number: 1 word, format: KF
Possible values: 0..2042
Meaning: offset within the data block or extended data block.

Length: 1 word, format: KF
Possible values: 1..2043
Meaning: length of the data block area in which the byte string can be written.

Sequence of "read byte string"

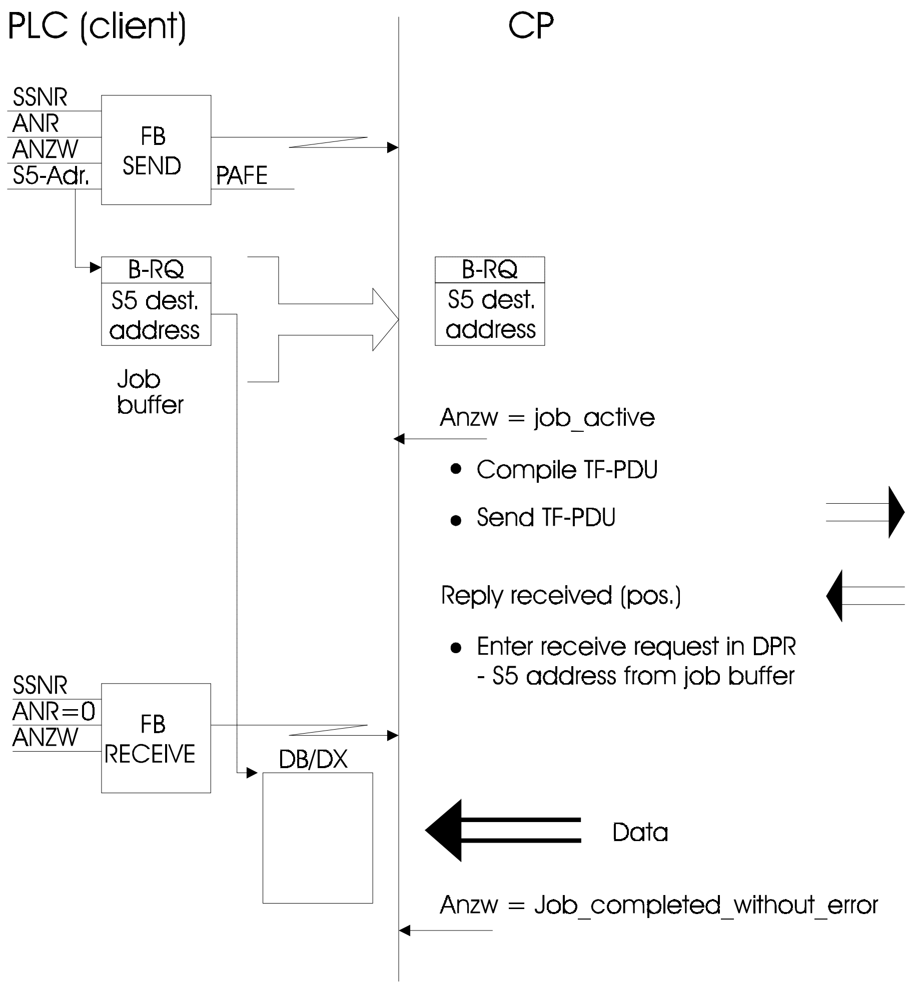


Fig. 10.2 Sequence "Read Byte String"

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").

10.3 Write Byte String (Client)

Transfer a data area to the partner

Job buffer "write byte string "

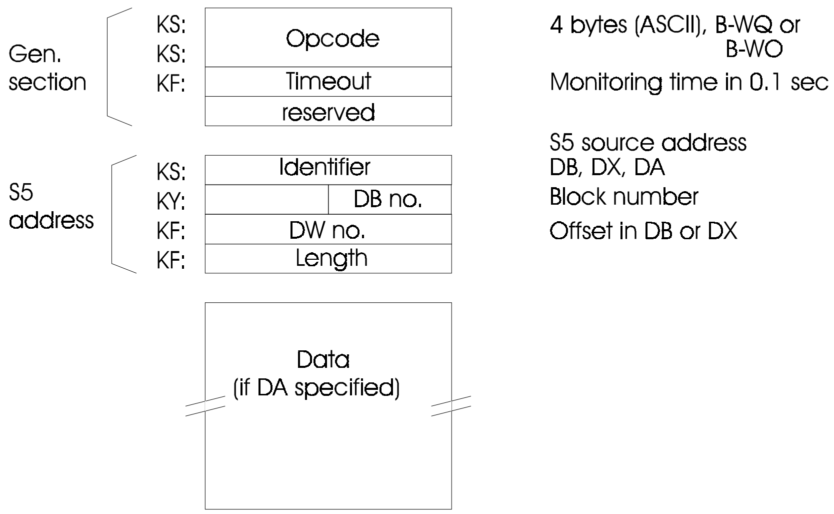


Fig. 10.3 Structure of the Job Buffer "Write Byte String"

Call Description

General section

Opcode: B-WQ ("write byte string with acknowledgment")
or
B-WO ("write byte string without acknowledgment")

Timeout: 1 word, format: KF
Specifies the maximum length of time the user program will wait for an acknowledgment 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 - 13.

Job-related section

Source ID: 1 word, format: KS
Possible 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
Possible values: high byte: 0
low byte: 1..255
Invalid with source ID DA

DW number: Format: KF
Possible values: 0..2042
Invalid with source ID DA

Length: Format: KF
Possible values: 1..2043
Meaning: length of the data block area to be transferred with the service.

Sequence of "write byte string"

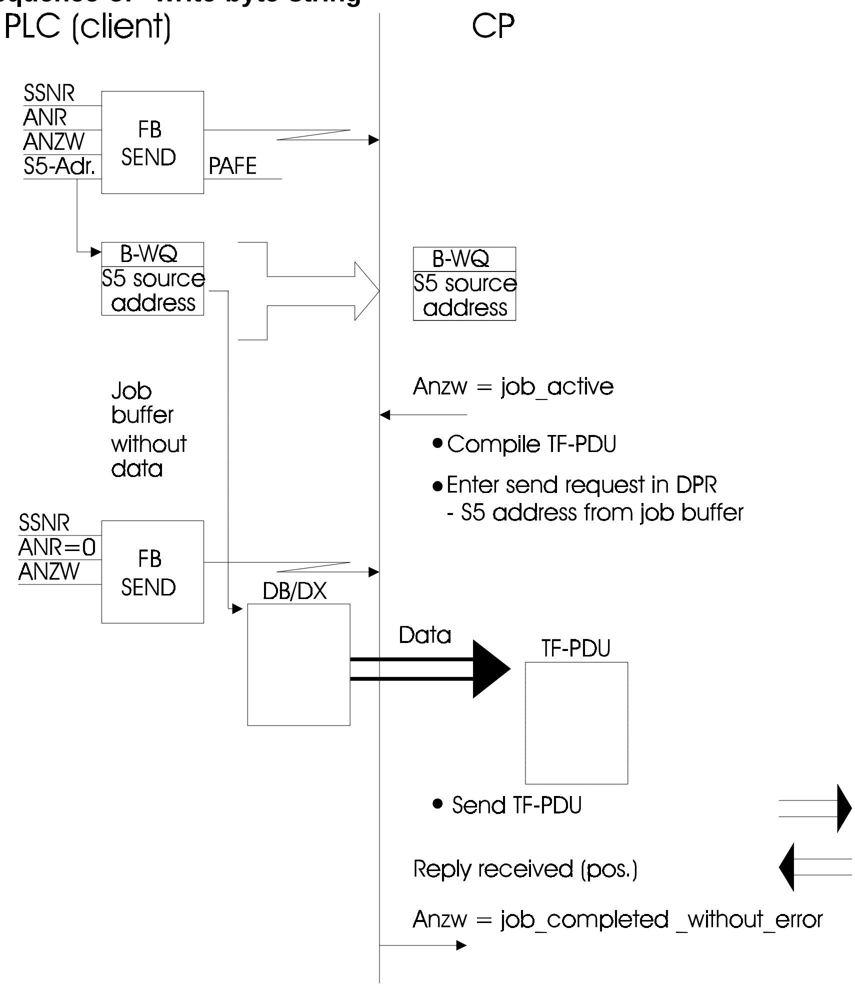


Fig. 10.4 Write Byte String with Acknowledgment (Example: Source ID = DA)

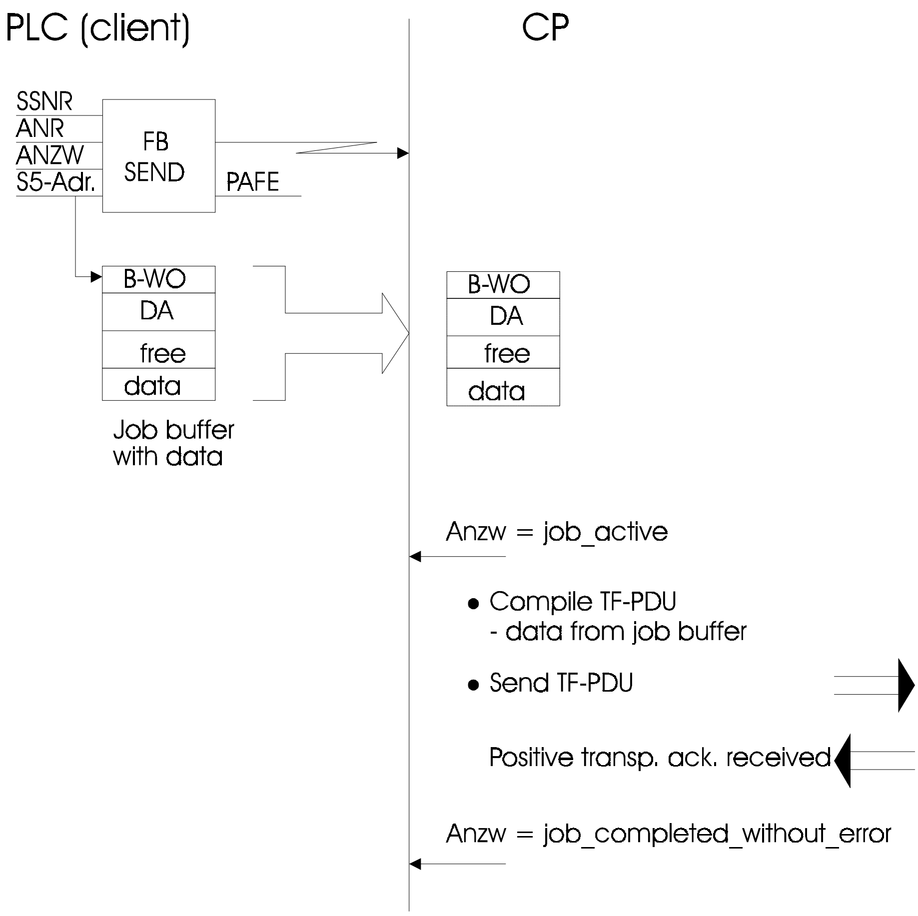


Fig. 10.5 Write Byte String without Acknowledgment (Example: Source ID = DB/DX)

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 acknowledgment (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.

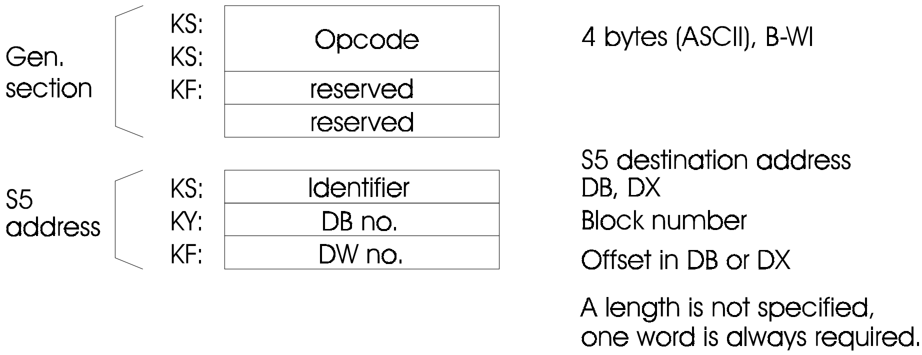


Fig. 10.6 Structure of the Job Buffer "Request Byte String Length"

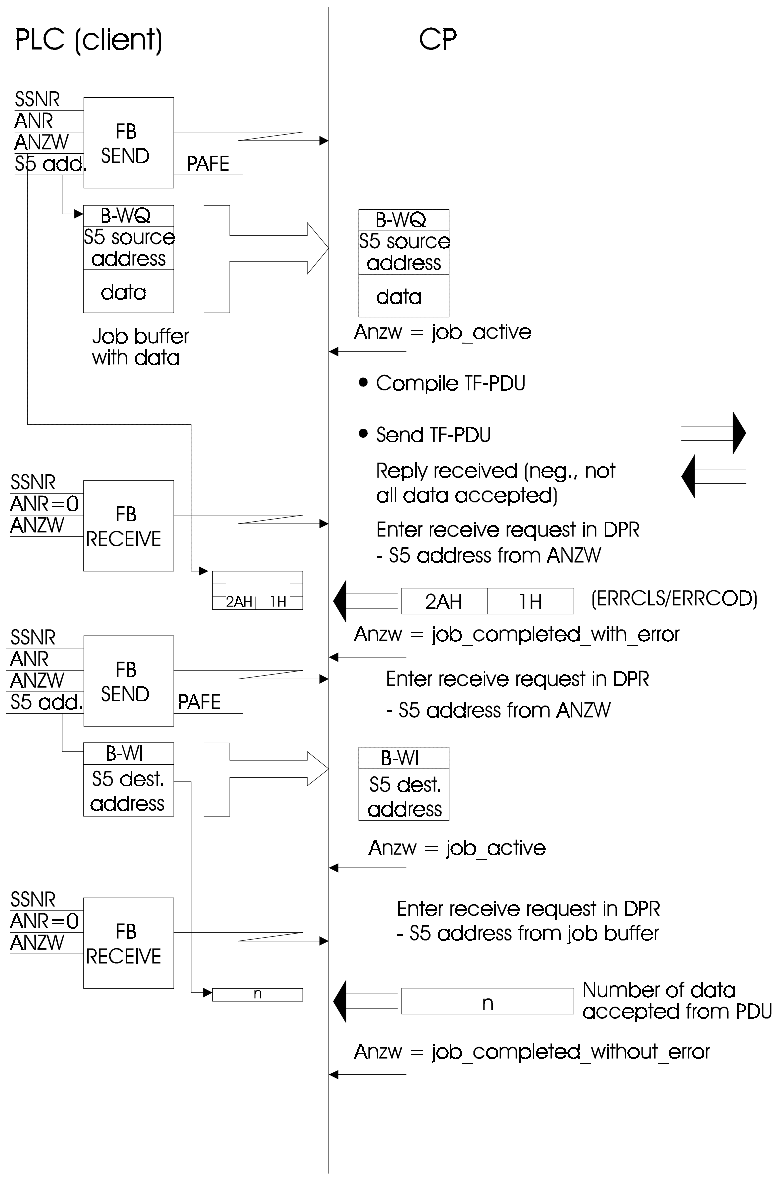


Fig. 10.7 Write Byte String with Ack. if Error Occurs, (Example: Source ID = DA)

10.4 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

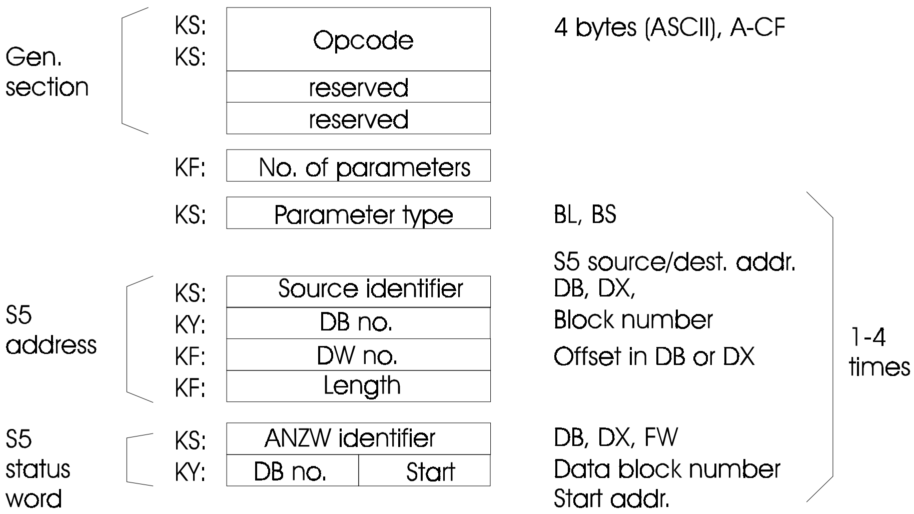


Fig. 10.8 Structure of the Job Buffer "Configure ANZW"

Call description

General section

Opcode:	A-CF

Job-related section

Number of parameters:	1 word, format:	KF
	Possible values:	1..4
	Meaning: up to four parameters can be transferred simultaneously.	
Parameter type:	1 word, format:	KS
	Possible 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/dest identifier:	1 word, format:	KS
	Possible values:	DB, DX
	Meaning: source/destination address identifier for configuring the address for the read/write byte string services.	
DB no.:	1 word, format:	KY
	Possible values:	high byte: 0 low byte: 1..255
	Meaning: data block number for the parameter types "BL", "BS".	
DW no:	1 word, format:	KF
	Possible values:	0..2042
Length:	1 word, format:	KF
	Possible 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	
	Possible values:	FW, DB, DX	
	Meaning: status word type		
	ANZW specification		
	Format:	KY	
	Possible values:	high byte:	block number
		low byte:	DW number, FW number

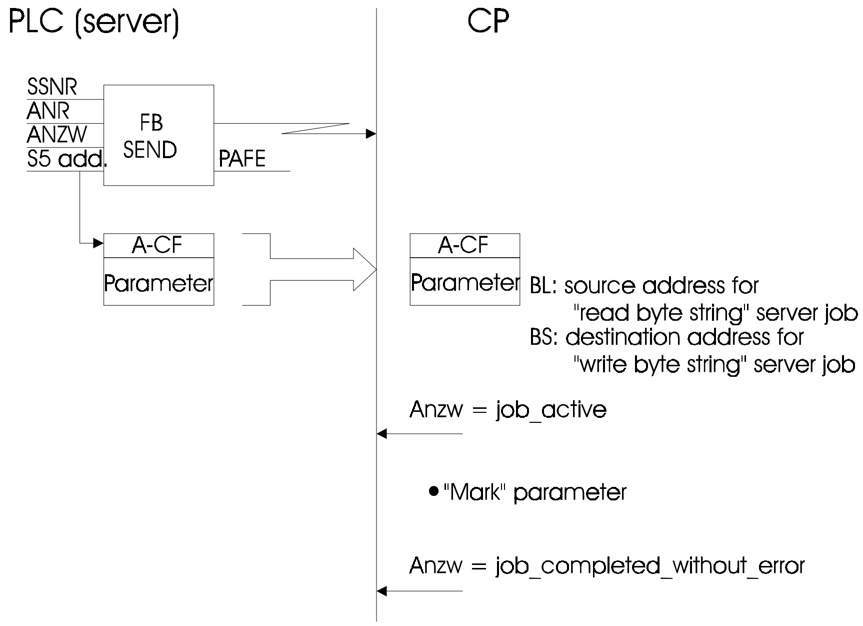


Fig. 10.9 Sequence: "Configure ANZW" (Server) for "Read/Write Byte String"

10.5 Transparent Data Exchange (Client)

A data area is transferred to the partner with the implicit request to send a reply with data.

"Transparent Data Exchange" Job Buffer

Gen. section	KS:	Opcode		4 bytes (ASCII), T-DQ
	KS:			T-DO
	KF:	Timeout		Monitoring time in 0.1 sec
		reserved		
S5 address	KS:	Source identifier		S5 source address DB, DX, DA
	KY:	0	DB no.	Block number
	KF:	DW no.		Offset in DB or DX
	KF:	Length		
S5 address	KS:	Dest. identifier		DB, DX
	KY:	0	DB no.	Data block number
	KF:	DW no..		Data word number
	KF:	Length		
		Data (if DA specified)		

Fig. 10.10 Structure of the Job Buffer "Transparent Data Exchange"

Call Description

General section

Opcode: T-DQ ("transparent data exchange with acknowledgment")
or
Opcode = T-DO ("transparent data exchange without acknowledgment")

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 acknowledgment 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 - 13.

S5 source address

Source ID: 1 word, format: KS
Possible 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:	1 word, format:	KY
	Possible values:	high byte: 0, low byte: 1..255
	Invalid for source ID DA	
DW number:	1 word, format:	KF
	Possible values:	0..2042
	Invalid for source ID DA	
Length:	1 word, format:	KF
	Possible values:	1..2043
	Meaning: length of the data block area to be transferred with the service.	

S5 destination address

Dest. identifier:	1 word, format:	KS
	Possible values:	DB, DX

Meaning: Address at which the data in the acknowledgment will be stored. This address can also be invalid (DB no. = 0). In this case, no data can be expected in the acknowledgment.

DB number:	1 word, format:	KY
	Possible values:	high byte: 0, low byte: 0..255
DW number:	1 word, format:	KF
	Possible values:	0..2042
Length:	1 word, format:	KF
	Possible values:	..2043, -1

Meaning: length of the data block area in which the data in the acknowledgment will be stored; the value -1 indicates that all the data in the acknowledgment from the DW number to the end of the data block can be accepted.

Sequence of "transparent data exchange"

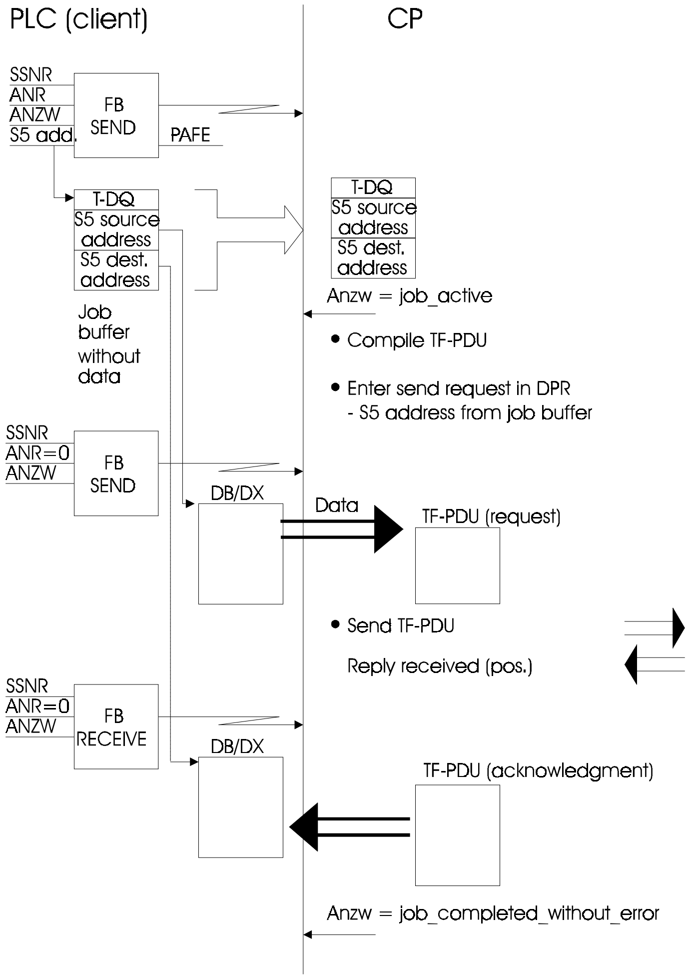


Fig. 10.11 Transparent Data Exchange with Source ID "DB" or "DX"

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 acknowledgment request (T-DO), or if the acknowledgment 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.

10.6 Transparent Data Exchange (Server)

For the non-open TF 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:

non-open TF services

Opcode
0B00: transparent data exchange without acknowledgment
0B01 transparent data exchange with acknowledgment

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.

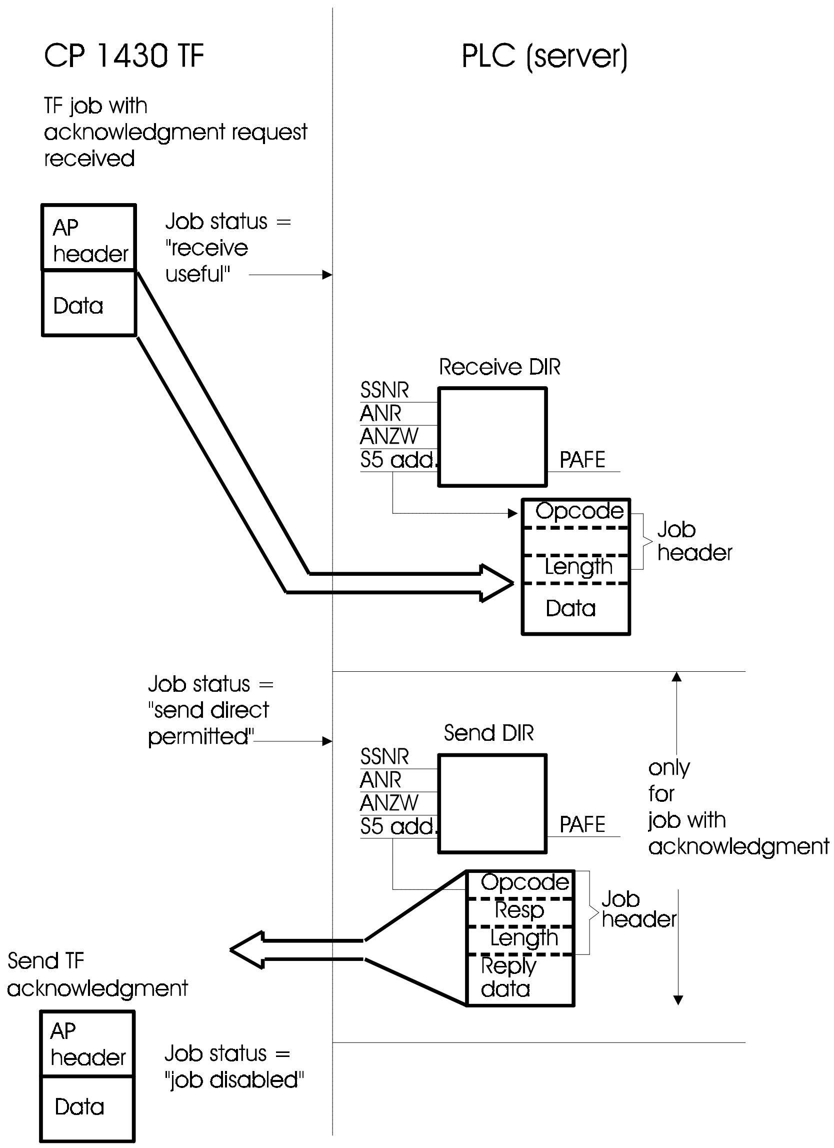


Fig. 10.12 Sequence "General Server Interface"

-

If the job does not require an acknowledgment, the status word is set to "job disabled" on completion of the receive job.

For a job with an acknowledgment, the PLC program must generate a reply that is transferred to the CP with a send direct call. The sending of the acknowledgment by the PLC program is not time-monitored by the CP. The first three words of the PLC acknowledgment 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 acknowledgment, 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 acknowledgment 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 acknowledgment).

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.❑

10.7 Addendum to Transparent Data Exchange

10.7.1 Status Word of the TRADA on the Server

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.

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

10.7.2 Example of a Program for Evaluating the Bits of the ANWZ with TRADA

Coordination of the status word corresponds to the normal status word evaluation. 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                :       connection 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      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.

❑

Notes

IV Appendix

Notes

A Example Programs

A.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 1430 TF communications processors

➢ 2 programmers (e.g. PG 730/PG 750)

The following software packages are also required:

➢ NCM COM 1430 TF

➢ PG software for the STEP 5 programming language,

➢ handling blocks for your PLCs.

➢ the example files supplied with COM 1430 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!**

A.2 Example 1: Using Variable Services

A.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 the data and using the variable services:

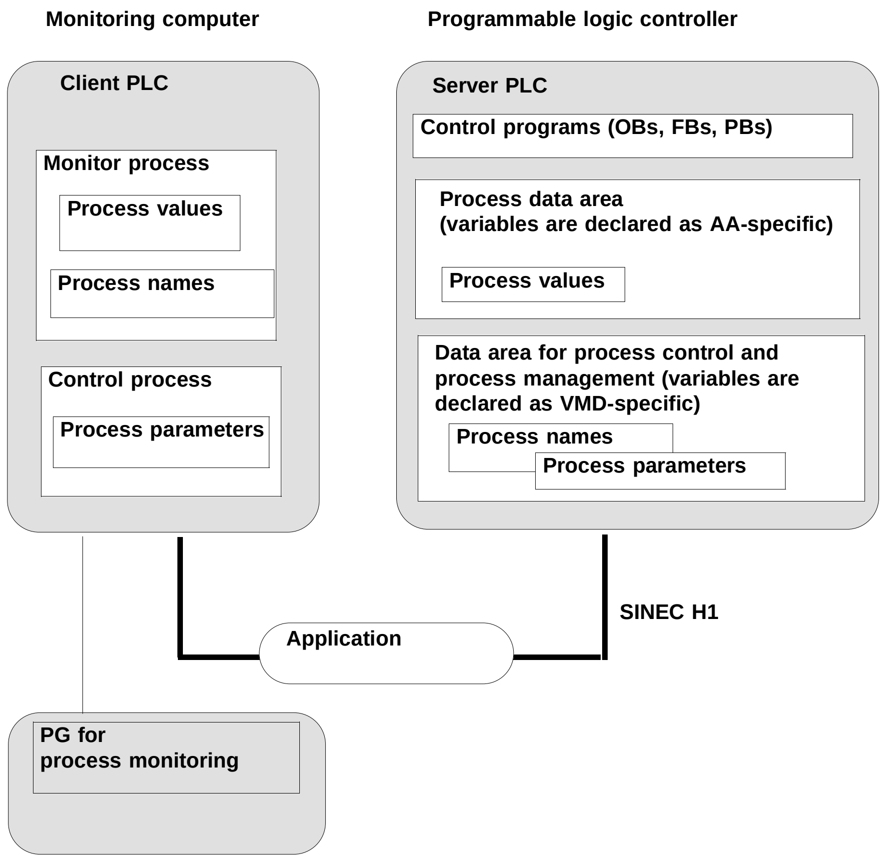


Fig. A.1: Example Configuration for Stage 1 - Variable Services

Tasks of the Client PLC and TF Services Used:

- ➢ Read process name (TF service: read variable)
- ➢ Write process parameters (TF service: write variable)

Task of the server PLC and TF Services Used:

- ➢ Report process values (TF service: information report)

A.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 application association-specific, so that the values can be only accessed via one application association.

- ➢ Process parameters
  The following parameters are stored to control the process:
  Rate 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.

A.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 1430.
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 1430.
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 1430 when defining the application association 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 "application association-specific" scope of validity, they must be configured in COM 1430 as application association-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 application association 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.

A.2.4 Creating the Client Configuration File

Start COM 1430 as described in the Section 'Introduction to the Configuration Software NCM/COM 1430'. 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 **File | Select** function to specify the configuration environment. Make the following entries or accept the proposed values if they are suitable in the **basic Settings** dialog:

CP type: CP1430

Status: OFFLINE FD

Database file: ABSPL.CLT

Enter your selections with the F7 key.

CP Basic Configuration - CP Basic Initialization Dialog

M 2-1

The next step is to create the SYSID block

✓ Select the **Edit | CP Init** function. In the dialog 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.

✓ Make the following entries:

➢ To address the PLC on SINEC H1:

MAC address: 08000601B010

➢ Select productive communication on interface 0 with the following settings:

Base SSNR:	0
Interface communication:	P for SSNR-OFFSET 0

➢ The "Firmware version" field is only a display field.

➢ To identify the PLC in the system, select a suitable text, for example:

Plant designation	Testsystem

Enter the current date in the "Date created" field (free format).

✓ Complete your entries with F7 (OK). The file ABSPL1.CLT is then set up on the hard disk. This completes input of the data specific to the CP 1430 and you now only need to assign parameters to the connection block.

✓ Answer the prompt about overwriting the module file with YES (if it appears). The basic configuration data are now saved on the hard disk.

✓ Now select the **Edit | Connections | Transport Connections** function to configure the connection block for the transport connection.

Configuring the Application Association

M2-4-4.1

You now specify the application association and its assignment to the transport connection

✓ Make your entries according to the section of dialog shown below.

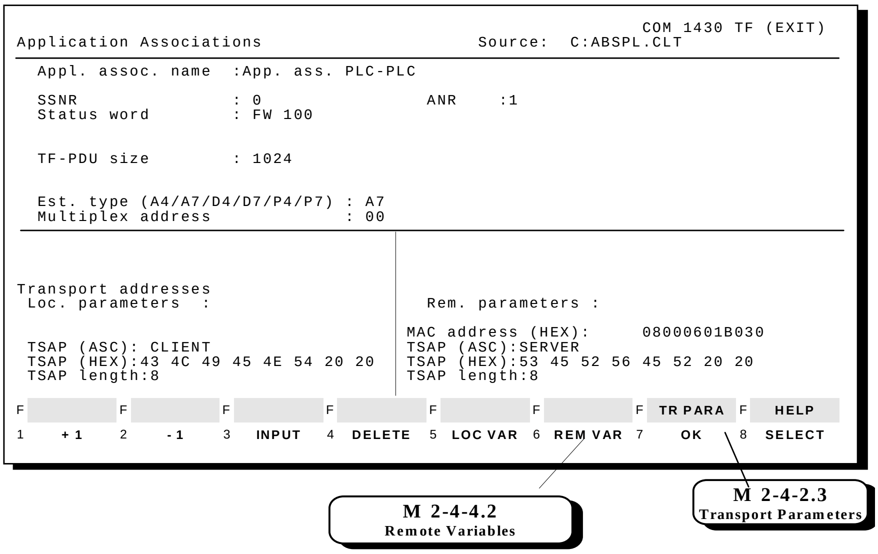


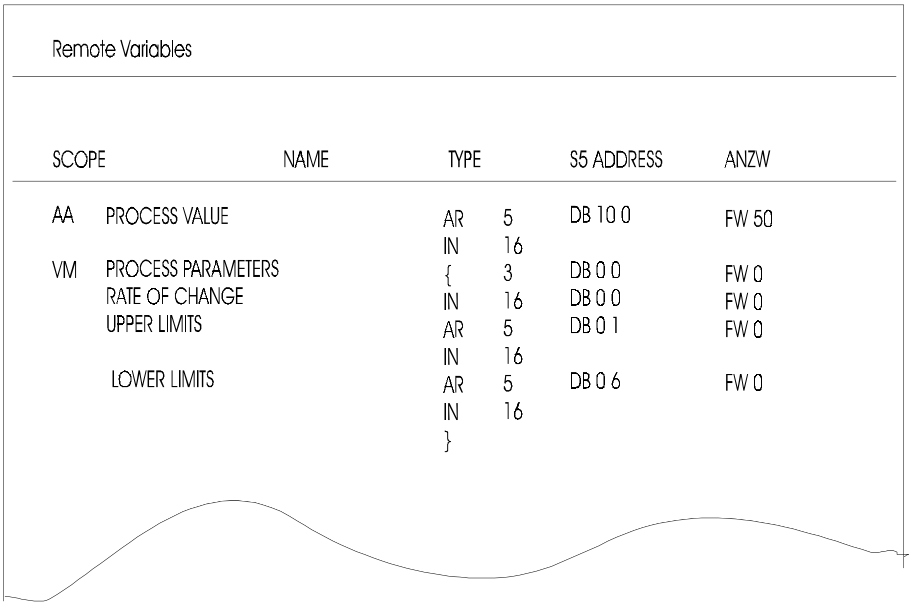
Fig. A.2: Application Association Definition

✓ Now press the F6 key (REM VAR) to call the dialog for configuring remote variables.

Configuring the application association-specific variables

M2-4-4.3 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 dialog shown below.



Remote Variables

SCOPE	NAME	TYPE		S5 ADDRESS	ANZW
AA	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. A.3: Remote Definition

✓ Complete your entries with F7 (OK). You return to the application association configuration dialog.

✓ Complete your input by pressing F7 (OK). Confirm the prompt about entering the data with YES.

A.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 "ABSPL.SRV".

The MAC address is 08000601B030.

For this connection, a local application association-specific object will be defined in the next step (PROCESS VALUE).

Following this, the VMD-specific variable structure PROCESS PARAMETERS must be configured.

M2-4-4.1

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 dialog shown below.

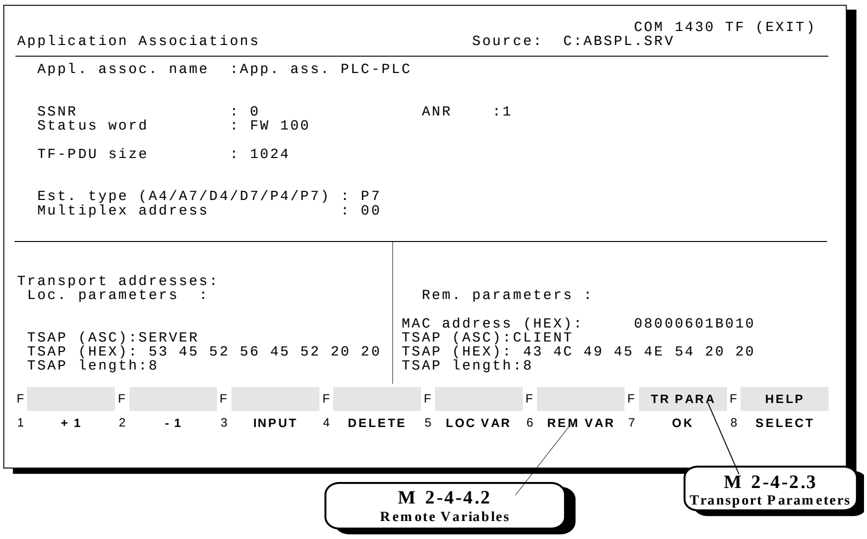


Fig. A.4: Application Association Definition (Server)

✓ Now press F5 (LOC VAR) to call the dialog for configuring local variables.

M2-4-2.3

Configuring application Association-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 dialog shown below; first for PROCESS VALUE.

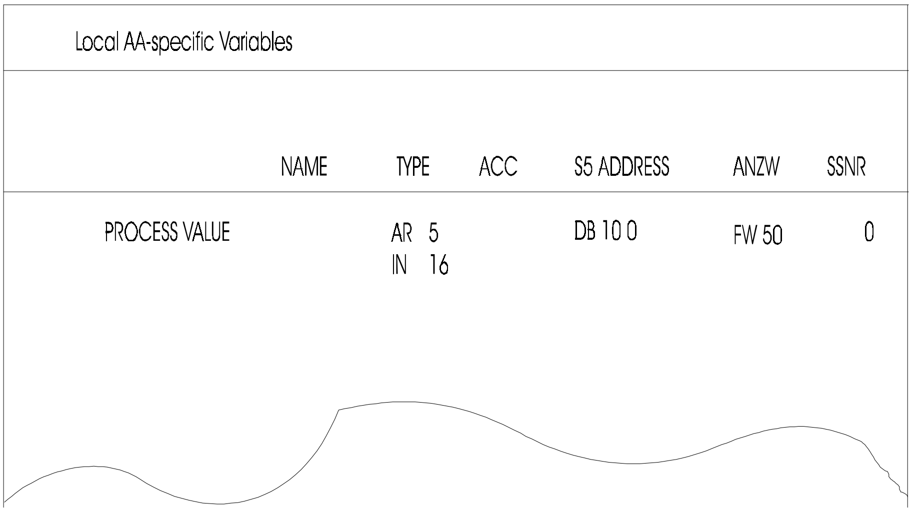


Fig. A.5: Local Variable Definition

✓ Complete your input with F7 (OK). You return to the application association configuration dialog.

✓ Complete your input with F7 (OK). Confirm the prompt about entering data with YES.

Configuring VMD-specific variables

M2-5.1 You now define the structure PROCESS PARAMETERS. This is a VMD-specific variable.

✓ Select the **Edit | VMD Variables Editor** function.

✓ Complete the dialog as shown below..

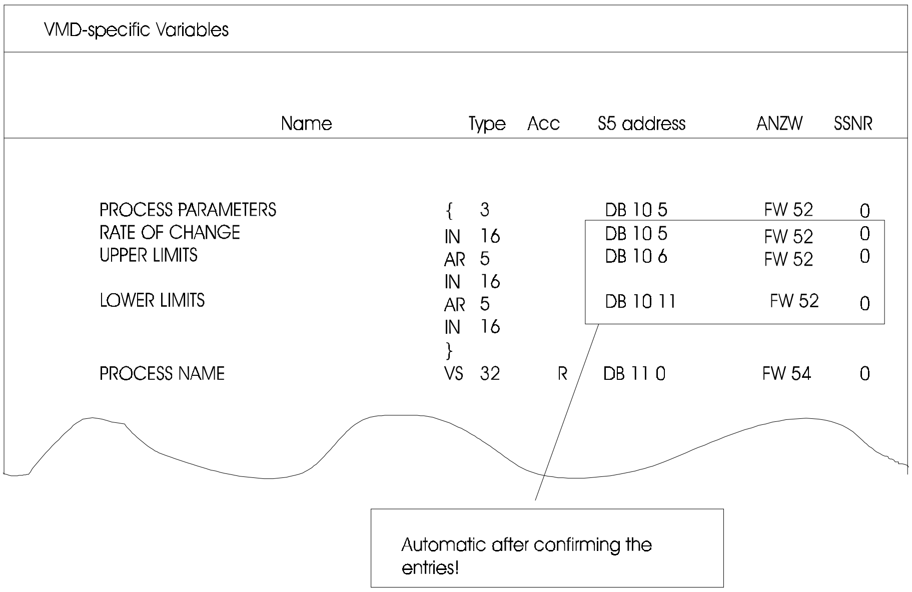


Fig. A.6: Local Variable Definition - Process Parameters

✓ After inputting the data, complete the entry by pressing F7 (OK). Confirm the prompt about entering the data with YES.

A.2.6 Creating the Job Buffers with the Request Editor

Client PLC

✓ To create the job buffers, you must start the Request Editor with **Request Editor | Select.**

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.

✓ Complete the dialog, as shown below:

Request Editor Initialization ...

PROGRAM FILE	**C:**	**BSPC@@ST.S5D**
BLOCK	**DB 20**	

Fig. A.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 **Request Editor | Create Job Buffer** function key F3 - NEW.

The following selection list is then displayed:

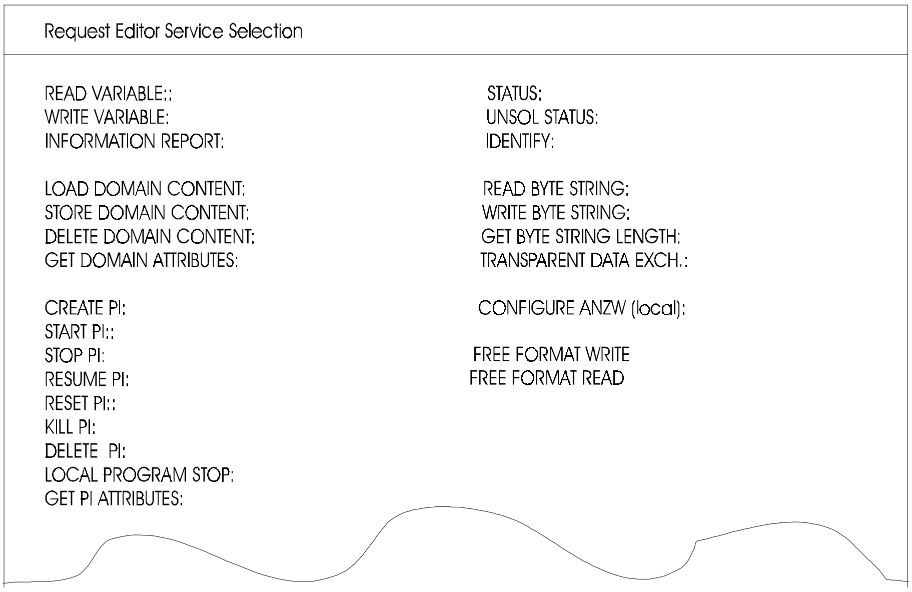


Fig. A.8: Request Editor Service Selection

✓ Select the job type ’read variable’. The appropriate dialog is then displayed.

Job Buffer for the TF Service 'Read Variable'

✓ Enter the variables as shown in the section of dialog below for the variable PROCESS NAME.

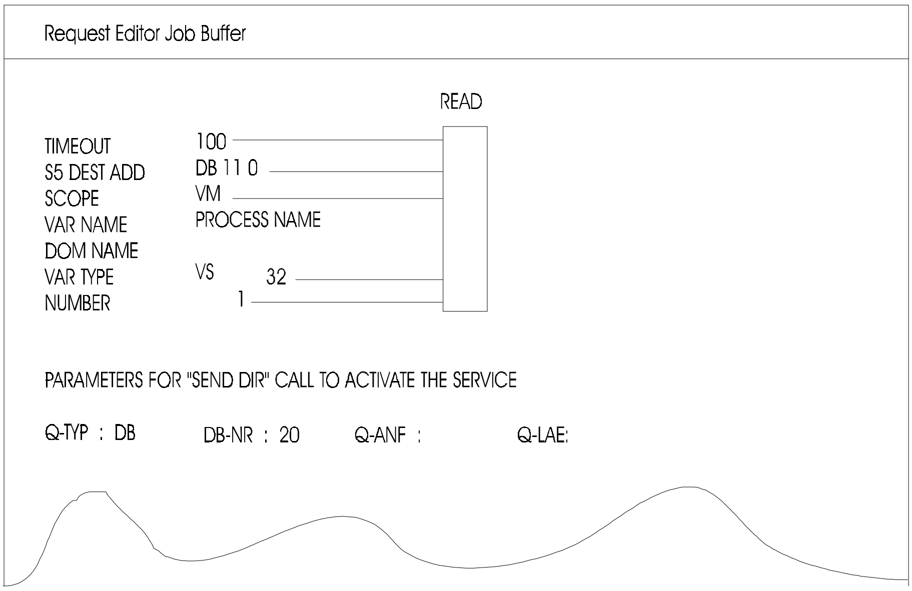


Fig. A.9: Job Buffer- Read Variable

✓ 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 dialog shown below for the variable PROCESS PARAMETERS.

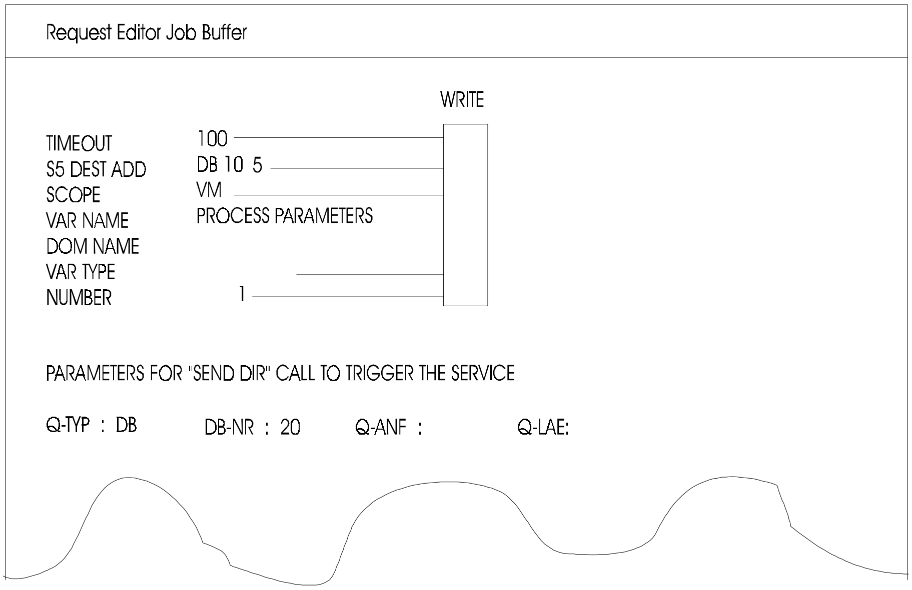
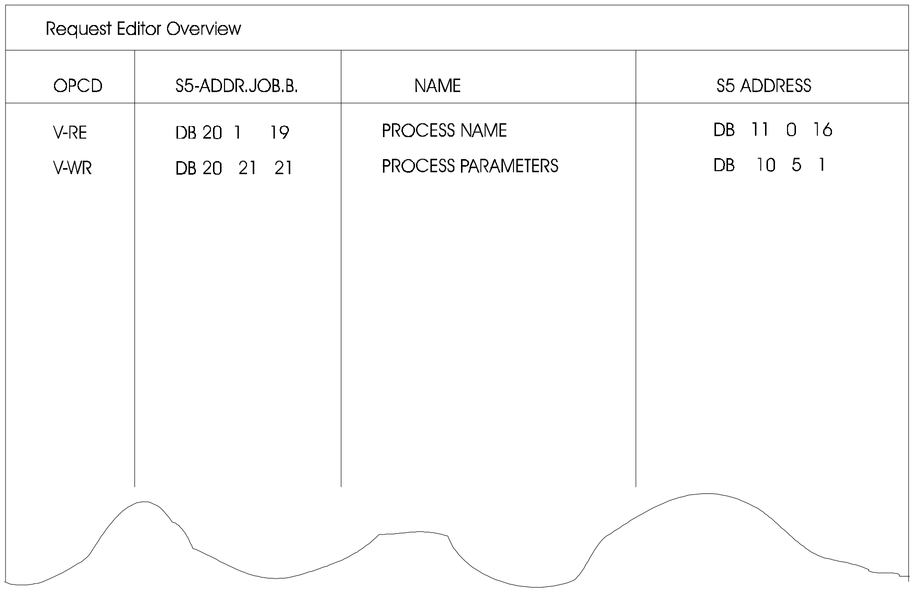


Fig. A.10: Job Buffer - Write Variable

Printout of the Job Buffer

✓ Once you have created the job buffers, you can use the menu function **Req.Editor | Documentation | Overview** to print out the job buffers and addresses.


Request Editor Overview

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 1

Fig. A.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.

✓ Enter the data as shown below.

Request Editor Setup ...

PROGRAM FILE	**C:**	**BSP1S@@ST.S5D**
BLOCK	**DB 20**	

Fig. A.12: Request Editor - Initialization of Server PLC

Job Buffer for the TF Service 'Information Report'

✓ Enter the values according to the section of dialog shown below for the variable PROCESS VALUE.

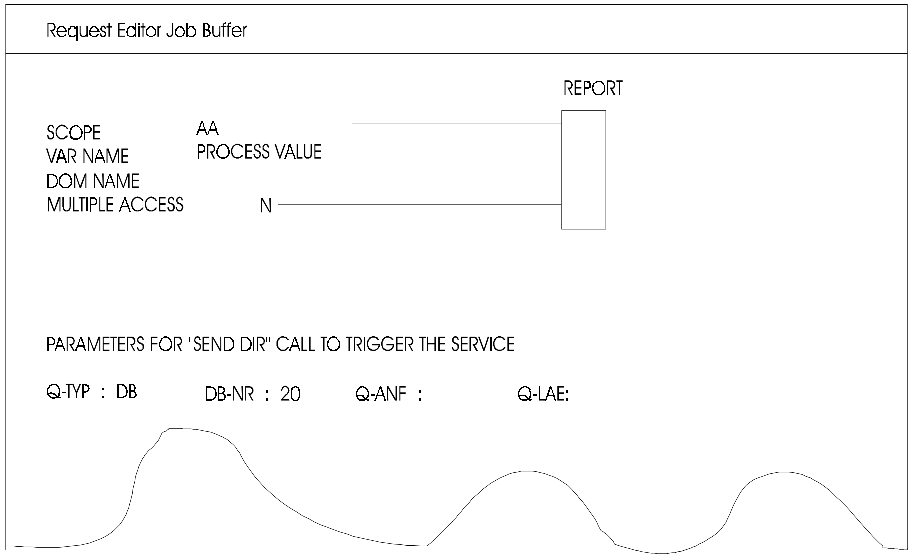


Fig. A.13: Job Buffer - Information Report

Printout of the Job Buffer

Once you have created the job buffers, you can use the menu function **Req.Editor | Documentation | Overview** to print out the job buffers and addresses.

Request Editor Overview

OPCD	S5-ADDR.BOX.B.	NAME	S5-ADDRESS
V-IN	DB 20 1 12	PROCESS VALUE	PROJECT

Fig. A.14: Job Buffer - Printout

A.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 example programs were written for an S5-135U, If you use other PLC types (S5-115U and S5-155U/ CPU 946/47), you must copy the appropriate handling blocks (HDBs) into the program files and adapt the HDB calls.**

☞ **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)

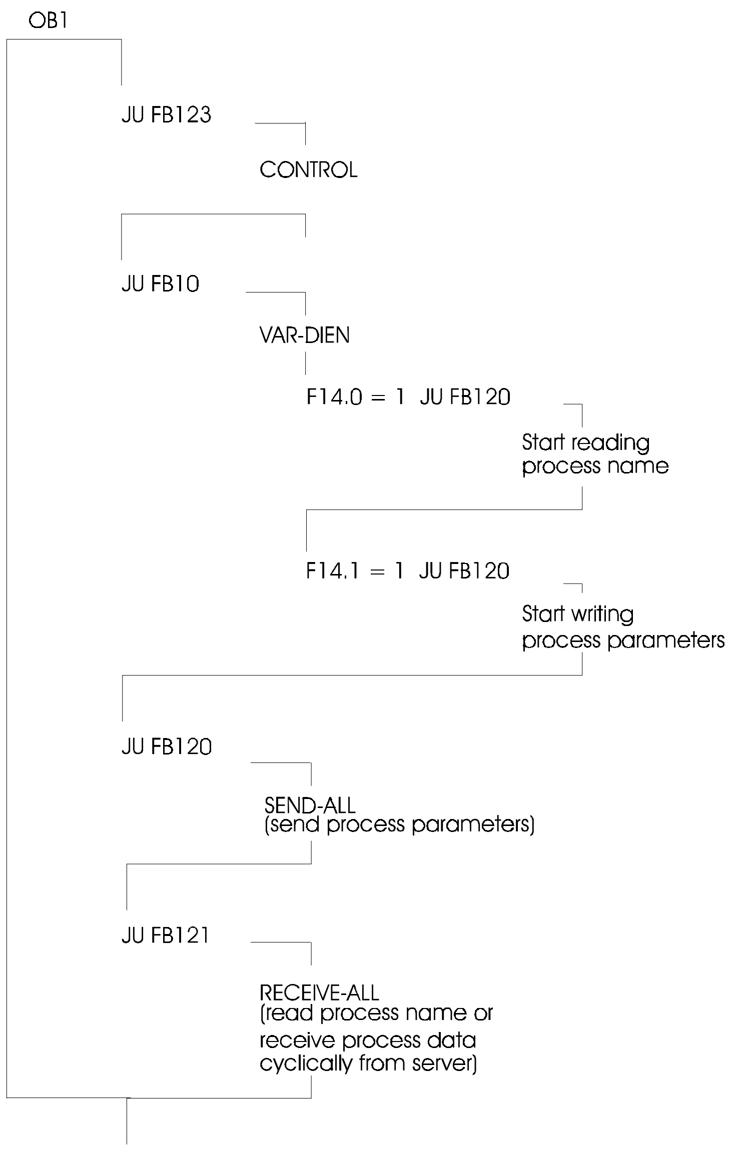


Fig. A.15: Program Structure of the PLC Programs on the Client PLC

Server PLC (programmable logic controller)

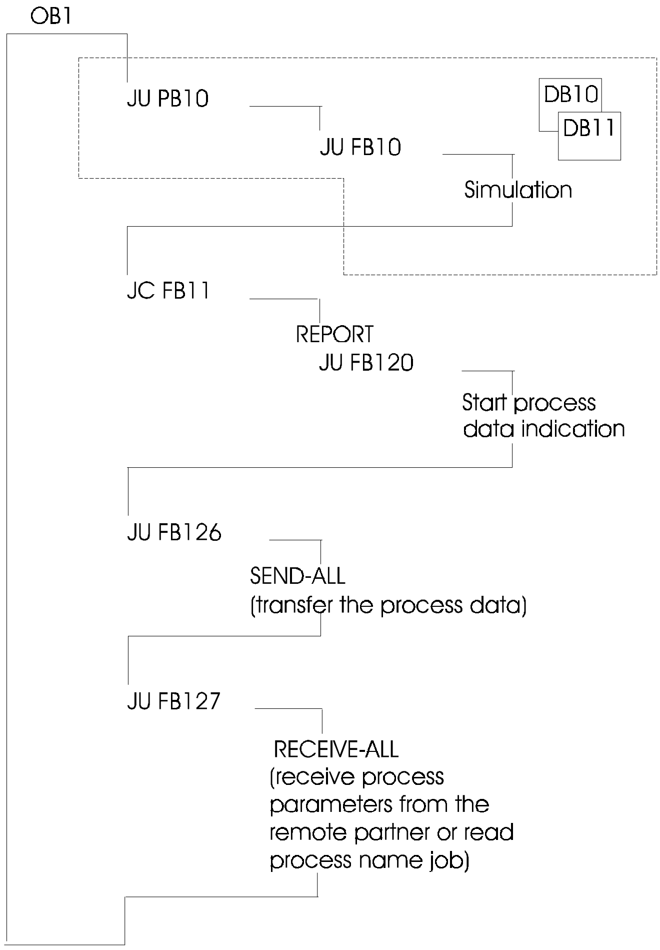


Fig. A.16: Program Structure of the PLC Programs on the Server PLC

PLC program on the client

```
DB10        B:BSPC@@ST.S5D                                     LEN=   21  /4
                                                                 PAGE   1
  0:        KF  =   + 00018;         Process value          1
  1:        KF  =   - 00020;               \=\              2
  2:        KF  =   + 00015;               \=\              3
  3:        KF  =   + 00002;               \=\              4
  4:        KF  =   - 00041;               \=\              5
  5:        KF  =   + 00050;         Update  factor
  6:        KF  =   + 00050;         Process upper limit    1
  7:        KF  =   + 00060;               \=\              2
  8:        KF  =   + 00070;               \=\              3
  9:        KF  =   + 00080;               \=\              4
 10:        KF  =   + 00090;               \=\              5
 11:        KF  =   + 00000;         Process lower limit    1
 12:        KF  =   - 00020;               \=\              2
 13:        KF  =   + 00000;               \=\              3
 14:        KF  =   - 00040;               \=\              4
 15:        KF  =   - 00050;               \=\              5
 16:
```

```
DB11        B:BSPC@@ST.S5D                                     LEN=   21 /4
                                                                 PAGE   1
  0:        KS  =   '                           ';
 12:        KS  =   '                           ';
 16:
```

```
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:BSPC@@ST.S5D                                    LEN= 52
                                                                   PAGE 1
SEGMENT 1       0000
0000            :                         Check the PLC-PLC AA
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            :                         Transfer the job buffer for
0009            :                         receiving the process name
000A            :                         or for sending the process
000B            :                         parameters to the CP;
000C            :                         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
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        :                       Jump block that calls the
0001        :                       simulation blocks.
0002        :JU FB  10
0003 NAME   :SIMUL1
0004        :
0005        :BE
```

```
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:BSP1S@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
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 connection
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

```
OB 20          B:BSP1S@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
```

OB 21

```
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

```
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
```

A.2.8 Starting Up

To start up, the configuration data must be transferred to the PLCs. Use the following functions:

➢ **Transfer | FD->CP** for the files
ABSPL.CLT -> client PLC and ABSPL.SRV -> server PLC

➢ The load function under S5DOS-KOMI to transfer the PLC data base created with the REQUEST-EDITOR.

A.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 application association (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

A.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.

A.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 PG Load on the programmer and can be activated in various dialogs. A PG is also connected to the system via a SINEC-H1 connection and used as the host computer in the example.

A.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 PG Load package in various dialogs. The PG connected to the system via the SINEC H1 connection is used to control the PLC simulation with the PG Load functions.

Overview of the device configuration

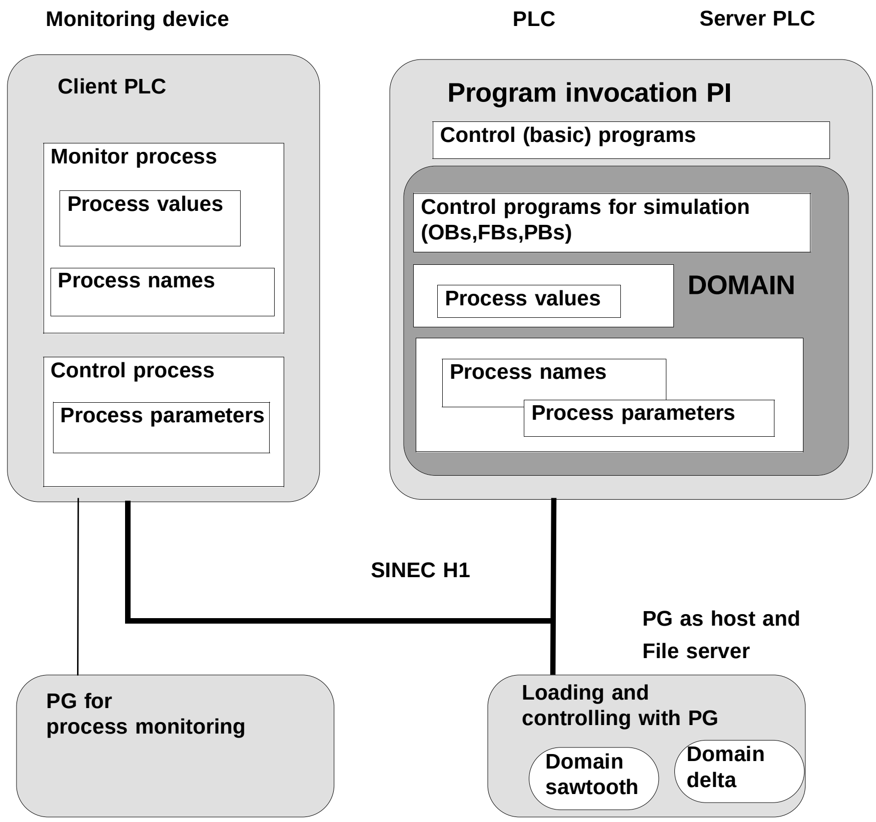


Fig. A.17: Device Configuration in Example 2

A.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 (acknowledgment).

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 1430 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.

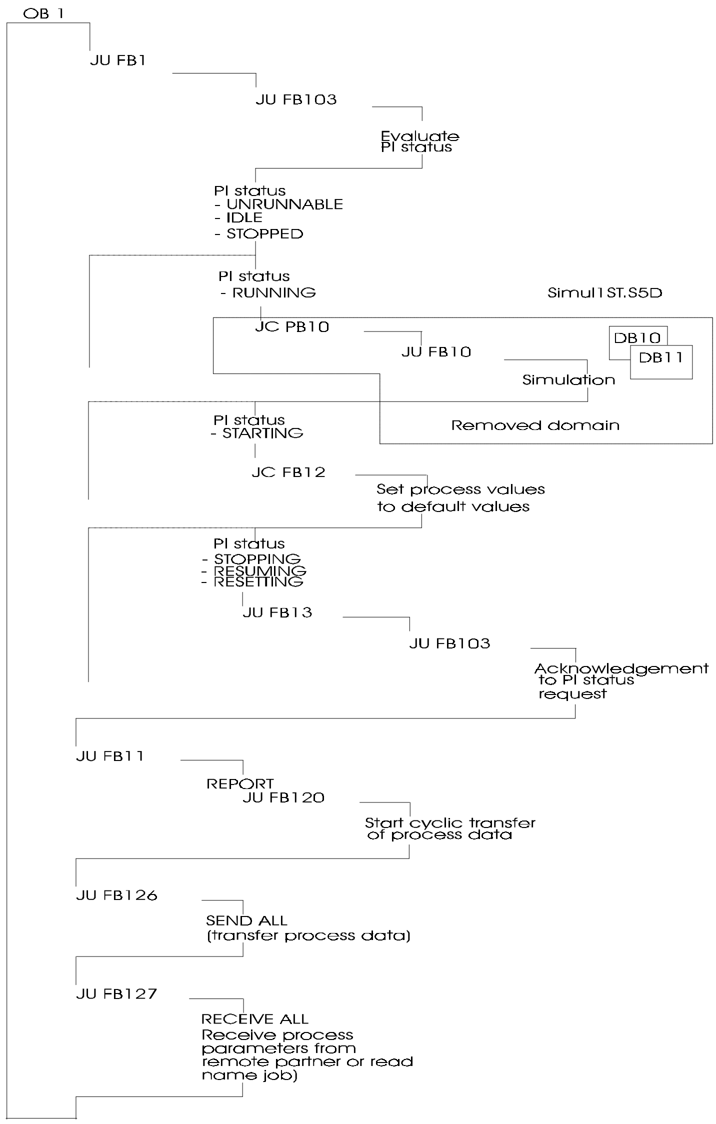


Fig. A.18 Taking the Domains out of the Original Program File of the Server PLC

```
DB20        B:BSP2S@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:

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        :                          See also:
0009        :                          TF instructions for
000A        :                          S5; 1.2.4.3 function block
000B        :                          "PI-ZUSTD" (FB103)
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 acknowledgment
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 acknowledgment
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 acknowledgment
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            :                           There is an acknowledgment
0043            :                           to the PI status scan
0044            :JC FB  13
0045 NAME       :ACKNOWL
0046            :
0047 ENDE :BE
```

```
FB 11           B:BSP2S@ST.S5D                                   LEN=  23
                                                                   PAGE   1
SEGMENT 1        0000
NAME            :INDIC
0005            :                           Send "INDICATE" job buffer
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
```

```
FB 12           B:BSP2S@ST.S5D                                   LEN=24
                                                                   PAGE   1
SEGMENT 1        0000
 NAME            :STARTING
0005            :                           Set process values to default
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
```

```
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          :                          Acknowledge PI status
0009          :JU FB 103
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          :                          There is no evaluation
0002          :                          of the PI statuses
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          :                     Jump block that caused the
0001          :                     simulation blocks
0002          :JU FB  10
0003 NAME     :SIMUL2
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
```

A.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 1430 and PLC-CPU must be established ("swing cable").

✓ The "PG Load" package is started on the host PG equipped with a SINEC H1 interface module

✓ In the initialization dialog form, you specify the name of a file containing the information about the PG connection to the required PLC (in this case "SERVERCP.LOD").

✓ After selecting the function PG Load | PLC Links, you can input the MAC address of the server PLC. The additional name of the application association allows you to display the currently selected PLC application association in the PG Load package.

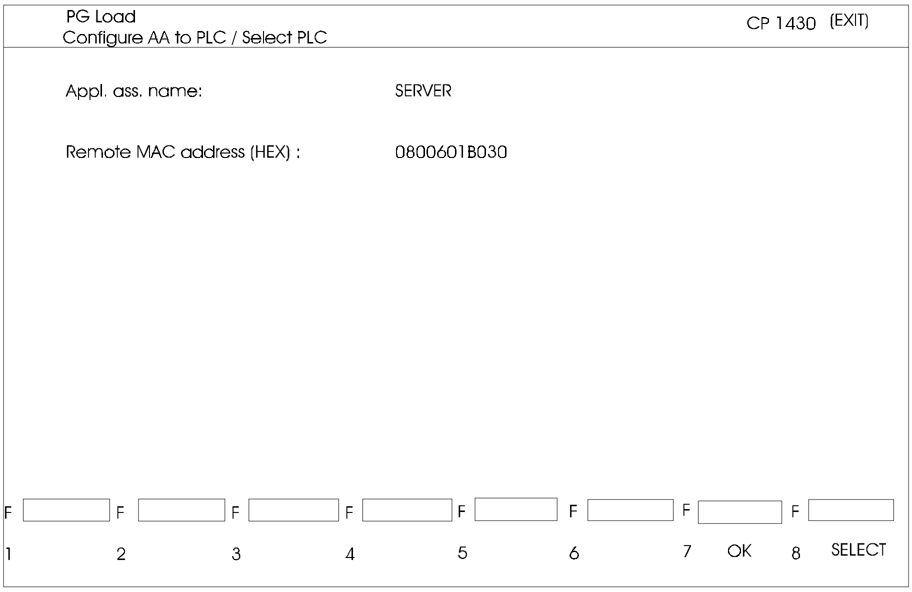


Fig. A.19: PG Load Configure AA to PLC / Select

After storing the information with "OK" (F7) and pressing F1, you return to the basic dialog.

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 **Transfer | PG Load | Transfer Functions**.

✓ In the next dialog (Transfer Functions) , you enter the name of the local S5-DOS program file (SIMUL1ST.S5D) from which the load file is generated with "CREATE" (F4)

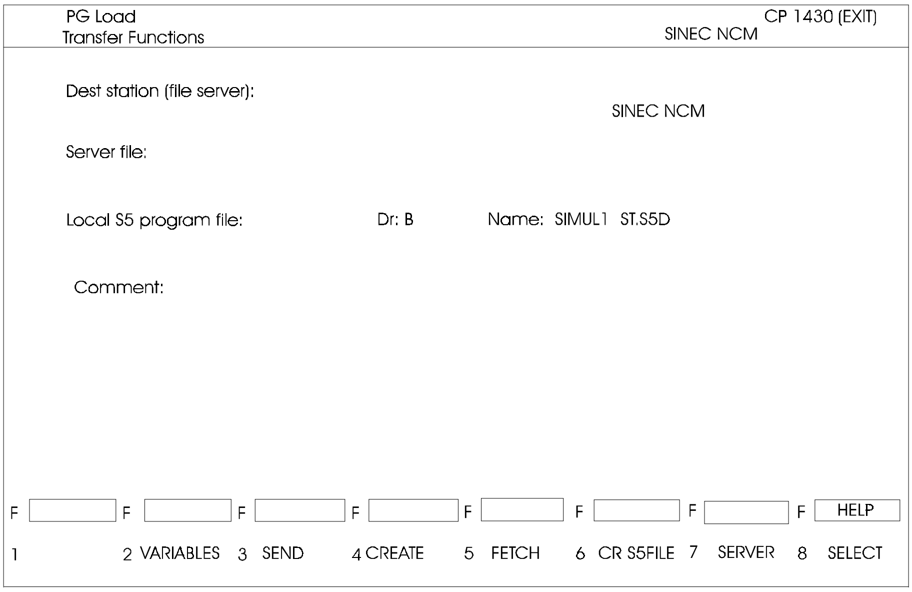


Fig. A.20: PG Load Transfer Functions

✓ 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 dialog, you return to the main menu.

Loading Domains

✓ Now select the function **Utilities | PG Load | Host Functions**.

✓ By pressing LOAD in the currently displayed dialog, call the dialog PG Load Load PLC.

First, the file "SIMUL1ST.S5D" should be loaded as a domain in the server station.

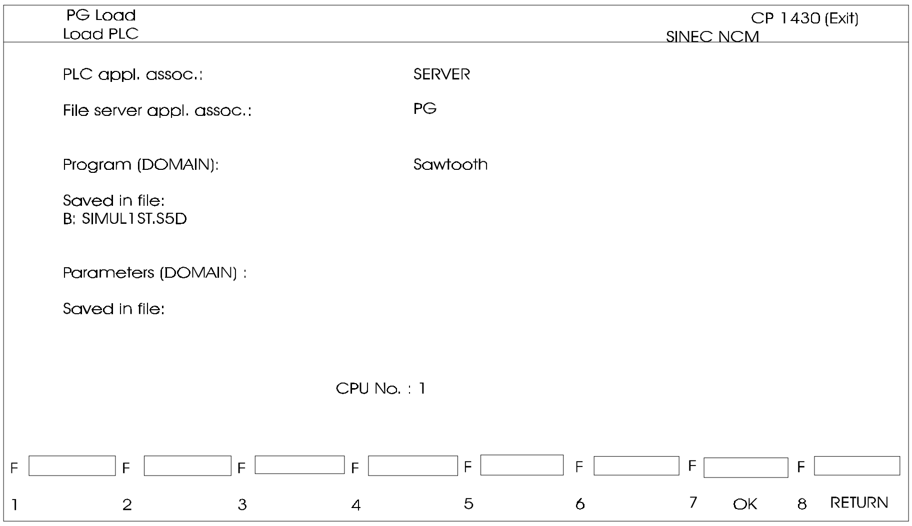


Fig. A.21: PG Load /Load PLC

✓ Enter the following parameters:

File server appl. assoc.:
"PG" as the server application association: 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".

Saved in file
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.

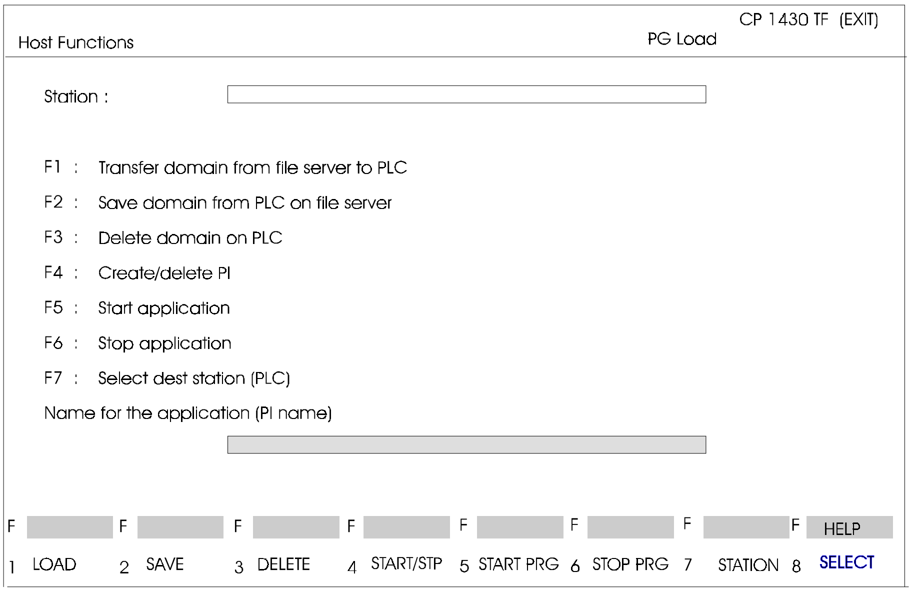


Fig. A.22 Input Dialog - Host Functions

✓ 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 dialog form and return to the dialog PG Load Host Functions:

Controlling the System

With this dialog 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 dialog (PG Load Host Functions), the function keys have the following effects:

F3 DELETE	The domains on the PLC are deleted.
F4 START/STP	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 START PRG	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 STOP PRG	

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	Using this key, you can select the next connection in the connection file (SERVER CP.LOD). Since there is only one connection 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.

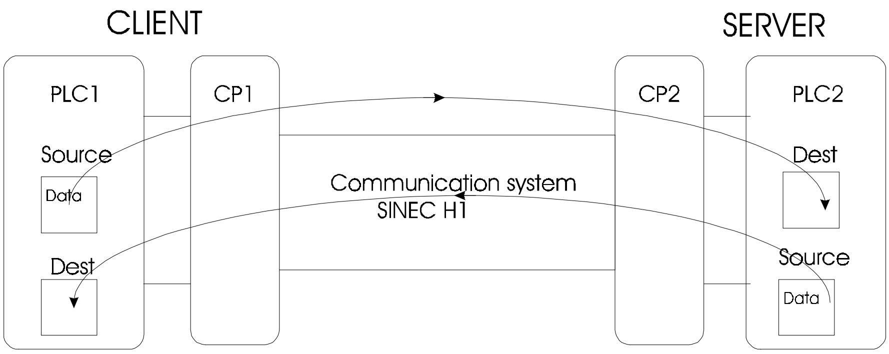


Fig. A.23: Transparent Data Exchange with Acknowledgment

A.4 Example 3: Transparent Data Exchange with Acknowledgment (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 acknowledgment.

Using the configuration tool
The following example illustrates the configuration steps required.

To configure this service, you require the two software packages COM 1430, TF editor and "LAD, CSF, STL" which are started on the PG 685 or PG 675 under S5-DOS. Within COM 1430 TF, the Request Editor is also used when creating job buffers.

COM 1430 TF
Using COM 1430, you assign the station address to the CPs and set up an application association (logical connection 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. 23 .

Configuration procedure

To configure the "transparent data exchange service", proceed as follows:

1. With the COM 1430 package

➢ Set up station address (6 bytes long MAC address) in both CP 1 and CP 2 if this does not yet exist. (Here you require the online mode)
Select: **Edit | CP Init** (CP in STOP state)

✓ Set up application association (logical connection layer 7a) from CP 1 to CP 2.
Select: **Edit-> Connections-> Appl. Associations**

Enter the following parameters in the dialog for the client (CP1):

– 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
– Multiplex address	(0)
– Remote TSAP	(SERV T-DQ)
– AA name	(only for information)
– Remote MAC address	(address of CP 2)

Enter the following parameters in the dialog for the server (CP2)

– 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)
– Multiplex address	(0)
– Remote TSAP	(CLIE T-DQ)
– AA name	(only for information)
– Remote MAC address	(address of CP 1)

2. With the Request Editor

Set up the job buffer T-DQ for the CLIENT (PLC 1) as follows:

✓ Load the Request Editor with **Req.Editor | Select**.

✓ Specify the program file in the initial dialog and select the data block that will contain the job buffers.

✓ Confirm your entries and select **Req.Editor | Create Job Buffer**.

✓ Select transparent data exchange and enter the following parameters:
- source address (data block)
- source length (length of the user data)
- dest. address (data block)
- dest. length (length of the user data)

✓ After you have input the data, press the F7 key (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 (S5DOS-KOMI)

➢ Follow the procedure below for PLC 1 (CLIENT) :

✓ Program OB 20, OB 21, OB 22, SYNCHRON blocks 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 connection block for CP 1.

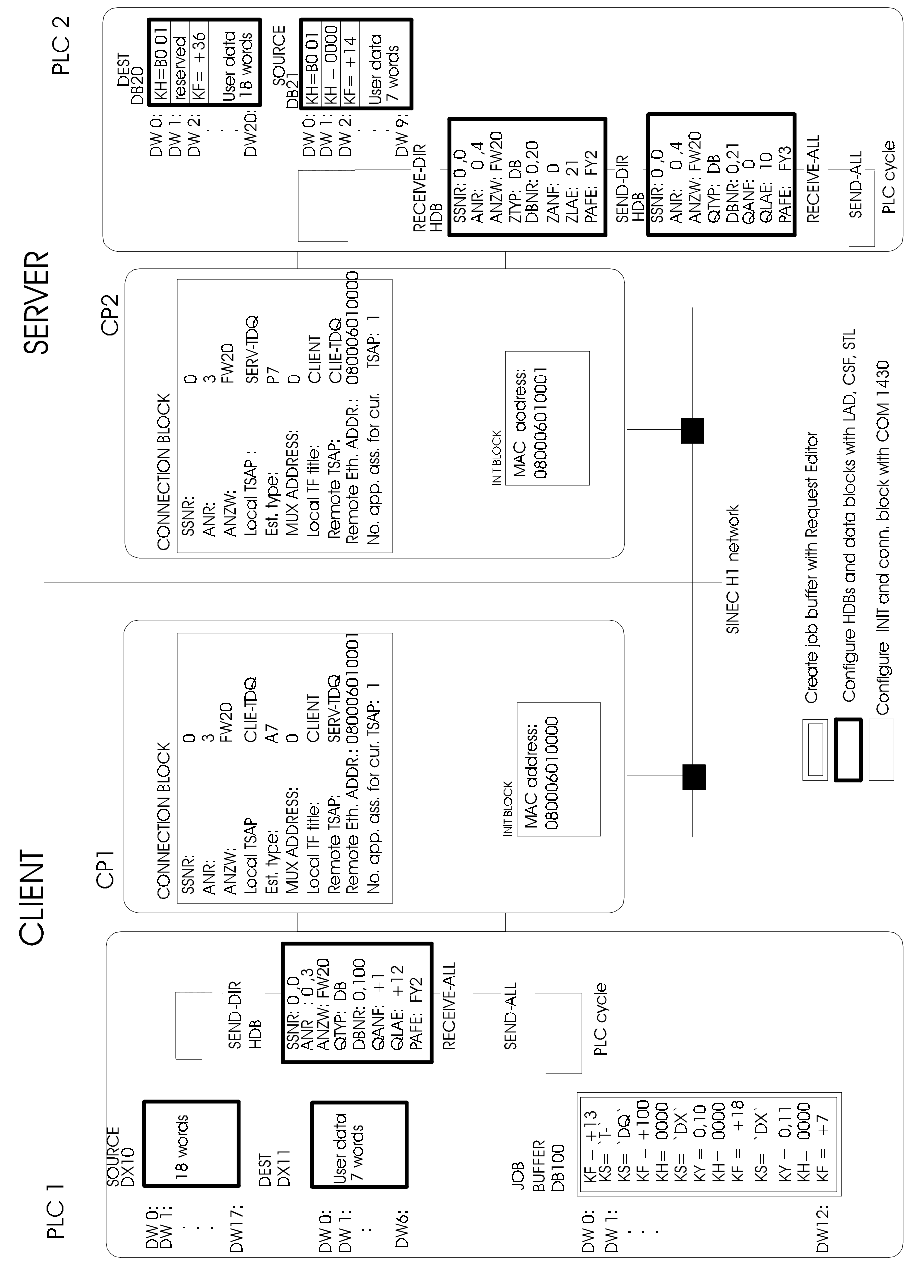


Fig. A.24: Transparent Data Exchange between two S5 PLCs

✓ Program RECEIVE ALL and SEND ALL that transfer the data between the CP and PLC.

☞ **The data block for the data source and for the data destination must exist.**

➢ Follow the procedure below for PLC 2 (SERVER):

✓ Program OB 20, OB 21, OB 22, SYNCHRON blocks 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 connection 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 acknowledgment to the CP. Select job number ANR+1 as selected in the connection 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 the RECEIVE-ALL and SEND-ALL HDBs that transfer data between the CP and PLC.

❑

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 1430. This information is necessary if you want to implement a connection to a non-SIMATIC system. Using the protocol implementation conformance statement, you can determine the following:

➢ which services the CP 1430 supports and

➢ at which degree of complexity the supported services are available.

The CP 1430 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: Services	CP 1430 TF
Serial transfer	
read byte string	X
write byte string	X
transparent data exchange	X
Time functions	
query time	X
set time	X
transfer time	X
Variable services	
read	X
write	X
information report	X
get variable attributes	S
Domain services	S
initiate up/download	S
up/download segment	S
terminate up/download	S
request upload sequence	S
upload segment	S
request upload sequence response	S
request download sequence	S
request upload sequence	S
load domain content	X
store domain content	X
delete domain content	X
get domain attributes	X

PICS 1: Services	CP 1430 TF
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

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 scopes 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.	

q

Notes

C TF Error Numbers used by the CP 1430

C.1 Preface

This appendix describes the error numbers and the causes of errors that can occur when operating the CP 1430. The error numbers consist of the parameters ERRCLS and ERRCOD of the AP protocol. There are two possible reasons for errors occurring:

➢ The CP 1430 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 1430 receives a TF indication (S5 as server) that is incorrect or cannot be executed. The CP 1430 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 1430 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 status word. 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.

Note:
In section A.2, the error numbers are arranged in service groups. Each service group is assigned an abbreviation to identify it. In section A.3, the error numbers are arranged in ascending order. The abbreviated identifier is also shown as a cross reference.

C.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 variable services
DOM	Errors in the domain services
PI	Errors in the program invocation services
SER	Errors in serial transfer
CONF	Errors in configuration

Error number	Service group	Meaning
00000_H	NONS	No error
00881_H	NONS	The job was aborted locally and not executed.
$009C2_H$	NONS	TF service not implemented or unknown.
01041_H	NONS	There are no resources available for the current job
02041_H	NONS	The monitoring time between issuing the request and the reply has elapsed
02843_H	NONS	The negotiated PDU length is too short for the current job
$029C2_H$	NONS	Protocol error: In the AP header or the received indication, the PDULG is not 22 + PARLG + DATLG.
$029C3_H$	NONS	Protocol error: 1. The PDU length is too short for the current job 2. DATLG does not match the length of data expected for the job.
$029C4_H$	NONS	Protocol error: the parameter length in the AP header of the indication is inconsistent with the lengths in the PDU.
$029C5_H$	GEN, VAR	The negotiated PDU length is too short for the reply.
02A01	SER	More data are to be read or written than permitted by the configured length or current length of the data block in the PLC.

Error number	Service group	Meaning
02A02	SER	Less data are to be read or written than permitted by the configured length.
02A04	SER	Access to the PLC is not possible since no S5 address is configured.
03001_H	NONS	The job is not long enough for all the parameters belonging to it.
03002_H	NONS	The job contains an illegal opcode Opcode.
03003_H	GEN, VAR, CONF	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
03004_H	CONF	The status word 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.
03005_H	VAR	The data are in the job buffer, and the job buffer is not long enough.
03006_H	VAR	The coding of the data type in the job buffer is incorrect.
03007_H	VAR	The coding of the scope in the job buffers is incorrect.
03008_H	VAR	The variable or group name in the job buffer is incorrect (e.g. too long).

Error number	Service group	Meaning
03009$_H$	PI,DOM	The domain name in the job buffer is incorrect (e.g. too long).
0300A$_H$	DOM	The file name in the job buffer is incorrect (e.g. too long).
0300B$_H$	DOM	The logical partner name in the job buffer is incorrect (e.g. too long).
0300C$_H$	PI	The program invocation name in the job buffer is incorrect (e.g. too long).
0300D$_H$	DOM	The capabilities length in the job buffer is incorrect (e.g. too long).
03010$_H$	VAR	The variable in the job buffer is not configured in the remote definitions .
03011$_H$	VAR	Nothing is configured in the remote definitions under the scope specified in the job buffer.
03012$_H$	VAR	The variable in the job buffer is not configured locally.
03013$_H$	VAR	Nothing is configured locally under the scope specified in the job buffer.
03014$_H$	VAR	Multiple variable access; the group specified in the job buffer does not exist or is empty.
03015$_H$	PI	The domain name does not exist in the job buffer.
03017$_H$	CONF	The job buffer contains an unknown configuration parameter (wrong coding).

Error number	Service group	Meaning
03018_H	GEN	The local extension in the job buffer is too long.
03019_H	VAR	The value of the "box identifier" specified in the job buffer does not correspond to the TF identifier for addressing via the free format address.
$0301F_H$	PI	The local stop (P-HL) is not permitted with this job number.
03021_H	VAR	Conversion error converting floating point from TF format: the TF number is larger than the range that can be represented in MC 5.
03022_H	VAR	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).
03023_H	VAR	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.
03024_H	VAR	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.
03025_H	SER	Error in the data transfer from the PLC to the CP (e.g. data block does not exist).
03026_H	GEN,PI,DOM ,SER	Error in the data transfer from the CP to the PLC (e.g. data block does not exist).

Error number	Service group	Meaning
03027_H	GEN,PI,DOM,SER	The data to be transferred to the PLC from the confirmation are longer than the data area made available on the PLC based on the job buffer entry.
03028	TRADA	A TRADA with acknowledgment was issued, the data length in the acknowledgment is however 0.
03030_H	VAR	Protocol: The parameter header of the received confirmation contains illegal values.
03032_H	VAR	Access to a variable is currently disabled (e.g. because it is being accessed via a different connection).
03037_H	VAR	Data type inconsistent: the data type received with the acknowledgment does not correspond to the data type expected for the variable.
03038_H	VAR	Protocol error in a received confirmation. The attributes belonging to the variables are inconsistent.
03039_H	VAR	The interface via which the variable is to be accessed is not synchronized.
$0303F_H$	VAR	Error in module.
03040	VAR	Access to a non-existent object or to an object configured under a different scope.
03041_H	VAR	1. Error in data transfer from CP to PLC (e.g. data block does not exist). 2. The data from the confirmation to be transferred to the PLC are longer than the data area specified in the job buffer.

Error number	Service group	Meaning
03042_H	VAR	Access to a variable is currently disabled (e.g. because it is being accessed via a different connection).
03043_H	VAR	The variable can only be read and not written to.
03044_H	VAR	Variable access via name: the variable does not exist (has not been configured).
03045_H	VAR	Variable access via address: the variable does not exist (has not been configured).
03046_H	VAR	The type of variable is not supported.
03047_H	VAR	Inconsistent data types: 1. The data type configured 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.
03048_H	VAR	Inconsistent attributes: 1. 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 free format address the scope is not VM or an illegal alternate access has been attempted.
03049_H	VAR	1. The type of variables access is not supported. 2. The interface via which the variable is to be accessed is not synchronized, or not configured. 3. The CPU number of the format-free address is greater than 4 (CPU number 4).

Error number	Service group	Meaning
03061_H	VAR	Conversion error converting floating point 32 from TF format: the TF number is larger than the range that can be represented in MC 5.
03062_H	VAR	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. 17:62 or 33.11.92)
03063_H	VAR	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.
03064_H	VAR	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.
03068_H	VAR	Conversion error converting Boolean from MC 5 format to TF format.
03069_H	VAR	Conversion error converting Boolean from TF format to MC 5 format.
$0306A_H$	VAR	Conversion error converting visible string from MC 5 format to TF format.
$0306E_H$	VAR	Conversion error converting visible string from TF format to MC 5 format.
$051C1_H$	NONS	TF service is not permitted in the current CP mode (STOP, ASYNCHRON).
$061C1_H$	NONS	Protocol error: in the header of the received indication, ROSCTR has an illegal value.

Error number	Service group	Meaning
061C2$_H$	NONS	Protocol error: in the header of the received indication, MODFR1 has an illegal value.
061C3$_H$	NONS	Protocol error: in the header of the received indication, MODFR2 has an illegal value (not 0).
069C1$_H$	NONS	Protocol error: in the header of the received indication, MPXADR has an illegal value.
069C2$_H$	NONS	Protocol error: in the header of the received indication, PROTID has an illegal value.
069C3$_H$	NONS	Protocol error: in the header of the received indication, COMCLS has a value that is not supported.
069C4$_H$	NONS	Protocol error: in the header of the received indication, COMCOD has a value that is not supported.
069C5$_H$	NONS	Protocol error: in the header of the received indication, TACTID has an illegal value (not 0) or has a value that is not supported.
069C6$_H$	NONS	Protocol error: in the header of the received indication, TASQNR has an illegal value (not 0).
069C7$_H$	NONS	The received indication was not negotiated when the application association was initiated.
08000$_H$	APPL	Negative acknowledgment of conclude application association.
08001$_H$	APPL	Initiating an application association not possible at present.

Error number	Service group	Meaning
08100_H	APPL	Application association already initiated when initiate application association job issued..
08102_H	DOM	1. The application association to the file server was aborted. 2. The connection on which the load job was requested has broken down.
08201_H	GEN, VAR	The start object specified in the indication is unknown or does not exist.
08205_H	PI,DOM	PI: Attempt to create a program invocation when a program invocation already exists. DOM: Load request: a domain with this name already exists.
08206_H	CONF	Protocol error: The entries for class or scope are illegal.
08210_H	DOM	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).
08240	VAR	Access to a non-existent object or to an object configured under a different scope.
08241_H	VAR	1. Error in data transfer from CP to PLC (e.g. data block does not exist). 2. The data from the confirmation to be transferred to the PLC are longer than the data area specified in the job buffer.

Error number	Service group	Meaning
08242_H	VAR	Access to a variable is currently disabled (e.g. because it is being accessed via a different connection).
08243_H	VAR	The variable cannot be accessed. The following reasons are possible: 1. The variable was configured as ‘read only’. Write access is rejected. 2. Bit 7 is set in the variable status word. Neither write nor read access is possible.
08244_H	VAR	Variable access using name: The variable does not exist, or is not configured.
08245_H	VAR	Variable access using address: The variable does not exist, or is not configured.
08246_H	VAR	The type of variable is not supported.
08247_H	VAR	Inconsistent data types: 1. The data type configured 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.
08248_H	VAR	Inconsistent attributes: 1. 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 free format address the scope is not VM or an illegal alternate access has been attempted.

Error number	Service group	Meaning
08249_H	VAR	1. The type of variables access is not supported. 2. The interface via which the variable is to be accessed is not synchronized, or not configured. 3. The CPU number of the format-free address is greater than 4 (CPU number 4).
08261_H	VAR	Conversion error converting floating point 32 from TF format: the TF number is larger than the range that can be represented in MC 5.
08262_H	VAR	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. 17:62 or 33.11.92)
08263_H	VAR	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.
08264_H	VAR	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.
08268_H	VAR	Conversion error converting Boolean from MC 5 format to TF format.
08269_H	VAR	Conversion error converting Boolean from TF format to MC 5 format.
$0826A_H$	VAR	Conversion error converting visible string from MC 5 format to TF format.
$0826E_H$	VAR	Conversion error converting visible string from TF format to MC 5 format.

Error number	Service group	Meaning
08300_H	PI	The PLC type is not supported (e.g. S5-100).
08301_H	DOM	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).
08304_H	PI,DOM	PI: 1. Error in serial transfer via the 511 interface to the PLC. 2. The serial interface is not plugged into the PLC-CPU configured as master. DOM: There is not enough space left in the background memory to store the file. 1. The serial 511 connection to the PLC cannot be switched through. 2. Physical error accessing the PLC via the serial connection. 3. Logical or protocol error accessing the PLC via the serial connection.
08305_H	GEN	Start resource unknown.
08310_H	PI,DOM	The syntax of the file name is not correct.
08311_H	DOM	Reading and writing the domain not possible.
08400_H	DOM	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.

Error number	Service group	Meaning
08402_H	PI	The service is not permissible in the current program invocation status (e.g. starting in the "unrunnable" status)
08405_H	PI,DOM	Access to the program invocation is not possible at present since it is already occupied by a different connection. Access to the domain is not possible at present since the module is already being accessed.
8700	PI,DOM	1. Interruption or communication error between CP and CPU (e.g. no swing cable). 2. When loading a domain, attempt to load a block that already exists on the PLC. 3. Not enough memory for variables or type definitions when loading. 4. The variablen block to be loaded is incorrect (e.g. format conversion 143->1430 not performed).
08701_H	GEN	Access to extended object class is not supported.
08702_H	GEN,PI, DOM	Access to list of domain-specific variables: the domain is unknown.
08703_H	PI,DOM	The domain must not be deleted.
08710_H	DOM	The indication does not contain a domain name.
08800_H	APPL	1. A local extension is not supported when initiating an application association. 2. The version number during initiation is 0.
08801_H	APPL	Conclude application association not possible, acknowledgments still expected.

Error number	Service group	Meaning
08810_H	APPL	1. Initiating an application association, the number of syntax is 0. 2. No syntax accepted when initiating an application association.
08811_H	DOM	
09041_H	NONS, VAR	Protocol error: 1. PARLG is 0 in the header, so that there is no service ID. 2. The contents of COMCLS, COMCOD and ROSCTR in the AP header do not match the service ID.
09044_H	NONS	TF service is unknown.
09045_H	NONS	TF service not negotiated when initiating the application association.
09046_H	APPL	Establishment already active when initiate application association job issued.
$091C1_H$	NONS	Protocol error: in the header of the received indication, SGSQNR has an illegal value (not 0).
$0A041_H$	NONS	The application association is being terminated so this request will no longer be processed.
$0A042_H$	NONS	The application association is no longer established.

❑

Notes

D Abbreviations

A

ANR	Job number (for handling blocks)
ANZW	Status word
AP	Automation protocol layers 5 to 7 of the ISO/OSI reference model
AS 511	511 interface, protocol for the communication between PLC and PG
ASCII	American Standard Code of Information Interchange

B

B	Block; unit of a CP database; e.g. connection block
BCD	Binary coded decimal
BE	Block end

C

CC	Central controller
CIM	Computer Integrated Manufacturing
COM	Abbreviation for programming software for SIMATIC S5 CPs
COR	Coordination module
CP	Communications Processor
CPU	Central Processing Unit
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

DPRAM 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

F

F Flag bit

FB Function block

FD Floppy Disk (data medium)

FD	Flag double word
FDDI	Fiber Distributed Data Interface
FO	Fibre Optic
FW	Flag word
FY	Flag byte

G

GRAPH 5	Software package for planning and programming sequence controllers

H

HDB	Handling blocks

I

IB	Input byte
IEC	International Electronics Commission
IEEE	Institution of Electrical and Electronic Engineers
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
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
MSB	Most significant bit

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)
PDU	Protocol Data Unit (frames consisting of PCI and SDU)
PG	Programming device

PI	Program invocation
PI	Process image
PII	Process image of the inputs
PIQ	Process image of the outputs
PLC	Programmable controller
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)

S

SA	Source Address
SAA	System Application Architecture
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
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 TF	SINEC technological functions
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

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

VMD Virtual Manufacturing Device

❑

Notes

E Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

❑

F 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

/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

/4/ Guidelines for installing the SINEC H1 bus system
SIEMENS AG, Order no.: AR 463-220

/5/ Guidelines for installing theSINEC H1FO bus system
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 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 package: HDB software + description
Order no.: 6 ES 5842-7-CB 01 for CPU 928A/B - CPU 948

For S5-155 can be ordered as package: HDB software + description
Order no.: 6 ES 5846-7-CA 01 for CPU 946 / 947

/8/ Manual for SINEC H1 Triaxial Networks
Siemens AG, Order no.:6GK1 970-1AA20-0AA1 Release 03

/9/ SINEC H1FO Network Manual
Siemens AG, Order no.: HIR: 943 320-011

❑

G Compatibility with the CP 143 TF / NCM COM 143 TF

Topics in this Chapter

The CP 1430 TF is designed so that it is largely compatible with the CP 143 TF module. This means the following:

➢ Applications written for the CP 143 can continue to be used without modification with the CP 1430.

➢ The CP 1430 provides improved performance and configuration is simpler with the NCM COM 1430 TF configuration tool.

➢ Databases created with NCM COM 143 can be converted easily with the converter supplied with NCM COM 1430 TF.

The following sections provide detailed information about the improvements and changes.

G.1 The CP 143/1430CP

G.1.1 Structure and Functions of the Module

DIL Switch and Jumpers

In contrast to the previous communications processors, the CP 535 and CP 143 TF, no DIL or jumper settings are necessary with the CP 1430 TF.

Automatic Detection of the Medium

The selected type of attachment, SINEC H!/H1FO or industrial twisted pair is recognized automatically by the CP 1430 TF.

Memory Expansion Memory Cards

In contrast to the EPROMs used with the CP 143, S5 memory cards are inserted into the CP 1430 TF.

Parameter Limits

The CP 1430 TF is available as a basic and an extended version. Compared with the basic version and the CP 143 TF, the extended version has different parameter limits in terms of transport connections and application associations. For more detailed information, refer to Volume 1, Chapter 4 of the manual.

G.1.2 Maximum 2 CPs Required for Backplane Bus Communication in the Multiprocessor Mode

CP 143 TF: 3 CPs are required for 4 CPUs

Using the CP 143, three CPs were required for this configuration. Backplane communication was only possible using interface numbers 232 and 236. An extra CP was necessary to handle the productive communication.

CP 1430 TF: 2 CPs are enough for 4 CPUs

The backplane bus communication and the productive communication in multiprocessor operation can be handled by two CP 1430 modules. This is achieved using the additional base interface number 244 that is reserved for backplane communication of up to 4 CPUs.

A higher number of CPs is only necessary when several H1 bus segments are being used.

G.1.3 Other Changes

Accuracy of the Hardware Clock

The accuracy of the integrated hardware clock of the extended version of the CP 1430 TF is 1 ms compared with 10 ms for the CP 143 TF.

'Node Initialization' Still Exists After Deleting

After the database has been deleted, the CP 1430 TF still retains the previously loaded/configured initialization data. After a restart, the CP changes to the RUN state and can also be reached using bus selection and the MAC address. On the CP 143 TF, a node initialization was necessary whenever the database was deleted.

Type Check

With variables of the type visible string (VS), the CP checks that the values of the bytes are valid. The range of validity is ASCII and corresponds fully to the range that can be represented with the S5 format KS. Values outside this range cause conversion errors (TF error 826A/826B or 306A/306B).

Checking the Value and Converting Time Variables

Any values can be specified for the variable types TI (time of day) and TD (time and date) (for example values higher than 23 for hours and values greater than 59 for seconds and minutes).

If the value exceeds the range, the program attempts a carry, incrementing the next higher unit. If no carry is possible, an error message is generated (TF error 3062).

On the CP 143, time variable values are not converted and must be specified correctly.

System PI and System Domain

The system PI and domain PI are connected. In contrast to the CP 143 TF, this is indicated when the PI/domain attributes are read.

To allow a user PI to use the system domain, its parameter ’multiple use’ was set to TRUE.

G.2 NCM COM 143/1430 TF

G.2.1 Configuring Several Jobs on One Transport Connection

Several Jobs per TSAP

Depending on the mode of the transport connection, up to 4 jobs can be assigned per TSAP.

On a full duplex connection, for example, one SEND and one RECEIVE job can be assigned.

NCM COM 143 TF

In NCM COM 143 TF, you can select the number of jobs in a follow-on dialog of the **Edit | S5-S5 Links** function. To configure further jobs, you then return to the basic dialog to configure the next jobs.

Simplified Procedure in NCM COM 1430 TF

In NCM COM 1430 TF, you can select the number of jobs in the Transport Connection dialog. You configure other jobs for the same TSAP in the same dialog.

G.2.2 Avoiding Inconsistencies: No Automatic Generation of TSAPs

CP 1430 TF

When configuring connections on the CP 1430 TF, you only need to specify the transport address (MAC address and TSAP).

CP 143 TF

With the CP 143, you also had to specify the job and interface number. This meant that inconsistencies were possible in the local and remote database files.

No Generation of TSAPs

It is, however, no longer possible for the program to propose an automatically generated, default remote TSAP during configuration of the CP 1430 TF.

To make editing easier, the values of the MAC address and the TSAPs from the last configuration are retained in the dialog when you create a new job.

G.2.3 Configuring Multicast Groups

Definition

The Multicast mode, allows connectionless transmission of single frames to all the stations with the selected multicast address and reception of frames from partners sending with this multicast address.

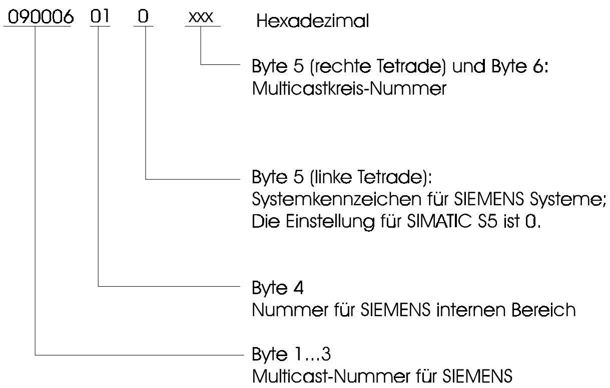
Stations with the same multicast address are known as a multicast group.

Configuring with NCM COM 143 TF

Multicast groups were defined explicitly in NCM COM 143 by specifying a multicast group number. This multicast group was used locally to form the MAC address.

Configuring with NCM COM 1430 TF

Multicast groups are now defined using the menu option **Edit | Connections | Datagram Services**. No additional multicast group numbers are assigned. A station is assigned to a multicast group by the entry in its MAC address, as follows:



G.2.4 Other Changes in NCM COM 1430 TF

The CP 143 Database can be Converted

The database of the CP 1430 is not compatible with that of CP 143. A converter is available with which CP 143 databases can be converted to the CP 1430 format.

No Password

There is no password on the CP 1430.

Transport Parameter Defaults in Configuration Dialogs

The default values are selected so that communication with CP 143 or CP 1413 TF modules configured with the default values of their COMs (NCM COM 143 or COML 1413 TF) is possible.

TF-PDU Size

Values between 128 and 65536 can be selected as the TF-PDU size.

Size of the Database

Functions are available for querying and adapting the size of the database.

Converting Domains

To allow use of the TF domain services, domains created with COM 143 TF can be converted to the format of COM 1430 TF.

Name of the Database File

The names of the database files created with COM 1430 TF begin with the letter A.

Variables Type Editor

You can create a library in the database for the TF variable types you require for your automation tasks.

NCM COM 1430 TF provides a TF variable type editor, with which you can define TF variable types. The library is saved in the CP block OB 14.

Test Functions

The error messages of COM 1430 TF differ from those of COM 143 TF.

The TF error messages are identical, however, several new ones have been added.

G.2.5 Terms

Previously used term	New term
S5-S5 link	Transport connection
Module (file)	Database (file)

❑

Notes

H Glossary

Application association
An application association is a connection for communication using TF services.

Application layer
The application layer is layer 7 in the ISO/OSI reference model for open systems interconnection. The task of the application layer is to provide uniform access to the services of the lower layers.

Backplane bus communication
Backplane bus communication allows Pg functions to be executed online on the path "PG - SINEC H1 - CP - parallel PLC backplane bus - CPU".

COM
Configuration software for SINEC CPs.

Configuration data
Parameters that can be set and loaded on the CP with the NCM COM 1430 configuration software and that control the way in which the CP operates.

CP
Communications processor (network interface card).

CP block
A CP block is a software block belonging to the CP database. CP blocks contain the configuration data required for a CP mode. CP blocks are managed in the CP database file on the PG. They can be loaded singly or with the entire CP database file (transfer functions) and can be copied (file functions).

CP database
The complete set of configuration data of the CP 1430 is known as the CP database. On the PG, the CP database is maintained in the database file.

CSMA/CD
Bus access technique complying with IEEE 802.3.

Datagram
A datagram is a data frame sent to

- ➢ one partner (datagram to a single address)
- ➢ several partners (multicast datagram)
- ➢ all partners (broadcast datagram)

without a connection being established in advance. On the CP 1430 TF, datagram jobs must be configured on the transport interface.

Datagram services
Datagram services allow a connectionless transmission of single frames to

- ➢ one partner (datagram to a single address)
- ➢ several partners (multicast datagram)
- ➢ all partners (broadcast datagram)

Database file
The CP database is managed in the database file on the PG.

Domain
This is a communications object consisting of a continuous memory area with a fixed length that can contain both data and program. Domains are used to supply devices with the required data and programs.

Domain services
Application service group providing services for uploading and downloading domains.

File server application association
Identifies an application association between a PLC and a file server on which PLC programs are stored.

File server application associations are configured with NCM COM 1430 TF.

Handling block (HDB)
HDBs are standard function blocks that allow the data exchange with modules capable of page addressing (dual-port RAM).

Job buffer
Job buffers are used in the TF services on the PLC to describe a communication service requested in the PLC program.

MAC address
Address to distinguish stations connected to a common medium (SINEC H1).

Medium access control
Controls to coordinate the access to a common transmission medium.

Memory Card
Simatic memory card for the CP 1430 complying with the PCMCIA specification.

NCM
SINEC management products.

PG Load
Tool belonging to the NCM COM 1430 TF configuration software for addressing and controlling PLCs via the TF interface.

Program invocation (PI)
Communication object with which a program in a programmable logic controller can be addressed.

Request editor
Tool belonging to the NCM COM 1430 TF configuration software for creating job buffers.

SINEC
Product name for networks and network components from Siemens.

SINEC TF
MMS-compatible application services in SINEC.

Station
A station is identified by a MAC address on SINEC H1.

TF interface
The TF interface is the access to the SINEC TF services of the application layer conforming with MMS. The TF interface presents itself to the control program in the form of handling blocks (HDBs).

TF variable type
Variable types are structure descriptions of variables that can be used as often as required. There are standard variable types such as INTEGER or BOOLEAN and self-defined variable types (structures).

In NCM COM 1430 TF, you can use the varaibale type editor to define the variable types you require.

Transport layer
The transport layer is layer 4 of the ISO/OSI refernce modue for open system interconnection. The task of the transport layer is the reliable transfer of data (raw information) from device to device. Both transport connections and connectionless services (datagram serivices) can be used.

Transport interface
The transport interface on the CP provides access to the connection-oriented and connectionless services of the transport layer. The transport interface presents itself to the control program in the form of handling blocks (HDBs).

Transport connection (in CP/COM 143 previously S5-S5 link)

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

Variable services
Application service group for transferring (reading or writing) variables.

Virtual manufacturing device (VMD)
A standardized image of a programmable logic controller in the form of a model. It is described by the objects it contains and the characteristics of the physical device. The practical use of such a model is that it allows a standardized interface to query the device status and device properties (VMD services).

VMD services
Standardized interface for querying device status and properties.❑