SIMATIC S5

S5-115F
Programmable Controller

Manual

Volume 2/2

EWA 4NEB 811 6149-02

STEP ® and SIMATIC ® are registered trademarks of Siemens AG.

Copyright © Siemens AG 1991

Subject to change without prior notice.

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

EWA 4NEB 811 6149-02

Preface

Introduction

Configuring with COM 115F 1

Programming

Introduction to STEP 5 2

STEP 5 Operations 3

Program Test 4

Error Diagnostics 5

Blocks 6

Application 7

Appendices A/B/C

Index

Contents

Page

Preface ix

Introduction xi

1 Configuring with COM 115F . . . . . 1 - 1

1.1 The Main Menu . . . . . 1 - 8
1.1.1 F1: Initialize Operating System . . . . . 1 - 8
1.1.2 F2: Setting Communications Parameters . . . . . 1 - 13
1.1.3 F3: I/O Configuration . . . . . 1 - 19
1.1.4 F4: PLC Functions . . . . . 1 - 20
1.1.5 F5: Diagnostics . . . . . 1 - 21
1.1.6 F6: Defaults . . . . . 1 - 22
1.1.7 F7: System Handling . . . . . 1 - 23

1.2 Configuring I/O Modules . . . . . 1 - 29
1.2.1 General . . . . . 1 - 29
1.2.2 Softkey Command Line . . . . . 1 - 32
1.2.3 Screen Forms for Configuring the I/O Modules . . . . . 1 - 37

1.3 Documentation with COM 115F . . . . . 1 - 52

1.4 Printing out Files Stored on the EPROM Submodule . . . . . 1 - 55

1.5 Configuration Error Messages . . . . . 1 - 56

2 Introduction to STEP 5 . . . . . 2 - 1

2.1 Writing a Program . . . . . 2 - 1
2.1.1 Methods of Representation . . . . . 2 - 1
2.1.2 Operand Areas . . . . . 2 - 3
2.1.3 Circuit Diagram Conversion . . . . . 2 - 4

2.2 Program Structure . . . . . 2 - 4
2.2.1 Linear Programming . . . . . 2 - 4
2.2.2 Structured Programming . . . . . 2 - 5

2.3 Block Types . . . . . 2 - 6
2.3.1 Organization Blocks (OBs) . . . . . 2 - 7
2.3.2 Program Blocks (PB) . . . . . 2 - 13
2.3.3 Sequence Blocks (SBs) . . . . . 2 - 13
2.3.4 Function Blocks (FBs) . . . . . 2 - 14
2.3.5 Data Blocks (DBs) . . . . . 2 - 18

2.4 Modifying the Program . . . . . 2 - 20

2.5 Number Representation . . . . . 2 - 20

Page

3 STEP 5 Operations . . . . . 3 - 1

3.1 Basic Operations . . . . . 3 - 1
3.1.1 Boolean Logic Operations . . . . . 3 - 1
3.1.2 Set/Reset Operations . . . . . 3 - 7
3.1.3 Load and Transfer Operations . . . . . 3 - 10
3.1.4 Timer Operations . . . . . 3 - 16
3.1.5 Counter Operations . . . . . 3 - 25
3.1.6 Comparison Operations . . . . . 3 - 29
3.1.7 Arithmetic Operations . . . . . 3 - 30
3.1.8 Block Call Operations . . . . . 3 - 32
3.1.9 Other Operations . . . . . 3 - 38
3.1.10 Segmentation with the BLD 255 Operation and the LPLZ Sequence in the Case of FBs . . . . . 3 - 39

3.2 Supplementary Operations . . . . . 3 - 42
3.2.1 Load Operation . . . . . 3 - 42
3.2.2 Enable Operation . . . . . 3 - 42
3.2.3 Bit Test Operations . . . . . 3 - 44
3.2.4 Digital Logic Operations . . . . . 3 - 46
3.2.5 Shift Operations . . . . . 3 - 49
3.2.6 Conversion Operations . . . . . 3 - 51
3.2.7 Decrement/Increment . . . . . 3 - 52
3.2.8 Disable/Enable Interrupt . . . . . 3 - 53
3.2.9 ”DO” Operation . . . . . 3 - 55
3.2.10 Jump Operations . . . . . 3 - 57
3.2.11 Substitution Operations . . . . . 3 - 59

3.3 System Operations . . . . . 3 - 65
3.3.1 Set Operations . . . . . 3 - 65
3.3.2 Load and Transfer Operations . . . . . 3 - 66
3.3.3 Arithmetic Operation . . . . . 3 - 68
3.3.4 Other Operations . . . . . 3 - 69

3.4 Condition Code Generation . . . . . 3 - 69

3.5 Sample Programs . . . . . 3 - 72
3.5.1 Transitional-Pulse Relay (Edge Evaluation) . . . . . 3 - 72
3.5.2 Binary Scaler . . . . . 3 - 73
3.5.3 Clock (Clock-Pulse Generator) . . . . . 3 - 74

3.6 Illegal Accesses . . . . . 3 - 75

3.7 Programming Notes . . . . . 3 - 76

3.8 LPLZ Incrementing Sequence . . . . . 3 - 76

Page

4 Program Test . . . . . 4 - 1

4.1 Signal Status Display . . . . . 4 - 1
4.1.1 Program-Dependent Signal Status Display ”STATUS” . . . . . 4 - 2
4.1.2 Direct Signal Status Display ”STATUS VAR” . . . . . 4 - 2

4.2 Forcing . . . . . 4 - 3
4.2.1 Forcing Outputs ”FORCE” . . . . . 4 - 3
4.2.2 Forcing Variables ”FORCE VAR” . . . . . 4 - 3

4.3 Search . . . . . 4 - 4

4.4 Programmer Operator Functions . . . . . 4 - 4

5 Error Diagnostics . . . . . 5 - 1

5.1 Interrupt Analysis . . . . . 5 - 1
5.1.1 ”ISTACK” Analysis . . . . . 5 - 1
5.1.2 Meaning of the ISTACK Displays . . . . . 5 - 3
5.1.3 LED Error Signalling . . . . . 5 - 5

5.2 Program Errors . . . . . 5 - 5
5.2.1 Determining an Error Address . . . . . 5 - 5
5.2.2 Program Trace with the Block Stack (”BSTACK”) Function . . . . . 5 - 7

5.3 System Parameters . . . . . 5 - 9

5.4 Error Detection and Entry in the Error DB . . . . . 5 - 10
5.4.1 Error Detection . . . . . 5 - 10
5.4.2 Error Entry in Error DBs 2 and 3 . . . . . 5 - 14
5.4.3 Ways of Evaluating the Error DBs . . . . . 5 - 16

5.5 Error Messages in Plaintext with COM 115F . . . . . 5 - 16
5.5.1 General Explanations . . . . . 5 - 16
5.5.2 Types of Error . . . . . 5 - 17

6 Blocks . . . . . 6 - 1

6.1 Integral Function Blocks . . . . . 6 - 2
6.1.1 Conversion Blocks . . . . . 6 - 2
6.1.2 Arithmetic Blocks . . . . . 6 - 3
6.1.3 Analog Value Matching Blocks . . . . . 6 - 4
6.1.4 The FB 252 Block Transfer Block . . . . . 6 - 9
6.1.5 The FB 253 Mailbox Transfer Block . . . . . 6 - 10
6.1.6 The FB 254 Synchronization Block . . . . . 6 - 10
6.1.7 Depassivation Block FB 255 . . . . . 6 - 14

6.2 OB 251 PID Algorithm . . . . . 6 - 15

6.3 Loadable Function Blocks . . . . . 6 - 25

Page

7 Application . . . . . 7 - 1

7.1 Hardware Installation . . . . . 7 - 1
7.1.1 Installing and Wiring the Hardware . . . . . 7 - 2

7.2 Entering the Subunit ID . . . . . 7 - 5

7.3 Initializing the Operating System . . . . . 7 - 5

7.4 Configuring the Hardware with COM 115F . . . . . 7 - 7

7.5 Transferring the Configuration Data to the PLC . . . . . 7 - 11

Appendices

A List of Operations . . . . . A - 1

A.1 Explanatory Notes on the Operands . . . . . A - 1
A.2 Basic Operation Set . . . . . A - 4
A.3 Supplementary Operations . . . . . A - 10
A.4 System Operations . . . . . A - 15
A.5 Evaluation of CC 1 and CC 0 . . . . . A - 16
A.6 Machine Code Listing . . . . . A - 17

B PLC Error Messages . . . . . B - 1

B.1 PLC Error Messages with COM 115F . . . . . B - 1
B.2 Evaluation of Error DBs (DB2 and DB3) without COM 115F . . . . . B - 88

C SIEMENS Addresses Worldwide . . . . . C - 1

Abbreviations

Index

Preface

The S5-115F is a failsafe programmable controller for the lower and mid performance ranges. It is for use wherever safety is the first priority and where potential dangers must be avoided. Hardwired solutions have dominated failsafe technology until now. With the S5-115F, you can now apply all the advantages of our programmable controllers (PLCs) to your safety-related controls. You write your process-oriented program as before and then enter all safety-related data with the COM 115F programming software.

You require detailed information in order to use the controller to its best advantage.

This manual aims to provide the necessary information in condensed form without overloading the user with superfluous knowledge, which means:

- Standardization of terminology and spelling
- More detailed breakdown of subjects
- Illustration of individual problems
- User-friendly arrangement of the contents
- Particular observance of the rules of safety technology

In this way, you will gain all the information required to operate your S5-115F. This manual is aimed at:

- Users with little previous experience
- SIMATIC S5 experts
- Officials conducting Licensing Authority acceptance tests.

However, the applications are so numerous that not all the problems that might occur can be dealt with in one manual. You will find a list of Siemens representatives in the Appendix who will help you in those cases where the manual cannot answer your questions.

Introduction

Please read the introduction carefully. You will find it easier to use the manual and this will save time.

This manual contains a detailed description of the S5-115F PLC with the CPU 942-7UF15.
For a description of the S5-115F PLC with the CPU 942-7UF11 or CPU 942-7UF12, please see the manual with the order No. 6ES5 998-1UF21. For a description of CPU 942-7UF13 and CPU 942-7UF14 (including product information), please see the manual with the order No. 6ES5 998-1UF23.

Description of Contents

This manual consists of two volumes and contains a detailed description of our SIMATIC S5-115F failsafe programmable controller.

Volume 1 contains the description of the hardware components. The contents of this volume can be divided into blocks according to topic:

- Reference manual
  (System Overview, Technical Description)
- Installation and operation
  (Installation Guidelines, Startup, Addressing)
- Special capabilities
  (Analog Value Processing, Communications Capabilities)
- Overview of technical specifications
- Safety-related rules and regulations

Volume 2 contains the description of the software components. The contents of this volume can be divided into blocks according to the following topics:

- Working with COM 115F programming software
  In the chapter "Configuring with COM 115F" you will find all the steps you require for configuring your S5-115F with the CPU 942-7UF15. Of course, you can also use the COM 115F system software, Version 3.2 for configuring programmable controllers with the CPU 942-7UF11 to CPU 942-7UF14.
- Programming Guide
  (Introduction to STEP 5, STEP 5 Operations)
- Testing
  (Program Test, Error Diagnostics)
- Blocks
  (Description of integral blocks and the use of loadable function blocks)
- Simple installation example with configuration of an S5-115F

You will find additional information in tabular form in the Appendices.
At the end of the book you will find correction forms. Please enter in these forms any suggestions you may have for improvements and corrections and send them to us. Your comments will help us to improve the next edition.

Conventions

In order to improve the readability of the manual, a menu-style breakdown was used, i.e.:

- The individual chapters can be quickly located by means of a thumb register.
- At the front of the manual is an overview containing the headings of the individual chapters.
- Each chapter has its own detailed table of contents.
  The individual chapters are subdivided into sections and subsections. **Bold-face** type is used for further subdivisions.
- Figures and tables are numbered separately in each chapter. The page following the chapter table of contents contains a list of the figures and tables appearing in that chapter.

Certain conventions were observed when writing the manual. These are explained below.

- A number of abbreviations have been used.
  Example: CPU (central processing unit)
- Footnotes are identified by superscripts consisting of a small digit (e.g. ”1”) or ”*”. The actual footnote is generally at the bottom left of the page.
- Cross-references are shown as follows:
  ”(7.3.2)” refers to subsection 7.3.2.
  No references are made to individual pages.
- Dimensions in drawings are given in mm with the value in inches given in brackets. Example: 187 (7.29).
- Values may be given in binary, decimal or hexadecimal numbers. The number system used is indicated in each case with a subscript, e.g. $F000_{H}$.
- Information of special importance is printed in black-framed ”windows”.

Warning

See the Safety-Related Guidelines for the User for definitions of the terms ”Note”, ”Caution”, ”Important”, ”Warning” and ”Danger”.

Manuals can only describe the current version of the programmable controller. Should modifications or supplements become necessary in the course of time, a supplement will be prepared and included in the manual the next time it is revised. The relevant version or edition of the manual appears on the cover. The present manual is edition ”1”. In the event of a revision, the edition number will be incremented by ”1”.

Courses

Siemens provide SIMATIC S5 users with extensive opportunities for training.

For more information, please contact your Siemens representatives.

Reference Literature

This manual is a comprehensive description of the S5-115F programmable controller. Topics not specific to the S5-115F, however, are only briefly dealt with. You will find more detailed information in the following literature:

- **Automating with the S5-115U**
  SIMATIC S5 Programmable Controllers

  Hans Berger
  Siemens AG, Berlin and Munich 1987

  Contents:
  - STEP 5 programming language
  - Program scanning
  - Integral program blocks
  - Interfaces to the I/Os

  Order No.: ISBN 3-8009-1484-0

You will find information on the range of controllers and programmers in the following catalogs:

- ST 52.3 ”S5-115U Programmable Controller”
- ST 57 ”Standard Functions Blocks and Driver Software for Programmable Controllers of the U Range”
- ST 59 ”Programmers”
- ET 1.1 ”ES 902 C 19 in. Packaging System”
- MP 11 Thermocouples, Compensation Boxes

Further components and modules (e.g. CPs and SINEC L1) have their own manuals. We will refer you to these sources at the appropriate points in the text.

1 Configuring with COM 115F

1.1 The Main Menu . . . . . 1 - 8
1.1.1 F1: Initialize Operating System . . . . . 1 - 8
1.1.2 F2: Setting Communications Parameters . . . . . 1 - 13
1.1.3 F3: I/O Configuration . . . . . 1 - 19
1.1.4 F4: PC Functions . . . . . 1 - 20
1.1.5 F5: Diagnostics . . . . . 1 - 21
1.1.6 F6: Defaults . . . . . 1 - 22
1.1.7 F7: System Handling . . . . . 1 - 23

1.2 Configuring I/O Modules . . . . . 1 - 29
1.2.1 General . . . . . 1 - 29
1.2.2 Softkey Command Line . . . . . 1 - 32
1.2.3 Screen Forms for Configuring the I/O Modules . . . . . 1 - 37

1.3 Documentation with COM 115F . . . . . 1 - 52

1.4 Printing out Files Stored on the EPROM Submodule . . . . . 1 - 55

1.5 Configuration Error Messages . . . . . 1 - 56

Figures

1-1. COM 115F Configuration Menu Hierarchy 1.- 3
1-2. The COM 115F Defaults Screen Form . . . . . 1 - 6
1-3. The COM 115F Main Menu . . . . . 1 - 7
1-4. 115F Operating System Initialization Menu for the CPU 942-7UF12 . . . . . 1 - 8
1-5. Basic Menu for Configuring Communications . . . . . 1 - 13
1-6. Typical Parameters for a Two-Channel SINEC L1 LAN . . . . . 1 - 14
1-7. Typical SINEC L1 Network . . . . . 1 - 15
1-8. Screen Form for SINEC L1 Interface (Mailbox Table) . . . . . 1 - 17
1-9. Screen Form for SINEC L1 Master Coordination Bytes (Example) . . . . . 1 - 18
1-10. Screen Form of the I/O Selection Menu . . . . . 1 - 19
1-11. ”PC Functions” Screen Form . . . . . 1 - 20
1-12. ”Diagnostics” Screen Form . . . . . 1 - 21
1-13. Cause of F Error Display . . . . . 1 - 22
1-14. System Handling . . . . . 1 - 23
1-15. Directory of Configuration DBs 1.- 24
1-16. Menu for Deleting DBs . . . . . 1 - 25
1-17. Menu for Transferring and Loading DBs 1.- 26
1-18. Menu for Printouts 1.- 27
1-19. Initialization Form for Digital Modules . . . . . 1 - 29
1-20. Initialization Form for Analog Modules 1.- 30
1-21. Softkey Commands of the Configuration Forms . . . . . 1 - 32
1-22. Characteristics Field of I/O Type 3 . . . . . 1 - 36
1-23. Initialization Form for I/O Type 1 . . . . . 1 - 37
1-24. Initialization Form for I/O Type 2 . . . . . 1 - 38
1-25. Initialization Screen Form for I/O Type 3 . . . . . 1.- 39
1-26. Initialization Form for I/O Type 8 . . . . . 1 - 40
1-27. Initialization Form for I/O Type 9 . . . . . 1 - 41
1-28. Initialization Form for I/O Type 10 . . . . . 1 - 42
1-29. Initialization Form for I/O Type 13 . . . . . 1 - 43
1-30. Initialization Form for CP 523 (I/O Type 13) 1.- 45
1-31. Initialization Form for I/O Type 14 . . . . . 1 - 46
1-32. Initialization Form for I/O Type 15 . . . . . 1 - 49
1-33. Initialization Form for I/O Type 16 . . . . . 1 - 50
1-34. Initialization Form for I/O Type 18 . . . . . 1 - 51
1-35. Main Menu of the Printout Package . . . . . 1 - 52
1-36. Printing the I/O Type 3 for CPU 942-7UF13 with Signal Group 28 . . . . . 1 - 54
1-37. Menu for Generating the Signature . . . . . 1 - 55

Tables

1-1. COM 115F Diskettes 1.- 1
1-2. COM 115F Files . . . . . 1 - 1
1-3. Programmers and Personal Computers Suitable for COM 115F . . . . . 1 - 2
1-4. Memory Submodules for Safety Mode . . . . . 1 - 26
1-5. Channel Types for Analog Input Modules (I/O Type 13) . . . . . 1 - 43
1-6. Channel Types for Analog Input Modules (I/O types 14 and 15) . . . . . 1 - 48
1-7. Channel Types for Analog Input Modules (I/O Type 16) . . . . . 1 - 51
1-8. COM 115F Configuration Error Messages 1.- 56
1-9. EPROM Function Error Messages . . . . . 1 - 58
1-10. Error Messages Occurring When Using Diskettes . . . . . 1 - 59
1-11. Error Messages to the PLC Interface . . . . . 1 - 59

1 Configuring with COM 115F

COM 115F is a software package for
- Configuring your S5-115F programmable controller
- Documenting that part of your system that is subject to official acceptance tests
- Error diagnostics with error display in plaintext
- System handling.

COM 115F is supplied on two diskettes that can be used on your programmer.

Table 1-1. COM 115F Diskettes

Supplied as	Programmer
1 5 ” diskette	PG 685 / PG 695 / PG 750 /PG 770
1 3 ” diskette	PG 635 / PG 720 / PG 710 / PG 730 / PG 740

The diskettes contain the following files:

Table 1-2. COM 115F Files

File name	Brief description
S5PEC15X.CMD	Program for configuring the S5-115F
S5PED15X.CMD	Program for documenting the part of the system subject to official acceptance tests
S5XEF01X.DAT	Text file with messages for error diagnostics
S5XEF02X.DAT	Texts which support you in configuring your system
S5XEF04X.DAT	Error texts to inform you of faults in the memory submodules
S5XEF03X.DAT	Screen forms for printing out your configurations
FB115FST.SSD	Block headers for S5-115F-specific function blocks (for off-line programming)
COMF.CMD	COM 115F CONFIGURE package call from the command line level under Personal CP/M-86
DRUF.CMD	COM 115F DOCUMENT package call from the command line level under Personal CP/M-86
S5CR59ST.S5D	Program file with standard error message texts for the memory submodule in the CP 523 communications processor

COM 115F programs run on the following programmers and personal computers:

Table 1-3. Programmers and Personal Computers Suitable for COM 115F

Programmers	Personal Computers
PG 635 PG 685 PG 695 PG 710 PG 720 PG 730 PG 740 PG 750 PG 770	Siemens PC 16-11 Siemens PC 16-20 IBM ' XT and 100 % compatibles IBM ' AT and 100 % compatibles

The S5-DOS operating system must be available on all systems.

Generating a working copy of the diskettes supplied

First, make a working copy of the diskettes supplied and then keep the original software in a safe place.

Copy the COM 115F programming software from diskette to hard disk:

Load the PCP/M86 operating system
Insert the diskette in drive A
Copy the contents of the diskettes to user area 0 of the hard disk (drive B)
e.g.: Type in B>USER 0 and press <RETURN>
Type in B>PIP B:=A:*.* and press <RETURN>
If necessary, assign the attributes [RO] and [SYS] to the copied files
e.g.: B>SET *.* [RO SYS]
The attribute [RO] protects the files from being overwritten. The attribute [SYS] allows the files to be accessed from all user areas.

The program section of COM 115F contains the following packages:

- COM 115F CONFIGURE for
  - Entering the configuration data in the programmer
  - Printing out the configuration data

- COM 115F DOCUMENT for
  - Safety-oriented readout of the configuration data from the programmable controller
  - Printing out the data in the programmable controller

Note

The printout of the configuration data using COM 115F DOCUMENT and a comparison of the printout with the configured data is a strict requirement in the Licensing Authority acceptance test.

The functions of both program packages are initiated by menu-driven operator prompting on the programmer. The menus are linked to each other in a tree structure.

You can reach all the submenus using softkeys **F1** to **F7**. **F8** brings you back up through the higher level menus. The **HELP** key gives you a detailed description of the functions of the keys.

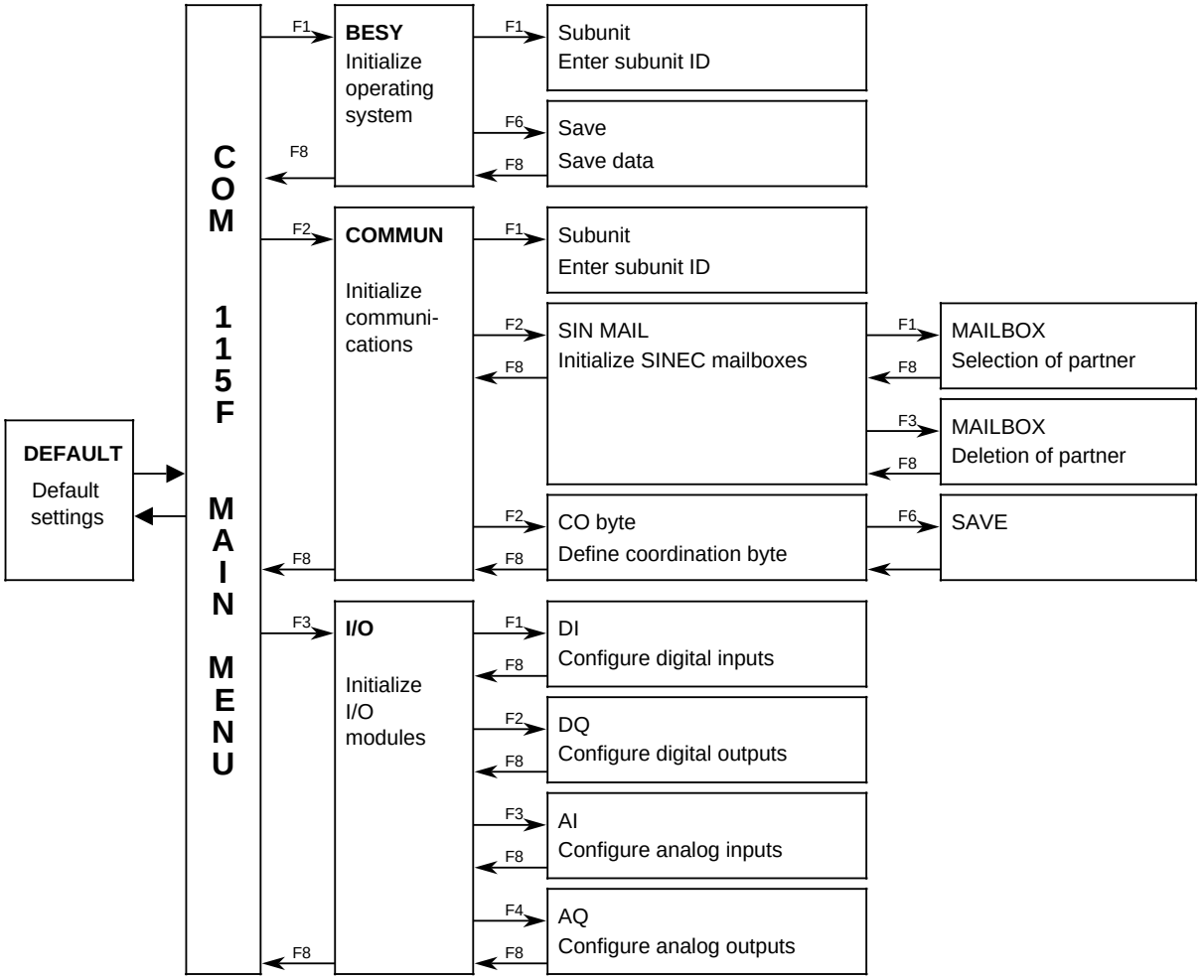


Figure 1-1. COM 115F Configuration Menu Hierarchy

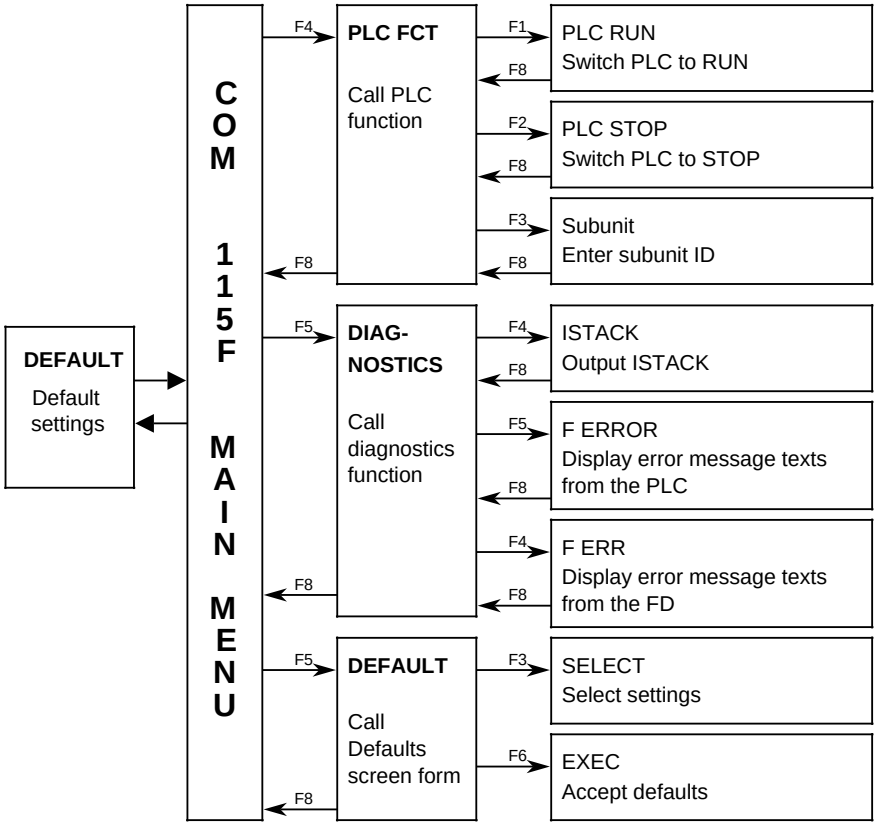


Figure 1-1. COM 115F Configuration Menu Hierarchy (Continued)

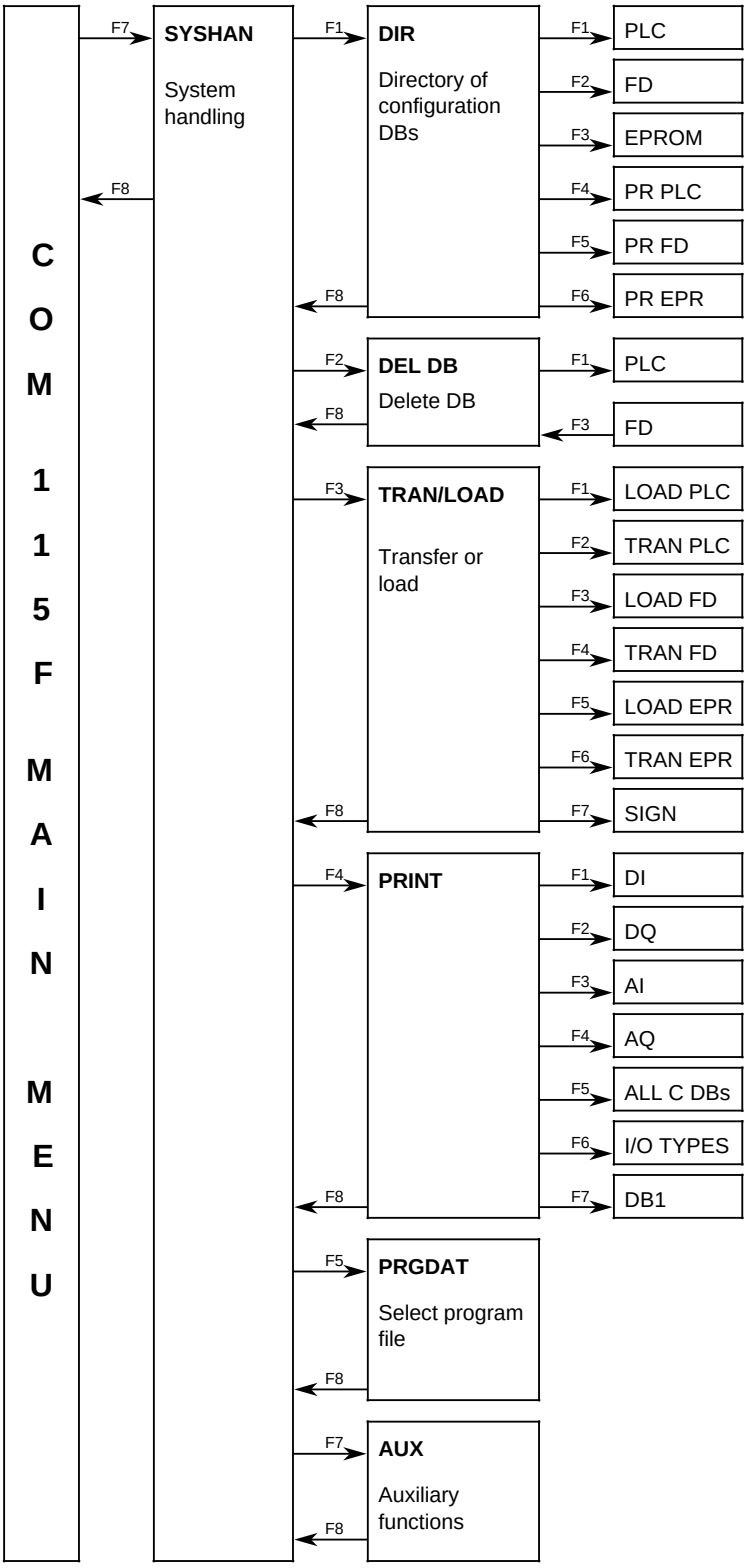


Figure 1-1. COM 115F Configuration Menu Hierarchy (Continued)

Starting COM 115F CONFIGURE

Make sure that all the COM 115F files you require are either
- on the hard disk of your programmer or
- on the diskette in your programmer drive.

Start the S5 command interpreter by pressing **S5 <CR>.**
A list of all the STEP 5 packages available on diskette or hard disk will then appear.

Select the desired package with the cursor control keys:
- COM 115F CONFIGURATION

Confirm your selection with the **F1** ”PACKAGE” key.
The ”Defaults” screen form will appear.

```
DEFAULTS                                        COM 115F / PDC15

CPU 9427UF...: 15                PROGRAM FILE   : B:@@@@@@ST.S5D

SYMBOLS      : NO                SYMBOLS FILE   :

FOOTER       : NO                FOOTER FILE    :

PRINT WIDTH  : SMALL             PRINTER FILE   :

OP MODE      : OFF

PATH NAME    :                   PATH FILE      :


Status:  SPECIFICATION OF TYPE
```

F1	F2	F3	F4	F5	F6	F7	F8
		SELECT			EXEC		

Figure 1-2. The COM 115F Defaults Screen Form

Completing the Defaults screen form

The following is the minimum information you must enter in the Defaults screen form:
- Program file name in which your entire configuration data will be stored.
- The last two digits of the CPU descriptor
- The operating mode for online or offline programming.

Key assignments for the Defaults form:

Cursor keys — The double-arrow keys < > or < > move the cursor to the right or left of the screen and the single-arrow keys < >, < >, < >, < > move the cursor to the input fields.

F3 (SELECT) — Shows the choices at the cursor position.

F6 (EXEC) Declares the selected and displayed parameters to be valid and calls the next menu.

Exec key The Exec key has the same function as the **F6** (EXEC) key.

Break key The programmer rejects the parameter just entered or changed. The programmer displays the Defaults form with the basic settings again.

With the COM 115F software package version 3.2, you can configure programmable controllers with the 942-7UF11, 942-7UF12, 942-7UF13, 942-7UF14 and 942-7UF15 central processing units. For this reason, check the CPU identifier specified in the Default.
If you are not yet familiar with the terms from the Defaults form, please read the explanation of terms in your programmer manual.

Complete the Defaults form and confirm the entries with the **F6** key.

You are now in the COM 115F MAIN MENU. If you now press the <HELP> key, the following form appears on the screen (Figure 1-3).

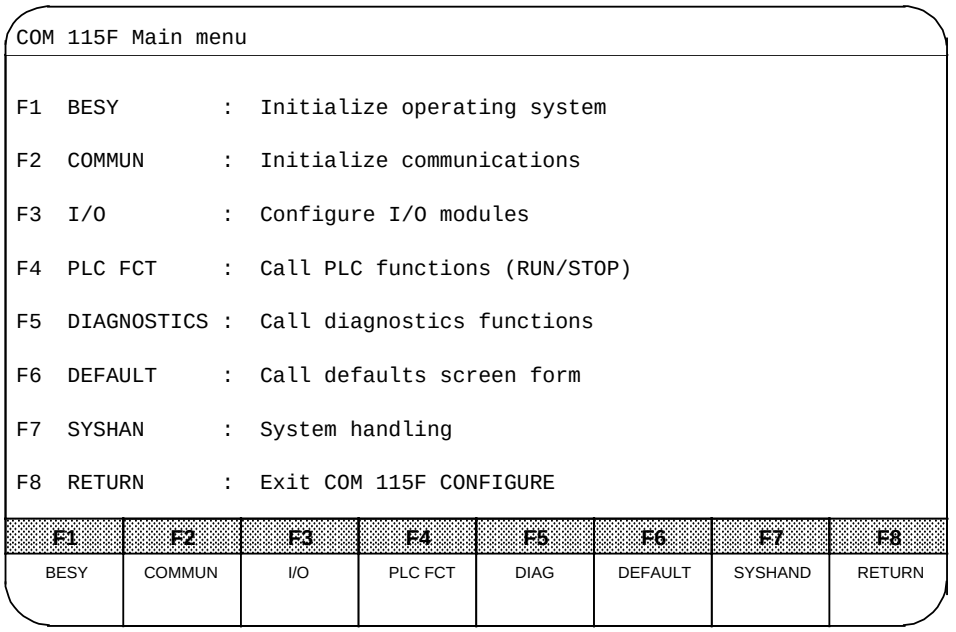


Figure 1-3. The COM 115F Main Menu

Note

If you want to process parameters already set, you must first load the program file from the diskette or the hard disk with the system handling menu.

1.1 The Main Menu

The main menu takes you to the first submenu level.
The meanings of the function keys are described in more detail in the following chapters.

1.1.1 F1: Initialize Operating System

The ”Initialize Operating System” menu appears when you press the <F1> softkey (Figure 1-4). If the configuration DBs from the PC, FD (diskette) or EPROM have already been loaded using the handling menu, the initialized data will then appear in the entry fields. When a system is being generated for the first time, some entry fields either have the default settings or are empty. A value must be entered in each empty entry field. In the case of some parameter entries, the value ”0” is permissible in the brackets. If you enter ”0” here, you do not have the relevant operating system function.
The <Cursor down> key takes you to the next line. The <Cursor up> key takes you to the line above. Every entry is checked for limits. In the case of illegal entries the ”ILLEGAL INITIALIZATION” error message appears. If <CR> is pressed in the last line, the question ”DATA CORRECTLY ENTERED?” appears. If you confirm with the <Exec> key, the system returns to the main menu and, if you press <Break>, the cursor jumps back to the first entry line.

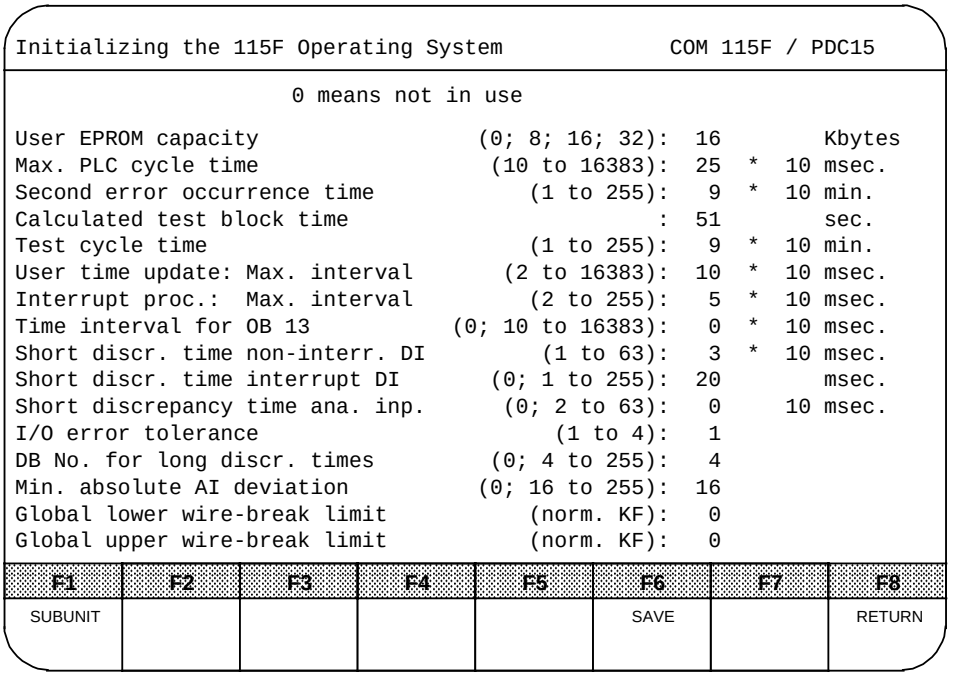


Figure 1-4. 115F Operating System Initialization Menu for the CPU 942-7UF15

Notes on Figure 1-4

- User EPROM capacity
  Enter the capacity of your memory submodule here. It is of no importance whether you
  - use a RAM submodule in test mode
  or
  - use an EPROM or EEPROM submodule in safety mode.

- Max. PLC cycle time
  Enter here the cycle time your PLC must on no account exceed (Vol. 1, 10.4 of the Manual).

- Second error occurrence time
  The second error occurrence time is prescribed by the licensing authority. It follows the specifications made in the system-specific regulation for redundant systems.
  The second error occurrence time is the time during which the occurrence of a second error in the redundant system can be ignored.

- Calculated test block time
  The complete selftest lasts some minutes and cannot be processed in one go. For this reason, the PLC selftest is divided into several test slices. One or more test slices are processed in succession in each PLC cycle.
  The test block time is not configured by the user but calculated autonomously by COM 115F. It depends on the following:
  - Number of Type 3 DI bits with 24 V input voltage
  - Number of Type 3 DI bits with 220 V input voltage
  - Number of Type 10 DQ bits
  - Number of Type 14 and Type 15 AI channels.

  The test component per PLC cycle is controlled using the test block time and the test cycle time.
  When you first open the menu for initializing the 115F operating system, a default value appears in the ”Calculated test block time” line. This value applies to the user EPROM specified in the menu and a device configuration without I/O modules.
  The test block time is calculated anew after every change:
  - if ”User EPROM capacity” is re-initialized

  or
  - if I/O modules of the I/O types named above are configured.

- Test cycle time
  For safety reasons, the complete PLC selftest must be run once per test cycle time (e.g. 8 hrs). One or more test slices per PLC cycle are processed consecutively in cyclic mode.
  The following conditions must be satisfied when defining the test cycle time:

  Test cycle time second error occurrence time
  Test cycle time 10 · test block time

  Note the different time units in the input form and use a common unit for checking the above conditions.

  The greater the test cycle time selected, the less will be the load on the operating system per PLC cycle caused by the PLC selftest component.

Note

When defining the test cycle time, note that the ”calculated test block time” increases with the number of I/O bits configured.

For this reason, check the validity of the condition ”Test cycle time 10 · test block time” after configuring the I/Os.

If the condition is not fulfilled, COM 115F displays the error message for illegal configuration when transferring the configuration data.

- Update user cycle time / interrupt handling: Max. time interval
  Max. interval between two calls of FB 254 SYNC with
  - Parameter Time update
  - Parameter Interrupt handling (6.1.6)

  The interval between calls must be at least 1 msec shorter than the value configured.
  If you do not want the user time update to be monitored, enter the value 16383.
  If you do not want interrupt handling to be monitored, enter the value 255.

- Time interval for OB 13 (2.3.1)
  If you do not use OB 13 in your control program, enter the value 0.

- Short discrepancy time for DIs without interrupt capability (Vol. 1, 10.6 of the Manual)

- Short discrepancy time for DIs with interrupt capability (Vol. 1, 10.6 of the Manual)

- Short discrepancy time for analog input modules
  Define a uniform discrepancy time for configuration of analog input modules. In the case of an analog input module discrepancy, the inputs are immediately scanned until the discrepancy time has expired or the deviation lies between the defined discrepancy values.
  If you enter 0 when setting parameters, the discrepancy time will be one PLC cycle.

- I/O modules: Error tolerance (Vol. 1, 10.17 of the Manual)
  The I/O error tolerance variant determines how the S5-115F responds to errors in the I/O modules.
  You have a choice of three I/O error tolerance variants (I/OETVs):

  I/O ETV 1: All I/O errors result in PLC STOP
  I/O ETV 2: Individual or technologically related modules can be deactivated via the 115F operating system.
  I/O-PFTV 3: Responses to I/O errors can be initiated via the user program (signal group 28)
  I/O-PFTV 4: Responses to I/O errors can be initiated via the user program (signal group 27 and 28)

- DB No. for long discrepancy times
  - Enter the number of the DB in which COM 115F is to store long discrepancy times.

  or
  - Enter a zero if you do not use long discrepancy times.

- Absolute AI deviation
  This value has two functions:

  The value configured here is interpreted during the AI test as the tolerable difference between the analog test value displayed and the analog value read in.

  The S5-115F monitors analog input deviations on both subunits.
  The value defined here for ”Min. absolute AI deviation” is always significant if you have selected a relative tolerance range (e.g. 5 %) when configuring analog input modules. Electromagnetic interference can quickly cause relative deviations to be exceeded, especially in the case of small analog values. Whenever the relative tolerance range is exceeded, the 115F operating system checks to establish whether or not the value for ***”Absolute AI deviation”*** has been exceeded. The system only responds if this minimum value has been exceeded.

- Global lower / upper wire-break limits
  Enter here the limits of the range within which a value is to be interpreted as a wire break. Set a default for wire break monitoring at this point. You will generally use the ”global wire break limits” when several analog inputs have the same standardized value range.
  The limits for wire break monitoring are also scanned when configuring FB 250 ANEI. If you want the default value for ”global wire break monitoring”, enter the value ”-32768” for the ”ODGR” (upper wire break limit) and ”UDGR” (lower wire break limit) parameters of FB 250 ANEI.
  If you do ***not*** want to use the ”global wire break limits” when configuring FB 250 ANEI, you must assign the local ”ODGR” and ”UDGR” parameters accordingly in FB 250 ANEI.

F1 Scanning or entering the subunit ID

The subunit ID of the connected PLC is read in online mode by pressing function key **F1**. The value read is displayed in the status line, e.g.:

”SUBUNIT A”

if the subunits have not yet been defined, the following appears:

”SUBUNIT ..”

”SUBUNIT NOT YET DEFINED”.

After reading out, the cursor jumps to the output subunit ID. You can change this ID.

Entering the subunit ID

Switch on the power supplies for both subunits.
Connect the programmer to the subunit which is to be subunit A.
Press the **F1** ”Subunit” key and read the subunit ID.
Change the subunit ID if required by entering the letter **A**.
Confirm the entry with the <Enter> key.
Connect the programmer to the subunit which is to be subunit B.
Press the **F1** ”Subunit” key and read the subunit ID.
Change the subunit ID if required by entering the letter **B**.
Confirm the entry with the <Enter> key.
Switch both PS 931 power supplies off and on again.

Connect the programmer to subunit A after entering the subunit ID. Data traffic between the programmer and the S5-115F is only possible if both subunits have received the correct subunit ID.

Note

Each CPU 942-7UF13 has subunit ID B as supplied from the factory. If you have not changed this ID in the CPUs, you need only enter ID A for subunit A.

1.1.2 F2: Setting Communications Parameters

Softkey F2 takes you from the COM 115F main menu to the basic menu for ”Setting 115F Communication Parameters”.

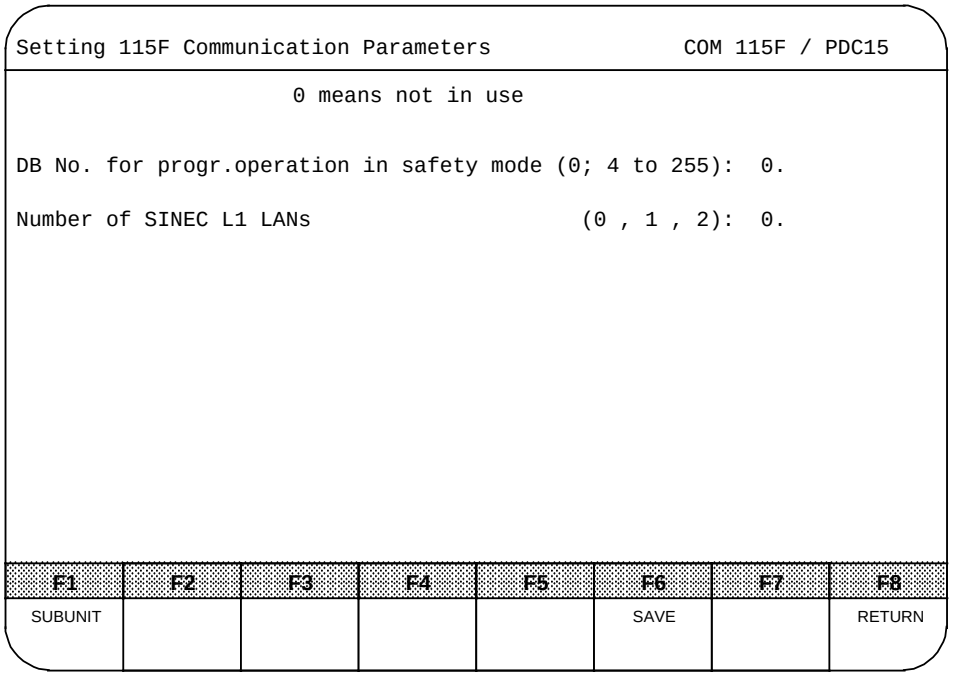


Figure 1-5. Basic Menu for Configuring Communications

Explanations of Figure 1-5

- DB No. for parameter entry DB in safety mode (4.4)
  Enter the block number of your parameter entry DB at this point. You can store process-specific parameters in this data block using the programmer. The programmer can only be used in safety mode when the PLC is at STOP.

- Number of SINEC L1 channels
  Specify here if you want to configure a SINEC L1 LAN.
  If you configure a SINEC L1 LAN, additional parameters are entered in the form and interrogated (see Figure 1-6).

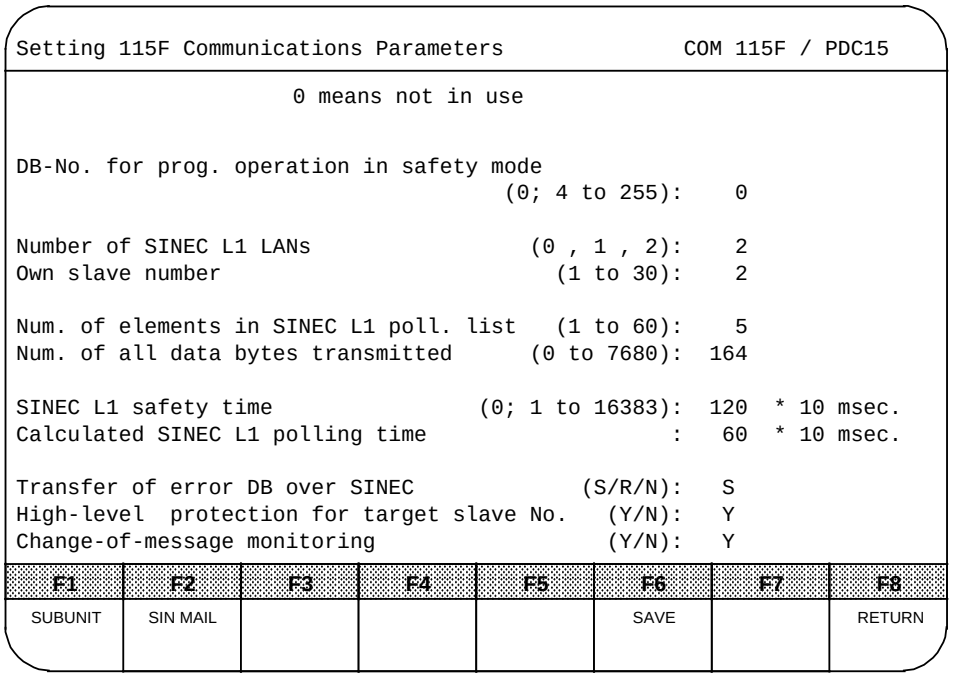


Figure 1-6. Typical Parameters for a Two-Channel SINEC L1 LAN

Explanations of Figure 1-6

- DB number for programmer operation in safety mode (Figure 1-5)

- Own slave number
  Both subunits always have the same number

- Number of elements in SINEC L1 polling list
  Enter the number of data paths here.
  The SINEC L1 polling list must contain the following elements:
  - The numbers of a slave that are sources of data transfers (source slave)
  - The numbers of a slave that are destinations of data transfers from the master and have not already been defined as source slaves.

  If several data transfers originate in one slave, it is necessary and advisable to specify the slave more than once. The same slave numbers should not appear next to each other in the polling list so that enough time remains for internal slave data processing.

  Example: Determining the number of elements in the SINEC L1 polling list
  The following data transfers are desired in a network consisting of slave 1, slave 2, slave 3, slave 4, slave 5 and the master:
  - from the master to slave 1
  - from slave 1 to the master
  - from slave 2 to slave 1
  - from slave 2 to slave 3
  - from slave 3 to slave 4
  - from the master to slave 5

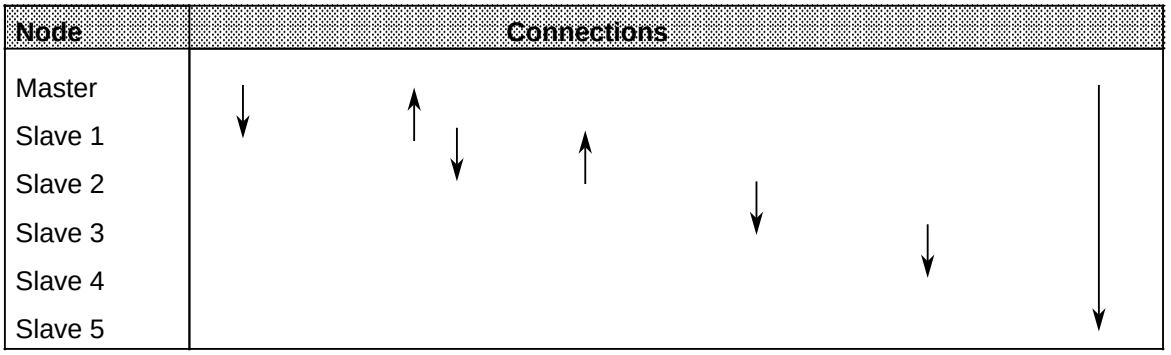


Figure 1-7. Typical SINEC L1 Network

At least the following must be entered in the SINEC L1 polling list according to the above rules:
Slave 1, slave 2, slave 3 and slave 5

Slave 4 is not entered since it is neither the source slave of a data transfer nor the destination slave of a data transfer from the master.

To spread the traffic load evenly over all data paths, slave names must be repeated when networking the S5-115F with SINEC L1. The following polling list is obtained at for the example if we observe the recommendation that the same slave numbers must not appear next to each other:
Slave 1, **slave 2**, slave 3, **slave 2**, slave 5

Since there are five elements in the SINEC L1 polling list, you must enter a ”6” in the ”Number of elements in the SINEC L1 polling list” line when setting the communications parameters.

Note

The SINEC L1 polling list is not defined in the case of point-to-point connections. In this case, you must define the number of elements as you would when implementing the connection with a CP 530.

- Number of all data bytes transmitted
  This number is used to calculate the SINEC L1 polling time.
  Enter the sum of all message lengths including two header bytes transmitted within one SINEC L1 polling cycle on the LAN. Please also take the data sent by the master into account.

- SINEC L1 safety time
  The SINEC L1 LAN safety time is used to monitor the SINEC L1 LAN response time. The response time begins when FB 254 is called with the SINEC L1 processing parameter in the source S5-115F and ends when the message is entered in the Receive mailbox of the destination S5-115F (Vol. 1, 10.3.5 of the Manual).

- SINEC L1 polling time
  The SINEC L1 polling time is calculated automatically. It must be less than the SINEC L1 safety time.
  You will find more notes on setting SINEC L1 LAN parameters in Chapter 7 of the S5-115F Manual (Vol. 1, 7.2 of the Manual).

- Transferring the error DB via SINEC L1
  If you have configured a SINEC L1 LAN, you can send the body of the error DB with the error message code to the master for evaluation. The S5-115F automatically sends the body of the error DB to the master in the event of a fault.

  You can choose one of the following three options:
  S = Only send an error message to the master if the error causes the S5-115F to assume STOP status
  R = Send all error messages to the master (even those when the S5-115F remains on RUN)
  N = Do not send any error messages to the master.

- High-level protection of the destination slave number
  High-level protection of the destination slave number is necessary when several SINEC L1 slaves are configured with the same mailbox length. However, we recommend that you also use high-level protection of the destination slave number in all other cases.
  The S5-115F carries out high-level protection of the destination slave number automatically. Please note the different message structure (Vol 1, 7.2.2 of the Manual).

- Change-of-message monitoring
  Facilities for recognition of a change of message are essential if you want to send safety-related broadcast messages via the SINEC L1 LAN. Broadcast messages are messages addressed to the master and all slaves at the same time. We recommend that you use the monitoring facility for recognition of a change of message even if you are not sending broadcast messages.
  The S5-115F handles monitoring for a change of message automatically. Please note the different message structure (Vol 1, 7.2.2 of the Manual).

> **Note**
>
> If you assign parameters for change-of-message monitoring, the S5-115F will also automatically perform high-level protection of the destination slave number.

F1 Subunit identifier
Press **F1** if you want to read or change the subunit ID.

If you have configured a SINEC L1 LAN in the basic menu for "Setting 115F Communications Parameters", press function key F2.

F2 SIN-MAIL
The "SINEC L1 interface" screen form appears with the partner matrices (0: master, 1 to 30: slave, 31: broadcast) of the Send and Receive mailboxes.

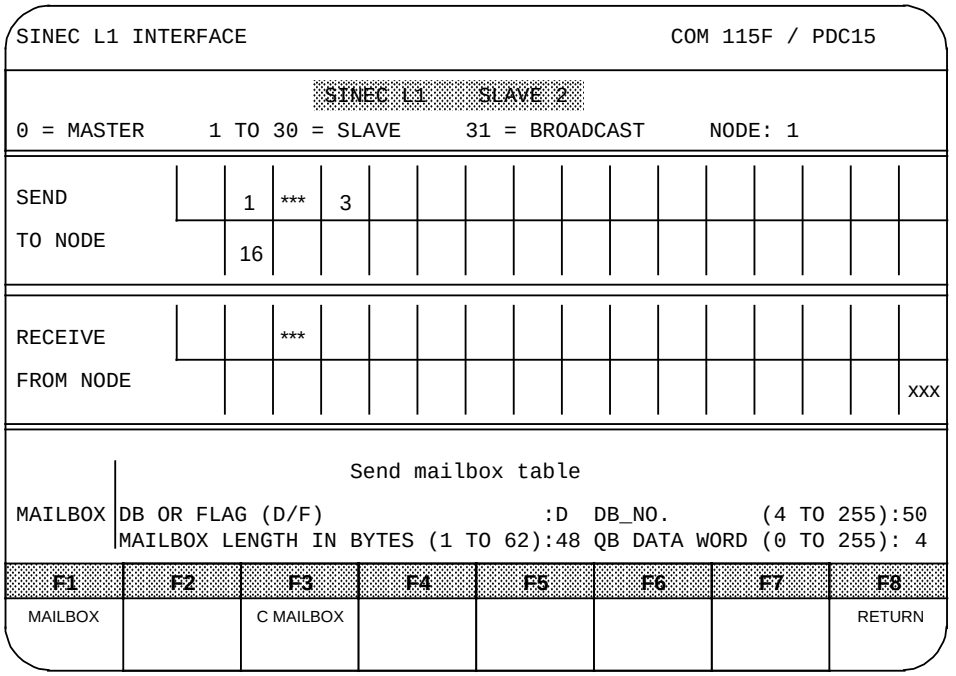


Figure 1-8. Screen Form for SINEC L1 Interface (Mailbox Table)

Use the cursor control keys to select the desired node.
The relevant mailbox table will be displayed provided it has already been selected once.
'***' will appear at those positions where the node numbers correspond to the slave numbers.

F1 Select a node
F3 Deselect a node

If a node has been selected, the cursor will jump to the 'Mailbox table' field. Enter the relevant mailbox address (DB or flag) and length here.

The following can be selected in this way:

- Single LAN
  - up to 30 nodes (incl. master) with one Send mailbox each
  - up to 30 nodes (incl. master) with one Receive mailbox each
- Redundant LAN:
  - up to 14 nodes with one Send mailbox each and one master with two Send mailboxes
  - up to 15 nodes (incl. master) with two Receive mailboxes each

Note

If you have assigned parameters for high-level protection of the destination slave number or monitoring for recognition of a change of message, please note when filling in the form that the mailbox must be two bytes longer than your net data (Vol 1, 7.2.2 of the Manual).

F3 Master coordination byte
Figure 1-9 will appear if you have entered a master in the SINEC L1 Interface screen form. You can enter the following:
Master coordination byte Receive B
Master coordination byte Send B
Master coordination byte Receive A (redundant LAN only)
Master coordination byte Send A (redundant LAN only)

```
SINEC L1 Master Coordination Bytes                COM 115F / PDC15

COORD BYTE RECEIVE      A |DB or flag          (D/F):  D
                          |DB No.         (4 to 255):   45
                          |DB word No.    (0 to 255):    1

COORD BYTE SEND         A |DB or flag          (D/F):  D
                          |DB No.         (4 to 255):   45
                          |DB word No.    (0 to 255):    2

COORD BYTE RECEIVE      B |DB or flag          (D/F):  D
                          |DB No.         (4 to 255):   45
                          |DB word No.    (0 to 255):    3

COORD BYTE SEND         B |DB or flag          (D/F):  D
                          |DB No.         (4 to 255):   45
                          |DB word No.    (0 to 255):    4

| F1 | F2 | F3 | F4 | F5 | F6   | F7 | F8     |
|    |    |    |    |    | SAVE |    | RETURN |
```

Figure 1-9. Screen Form for SINEC L1 Master Coordination Bytes (Example)

1.1.3 F3: I/O Configuration

After selecting the I/Os in the main menu with the **F3** key, the ”I/O modules” menu for configuring the I/Os will appear on the screen (Figure 1-10). You now have a choice between digital and analog modules. Softkeys **F1** to **F4** will call up the relevant I/O screen forms, in which you can then enter your configuration data.
You will find a detailed description of these I/O screen forms in the chapter on ”Configuration Forms for I/O Modules” (1.2.3).

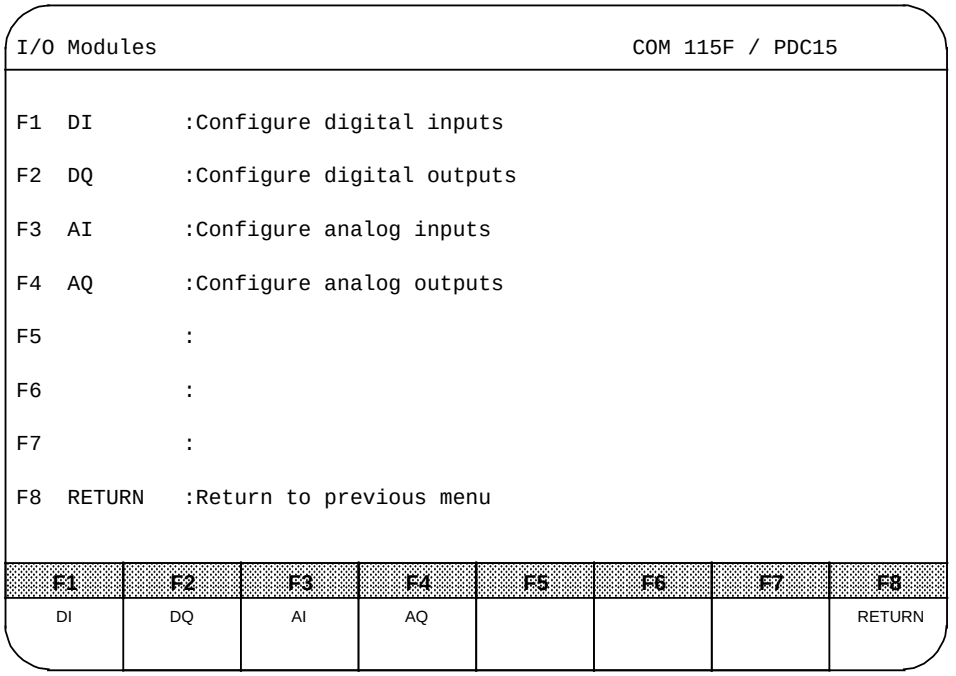


Figure 1-10. Screen Form of the I/O Selection Menu

1.1.4 F4: PLC Functions

Softkey **F4** will take you to the ”PLC Functions” menu. From this menu you can start or stop the S5-115F in test mode.

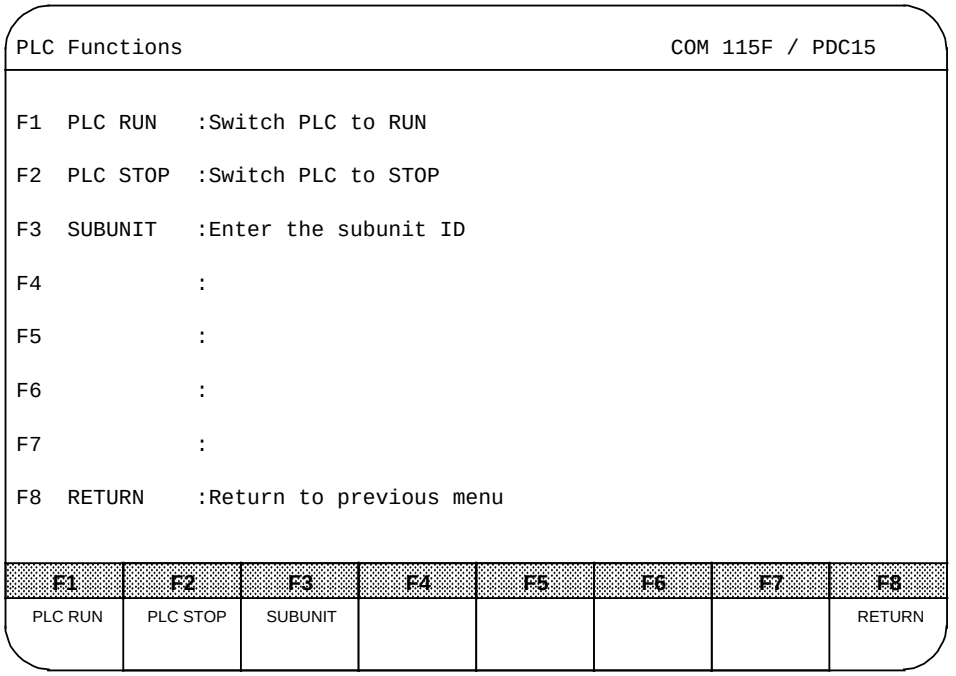


Figure 1-11. ”PLC Functions” Screen Form

F1 PLC RUN
F2 PLC STOP
F3 Subunit identifier

1.1.5 F5: Diagnostics

Softkey **F5** takes you to the error diagnostics of the COM 115F package.
If you also press the HELP key, the following form appears on your programmer screen:

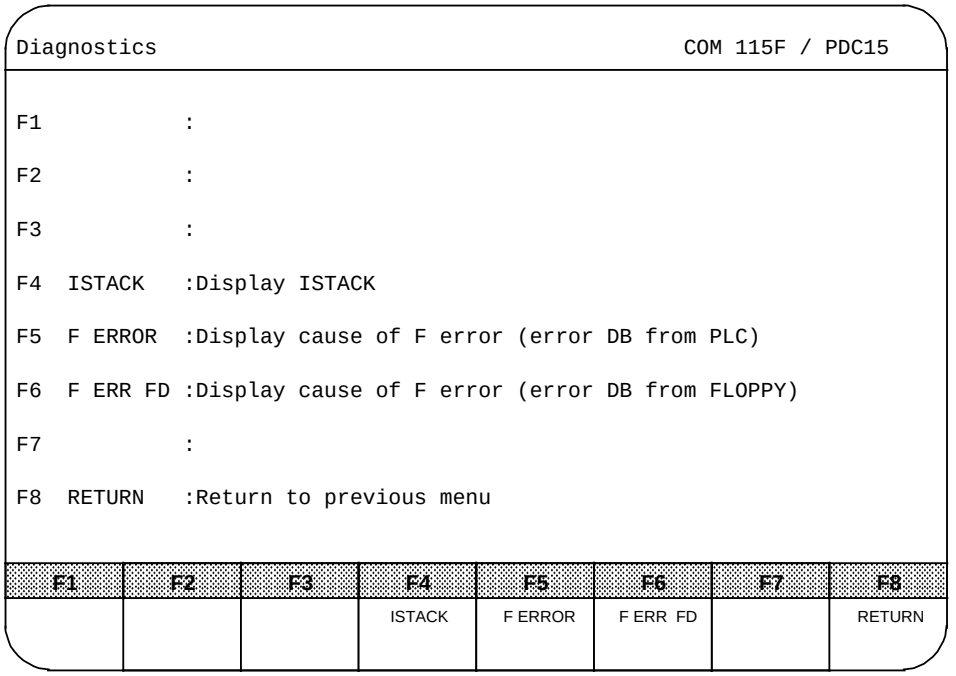


Figure 1-12. ”Diagnostics” Screen Form

F4 ISTACK

The interrupt stack is displayed (5.1.1).

F5 Display of cause of F error from the PLC

The S5-115F errors are displayed.

F6 Display of cause of F error from diskette (FD)

You can have plaintext error messages displayed from error DBs stored on diskette.

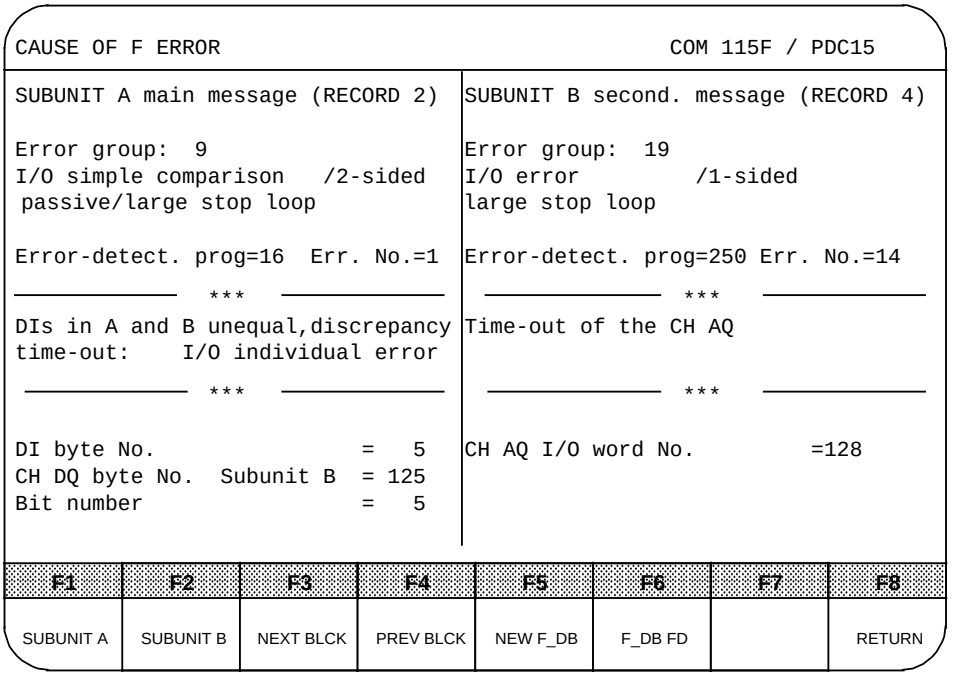


Figure 1-13. Cause of F Error Display

You will find a detailed description of the error messages in appendix B.

1.1.6 F6: Defaults

Press **F6** to call the ”DEFAULTS” menu (Figure 1-2).

1.1.7 F7: System Handling

Press softkey **F7** to display the ”System Handling” menu (Figure 1-14).

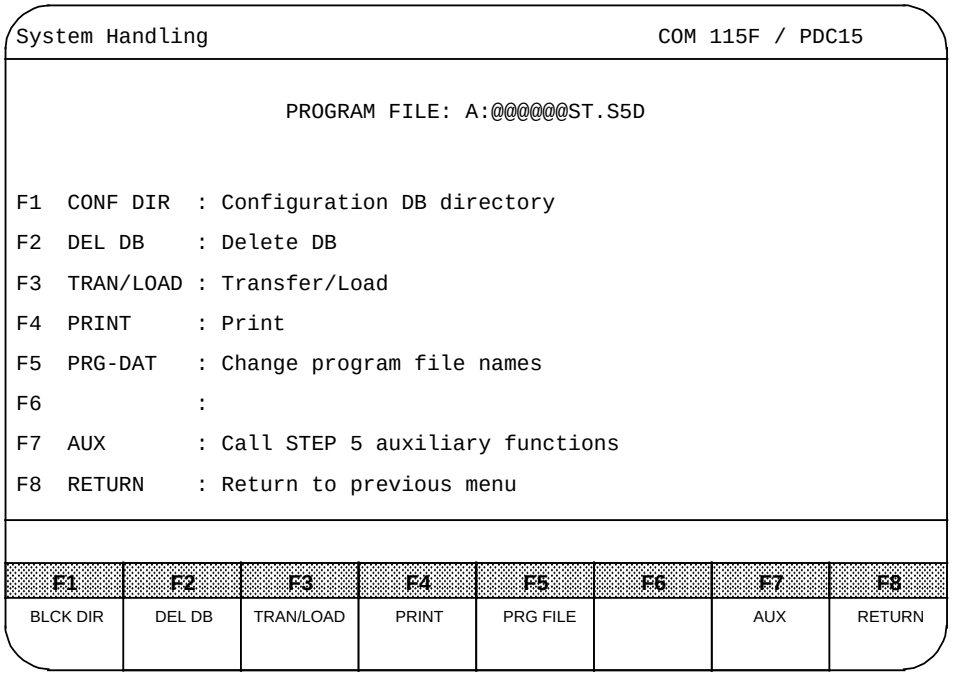


Figure 1-14. System Handling

In this menu you can
- Delete, transfer or load configuration data blocks.
- Display the directory of the configuration data blocks on a peripheral device of your choice.
- Print out configuration data blocks for documentation purposes.

> **Note**
>
> The functions can only be implemented if DB 1 is available. This data block is required to differentiate between configuration DBs and ”normal” DBs.

Since COM 115F works with the S5-DOS operating system, a program file must be selected when accessing the hard disk or diskettes.
This is done with softkey F5. With the exception of the ”TRANSFER TO EPROM” function, the possible operating functions are concerned only with configuration data blocks, since only these are relevant for configuring an S5-115F. The individual softkey functions are described in more detail in the following.

F1 Display block directory

Press the **F1** key to call the ”DIRECTORY OF CONFIGURATION DBs” menu (Figure 1-15).

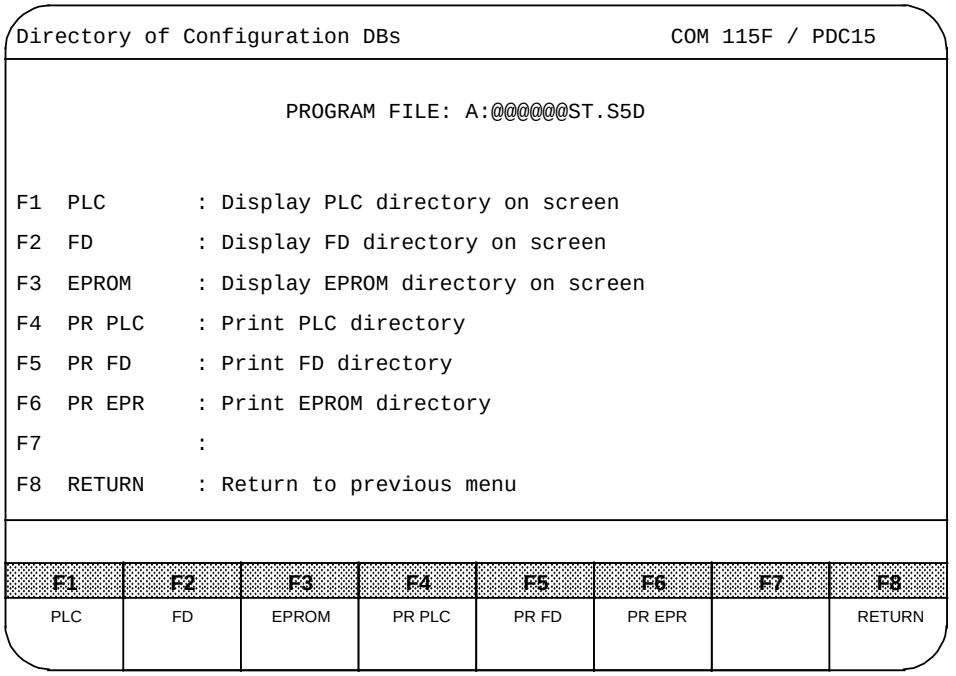


Figure 1-15. Directory of Configuration DBs

The directory of the available configuration DBs can be output from the PLC, FD or EPROM. The output can be terminated with the <Break> key. The question ”CONTINUE?” will then appear. If you acknowledge with the <Exec> key, output will continue.
The question ”CONTINUE?” will also appear at the end of the output. If you acknowledge with the <Exec> key, the directory menu will appear again.
The directories of the PLC, FD and EPROM can be printed out for documentation purposes using softkeys F4, F5 and F6.

F2 Delete DB

Press the **F2** softkey to call the Delete menu (Figure 1-16).

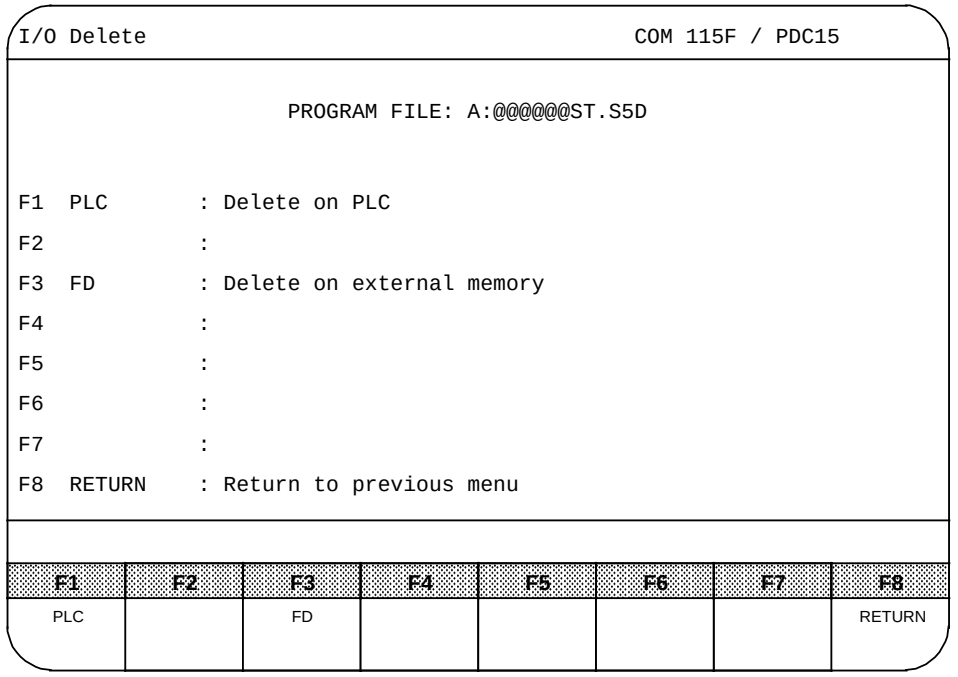


Figure 1-16. Menu for Deleting DBs

The Delete routine can be used to delete individual DBs and successive DBs. The following prompt appears in an input line when the function is selected:

”ENTER DB NUMBER(S):”

Use the following syntax when entering DB numbers:

number X
or
number X - number Y

Confirm the entry by pressing the EXEC key. To avoid inadvertent deletion, you will be asked again:

”DELETE DB x” or ”DELETE DB x TO DB y?”

The DBs specified will be deleted when you press the <Exec> key. The program then returns to DB number input. Only configuration DBs can be deleted in this menu.
If you attempt to delete DB 1, you will be prompted to confirm the deletion:

”DELETE DB 1?”

DB 1 is deleted when you press the <Exec> key. All attempts to delete are cancelled with the <Break> key. If DB 1 is missing, configuration DBs will not be recognized and therefore not deleted.

F3 Transferring and loading DBs

Press the F3 key to call the Transfer menu (Figure 1-17). The PLC, FD and EPROM can also be referenced from here.

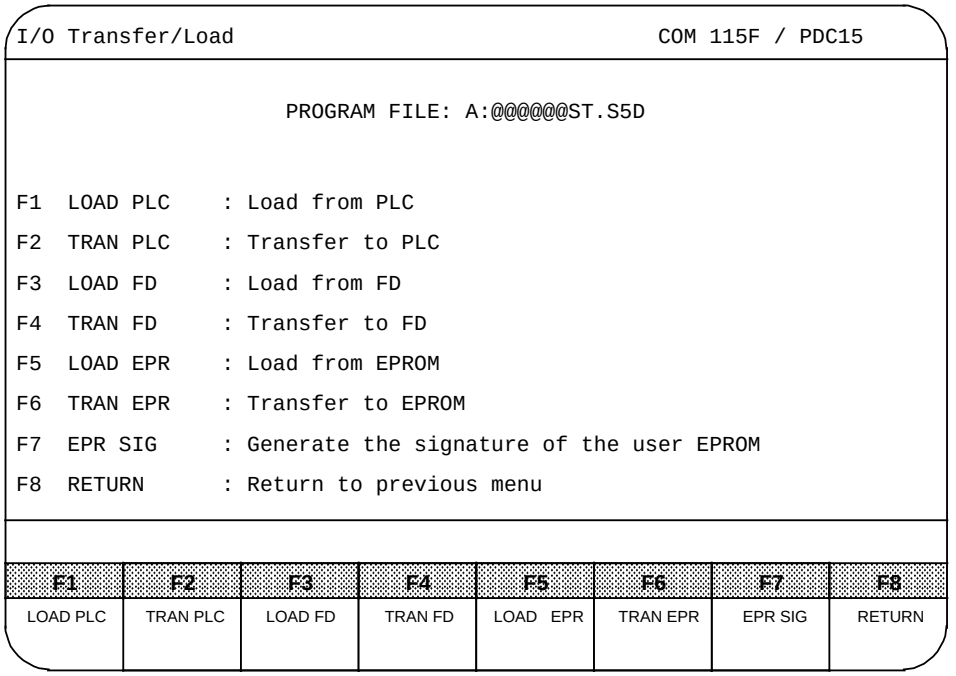


Figure 1-17. Menu for Transferring and Loading DBs

Loading the configuration data
You can load your configuration data into the programmer memory with the load functions.
Your data can be

- on a diskette or hard disk drive
- in the PLC memory

If you select EPROM, you will also be prompted for the EPROM type.

Table 1-4. Memory Submodules for Safety Mode

Order Nr. of Memory Submodule/ Type	Capacity	Programming Number
6ES5 375-0LA15 / EPROM	8x2^{10} Byte	11
6ES5 375 -0LA21 / EPROM	16x2^{10} Byte	12
6ES5 375 -0LA41 / EPROM	32x2^{10} Byte	17
6ES5 375-1LA15 / EPROM	8x2^{10} Byte	411
6ES5 375-1LA21 / EPROM	16x2^{10} Byte	412
6ES5 375-1LA41 / EPROM	32x2^{10} Byte	417
6ES5 375-0LC31 / EEPROM	8x2^{10} Byte	211
6ES5 375-0LC41 / EEPROM	16x2^{10} Byte	212

Transferring the configuration data
You can transfer the data stored in the programmer with the transfer functions to any of the following:
- Diskette drive or hard disk drive
- PLC memory
- EPROM/EEPROM submodule

Transferring the control program and the configuration data to EPROM/EEPROM submodules

If you want to store the control program and the configuration data on EPROM/EEPROM submodules, you always require two identical memory submodules. Before you transfer your control program and the configuration data to the EPROM/EEPROM submodule, you must load this data into the programmer memory.

After you have transferred your user program and the configuration DB to both EPROM/EEPROMs, you must sign the EPROM/EEPROM submodules and enter the subunit ID. For this purpose, press F7 ”EPR. SIG”. The COM 115F software will prompt you for the subunit ID of the EPROM/EEPROM submodule and sign the EPROM/EEPROM submodule automatically.

Note

Please note that the order of blocks in the both EPROM/EEPROM submodules for subunit A and B must be identical so that the signatures are identical.
The COM 115F programming software requires 26 bytes of free memory in the memory submodule for entering the signature and other internal data.
After entering the signature you cannot transfer any further blocks to the EPROM/EEPROM submodule.

F4 Print
Press the F4 key to call the Print menu (Figure 1-18). You can print out your configuration DBs in plaintext using this menu. If the configuration DBs for PLC, FD or EPROM/EEPROM are to be printed out, they must first be loaded into the user RAM.

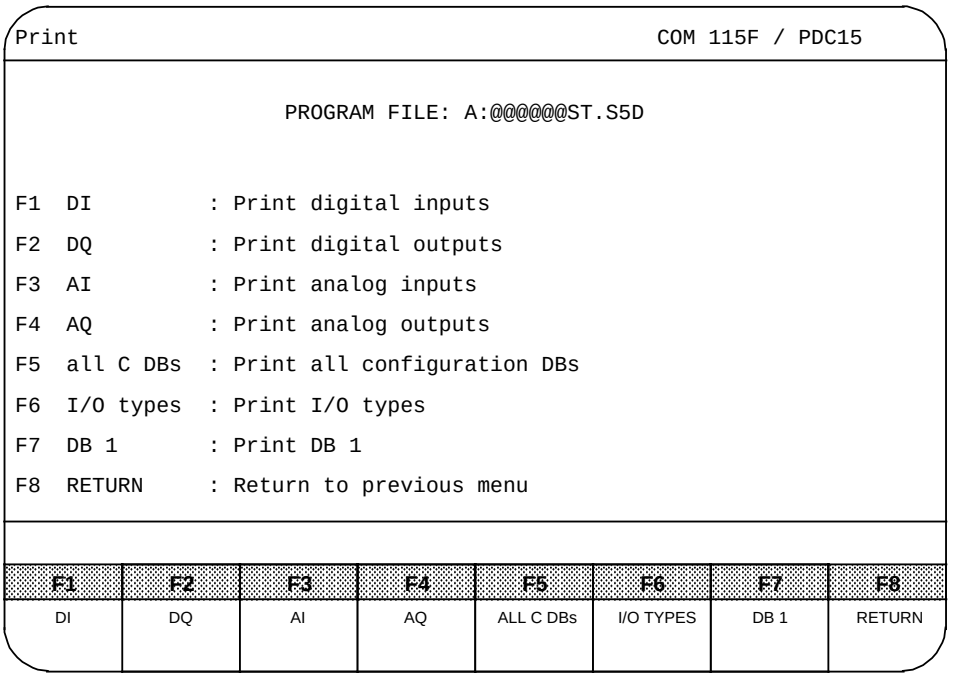


Figure 1-18. Menu for Printouts

Press keys F1 to F7 to execute the following functions:

F1 Print all DIs
All digital inputs are printed as type matrices corresponding to the I/O configuration DB 1.

F2 Print all DQs
All digital inputs are printed as type matrices corresponding to the I/O configuration DB 1.

F3 Print all AIs
All analog inputs are printed as type matrices corresponding to I/O configuration DB 1.

F4 Print all AQs
All analog outputs are printed as type matrices corresponding to I/O configuration DB 1.

F5 Print all configuration DBs
The entire configuration is printed out.

F6 Printing out a specific I/O type. All inputs/outputs of this type are printed out using this function. You will be prompted for the type number after pressing the F6 key. This is checked for accuracy after pressing <CR>. Wrong type numbers are rejected with the error message

”WRONG TYPE”.

If the type is not available, the following message appears:

”DB(S) NOT AVAILABLE”.

The function is aborted with the <Break> key.

F7 Print DB1
Using this function, you can print DB1 out together with information about the operating system, communications and mailbox table. The parameters are grouped according to function in the printout.

F5 Select program file
After pressing the F5 key in the System Handling menu, you can select a program file. If you access the hard disk or floppy before defining the program file, the ”B:@@@@@@ST.S5D” file will be accessed.

After selecting the program file, you can access the Delete, Configure, Transfer, Print etc. functions as often as you like on the floppy data blocks.

F7 Auxiliary functions
This key calls the STEP 5 auxiliary functions. For more detailed information, please see your programmer manual.

1.2 Configuring I/O Modules

1.2.1 General

There are configuration forms to help you in configuring the I/O modules of the S5-115F. The following is a more detailed description of the screen forms.

The type matrix (Vol. 1, 10.8 of the Manual) is designed to provide the user with an overview of his configured system. It is also useful in assigning I/O bits or I/O words to an I/O type.

Configuration screen form

In the case of digital I/Os, both bytes of the specified word are represented horizontally with bits 0 to 7 in each case.
Input words 0 to 126 or output words 0 to 124 are entered vertically.

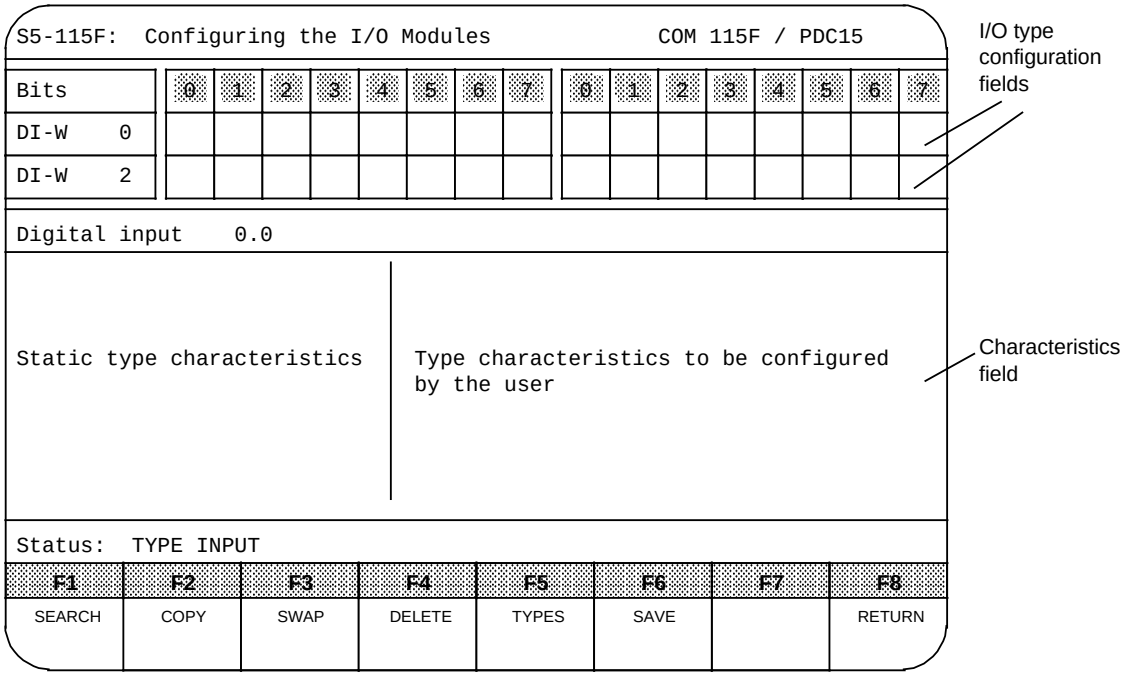


Figure 1-19. Initialization Form for Digital Modules

In the case of analog modules, the input/output word is located on the left side of the I/O type configuration field window and the type number on the right side.
I/O words 128 to 254 are listed vertically.

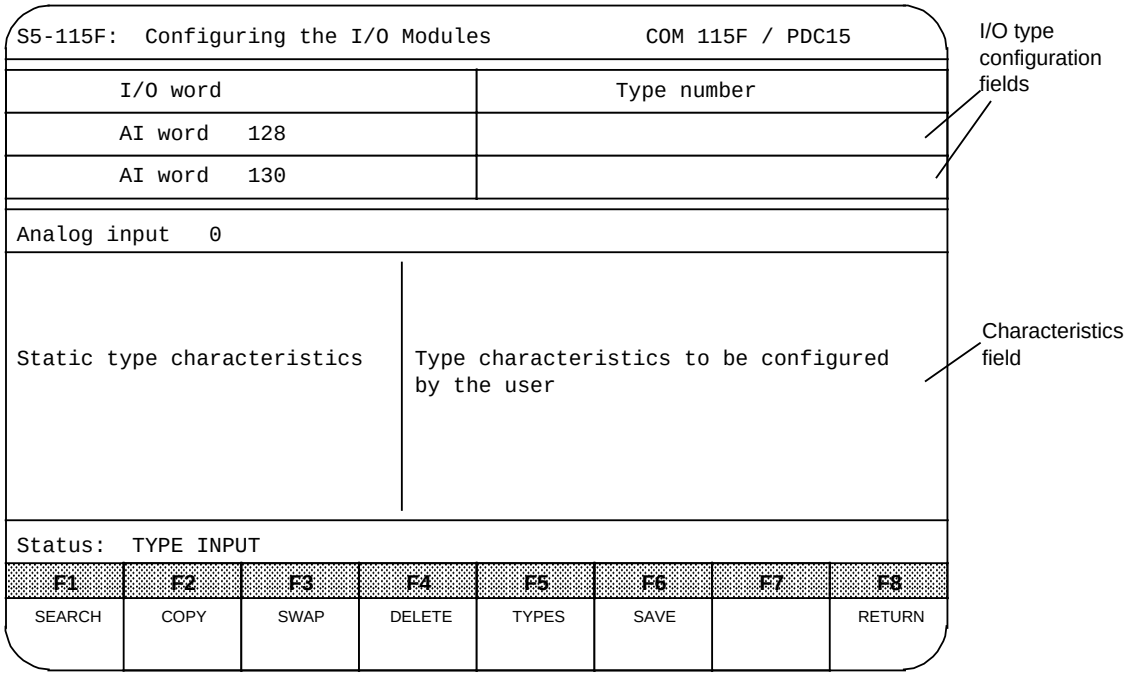


Figure 1-20. Initialization Form for Analog Modules

You can use the cursor control keys to position the cursor anywhere within the configuration menu.
If you exceed the bottom right edge of the matrix, the matrix is scrolled up and if you exceed the top left edge, the matrix is scrolled down.

I/O type configuration field

The I/O type number is entered at the cursor position and confirmed with <CR>.
A check is made after the entry of every type number and, in the event of an error, the message

”WRONG TYPE”

appears.

When a valid type has been entered, the cursor jumps to the characteristics field and from there returns to the next type matrix field.
You can overwrite a type already entered. Use the space key to delete a type.
The I/O type number depends on the I/O module and on the characteristics of the particular signal. The following relationship applies:

Digital inputs:	TYPE 1 to 3
Digital outputs:	TYPE 8 to 10
Analog inputs:	TYPE 13 to 16
Analog outputs:	TYPE 18

Plaintext Line

The plaintext line is assigned in each case to the I/O bit or word at which the cursor is currently located. At the beginning of the line is the identifier indicating the bit or word at which the cursor is located (e.g. digital input 2.3 or analog input 3). The plaintext display then follows in the form of eight characters of short plaintext and 40 characters of expanded plaintext. A typical line might read:

Example:
Digital input 2.3 VALVE 1 OPEN VALVE PUMP

The plaintext must have been generated using the available STEP 5 tools. If there is no plaintext available for a specific purpose, only the number of the I/O bit or word is displayed.

Characteristics Field

The characteristics field is the second window in the I/O configuration form. This field, too, is assigned to the current cursor position, i.e. the characteristics of the current type are displayed, provided it is set.
The characteristics field is divided into two parts:
The left part contains the fixed data (characteristics) allocated to a specific type. The right part contains the data to be initialized by the user.

The entries in the characteristics field are checked by the system for validity. When an invalid entry is made, the user receives the following messages:

- Double assignment of DBs with different types
  Message: DB assigned

- Double assignment of feedback addresses or any attempt to define an address assigned with a valid type as a feedback address.
  Message: e.g. DQ assigned

- Mixing types within one word
  Message: Type combination invalid

After entering the type, enter the DB number in the right-hand part of the characteristics field. Every entry must be confirmed with <CR>. All specifications are mandatory. The cursor leaves the entry field only when a valid value has been entered. After the last entry field has been filled, the cursor returns to the type matrix for the next I/O bit or word.

The characteristics field is deleted from the command line by pressing a softkey (1.2.2) and is used as a command data window. After the command has been executed, the characteristics field is again filled with the characteristics of the type at which the cursor is currently located.

Status and Error Line

The status and error line is used to display the current processing status and error messages. The status appears in the left-hand part of the line and the right-hand part is reserved for error messages.

1.2.2 Softkey Command Line

The softkey commands (Figure 1-21) offer functions such as Search, Copy, Swap, Delete etc. The various parameters must be set for these softkey routines. The type characteristics in the characteristics field are faded out for the purpose of entering this data. The designations of the individual parameters are now displayed in this window and you are expected to make a valid entry. After the selected function has been executed, the type characteristics are faded back in. The softkeys can only be used when the cursor is in the type matrix.

The following is a more detailed description of these functions.

F1	F2	F3	F4	F5	F6	F7	F8
SEARCH	COPY	SWAP	DELETE	TYPES	SAVE		RETURN

Figure 1-21. Softkey Commands of the Configuration Forms

F1 Search

This routine enables you to set a default value in the type matrix for a desired byte in digital modules and for a desired word in analog modules in the configuration screen form. Correspondingly, the following message appears in the command window:

BYTE NUMBER: or WORD NUMBER:.

You are then expected to enter a valid number (in the case of digital modules: DI byte numbers from 0, 1, 2, 3 to 127 or DQ byte numbers 0, 1, 2, 3 to 125; in the case of analog modules: word numbers from 128, 130 to 254). If the jump address is not within the valid range, the BYTE NUMBER INVALID or WORD NUMBER INVALID message is displayed in the error line and you will be returned to the basic position for entering the type number.
Press only <CR> to abort the function. When a valid entry is made, the relevant word appears in the type configuration field. In the case of digital modules, the cursor is located in the bit 0 column of the selected byte and, in the case of analog modules, it is located in the relevant type column. You can then continue entering the type.

F2 Copy

You can use this function to transfer the configuration data, one or more I/O words in the analog area and one or more bytes/bits in the digital area to other addresses. Each entry, which is checked for validity, must be confirmed with the EXEC key. Any discrepancies are indicated with an error message in the status line.

The first prompt to appear in the analog range is WORD(S) TO COPY: You can then enter two numbers, divided with a hyphen, for a word block.

In reply to the prompt "DEST. NO(S):", the system expects the destination address starting from which the word block or the word is to be stored. The destination end is calculated internally and displayed in the status line with the message "DEST. END AT WORD xx".

Please note the following rules when entering addresses:

- Word numbers may only contain even numbers, i.e. 0, 2, 4 to 254.
- The starting address must be smaller than the end address.
- The destination address must not lie within the word block to be copied.
- When copying word blocks, make sure that there are sufficient addresses available after the destination address.
- Before terminating an input, the cursor can be moved with the cursor control keys and the space bar can be used for deletion.
- If the space bar is used, the routine is aborted.
  If there is a feedback address within the address range (source and destination), the routine will be aborted with the message "FEEDBACK I/O".
- If a double assignment of feedback addresses arises due to copying or swapping, this is indicated in the following ways:
  - Type numbers in the type configuration field are displayed in inverse video.
  - A "?" is displayed in place of the feedback address.

Example 1

```
WORD (S) TO COPY 16-24
DEST. NO(S)      :  42
```

This causes the following transfer to be executed:

```
Word 16 ---> Word 42
Word 18 ---> Word 44
          :
          :
Word 24 ---> Word 50
```

Example 2

```
WORD(S) TO COPY: 16
DEST. NO(S)    : 20-30
```

This statement causes the following transfer to be executed:

```
Word 16 ---> Word 20
Word 16 ---> Word 22
          :
          :
Word 16 ---> Word 30
```

This gives words 16, 20, 22, 24, 26, 28 and 30 the same contents.

The ”COPY (W)ORD/(B)YTE?” prompt will appear to determine whether a word or byte is to be copied. The above applies if you select word. If you select byte, further prompts appear:
The system expects you to enter word numbers in response to the ”FROM BYTE(S)” and ”TO BYTE(S)” prompts. For DI modules, all numbers from 0 to 127 are permissible here, and for DQ modules, all numbers from 0 to 125 are permissible.

Additional bits can be transferred to other addresses in the case of digital input modules. It is also possible to specify the same number for both source and destination address, which means that it is possible to copy within one byte.
If only one byte is copied, the bit transfer prompt appears. This allows you to select the bits to be copied:

```
BIT TRANSFER:
TO BIT:0 1 2 3 4 5 6 7
FROM BIT:
```

The cursor appears in the ”FROM BIT” line. You can then enter a bit number using the keys <0> to <7>. Use the space bar to delete a number and the cursor control keys to select a bit position. If no bit number is entered, i.e. all the bit positions are unused, the whole line will be transferred.
Special care is required when copying bits within a byte. Do not enter a bit number which has already been copied to! It is best to use only one bit number during a copy procedure.

Example 3

```
WORD(S) TO COPY:    30
DEST. NO ?          45
BIT TRANSFER:
TO BIT:0 1 2 3 4 5 6 7
FROM BIT:<CR>
```

This causes the following to be copied:

```
Bit 30.0 ---> Bit 45.0
Bit 30.1 ---> Bit 45.1
Bit 30.2 ---> Bit 45.2
           :
           :
Bit 30.7 ---> Bit 45.7
```

If only <CR> is pressed in answer to any of the following prompts

”COPY (W)ORD/(B)YTE?”,
” WORD(S)/BYTE(S) TO COPY”,
”DEST. NO(S)”,

the copy routine will be aborted and you will be returned to type entry.

In order to avoid erroneous copy commands, the ”COPY?” message is displayed after completion of the last initialization. Press the <Exec> key to start, or press the <Break> key to abort, the Copy function. The program returns to the initial state for entering the type number.

F3 Swap

The parameters required for this function must be generated in the same way as those for the Copy function. The syntax for assigning the parameters is described in the same chapter. Byte-wise and bit-wise swapping is not possible.

The swap procedure is the same as the copy procedure. The prompts are as follows:

”SWAP WORD(S)
AND WORD(S)”

After the parameters have been set, the SWAP prompt appears. You can initiate the procedure by pressing the <Exec> key or you can abort the function with the <Break> key, and return to the initial state for type entry.

F4 Delete

This function enables the user to delete one or more successive bytes. Every input, which is checked internally for validity, is confirmed with <CR>. Any discrepancies are displayed in the status line with an error message.

The following prompt appears in the case of digital modules:

”DELETE (W)ORD/(B)YTE?”:

This prompt does not appear in the case of analog modules since only words can be deleted here. When deleting words, enter a single number or the block start and block end, separated by a hyphen, in answer to the ”DELETE WORD(S):” prompt. The block start must be less than the block end.
The numbers of words may only be even numbers from 0 to 254!

The following prompt appears in the case of digital modules when you select a byte with 'B':

”DELETE BYTE(S)”.

The rules for word numbers also apply to the input of byte numbers but for the fact that all integers between 0 and 127 can be used here. When you delete, you remove all bits in the byte from 0 to 7. In order to delete a single bit, you need only press the space bar when the cursor is on the type number.
After the parameters have been entered, the ”DELETE?” message appears. You can initiate execution of the function by pressing the <Exec> key, and you can abort the function by pressing the <Break> key, and returning to the initial state for type entry. If you press only <CR> in reply, the routine will be aborted and you will be returned to the initial state for type entry. This completes parameter setting in the analog area.

F5 Types

This softkey routine gives you an overview of the available type selection (1.2.1).
Starting from the initial state for type entry, the cursor is located in an element of the type configuration field. The system expects you to enter a valid type number at this position. At this point in the program, it is possible to transfer the types offered for selection together with their individual characteristics into the second window of the configuration form using the F5 softkey.
To do so, start with the lowest valid type number and fade in each subsequent type by pressing the <Types> function key.
When the last valid type has been reached, the first is displayed again.
You can enter further type numbers after each change in the type display.

A typical type and characteristics field is shown in Figure 1-22.

```
Type number           : 3       DB number        (4 to 255):       13
Num. of I/O channels : 2        Signal group      (0 to 28):        2
Safety-related        : Yes     Interrupt generating  (Y/N): N
Intermittent          : No      Inp. volt.         (24/220):       24 V
Feedback module required :      Discrepancy time      (S/L): S     30 msec.
     Check DQ 1/2 chann.        CH DQ Channel No.     (1/2):        2
                                CH DQ Byte addr. (0 to 125):      120
```

Figure 1-22. Characteristics Field of I/O Type 3

F6 Save

This function is used to save data from the configuration level on diskette or hard disk. If no program file has been defined in the System Handling menu or in the Defaults menu, the B:@@@@@FST.S5D program file is used. If this is the case, select the Handling menu with the <RETURN> key and enter the file name there. Finally, store the data.

1.2.3 Screen Forms for Configuring the I/O Modules

This chapter presents the configuration screen forms for the I/O modules.

Configuring the I/O modules means:

- Selecting an I/O type for each input or output
- Defining specific characteristics of these inputs and outputs

> **Note**
>
> Type mixes within one DI/DQ word are not permissible.

I/O Type 1

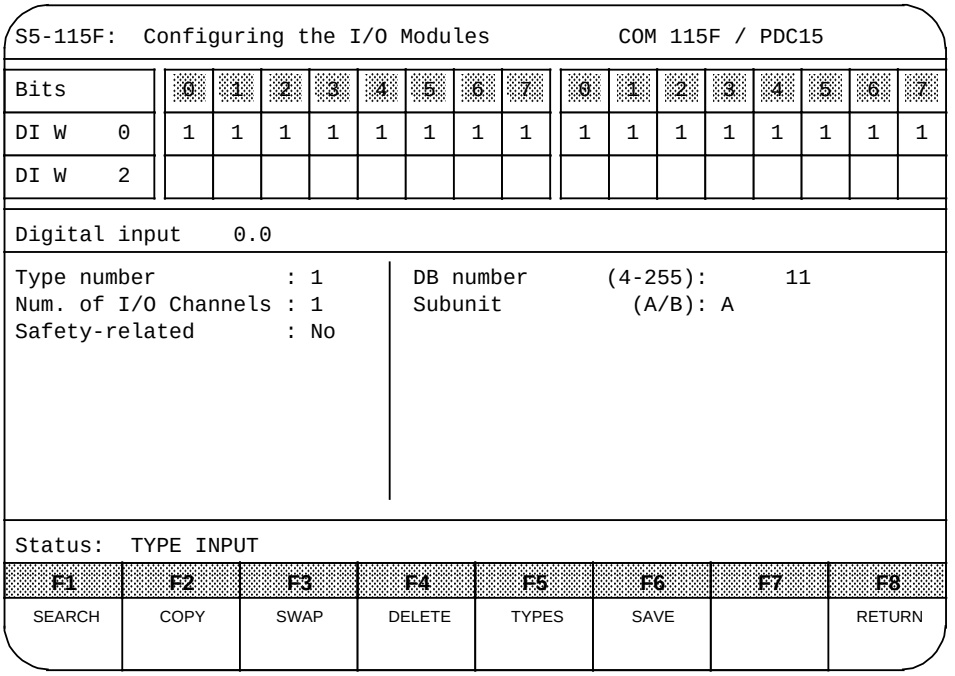


Figure 1-23. Initialization Form for I/O Type 1

- DB number
  COM 115F stores the configuration data for all DI bits of this type in this DB.

- The subunit definition applies to the whole input word.

I/O Type 2

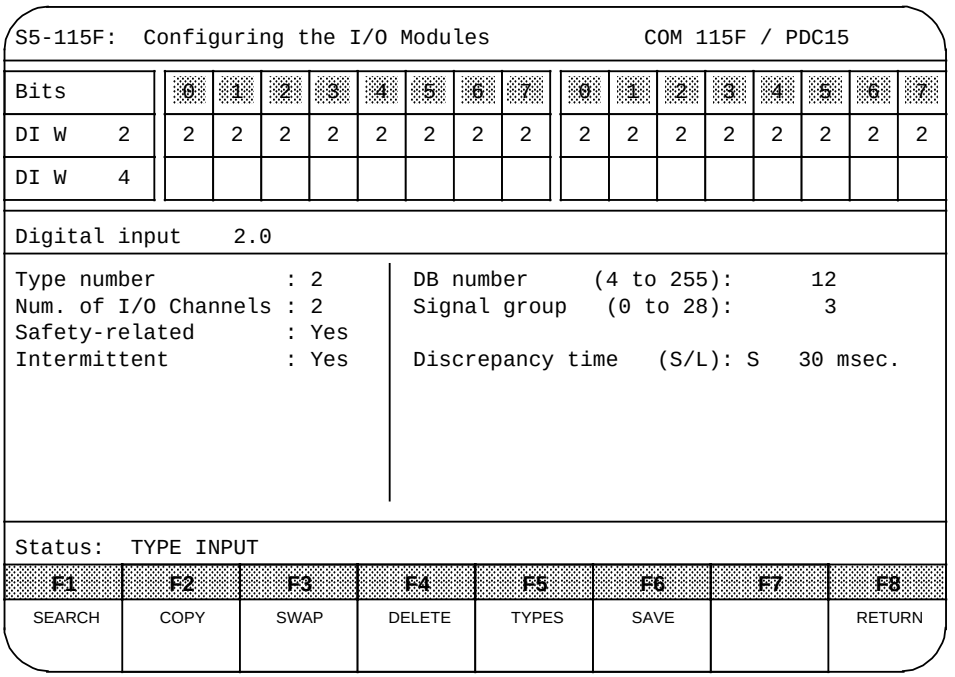


Figure 1-24. Initialization Form for I/O Type 2

- An intermittent signal must be subject to sufficiently frequent status changes detectable by the CPU. For this purpose, the signal must assume the states ”0” and ”1” for a period greater than the PLC cycle time at least once within the second error occurrence time.
- Signal group (Vol. 1, 10.17 of the Manual)
- Discrepancy time (Vol. 1, 10.6 of the Manual)
  Enter ”S” here for short and ”L” for long discrepancy times.
- Short and long discrepancy times may be mixed within one word.

I/O Type 3

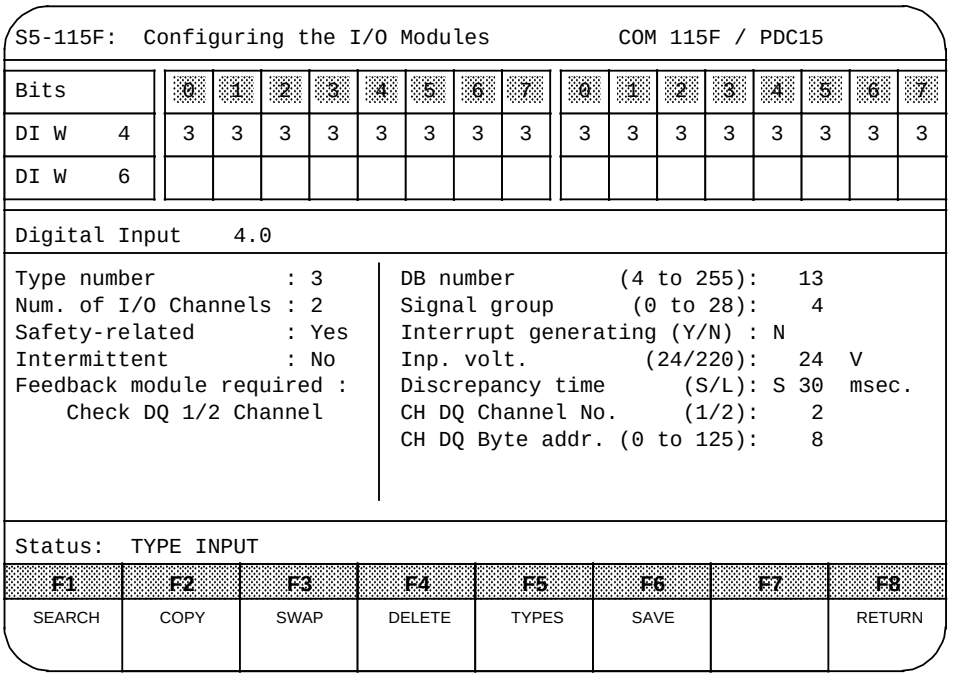


Figure 1-25. Initialization Screen Form for I/O Type 3

- Intermittent: no
  It cannot be guaranteed that the input signal will be subject to sufficiently frequent status changes detectable by the CPU. For this reason, the signal must assume the states ”0” and ”1” for a period greater than the PLC cycle time at least once within the second error occurrence time.

- Inp. volt.:
  Enter the supply voltage of the input module here. COM 115F requires the entry for calculating the test block time.

- Signal group (Vol. 1, 10.17)

- Discrepancy time:
  Specify here whether you want to enter the configured ”Short discrepancy time” or a ”Long discrepancy time”.

- DI modules and CH DQ modules are assigned to each other wordwise.
  Example:

DIs	2.0	2.1	2.2	2.3	2.4	2.5	2.6	2.7	3.0	3.1	3.2	3.3	3.4	3.5	3.6	3.7
CH DQs	8.0	8.1	8.2	8.3	8.4	8.5	8.6	8.7	9.0	9.1	9.2	9.3	9.4	9.5	9.6	9.7

I/O Type 8

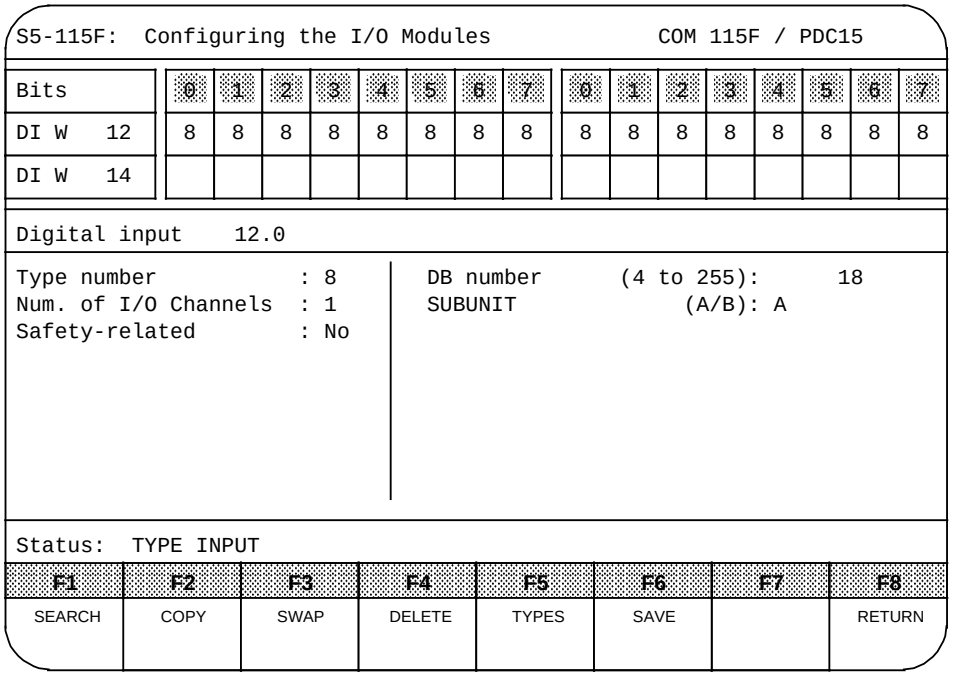


Figure 1-26. Initialization Form for I/O Type 8

- The definition of the subunit applies to the whole output word.

I/O Type 9

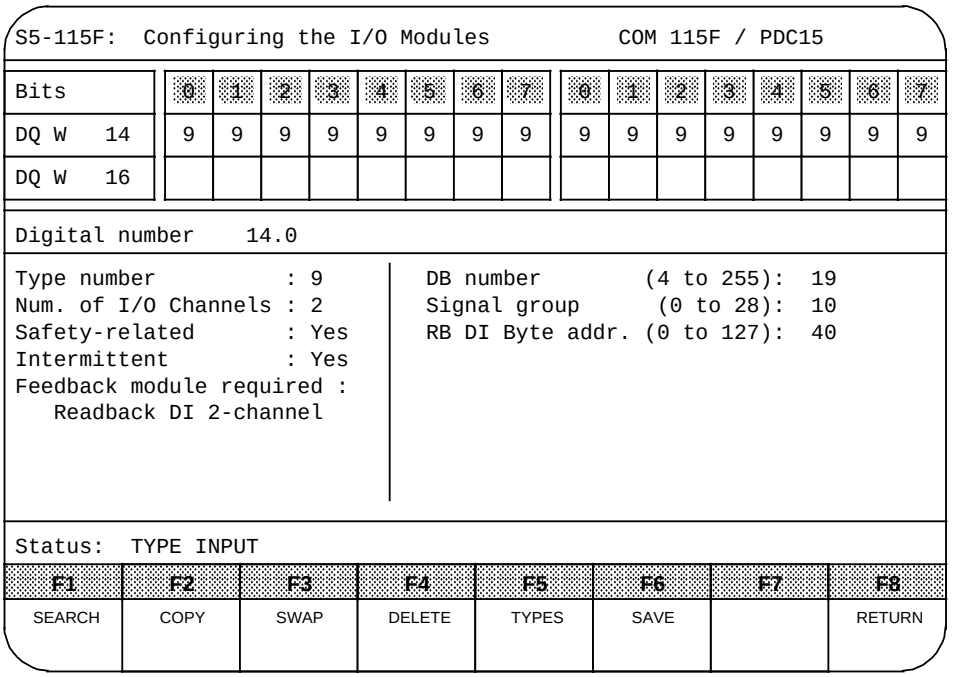


Figure 1-27. Initialization Form for I/O Type 9

- An intermittent signal must be subject to sufficiently frequent status changes detectable by the CPU. For this purpose, the signal must assume the states ”0” and ”1” for a period greater than the PLC cycle time at least once within the second error occurrence time.
- Only one access type and one signal group are permissible within one output word.
- DQ modules and R DI modules are assigned to each other wordwise.
  Example:

DQs	38.0	38.1	38.2	38.3	38.4	38.5	38.6	38.7	39.0	39.1	39.2	39.3	39.4	39.5	39.6	39.7
R DIs	40.0	40.1	40.2	40.3	40.4	40.5	40.6	40.7	41.0	41.1	41.2	41.3	41.4	41.5	41.6	41.7

Note

When using DI/DQ module 482, outputs and readback inputs must always have the same address.

- Signal group (Vol 1, 10.17 of the Manual)
  The following overview shows the relationship between error responses and the selected signal group.

I/O Type 10

S5-115F: Configuring the I/O Modules COM 115F / PDC15

Bits	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
DQ W 20	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
DQ W 22																

Digital output 20.0

Type number : 10
Num. of I/O Channels : 2
Safety-related : Yes
Intermittent : No
Feedback module required :
Readback DI 2-channel

DB number (4 to 255): 20
Signal group (0 to 28): 12
RB DI Byte addr. (0 to 127): 42

Status: TYPE INPUT

F1	F2	F3	F4	F5	F6	F7	F8
SEARCH	COPY	SWAP	DELETE	TYPES	SAVE		RETURN

Figure 1-28. Initialization Form for I/O Type 10

- An intermittent signal must be subject to sufficiently frequent status changes detectable by the CPU. For this purpose, the signal must assume the states ”0” and ”1” for a period greater than the PLC cycle time at least once within the second error occurrence time.

- Signal group (Vol. 1, 10.17 of the Manual)
  The signal group selected is always valid for the entire I/O word.

- DQ modules and R DI modules are assigned to each other wordwise.
  Example:

DQs	38.0	38.1	38.2	38.3	38.4	38.5	38.6	38.7	39.0	39.1	39.2	39.3	39.4	39.5	39.6	39.7
R DIs	40.0	40.1	40.2	40.3	40.4	40.5	40.6	40.7	41.0	41.1	41.2	41.3	41.4	41.5	41.6	41.7

Note

When using DI/DQ module 482, outputs and readback inputs must always have the same address.

I/O Type 13

Type 13 modules are used to configure
- Nonsafety-related analog inputs
- The CP 523 communications processor

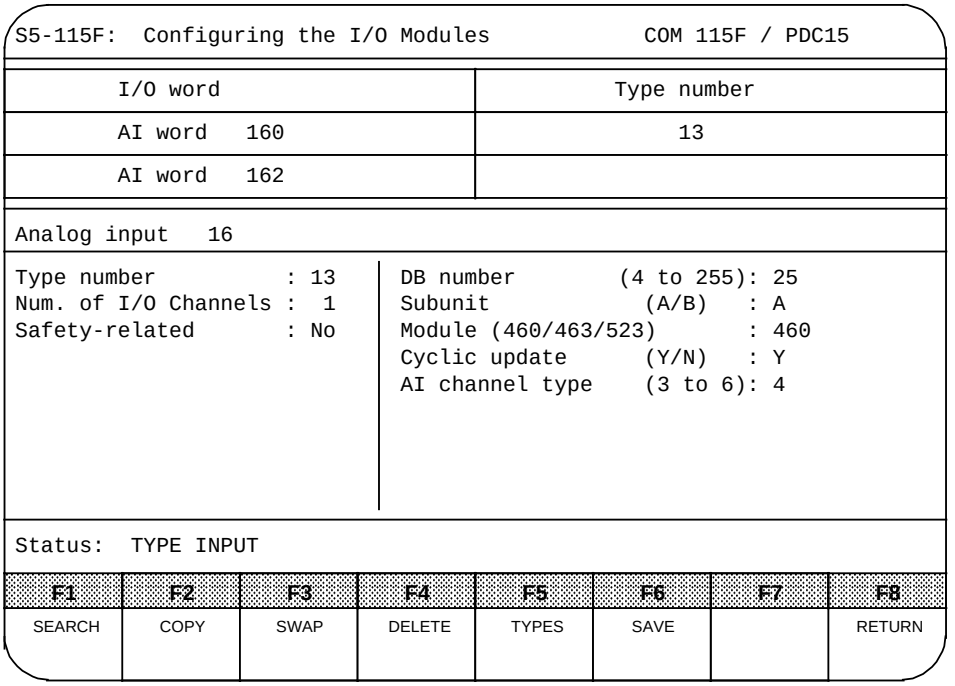


Figure 1-29. Initialization Form for I/O Type 13

- Cyclic update. Enter ”Y” here if you do not wish selective sampling.
- Module: Enter whether you want to configure a 460 AI module, a 463 AI module or a CP 523.
- Channel type

Table 1-5. Channel Types for Analog Input Modules (I/O Type 13)

	Module	Switch Position	Range Card (only for 460 AI)	Measuring Range	Nominal Range (Decimal Units)	CH AQ Module (463 only)
Channel type 3	460-...	500 mV/mA	1AA51 1AA71	4 to 20 mA	+512 to +2560	
	463-...	OFF	Measuring range programmable on the module	4 to 20 mA	+256 to +1280	---

Table 1-5. Channel Types for Analog Input Modules (I/O Type 13) (Continued)

	Module	Switch Position	Range Card (460 only)	Measuring Range	Nominal Range (Decimal Units)	CH AQ Module (463 only)
Channel type 4	460-...	500 mV/mA	1AA41 1AA11 1AA21 1AA31 1AA61	0 to 20 mA 0 to 500 mV 0 to 1 V 0 to 10 V 0 to 5 V	0 to +2048	
		50 mV	1AA41 1AA11 1AA21 1AA31 1AA61	0 to 2 mA 0 to 50 mV 0 to 100 mV 0 to 1 V 0 to 500 mV		
	463-...	ON	Measuring range pro-grammable on the module	4 to 20 mA	0 to +1024	---
Channel type 5	460-...	500 mV	1AA11 1AA21 1AA31 1AA61	- 500 to+500 mV - 1 to + 1 V - 10 to +10 V - 5 to +5 V	0 to +2048	
		50 mV	1AA11 1AA21 1AA31 1AA61	- 50 to +50 mV -100 to+100 mV - 1 to + 1 V - 500 to+500 mV		
	463-...	OFF	Measuring range pro-grammable on the module	0 to 20 mA 0 to 1 V 0 to 10 V	0 to +1024	---
Channel type 6	460-...	500 mV	1AA11 1AA21 1AA31 1AA61	- 500 to+500 mV - 1 to + 1 V - 10 to +10 V - 5 to +5 V	- 2048 to +2048	
		50 mV	1AA11 1AA21 1AA31 1AA61	- 50 to +50 mV - 100 to+100 mV - 1 to + 1 V - 500 to+500 mV		
	463-...	OFF	Measuring range pro-grammable on the module	0 to 20 mA 0 to 1 V 0 to 10 V	0 to +1024	---

I/O type 13

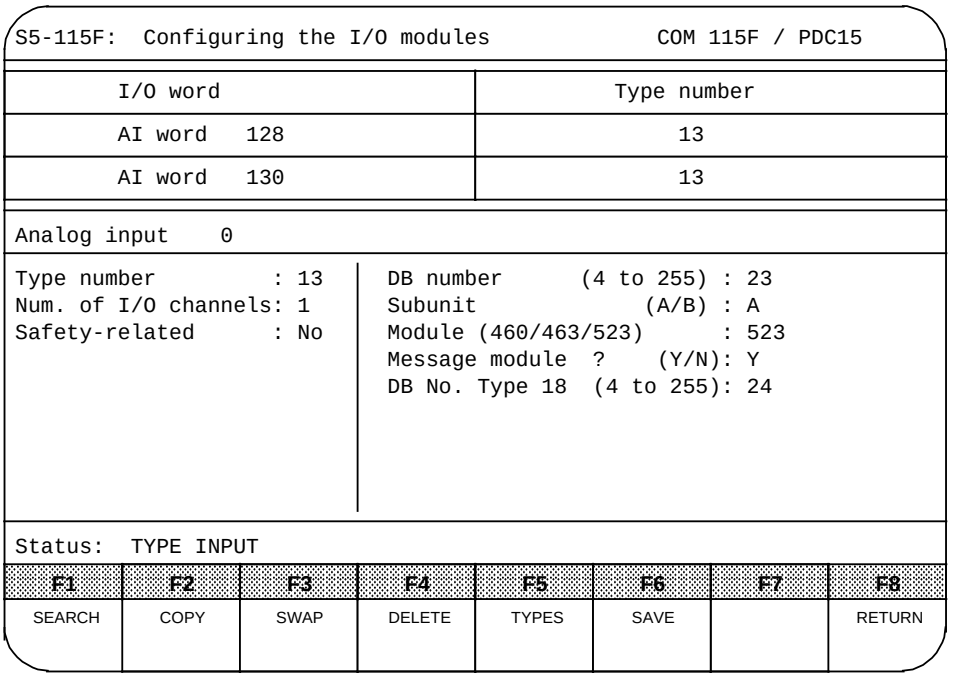


Figure 1-30. Initialization Form for CP 523 (I/O Type 13)

- The CP 523 uses the addresses of four AI words and four AQ words.
  You need only configure the AI for the initial address of the module. COM 115F handles further assignments of the AI and AQ words autonomously.

- The CP 523 can be used up to eight times in one S5-115F but only once as a message module for outputting operating system error messages.

I/O Type 14

```
S5-115F:  Configuring the I/O Modules              COM 115F / PDC15

        I/O word                          Type number
        AI word   192                          14
        AI word   194

Analog input    32

Type number          : 14  DB number       (4 to 255): 34
Num. of I/O channels :  2  Signal group     (0 to 28):  3
Num. of sensors      :  1  Discr. crit.: Rel. to mean value: R
Safety-related       : Yes Discrep. value (0 to 2048 / 1 to 100): 10 %
Intermittent         : No  Valid value  (1:min 2:max 3:mean):  2
Feedback module required:  CH AQ word (128 to 254): 240 Subunit :  A
   Check AQ 1-channel      CH DQ byte(0 to 125): 120 Half-byte(H/L): L
4x Check DQ 1-channel      Check value (KH) Lower: O   Upper :  1024
                           AI Channel type  (4 to 6): 4

Status:  TYPE INPUT

 F1      F2      F3      F4      F5      F6      F7      F8
SEARCH  COPY    SWAP    DELETE  TYPES   SAVE            RETURN
```

Figure 1-31. Initialization Form for I/O Type 14

- Signal group (Vol. 1, 10.17 of the Manual)
- Discrepancy value
  - If you have selected the ”Absolute” discrepancy criterion, enter the discrepancy value as a number up to 2048 units. The system responds with a discrepancy error if the deviation configured here is exceeded.
  - If you have selected the ”Relative” discrepancy criterion, enter the maximum permissible relative deviation (in %) as a decimal number. The system responds with a discrepancy error if the percentage deviation configured here ***and*** the ”minimum AI deviation” configured when ”assigning the operating system parameters” are exceeded.

The relative deviation always refers to the value which you have selected for standard value generation according to the regulations. You can refer the relative deviation to the minimum, the mean or the maximum of the values read into subunits A and B.
The following relationships exist between the three types of relative deviation:
(the equations apply to A > B)

The following applies to relative deviation referred to the maximum value:

$$r_{max.\ value} = \frac{A - B}{A}$$

The following applies to relative deviation referred to the minimum value:

$$r_{min.\ value} = \frac{A - B}{B}$$

The following applies to relative deviation referred to the mean value:

$$r_{mean\ value} = \frac{A - B}{(A+B)}$$

where

$r_{max.\ value}$ = Relative deviation referred to the maximum value
$r_{min.\ value}$ = Relative deviation referred to the minimum value
$r_{mean\ value}$ = Relative deviation referred to the mean value
A = Read value in subunit A
B = Read value in subunit B

Example:
The read value of the signal in subunit A is 100 units and, in subunit B, 70 units.

Therefore:

$$r_{max.\ value} = \frac{A - B}{A} = \frac{100 - 70}{100} \hat{=} 30\ \%$$

$$r_{min.\ value} = \frac{A - B}{B} = \frac{100 - 70}{70} \hat{=} 42.9\ \%$$

$$r_{mean\ value} = \frac{A - B}{(A+B)} = \frac{100 - 70}{(100+70)} \hat{=} 35.3\ \%$$

- Valid value
  Enter here how the unit value is to be generated from the values read into subunit A and B.

> **Note**
>
> If you have initialized software wire-break detection in FB 250 ANEI (6.1.3), you must enter ”1” here.

- CH DQ byte/Half-byte (H/L): This assignment defines four bits for switching the analog input module between sensor and check output module.

- Lower/Upper check value: Enter the lower and upper check values from the raw value range of your analog signal here. The check values must be between 0 and 1535 (Vol. 1, 10.11.4 of the Manual).

> **Note**
>
> The check values have safety characteristics. You must check the configured values against a printout generated with COM 115F DOCUMENT.

- AI channel type

Table 1-6. Channel Types for Analog Input Modules (I/O types 14 and 15)

	Module	Switch Position	Range Card (460 only)	Measuring Range	Nominal Range (Decimal Units)	CH AQ Module (463 only)
Channel type 4	463-...	ON	Measuring range programmable on the module	4 to 20 mA	0 to +1024	470-7LC12
Channel type 5	463-...	OFF	Measuring range programmable on the module	0 to 20 mA 0 to 10 V	0 to +1024	470-7LA12 470-7LA/B12
Channel type 6	463-...	OFF	Measuring range programmable on the module	0 to 20 mA 0 to 10 V	0 to +1024	470-7LA12 470-7LA/B12

I/O Type 15

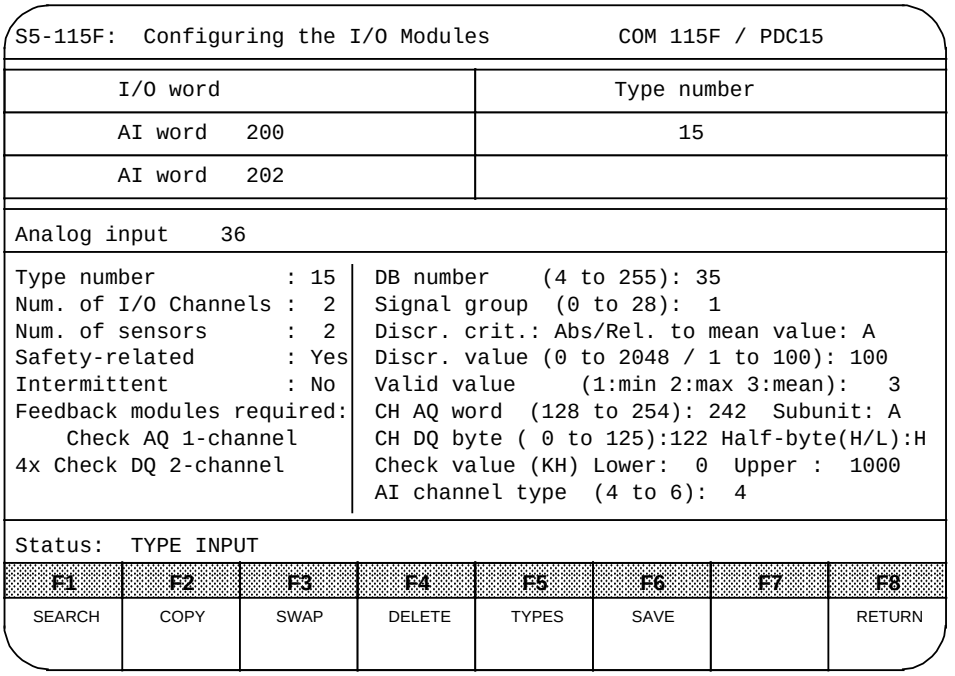


Figure 1-32. Initialization Form for I/O Type 15

- Signal group (Vol. 1, 10.17 of the Manual)
- AI channel type (Table 1.6)
- Lower/Upper check value: Enter the lower and upper check values from the raw vale range of your analog signal here. The check values must be between 0 and 1535.
  (Vol. 1, 10.11.4 of the Manual)

I/O Type 16

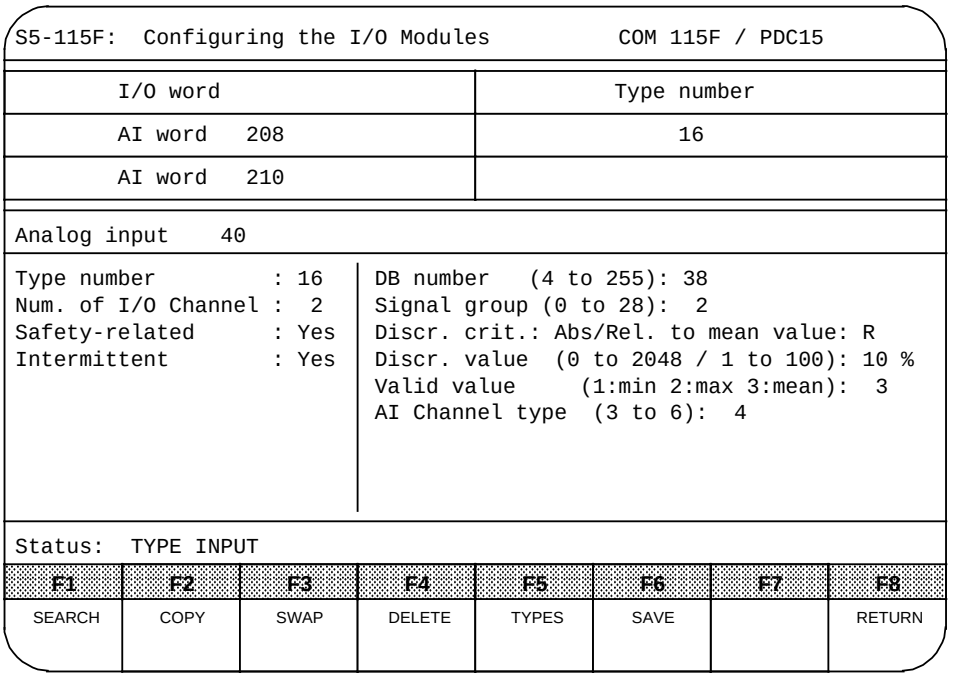


Figure 1-33. Initialization Form for I/O Type 16

- Intermittent: Proof of an intermittent AI signal is not elementary.
  An AI signal is intermittent if the whole value range relevant to the evaluation is run through, read and coded at least once within the second error occurrence time.
  With reference to the coding, this means that all input bits must appear at both levels ”0” and ”1”.

- Signal group (Vol. 1, 10.17 of the Manual)

- AI channel type

Table 1-7. Channel Types for Analog Input Modules (I/O Type 16)

	Module	Switch Position	Range Card (460 only)	Measuring Range	Nominal Range (Decimal Units)	PR AQ Module (463 only)
Channel type 3	463-...	OFF	Measuring range programmable on the module	4 to 20 mA	+256 to +1280	---
Channel type 4	463-...	ON	Measuring range programmable on the module	4 to 20 mA	0 to +1024	---
Channel type 5	463-...	OFF	Measuring range programmable on the module	0 to 20 mA 0 to 1 V 0 to 10 V	0 to +1024	---
Channel type 6	463-...	OFF	Measuring range programmable on the module	0 to 20 mA 0 to 1 V 0 to 10 V	0 to +1024	---

I/O Type 18

```
S5-115F:  Configuring the I/O Modules              COM 115F / PDC15

        I/O word                          Type number
        AQ word   192                           18
        AQ word   194

Analog output    32

Type number         : 18      DB number (4 to 255): 24
Num. of I/O Channel :  1      Subunit        (A/B): A
Safety-related      : No

Status:  TYPE INPUT
```

F1	F2	F3	F4	F5	F6	F7	F8
SEARCH	COPY	SWAP	DELETE	TYPES	SAVE		RETURN

Figure 1-34. Initialization Form for I/O Type 18

1.3 Documentation with COM 115F

The program section of COM 115F contains the following packages:
- COM 115F CONFIGURE
- COM 115F DOCUMENT for
  - Safety-related readout of configured data from the PLC
  - Printout of the data in the programmer.

> **Note**
>
> For the individual Licensing Authority acceptance tests, it is essential that you are able to print out the configuration data with COM 115F DOCUMENT and that you can compare the printout with the configured data.

Proceed as follows to start COM 115F DOCUMENT:

Select the desired menu from the ”Package selection” menu using the cursor control keys:
- COM 115F DOCUMENT

Confirm your choice with the F1 ”PACKAGE” key.
The package will appear with the ”DEFAULTS” screen form (Figure 1-2).

Confirm the defaults with the F6 ”EXEC” key.
The main menu will then appear.
If you also press the HELP key, the following form appears on your screen.

```
PRINTING THE S5 115F PARAMETERS                    COM 115F / PDC15

                 PROGRAM FILE:A:@@@@@@ST.S5D

F1 ALL C DB  :  Print all configuration DBs

F2 I/O TYPES :  Print a specific I/O type

F3 DB 1      :  Print DB 1

F4           :

F5           :

F6           :

F7           :

F8 RETURN    :   Exit COM 115F DOCUMENT
```

F1	F2	F3	F4	F5	F6	F7	F8
ALL C DB	I/O TYPES	DB 1		PRG DAT	DEFAULTS	AUX	RETURN

Figure 1-35. Main Menu of the Printout Package

Only the configuration DBs of a logical device (PLC, FD or EPROM) can be printed out, i.e. the configuration must be available on the relevant device. The footer can also be printed out.

The following functions will be executed if you press F1 to F6:

F1 The whole configuration is printed out (F2+F3). Comprehensive documentation is generated.

F2 This function can be used to print out a specific I/O type. After the type has been selected, all the inputs of this type are printed out. This takes the form of a table of the I/O modules (Figure 1-36). The short text is printed from the plaintext line. When you press the F6 key, the prompt

”ENTER TYPE NO.”

will appear in the command line. The system then expects input of the type number. When you press the <CR> key, this entry will be tested for validity and the error message

”WRONG TYPE”

may be displayed, if applicable. If the type is not available, the message

”DB(S) NOT AVAILABLE”

will appear.

If you press <Break>, the function will be aborted.

F3 Print DB1
Use this function to print out DB 1 with information about the operating system, communications and mailbox table. The parameters are grouped according to function in the printout.

F5 Select program file
After pressing the F5 key in the Handling menu, the user can select a program file on the diskette or hard disk. If the program file is accessed before it has been declared, B:@@@@@@ST.S5D will be accessed.

F6 Default
The ”Default” menu appears again here (Figure 1-2).

F7 Auxiliary functions
This key calls the STEP 5 auxiliary functions (overlay file). See your programmer manual for further details.

The following figure shows a typical I/O type 3 configuration printout:

COM 115F DOCUMENT CPU 942-7UF13
I/O Type 3 : Double-channel non-intermittent digital inputs

DI	Symbol	Signal Group	Interrupt Genera-ting	Input Voltage/V	Discrepan-cy Time/ msec.	CH DQ	Sub-unit
4.0		4	NO	24	K 30	8	A, B
4.1		4	NO	24	K 30	8	A, B
4.2		4	NO	24	K 30	8	A, B
4.3		4	NO	24	K 30	8	A, B
4.4		4	NO	24	K 30	8	A, B
4.5		4	NO	24	K 30	8	A, B
4.6		4	NO	24	K 30	8	A, B
4.7		4	NO	24	K 30	8	A, B
5.0	VALVE 1	4	NO	220	L 500	9	A, B
5.1	VALVE 2	4	NO	220	L 500	9	A, B
5.2	VALVE 3	4	NO	220	L 500	9	A, B
5.3		4	NO	220	L 500	9	A, B
5.4		4	NO	220	L 500	9	A, B
5.5		4	NO	220	L 500	9	A, B
5.6		4	NO	220	L 500	9	A, B
5.7		4	NO	220	L 500	9	A, B
9.0		10	YES	24	K 20	33	A
9.1		10	YES	24	K 20	33	A
9.2		10	YES	24	K 20	33	A
9.3		10	YES	24	K 20	33	A
9.4		10	YES	24	K 20	33	A
9.5		10	YES	24	K 20	33	A
9.6		10	YES	24	K 20	33	A
9.7		10	YES	24	K 20	33	A
70.0		28*	NO	220	K 30	80	B
70.1		28*	NO	220	K 30	80	B
70.2		28*	NO	220	K 30	80	B
70.3		28*	NO	220	K 30	80	B
70.4		28*	NO	220	K 30	80	B
70.5		28*	NO	220	K 30	80	B
70.6		28*	NO	220	K 30	80	B
70.7		28*	NO	220	K 30	80	B
71.0		28*	NO	220	K 30	81	B
71.1		28*	NO	220	K 30	81	B
71.2		28*	NO	220	K 30	81	B
71.3		28*	NO	220	K 30	81	B
71.4		28*	NO	220	K 30	81	B
71.5		28*	NO	220	K 30	81	B
71.6		28*	NO	220	K 30	81	B
71.7		28*	NO	220	K 30	81	B

* = Caution: I/O error tolerance variant 3

Figure 1-36. Printing the I/O Type 3 for CPU 942-7UF13 with Signal Group 28

Note

COM 115F prints an additional warning when using
- I/O-ETV 3 with signal group 28
- I/O-ETV 4 with signal group 27 and 28

1.4 Printing out Files Stored on the EPROM Submodule

From COM 115F Version 3.1 onwards, the configuring data from an EPROM submodule are printed out with a footer containing information on the creation date, reason for the update and the author.

You store the necessary data when generating the signature with COM 115F CONFIGURE (Fig. 1-37).

COM 115F requires 62 free bytes on the EPROM for entering the information.

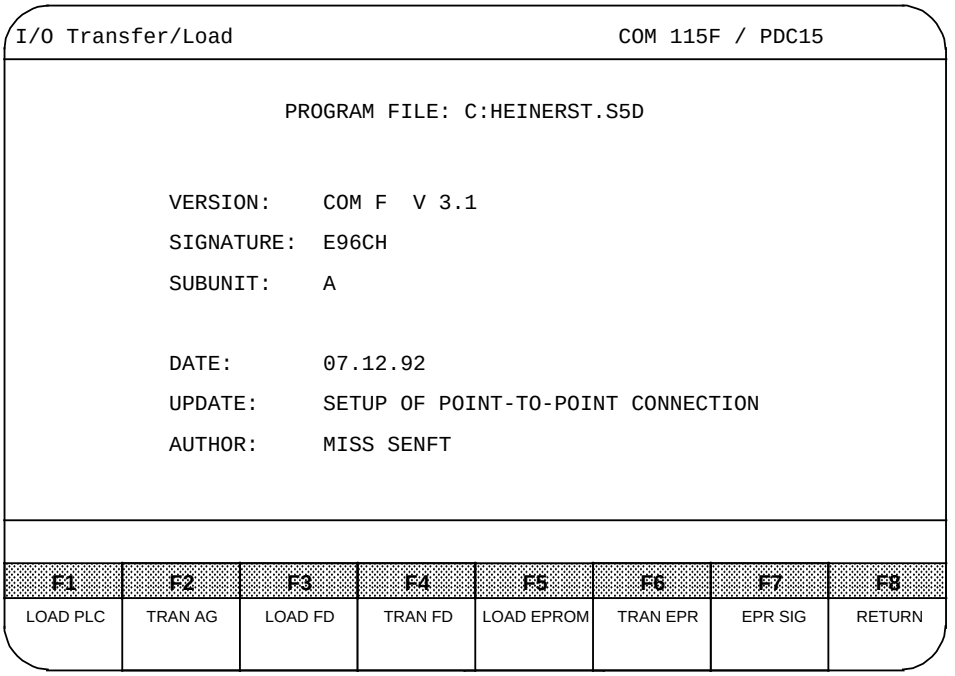


Fig. 1-37. Menu for Generating the Signature

1.5 Configuration Error Messages

The following tables contain error messages that may appear in conjunction with COM 115F CONFIGURE.

Table 1-8. COM 115F Configuration Error Messages

Error Test	Cause of Error	Explained in
NOT ENOUGH MEMORY	Error when loading the auxiliary functions	1.1.7
NO HELP TEXT AVAILABLE	File with Help texts missing	1.1, 1.2
ERROR TEXT CANNOT BE READ	File with F error texts missing	1.1.5
INITIALIZATION NOT PERMISSIBLE	Occurs when reading in parameters (BESY, COMMUN, type characteristics)	1.1, 1.2.1
ONLY 255 LONG DISCREP. TIMES	Only 255 long discrepancy times are permissible	1.2.1
ONLY EVEN VALUES PERMISSIBLE	Appears when entering word numbers	1.2.1
ONLY 3-DIGIT VALUE FIRST VALUE INVALID SECOND VALUE INVALID FIRST VALUE > SECOND VALUE	Appears when entering number ranges	1.1.7, 1.2.2 1.1.7, 1.2.2 1.1.7, 1.2.2 1.1.7, 1.2.2
FEEDBACK I/O: ABORT DOUBLE-ASSIGNED CHECKBACK	Exchange, copying and deleting: Feedback modules for 2 I/O modules assigned	1.2.2 1.2.2
WRONG TYPE	Invalid type number for this I/O range	1.2.1
DB NUMBER ASSIGNED	Selection of the number of an configuration DB already assigned	1.1.2, 1.2.1
I/O ASSIGNED	I/O to which the feedback module is connected is already assigned	1.2.2
OCCUPIED BY INTERRUPT-GENERATING MODULE	Output or check output occupied by configuration output or interrupt generating module	1.2.1

Table 1-8. COM 115F Configuration Error Messages (Continued)

Error Text	Cause of Error	Explained in
SUBUNIT NOT YET DEFINED	Subunit identifier not yet in PLC	1.1
SOURCE WITHIN DESTINATION DESTINATION WITHIN SOURCE INVALID BYTE NUMBER INVALID WORD NUMBER	Invalid ranges or numbers for searching, copying, exchanging and deleting	1.2.2 1.2.2 1.2.2 1.2.2
WRONG PLC MODE	The ISTACK in the PLC cannot be read	1.1.5
NO ERROR FOUND	No error found during F error output	1.1.5
DISKETTE ERROR		1.1.7, 1.2.2
FILE NOT AVAILABLE	Occurs when reading from or writing to diskette	1.1.7
NO CONNECTION TO PLC	Occurs when reading or writing from PLC	1.1.7
DB NOT AVAILABLE		1.1.7
NO CONFIGURATION DB WRONG CONFIGURATION DB	Configuration DB missing or wrong	1.1.7 1.1.7
EPROM ALREADY SIGNED	Attempt to sign a signed EPROM	1.1.7
ILLEGAL TYPE MIX WRONG TYPE MIX IN FEEDBACK	Configuration rules have not been followed	1.1.7, 1.2 1.1.7, 1.2
ONLY 15 NODES PERMITTED	Occurs if an attempt is made to configure a mailbox table for the 16th node in a redundant SINEC L1 LAN	1.1.2
ILLEGAL: OVERLAP	Overlap of the mailboxes of several nodes	1.1.2
FLAGS BEYOND 255	Flag address+mailbox > 255	1.1.2
MESSAGE XY	PLC interface error message	Refer to your programmer manual.

Table 1-9. EPROM Function Error Messages

No.	Error Text
101_H	EPROM HAS NO IDENTIFIER
102_H	RESULT OF COMPARISON NOT EQUAL
103_H	INVALID EPROM CONTENTS
104_H	WRONG MODE
105_H	WRONG FRAME LENGTH
106_H	WRONG SCREEN LINE
107_H	WRONG SCREEN COLUMN
108_H	ILLEGAL PROGRAMMER NUMBER
109_H	UNKNOWN EPROM
$10A_H$	EPROM NOT ERASED
$10B_H$	NO EEPROM
$10C_H$	WRONG DISKETTE
$10D_H$	ERROR IN FILE
$10E_H$	HARDWARE ERROR
$10F_H$	ERROR IN DISKETTE/DRIVE
110_H	EMPTY EPROM
111_H	DUPLICATED BLOCK
112_H	DIRECTORY OVERFLOW
113_H	DRIVER CALL UNSUCCESSFUL
114_H	BLOCK NOT AVAILABLE
115_H	BLOCK ALREADY EXISTS
116_H	TYPE NOT IN DIRECTORY
117_H	ILLEGAL CALL
118_H	DIRECTORY FULL
119_H	ILLEGAL SYS-ID WRITE
$11A_H$	SUBMODULE NOT PROVIDED
$11B_H$	SYS-ID NOT FOUND
$11C_H$	BUFFER TOO SMALL
$11D_H$	SYS-ID BLOCK WRITE FORBIDDEN

Table 1-10. Error Messages Occurring When Using Diskettes

No.	Error Text
201_H	ILLEGAL DRIVE
202_H	DEFAULT DIRECTORY NOT DEFINED
203_H	DISKETTE/HARD DISK FULL
204_H	DISKETTE ERROR
205_H	DISKETTE WRITE-PROTECTED
206_H	BLOCK TOO LONG FOR PG BUFFER
207_H	SUBDIRECTORY LOWER THAN LEVEL 2
208_H	FILE NOT AVAILABLE
209_H	FILE ALREADY EXISTS
$20A_H$	MORE THAN 255 ENTRIES IN SUBDIRECTORY
$20B_H$	FILE CANNOT BE DELETED, SUBDIRECTORIES
$20C_H$	FILE WRITE-PROTECTED
$20D_H$	BLOCK OR TEXT ELEMENT MISSING
$20E_H$	BLOCK OR TEXT ELEMENT ALREADY EXISTS
$20F_H$	COMF ERROR (WRONG TYPE OF DIRECTORY)
210_H	FILE CANNOT BE INTERPRETED
211_H	WRONG FILE NAME

Table 1-11. Error Messages to the PLC Interface

No.	Error Test
301_H	PLC STOP
302_H	PLC MEMORY NOT OCCUPIED
303_H	PLC TIME OVERFLOW
304_H	PLC INTERFACE IN PROGRAMMER NOT READY
306_H	UNDEFINED CHARACTER FROM THE AS
307_H	WRONG OPERATING MODE IN PLC
309_H	MEMORY ERROR IN PLC
$30A_H$	OVERFLOW IN DATA EXCHANGE WITH PROGRAMMER
$30B_H$	AS STATUS CANNOT BE INTERPRETED
$30C_H$	SEND LINE TO AS INTERRUPTED

2 Introduction to STEP 5

2.1 Writing a Program . . . . . 2 - 1
2.1.1 Methods of Representation . . . . . 2 - 1
2.1.2 Operand Areas . . . . . 2 - 3
2.1.3 Circuit Diagram Conversion . . . . . 2 - 4

2.2 Program Structure . . . . . 2 - 4
2.2.1 Linear Programming . . . . . 2 - 4
2.2.2 Structured Programming . . . . . 2 - 5

2.3 Block Types . . . . . 2 - 6
2.3.1 Organization Blocks (OBs) . . . . . 2 - 7
2.3.2 Program Blocks (PBs) . . . . . 2 - 13
2.3.3 Sequence Blocks (SBs) . . . . . 2 - 13
2.3.4 Function Blocks (FBs) . . . . . 2 - 14
2.3.5 Data Blocks (DBs) . . . . . 2 - 18

2.4 Modifying the Program . . . . . 2 - 20

2.5 Number Representation . . . . . 2 - 20

Figures

2-1. Compatibility of STEP 5 Methods of Representation . . . . . 2 - 2
2-2. Nesting . . . . . 2 - 6
2-3. Structure of a Block Header . . . . . 2 - 7
2-4. Example of Organization Block Use . . . . . 2 - 9
2-5. Setting the Restart Characteristics . . . . . 2 - 12
2-6. Programming a Function Block with Block Parameters . . . . . 2 - 15
2-7. Programming a Function Block . . . . . 2 - 18
2-8. Data Block Contents . . . . . 2 - 19
2-9. Validity Areas of Data Blocks . . . . . 2 - 19
2-10. Bit Assignment of a 16-Bit Fixed-Point Binary Number . . . . . 2 - 20

Tables

2-1. Permissible Methods of Representation in the Blocks . . . . . 2 - 3
2-2. Comparison of Block Types . . . . . 2 - 6
2-3. Overview of Organization Blocks . . . . . 2 - 8
2-4. Block Parameter Types and Data Types with Permissible Actual Operands . . . . . 2 - 16
2-5. Examples of Number Representation in the PLC . . . . . 2 - 20

2 Introduction to STEP 5

This chapter explains how to program the S5-115F. It describes how to write a program, how the program is structured, the types of blocks the program uses, and the number representation of the STEP 5 programming language.

Safety-oriented system configuration

Before you start actual programming, the following safety-oriented procedures should be observed:

- transfer of safety-oriented operating system parameters (e.g. ”Maximum PLC cycle time”),
- organization and transfer of I/O configuration:
  Assignment of I/O types to the reserved I/O addresses and initialization of these I/O types
  (e.g. address of the readback digital input module)
- organization and transfer of the SINEC L1 structure to the operating system with the help of the COM 115F programming package.

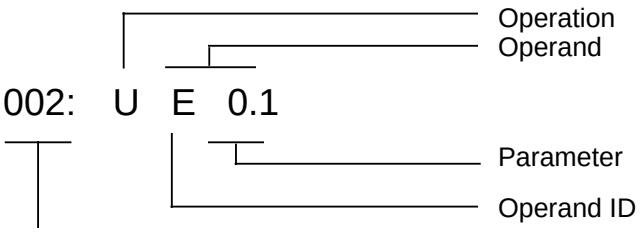
2.1 Writing a Program

A control program specifies a sequence of operations that tell the programmable controller (PLC) how it has to control a system. You must write the program in a special language and according to specific rules so that the PLC can ”understand” it. The standard programming language that has been developed for the SIMATIC S5 family is called STEP 5.

2.1.1 Methods of Representation

The following methods of representation are possible with the STEP 5 programming language:

- **Statement list (STL)**
  STL represents the program as a sequence of operation mnemonics. A statement has the following format:



  The operation instructs the PLC what to do with the operand. The parameter indicates the operand address.

- C**ontrol system flowchart (CSF)**
  CSF represents logic operations with symbols.

- **Ladder diagram (LAD)**
  LAD represents control functions with circuit diagram symbols.

Each method of representation has its special characteristics. Consequently, a program block that has been programmed in STL cannot necessarily be output in CSF or LAD form. The graphic representations are not compatible to each other either. However, programs in CSF or LAD can always be converted to STL. Figure 2-1 illustrates these points in a diagram.

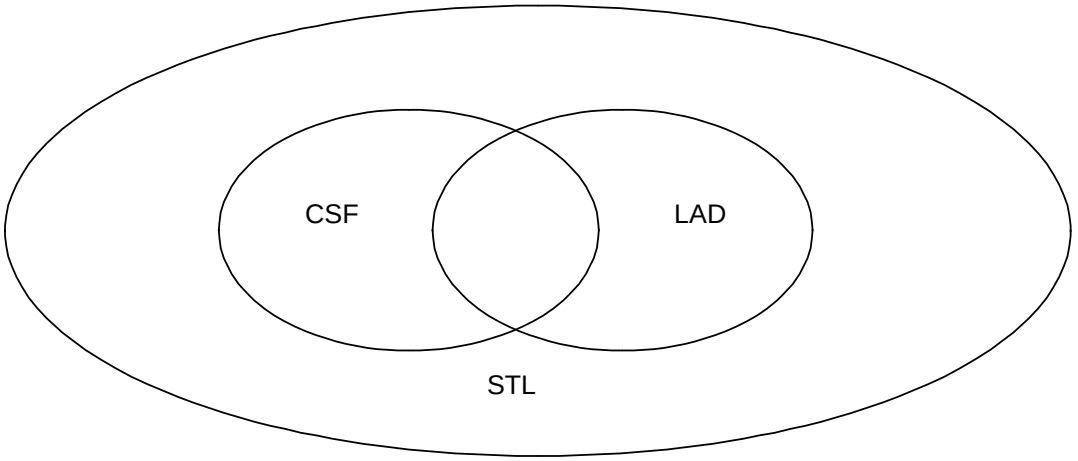


Figure 2-1. Compatibility of STEP 5 Methods of Representation

Note

During the individual system acceptance test of the safety-related S5-115F, the inspector also checks your program. For this reason, the user program must be simply structured and easy to read.
Write your programs mainly in the CSF form of representation.
Operations which cannot be represented as CSF or LAD must be written in STL networks.
Divide your OBs, PBs and SBs into:
- CSF or LAD networks
  and
- STL networks

The STEP 5 programming language has the following three operation types:

- Basic operations
- Supplementary operations
- System operations

Table 2-1 provides further information on these operations.

Table 2-1. Permissible Methods of Representation in the Blocks

Operations	Method of Representation						
	LAD/CSF possible in			STL possible in			
	OB	PB	SB	FB	OB	PB	SB
Basic operations 1. Logic operations 2. Set/Reset operations 3. Load and transfer operations 4. Timer operations 5. Counter operations 6. Comparison operations	Yes			Yes			
7. Arithmetic operations 8. Block call operations 9. Other operations 10. Display construction operations	No			Yes			
Supplementary operations 1. Logic operations 2. Bit test operations 3. Set/Reset operations 4. Timer and counter operations 5. Load and transfer operations 6. Conversion operations 7. Shift operations 8. Jump operations 9. Other operations	---[1]			Yes	---[1]		
System operations 1. Set operations 2. Load and transfer operations 3. Block call operations 4. Arithmetic operations 5. Other operations	---[1]			Yes	---[1]		

1 The operations listed are not permissible in OBs, PBs and SBs.

2.1.2 Operand Areas

The STEP 5 programming language has the following operand areas:

I	(Inputs)	Interfaces from the process to the PLC (via process I/O image)
Q	(Outputs)	Interfaces from the PLC to the process (via process I/O image)
F	(Flags)	Memory for intermediate results of binary operations
D	(Data)	Memory for intermediate results of digital operations
T	(Timers)	Memory for implementing timers
C	(Counters)	Memory for implementing counters
P	(Peripherals)	Interface from the process to the PLC (direct)
K	(Constants)	Defined numeric values
OB, PB, SB, FB, DB	(Blocks)	Program structuring aids

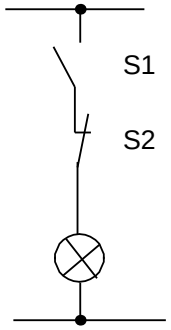
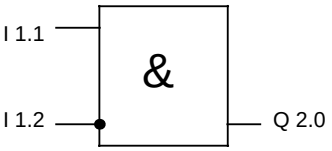
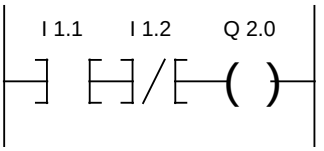
You will find a detailed description of all operations and programming examples in Chapter 3. Refer to Appendix A for a listing of all operations and operands.

2.1.3 Circuit Diagram Conversion

If your automation task is in the form of a circuit diagram, you must convert it to STL, CSF, or LAD.

Example: **Hard-wired control**
A signal lamp is to light up when a normally open contact (S1) is activated and a normally closed contact (S2) is not activated.

Programmable control
The signal lamp is connected to a PLC output (Q 2.0). The signal voltages of the two contacts are connected to two PLC inputs (I 1.1 and I 1.2).
The PLC scans to see if the signal voltages are present (signal state ”1” at the activated normally open contact or nonactivated normally closed contact). Both signal states are combined through logic AND. The result of the logic operation (RLO) is assigned to output 2.0 (the lamp lights up).

Circuit Diagram	STL	CSF	LAD
	A I 1.1 AN I 1.2 = Q 2.0		

2.2 Program Structure

An S5-115F program can be one either linear or structured.
Sections 2.2.1 and 2.2.2 describe these program types.

2.2.1 Linear Programming

Programming individual operations in one section (block) is sufficient for handling simple automation jobs.
For the S5-115F, this is organization block 1 (OB 1) (2.3.1). The S5-115F scans this block cyclically (i.e. after it scans the last statement, it goes back to the first statement and begins scanning again). A linear program can have 8 Kbytes.

Please note the following:
- When OB 1 is called, five words are assigned to the block header.
- Normally, a statement takes up one word in the program memory. Two-word statements also exist (e.g. with the operation ”Load a constant”). Count these statements twice when calculating the program length.
- Like all blocks, OB 1 must be terminated by a Block End statement (BE).

2.2.2 Structured Programming

To solve complex tasks, it is advisable to divide an entire program into individual, self-contained parts (blocks).

This procedure has the following advantages:

- Simple and clear programming, even for large programs
- Capability to standardize program parts
- Easy alteration
- Simple program test
- Simple start-up
- Subroutine techniques (block call from different locations)

The STEP 5 programming language has the following five block types:

- **Organization block (OB)**
  Organization blocks manage the control program.

- **Program block (PB)**
  Program blocks arrange the control program according to functional or technological aspects.

- **Sequence block (SB)**
  Sequence blocks are special blocks that program sequence controls. They are handled like program blocks.

- **Function block (FB)**
  Function blocks are special blocks for programming frequently recurring or especially complex program parts (e.g. reporting and arithmetic functions). You can assign parameters to them. They have an extended set of operations (e.g. jump operations within a block).

- **Data block (DB)**
  Data blocks store data needed to process a control program. Actual values, limiting values, and texts are examples of data.

The program uses block calls to exit one block and jump to another. You can therefore nest program, function, and sequence blocks randomly in up to 16 levels.

> **Note**
>
> When calculating the nesting depth, note that the system program itself can call an organization block under certain circumstances.

The total nesting depth is the sum of the nesting depths of all programmed organization blocks. If nesting goes beyond 16 levels, the PLC goes into the ”STOP” mode with the error message ”STUEB” (block stack overflow) (5.1.2).

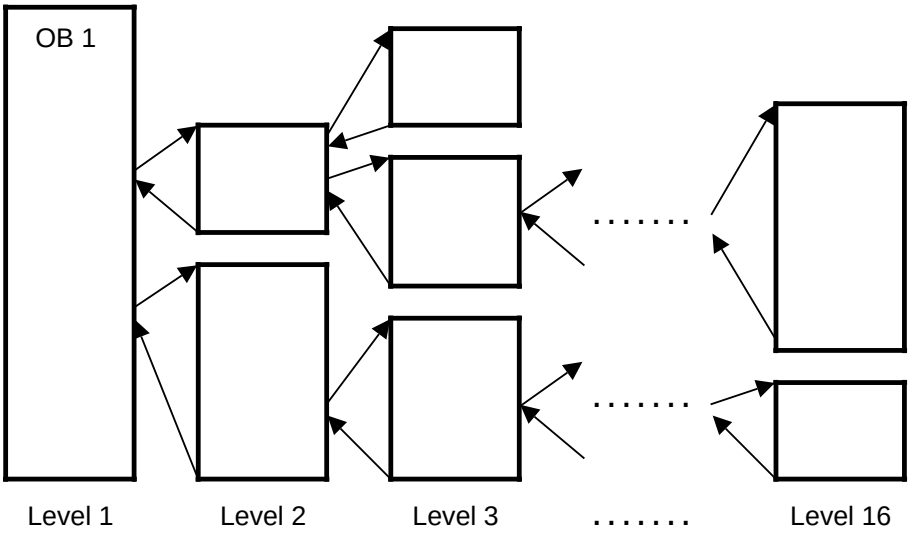


Figure 2-2. Nesting

2.3 Block Types

Table 2-2 lists the most important features of the block types:

Table 2-2. Comparison of Block Types

	OB	PB	SB	FB	DB
Quantity	255 [1] OB 1 to OB 255	256 [2] PB 0 to PB 255	256 [2] SB 0 to SB 255	256 [3] FB 0 to FB 255	252 [4] DB 4 to DB 255
Max. length	8 Kbytes	8 Kbytes	8 Kbytes	8 Kbytes	4096 data words [5]
Operations set (contents)	Basic operations	Basic operations	Basic operations	Basic supple-mentary system operations	Bit pattern numbers texts
Representa-tion methods	STL, CSF [6], LAD [6]	STL, CSF [6], LAD [6]	STL, CSF [6], LAD [6]	STL	
Block header length	5 words	5 words	5 words	5 words	5 words
Blocks calls	JU, JC possible only in FBs	JU, JC	JU, JC	JU, JC	C, G

1 The operating system can call special OBs by itself (2.3.1 and 6.2).

2 When using standard function blocks, do not use parameter ”0”.

3 Special function blocks are already integrated in the operating system (6.1).

4 Data blocks DB 0 and DB 1 are reserved for the block address list, the configuration DB and the error DBs.

5 A data block can be referenced up to DW 255 with ”L DW” or ”T DW”, with DB 2 and DB 3 reserved as error DBs.

6 The possible basic operation set is limited for the CSF and LAD methods of representation (Table 2-1).

Block structure

Each block consists of the following:
- Block header specifying the block type, number, and length.
  The programmer generates the block header when it transforms the block.
- Block body with the STEP 5 program or data.

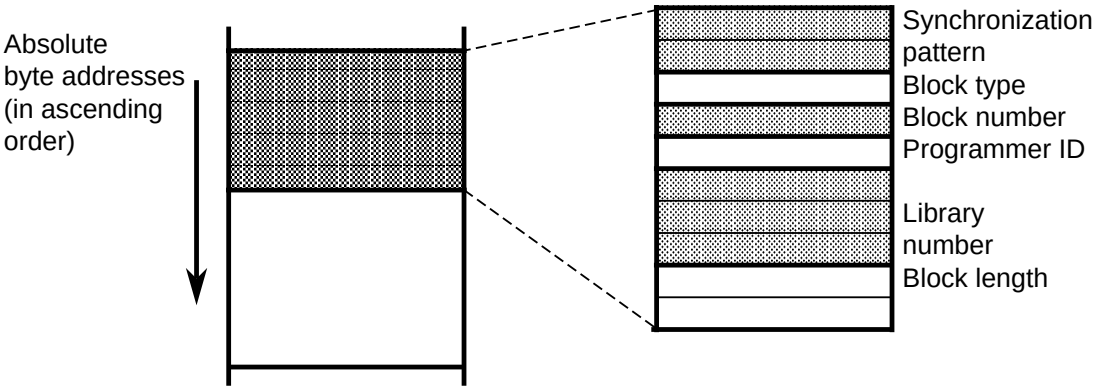


Figure 2-3. Structure of a Block Header

Programming

Program your blocks as follows (does not apply to data blocks):
1. Specify the block type (e.g. PB).
2. Specify the block number (e.g. 27).
3. Enter the control program statements.
4. Terminate the block with the ”BE” statement.

2.3.1 Organization Blocks (OBs)

Organization blocks form the interface between the operating system and the control program.

Organization blocks are handled in either of the following two ways:
- The operating system calls them cyclically, in response to events or at certain times.
- The control program can call them as operating functions (6.2).

Table 2-3 provides an overview of organization blocks.

All organization blocks can be programmed with parameters from the permissible range of 0 to 255. However, they must be called in the control program.

All OBs must start with the LPLZ (logical program counter) sequence of operations (3.8):
- LFW0
- LKF+1
- +F
- TFW0

The length of the OBs is calculated as follows:

	Operation code max.	127	words
+	Header	5	words
+	BE operation	1	word

If the block is longer than 127 words (operation code), a BLD 255 operation with a program counter sequence will result at the latest in the 128th codeword. Interposed BLD segmentation operations are unaffected by this.

The following OBs are provided by the CPU 942F:

Table 2-3. Overview of Organization Blocks

OB No.	Function
OB must be user-programmed and is called by the operating system.	
OB 1	Cyclic program scanning
Interrupt-driven program scanning	
OB 2	Process interrupt A: Interrupt generation by the 434 digital input module
OB 13	Time interrupt: (variable in each case: 100 msec. to 10 min. 55 sec.)
Handling of restart procedures	
OB 21	Manual power-up
OB 22	Automatic power-up when power is restored
OB is already programmed. OB must be called by the user.	
OB 251	PID control algorithm [1]

1 This block does not appear in the ”DIR PLC” display on the programmer.

Figure 2-4 shows how to set up a structured control program. It also illustrates the significance of organization blocks.

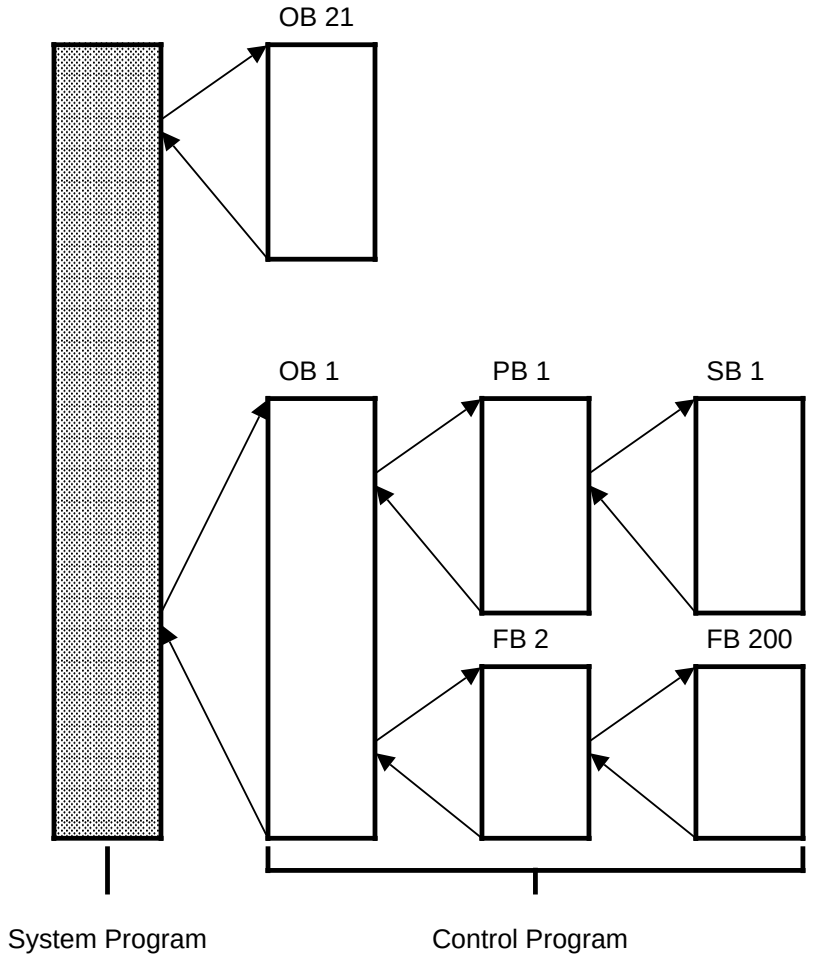


Figure 2-4. Example of Organization Block Use

The functions of the various organization blocks are described on the following pages.

OB 1: Cyclical program scanning

Control program structure is established in OB 1. Using a number of block calls in OB 1, you can specify the PB or FB processing sequence. Calls can be conditional or unconditional.

OB 2: CPU 942F interrupt handling

CPU 942F implements ”interrupt-driven” processing. Interrupt-driven processing occurs when a signal from the process causes the CPU in the PLC to interrupt cyclic scanning and to execute a specific program. After the CPU has executed the special program, it returns to the point of interruption in the cyclical program and continues scanning from there. The interrupt is triggered by digital input modules with process interrupt capability.

- **User interface:**
  When hardware interrupt A occurs, the operating system processes OB 2 after a maximum of 30 msec.
  If the interrupt OB is not programmed, cyclic program scanning continues.

- **Interrupt locations:**
  After a maximum of 20 msec., a program is called in the operating system to see if an interrupt OB must be processed. If an interrupt is pending in both subunits, the program branches into OB 2.
  If the 20 msec. interrupt processing intervals (30 msec. max. interrupt response time) are also to be observed in the user program, an FB 254 must be called in the user program after a maximum of 20 msec. This FB then synchronizes interrupt processing in both subunits by assigning the relevant parameters (6.1.6).

- **Disable interrupt:**
  The IA operation disables interrupt handling (OB 2, OB 13); RA enables it. The default setting is RA. An interrupt can be stored during interrupt disable.
  If interrupt handling is not required, the user can reduce PLC scan time.
  Disable interrupt handling with the IA operation and switch off interval monitoring when initializing the operating system with COM 115F.

- **Nesting depth:**
  Even with interrupt handling, the general block nesting depth of 16 levels must not be exceeded.

- **Saving data:**
  If the interrupt block uses scratch flags that are also used in the cyclic or time-driven control program, these flags must be saved in a data block while the interrupt is being serviced.

- **Synchronization of process interrupt handling**
  There is a considerable difference between implementation of interrupt handling in the S5-115F and in the S5-115U. This is due to the synchronization of the two interrupts in subunits A and B.

  Possible methods of interrupt handling:

  - Operating system: every 20 msec.
  - Control program: by calling an FB 254 SYNC (6.1.6)

  The operating system monitors the call interval in the control program. The user configures the desired maximum time in the operating system with COM 115F.
  Actual interrupt processing takes place in OB 2. The user uses the LPB operation exclusively to read in the interrupt register. In doing so, the contents of the interrupt register are exchanged between both subunits and compared. Interrupt bits set in both subunits are processed immediately, and interrupt bits set in only one subunit are stored until the next interrupt is serviced, but they must be set at the latest after expiry of the interrupt discrepancy time in the other subunit. Otherwise the S5-115F will stop and set an error flag.

OB 13: Time-driven program processing

The CPU 942F has OB 13 for time-driven program processing. The operating system processes the time-interrupt OB at intervals specified by the user. The time-interrupt OB can interrupt the cyclic control program but not the program servicing a process interrupt. If a time-interrupt OB is not programmed, cyclic program scanning continues.

- **Setting the call interval:**
  The call interval can be set as a multiple of 10 msec. (similar to setting scan monitoring). This means times from 100 msec. to approximately 10 min. can be programmed. The default is 0 msec.
  The call is suppressed by writing the value ”0” into the appropriate system data word when initializing the operating system.

- **Interrupt points:**
  After every 20 msec. (max.), a program is called in the operating system, which checks to see if a time-interrupt OB has to be processed. If a time interrupt request is pending in both sub-units, the program branches to OB 13.
  If the 20 msec. response time is also to be observed in the user program, an FB 254 must be called after a maximum of 20 msec. user program run time. This FB synchronizes processing of the OBs for servicing the time interrupt in both subunits (6.1.6).
  The OB 13 call may be delayed:
  Delay interval between FB 254 calls. The intervals between calls can vary:
  Difference ± interval between FB 254 calls.

  Example

Desired instant	Call	Interval between calls	Difference
100 msec.	120 msec.		
		80 msec.	
200 msec.	200 msec.		
100 msec.	100 msec.		40 msec.
		120 msec.	
200 msec.	220 msec.		

  When calculating the timing accuracy, note the following:
  - The maximum absolute error is independent of the length of the interval involved.
    The relative accuracy improves with increasing length of the interval involved.

  Example

Interval	Length of interval	Absolute difference	Relative error
100 to 200 msec.	100 msec.	40 msec.	40 %
100 to 500 msec.	400 msec.	40 msec.	10 %

- **Disabling the time-interrupt OB call:**
  The IA operation disables the call of all time-interrupt OBs. The RA operation enables it. One call request can be stored during call disable. If no time/process interrupt processing is required, it is useful to program the IA operation, since this speeds up program scanning.

- **Nesting depth:**
  Even when an OB for servicing a time interrupt is processed, the general block nesting depth of 16 must not be exceeded.

- **Initialization of time intervals for processing OB 13**
  The time interval is transferred to the S5-115F operating system via COM 115F.

- **Saving data**
  If a time interrupt service OB uses scratch flags that are also used in the cyclic control program, these flags must be saved in a data block during processing of the OB.

OB 21/22: Setting the restart characteristics

OB 21 is processed on a manual restart (programmer selection, mode selector), OB 22 on a cold restart after power failure (Figure 2.5). You program these blocks to obtain specific presettings.
If OB 21 or OB 22 is not programmed, the system branches directly into the ”RUN” mode (cyclic program scanning).

Features of the restart blocks (OB 21, OB 22) include the following:
- Timers are processed
- Scan time monitoring is not activated
- Interrupt blocks are not processed
- Digital output modules are disabled

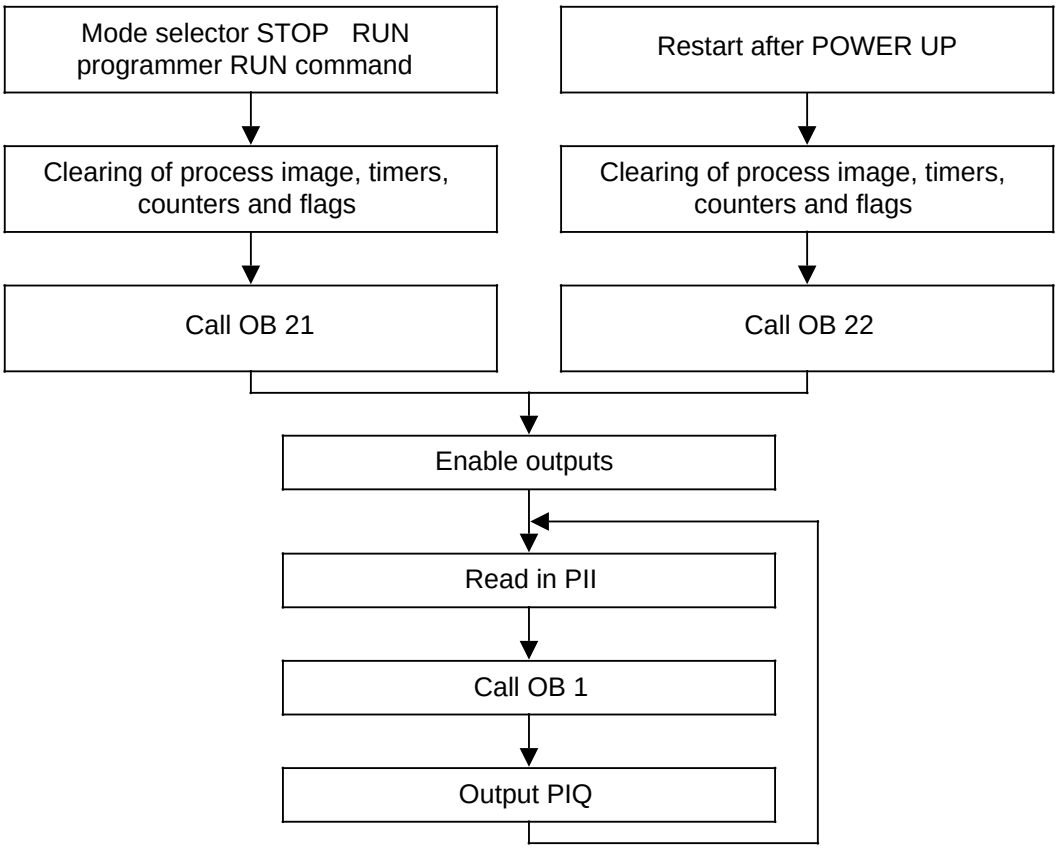


Figure 2-5. Setting the Restart Characteristics

OB 251 PID algorithm (6.2)

2.3.2 Program Blocks (PBs)

Self-contained program parts are usually programmed in blocks.

Call

Block calls JU and JC activate program blocks. You can program these operations in all block types except data blocks. Block call and block end cause the RLO to be reloaded. However, the RLO can be included in the ”new” block and evaluated.

The program counter (LPLZ) sequence contains the first operations for all PBs (3.8):
- LFW0
- LKF+1
- +F
- TFW0

The length of a PB is calculated as follows:

	Operation code max.	127	words
+	Header	5	words
+	BE operation	1	words

If the PB is longer than 127 words (operation code), a ”BLD 255 operation” with a program counter sequence appended will appear at the latest in the 128th codeword. Interposed BLD 255 segmentation operations are unaffected by this.

2.3.3 Sequence Blocks (SBs)

Sequence blocks are special program blocks that process sequence controls. They are treated like program blocks.

The program counter (LPLZ) sequence contains the first operations for all PBs (3.8):
- LFW0
- LKF+1
- +F
- TFW0

The length of a PB is calculated as follows:

	Operation code max.	127	words
+	Header	5	words
+	BE operation	1	words

If the FB is longer than 127 words (operation code), a ”BLD 255 operation” with a program counter sequence appended will appear at the latest in the 128th codeword. Interposed BLD 255 segmentation operations are unaffected by this.

2.3.4 Function Blocks (FBs)

Frequently recurring or complex control functions are programmed in function blocks.

Function blocks have the following special features:
- FBs can be assigned parameters.
  Actual parameters can be assigned when the block is called.
- FBs have a supplementary set of operations not available to other blocks.
- The FB program can be written and documented in STL only.

The S5-115F has the following types of function blocks:
- FBs that you can program
- FBs that are integrated in the operating system
- FBs that are available as software packages (Standard Function Blocks, Catalog ST 57)

The program counter (LPLZ) sequence is the first operation for all FBs (3.8):
- LFW0
- LKF+ 1
- + F
- TFW0

Memory requirement

In addition to the block header, function blocks have more organization information than other blocks.

Memory requirements are as follows:
- Block header (5 words)
- Block name (5 words) including jump operation (1W)
- Block parameters for initialization (3 words per parameter; 40 parameters max.)
- Block code (max. 127 W)
- BE operation (1 W)

Creating a function block

In contrast to other blocks, the following additional information can be stored in a function block:

- Library number
  The block can be assigned a number from 0 to 65535. This number is not dependent on symbolic or absolute parameters of the FB.
  A library number should be prescribed only once to identify a particular function block clearly. Standard function blocks have their own product number.

- Name
  A function block can be ”labelled” with a name of up to eight characters.

To assign parameters, you must program the following block parameter information:

- Block parameter name (formal operand)
  Each block parameter as formal operand is given a designation (DECL). Under this designation it is replaced by an actual operand when the function block is called.
  The name can be up to four characters long and must begin with a letter of the alphabet. You can program up to 40 parameters per function block.

- Block parameter type
  You can enter the following parameter types:
  - I Input parameters
  - Q Output parameters
  - D Data
  - B Blocks
  - T Timers
  - C Counters

In graphic representation, output parameters appear to the right of the function symbol. Other parameters appear to the left.

- Block data type
  You can specify the following data types:
  - BI Operands with bit address
  - BY Operands with byte address
  - W Operands with word address
  - K Constants

When assigning parameters, enter all block parameter specifications.

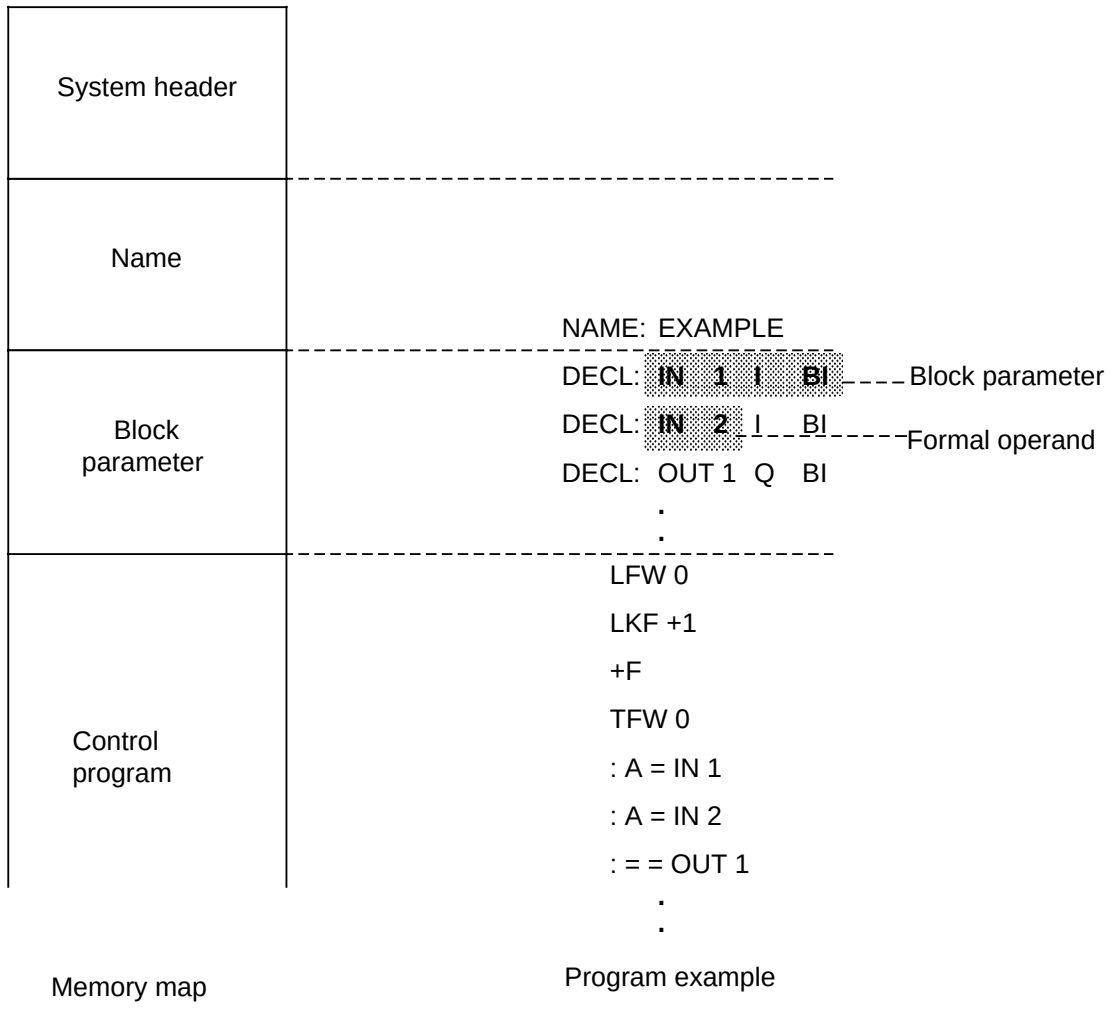


Figure 2-6. Programming a Function Block with Block Parameters

Table 2-4. Block Parameter Types and Data Types with Permissible Actual Operands

Parameter Type	Data Type	Permissible Actual Operands
I, Q	BI for an operand with bit address	I x.y inputs Q x.y outputs F x.y flags
	BY for an operand with byte address	IB x input bytes QB x output bytes FB x flag bytes DL x data bytes left DR x data bytes right PB x peripheral bytes
	W for an operand with word address	IW x input words QW x output words FW x flag words DW x data words PW x peripheral words
D	KM for a binary pattern (16 digits) KY for two absolute numbers, one byte each, each in the range from 0 to 255 KH for a hexadecimal pattern (maximum 4 digits) KS for a character (maximum 2 alphanumeric characters) KT for a time (BCD-coded time) with time base 1.0 to 999.3 KC for a count (BCD-coded) 0 to 999 KF for a fixed-point number in the range from -32768 to +32767	Constants
B	Type designation not permitted	DB x Data blocks. The C DBx operation is executed. FB x Function blocks (permissible without parameters only) are called unconditionally (JU..x). PB x Program blocks are called unconditionally (JU..x). SB x Sequence blocks are called unconditionally (JU..x).
T	Type designation not permitted	T Timer. The time should be assigned parameters as data or be programmed as a constant in the function block.
C	Type designation not permitted	Z Counter. The count should be assigned parameters as data or be programmed as a constant in the function block.

Call

Like other blocks, function blocks are stored under a specific number in the program memory (e.g. FB 47). The numbers 240 to 255 are reserved for integrated function blocks. You can program function block calls in all blocks except data blocks.

The call consists of the following parts:

- Call statement
  - JU FBx unconditional call
  - JC FBx call if RLO = 1
- Parameter list (only when parameters are assigned)

Function blocks can be called only if they have been programmed. When a function block call is being programmed, the PLC requests function block data automatically.

Assigning parameters

The program in the function block specifies how the formal operands are to be processed.
The block in which the FB is called must specify the operand with which the FB is to work (parameter list). This specification is written after the jump statement. The valid operands are also called actual operands.

- Parameter list:

After the call statement, the input and output variables and the data are defined, assigning each formal operand an actual operand. The length of the parameter list is based on the number of formal operands. You can therefore program up to 40 actual operands in the parameter list.
When the function block is processed, the actual operands from the parameter list are used in place of the formal operands. The PLC monitors the sequence of the variables.

Figure 2-7 illustrates function block programming.

- Other features:

The function block call takes up two words in the program memory. Each parameter takes up an additional memory word.
Catalog ST 57 gives the run time and the required memory length of standard function blocks.
The designators (DES) that appear on the programmer for the function block inputs and outputs during programming are stored in the function block itself, along with the name. You must therefore transfer all necessary function blocks to the program diskette (for off-line programming) or directly into the program memory of the programmable controller before programming on the programmer.

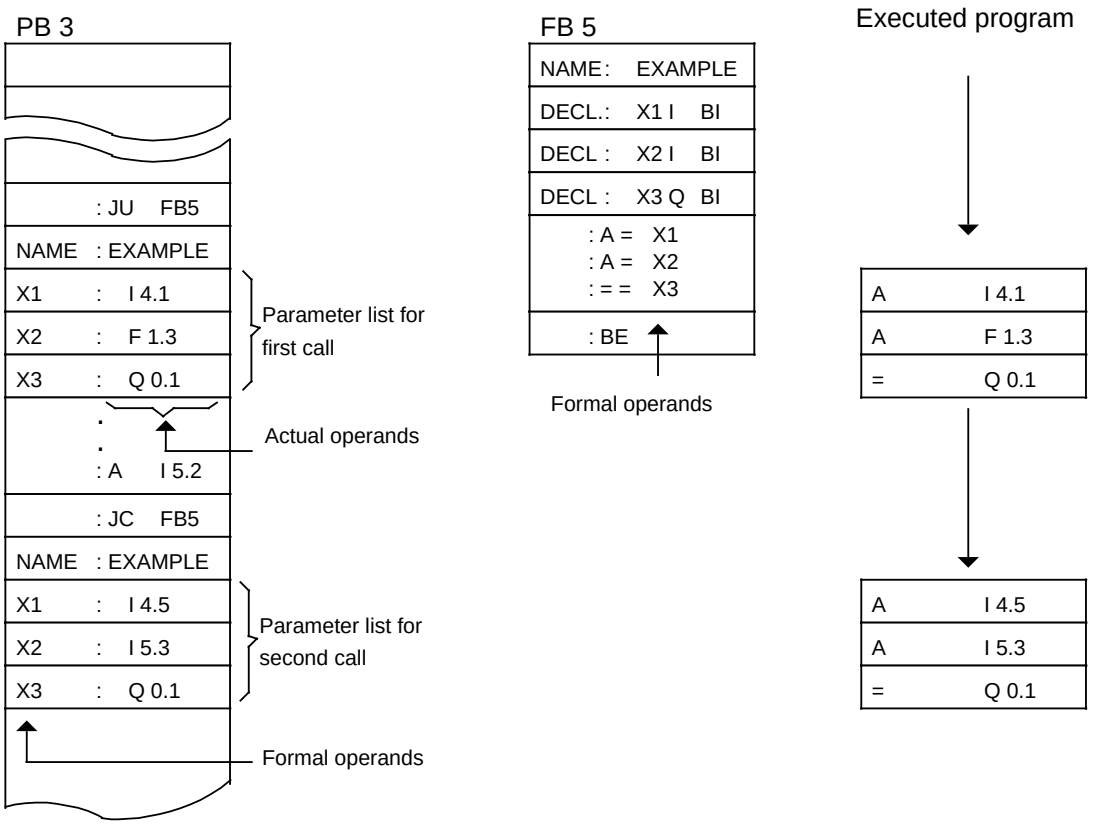


Figure 2-7. Programming a Function Block

2.3.5 Data Blocks (DBs)

Data blocks store data to be processed in a program.

The following data types are permissible:

- Bit pattern (representation of process states)
- Hexadecimal, binary, or decimal numbers (times, results of arithmetic operations)
- Alphanumeric characters (message texts)

Programming:

Begin data block programming by specifying a block number between 4 and 255. (DB 0 is reserved for the block address list. DB 1 is reserved for configuration, and DB 2 and DB 3 are reserved for error DBs.) Data is stored in words. If the information takes up less than 16 bits, the remaining bits are assigned zeros. Data input begins at data word 0 and continues in ascending order. A data block can hold up to 4096 data words. You can access a data block up to DW 255 with the ”L DW””and ”T DW” operations. Figure 2-8 shows an example of data block contents.

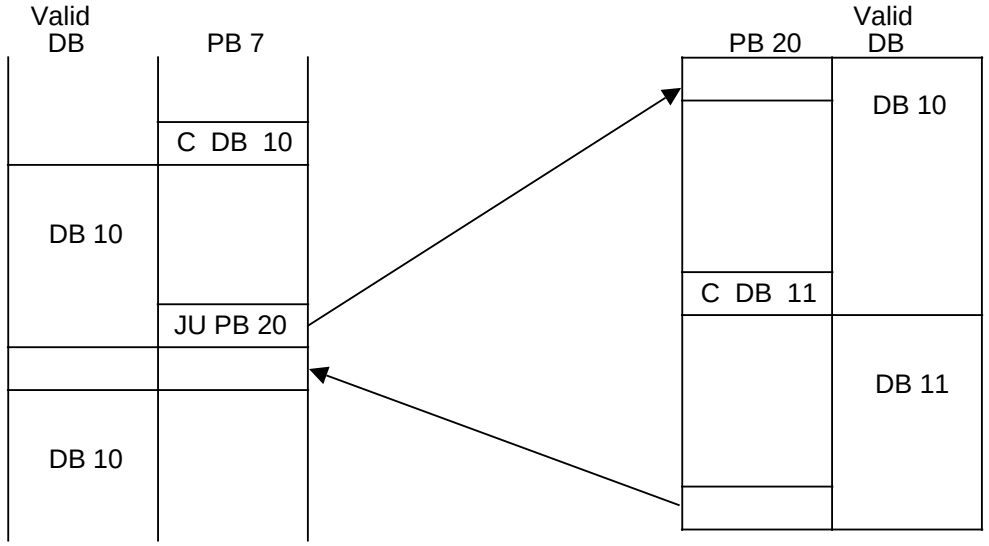
Input				Stored Values	
0000	:	KH =	A13C	DW0	A13C
0001	:	KT =	100.2	DW1	2100
0003	:	KF =	+21874	DW2	5572

Figure 2-8. Data Block Contents

Data blocks can also be created or erased in the control program (3.1.8).

Program processing with data blocks

- A data block must be called in the program with the C DB x operation (x = block No.).
- Within a block, a data block remains valid until another data block is called.
- When the program returns to the block that issued the call, the data block that was valid before the block call is again valid.



When PB 20 is called, the valid data area is entered in a memory.
When the program returns to the calling block, this area is reopened.

Figure 2-9. Validity Areas of Data Blocks

2.4 Modifying the Program

> **Note**
>
> The program can only be modified in Test mode!

You can modify the program while the PLC is in the STOP mode, regardless of block type, with the following programmer functions:
- ”INPUT”
- ”DISPLAY”
- ”STATUS” (4.1.1)

With the above functions, you can make the following changes:
- Insert, delete, or overwrite statements
- Insert or delete segments

2.5 Number Representation

With STEP 5 you can work with numbers in the following representations:
- decimal numbers from -32768 to +32767 (KF)
- hexadecimal numbers from 0000 to FFFF (KH)
- 16-bit fixed-point binary numbers from 00000000 00000000 to 11111111 11111111
- BCD-coded numbers (4 tetrads) from 0000 to 9999
- bit patterns (KM)

The S5-115F represents all numbers internally as 16-bit binary numbers or as bit patterns. Negative values are represented by their two's complement.

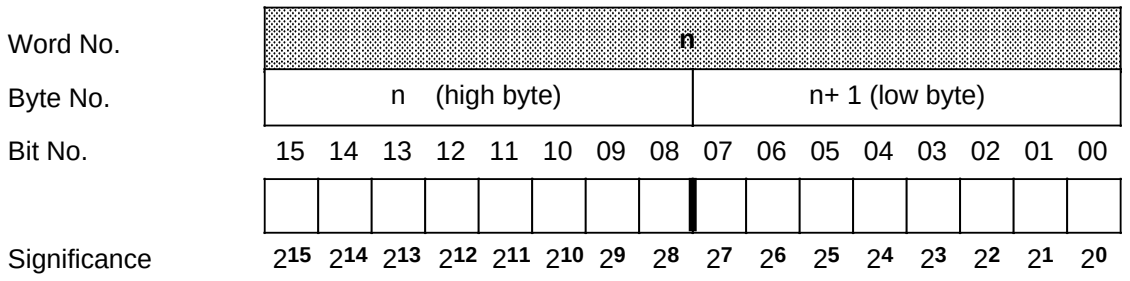


Figure 2-10. Bit Assignment of a 16-Bit Fixed-Point Binary Number

The following table contains two examples of number representation in the PLC:

Table 2-5. Examples of Number Representation in the PLC

Value entered	Representation in the PLC
KF - 50	1111 1111 1100 1110
KH A03F	1010 0000 0011 1111

3 STEP 5 Operations

3.1 Basic Operations . . . . . 3 - 1
3.1.1 Boolean Logic Operations . . . . . 3 - 1
3.1.2 Set/Reset Operations . . . . . 3 - 7
3.1.3 Load and Transfer Operations . . . . . 3 - 10
3.1.4 Timer Operations . . . . . 3 - 16
3.1.5 Counter Operations . . . . . 3 - 25
3.1.6 Comparison Operations . . . . . 3 - 29
3.1.7 Arithmetic Operations . . . . . 3 - 30
3.1.8 Block Call Operations . . . . . 3 - 32
3.1.9 Other Operations . . . . . 3 - 38
3.1.10 Segmentation with the BLD 255 Operation and the LPLZ Sequence in the Case of FBs . . . . . 3 - 39

3.2 Supplementary Operations . . . . . 3 - 42
3.2.1 Load Operation . . . . . 3 - 42
3.2.2 Enable Operation . . . . . 3 - 42
3.2.3 Bit Test Operations . . . . . 3 - 44
3.2.4 Digital Logic Operations . . . . . 3 - 46
3.2.5 Shift Operations . . . . . 3 - 49
3.2.6 Conversion Operations . . . . . 3 - 51
3.2.7 Decrement/Increment . . . . . 3 - 52
3.2.8 Disable/Enable Interrupt . . . . . 3 - 53
3.2.9 ”DO” Operation . . . . . 3 - 55
3.2.10 Jump Operations . . . . . 3 - 57
3.2.11 Substitution Operations . . . . . 3 - 59

3.3 System Operations . . . . . 3 - 65
3.3.1 Set Operations . . . . . 3 - 65
3.3.2 Load and Transfer Operations . . . . . 3 - 66
3.3.3 Arithmetic Operation . . . . . 3 - 68
3.3.4 Other Operations . . . . . 3 - 69

3.4 Condition Code Generation . . . . . 3 - 69

3.5 Sample Programs . . . . . 3 - 72
3.5.1 Transitional-Pulse Relay (Edge Evaluation) . . . . . 3 - 72
3.5.2 Binary Scaler . . . . . 3 - 73
3.5.3 Clock (Clock-Pulse Generator) . . . . . 3 - 74

3.6 Illegal Accesses . . . . . 3 - 75

3.7 Programming Notes . . . . . 3 - 76

3.8 LPLZ Incrementing Sequence . . . . . 3 - 76

Figures

3-1. Accumulator Structure . . . . . 3 - 10
3-2. Execution of the Load Operation . . . . . 3 - 12
3-3. Transferring a Byte . . . . . 3 - 13
3-4. Output of the Current Time . . . . . 3 - 18
3-5. Outputting the Current Counter Status . . . . . 3 - 26
3-6. Construction of a Single-Segment-FB for Parameters and Operations . . . . . 3 - 40
3-7. Structure of a Multi-Segment FB (max. 4096 words) . . . . . 3 - 40
3-8. Structure of a Single-Segment OB, PB, SB (max. 261 words) . . . . . 3 - 41
3-9. Structure of a Multi-Segment OB, PB, SB (max. 4096 words) . . . . . 3 - 41
3-10. Executing a ”DO” Operation . . . . . 3 - 56

Tables

3-1. Overview of Boolean Logic Operations . . . . . 3 - 2
3-2. Overview of Set/Reset Operations . . . . . 3 - 7
3-3. Overview of Load and Transfer Operations . . . . . 3 - 11
3-4. Overview of Timer Operations . . . . . 3 - 16
3-5. Overview of Counter Operations . . . . . 3 - 25
3-6. Overview of Comparison Operations . . . . . 3 - 29
3-7. Overview of Arithmetic Operations . . . . . 3 - 30
3-8. Overview of Block Call Operations . . . . . 3 - 32
3-9. Other Operations . . . . . 3 - 38
3-10. Load Operation . . . . . 3 - 42
3-11. Enable Operation . . . . . 3 - 43
3-12. Overview of Bit Test Operations . . . . . 3 - 44
3-13. Effect of ”TB” and ”TBN” on the RLO . . . . . 3 - 44
3-14. Overview of Digital Logic Operations . . . . . 3 - 46
3-15. Overview of Shift Operations . . . . . 3 - 49
3-16. Overview of Conversion Operations . . . . . 3 - 51
3-17. Decrement/Increment Operations . . . . . 3 - 52
3-18. Disable/Enable Interrupt Operations . . . . . 3 - 53
3-19. Overview of the ”DO” Operation . . . . . 3 - 55
3-20. Overview of Jump Operations . . . . . 3 - 57
3-21. Overview of Binary Logic Operations . . . . . 3 - 59
3-22. Overview of Set/Reset Operations . . . . . 3 - 60
3-23. Overview of Load and Transfer Operations . . . . . 3 - 61
3-24. Overview of Timer and Counter Operations . . . . . 3 - 62
3-25. DO Operation . . . . . 3 - 64
3-26. Overview of Set Operations . . . . . 3 - 65
3-27. Overview of Load and Transfer Operations . . . . . 3 - 66
3-28. Overview of the ”ADD” Operation . . . . . 3 - 68
3-29. The ”TAK” and ”STS” and ”DI” Operations . . . . . 3 - 69
3.30. Condition Code Settings for Comparison Operations . . . . . 3 - 70
3-31. Condition Code Settings for Fixed-Point Arithmetic Operations . . . . . 3 - 70
3-32. Condition Code Settings for Digital Logic Operations . . . . . 3 - 70
3-33. Condition Code Settings for Shift Operations . . . . . 3 - 71
3-34. Condition Code Settings for Conversion Operations . . . . . 3 - 71
3-35. Illegal Access Operations . . . . . 3 - 75

3 STEP 5 Operations

The STEP 5 programming language has the following three operation types:

- The basic operations include functions that can be executed in:
  - Organization blocks (OBs)
  - Program blocks (PBs)
  - Sequence blocks (SBs)
  - Function blocks (FBs)

  Apart from a few exceptions, basic operations can be represented in CSF and LAD.

  The exceptions are:
  - Arithmetic operations (3.1.7)
  - Block operations (3.1.8)
  - Other operations (3.1.9)
  - The BLD 255 operation for segmenting the user program (3.1.10)

- Supplementary operations can only be used in FBs and contain complex functions such as substitution statements, check functions, shift and conversion operations.
  Supplementary operations can only be input and output in STL.

- System operations can only be used in FBs and they access the operating system direct. They must only be used by an experienced programmer.
  System operations can only be input and output in STL.

3.1 Basic Operations

Sections 3.1.1 through 3.1.9 describe the basic operations with examples.

3.1.1 Boolean Logic Operations

These operations implement sequential Boolean logic. The new RLO (result of the logic operation) is generated from the previous RLO, internal information and the current operand status.

Table 3-1 provides an overview of Boolean logic operations. Examples follow the table.

Table 3-1. Overview of Boolean Logic Operations

Operation	Operand		Meaning
O			**Combine AND operations through logic OR.** Combine the result of the next AND logic operation (RLO) with the previous RLO through logic OR.
A(***			**Combine expression enclosed in parentheses through logic AND.** Combine the RLO of the expression enclosed in parentheses with the previous RLO through logic AND.
O(***			**Combine expression enclosed in parentheses through logic OR.** Combine the RLO of the expression enclosed in parentheses with the previous RLO through logic OR.
) ***			**Right parentheses.** Conclude the expression enclosed in parentheses.
A			**Scan operand for "1" and combine with RLO through logic AND.** The result is "1" when the operand in question carries signal state "1". Otherwise the scan results in "0". Combine this result with the RLO in the processor through logic AND. *
O			**Scan operand for "1" and combine with RLO through logic OR.** The result is "1" when the operand in question has signal state "1". Otherwise the scan results in "0". Combine this result with the RLO in the processor through logic OR. *
AN			**Scan operand for "0" and combine with RLO through logic AND.** The result is "1" when the operand in question has signal state "0". Otherwise the scan results in "0". Combine this result with the RLO in the processor through logic AND. *
ON			**Scan operand for "0" and combine with RLO through logic OR.** The result is "1" when the operand in question has signal state "0". Otherwise the scan results in "0". Combine this result with the RLO in the processor through logic OR. *

ID	Parameter	CPU 942F
I		0.0 to 127.7
Q		0.0 to 125.7
F		0.0 to 255.7 **
T		0 to 127
C		0 to 127

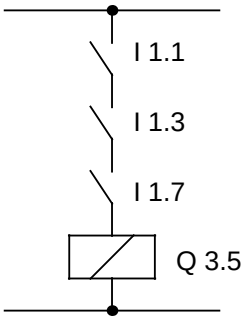
* If the scan immediately follows an operation resulting in reloading of the RLO (first scan), the result of the scan is reloaded as the new RLO.

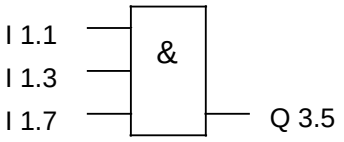
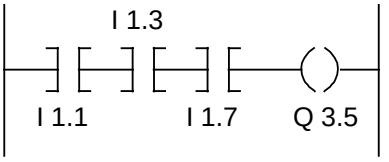
** FW 0 (F 0.0 to 1.7) reserved for logical program counter (3.8)
FW 2 to FW 198 (F 2.0 to F 199.7) permissible for the user program
FW 200 to FW 254 (F 200.0 to F 255.7) only permissible as scratch flags within a block

*** For reasons of safety, lengthy parenthesized structures are illegal, since the CPU hardware cannot monitor the parenthesis stack for overflow. Boolean logic operations should therefore be kept as compact as possible.

AND operation

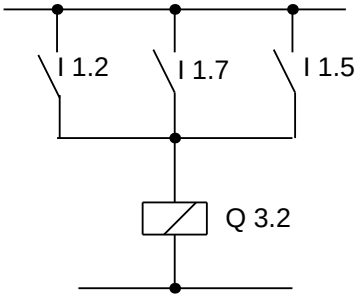
The AND operation scans to see if various conditions are satisfied simultaneously.

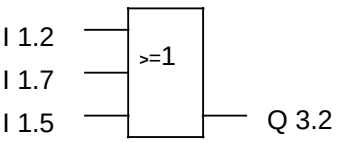
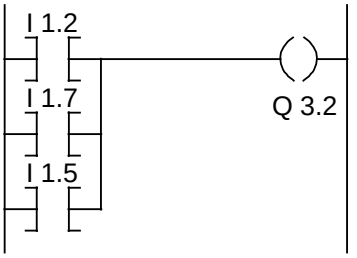
Example	Circuit Diagram
Output Q 3.5 is ”1” when all three inputs are ”1”. The output is ”0” if at least one input is ”0”. The number of scans and the sequence of the logic statements are optional.	

STL	CSF	LAD
A I 1.1 A I 1.3 A I 1.7 = Q 3.5		

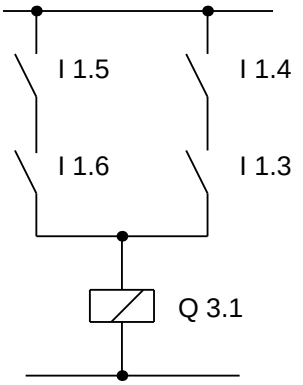
OR operation

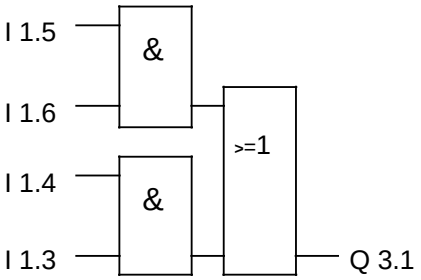
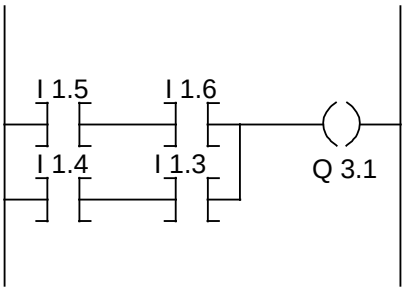
The OR operation scans to see if one of two (or more) conditions has been satisfied.

Example	Circuit Diagram
Output Q 3.2 is ”1” when at least one of the inputs is ”1”. Output Q 3.2 is ”0” when all inputs are ”0” simultaneously. The number of scans and the sequence of their programming are optional.	

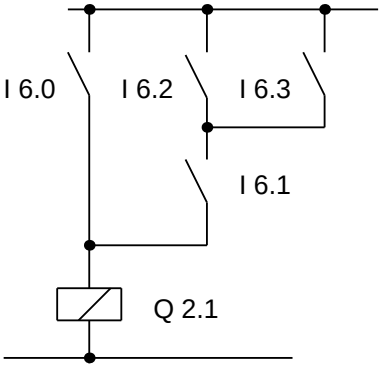
STL	CSF	LAD
O I 1.2 O I 1.7 O I 1.5 = Q 3.2		

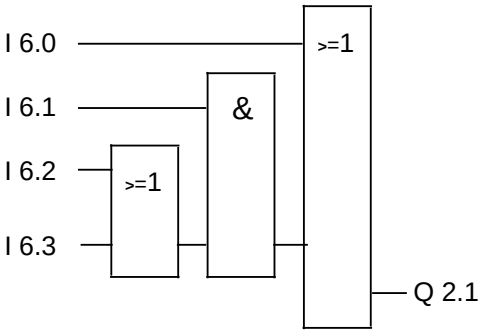
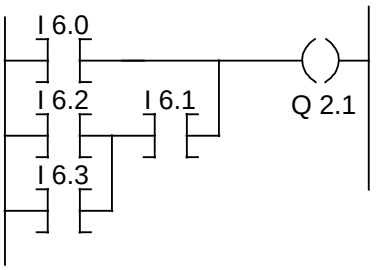
AND-before-OR operation

Example	Circuit Diagram
Output Q 3.1 is ”1” when at least one AND condition has been satisfied. Output Q 3.1 is ”0” when neither of the two AND conditions has been satisfied.	

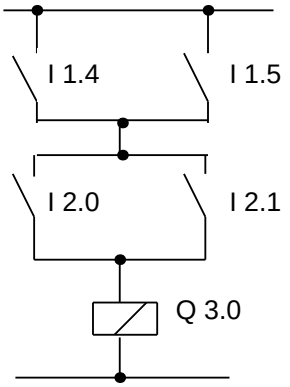
STL	CSF	LAD
A I 1.5 A I 1.6 O A I 1.4 A I 1.3 = Q 3.1		

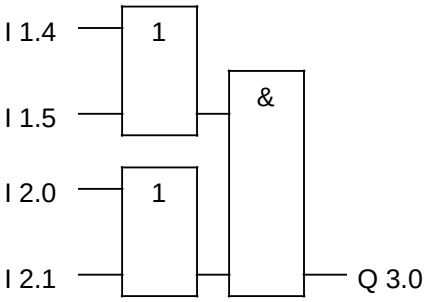
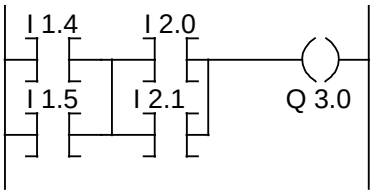
OR-before-AND operation

Example	Circuit Diagram
Output Q 2.1 is ”1” when one of the following conditions has been satisfied: • Input I 6.0 is ”1”. • Input I 6.1 and either input I 6.2 or I 6.3 are ”1”.	

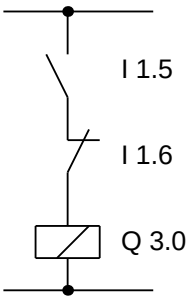
STL	CSF	LAD
O I 6.0 O A I 6.1 A(O I 6.2 O I 6.3) = Q 2.1		

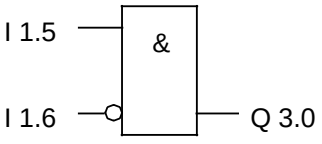
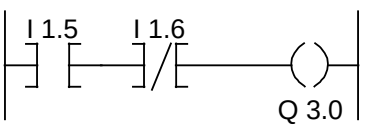
OR-before-AND operation

Example	Circuit Diagram
Output Q 3.0 is ”1” when both OR conditions have been satisfied. Output Q 3.0 is ”0” when at least one OR condition has not been satisfied.	

STL	CSF	LAD
A(O I 1.4 O I 1.5) A(O I 2.0 O I 2.1) = Q 3.0		

Scan for signal state ”0”

Example	Circuit Diagram
Output Q 3.0 is ”1” only when input I 1.5 is ”1” (normally open contact activated) and input I 1.6 is ”0” (normally closed contact not activated).	

STL	CSF	LAD
A I 1.5 AN I 1.6 = Q 3.0		

3.1.2 Set/Reset Operations

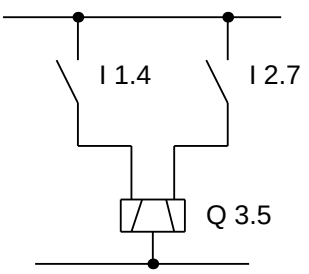
Set/reset operations store the result of the logic operation (RLO) generated in the processor. The stored RLO represents the signal state of the addressed operand. Storage can be dynamic (assignment) or static (set and reset). Table 3-2 provides an overview of the set/reset operations. Examples follow the table.

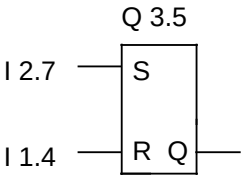
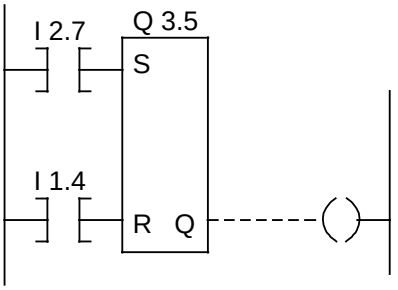
Table 3-2. Overview of Set/Reset Operations

Operation	Operand		Meaning
S			**Set** The first time the program is scanned with RLO = ”1”, signal state ”1” is assigned to the addressed operand. An RLO change does not affect this status.
R			**Reset** The first time the program is scanned with RLO = ”1”, signal state ”0” is assigned to the addressed operand. An RLO change does not affect this status.
=			**Assign** Every time the program is scanned, the current RLO is assigned to the addressed operand.
ID		**Parameter**	**CPU 942F**
	I		0.0 to 127.7
	Q		0.0 to 125.7
	F		0.0 to 255.7 *

***** FW 0 is reserved for the logical program counter.
FW 2 to FW 198 (F 2.0 to F 199.7) permissible for the user program.
FW 200 to FW 254 (F 200.0 to F 255.7) only permissible as scratch flags within a block. Scratch flags must be defined, set or reset at the start of a block run and also after conditional and unconditional block calls. In this way, data exchange with other blocks is avoided.

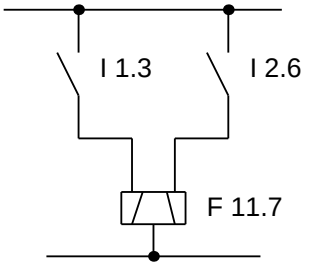
RS flip-flop for a latching signal output

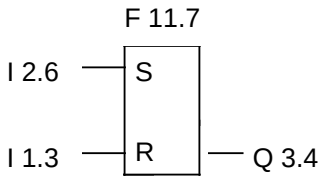
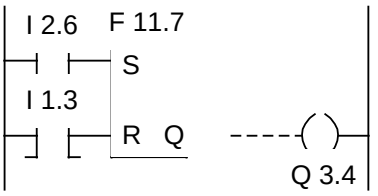
Example	Circuit
”1” at input I 2.7 sets flip-flop Q 3.5 (signal state ”1”). If the signal state at input I 2.7 changes to ”0”, the state of output Q 3.5 is maintained, i.e., the signal is latched. A ”1” at input I 1.4 resets the flip-flop (signal state ”0”). When the ”SET” signal (input I 2.7) and the ”RESET” signal (input I 1.4) are applied at the same time, the scanning operation that was programmed last (in this case A I 1.4) is in effect during processing of the rest of the program. In this example, resetting output Q 3.5 has priority.	

STL	CSF	LAD
A I 2.7 S Q 3.5 A I 1.4 R Q 3.5 **NOP 0** *		

* **NOP 0** is necessary if the program is to be represented in LAD or CSF form on the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. During programming in LAD and CSF, such ”NOP 0” operations are allotted automatically.

RS flip-flop with flags

Example	Circuit Diagram
A ”1” at input I 2.6 sets flip-flop F 11.7 (signal state ”1”). If the signal state at input I 2.6 changes to ”0”, the state of flag F 11.7 is maintained, i.e., the signal is latched. A ”1” at input I 1.3 resets the flip-flop (signal state ”0”). If the signal state at input I 1.3 changes to ”0”, flag F 11.7 maintains signal state ”0”. The signal state of the flag is scanned and transferred to output Q 3.4.	

STL	CSF	LAD
A I 2.6 S F 11.7 A I 1.3 R F 11.7 A F 11.7 = Q 3.4		

3.1.3 Load and Transfer Operations

Use load and transfer operations to do the following:
- Exchange information between various operand areas
- Prepare times and counts for further processing
- Load constants for program processing

Information flows indirectly via accumulators (ACCUM 1 and ACCUM 2). The accumulators are special registers in the CPU for temporary storage. In the S5-115F they are each 16 bits long. The accumulators are structured as shown in Figure 3-1.

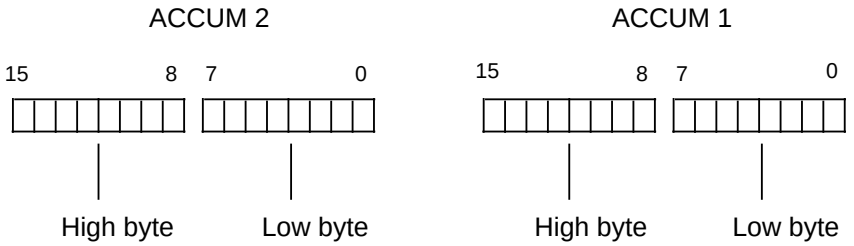


Figure 3-1. Accumulator Structure

You can load and transfer permissible operands in bytes or words.
For exchange in bytes, information is stored right-justified, i.e., in the low byte.
The remaining bits are set to zero.
You can process the information in the two accumulators using various operations.
Load and transfer operations are executed independently of condition codes. Execution of these operations does not affect the condition codes.
You can program load and transfer operations graphically only in combination with timer or counter operations; otherwise you can represent them only in STL form.

Table 3-3 provides an overview of the load and transfer operations.

Table 3-3. Overview of Load and Transfer Operations

Operation	Operand		Meaning
L			**Load** The operand contents are copied into ACCUM 1 regardless of the RLO. The RLO is not affected.
T			**Transfer** The contents of ACCUM 1 are assigned to an operand regardless of the RLO. The RLO is not affected.
ID		**Parameter**	**CPU 942F**
	IB		0 to 127
	IW		0 to 126
	QB		0 to 125
	QW		0 to 124
	FY[1]		0 to 255
	FW[1]		0 to 254
	DR		0 to 255
	DL		0 to 255
	DW		0 to 255
	T [2]		0 to 127
	C [3]		0 to 127
	PY [4]		0 to 127
	PW [4]		0 to 126
	KM [3]		random bit pattern (16 bit)
	KH [3]		0 to FFFF
	KF [3]		-32768 to +32767
	KY [3]		0 to 255 per byte
	KB [3]		0 to 255
	KC [3]		any 2 alphanumeric characters
	KT [3]		0.0 to 999.3
	KZ [3]		0 to 999
LC			**Load in BCD** Binary times and counts are loaded into ACCUM 1 in BCD code regardless of the RLO.
ID		**Parameter**	
	T		0 to 127
	C		0 to 127

1 FW 0 is reserved for the logical program counter.
FW 2 to FW 198 (F 2.0 to F 199.7) permissible for the user program.
FW 200 to FW 254 (F 200.0 to F 255.7) only permissible as scratch flags within a block. Scratch flags must be defined, set or reset at the start of a block run and also after conditional and unconditional block calls. In this way, data exchange with other blocks is avoided.

2 The LT operation loads a time which has been updated by the SYNC call last processed. It is not updated every 10 msec.

3 These operands cannot be used for transfer.

4 TPY and TPW can only be used in the digital range 0 to 125. FB 250 ANEI must be used in the analog range 128 to 255, and FB 251 ANAU must be used for analog output modules.

Load Operation:

During loading, information is copied from a memory area, e.g., from the PII, into ACCUM 1.
The previous contents of ACCUM 1 are shifted to ACCUM 2.
The original contents of ACCUM 2 are lost.

Example: Two consecutive bytes (IB 7 and IB 8) are loaded from the PII into the accumulator. Loading does not change the PII (Figure 3-2).

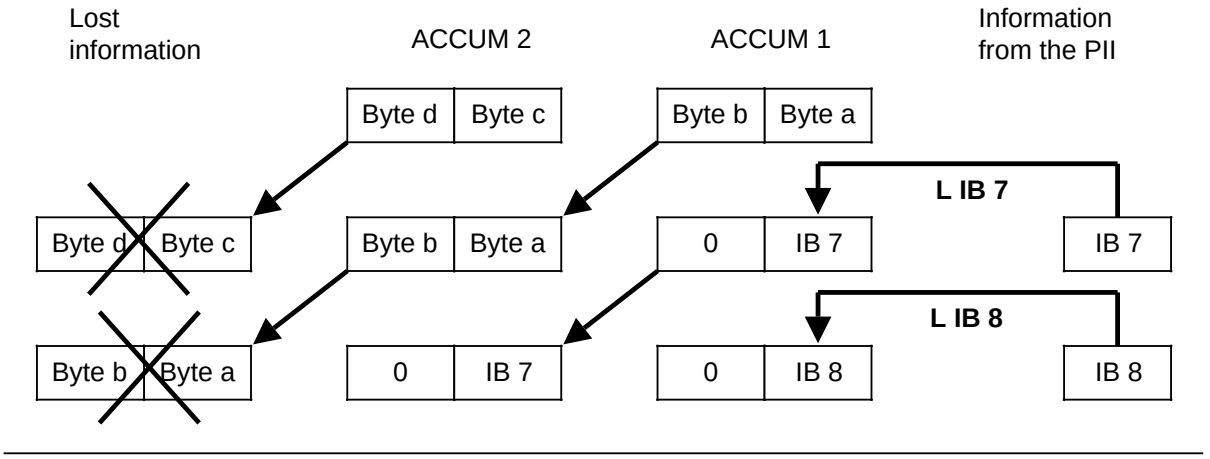


Figure 3-2. Execution of the Load Operation

Loading an input/output value with LPY/LPW operations:

Execution of an LPY/LPW operation triggers the following functions:

- Reading of the value.
- Exchange of data between the two subunits.
- Comparison of the two values of subunits A and B.
  If the values are not identical, the response depends on whether the relevant module is interrupt-generating or not.
  - Nonmatching values in the case of noninterrupt-generating DI modules.
    To achieve minimum execution time, removal of the discrepancy is not interrogated. The last valid value is taken as the standard value.
  - Nonmatching values in the case of interrupt DI modules
    Discrepancy analysis cannot be avoided here because 434 module interrupts are not stored beyond the access.
- Accesses in the AI address area
  Accesses in this area are only permissible on the CP 523 implemented in single-channel connection.

Transfer operation

During transfer, information from ACCUM 1 is copied into the addressed memory area, e.g. into the PIQ.
This transfer does not affect the contents of ACCUM 1.
When transfer to the digital output area occurs, the byte or word in question in the PIQ is automatically updated.

Example: Figure 3-3 shows how byte a (the low-order byte in ACCUM 1) is transferred to QB 5.

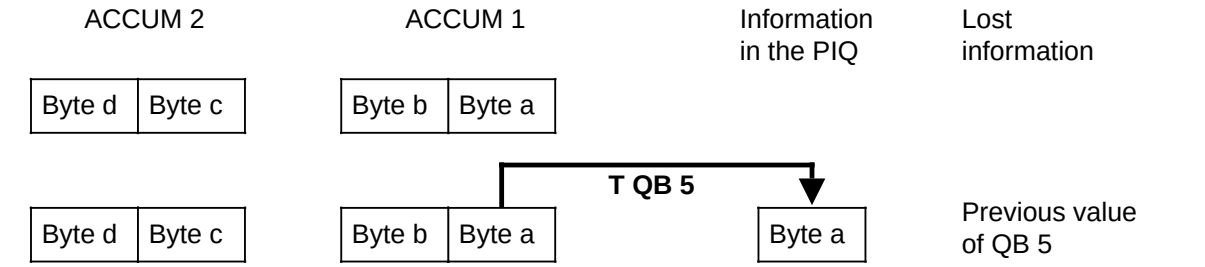
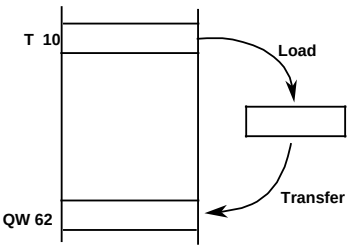


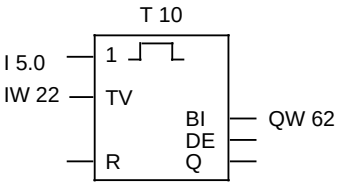
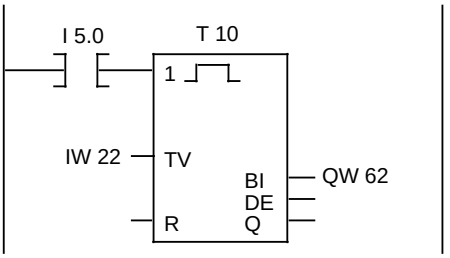
Figure 3-3. Transferring a Byte

Transferring an input/output value with TPY/TPW operations:

The TPB/TPW transfer operations are output in both subunits. The PIQ is updated at the same time.
In the case of single-channel I/O modules, these operations only affect the occupied subunit.

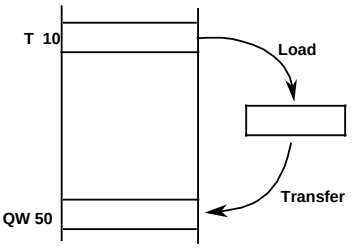
Loading and transferring a time (See also timer and counter operations)

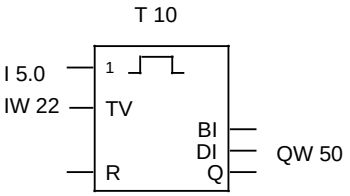
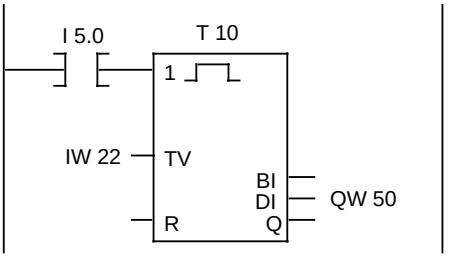
Example	Representation
During graphic input, QW 62 is assigned to output BI of a timer. The programmer automatically stores the corresponding load and transfer operation in the control program. Thus the contents of the memory location addressed with T 10 are loaded into ACCUM 1. Afterwards, the contents of the accumulator are transferred to the process image addressed with QW 62. In this example, you can see timer T 10 at QW 62 in binary code. Outputs BI and DE are digital outputs. The time at output BI is in binary code. The time at output DE is in BCD code with time base.	

STL	CSF	LAD
A I 5.0 L IW 22 SP T 10 **NOP 0*** L T 10 T QW 62: **NOP 0*** **NOP 0***		

* **NOP 0** is required if the program is to be represented in LAD or CSF on the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

Loading and transferring a time (coded)

Example	Representation
The contents of the memory location addressed with T 10 are loaded into the accumulator in BCD code. Then a transfer operation transfers the accumulator contents to the process image memory location addressed by QW 50. A coding operation is possible only indirectly for the graphic representation forms LAD and CSF by assigning an address to output DE of a timer or counter location. However, with STL, this operation can be entered with a separate statement.	

STL	CSF	LAD
A I 5.0 L IW 22 SP T 10 **NOP 0*** **NOP 0*** LC T 10 T QW 50 **NOP 0***		

* **NOP 0** is required if the program is to be represented in LAD or CSF in the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

3.1.4 Timer Operations

The program uses timer operations to implement and monitor chronological sequences. Table 3-4 provides an overview of timer operations. Examples follow the table.

Table 3-4. Overview of Timer Operations

Operation	Operand		Meaning
SP			**Pulse timer** The timer is started on the leading edge of the RLO. When the RLO is ”0”, the timer is set to ”0”. Scans result in signal state ”1”as long as the timer is running.
SE			**Extended pulse timer** The timer is started on the leading edge of the RLO. When the RLO is ”0”, the timer is not affected. Scans result in signal state ”1” as long as the timer is running.
SR			**On-delay timer** The timer is started on the leading edge of the RLO. When the RLO is ”0”, the timer is set to ”0”. Scans result in signal state ”1” when the timer has run out and the RLO is still pending at the input.
SS			**Retentive on-delay timer** The timer is started on the leading edge of the RLO. When the RLO is ”0”, the timer is not affected. Scans result in signal state ”1” when the timer has run out. The signal state becomes ”0” when the timer is reset with the ”R” operation.
SF			**Off-delay timer** The timer is started on the trailing edge of the RLO. When the RLO is ”1”, the timer is set to its initial value. Scans result in signal state ”1” as long as the RLO at the input is ”1” or the timer is still running.
R			**Reset timer** The timer is reset to its initial value as long as the RLO is ”1”. When the RLO is ”0”, the timer is not affected. Scans result in signal state ”0” as long as the timer is reset or has not been started yet.
ID T			**Parameter** 0 to 127

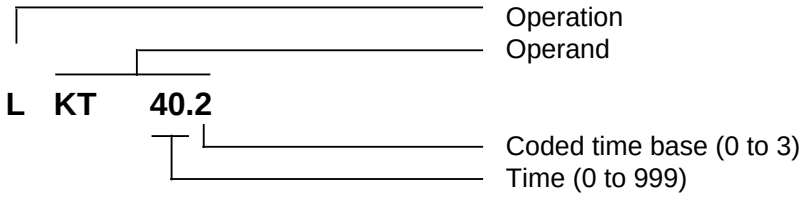
Loading a time

Timer operations call internal timers.
When a timer operation is started, the word in ACCUM 1 is used as a time value.
You must therefore first specify time values in the accumulator.

You can load a timer with any of the following data types:

KT	constant time value	
	or	
DW	data word	These data types must be in BCD code.
IW	input word	
QW	output word	
FW	flag word	

Loading a constant time value
The following example shows how you can load a time value of 40 sec.



Key for time base:

Base	0	1	2	3
Factor	0.01 sec.	0.1 sec.	1 sec.	10 sec.

Example: KT 40.2 corresponds to 40 x 1 sec.

Tolerances:

The time tolerance is calculated as the sum of the time base and the maximum interval between the FB 254 SYNC synchronization block calls.

Examples	Operand	Time Interval			
Possible settings for the time 40 sec.	KT 400.1	400	x 0.1 sec. ±	0.1 sec.	39.9 sec. ... 40.1 sec.
	KT 40.2	40	x 1 sec. ±	1 sec.	39 sec. ... 41 sec.
	KT 4.3	4	x 10 sec. ±	10 sec.	30 sec. ... 50 sec.

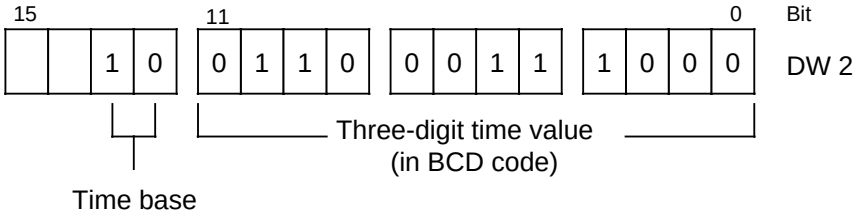
Note

Always use the smallest time base possible. You achieve more accurate times in this way.

Loading a time as input, output, flag, or data word

Load statement: L DW 2

The time 638 sec. is stored in data word DW 2 in BCD code.
Bits 14 and 15 are insignificant for the time value.



Key for time base:

Base BINARY	0 0	0 1	1 0	1 1
Factor	0.01 sec.	0.1 sec.	1 sec.	10 sec.

You can also use the control program to write to data word DW 2.

Example: Store the value 270 x 100 msec. in data word DW 2 of data block DB 3.

```
C   DB 3
L   KT 270.1
T   DW 2
```

Output of the current time [1]

You can use a load operation to put the current time into ACCUM 1 and process it further from there (Figure 3-4). The value is not updated every 10 msec. Last update at processing of FB 254 SYNC synchronization block.
Use the ”Load in BCD” operation for digital display output.

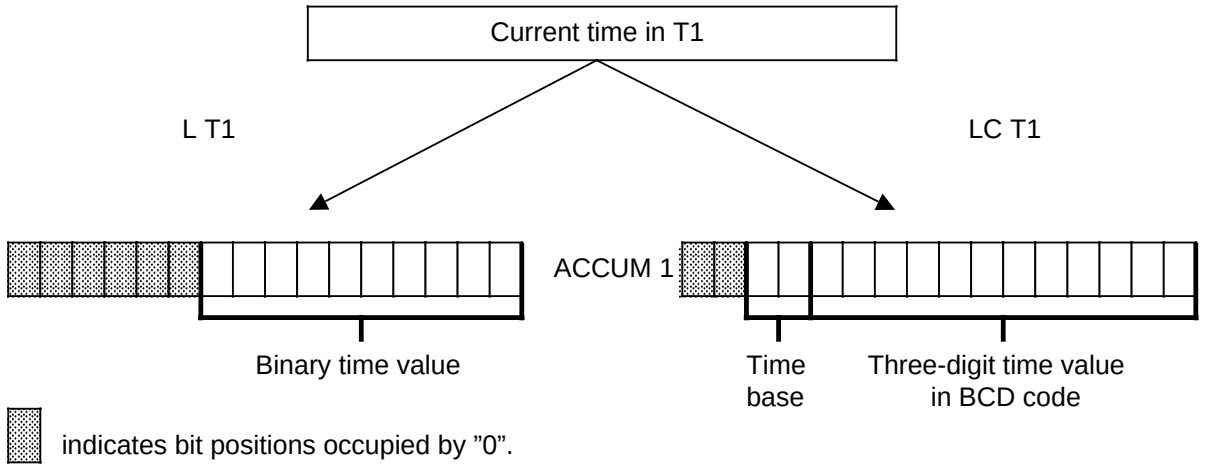


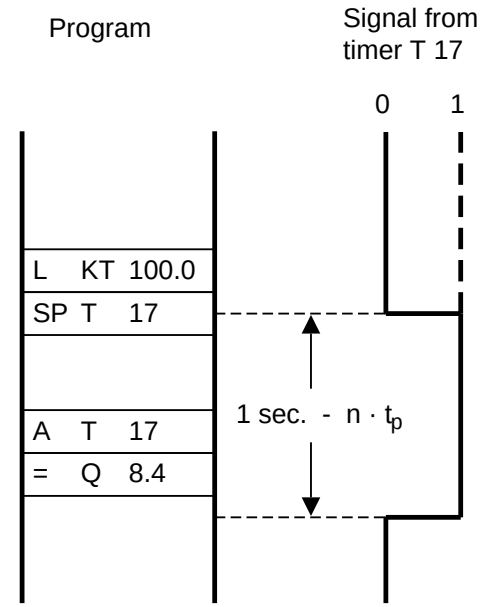
Figure 3-4. Output of the Current Time (Example)

1 The current time is the time value in the addressed timer.

Starting a timer

In the PLC, timers run asynchronously to program scanning. The time that has been set can run out during a program scanning cycle. It is evaluated by the next time scan. In the worst case, an entire program scanning cycle can go by before this evaluation. Consequently, timers should not activate themselves.

Example:

Schematic Representation	Explanation
 n: number of program scanning cycles t_p: program scan time	The schematic shows the "n^{th} + 1" processing cycle since timer T 17 * was started. Although the timer ran out shortly after the statement "= Q 8.4", output Q 8.4 remains set. The change is not considered until the next program scanning cycle. * KT 100.0 is equal to 1 sec.

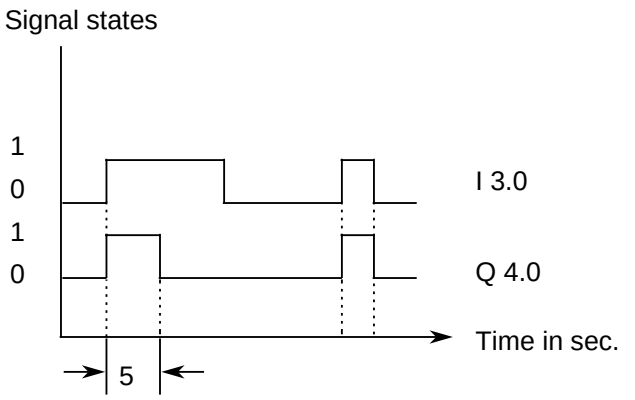
Except for "Reset timer", all timer operations are started only on an edge of the RLO. (The RLO alternates between "0" and "1") .
After being started, the loaded time is decremented in units corresponding to the time base until it reaches zero.
If there is an edge change while the timer is running, the timer is reset to its initial value and restarted.
The signal state of a timer can be interrogated with Boolean logic operations.

Pulse

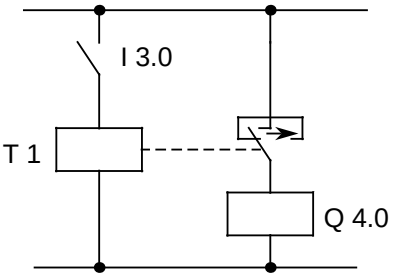
Example:

Output Q 4.0 is set when the signal state at input I 3.0 changes from ”0” to ”1”.
However, the output should not remain set longer than 5 sec.

Timing Diagram



Circuit Diagram

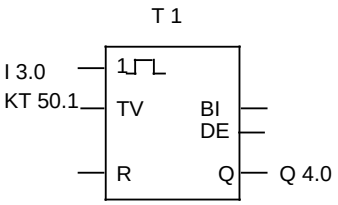


T 1: Time relay with transition-sensing NO contact

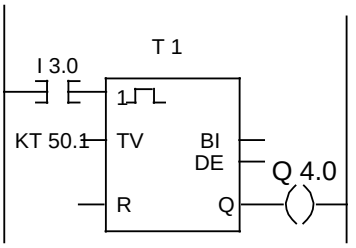
STL

```
A     I     3.0
L     KT    50.1
SP    T     1
NOP 0*
NOP 0*
NOP 0*
A     T     1
=     Q     4.0
```

CSF



LAD



***** **NOP 0** is required if the program is to be represented in LAD or CSF in the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

Note

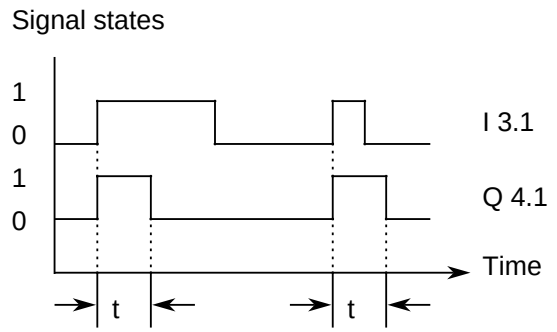
The time tolerance is calculated as the sum of the time base and the maximum interval between the SYNC synchronization block calls.

Extended pulse

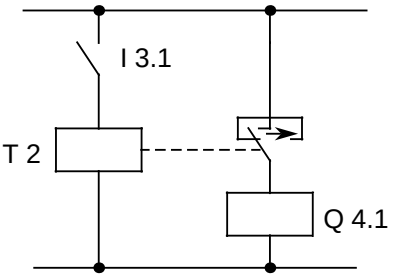
Example:

Output Q 4.1 is set for a specific time when the signal at input I 3.1 changes to ”1”. The time is indicated in IW 15.

Timing Diagram



Circuit Diagram

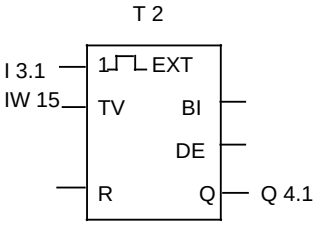


T 2: Time relay with pulse generation

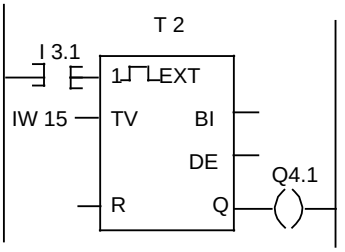
STL

```
A    I    3.1
L    IW   15
SE   T    2
NOP 0*
NOP 0*
NOP 0*
A    T    2
=    Q    4.1
```

CSF



LAD



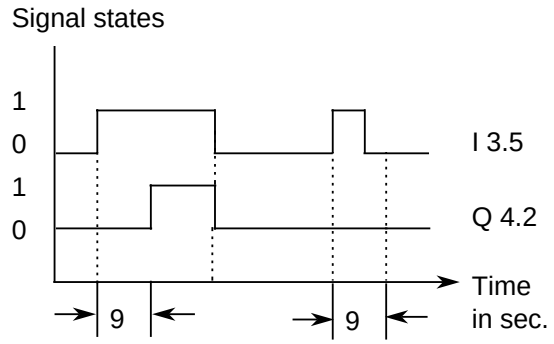
***** **NOP 0** is required if the program is to be represented in LAD or CSF in the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

On-delay

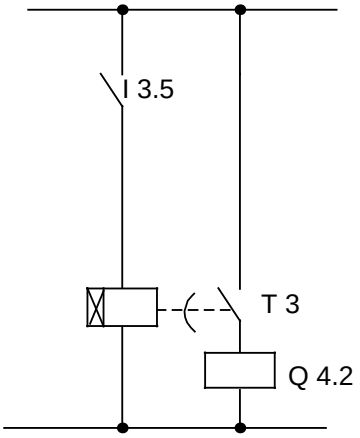
Example:

Output Q 4.2 is set 9 sec. after input I 3.5. It remains set as long as the input is ”1”.

Timing Diagram



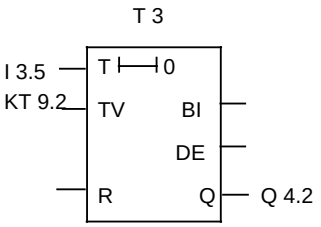
Circuit Diagram



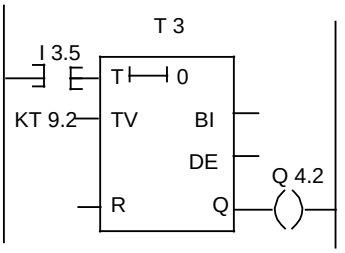
STL

```
A     I    3.5
L     KT   9.2
SR    T    3
NOP 0*
NOP 0*
NOP 0*
A     T    3
=     Q    4.2
```

CSF



LAD



***** **NOP 0** is required if the program is to be represented in LAD or CSF in the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

Note

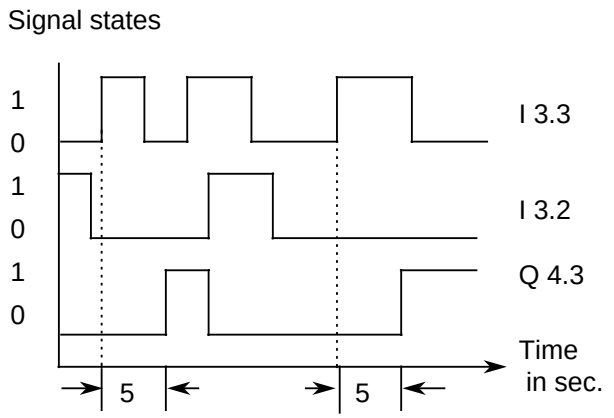
The time value ”9 sec.” will be slightly inaccurate if you load the timer with the statement ”L KT 900.0”.

Latching on-delay and reset

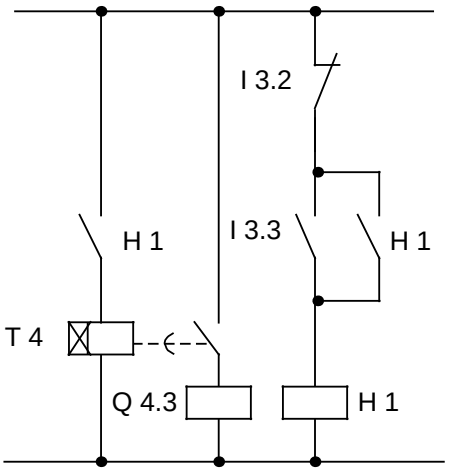
Example:

Output Q 4.3 is set 5 sec. after input I 3.3.
Further changes in the signal state at input I 3.3 do not affect the output.
Input I 3.2 resets timer T 4 to its initial value and sets output Q 4.3 to zero.

Timing Diagram



Circuit Diagram

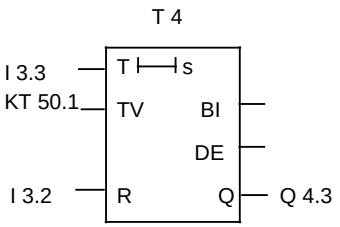


H 1: Auxiliary relay

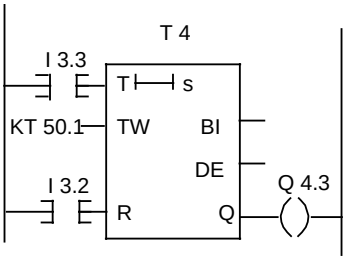
STL

```
A     I     3.3
L     KT    50.1
SS    T     4
A     I     3.2
R     T     4
NOP 0*
NOP 0*
A     T     4
=     Q     4.3
```

CSF



LAD



***** **NOP 0** is required if the program is to be represented in LAD or CSF in the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

Note

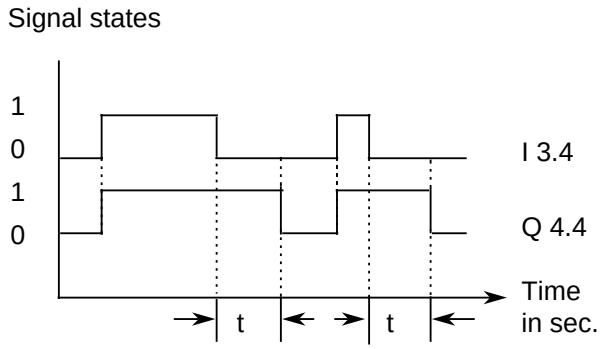
The time tolerance is calculated as the sum of the time base and the maximum interval between the FB 254 SYNC synchronization block calls.

Off-delay

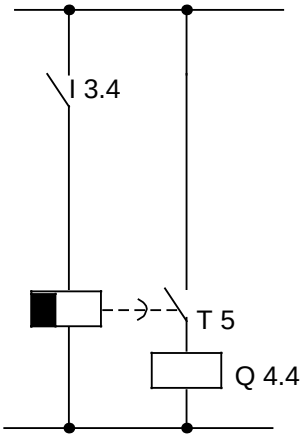
Example:

When input I 3.4 is reset, output Q 4.4 is set to zero after a certain delay (t). The value in FW 13 specifies the delay time.

Timing Diagram



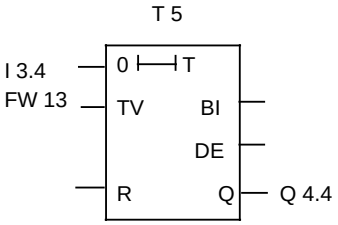
Circuit Diagram



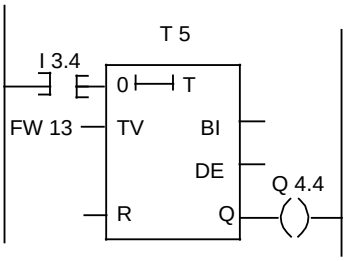
STL

```
A    I    3.4
L    FW   13
SF   T    5
NOP 0*
NOP 0*
NOP 0*
A    T    5
=    Q    4.4
```

CSF



LAD



***** **NOP 0** is required if the program is to be represented in LAD or CSF on the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

3.1.5 Counter Operations

The CPU uses counter operations to handle counting jobs directly. Counters can count up and down. The counting range is from 0 to 999 (three decades). Table 3-5 provides an overview of the counter operations. Examples follow the table.

Table 3-5. Overview of Counter Operations

Operation	Operand		Meaning
S			**Set counter** The counter is set on the leading edge of the RLO.
R			**Reset counter** The counter is set to zero as long as the RLO is ”1”.
CU			**Count up** The count is incremented by 1 on the leading edge of the RLO. When the RLO is ”0”, the count is not affected.
CD			**Count down** The count is decremented by 1 on the leading edge of the RLO. When the RLO is ”0”, the count is not affected.
ID C			**Parameter** 0 to 127

Loading a count

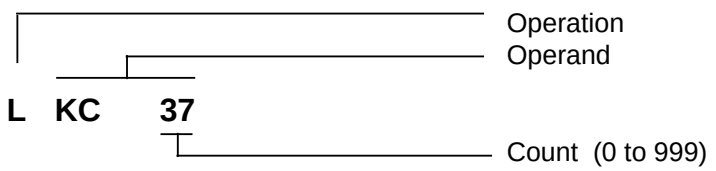
Counter operations call internal counters. When a counter is set, the word in ACCUM 1 is used as a count. You must therefore first store counts in the accumulator.

You can load a count with any of the following data types:

KC	constant count	
	or	
DW	data word	The data for these words must be in BCD code.
IW	input word	
QW	output word	
FW	flag word	

Loading a constant count:

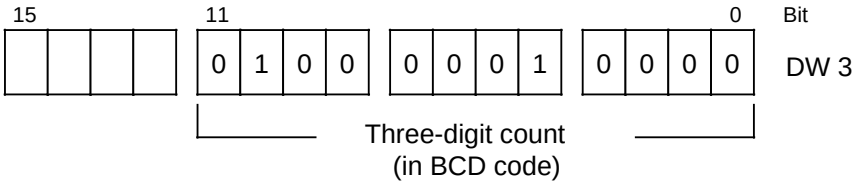
The following example shows how the count 37 is loaded.



Loading a count as input, output, flag, or data word

Load statement: **L DW 3**

The count 410 is stored in data word DW 3 in BCD code.
Bits 12 to 15 are insignificant for the count.



Scanning the counter

Use Boolean logic operations to scan the counter status (e.g., A Cx). As long as the count is not zero, the scan result is signal state ”1”.

Outputting the current counter status

You can use a load operation to put the current counter status into ACCUM 1 and process it further from there. The ”Load in BCD” operation outputs a digital display (Figure 3-5).

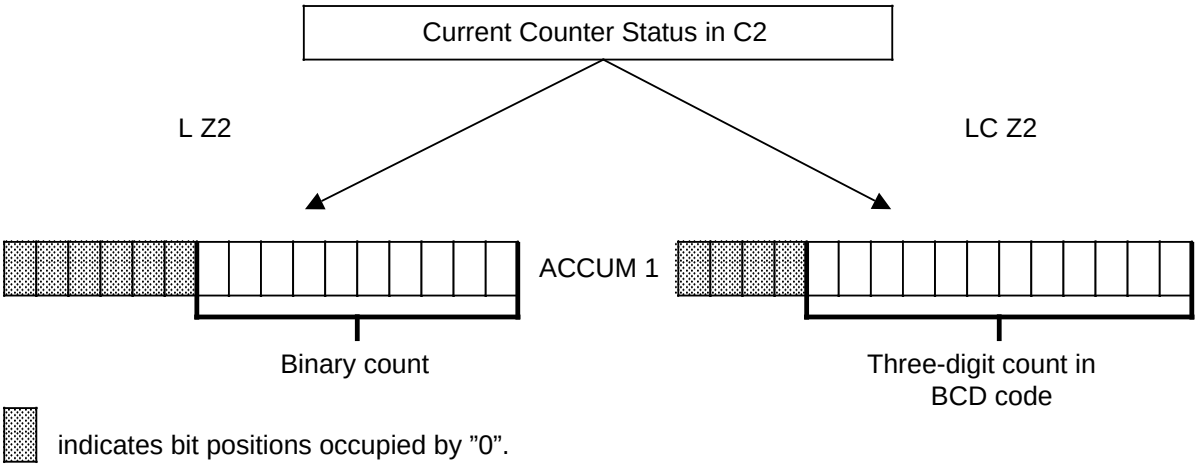


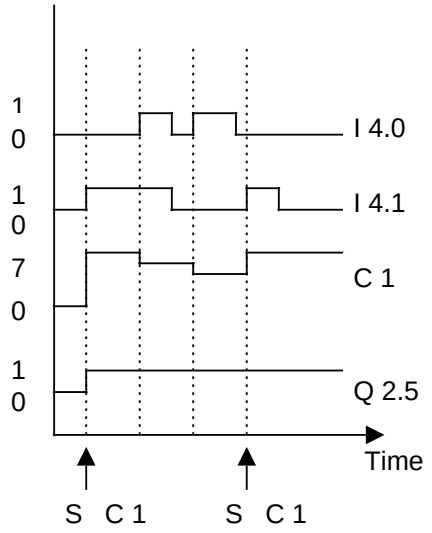
Figure 3-5. Outputting the Current Counter Status

Setting a counter ”S” and counting down ”CD”

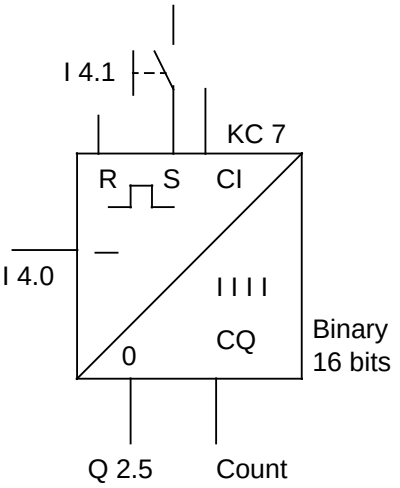
Example:

When input I 4.1 is switched on (set), counter 1 is set to the count 7. Output Q 2.5 is now ”1”.
Every time input I 4.0 is switched on (count down), the count is decremented by 1.
The output is set to ”0” when the count is ”0”.

Timing Diagram



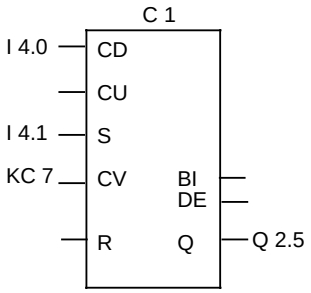
Circuit Diagram



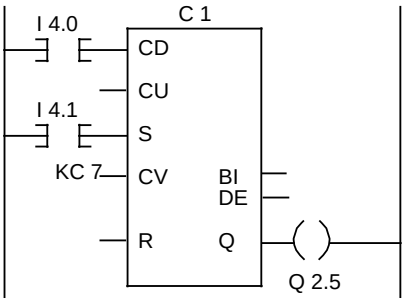
STL

```
A    I    4.0
CD   C    1
NOP  0*
A    I    4.1
L    KC   7
S    C    1
NOP  0*
NOP  0*
NOP  0*
A    C    1
=    Q    2.5
```

CSF



LAD



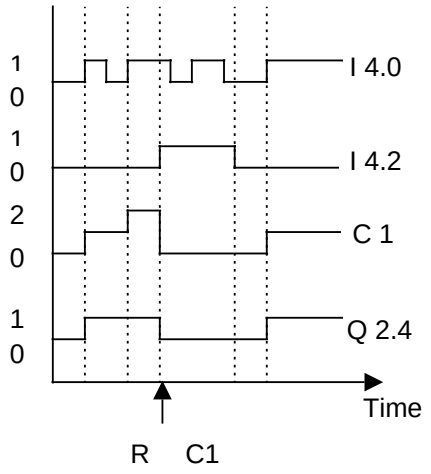
***** **NOP 0** is required if the program is to be represented in LAD or CSF in the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

Resetting a counter ”R” and counting up ”CU”

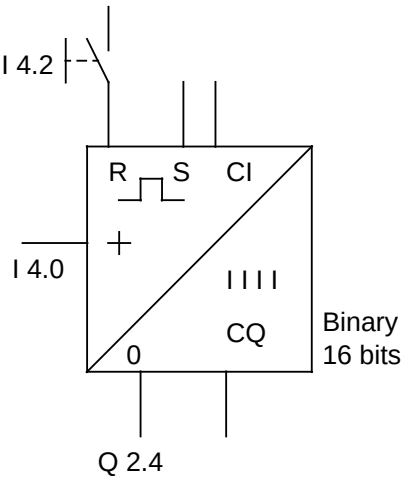
Example:

When input I 4.0 is switched on, the count in counter 1 is incremented by 1. As long as a second input (I 4.2) is ”1”, the count is reset to ”0”.
The A C1 operation results in signal state ”1” at output Q 2.4 as long as the count is not ”0”.

Timing Diagram



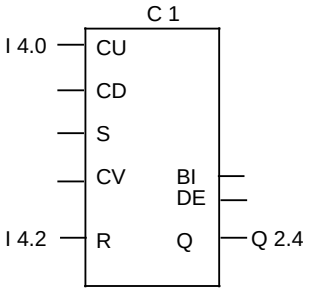
Circuit Diagram



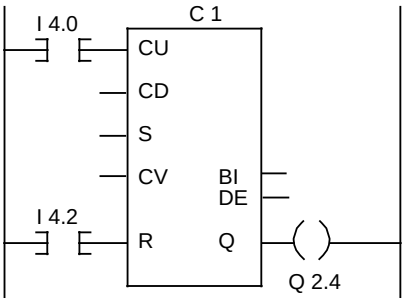
STL

```
A    I    4.0
CU   C    1
NOP  0*
NOP  0*
NOP  0*
A    I    4.2
R    C    1
NOP  0*
NOP  0*
A    C    1
=    Q    2.4
```

CSF



LAD



***** **NOP 0** is required if the program is to be represented in LAD or CSF on the PG 635, PG 670, PG 675, PG 685, PG 695 or PG 750 programmers. NOP 0 operations are automatically assigned when programming in LAD and CSF.

3.1.6 Comparison Operations

Comparison operations compare the contents of the two accumulators. The comparison does not change the accumulators' contents. Table 3-6 provides an overview of the comparison operations. An example follows the table.

Table 3-6. Overview of Comparison Operations

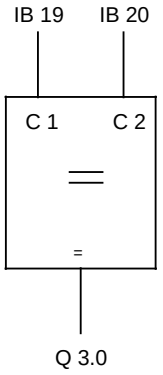
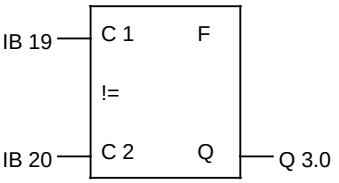
Operation	Operand		Meaning
! = F			**Compare for ”equal to”.** The contents of the two accumulators are interpreted as bit patterns and scanned to see if they are equal.
> < F			**Compare for ”not equal to”.** The contents of the two accumulators are interpreted as bit patterns and compared to see if they are not equal.
> F			**Compare for ”greater than”.** The contents of the two accumulators are interpreted as fixed-point numbers. They are compared to see if the operand in ACCUM 2 is greater than the operand in ACCUM 1.
> = F			**Compare for ”greater than or equal to”.** The contents of the two accumulators are interpreted as fixed-point numbers. They are compared to see if the operand in ACCUM 2 is greater than or equal to the operand in ACCUM 1.
< F			**Compare for ”less than”.** The contents of the two accumulators are interpreted as fixed-point numbers. They are compared to see if the operand in ACCUM 2 is less than the operand in ACCUM 1.
< = F			**Compare for ”less than or equal to”.** The contents of the two accumulators are interpreted as fixed-point numbers. They are compared to see if the operand in ACCUM 2 is less than or equal to the operand in ACCUM 1.

Processing comparison operations

To compare two operands, load them consecutively into the two accumulators.
Execution of the operations is independent of the RLO. The result is binary and is available as RLO for further program scanning. If the comparison is satisfied, the RLO is ”1”. Otherwise it is ”0”.
Executing the comparison operations sets the condition codes (3.4).

> **Note**
>
> When using comparison operations, make sure the operands have the same number format.

Example: The values of input bytes IB 19 and IB 20 are compared. If they are equal, output Q 3.0 is set.

Circuit Diagram	STL	CSF/LAD
	L IB 19 L IB 20 !=F = Q 3.0	

3.1.7 Arithmetic Operations

Arithmetic operations interpret the contents of the accumulators as fixed-point numbers and manipulate them. The result is stored in ACCUM 1. Table 3-7 provides an overview of the arithmetic operations. An example follows the table.

Table 3-7. Overview of Arithmetic Operations

Operation	Operand		Meaning
+ F			**Addition** The contents of both accumulators are added.
- F			**Subtraction** The contents of ACCUM 1 are subtracted from the contents of ACCUM 2.
Method of representation: only STL possible			

The S5-115F has integrated function blocks for multiplication and division.

Processing an arithmetic operation

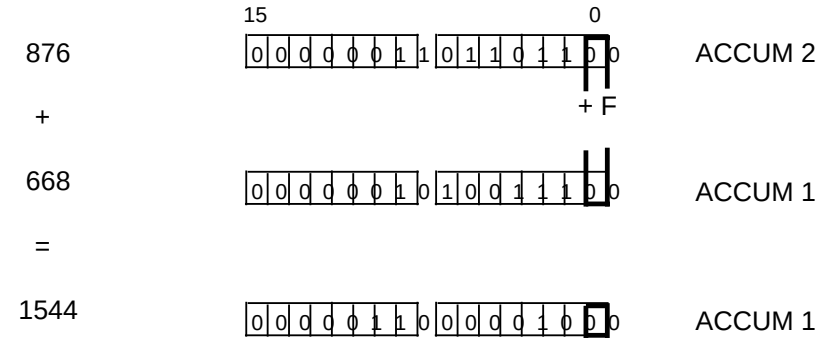
Before an arithmetic operation is executed, both operands must be loaded into the accumulators.

> **Note**
>
> When using arithmetic operations, make sure the operands have the same number format.

Arithmetic operations are executed independently of the RLO. The result is available in ACCUM 1 for further processing. The contents of ACCUM 2 are not changed. These operations do not affect the RLO. The condition codes are set according to the results.

STL		Explanation
L	C 3	The value of counter 3 is loaded into ACCUM 1.
L	C 1	The value of counter 1 is loaded into ACCUM 1. The previous contents of ACCUM 1 are shifted to ACCUM 2.
+ F		The contents of the two accumulators are interpreted as 16-bit fixed-point numbers and added.
T	QW 12	The result, contents of ACCUM 1, is transferred to output word QW 12.

Numeric Example



3.1.8 Block Call Operations

Block call operations specify the sequence of a structured program. Table 3-8 provides an overview of the block call operations. Examples follow the table.

Table 3-8. Overview of Block Call Operations

Operation	Operand		Meaning
JU			**Jump unconditionally** Program scanning continues in a different block regardless of the RLO. The RLO is not affected.
JC			**Jump conditionally** Program scanning jumps to a different block when the RLO is ”1”. Otherwise program scanning continues in the previous block. The RLO is set to ”1”.
ID		**Parameter**	
	OB	0 to 255 *	
	PB	0 to 255	
	FB	0 to 255	
	SB	0 to 255	
C			**Call a data block.** A data block is activated regardless of the RLO. Program scanning is not interrupted. The RLO is not affected.
G			**Generate and delete a data block.** An area is set up in the RAM to store data regardless of the RLO.
ID		**Parameter**	
	DB	1 to 255 **	
BE			**Block end** The current block is terminated regardless of the RLO. Program scanning continues in the block in which the call originated. The RLO is ”carried along” but not affected.
BEU			**Block end, unconditional** The current block is terminated regardless of the RLO. Program scanning continues in the block in which the call originated. The RLO is ”carried along” but not affected.
BEC			**Block end, conditional** When the RLO is ”1”, the current block is terminated. Program scanning continues in the block in which the call originated. During the block change, the RLO remains ”1”. If the RLO is ”0”, the operation is not executed. The RLO is set to ”1” and linear program scanning continues.
Method of representation: only STL possible			

***** Please note that certain OBs are assigned by the operating system.

****** Data blocks DB 0 to DB 3 are reserved and may only be read (G DB operation illegal):
DB 0 Block address list
DB 1 Configuration DB (1.1.7)
DB 2 Error DB, subunit A (5.4.2)
DB 3 Error DB, subunit B (5.4.2)

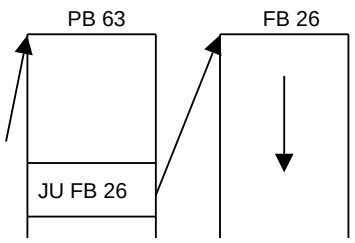
Unconditional block call ”JU”

One block is called within another block, regardless of conditions.

In the case of jump operations to missing blocks, the operating system of the CPU 942-7UF12 responds differently depending on the operating mode.

- If a jump is called to a missing block in safety mode, the S5-115F stops.
- If a jump is called to a missing block in test mode, the jump is not executed and the program processes the next operation. The S5-115F remains in RUN.
  For this reason, construction of the complete user program step by step by successively loading blocks is possible without continually rewriting OB 1. It is therefore possible to program OB 1 with all planned block calls.

Example: A special function has been programmed in FB 26. It is called at several locations in the program, e.g. in PB 63, and processed.

Program Sequence	STL	Explanation
	. . . . . JU FB 26 .	The ”JU FB 26” statement in program block PB 63 calls function block FB 26.

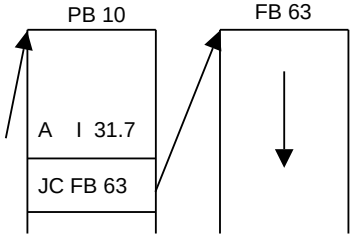
Conditional Block Call ”JC”

One block is called within another block when the previous condition has been satisfied (RLO=”1”).

In the case of jump operations to missing blocks, the operating system of the CPU 942-7UF12 responds differently depending on the operating mode.

- If a jump is called to a missing block in safety mode, the S5-115F stops.
- If a jump is called to a missing block in test mode, the jump is not executed and the program processes the next operation. The S5-115F remains in RUN.
  For this reason, construction of the complete user program step by step by successively loading of blocks is possible without continually rewriting OB 1. It is therefore possible to program OB 1 with all planned block calls.

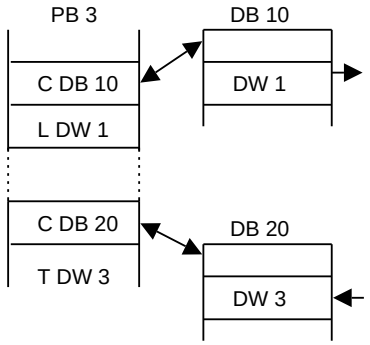
Example: A special function has been programmed in FB 63. It is called and processed under certain conditions, e.g. in PB 10.

Program Sequence	STL	Explanation
	. . . S F 10.0 A I 31.7 JC FB 63 .	The ”JC FB 63” statement in program block PB 10 calls function block FB 63 if input I 31.7 is ”1”.

Calling a data block "C DB"

Data blocks are always called unconditionally. All data processed following the call refers to the data block that has been called.
This operation cannot generate new data blocks. Blocks that are called must be programmed before program scanning.

Example: Program block PB 3 needs information that has been programmed as data word DW 1 in data block DB 10. Other data, e.g. the result of an arithmetic operation, is stored as data word DW 3 in data block DB 20.

Program Sequence	STL	Explanation
	C DB 10 L DW 1 . . . . C DB 20 T DW 3	The information from data word DW 1 in data block DB 10 is loaded into the accumulator. The contents of ACCUM 1 are stored in data word DW 3 of data block DB 20.

Generating and deleting a data block

The "G DB x" statement does not call a data block. Instead, it generates a new block. If you want to use the data in this data block, call it with the "C DB" statement.
Before the "G DB" statement, indicate in ACCUM 1 the number of data words the block is to have.

If you specify zero as the data block length, the data block in question is deleted, i.e. it is removed from the address list. It is considered nonexistent.

If you try to set up a data block that already exists, the "G DB x" statement is not executed.

The length of data blocks set up in this way is optional. However, please note that programmers can process blocks of limited length only.

Generating a data block

Example	STL	Explanation
Generate a data block with 128 data words without the aid of a programmer.	L KF + 127 **G DB 5**	The constant fixed-point number +127 is loaded into ACCUM 1. At the same time, the old contents of ACCUM 1 are shifted to ACCUM 2. Data block 5 is generated with a length of 128 data words in the RAM of the PLC and entered in the block address list. The next time the ”G DB5” operation is processed, it has no effect if the contents of ACCUM 1 are not 0.

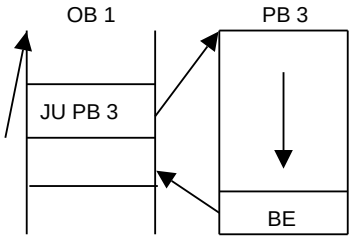
Deleting a data block

Example	STL	Explanation
Delete a data block that is no longer needed.	L KF + 0 **G DB 5**	The constant fixed-point number +0 is loaded into ACCUM 1. At the same time, the old contents of ACCUM 1 are shifted to ACCUM 2. Data block 5, which is in the RAM of the PLC, is declared invalid and removed from the block address list.

Block end ”BE”

The ”BE” operation terminates a block. Data blocks do not need to be terminated. ”BE” is always the last statement in a block.
In structured programming, program scanning jumps back to the block where the call for the current block was made.
Binary logic operations cannot be continued in a higher-order block.

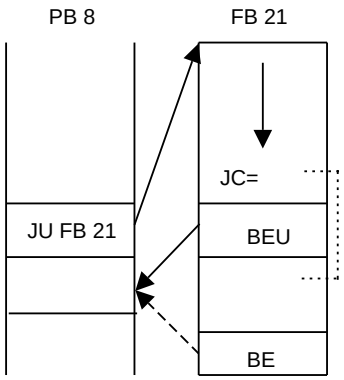
Example: Program block PB 3 is terminated by the ”BE” statement.

Program Sequence	STL	Explanation
	. . . . . . BE	The ”BE” statement terminates program block PB 3 and causes program scanning to return to organization block OB 1.

Unconditional block end ”BEU”

The ”BEU” operation causes a return within a block. However, jump operations can by-pass the ”BEU” operation in function blocks (3.2.9).
Binary logic operations cannot be continued in a higher-order block.

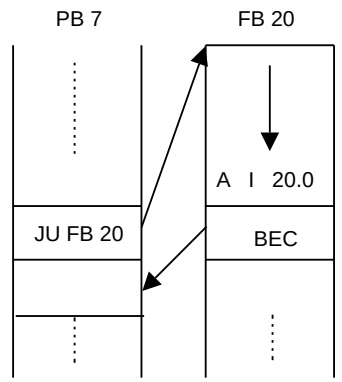
Example: Scanning of function block FB 21 is terminated regardless of the RLO.

Program sequence	STL	Explanation
	. . . . JC= BEU . . . . BE	The ”BEU” statement causes program scanning to leave function block FB 21 and return to program block PB 8.

Conditional block end ”BEC”

The ”BEC” operation causes a return within a block if the previous condition has been satisfied (RLO = 1).
Otherwise, linear program scanning is continued with RLO ”1”.

Example: Scanning of program block FB 20 is terminated if the RLO = ”1”.

Program Sequence	STL	Explanation
	. . . . S Q 1.0 A I 20.0 BEC	The ”BEC” statement causes program scanning to return to program block PB 7 from function block FB 20 if input I 20.0 is ”1”.

3.1.9 Other Operations

Table 3-9 lists other basic operations.

Table 3-9. Other Operations

Operation	Operand	Meaning
STP		**Stop at the end of program scanning.** Current program scanning is terminated. The PIQ is read out. Then the PLC goes into the ”STOP” mode.
NOP 0		**”No” Operation** Sixteen bits in the RAM are set to ”0”.
NOP 1		**”No” Operation** Sixteen bits in the RAM are set to ”1”.
BLD		**Display Generation Operation** ”BLD” means a display generation operation for the programmer.
ID	**Parameter** 130, 131, 132, 133, 255	
Method of representation: only STL possible		

Stop Operation

The ”STP” operation puts the PLC into the ”STOP” mode. This can be desirable for time-critical system circumstances or when a PLC error occurs.
After the statement is processed, the control program is scanned to the end, regardless of the RLO. Afterwards the PLC goes into the ”STOP” mode with the error ID ”STS”. It can be restarted with the mode selector (STOP RUN) or with a programmer.

”NOP” (No Operations)

The ”NOP” operations reserve or overwrite memory locations.

3.1.10 Segmentation with the BLD 255 Operation and the LPLZ Sequence in the Case of FBs

Within a block, program sections are subdivided into segments using ”BLD 255” display generation operations.

”NOP” operations and display generation operations are significant only for the programmer when representing the STEP 5 program.
The CPU does not execute any operations when these statements are processed. The BLD 255 statement has safety significance for the CPU 942F:

- An FB with more than 127 words of code must have a BLD 255 operation at the latest in the 128th word. The operation is followed by the following sequence
  LFW0
  LKF+1
  +F
  TFW0
  which increments the logical program counter.

- The test for the STEP address counter uses LPLZ sequences which
  - are located at the beginning of an FB
  - and follow BLD 255 operations at the 256 word boundary.

  Other BLD 255 operations have no significance for safety.

The following diagrams illustrate the structure of different blocks after segmentation.

> **Note**
>
> The BLD 255 operation can only be programmed in STL.

FB Header		5 W
Skip name and parameter list		1 W
Block name		4 W
Parameter list:	3 words, 6 characters/ parameter:	max. 40 parameters max. 120 W
LPLZ incrementation sequence		5 W
User operations:		max. 127 W
BE		1 W

Total length
max. 143 words (with 0 parameters)
max. 146 words (with 1 parameter)
:
max. 263 words (with 40 parameters)

Figure 3-6. Construction of a Single-Segment-FB for Parameters and Operations

FB Header:		5 W
Skip name and parameter list		1 W
Block name		4 W
Parameter list:	3 words, 6 characters/ parameter:	max. 40 parameters max. 120 W
LPLZ incrementation sequence		5 W
User operations:		max. 127 W
BLD 255 operations		1 W
LPLZ incrementation sequence		5 W
User operations:		max. 127 W
BLD 255 operations		1 W
BE in last segment instead of BLD 255		1 W

Figure 3-7. Structure of a Multi-Segment FB (max. 4096 words)

OB, PB, SB Header		5 W
LPLZ incrementation sequence		5 W
User operations:	max.	127 W
BE-operations		1 W

Figure 3-8. Structure of a Single-Segment OB, PB, SB (max. 261 words)

Header: OB, PB, SB		5 W
LPLZ incrementation sequence		5 W
User operations:	max.	127 W
BLD 255 operation		1 W
LPLZ incrementation sequence		5 W
User operations:	max.	127 W
BLD 255 operation		1 W
BE in last segment instead of BLD 255		1 W

Figure 3-9. Structure of a Multi-Segment OB, PB, SB (max. 4096 words)

Note

Only the last BLD 255 operation before the 128 word boundary need to be followed by a sequence for incrementing the LPLZ.

3.2 Supplementary Operations

Supplementary operations extend the operations set. However, compared to basic operations, which can be programmed in all blocks, supplementary operations have the following limitations:

- They can be programmed in function blocks only.
- They can be represented in STL form only.

Sections 3.2.1 through 3.2.10 describe the supplementary operations.

3.2.1 Load Operation

As with the basic load operations, the supplementary load operation copies information into the accumulator. Table 3-10 explains the load operation. An example follows the table.

Table 3-10. Load Operation

Operation	Operand		Meaning
L			**Load** A word from the system data is loaded into ACCUM 1 regardless of the RLO.
ID	BS	**Parameter**	0 to 255

3.2.2 Enable Operation

Use the enable operation (FR) to execute the following operations even without edge change:
- Start a timer
- Set a counter
- Count up and down

Table 3-11 presents the enable operation. An example follows the table.

Table 3-11. Enable Operation

Operation	Operand	Meaning
FR		**Enable a Timer/Counter** Timers and counters are enabled on the leading edge of the RLO. This operation restarts a timer, sets a counter, or causes a counter to count up or down when the RLO "1" is pending at the "Start" operation.
ID	**Parameter**	
T	0 to 127	
C	0 to 127	

Example	STL			Explanation
Input I 2.5 starts a timer T 2 as extended pulse (pulse width 50 sec.). This timer sets output Q 4.2 for the duration of the pulse.	A	I	2.5	
	L	KT	5.3 *	
	SE	T	2	Start a timer T 2 as extended pulse.
	A	T	2	
	=	Q	4.2	Output Q 4.2 is set for 50 sec.
.		.		
.		.		
.		.		
.		.		
If output Q 3.4 is reset repeatedly, the timer should also be restarted repeatedly.	A	Q	3.4	If output Q 3.4 is set (positive edge change of the RLO) during the time in which input I 2.5 is set, the timer T 2 is restarted. Output Q 4.2 therefore remains set at the restarted time or is reset. If input I 2.5 is not set during the edge change of output Q 3.4, the timer is not restarted.
	FR	**T**	**2**	
	BE			

***** This time has a tolerance of ±10 sec. Use a smaller time base if necessary.

3.2.3 Bit Test Operations

Bit test operations scan digital operands bit by bit and affect them. Bit test operations must always be at the beginning of a logic operation. Table 3-12 provides an overview of these operations.

Table 3-12. Overview of Bit Test Operations

Operation	Operand		Meaning
TB			**Test a bit for signal state "1".** A single bit is scanned regardless of the RLO. The RLO is affected according to the bit's signal state (Table 3-13).
TBN			**Test a bit for signal state "0".** A single bit is scanned regardless of the RLO. The RLO is affected according to the bit's signal state (Table 3-13).
SU			**Set a bit unconditionally.** The addressed bit is set to "1" regardless of the RLO. The RLO is not affected.
RU			**Reset a bit unconditionally.** The addressed bit is set to "0" regardless of the RLO. The RLO is not affected.
ID		**Parameter**	
	T	0 to 127.15	
	C	0 to 127.15	
	D	0 to 255.15	
	RS[1]	0 to 255.15	

1 RS applies only to TB and TBN.

Table 3-13 shows how the RLO is formed during the bit test operations "TB" and "TBN". An example for applying the bit operations follows the table.

Table 3-13. Effect of "TB" and "TBN" on the RLO

Operation	TB		TBN	
Signal state of the bit in the operation indicated	**0**	**1**	0	1
Result of logic operation	**0**	1	1	0

Example	STL	Explanation
A photoelectric barrier that counts piece goods is installed at input I 2.0. After every 100 pieces, the program is to jump to FB 5 or FB 6. After 800 pieces, counter 10 is to be reset automatically and start counting again.	C DB 10 A I 2.0 CU C 10 A I 3.0 L KC 0 S C 10 O I 4.0 O F 5.2 R C 10 LC C 10 T DW 12 **TBN D 12.8** JC FB 5 **TB D 12.8** JC FB 6 **TB D 12.11** = F 5.2	Call data block 10. Input I 3.0 loads the count of counter 10 with the constant 0. With each positive edge change at I 2.0, the counter is incremented by 1. The counter is reset by either input I 4.0 or flag F 5.2. The current count of the counter is stored in data word DW 12 in BCD code. As long as bit 8 of data word DW 12 is zero, program processing jumps to function block FB 5. This is the case for the counts 1, 3, 5 to 97, 99. As long as bit 8 of data word DW 12 is ”1”, program scanning jumps to function block FB 6. This is the case for the counts 2, 4, 6 to 98, 100. When data bit 11 of data word DW 12 becomes ”1” (the count is then 800), flag F 5.2 is set conditionally.
A photoelectric barrier that counts piece goods is installed at input I 10.0. After every 256 pieces, the counter is supposed to be reset and start counting again.	:A I 10.0 :CU C 20 :A I 11.0 :L KC 0 :S C 20 **:TB C 20.8** :JC = FULL :BEU **FULL:RU C 20.8** :BE	Input I 11.0 loads the count of counter 20 with the constant 0. The count is incremented by 1 with each positive edge change at input I 10.0. If the count has reached 256 = 100_H (bit 8 is ”1”), program scanning jumps to the label ”FULL”. Otherwise the block is terminated. Bit 8 of counter C 20 is set to ”0” unconditionally. Then the count is again 000_H.

Note

Times and counts are stored in the timer/counter word in hexadecimal notation in the 10 least significant bits (bits 0 to 9).
The time base is stored in bits 12 and 13 of the timer word.

3.2.4 Digital Logic Operations

Digital logic operations combine the contents of both accumulators logically bit by bit. Table 3-14 provides an overview of these digital logic operations. Examples follow the table.

Table 3-14. Overview of Digital Logic Operations

Operation	Operand	Meaning
AW		**Combine bit by bit through logic AND.**
OW		**Combine bit by bit through logic OR.**
XOW		**Combine bit by bit through EXCLUSIVE OR.**

Processing a digital logic operation

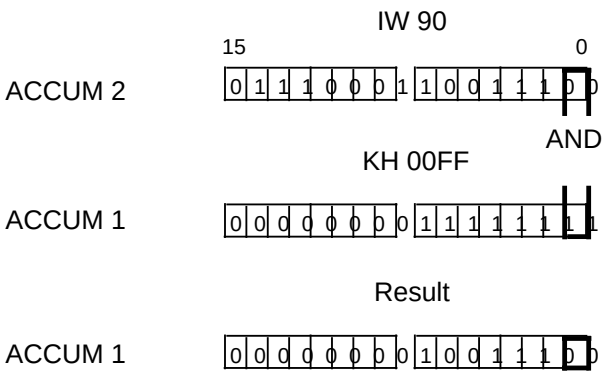
A digital logic operation is executed regardless of the RLO. It also does not affect the RLO. However, it sets condition codes according to the result of the arithmetic operation (3.4).

> **Note**
>
> Make sure that both operands have the same number format. Then load them into the accumulators before executing the operation.

The result of the arithmetic operation is available in ACCUM 1 for further processing. The contents of ACCUM 2 are not affected.

STL		Explanation
L	IW 92	Load input word IW 92 into ACCUM 1.
L	KH 00FF	Load a constant into ACCUM 1. The previous contents of ACCUM 1 are shifted to ACCUM 2.
AW		Combine the contents of both accumulators bit by bit through logic AND.
T	QW 82	Transfer the resulting contents from ACCUM 1 to output word QW 82.

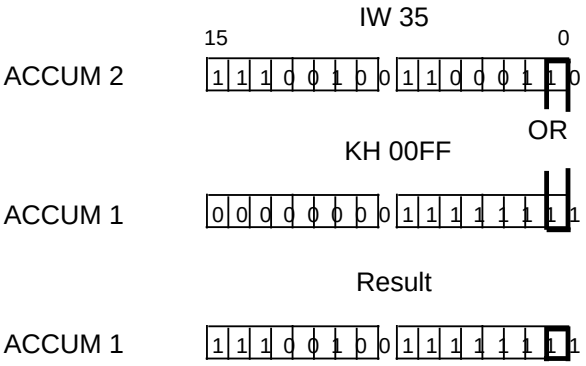
Numeric Example



Set the 8 high-order bits in input word IW 90 to ”0”.
Compare both words bit by bit. If corresponding bits are both ”1”, the result bit is set to ”1”.

STL		Explanation
L	IW 35	Load input word IW 35 into ACCUM 1.
L	KH 00FF	Load a constant into ACCUM 1. The previous contents of ACCUM 1 are shifted to ACCUM 2.
OW		Combine the contents of both accumulators bit by bit through logic OR.
T	IW 35	Transfer the result (contents of ACCUM 1) to input word IW 35.

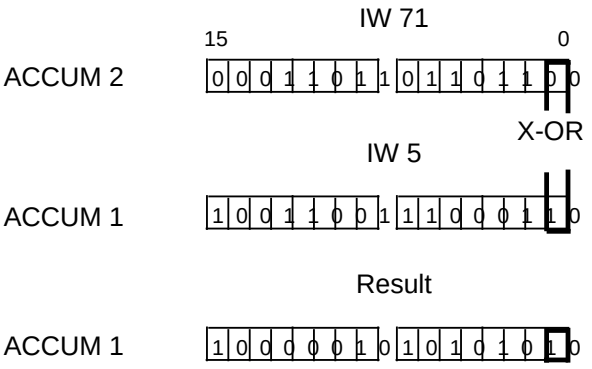
Numeric Example



Set the 8 low-order bits in input word IW 35 to ”1”. Compare both words bit by bit.
If corresponding bits are both ”1”, a ”1” is set in the result word.

STL		Explanation
L	IW 71	Load input word IW 71 into ACCUM 1.
L	IW 5	Load input word IW 5 into ACCUM 1. The previous contents of ACCUM 1 are shifted to ACCUM 2.
XOW		Combine the contents of both accumulators bit by bit through EXCLUSIVE OR.
T	QW 86	Transfer the result (contents of ACCUM 1) to output word QW 86.

Numeric Example



Check to see if input words IW 71 and IW 5 are equal. The result bit is set to ”1” only if corresponding bits in ACCUM 1 and ACCUM 2 are unequal.

3.2.5 Shift Operations

Shift operations shift a bit pattern in ACCUM 1. The contents of ACCUM 2 are not affected. Shifting multiplies or divides the contents of ACCUM 1 by powers of two. Table 3-15 provides an overview of the shift operations. Examples follow the table.

Table 3-15. Overview of Shift Operations

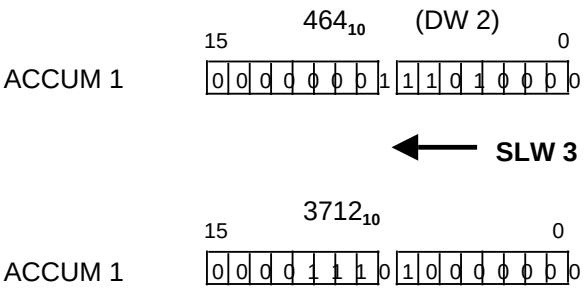
Operation	Operand		Meaning
SLW			**Shift to the left.** The bit pattern in ACCUM 1 is shifted to the left.
SRW			**Shift to the right.** The bit pattern in ACCUM 1 is shifted to the right.
		↑	**Parameter** 0 to 15

Processing a shift operation

Execution of shift operations is unconditional. The RLO is not affected. However, shift operations set condition codes.
Consequently, the status of the last bit that is shifted out can be scanned with jump functions.
The shift statement parameter indicates the number of bit positions by which the contents of ACCUM 1 are to be shifted to the left (SLW) or to the right (SRW). Bit positions vacated during shifting are assigned zeros.
The contents of the bits that are shifted out of ACCUM 1 are lost.
A shift operation with parameter ”0” is handled like a ”NOP” operation. The central processor processes the next STEP 5 statement with no further reaction.
Before executing a shift operation, load the operand to be processed into ACCUM 1. The altered operand is available there for further processing.

STL		Explanation
L	DW 2	Load the contents of data word DW 2 into ACCUM 1.
SLW	3	Shift the bit pattern in ACCUM 1 three positions to the left.
T	DW 3	Transfer the result (contents of ACCUM 1) to data word DW 3.

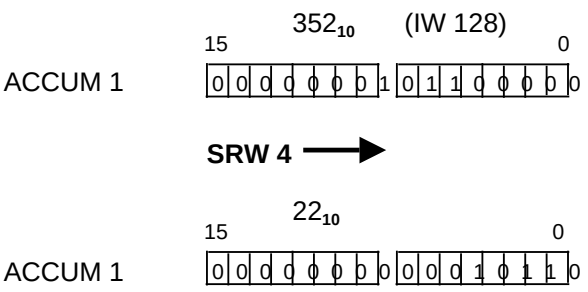
Numeric Example



The value 464_{10} is stored in data word DW 2. Multiply this value by 2^3=8. Do so by shifting the bit pattern of DW 2 in ACCUM 1 three positions to the left.

STL		Explanation
L	IW 128	Load the value of input word IW 128 into ACCUM 1.
SRW	4	Shift the bit pattern in ACCUM 1 four positions to the right.
T	QW 160	Transfer the result (contents of ACCUM 1) to output word QW 160.

Numeric Example



The value 352_{10} is stored in IW 128. Shift the corresponding bit pattern in ACCUM 1 four positions to the right to divide the value 352_{10} by 2^4 = 16.

3.2.6 Conversion Operations

Conversion operations convert the values in ACCUM 1. Table 3-16 provides an overview of the conversion operations. Examples follow the table.

Table 3-16. Overview of Conversion Operations

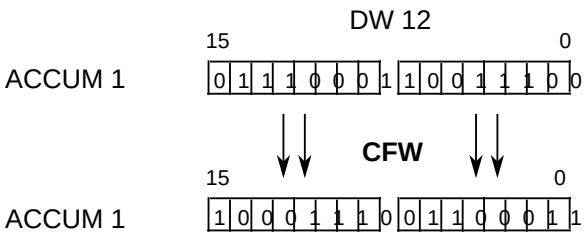
Operation	Operand	Meaning
CFW		**One's complement** The contents of ACCUM 1 are inverted bit by bit.
CSW		**Two's complement** The contents of ACCUM 1 are inverted bit by bit. Afterwards the word 0001_H is added.

Processing conversion operations

Execution of these operations does not depend on the RLO nor does it affect the RLO. The ”CSW” operation sets the condition codes (3.4).

STL		Explanation
L	DW 12	Load the contents of data word DW 12 into ACCUM 1**.**
CFW		Invert all bits in ACCUM 1.
T	QW 20	Transfer the new contents of ACCUM 1 to output word QW 20**.**

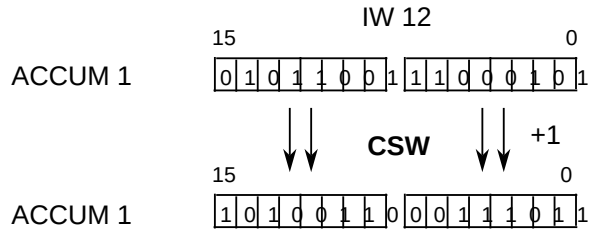
Numeric Example



In a system, normally open contacts have been replaced by normally closed contacts. If the information in data word DW 12 is to maintain its previous effect, DW 12 must be inverted.

STL		Explanation
L	IW 12	Load the contents of input word IW 12 into ACCUM 1.
CSW		Invert all bits. Add a ”1” at the least significant position.
T	DW 100	Transfer the altered word to data word DW 100.

Numeric Example



Form the negative value of the value in input word IW 12.

3.2.7 Decrement/Increment

The decrement/increment operations change the data loaded into ACCUM 1. Table 3-17 provides an overview of the decrement/increment operations. An example follows the table.

Table 3-17. Decrement/Increment Operations

Operation	Operand		Meaning
D			**Decrement** Decrement the contents of the accumulator.
I			**Increment** Increment the contents of the accumulator. The contents of ACCUM 1 are either decremented or incremented by the number indicated in the parameter. Execution of the operation is unconditional and is limited to the right-hand byte (without carry).
		Parameter	0 to 255

Processing

Execution of the decrement and increment operations is independent of the RLO and does not affect the RLO or the condition codes.
The parameter indicates the value by which the contents of ACCUM 1 are to be changed. The operations refer to decimal values; however, the result is stored in ACCUM 1 in binary form. Changes relate only to the low byte in the accumulator.

Example	STL		Explanation
Increment the hexadecimal constant 1010_H by 16 and store the result in data word DW 8.	C	DB 6	Call data block DB 6.
	L	KH 1010	Load hexadecimal constant 1010_H into ACCUM 1.
	I	16	Increment the low byte of ACCUM 1 by 16. The result, 1020_H, is located in ACCUM 1.
In addition, decrement the incrementation result by 33 and store the new result in data word DW 9.	T	DW 8	Transfer the contents of ACCUM 1 (1020_H) to data word DW 8. Since the incrementation result is still in ACCUM 1, you can decrement by 33 directly. The result would be $0FFF_H$. However,
	D	33	since the high byte of ACCUM 1 is not decremented along with the low byte, the result in ACCUM 1 is $10FF_H$.
	T	DW 9	The contents of ACCUM 1 are transferred to DW 9 ($10FF_H$).

3.2.8 Disable/Enable Interrupt

The disable/enable interrupt operations affect interrupt and time driven program scanning. They prevent process or time interrupts from interfering with the processing of a sequence of statements or blocks. Table 3-18 lists the disable/enable interrupt operations. An example follows the table.

Table 3-18. Disable/Enable Interrupt Operations

Operation	Operand		Meaning
AS			**Disable interrupt**
AF			**Enable interrupt**

Processing

Execution of the disable/enable interrupt operations does not depend on the RLO. These operations do not affect the RLO or the condition codes. After the ”IA” statement is processed, no more interrupts are executed. The ”RA” statement cancels the effect of ”IA”.

Note

If interrupt processing is not required, you can reduce the PLC scan time.
Inhibit interrupt processing with the IA operation and switch off interval monitoring when setting the operating system parameters with COM 115F.

Example	STL	Explanation
Disable interrupt processing in a specific program section and then enable it again.	. . . . . = Q 7.5	
	IA	Disable interrupt.
	A I 2.3	
	JU FB 254	Synchronization FB call for servicing process/time interrupts
	NAME: SYNC	
	SYNC: KM000-0010	
	:	
	:	
	:	
	JU FB 3	If an interrupt occurs, the program section between the operations IA and RA is scanned without interruption.
	.	
	.	
	.	
	RA	Enable interrupt. Interrupts that occurred in the meantime are processed after the ”RA” operation.[1]
	.	
	.	
	.	

1 Only one interrupt can be stored for each interrupt line.

3.2.9 "DO" Operation

Use the "DO" operation to process STEP 5 statements as indexed operations. This allows you to change the parameter of an operand during control program processing (see Table 3-19).

Table 3-19. Overview of the "DO" Operation

Operation	Operand		Meaning
DO			**Processing a flag word or data word**
	ID ↑ FW DW	↑ **Parameter** 0 to 254 0 to 255	

"DO" Statements

"DO flag word or data word x" is a two-word statement that is unaffected by the RLO.

"DO" consists of the following two statements:

- The first statement contains the "DO" operation and a flag word or data word.
- The second statement defines the operation and the operand identifier you want the control program to process. **You must enter 0 or 0.0 as the parameter**.

The control program works with the parameter that is stored in the flag word or data word. This parameter is the one called up in the first statement. If you want to index binary operations, inputs, outputs, or flags, you input the bit address in the high byte of this word. You input the byte address in the low byte.
In any other instance, the high byte must be "0".

The following operations can be combined with the "DO" statement:

Operations	Explanations
`L FY, T FY, L FW, T FW, L IB, T IB, L QB, T QB, L IW, T IW, L QW, T QW, L DL, T DL, L DR, T DR, L DW, T DW`	Load and transfer operations
`CDB, JU OB, JU SB, JU FB, JU PB, JC OB, JC SB, JC FB, JC PB`	Block call operations (FBs must have no formal parameters)
`A F, = F, A T, AN T`	Boolean logic operations
`S F, R F, R T`	Set/reset operations
`SS T, SE T`	Timer operations
`SLW, SRW`	Shift operations



Warning

Operations other than that listed in the above table are not permissible and will be rejected by the operating system.

Figure 3-10 shows how the contents of a data word determine the parameter of the next statement.

	DB6
DW 12	KH=**0108**
DW 13	KH=0001

```
FB x
:C   DB   6
      .
      .
      .
:DO  DW   12
:A   F    0.0
:DO  DW   13
:FR  T    0
```

Actual program

```
:C   DB   6
      .
      .
      .
      .
:A   F    8.1
      .
:FR  T    1
```

Figure 3-10. Executing a “DO” Operation

The following example illustrates how new parameters are generated in every program scan.

Example	STL	Explanation
Set the contents of data words DW20 to DW100 to signal state “0”. The index register for the parameter for the data words is DW1.	:C DB 202	Call data block DB202.
	:L KB 20	Load constant number 20 in ACCU 1.
	:T DW 1	Transfer contents from ACCU 1 to data word DW1.
	F 1 :L KH 0	Load hex constant 0 in ACCU 1.
	:DO DW 1	DO data word DW1.
	:T DW 0	Transfer the contents from ACCU 1 to the data word whose address is stored in data word DW1.
	:L DW 1	Load data word DW1 in ACCU 1.
	:L KB 1	Load constant number 1 in ACCU 1. Data word DW1 is shifted to ACCU 2.
	:+F	ACCU 2 und ACCU 1 are added, and the result is stored in ACCU 1 (data word address is higher).
	:T DW 1	Transfer contents of ACCU 1 to data word DW1 (new data word address).
	:L KB 100	The constant number 100 is loaded in ACCU 1 and the new data word address is shifted to ACCU 2.
	:<=F	Compare the ACCUs for less than or equal to: ACCU 2 ACCU 1.
	:JC = F 1	Jump conditionally to label F1, if ACCU 2 ACCU 1.

3.2.10 Dump Operations

Table 3-20 provides an overview of the jump operations. An example follows the table.

Table 3-20. Overview of Jump Operations

Operation	Operand		Meaning
JU =			**Jump unconditionally.** The unconditional jump is executed independently of conditions.
JC =			**Jump conditionally.** The conditional jump is executed if the RLO is ”1”. If the RLO is ”0”, the statement is not executed and the RLO is set to ”1”.
JZ =			**Jump if the result is ”zero”.** The jump is executed only if CC 1 = 0 and CC 0 = 0. The RLO is not changed.
JN =			**Jump if the result is ”not zero”.** The jump is executed only if CC 1 CC 0. The RLO is not changed.
JP =			**Jump if the result is positive.** The jump is executed only if CC 1 = 1 and CC 0 = 0. The RLO is not changed.
JM =			**Jump if the result is negative.** The jump is executed only if CC 1 = 0 and CC 0 = 1. The RLO is not changed.
JO =			**Jump on overflow.** The jump is executed if an overflow occurs. Otherwise the jump is not executed. The RLO is not changed.
ID Jump label (up to 4)			

Processing the jump operations

A symbolic jump destination (jump label) must always be entered next to a jump operation. This jump label can have up to four characters, the first of which must be a letter of the alphabet.

When programming, please be aware of the following:

- The absolute jump displacement cannot exceed + 127 or - 128 words in the program memory. Some statements take up two words (e.g. ”Load a constant”). For long jumps, insert an intermediate destination.
- Jumps can be executed only within a block.
- Jumping over segment boundaries (”BLD 255”) is not permitted, i.e. jumps to networks are illegal.
- Jumps from expressions in parentheses are not permitted.

Example	STL			Explanation
If no bit of input word IW 1 is set, program scanning jumps to the label ”AN 1”. If input word IW 1 and output word QW 3 do not agree, program processing jumps back to the label ”AN 0”. Otherwise input word IW 1 and data word DW 12 are compared. If input word IW 1 is greater than or less than data word DW 12, program scanning jumps to the ”Destination” label.	AN0	:L	IW 1	Load input word IW 1 into ACCUM 1. If the contents of ACCUM 1 equal zero*, jump to the label ”AN 1”. Otherwise process the next statement (A I 1.0).
		:L	KH 0000	
		:+F		
		:JZ=	**AN 1**	
		:A	I 1.0	
			.	
			.	
			.	
			.	
			.	
			.	
	AN1	:L	IW 1	Compare input word IW 1 and output word QW 3. If they are not equal, set individual bits in ACCUM 1.
		:L	QW3	
		:XOW		
		:JN =	**AN 0**	If the contents of ACCUM 1 are not zero, jump to the label ”AN 0”. Otherwise process the next statements.
		:L	IW 1	Compare input word IW 1 and data word DW 12. If they are not equal, set RLO to ”1”.
		:L	DW12	
		:><	F	
		:JC =	**Destination**	If the RLO = ”1”, jump to the ”Destination” label. If the RLO = ”0”, process the next statement.
			.	
			.	
			.	
			.	
			.	
	DEST	:A	I 12.2	
			.	
			.	

* The ”L...” statement does not affect the condition codes. An addition (+F) is executed with the constant 0000_H so that the ”JZ” operation can evaluate the contents of the accumulator.

3.2.11 Substitution Operations

If you plan to process a program with various operands and without a lot of changes, it is advisable to assign parameters to individual operands (2.3.4).
If you have to change the operands, you only need to reassign the parameters in the function block call.
These parameters are processed in the program as ”formal operands”.
Special operations are necessary for this processing. However, these special operations are no different in their effect than operations without substitution. A brief description of these operations and examples follows.

Binary logic operations

Table 3-21 provides an overview of binary logic operations.

Table 3-21. Overview of Binary Logic Operations

Operation	Operand	Meaning
A =		**AND operation** Scan a formal operand for ”1”.
AN =		**AND operation** Scan a formal operand for ”0”.
O =		**OR operation** Scan a formal operand for ”1”.
ON =		**OR operation** Scan a formal operand for ”0”.

Formal operand	Actual operands permitted	Parameter Type	Data Type
	Inputs, outputs, and flags addressed in binary form	I, Q	BI
	Timers and counters	T , C	

Set/Reset Operations

Table 3-22 provides an overview of the set/reset operations. An example follows the table.

Table 3-22. Overview of Set/Reset Operations

Operation	Operand	Meaning
S =		**Set a formal operand (binary).**
RB =		**Reset a formal operand (binary).**
= =		**Assign** The RLO is assigned to a formal operand.

Formal operand	Actual operands permissed	Parameter Type	Data Type
	Inputs, outputs, and flags addressed in binary form	I, Q	BI

Example: FB 30 is assigned parameters in OB 1.

Call in OB 1	Program in FB 30		Executed Program		
:SPA FB 30	**:A**	**=ON 1**	:A	I	2.0
NAME :LOGIC	**:AN**	**=ON 2**	:AN	I	2.1
ON 1 : I 2.0	**:O**	**=ON 3**	:O	I	2.2
ON 2 : I 2.1	**:S**	**=MOT 5**	:S	Q	7.3
ON 3 : I 2.2	**:=**	**=OFF 1**	:=	Q	7.1
VAL1 : I 2.3	**:A**	**=VAL 1**	:A	I	2.3
OFF1 : Q 7.1	**:A**	**=ON 2**	:A	I	2.1
OFF2 : Q 7.2	**:ON**	**=ON 3**	:ON	I	2.2
MOT5 : Q 7.3	**:RB**	**=MOT 5**	:R	Q	7.3
: BE	**:=**	**=OFF 2**	:=	Q	7.2
	:BE		:BE		

Load and transfer operations

Table 3-23 provides an overview of the load and transfer operations. An example follows the table.

Table 3-23. Overview of Load and Transfer Operations

Operation	Operand	Meaning
L =		**Load a formal operand.**
LC =		**Load a formal operand in BCD code.**
LW =		**Load the bit pattern of a formal operand.**
T =		**Transfer to a formal operand.**

Formal operand	Actual operands permitted	Parameter Type	Data Type
For L =	Inputs, outputs, and data words[1] addressed in binary form Timers and counters	I , Q T , C	BY, W
For LC =	Timers and counters	T , C	
For LW =	Bit pattern	D	KF, KH, KM, KY, KS, KT, KC
For T =	Inputs, outputs, and data words[1] addressed in binary form	I, Q	BY, W

1 DW, DR, DL

Example: FB 34 is assigned parameters in PB 1.

Call in PB 1	Program in FB 34			Executed Program		
:JU FB 34	:A	=I	0	:A	I	2.0
NAME :LOAD/TRAN	**:L**	**=L**	**1**	:L	FW	10
I 0 : I 2.0	:S	C	6	:S	C	6
I 1 : I 2.1	:A	=I	1	:A	I	2.1
L 1 : FW 10	**:LW**	**=LW**	**1**	:L	KC	140
LW 1 : KC 140	:S	C	7	:S	C	7
LC 1 : C 7	:A	I	2.2	:A	I	2.2
T 1 : QW 4	:CU	C	6	:CU	C	6
LW 2 : KC 160	:CU	C	7	:CU	C	7
:BE	**:LC**	**=LC**	**1**	:LC	C	7
	:T	**=T**	**1**	:T	QW	4
	:A	I	2.7	:A	I	2.7
	:R	C	6	:R	C	6
	:R	C	7	:R	C	7
	:LW	**=LW**	**2**	:L	KC	160
	:LC	**=LC**	**1**	:LC	C	7
	:!=F			:!=F		
	:R	C	7	:R	C	7
	:BE			:BE		

Timer and counter operations

Table 3-24 provides an overview of timer and counter operations. Examples follow the table.

Table 3-24. Overview of Timer and Counter Operations

Operation	Operand	Meaning
FR =		**Enable** a formal operand for cold restart. (For a description, see ”FT or ”FC”, according to the formal operand.)
RD =		**Reset** a formal operand (digital).
SP =		**Start** a pulse timer specified as a formal operand using the value stored in the accumulator.
SR =		**Start** an on-delay timer as a formal operand using the value stored in the accumulator.
SEC =		**Start** an extended pulse timer specified as a formal operand using the value stored in the accumulator or **set** a counter specified as a formal operand using the count specified in the accumulator.
SSU =		**Start** a stored on-delay timer specified as a formal operand using the value stored in the accumulator or start the **count up** of a counter specified as a formal operand.
SFD =		**Start** an off-delay timer specified as a formal operand using the value stored in the accumulator or start the **count down** of a counter specified as a formal operand.

Formal operand	Actual operands permitted	Parameter Type	Data Type
	Timers and counters [1]	T, C[1]	

1 ”SP” and ”SR” do not apply to counters.

Specifying times and counts

As with the basic operations, you can specify a time or count as a formal operand. In this case, you must distinguish as follows whether the value is located in an operand word or is specified as a constant.

- Operand words can be of parameter type ”I” or ”Q” and of data type ”W”.
  Use the ”L=” operation to load them into the accumulator.
- Constants can be of parameter type ”D” and of data type ”KT” or ”KC”.
  Use ”LW=” to load these formal operands into the accumulator.

The following examples show how to work with timer and counter operations.

Example 1:

Function Block Call	Program in Function Block (FB32)	Executed Program
:JU FB 32 NAME :TIME I 5 : I 2.5 I 6 : I 2.6 TIM 5 : T 5 TIM 6 : T 6 OUT 6: Q 7.6 :BE	:AN =I 5 :U =I 6 :L KT 5.2 **:SFD =TIM 5** :A =I 5 :AN =I 6 :L KT 5.2 **:SSU =TIM 6** :A =TIM 5 :O =TIM 6 := =OUT 6 :A I 2.7 **:RD =TIM 5** **:RD =TIM 6** **:BE**	:AN I 2.5 :A I 2.6 :L KT 5.2 :SF T 5 :A I 2.5 :AN I 2.6 :L KT 5.2 :SS T 6 :A T 5 :O T 6 := Q 7.6 :A I 2.7 :R T 5 :R T 6 :BE

Example 2:

Function Block Call	Program in Function Block (FB33)	Executed Program
STL :JU FB 33 NAME :COUNT I 2 : I 2.2 I 3 : I 2.3 I 4 : I 2.4 COU 5: C 5 OUT 3: Q 7.3 :BE	:A =I 2 :L KC 17 **:SEC =COU 5** :A =I 3 **:SSU =COU 5** :A =I 4 **:SFD =COU 5** :A =COU 5 := =OUT 3 :A I 2.7 **:RD =COU 5** :BE	:A I 2.2 :L KC 17 :S C 5 :A I 2.3 :CU C 5 :A I 2.4 :CD C 5 :A C 5 := Q 7.3 :A I 2.7 :R C 5 :BE

DO operation

Table 3-25 and the example that follows explain the processing operation.

Table 3-25. DO Operation

Operation	Operand	Meaning		
B =		**Process formal operand.** The substituted blocks are called unconditionally.		
Formal operand		**Actual Operands Permitted**	**Parameter Type**	**Data Type**
		DB, PB, SB, FB [1]	B	

1 As actual operands, function blocks cannot have block parameters.

Example:

Function Block Call			Program in Function Block FB35			Executed Program		
STL								
	:JU	FB 35	**:DO**	**=D**	**5**	:C	DB	5
NAME	:PROCES		:L	=DW	2	:L	DW	2
D 5	:	DB 5	**:DO**	**=D**	**6**	:C	DB	6
DW 2	:	DW 2	:T	=DW	1	:T	DW	1
D 6	:	DB 6	:T	=Q	4	:T	QW	4
DW 1	:	DW 1	**:DO**	**=MOT**	**5**	:JU	FB	36
Q 4	:	QW 4	:BE			:BE		
MOT 5	:	FB 36						
	:BE							

3.3 System Operations

System operations and supplementary operations have the same limitations.
You can program them only as follows:

- In function blocks
- In the STL method of representation

Since system operations access system data, only users with system knowledge should use them.
If you want to program system operations, you must select ”SYS: OPS. Y” in the presets menu.

3.3.1 Set Operations

Like the supplementary bit operations, these set operations can change individual bits.
Table 3-26 provides an overview of the set operations.

Table 3-26. Overview of Set Operations

Operation	Operand		Meaning
SU			**Set bit unconditionally.** A specific bit is set to ”1” in the system data range.
RU			**Reset bit unconditionally.** A specific bit is set to ”0” in the system data range.
ID	RS	**Parameter**	0.0 to 255.15

Processing set operations

Execution of set operations does not depend on the RLO.

> **Important**
>
> These operations can only be used by personnel with a thorough knowledge of the S5-115F operating system.

3.3.2 Load and Transfer Operations

Use these load and transfer operations to address the entire program memory of the CPU. They are used mainly for data exchange between the accumulator and memory locations that cannot be addressed by operands. Table 3-27 provides an overview of the load and transfer operations.

> **Note**
>
> Load and transfer operations may only access memory area $B000_H$ to $C3FF_H$ (LIR, TIR, TNB).

Table 3-27. Overview of Load and Transfer Operations

Operation	Operand		Meaning
LIR			**Load the register indirectly.** The contents of the memory word whose address is in ACCUM 1 are loaded into the specified register.
TIR			**Transfer the register indirectly.** The contents of the indicated register are transferred to the memory location whose address is in ACCUM 1.
		↑	**Parameter** 0 (for ACCUM 1), 2 (for ACCUM 2)
TNB			**Transfer a data block (byte by byte).** A memory area is transferred in the program memory as a block: ACCUM 1 = Destination address ACCUM 2 = Source address Parameter = Length in bytes
T			**Transfer** A word is transferred to the system data range. Parameter = Word No. in system data area.
ID (System data)	↑ BS	↑	**Parameter** 0 to 255

Loading and transferring register contents

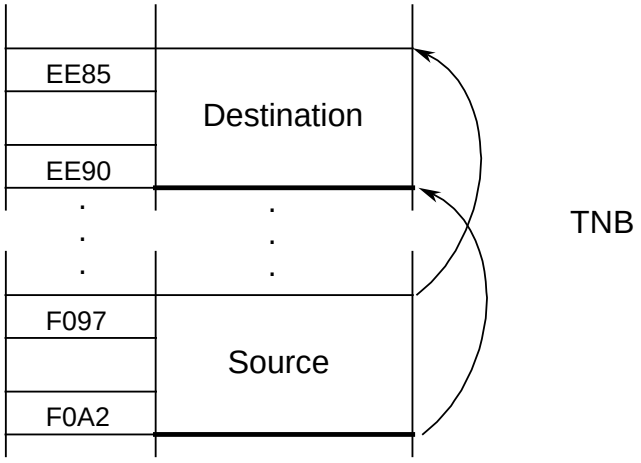
Both accumulators can be addressed as registers. Each register is 16 bits wide. Since the ”LIR” and ”TIR” operations transmit data by words, the S5-115F registers are addressed in pairs.
Loading and transferring register contents are independent of the RLO. The processor goes to ACCUM 1 to get the address of the memory location referenced during data exchange.
Consequently, make sure that the desired address is stored in ACCUM 1 before this system operation is processed.

Example:

LST			Explanation
	.		
	.		
L	KH	F100	Load the address $F100_H$ into ACCUM 1.
LIR	0		Load the information from the memory location with the address $F100_H$ into ACCUM 1.

Processing a field transfer

A field transfer is processed independently of the RLO.
The parameter indicates the length of the data field (in bytes) that is to be transferred. The field can be up to 255 bytes long.
The address of the source field is in ACCUM 2. The address of the destination field is in ACCUM 1.
The highest address of each field must be specified. The bytes in the destination field are overwritten during the transfer.

Example	Representation
Transfer a 12-byte data field from address $F0A2_H$ to address $EE90_H$.	

STL			Explanation
:L	KH	F0A2	Load the upper address of the source field into ACCUM 1.
:L	KH	EE90	Load the address of the destination field into ACCUM 1. The source address is shifted to ACCUM 2.
:TNB	12		Transfer the data field to the destination field.

3.3.3 Arithmetic Operation

Arithmetic operations increase the contents of ACCUM 1 by a specified value. The parameter represents this value as a positive or negative number. Table 3-28 shows the essential features of the ”ADD” operation. An example follows the table.

Table 3-28. Overview of the ”ADD” Operation

Operation	Operand		Meaning
ADD			**Add a constant.** Add byte or word constants.
ID		**Parameter**	
	BF		-128 to +127
	KF		-32768 to +32767

Processing

An arithmetic operation is executed independently of the RLO. It does not affect the RLO or the condition codes.
You can subtract by entering a negative parameter.
Even if the result cannot be represented by 16 bits, no carry is made to ACCUM 2, i.e. the contents of ACCUM 2 are not changed.

Example	STL			Explanation
Decrement the constant 1020_H by 33 and store the result in flag word FW 28. Afterwards add the constant 256 to the result and store the sum in flag word FW 30.	L	KH	1020	The constant 1020_H is loaded into ACCUM 1.
	ADD	BF	-33	The constant -33_{10} is added to the ACCUM contents.
	T	FW	28	The new ACCUM contents ($0FFF_H$) are stored in flag word FW 28.
	ADD	KF	256	The constant 256_{10} is added to the last result.
	T	FW	30	The new ACCUM contents ($10FF_H$) are stored in flag word FW 30.

3.3.4 Other Operations

Table 3-29 provides an overview of further system operations.

Table 3-29. The ”TAK” and ”STS” and ”DI” Operations

Operation	Operand	Meaning
TAK		**Exchange accumulator contents** The contents of ACCUM 1 and ACCUM 2 are exchanged independent of the RLO. The RLO and condition codes are not affected.
STS		**Immediate Stop** The CPU is brought to STOP independently of the RLO.
DI*	Formal operand	**Execute indirect** An operation is indicated via a formal operand.

***** DI must not be used in safety-related program sections

Processing the ”STS” operation

When the ”STS” operation is executed, the CPU goes into the ”STOP” mode immediately. Program scanning is terminated at this point. The ”STOP” state can only be canceled manually (with the mode selector) or with the programmer function ”PLC START”.

3.4 Condition Code Generation

The processor of the S5-115F programmable controller has the following three condition codes:
- CC 0
- CC 1
- OV (overflow)

The following operations affect the condition codes:
- Comparison operations
- Arithmetic operations
- Shift operations
- Some conversion operations

The state of the condition codes represents a condition for the various jump operations.

Condition code generation for comparison operations

Execution of comparison operations sets condition codes CC 0 and CC 1 (Table 3-30). The overflow condition code is not affected. However, comparison operations affect the RLO. When a comparison is satisfied, the RLO is 1. Consequently, the conditional jump operation ”JC” can also be used after a comparison operation.

Table 3-30. Condition Code Settings for Comparison Operations

Contents of ACCUM 2 as compared to contents of ACCUM 1	Condition codes			Possible jump operations
	CC 1	CC 0	OV	
Equal to	0	0		JZ
Less than	0	1		JN, JM
Greater than	1	0		JN, JP

Condition code generation for arithmetic operations

Execution of arithmetic operations sets all condition codes according to the result of the arithmetic operation (Table 3-31).

Table 3-31. Condition Code Settings for Fixed-Point Arithmetic Operations

Result after arithmetic operation is executed	Condition codes			Possible jump operations
	CC 1	CC 0	OV	
< - 32768	1	0	1	JN, JP, JO
- 32768 to - 1	0	1	0	JN, JM
0	0	0	0	JZ
+1 to +32767	1	0	0	JN, JP
> +32767	0	1	1	JN, JM, JO
(-) 65536 *	0	0	1	JZ, JO

***** This number is the result of the calculation -32768 - 32768

Conditon code generation for digital logic operations

Digital logic operations set CC 0 and CC 1. They do not affect the overflow condition code (Table 3-32). The setting depends on the contents of the ACCUM after the operation has been processed.

Table 3-32. Condition Code Settings for Digital Logic Operations

Value of the last bit shifted out	Condition codes			Possible jump operations
	CC 1	CC 0	OV	
Zero (KH = 0000)	0	0		JZ
Not zero	1	0		JN, JP

Conditon code generation for shift operations

Execution of shift operations sets CC 0 and CC 1. It does not affect the overflow condition code (Table 3-33).
Code setting depends on the state of the last bit shifted out.

Table 3-33. Condition Code Settings for Shift Operations

Value of the last bit shifted out	Condition codes			Possible jump operations
	CC 1	CC 0	OV	
”0”	0	0		JZ
”1”	1	0		JN, JP

Condition code generation for conversion operations

The formation of the two's complement (CSW) sets all condition codes (Table 3-34). The state of the condition codes is based on the result of the conversion function.

Table 3-34. Condition Code Settings for Conversion Operations

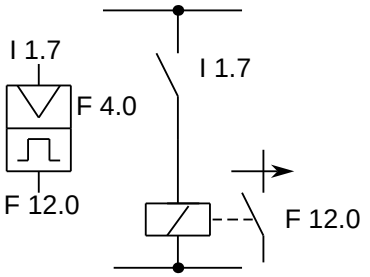
Result after arithmetic operation is executed	Condition codes			Possible jump operations
	CC 1	CC 0	OV	
- 32768*	0	1	1	JN, JP, JO
- 32767 to - 1	0	1	0	JN, JM
0	0	0	0	JZ
+1 to +32767	1	0	0	JN, JP

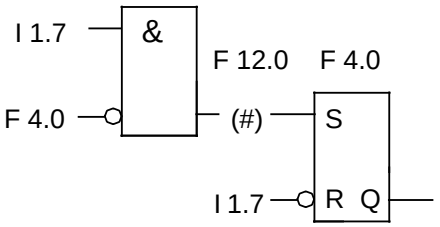
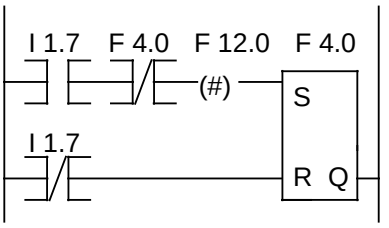
***** This number is the result of the conversion of KH = 8000**.**

3.5 Sample Programs

Sections 3.5.1 through 3.5.3 provide a few sample programs that you can enter and test in all three methods of representation on a programmer with a screen (e.g. the PG 675).

3.5.1 Transitional-Pulse Relay (Edge Evaluation)

Example	Circuit Diagram
On each leading edge of the signal at input I 1.7, the AND condition ”A I 1.7 and AN F 4.0” is satisfied; the RLO is ”1”. This sets flags F 4.0 and F 12.0 (”edge flags”). In the next processing cycle, the AND condition ”A I 1.7 and AN F 4.0” is not satisfied since flag F 4.0 has already been set. Flag F 12.0 is reset. Therefore, flag F 12.0 is ”1” for only one program run. When input I 1.7 is switched off, flag F 4.0 is reset. This resetting prepares the way for evaluation of the next leading edge of the signal at input I 1.7.	

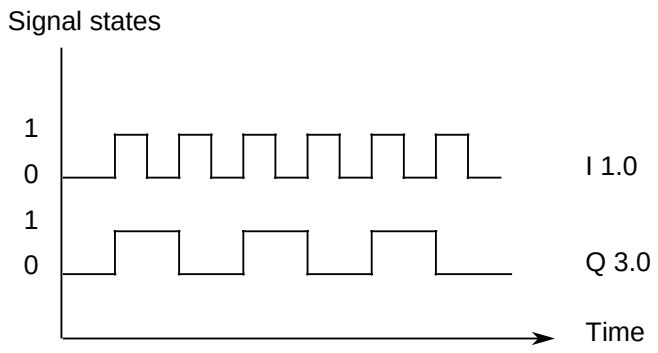
STL	CSF	LAD
A I 1.7 AN F 4.0 = F 12.0 A F 12.0 S F 4.0 AN I 1.7 R F 4.0 **NOP 0**		

3.5.2 Binary Scaler

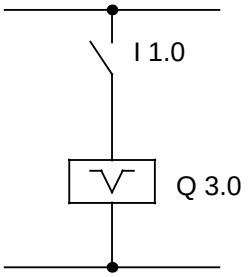
This subsection describes how to program a binary scaler.

Example: The binary scaler (output Q 3.0) changes its state each time I 1.0 changes its signal state from ”0” to ”1” (leading edge). Therefore, half the input frequency appears at the output of the memory cell.

Timing Diagram



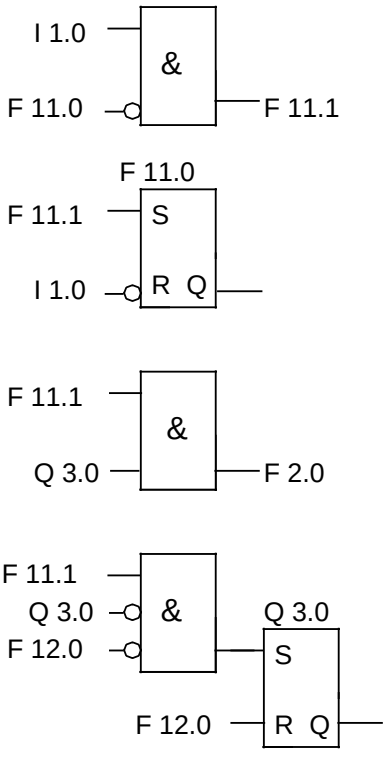
Circuit Diagram



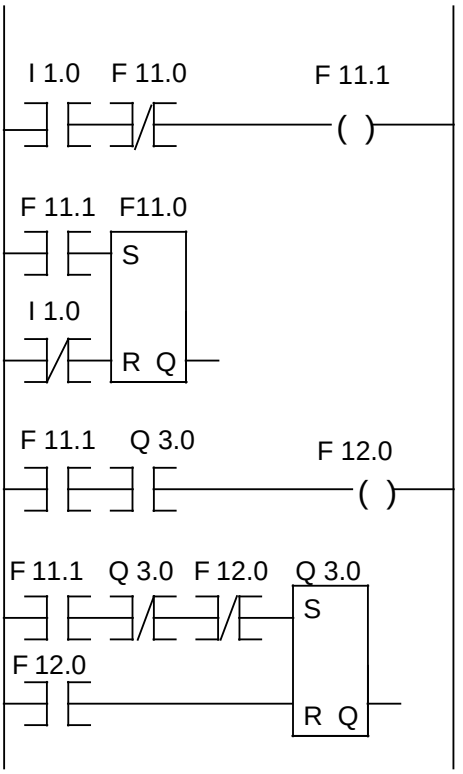
STL

```
A     I     1.0
AN    F    11.0
=     F    11.1
***
A     F    11.1
S     F    11.0
AN    I     1.0
R     F    11.0
NOP 0
***
A     F    11.1
A     Q     3.0
=     F    12.0
***
A     F    11.1
AN    Q     3.0
AN    F    12.0
S     Q     3.0
A     F    12.0
R     Q     3.0
NOP 0
```

CSF



LAD



Note

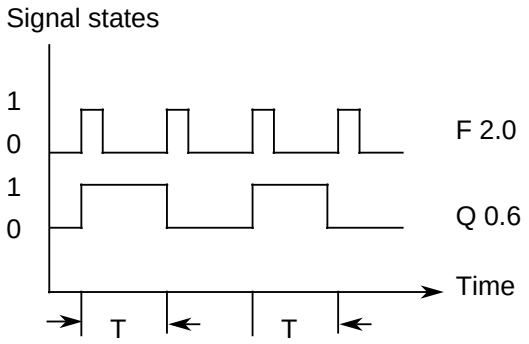
Output in CSF or LAD is possible only if you enter the segment boundaries ”*******” when programming in STL.

3.5.3 Clock (Clock-Pulse Generator)

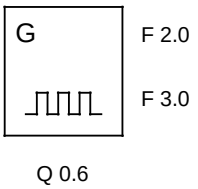
This section describes how to program a clock-pulse generator.

Example: A clock-pulse generator can be implemented using a self-clocking timer that is followed in the circuit by a binary scaler. Flag F 2.0 restarts timer T 7 each time it runs down, i.e., flag F 2.0 is ”1” for one cycle each time the timer runs down. The pulses of flag F 2.0 applied to the binary scaler result in a pulse train with pulse duty factor 1:1 at output Q 0.6. The period of this pulse train is twice as long as the time value of the self-clocking timer.

Timing Diagram



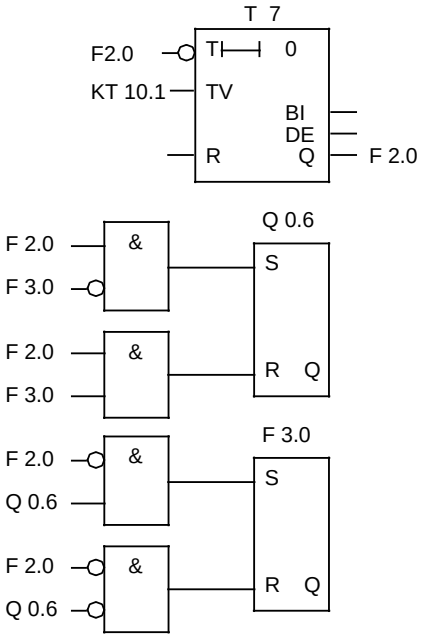
Circuit Diagram



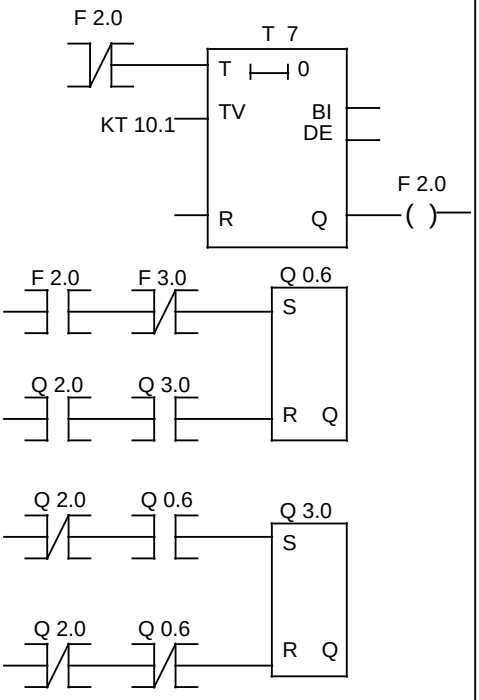
STL

```
AN    F     2.0
L     KT    10.1
SR    T     7
NOP   0
NOP   0
NOP   0
A     T     7
=     F     2.0
***
A     F     2.0
AN    F     3.0
S     Q     0.6
A     F     2.0
A     F     3.0
R     Q     0.6
NOP   0
***
AN    F     2.0
A     Q     0.6
S     F     3.0
AN    F     2.0
AN    Q     0.6
R     F     3.0
NOP   0
```

CSF



LAD



3.6 Illegal Accesses

Various access operations are illegal in the S5-115F. An attempt to access illegal addresses is rejected with an error message in the case of LIR, TIR and TNB and the PLC stops (Table 3-35).

Table 3-35. Illegal Access Operations

Range	STL operations	Remarks
0000_H to $0FFF_H$	LIR, TIR, TNB	Area for special intelligent I/Os not used with the S5-115F
$C400_H$ to $EFFF_H$	TIR, TNB	Operating system data: read only!
$F000_H$ to EFFF	LIR, TIR, TNB	I/O area, page area
$F080_H$ to $F0FF_H$ (analog input/output modules)	L PY, L PW	Use FB 250 ANEI for reading the coded values of analog input modules. If you use a CP 523, you can use L PW, L PY to access the addresses assigned by the CP 523.
$F080_H$ to $F0FF_H$ (analog input/output modules)	T PY, T PW	Use FB 251 ANAU to transfer the coded values of analog output modules. If you use a CP 523, you can use T PW, T PY to access the addresses assigned by the CP 523.

3.7 Programming Notes

In order to guarantee the safety of systems requiring official authorization, the possibility of systematic errors by the user must be precluded. Systematic errors have the same effect on both subunits.

Note

Organize your programs as clearly as possible!

The following are not permitted
- Trick programming
- Linking of Boolean logic operations to actions on different parentheses levels
  The hardware does not monitor the parenthesis stack for overflow.
- The use of the operations DI, and JUR in safety-oriented programs.

3.8 LPLZ Incrementing Sequence

Operating system

Monitoring measures are integrated.

User program

An operation sequence for incrementing the logical program counter must be inserted at the latest every 128 statements/operations in order to guarantee proper program execution in both subunits.
The program is monitored by the logical program counter (LPLZ). It is located in FW 0 and must be incremented by the user program at the latest after 127 codewords.
The following rule applies for structured programming with blocks:
- The sequence for incrementing the LPLZ is located at the beginning of each block
- Blocks longer than 127 words are segmented with ”BLD 255”
- The BLD 255 operation before the 127 word boundary is followed by the LPLZ sequence (10 bytes)
  - LFW0
  - LKF+1
  - +F
  - TFW0
- Flag word 0 must not be used by the user in any other way.

4 Program Test

4.1 Signal Status Display . . . . . 4 - 1
4.1.1 Program-Dependent Signal Status Display ”STATUS” . . . . . 4 - 2
4.1.2 Direct Signal Status Display ”STATUS VAR” . . . . . 4 - 2

4.2 Forcing . . . . . 4 - 3
4.2.1 Forcing Outputs ”FORCE” . . . . . 4 - 3
4.2.2 Forcing Variables ”FORCE VAR” . . . . . 4 - 3

4.3 Search . . . . . 4 - 4

4.4 Programmer Operator Functions . . . . . 4 - 4

Figures

4-1. Comparison of the ”STATUS” and ”STATUS VAR” Test Functions . . . . . 4 - 1
4-2. Representation of Signal States on a Screen (for LAD and CSF) . . . . . 4 - 2

Tables

4-1. Overview of the Operator Functions . . . . . 4 - 5

4 Program Test

Use the following test functions to find and eliminate logic errors in program scanning.

Note

Test blocks are not possible with the S5-115F.

4.1 Signal Status Display

Two test functions indicate operand signal states and the result of the logic operation (RLO).

Depending on when signal states are examined, a distinction is made between program-dependent signal status display (STATUS) and direct signal status display (STATUS VAR).

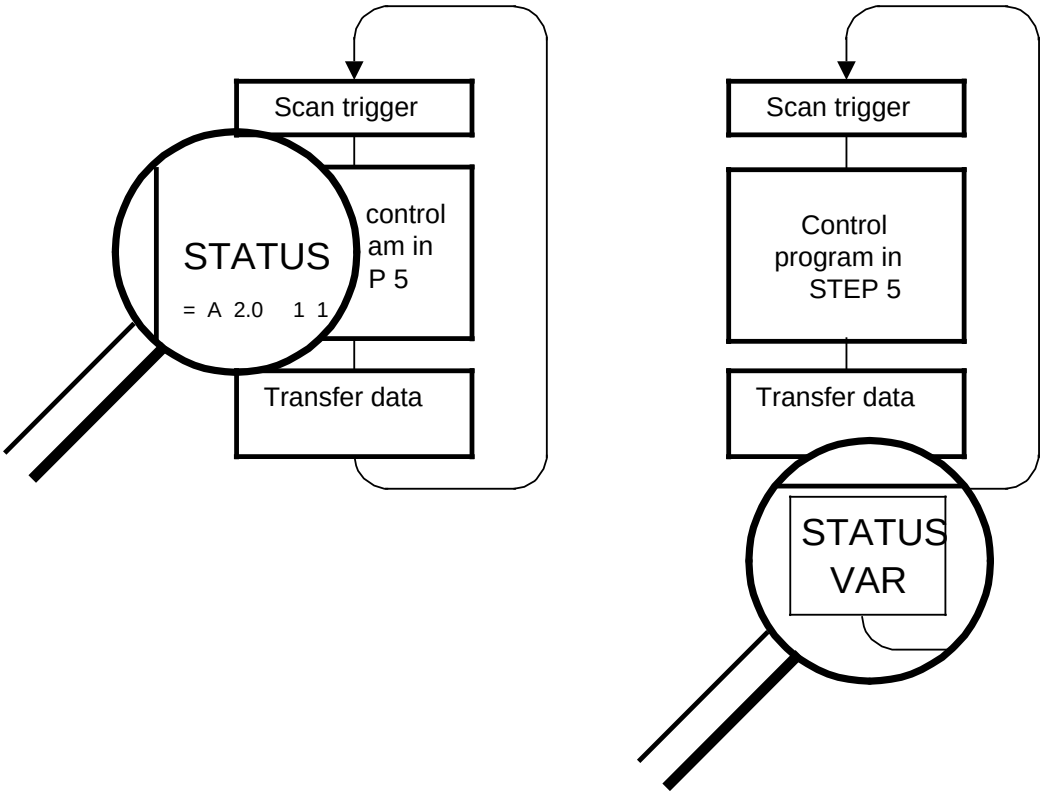


Figure 4-1. Comparison of the ”STATUS” and ”STATUS VAR” Test Functions

Displaying signal states on the screen

The display of signal states on the screen differs depending on the method of representation used:

STL:
Signal states are represented as a listing of information.

CSF/LAD:
Signal states are represented by different types of connecting lines as shown in Figure 4-2:

═══════	Signal state 1
- - - - - - - - - -	Signal state 0
───────	Signal state cannot be represented. (E.g., the signal state is outside the range of the 20 operands that can be represented.)

Figure 4-2. Representation of Signal States on a Screen (for LAD and CSF)

4.1.1 Program-Dependent Signal Status Display ”STATUS”

Use the ”STATUS” test function to display the current signal states and the RLO of the individual operands in the selected block (FB, PB, OB, SB) during program scanning. It can only be activated in the S5-115F in Test mode.
In contrast to the S5-115U, this test function in S5-115F RUN permits no modification of the selected block

4.1.2 Direct Signal Status Display ”STATUS VAR”

This test function can only be activated in the Test mode. It displays the state of a random operand (input, output, flag, data word, counter or timer) at the end of a program scan. The information is taken from the process image of the operands in question.

4.2 Forcing

You can use the Forcing functions to influence specific binary and digital operands. A distinction is made between Forcing Outputs (FORCE) and Forcing Variables (FORCE VAR), depending on whether the process image or program scanning is influenced. Both functions are possible in the S5-115F only in Test mode.

4.2.1 Forcing Outputs ”FORCE”

You can set ouputs to a specific signal state directly without using the control program. Use this direct method to check the wiring and functioning of output modules. This procedure does not change the process image but it does cancel the output disable state.

> **Note**
>
> For the ”FORCE” test function, the PLC must be in ”STOP” mode.

4.2.2 Forcing Variables ”FORCE VAR”

With the ”FORCE VAR” test function, the process image of binary and digital operands is modified regardless of the PLC mode.
The following variables can be modified: I, Q, F, T, C, and D.
Program scanning with the modified process variables is executed in the ”RUN” mode. However, the variables can be modified again in the remaining program run, without a checkback signal. Process variables are forced asynchronously to the program run.

Special characteristics include the following:

- Modify the variables I, Q, and F only by bytes or by words in the process image.
- For the variables T and C in the KM and KH format, proceed as follows:
  - Enter a ”Y” in the SYS. OPS. input field in the Defaults menu (only in the case of the PG 675).
  - Pay particular attention to forcing of edge trigger flags.
- An incorrect format or operand input interrupts the signal status display. The system outputs the message ”NO FORCE POSSIBLE”.

4.3 Search

The Search function looks for specific keys in the program and lists them on the programmer display. You can then make program changes at this point.
This function is only possible in Test mode.

You can execute Searches with the following programmer functions:
- INPUT
- DISPLAY
- STATUS

Possible search keys include the following:
- Statements (e.g. A I 0.1)
- Operands (e.g. Q 1.0)
- Labels (e.g. X 01), possible only in function blocks
- Addresses (e.g. 006)

> **Note**
>
> The Search function is executed differently by different programmers. It is described in detail in the appropriate operator's guide.

4.4 Programmer Operator Functions

Programmer operator functions are limited in the case of the S5-115F for safety reasons.
The limitations depend on the operating mode and are most restrictive in safety operation. Programmers cannot be used in RUN mode and, in STOP mode, they can only be used to read data.

One exception is data input via the parameter entry DB. In safety mode, you can change the parameter entry DB via the programmer even in PLC STOP. The parameter entry DB offers you the advantage of being able to delay definition of certain operating parameters until ”during operation”. Of course, you must check the parameters entered for safety limits with a filter program in the control program. This filter program is checked like all other control program blocks by the expert during the individual system acceptance test.

Only the parameter entry DB configured with COM 115F can be changed using the programmer.

> **Note**
>
> The configured parameter entry DB must not be located on the EPROM submodule. For this reason, load the parameter entry DB into the PLC memory using the ”Transfer” programmer function, or generate the parameter entry DB in the restart OBs (OB 21, OB 22).

Example:

Entering a pressure setpoint via the parameter control DB. Within the limits 10 to 15 bars, this value is not critical for the process. The value for the pressure setpoint is located at a definite point in the parameter entry DB. The error program scans the input and accepts only those values that are within the configured limits.

Table 4-1 shows all the programmer parameter entry functions for the S5-115F PLC with CPU 942-7UF15.

Table 4-1. Overview of Operator Functions

PG←→PLC Operator functions Bold type: Operator entry function in the S5-115F	Test mode			Safety mode		
	Large STOP loop	Lesser STOP loop	RUN	Large STOP loop	Lesser STOP loop	RUN
1. Input DB, FB, PB, OB, SB	X	-	-	**Only Param. entry DB**	-	-
2. Display (with correction) DB, FB, PB, OB, SB	X	-	Without correction	Without correction	-	-
3. Compare DB, FB, PB, OB, SB	X	-	X	X	-	-
4. Transfer to PLC DB, FB, PB, OB, SB	X	-	-	**Only Param. entry DB**	-	-
5. Transfer from PLC DB, FB, PB, OB, SB	X	-	X	X	-	-
6. Removal of blocks	X	-	-	-	-	-
7. Start PLC	X	-	-	-	-	-
8. Stop PLC	-	-	X	-	-	-
9. Directory Display	X	-	X	X	-	-
10. Memory Configuration Display	X	X	X	X	X	-
11. Display ISTACK/BSTACK	X	-	-	X	-	-
12. Display SYSPAR	X	X	X	X	X	-
13. Display ADR (memory location) with input[1]	X	Without correction	**Without correction[2]**	**Without correction**	Without correction	-
14. FORCE (outputs only)	X	-	-	-	-	-
15. FORCE VAR (I,Q, F, T, C, D)	X	-	X	-	-	-
16. STATUS (FY,PB,OB,SB) with correction	X	-	Without correction	-	-	-
17. STATUS VAR (I, Q, F, T, C, D)	X	-	X	-	-	-

1 The entry under address $EA0C_H$ with a length of two bytes only affects bit 2^7 of byte $EA0C_H$.

2 The display ”ADR” can last up to 200 ms. This can result in OB 13 not being able to properly service time interrupts of less than 200 ms.

5 Error Diagnostics

5.1 Interrupt Analysis . . . . . 5 - 1
5.1.1 ”ISTACK” Analysis . . . . . 5 - 1
5.1.2 Meaning of the ISTACK Displays . . . . . 5 - 3
5.1.3 LED Error Signalling . . . . . 5 - 5

5.2 Program Errors . . . . . 5 - 5
5.2.1 Determining an Error Address . . . . . 5 - 5
5.2.2 Program Trace with the Block Stack (”BSTACK”) Function . . . . . 5 - 7

5.3 System Parameters . . . . . 5 - 9

5.4 Error Detection and Entry in the Error DB . . . . . 5 - 10
5.4.1 Error Detection . . . . . 5 - 10
5.4.2 Error Entry in Error DBs 2 and 3 . . . . . 5 - 14
5.4.3 Ways of Evaluating the Error DBs . . . . . 5 - 16

5.5 Error Messages in Plaintext with COM 115F . . . . . 5 - 16
5.5.1 General Explanations . . . . . 5 - 16
5.5.2 Types of Error . . . . . 5 - 17

Figures

5-1. Control Bit Display (SD 5, 6, 7) . . . . . 5 - 2
5-2. Interrupt Stack Display . . . . . 5 - 2
5-3. Structured Program with Illegal Statement . . . . . 5 - 5
5-4. Addresses in the CPU Program Memory . . . . . 5 - 6
5-5. Program Trace with the ”BSTACK” Function . . . . . 5 - 7
5-6. ”BSTACK” Display on a PG 675 Programmer . . . . . 5 - 8
5-7. Error DB 2, 3: Memory Map . . . . . 5 - 14
5-8. Screen Display in COM 115F . . . . . 5 - 16

Tables

5-1. Mnemonics for Control Bits . . . . . 5 - 3
5-2. Meaning of the ISTACK Displays . . . . . 5 - 4
5-3. Meaning of the Error LEDs on the CPUs . . . . . 5 - 5
5-4. CPU System Parameters . . . . . 5 - 9
5-5. Assignment of the Error Record . . . . . 5 - 10
5-6. Error-Detecting Programs of the Operating System . . . . . 5 - 11
5-7. Operating System Central Functions . . . . . 5 - 13
5-8. Integrated Standard FBs which Generate 115F Operating System Error Messages . . . . . 5 - 13
5-9. Types of Errors which Can Be Interpreted by COM 115F . . . . . 5 - 17

5 Error Diagnostics

The S5-115F operating system detects errors which can occur in connection with any of the following:
- Hardware
- Programming (STEP 5)
- Configuring (e.g. PLC parameters)
- Data handling (e.g. wrong address setting on interface modules).

The S5-115F offers the following methods of displaying error messages:
- ISTACK (Interrupt stack)
- BSTACK (Block stack)
- COM 115F - plaintext

5.1 Interrupt Analysis

When malfunctions occur, the operating system sets various ”analysis bits” that can be scanned with the programmer using the ”ISTACK” function. LEDs on the CPU also report some malfunctions.

5.1.1 ”ISTACK” Analysis

The interrupt stack (ISTACK) is an internal memory area of the CPU where malfunction reports are stored. When a malfunction occurs, the appropriate bit is set.
Use a programmer to read this memory byte by byte.

Note

You cannot read the ISTACK when the PLC is in the ”RUN” mode.

ISTACK display on the PG 635/675/685/695 and 750 with COM 115F

The following tables show the ISTACK as it is displayed on CRT-based programmers. Set bits are marked by ”X” (here, for example, FITEKO-OK, STOANZ., etc.).
Some bit identifiers are different when using LAD, CSF and STL .

CONTROL BITS								
FISIN_PR	FIALA_TE	BSTSCH	SCHTAE	ADRBAU	TIMR_AKT	FIFEDB_S	FISTPS	SD 5 (EA0A)
FIF_BETR	FIBS_AKT	FITL_AG	FITEKO_OK X	ALARMREQ	FIEA_PER	NB	FIALA_SP	
STOZUS	STOANZ X	NEUSTA	FIURLOE	BATPUF	FIWINT	BARB	BARBEND	SD 6 (EA0C)
FINT_GEW	FIQV	FIR_BUS	FIEMPF	FISEND	ABFS_AKT X	FIWEBI_2	FIKSTOP	
ASPNEP	ASPNRA X	KOPFNI	FIDE_TE	ASPNEEP	FIOBT_RQ	FIDE_M_T	FIOBT_FE	SD 7 (EA0E)
KEINAS	SYNFEH	NINEU	NB	NB	NB	NB	URLAD	

Figure 5.1. Control Bit Display (SD 5, 6, 7)

INTERRUPT STACK												
DEPTH: 1												
OP REG: 4080 BLK STP: EAFF		SAZ: 0000 NO.: REL SAZ:		DB ADR: 0000 DB NO.:								
ACCUM1: FFF1		ACCUM2: 00FF										
BRACKETS:	KE1	000	KE2	000	KE3	000	KE4	000	KE5	000	KE6	000
RESULT DISPLAY:	ANZ1 (CC1)		ANZ0 (CC0)		OVFL		CARRY		ODER	STATUS	VKE (RLO)	ERAB
CAUSE OF FAULT:	STOPS X			SUF		TRAF		NNN		STS	STUEB	FIF_ FEHLER
	NAU		QVZ			ZYK		SYSFE		PEU	BAU	ASPFA

UAW

Figure 5-2. Interrupt Stack Display

5.1.2 Meaning of the ISTACK Displays

Besides malfunction analysis, the ISTACK provides other information.

Table 5-1. Mnemonics for Control Bits

Control Bit Mnemonics		Interrupt Display Mnemonics	
FISIN_PR	SINEC L1 LAN has been configured	UAW	Interrupt condition code word
FIALA_TE	Interrupt DI test active	STOPS	Mode selector at STOP
BSTSCH	Block shift requested	SUF	Substitution error
SCHTAE	Block shift active (function: COMP: PLC)	TRAF	Transfer error for data block statements: data word number > data block length.
ADRBAU	Construction of address lists	NNN	Statement cannot be interpreted in the S5-115F (e.g. a 150S statement).
TIMR_AKT	ISR[1] timer(10 msec.) running	STS	Operation interrupted by a programmer STOP request or programmed STOP statements.
FIFEDB_S	Send error DB on SINEC L1 LAN	STUEB	Block stack overflow: The maximum block call nesting depth of 16 has been exceeded.
FISTPS	STOP-RUN switch pulse has been sent	FIF_FEHLER	Error in S5-115F with PLC Stop response
FIFBETR	Safety mode	NAU	Power failure
FIBS_AKT	115 operating system executing, not user program.	QVZ	Time-out from I/Os: A nonexistent module has been referenced.
FITL_AG	=0 >>> subunit A =1 >>> subunit B	ZYK	Scan time exceeded: The set maximum permissible program scan time of 0.5 sec. has been exceeded.
FITEKO_OK	Test component check completed, start up next one	SYSFE	Error in the SYSID block
ALARMREQ	Interrupt request after interrupt synchronization, for idle loop.	PEU	I/Os not ready: power failure in the I/O expansion unit; connection to the I/O expansion unit interrupted No terminator in the central controller
FIEA_PER	Memory address in I/O range: Interpreter	BAU	Battery failure
FIALA_SP	Interrupt disable active	ASPFA	Illegal memory submodule
STOZUS	"STOP" mode (external request)	ANZ1/ANZ0	00: ACCUM1 = 0 or 0 shifted 01: ACCUM1> 0 or 1 shifted 10: ACCUM1< 0
STOANZ	"STOP" display (internal request)	OVF	Arithmetic overflow (+or -)
NEUSTA	PLC in Cold Restart	CARRY	Overflow bit
FIURLOE	Overall reset: PG request	ODER	OR memory (set by "0" operation)
BATPUF	Battery backup okay	STATUS	STATUS of the operand of the last binary statement executed
FIWINT	Interrupt change flag, lesser stop loop	VKE	Result of logic operation
BARB	Program check	ERAB	First scan
BARBEND	Request for end of program check	BEF_REG	Operation register
FINT_GEW	Interrupt past	SAZ	Step address counter
FIQV	SINEC L1 internode traffic	DB_ADR	Data block address
FIR_BUS	Redundant SINEC L1 LAN	BST_STP	Block stack pointer
FIEMPF	Receive permission SINEC L1	DB_NR	Data block number
FISEND	Send permission SINEC L1	REL_SAZ	Relative step address counter
ABFS_AKT	Interrupt enable		
FIWEBI_2	Change bit for job 2		
FIKSTOP	Lesser Stop loop		
ASPNEP	Memory submodule is an EPROM		
ASPNRA	Memory submodule is a RAM		
KOPFNI	Block header cannot be interpreted		
FIDE_TE	DI test active		
ASPNEEP	Memory submodule is an EEPROM		
FIOBT_RQ	Time interrupt request from 10 msec. ISR[1]		
FIDE_M_T	DI multitest		
FIOBT_FE	OB 13 error		
KEINAS	No memory submodule		
SYNFEH	Synchronization error (blocks are incorrect)		
NINEU	Cold Restart not possible		
URLAD	Bootstrapping required		

[1] ISR = Interrupt service routine

Use Table 5-2 to determine the cause of a fault or an error when program scanning is interrupted. In each case, the CPU goes into the ”STOP” mode.

Table 5-2. Meaning of the ISTACK Displays

Fault/Error	Fault/Error ID	Cause	Remedy
Cold restart is not possible	NINEU SYNFEH/ KOPFNI	Faulty block: - System startup fault. - Compressing has been interrupted by a power failure. - Block transfer between programmer and PLC was interrupted by a power failure. - Program error (TIR/TNB/DO FW).	Perform an Overall Reset. Reload the program.
Faulty submodule	ASPFA	The submodule ID is illegal: - S5-110S/S5-135U/S5-150U submodule	Plug in the correct submodule.
Battery fault	BAU	There is no battery or the battery is low.	Replace the battery. Perform an Overall Reset. Reload the program.
I/Os not ready	PEU	The I/Os are not ready: - There has been a power failure in the expansion unit. - The connection to the expansion unit has been interrupted. - There is no terminator in the central controller.	- Check the power supply in the expansion unit. - Check the connection. - Install a terminator in the central controller.
Program scanning interrupted	STOPS	The mode selector is on STOP.	Put the mode selector on RUN.
	SUF	Substitution error: A function block was called with an incorrect actual parameter.	Correct the function block call.
	TRAF	Transfer error: - A data block statement has been programmed with data word number greater than the data block length. - A data block statement has been programmed without opening a DB first.	Correct the program error.
	NNN	- A statement cannot be decoded. - The nesting level has been exceeded. - A parameter has been exceeded.	Correct the program error.
	STS	- Software stop by statement (STP) - STOP request from programmer - STOP request from SINEC L1 master	
	STUEB	Block stack overflow: - The maximum block call nesting depth of 16 has been exceeded.	Correct the program error.
	FIF-FEHLER	Multiple message S5-115F. Error with ”STOP” response.	Error analysis with DB 2,3 with COM 115F (5.5)
	NAU	There has been a power failure.	
	QVZ	Time-out from I/Os: - A peripheral byte that was not addressed has been referenced in the program. - An I/O module does not acknowledge.	Correct the program error or replace the I/O module.
	ZYK	Scan time exceeded: - The program scanning time is greater than the set monitoring time.	Check the program for continuous loops. If necessary, change the monitoring time.
	SYSFE	Error in SYSID block	Check program

5.1.3 LED Error Signalling

Certain errors are indicated by LEDs on the CPU. Table 5-3 explains these error signals.

Table 5-3. Meaning of the Error LEDs on the CPUs

LED	Meaning
QVZ lights up	Time out (The CPU has entered the ”STOP” mode.)
ZYK lights up	Scan time exceeded (The CPU has entered the ”STOP” mode.)
BASP	Digital outputs are disabled. (The CPU is in the ”RESTART” or the ”STOP” mode.)

5.2 Program Errors

5.2.1 Determining an Error Address

The STEP address counter (SAZ) in the ISTACK (bytes 25 and 26) indicates the absolute memory address of the STEP 5 statement in the PLC **before** which the CPU has entered the ”STOP” mode.

You can use the ”DIR PLC” programmer function to determine the appropriate block starting address.

Example:

Figure 5-3 shows a program consisting of OB 1, PB 0, and PB 7. PB 7 contains an illegal statement.

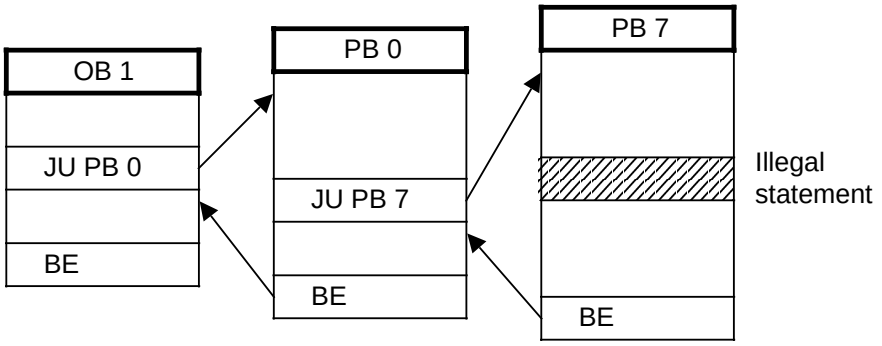


Figure 5-3. Structured Program with Illegal Statement

When the CPU reaches the illegal statement, it interrupts program scanning and goes into the ”STOP” mode with the ”NNN” error message. The STEP address counter is at the absolute address of the next statement in the program memory that still has not been processed.

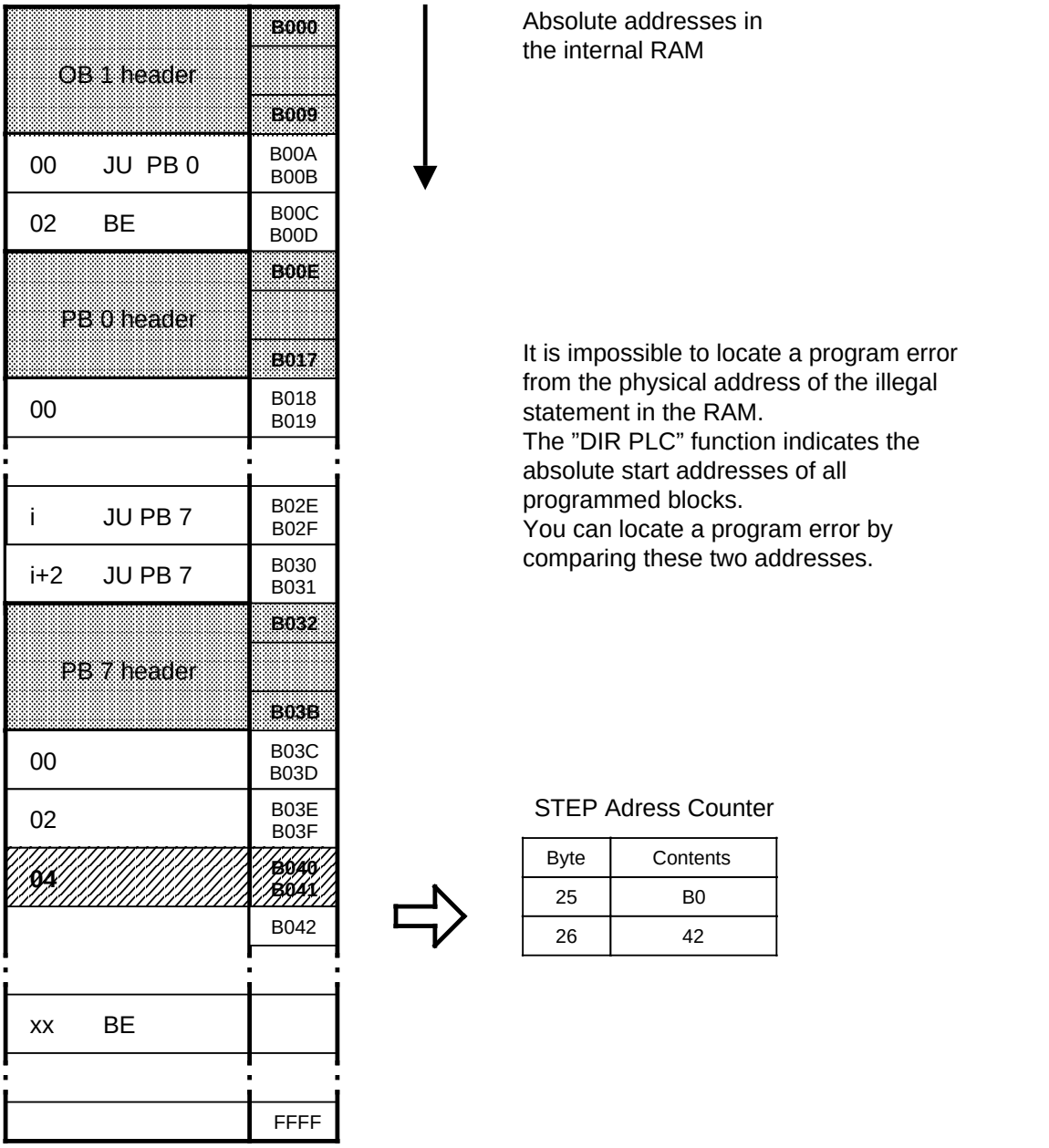


Figure 5-4. Addresses in the CPU Program Memory

Display of Statements Containing Errors

The PG ”SEARCH” function is used to find specific program locations (4.3). You can use this function to find the relative error address.

5.2.2 Program Trace with the Block Stack ("BSTACK") Function

During program scanning, jump operations enter the following information in the block stack:

- The data block that was valid before program scanning exited a block
- The relative return address. This address indicates the location at which program scanning continues after it returns from the block that was called.
- The absolute return address. This address indicates the location in the program memory at which program scanning continues after it returns from the block that was called.

You can call the information listed above using the "BSTACK" programmer function in the "STOP" mode if the CPU has entered this mode as the result of a malfunction. The "BSTACK" reports the status of the block stack at the time the interruption occurred.

Example: Program scanning was interrupted at function block FB 2. The CPU went into the "STOP" mode with the error message "TRAF" because of incorrect access. (DB 5 is two words long. DB 10 is twelve words long.)
You can use the "BSTACK" function to determine the path used to reach FB 2 and to determine which block has passed the wrong parameter. The "BSTACK" contains the three return addresses.

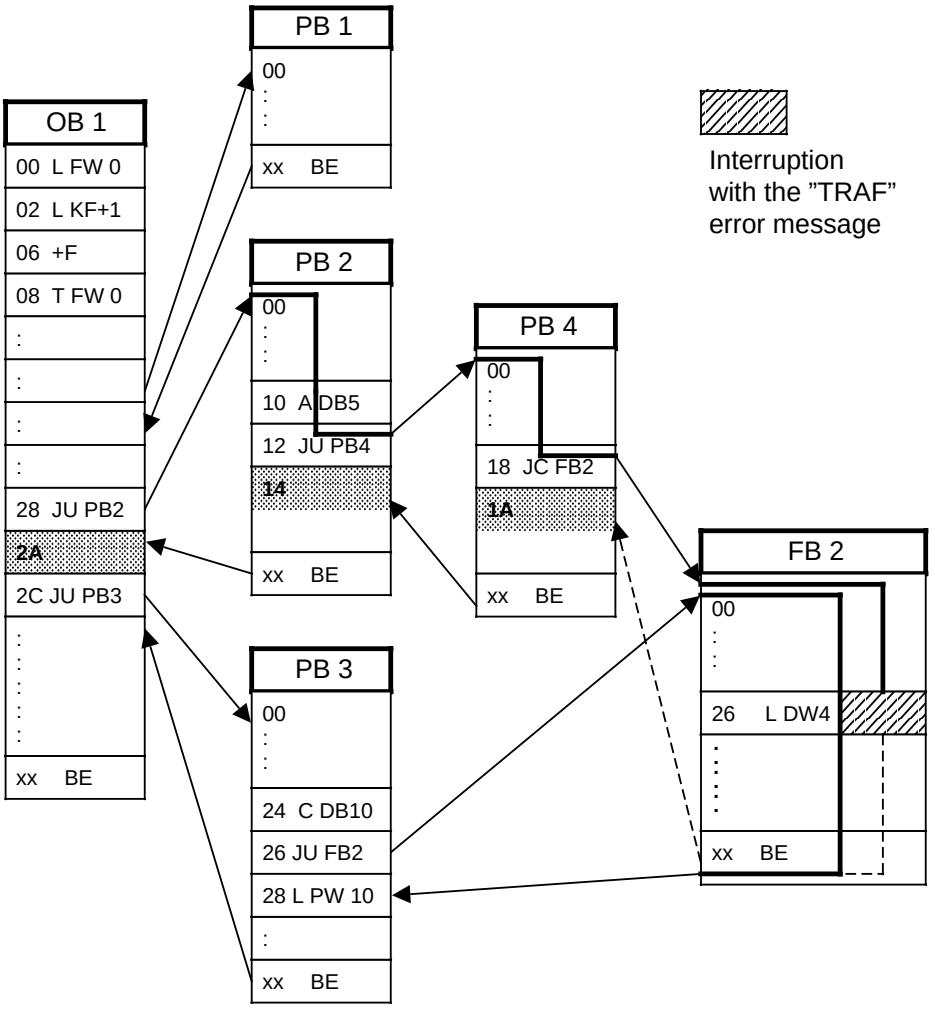


Figure 5-5. Program Trace with the "BSTACK" Function

Display

BLOCK STACK

BLOCK NO.	BLOCK ADDR.	RETURN ADDR.	REL. ADDR.	DB NO.	DB ADDR.
PB 4	B19A	B1B4	001A	5	B214
PB 2	B168	B17C	0014	5	B214
OB 1	B1CE	B1F8	002A	5	B214
		06FS			

Figure 5-6. ”BSTACK” Display on a PG 675 Programmer

The display in Figure 5-6 indicates that DB 5 was accessed incorrectly on the path OB 1 PB 2 PB 4.

5.3 System Parameters

Use the ”SYSPAR” programmer function to read the system parameters out of the CPU.

Table 5-4. CPU System Parameters

System Parameter			Contents	Explanation
1	Baud rate		9600	Absolute start addresses in the CPU memory
2	Signal states of the inputs		F000	
3	Signal states of the outputs		F000	
4	Process image of the inputs		EF00	
5	Process image of the outputs		EF80	
6	Flag area		EE00	
7	Timer area		EC00	
8	Counter area		ED00	
9	SD range		EA00	
10	PLC software status		9F03	
11	Program memory		D800	End addresses
12	System memory		0000	
13	Bytes DB list		0200	Indication of length in bytes
14	Bytes SB list		0200	
15	Bytes PB list		0200	
16	Bytes FB list		0200	
17	Bytes OB list		0200	
18	Bytes FX list		0000	
19	Bytes DX list		0000	
20	Length of DB0		0A00	
21	2nd CPU ID		EF04	
22	Block header length		000A	Length specification in bytes
23	CPU ID	PLC software release	U000	

5.4 Error Detection and Entry in the Error DB

The S5-115F has user-friendly methods of error detection and error handling.

5.4.1 Error Detection

The BS 115F exposes errors in all program sections caused by configuring, programming, handling and hardware defects. In addition, it contains extensive test routines for hardware components. These tests are carried out during a restart and during the PLC cycle, with the restart test running independently. The test during the PLC cycle is divided into time slices lasting between 10 msec. and, in extreme cases, 140 msec. The operating system organizes the test in such a way that the whole test is completed after expiry of the configured test cycle time. This test cycle time must be configured smaller than or equal to the second error entry time (specified by the licensing authority). The longer the test cycle time, the shorter will be the average load put on the PLC cycle by the self-test.
After they have been detected, the errors are stored in a data field (record) of eight bytes.

Table 5-5. Assignment of the Error Record

Byte No.		List in ...
0	Error group, Type of error	5.5.2, Appendix B
1	Number of the program section which detected the error	Table 5-6., Appendix B
2	Current number within the program section in byte 1	Appendix B
3 to 7	Additional information	

Meaning of addresses in the additional information

You will find the following locations specified in the additional information:
- Subunit
- I/O word number
- I/O byte number
- Bit number

The entries for subunit, I/O word number and I/O byte number indicate the actual error location. This is also the case if two alternative locations are specified, such as in comparison error 16.01:

”DIs in A and B not identical, discrepancy time expired”

Either the specified I/O address in subunit A or the same address in subunit B can be the error locations. In contrast, the bit number does not always indicate the actual I/O error location. For example, in the case of comparison errors, only the number of the erroneous bit found first is displayed and any further erroneous bits are not specified.

The following operating system programs detect errors:

Table 5-6. Error-Detecting Programs of the Operating System

Byte 1*	Function of the Program Section
2	Updating of user locations T0 to T127 and of ”long discrepancy times” (times for 256 digital inputs)
4	Synchronization of user servicing of process (OB2) and time (OB13) interrupts
14	Coordination of the modes in both subunits
16	Comparison of the digital input images (PII) of both subunits and analysis of non-identical signals
17	Read and exchange analog inputs
18	Comparison of the output images (PIQ) of both subunits
23	Synchronization of SINEC L1 interfaces
25	Copying of the RAM from subunit A to subunit B, since subunit B does not permit PG operator input
26	RAM comparison of both subunits
27	Bit test of the parallel link
28	Test of the non-intermittent digital I/Os
29	STEP address counter (SAZ) test
30	Test manager; supports time slice tests of the various hardware components
31	Logic gating of error DBs 2 and 3 for subunit A and B
32	Edge-triggered DI matrix test
33	Test of the non-intermittent analog inputs

***** Byte 1 in error record

Table 5-6. Error-Detecting Programs of the Operating System (continued)

Byte 1*	Function of the Program Section
45	Bit test of the external µP8032 RAM
46	µP8032 function test
47	Test of the internal µP8032 RAM
48	SSP function test
49	Operating system EPROM test (64 Kbytes)
50	User EPROM test (8, 16, 32 Kbytes depending on control program)
51	Assigning a page of the parallel interface
53	Job header, common to all parallel interface jobs
54	Central error routine for evaluating and storing an error and activating the error response
55	Self-test; organizes, together with the test manager, the ”high-value component test” to be processed in time slices
56	Short test; organizes a ”high-value” test before the S5-115F enters the RUN mode. This test is not executed in time slices.
58...63	Auxiliary programs with access to pages of the parallel interface
74	Assigning information about number and type of connected modules to data fields; comparison with configuration
99	Synchronization of the subunits

***** Byte 1 in error record

Table 5-7. Operating System Central Functions

Byte 1*	Function of the Program Section
100	Cold restart after POWER UP
101	Stop program
102	Restart from Stop
103	Cycle control
104	Idle loop
105	10 msec. interrupt service routine
106	PG interrupt service routine
107	SINEC L1 interrupt service routine
108	Process interrupt service routine
109	Interpreter of the STEP 5 operations not running in the SSP

* Byte 1 in error record

Table 5-8. Integrated Standard FBs which Generate 115F Operating System Error Messages

Byte 1*	Function of the Program Section
250	FB250, ANEI for analog input module
251	FB251, ANAU for analog output module
252	FB 252, BLUE for block transfer CP 523
253	FB253, MBXT for mailbox transfer
254	FB254, SYNC for synchronization of user time update, process/time interrupt handling and PG/SINEC L1 processing

* Byte 1 in error record

5.4.2 Error Entry in Error DBs 2 and 3

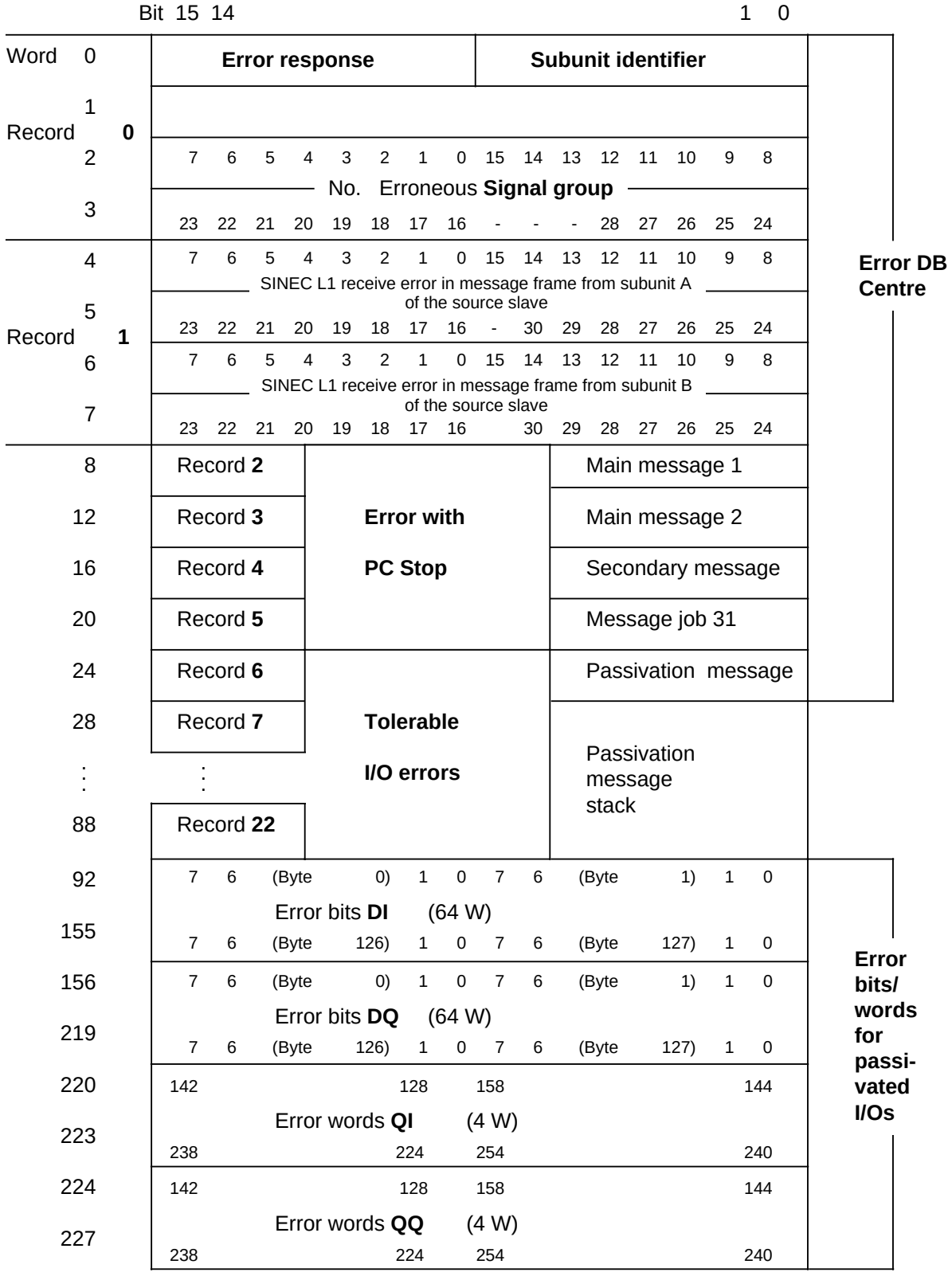
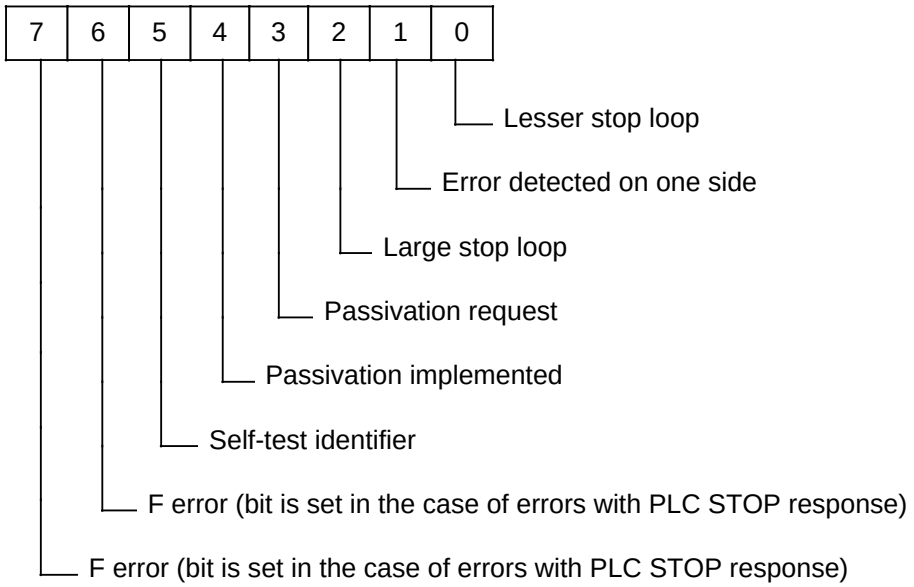


Figure 5-7. Error DB 2, 3: Memory Map

Any errors found are evaluated by the central error routine and entered in error DBs 2 and 3. Subunit A writes its error into error DB 2 and subunit B writes its error into error DB 3. If the error does not cause the link between the two subunits to be interrupted, the subunits are synchronized by exchanging the error DBs. Figure 5-7 describes the structure of the error DBs.

Meaning of the errors in the error DB:

Error response; meaning of the set bit:



Subunit identifier
ASCII characters A or B

Signal group number:
A bit is assigned to each signal group.

If there has been a fault on a SINEC L1 data path, the number of the source slave is stored here.

Error buffer
The error buffer contains error messages in eight-byte records, just as they are output by the operating system. The first error message with stop response reaches record 2 of the error stack.
Errors with stop response can attract other messages as well as the main message (records 2 to 5). After passivation of a signal group, the S5-115F remains in RUN mode. These messages are stored in the passivation message stack. Record six contains the message of the last passivation error to occur.

Marking field for passivated I/O modules
The user can see from this which I/Os have been passivated:
Bit=1: The relevant I/O word or the relevant I/O channel
- has caused passivation and
- is passivated.

COM 115F constructs the error messages using the information stored here (Appendix B).

5.4.3 Ways of Evaluating the Error DBs

There are four ways of evaluating error DBs:
- Direct readout of the error DB as a record with the programmer or the operator panel
- Interpretation and plaintext message of Blocks 2 to 22 with COM 115F
- SINEC L1 LAN
- The error DB centre is sent to the higher-level operator PLC containing the SINEC L1 LAN master: evaluation of the DBs at the user level.
- Via programmer bus

5.5 Error Messages in Plaintext with COM 115F

With the S5-115F it is possible to display error messages in plaintext on the monitor of your programmer using the COM 115F programming package.

5.5.1 General Explanations

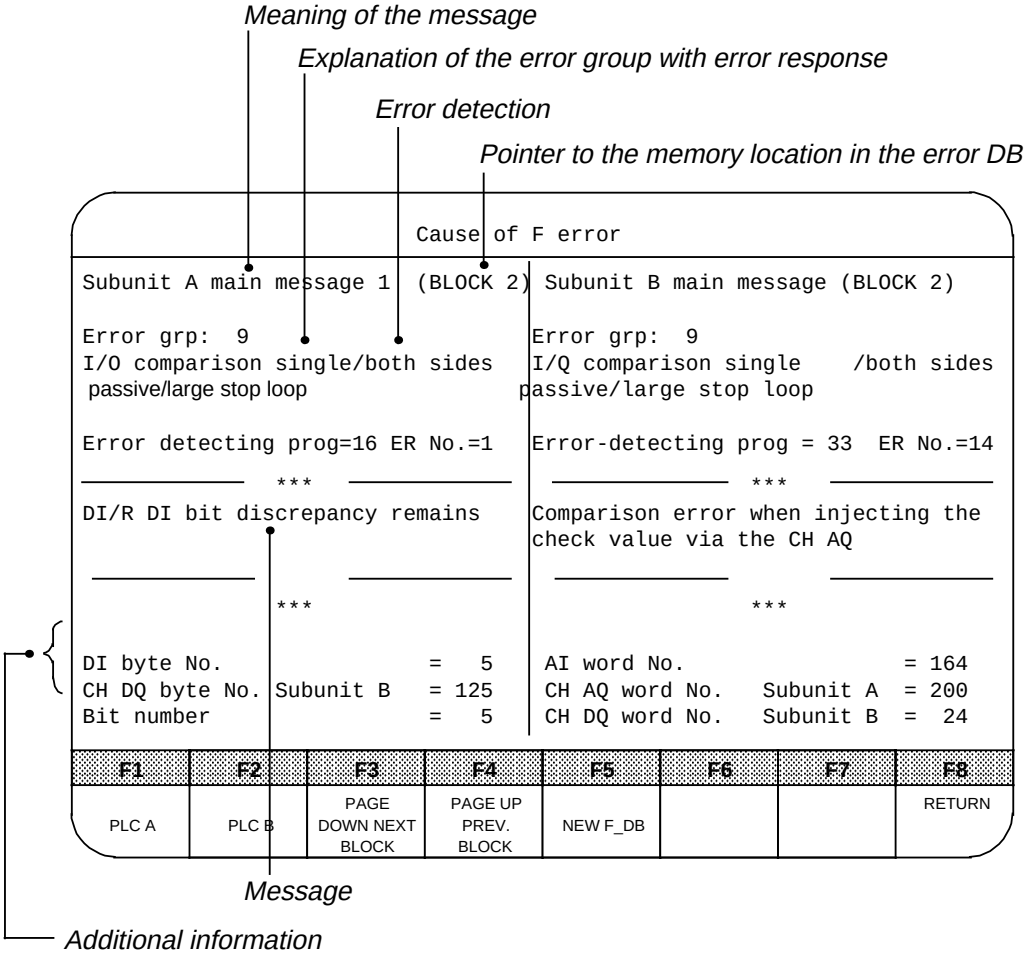


Figure 5-8. Screen Display in COM 115F

COM 115F evaluates an 8-byte error record

Byte 0	:	Error group or type of error Displayed in plaintext (Appendix B)	Byte 2	:	Consecutive numbering within the job)
Byte 1	:	Program or job No.	Bytes 3 to 7	:	Additional information

5.5.2 Types of Error

Table 5-9 is an overview of all types of error which can be interpreted by COM 115F and the relevant responses of the operating system:

Table 5-9. Types of Errors which Can Be Interpreted by COM 115F

Value Byte 0	Type of Error	Response in Safety Mode *
1	CPU, PL, PS	Lesser stop loop
2	Processor µP 8032	Lesser stop loop
3	Standard STEP 5 Processor SSP	Large stop loop
4	Bit test PL	Lesser stop loop
5	BS EPROM	Lesser stop loop
6	User EPROM	Large stop loop
7	Bit test RAM	Lesser stop loop
8	I/O comparison or test: error accumulation **	Lesser stop loop
9	I/O comparison: individual error	Large stop loop or passivation
10	I/O test: individual error	Large stop loop or passivation
11	SINEC error	Delete Receive mailbox
12	Programming	Large stop loop
13	BS 115F	Lesser stop loop
14	Configuration	Large stop loop
15	RAM comparison	Lesser stop loop

* Vol. 1, 4.1.4 of the Manual

Table 5-9. Types of Error which Can Be Interpreted by COM 115F (Continued)

Value Byte 0	Type of Error	Response in Safety Mode *
16	Handling	Large stop loop
17	Handling	Large stop loop
18	Network interrupt	Large stop loop
19	I/O modules	Large stop loop
20	CPU, PL, PS	Lesser stop loop
21	Scan time exceeded	Lesser stop loop
22	Configuration	Large stop loop
23	Error in other subunits or CPU error	Partner response Lesser stop loop
24	Handling	Lesser stop loop
25	Battery fault	Large stop loop
26	STS operation in user program (immediate stop)	Large stop loop
29	Cannot be evaluated	Lesser stop loop
30	Handling	Lesser stop loop
31	Handling	Wait for programmer input

* Vol. 1, 4.1.4 of the Manual

6 Blocks

6.1 Integral Function Blocks . . . . . 6 - 2
6.1.1 Conversion Blocks . . . . . 6 - 2
6.1.2 Arithmetic Blocks . . . . . 6 - 3
6.1.3 Analog Value Matching Blocks . . . . . 6 - 4
6.1.4 The FB 252 Block Transfer Block . . . . . 6 - 9
6.1.5 The FB 253 Mailbox Transfer Block . . . . . 6 - 10
6.1.6 The FB 254 Synchronization Block . . . . . 6 - 10
6.1.7 Depassivation Block FB 255 . . . . . 6 - 14

6.2 OB 251 PID Algorithm . . . . . 6 - 15

6.3 Loadable Function Blocks . . . . . 6 - 25

Figures

6-1. Schematic Illustrating the Scaling of Analog Values . . . . . 6 - 5
6-2. FB 254 Calls (Ideal Schematic) . . . . . 6 - 12
6-3. FB 254 Calls (Final Schematic) . . . . . 6 - 13
6-4. Block Diagram of the PID Controller . . . . . 6 - 15
6-5. Process Schematic . . . . . 6 - 20

Tables

6-1. Overview of Integral Blocks . . . . . 6 - 1
6-2. Meaning of the Control Bits in Control Word STEU . . . . . 6 - 16
6-3. Structure of the Data Transfer Block . . . . . 6 - 18

6 Blocks

You can use the following to support the user program in the safety-related S5-115F:
- Integral blocks
- Loadable standard function blocks
- Function blocks generated by the user (subject to acceptance test)

Some standard function and organization blocks are integrated in the operating system of the central processing units. These integral blocks are programmed in machine code and therefore execute at high speed. They do not occupy user memory. Integral blocks are called in the same way as other blocks in the user program and can usually be interrupted by process interrupts. See 6.1 for further information on integral function blocks.
Loadable standard function blocks are supplied on diskette and are loaded into the user memory of the CPU. 6.3 describes the use of loadable function blocks.

Table 6-1. Overview of Integral Blocks

Block			Call Length in Words	Execution Time in msec.	Function
Type	No.	Name			
FB	240	COD: B4	5	< 0.8	Converts BCD-coded nos. (4-tetrads)
FB	241	COD: 16	6	< 1.3	Converts 16-bit fixed-point numbers
FB	242	MUL: 16	7	< 1.1	Multiplies 16-bit binary numbers
FB	243	DIV : 16	10	< 2.6	Divides 16-bit binary numbers
FB	250	AGF: ANEI	17	5*	Reads in analog value
FB	251	AGF: ANAU	9	6	Outputs analog value
FB	252	AGF: BLUE	8	Send Block 1: 1.4 32 blocks: 10 Receive 1 block: 2 32 blocks: 20	Frame transfer to and from the CP 523
FB	253	AGF: MBXT	3	< 0.8	Mailbox transfer for two-channel SINEC L1 LAN
FB	254	AGF: SYNC	3	(2 to 7 msec.)+ 40 µsec. x Y **	Updating of user timers
				2 msec. + OB 2- / OB 13 timer	Interrupt synchronization
				3	SINEC L1 synchronization
				2	Programmer synchronization
FB	255	AGF: DEPA	5	3 to 15***	Depassivation of I/O modules
OB	251			1.7	PID control algorithm

***** This value is sometimes exceeded by the discrepancy time for analog values

****** Y = Number of user timers.

******* Plus processing time of the test slice (10.4)

6.1 Integral Function Blocks

Integral function blocks can be divided into separate groups according to function.

6.1.1 Conversion Blocks

Use blocks FB 240 and FB 241 to convert numbers in BCD code to fixed-point binary numbers and vice versa.

Code Converter: B4 -FB 240-

Use function block FB 240 to convert a number in BCD code (four tetrads) with sign to a fixed-point binary number (16 bits).
A two-tetrad number must be changed to a four-tetrad number before conversion by padding it with ”0 bits”.

Call and Parameter Assignment

Parameter	Type	Data Type	Assignment	Meaning	STL
BCD	I	W	-9999 to+9999	BCD number	: JU FB 240 Name : COD : B4 BCD : SBCD : DUAL :
SBCD	I	Bi	”1” for ”-” ”0” for ”+”	Sign of the BCD number	
BINARY	Q	W	16 bits ”0” or ”1”	Binary number	

Code Converter: 16 -FB 241-

Use function block FB 241 to convert a fixed-point binary number (16 bits) to a number in BCD code with additional consideration of the sign. An eight-bit binary number must be changed to a 16-bit word before conversion.

Call and Parameter Assignment

Parameter	Type	Data Type	Assignment	Meaning	STL
DUAL	I	W	-32768 to+32767	Binary number	: JU FB 241 Name : COD : 16 DUAL : SBCD : BCD2 : BCD1 :
SBCD	Q	Bi	”1” for ”-” ”0” for ”+”	Sign of the BCD number	
BCD2	Q	By	2 tetrads	BCD number tedrads 4 and 5	
BCD1	Q	W	4 tetrads	BCD number tedrads 0 to 3	

6.1.2 Arithmetic Blocks

Use function blocks FB 242 and FB 243 to multiply and divide.

Multiplier: 16 -FB 242-

Use function block FB 242 to multiply one fixed-point binary number (16 bits) by another. The product is represented by two fixed-point binary numbers (16 bits). The result is also scanned for zero. An eight-bit number must be changed to a 16-bit word before multiplication.

Call and Parameter Assignment

Parameter	Type	Data Type	Assignment	Meaning	STL
Z1	I	W	-32768 to +32767	Multiplier	: JU FB 242 Name : MUL : 16 Z1 : Z2 : Z3=0 : Z32 : Z31 :
Z2	I	W	-32768 to +32767	Multiplicand	
Z3=0	Q	Bi	”1” if the product is zero	Scan for zero	
Z32	Q	W	16 bits	Product high word	
Z31	Q	W	16 bits	Product low word	

Divider:16 -FB243-

Use function block FB 243 to divide one fixed-point binary number (16 bits) by another. The result (quotient and remainder) is represented by two fixed-point binary numbers (16 bits each). The divisor and the result are also scanned for zero. An eight-bit number must be changed to a 16-bit word before division.

Call and parameter Assignment

Parameter	Type	Data Type	Assignment	Meaning	STL
Z1	I	W	-32768 to+32767	Dividend	: JU FB 243
Z2	I	W	-32768 to+32767	Divisor	Name : DIV : 16
OV	Q	Bi	”1” if overflow	Overflow display	Z1 :
					Z2 :
FEH	Q	Bi	”1” for division by zero		OV :
					FEH :
Z3=0	Q	Bi	”1”: quotient is zero	Scan for zero	Z3=0 :
					Z4=0 :
Z4=0	Q	Bi	”1”: remainder is zero	Scan for zero	Z3 :
					Z4 :
Z3	Q	W	16 bits	Quotient	
Z4	Q	W	16 bits	Remainder	

6.1.3 Analog Value Matching Blocks

These blocks match the nominal range of an analog module to a standard range that you can specify.

Reading in and scaling an analog value - FB 250 -

Analog input modules convert analog process signals into digital values and store them in the module. From there they are transferred cyclically to the CPU memory and also transferred to the other subunit.
The FB 250 ANEI function block enables two types of access to the analog value:

- Access to the cyclically updated CPU memory
  or
- Direct access to the analog input module memory

The type of access is defined when setting the FB 250 ANEI parameters. Figure 6-1 illustrates further processing.

Selective Sampling

The 460 analog input module permits sampling of analog values

- cyclically
- selectively

In the case of cyclical sampling, the analog variables on the module are coded and catched one after another. The user program has no influence on coding.

In the case of selective sampling, the user program initiates coding of the desired analog variables. Coding begins when the module is referenced.

If you want selective sampling, proceed as follows:

Set the function selector switch on the 460 AI to selective sampling.
Call FB 250 ANEI with Parameter EINZ = ”1”.
During coding, FB 250 ANEI will set the output parameters T Bit = ”1”. The result is available when T Bit = ”0”.
If you want to scan the T Bit parameter, call FB 250 ANEI with EINZ = ”1”.
In the case of programmable controllers with the CPU 942-7UF13:
You cannot activate selective sampling on the relevant AI module as long as T BIT = 1.

Note

Assign different bits to the activity bit when different channels are called. You will then be able to recognize which channel has been activated for coding.

FB 250 transmits a value XA within a (scaled) range defined by the user. The user defines the desired range with the ”Upper limit” (OGR) and ”Lower limit” (UGR) parameters.

Scaling schematic:

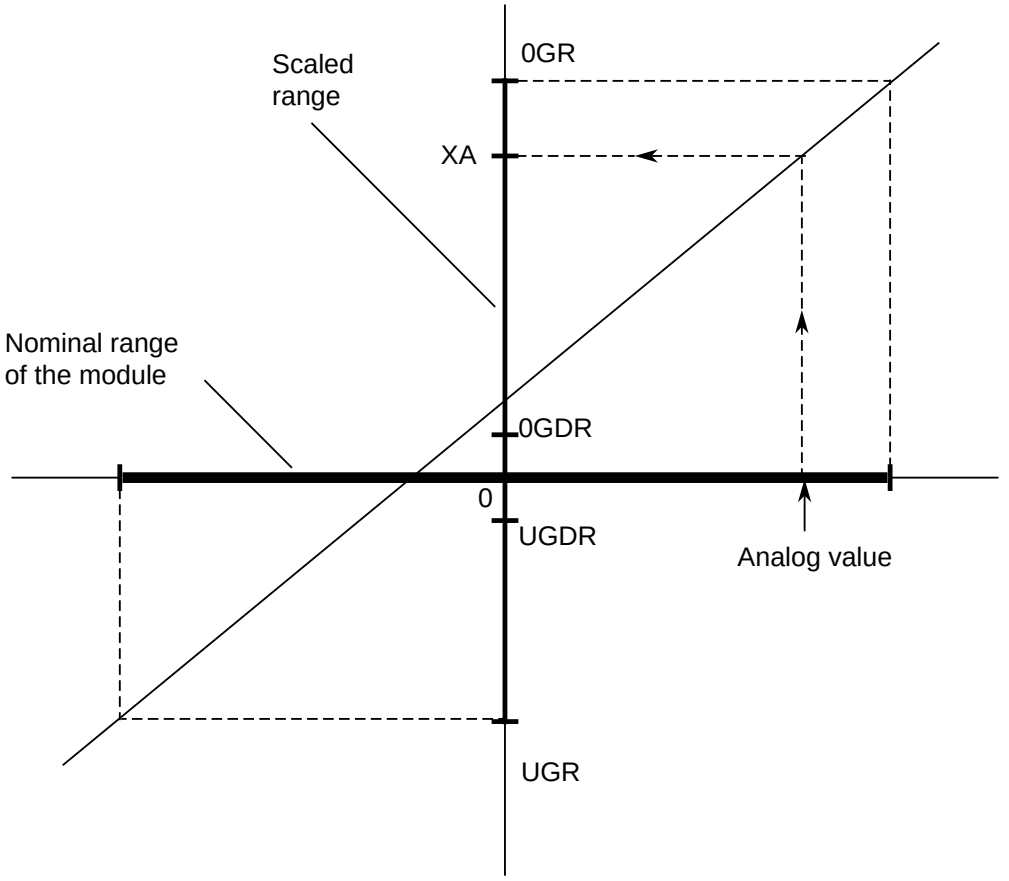


Figure 6-1. Schematic Illustrating the Scaling of Analog Values

COM 115F is used to configure the analog value representation of the module (channel type).
There are three methods of wire-break detection:

Hardware wire-break detection
This is implemented on the 460 AI module with the 498-1AA11 range card. The condition code bit F is set in the event of a wire-break and evaluated by FB 250 ANEI.

Wire-break detection by FB 250 ANEI
The 460 AI module with range cards 498-1AA51 or 498-1AA71 and the 463 AI module with the jumper setting for the range 4 to 20 mA, use the life-zero procedure for wire-break detection:
If the sensors, the signal path and the input module are intact, the smallest permissible value is 4 mA. All values less than 3 mA are interpreted by FB 250 ANEI as wire-break.

Wire-break detection by the user program and FB 250 ANEI
All measuring ranges of both AI modules which do not permit wire-break detection as in and must use the following procedure supported by the user program.
For this purpose, a wire-break range in the standard value range must be set with the ”ODGR” and ”UDGR” parameters of FB 250 ANEI.
The error bit ”F” is set in FB 250 if

- the analog value is within the ”ODGR” and ”UDGR” limits
  and
- the ”DRAK” bit is set in the control program (6.9).

Testing the line for wire-break only makes sense with this procedure if the analog value is outside the wire-break limits.

Call and Parameter Assignment:

Parameter	Meaning	Type	Data Type	Assignment / Explanation	STL
BG	Module address	D	KF	AI 460: 128 to 240 (16-byte grid) AI 463: 128 to 248 (8-byte grid)	: JU FB 250 NAME : AGF : ANEI BG : KN : OGR : UGR : DRAK : ODGR : UDGR : PASS : EINZ : DIR : XA : FB : BU : TBIT : TKON :
KN	Channel number	D	KF	KY = 0 to 7 in AI 460 0 to 3 in AI 463	
OGR	Upper limit of the output value	D	KF	-32768 to +32767	
UGR	Lower limit of the output value	D	KF	-32768 to +32767	
DRAK	User-controlled wire-break monitor active	I	BI	Set DRAK = ”1” • if you want to implement wire-break monitoring according to above and • if the relevant analog value is outside the wire-break limits at this point	
ODGR [1]	Wire-break upper limit	D	KF	Entry is only relevant if DRAK = 1 Enter the following here • The limits of the range within which values are interpreted as wire-breaks (DRAK) • - 32768 if you have configured a global limit value for wire-break in DB1.	
UDGR [1]	Wire-break lower limit	D	KF		
PASS	Passivation value	D	KF	At passivation of the module, the safety-related passivation value is written into the result word instead of the AI value read in.	
EINZ	Selective sampling	I	BI	A ”1” initiates selective sampling (only in the case of the AI 460)	
DIR	Direct access	I	BI	Set DIR=”1” if direct access is desired	
XA	Output value	Q	W	The scaled analog value is ”0” in the event of a wire-break	
F [2]	Error bit	Q	BI	Becomes ”1” in the event of wire-break	
BU [2]	Overrange	Q	BI	Becomes ”1” if the analog input of one or both subunits exceeds the nominal range	
TBIT	Activity bit of the accessed module	Q	BI	Becomes ”1” if the accessed module is currently executing selective sampling	
TKON	Time conflict	Q	BI	Becomes ”1” in the following cases • if FB 250 accesses an analog input which is currently being tested or • if a discrepancy results in the case of direct access with the parameter DIR = 1 and the last valid value instead of the current value is assigned to output value XA. The values XA, FB and BU are not up-dated. Only the old values are available for them.	

1 Values of the scaled range (UGR to OGR)

2 If both bits F and BU = ”1”, passivation has taken place in the relevant modules. The passivation value is in the value XA.

Outputting an analog value -FB 251-

Use function block FB 251 to output analog values to analog output modules. Specify the type of analog representation of the module (channel type) in the KNKT parameter. Values from the range between the “lower limit” (UGR) and the “upper limit” (OGR) parameters are converted to the nominal range of the module in question.

Call and Parameter Assignment

Parameter	Meaning	Type	Data Type	Assignment	STL
XE	Analog value to be output	I	W	Input value (fixed-point) in the UGR to OGR range	: JU FB 251 NAME : AGF : ANAU XE : BG : KNKT : OGR : UGR : FEH : BU :
BG	Module address	D	KF	128 to 240	
KNKT	Channel number Channel type	D	KY	KY = x,y x = 0 to 7 y = 0;1 0: unipolar representation 1: bipolar fixed-point number	
OGR	Upper limit of the output value	D	KF	-32768 to+32767	
UGR	Lower limit of the output value	D	KF	-32768 to+32767	
FEH	Error when setting the limit value	Q	BI	”1” if UGR = OGR, for illegal channel or slot number, or illegal channel type	
BU	Input value exceeds UGR or OGR	Q	BI	At ”1,”XE is outside the range (UGR;OGR). XE accepts the limit value	

6.1.4 The FB 252 Block Transfer Block

FB 252 handles control of data transfer between the CPU and the CP 523 serial input/output module. FB 252 transfers fixed-length data. Up to 32 data blocks of eight bytes each are read out of or written into the memory area per call. Memory areas are data blocks or flag areas.

Parameter	Meaning	Type	Data Type	Assignment	STL
BADR	Initial module address	D	KF	128 to 248 (in eight-byte steps)	: JU FB 252 NAME : AGF: BLUE BADR : FUNK : TYP : DBNR : ANFA : BLCK :
FUNK	Function	D	KC	S: = Send data R: = Receive data	
TYP	Memory type of source/destination in the case of S/R	D	KC	D: = Data block F: = Flag area XX: = Indirect block initialization (the block parameters are stored in a data block)	
DBNR	Block number	D	KF	In the case of data block: 4 to 255 In the case of flag area: 0 in the case of indirect block Initialization: 0; 4 to 255	
ANFA	First word of data block or flag area	D	KF	In the case of data block: 0 to 252 In the case of flag area: 2 to 248	
BLCK	Number of blocks	D	KF	1 - 32 Specification of eight-byte data blocks sent/received	

Indirect assignment of FB 252 parameters

If you want to assign the FB 252 parameters indirectly, store the six block parameters in a data block in consecutive data words. Please note the order of the parameters!

When calling FB 252 you must also specify the following:
- The value ”XX” in the parameter type
- The number of the DB (DB 4 to 255) with the block parameter assignments for the DBNR parameter
  or
  the value ”C” for the DBNR parameter if the last opened DB contains the block parameter assignments
- The number of the data word containing the first block parameter (BADR parameter) for the ANFA parameter.

The BADR, FUNK and BLCK parameters must be assigned permissible values but the values themselves are insignificant in this case.

6.1.5 The FB 253 Mailbox Transfer Block

If you are using a SINEC L1 LAN, it is connected to the B subunits of the relevant S5-115Fs. The user program accesses the Receive mailbox direct without the FB 253.

If you have installed a dual-bus SINEC L1 LAN, you will have two Receive mailboxes (Bus A and Bus B). Call the 253 MBXT mailbox transfer block if you wish to exploit the fault tolerance of the dual-channel SINEC L1 LAN. FB 253 copies the contents of the Receive mailbox of Bus A into the Receive mailbox of Bus B if an error has been reported in connection with the data traffic of Bus B. Your program must then access the Receive mailbox of Bus B. If the other bus also develops a fault, the system responds with an error message and the PLC enters the STOP mode.

Call and Parameter Assignment

Parameter	Meaning	Type	Data Type	Assignment	STL
QSLN	Source slave No.	D	KF	KF = 1 to 30 Any number between 1 and 30 can be given as the source slave No., with the exception of the number of the slave in question. A Receive mailbox for both SINEC L1 buses must be defined for every source slave.	: JU FB 253 Name : AGF:MBXT QSLN :

6.1.6 The FB 254 Synchronization Block

The following are regularly synchronized during the operating system run:

- Updating of timers (beginning and end of the OS)
- Requests of the time interrupt OB 13 (every 20 ms)
- Requests of the process interrupt OB 2 (every 20 ms) (2.3.1)
- PG execution
- SINEC L1 requests (every 20 ms).

Call the FB 254 synchronization block every time you wish to synchronize these functions within your program.
You can define which functions are to be synchronized with the SYNC parameter.

Note

If you implement interval monitoring for the interrupt handling calls in the user program, you must also always use the SYNC parameter to synchronize user times when synchronizing interrupt handling.

Call and Parameter Assignment

Parameter	Meaning	Type	Data Type	Assignment	STL
SYNC	Synchronization parameter	D	KM	Bit 0 = 1 : Update of user times Bit 1 = 1 : Synchronize handling of interrupt and time OB 13 Bit 2 = 1 : Programmer processing Bit 3 = 1 : SINEC L1 processing All 16 possible combinations are permissible. Both subunits are synchronized at 0000. Bits 4 to 7 = 0 Bits 8 to 15: Freely programmable localization parameters	: JU FB 254 Name : AGF:SYNC SYNC :

You can continue to specify a location parameter in bits 8 to 15 of the synchronization parameter. Assign an ordinal number in your user program at each FB 254 SYNC call.
In the event of an error message as a result of intervals between FB 254 SYNC calls being too long, the location parameter of the first wrongly located FB 254 SYNC is output. This makes it easier for the user to correct errors.

The location parameter is displayed in hexadecimal form. The location parameter 00_H is always specified in the case of error messages when the monitoring facility has detected synchronization intervals in the monitoring system which are too long.

However, if you want to dispense with assignment of location parameters, set bits 8 to 15 of the synchronization parameter to ”0”. Please note that all error messages take the location parameter ”00_H” in this case because of excessive call intervals. You receive no indication as to whether the monitoring facility has detected excessive synchronization intervals in the operating system or in the user program.

Note

Make the intervals between the FB 254 calls 10 ms shorter for a desired interrupt response time.

Note

FB 254 calls make the scan time longer. Always try to activate as many functions as possible with as few calls as possible.

Optimization of Call Frequency

Establish which functions you require frequently

Draw up a diagram of the PLC cycle and mark all calls
- on their individual axis and
- on a shared axis

Example:

The following are used for a PLC cycle time of 300 ms, (user program 200 ms, operating system 100 ms)

a every 160 ms User timer updating
b every 30 ms Interrupt handling (OB 2) and time OB processing (OB 13)
c every 100 ms Programmer input processing
d every 120 ms SINEC L1 processing

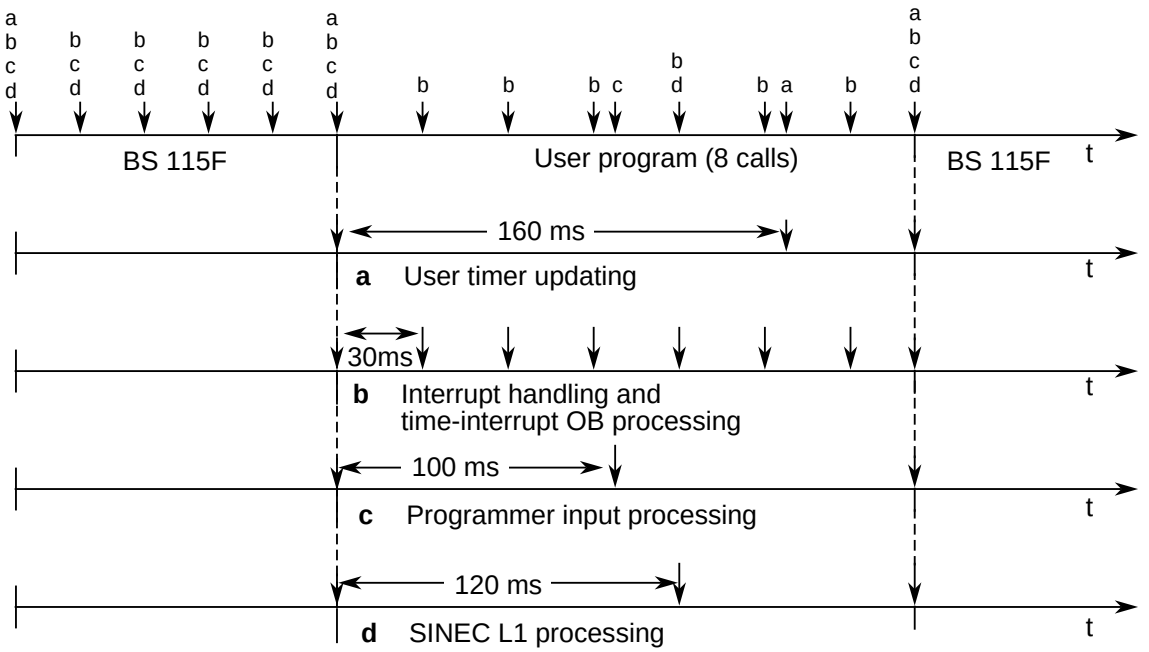


Figure 6-2. FB 254 Calls (Ideal Schematic)

FB 254 is called 8 times per cycle.

Reduce the intervals to a common time multiple. Optimize the call intervals to keep the frequency of calls to a minimum.

Example:

		30	ms	Common time multiple
a	all	150	ms	User time updating
b	all	30	ms	Interrupt handling (OB 2) and time OB processing (OB 13)
c	all	90	ms	Programmer input processing
d	all	120	ms	SINEC L1 processing

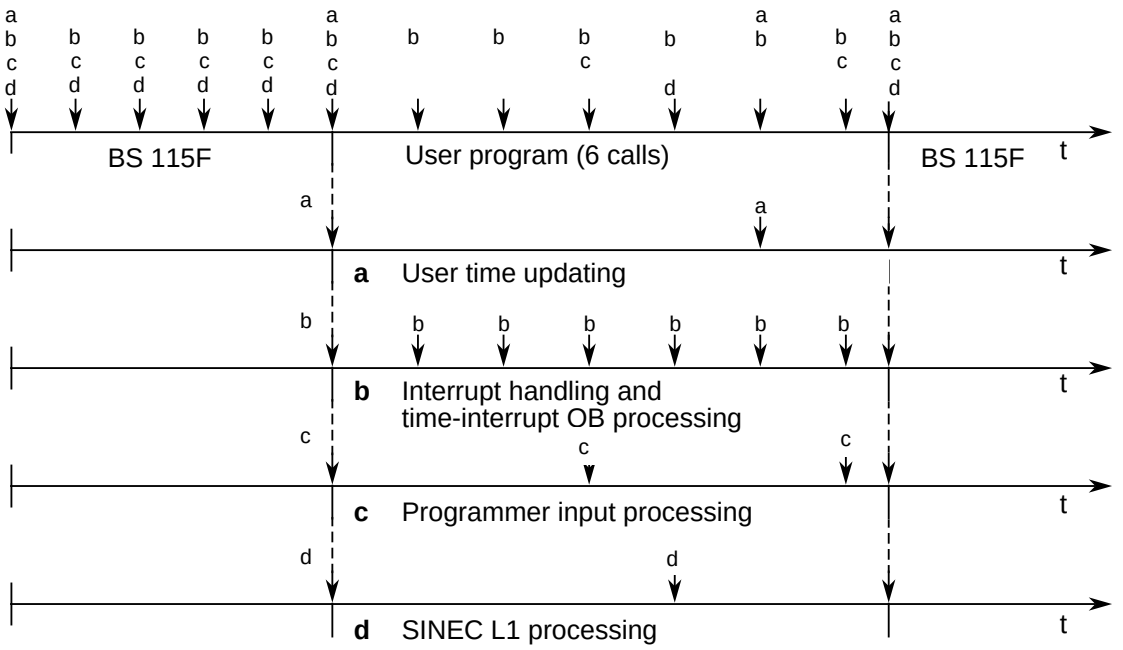


Figure 6-3. FB 254 Calls (Final Schematic)

FB 254 is called only six times.

Distribute the calls in your program in such a way that a call becomes effective at the latest every 29 ms Note all paths along which the program can run.

Note

Take account of the call intervals defined here when you are setting the monitoring times with COM 115F.

6.1.7 Depassivation Block FB 255

Defective sensors and actuators are often the cause of passivation of I/O modules. If you can rectify the faults in PLC RUN, FB 255 allows passivation to be revoked; the depassivated I/O module is then referenced again by the PLC.

You may only call FB 255 in the cyclic part of the program (OB 1). FB 255 must not be called in time-controlled or interrupt-driven program parts.

When assigning the FB 255 parameters, you must specify the following:
- a bit which initiates depassivation in the event of an edge change from 0 to 1
- a datum in KF format with the signal group for the I/O module to be depassivated
- a byte for FB 255 messages

Calling and assigning parameters

Parameter	Meaning			Assignment	STL
SIGR	Signal group	D	KF	2...28	: SPA FB 254 Name : DEPASS : SIGR : DEPA : PAFE
DEPA	Depassivation bit (positive edge)	I	BI	I0.0 to 127.7 F2.0 to 199.7 Q0.0 to 125.7	
PAFE	Message byte	Q	BI	00_H = Depassivation successful 11_H = Signal group has not been configured or passivated 21_H = Signal group can no longer be passivated (error no longer in error DB) 31_H = Depassivation has not been tested 41_H = FB 255 parameters wrongly assigned 51_H = No signal group has been passivated DO_H = Depassivation being executed now	

Note

Before depassivating the I/O module of a signal group with FB 255, you must branch to a subroutine in your control program. You must evaluate all the variables required for the process in this subroutine and if necessary update or initialize them

6.2 OB 251 PID Algorithm

The operating sytem of the CPU 942F has a PID algorithm which the user can apply to his needs with the help of the OB 251 organization block.
Before calling OB 251, you must open a data block (PID DB) which contains the PID parameters and other controller-specific data. The PID algorithm is called in a specific time grid (sampling interval) and generates a manipulated variable. The greater the accuracy of the sampling interval, the more accurately the controller can perform its tasks. The PID control parameters specified in the PID DB must be adapted to the sampling interval.
Time-controlled processing is generally implemented with the time-interrupt OB (OB 13). Organization block OB 13 can be programmed in call intervals of 10 ms to 10 min. Execution of the PID control algorithm can take up to 1.7 ms.

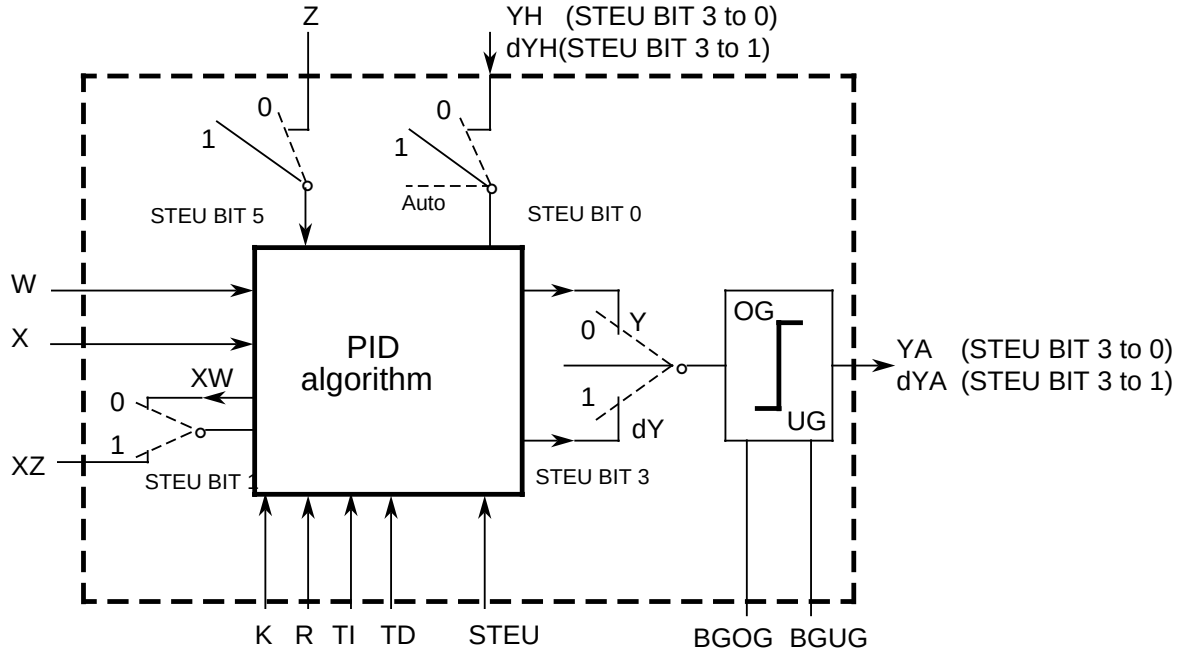


Figure 6-4. Block Diagram of the PID Controller

Legend:

K	= Proportional action coefficient K>0 positive control direction K<0 negative control direction	STEU	= Control word
R	= R parameter (usually 1000)	Y	= Manipulated variable
TA	= Sampling interval	dY	= Correcting increment
TN	= Integral-action time	YH	= Value for manual input
TV	= Derivative-action time	dYH	= Correcting increment for manual input
TI	= TA/TN	BGOG	= Upper limit
TD	= TV/TA	BGUG	= Lower limit
W	= Setpoint	X	= Actual value input
XW	= Control deviation	Z	= Disturbance variable
		XZ	= Substitute variable for control deviation
		YA	= Controller output: Manipulated variable limited
		dYA	= Controller output: Correcting increment limited

The continuous-action controller is designed for controlled systems involving pressure, temperature, or flow rate control.

The “R” parameter sets the proportional-action component of the PID controller.
If proportional action is required, most controller designs use the value R=1.

The individual proportional-action, integral-action, and derivative-action components can be deactivated via their particular parameters (R, TI, and TD) by presetting the pertinent data words with zero. This enables you to implement all desired controller structures easily, e.g. PI, PD, or PID controllers.

You can apply the control deviation XW or, using the XZ input, any disturbance variable or the inverted actual value X to the differentiator. Specify a negative K value for an inverted control direction.
When the correction information (dY or Y) is at a limit, the integral-action component is automatically deactivated in order not to impair the dynamic response of the controller.

The switch settings in the block diagram are implemented when programming the PID controller. This is done by setting the appropriate bits in the “STEU” control word (Table 6-2).

Table 6-2. Meaning of the Control Bits in Control Word STEU

Control Bit	Name	Signal State	Meaning
0	AUTO	0 1	Manual mode Automatic mode
1	XZ EIN	0 1	XW_k is applied to the differentiator. No account is taken of the XZ input. A variable other than XW_k is applied to the differentiator via the XZ input.
2	REG AUS	0 1	Normal control operation When the controller is called (OB 251), all other variables (DW 18 to DW 48) are erased with the exception of D, R, TI, TD, BGOG, BGUG, YH_k and W_k. The controller is deactivated.
3	GESCHW	0 1	Correction algorithm Correction rate algorithm
4	HAND-ART	0 1	For GESCHW=0: The controller output signal YA is applied exponentially to the set manual value in four sampling steps after switching over to manual mode. Other manual values are then accepted immediately at the controller output. For GESCHW=1: The manual values are routed through to the controller output immediately. The limits are effective in manual mode. The following variables are updated in manual mode: 1) X_K, XW_{K-1} and PW_{K-1} 2) XZ_K, XZ_{K-1} and PZ_{K-1}, if STEU bit 1=1 3) Z_K and Z_{K-1}, if STEU bit 5=0 The variable dD_{K-1} is set to = 0. The algorithm is not computed. For GESCHW=0: The last manipulated variable output is retained. For GESCHW=1: The correcting increment dY_K is set to zero.
5	NO Z	0 1	With feedforward injection of disturbance variable No feedforward injection of disturbance variable
6 to 15	-		These bits are used by the PID algorithm as flag bits.

The control program can be initialized with fixed values or parameters. Parameters are input via the assigned data words. The controller is based on a PID algorithm. Its output signal can be either a manipulated variable (correction algorithm) or a manipulated variable change (correction rate algorithm).

Correction Rate Algorithm

A particular correcting increment dY_k is calculated at a particular instant t= k · TA according to the following formula:

- Without feedforward injection of disturbance variable (D11.5 = 1) and application of XW to the differentiator (D11.1 = 0)

$$dY_k = K[(XW_k - XW_{k-1})R + TI \cdot XW_k + (TD(XW_k - 2XW_{k-1} + XW_{k-2}) + dD_{k-1})]$$
$$= K(dPW_kR + dI_k + dD_k)$$

- With feedforward injection of disturbance variable (D11.5 = 0) and application of XW to the differentiator (D11.1 = 0)

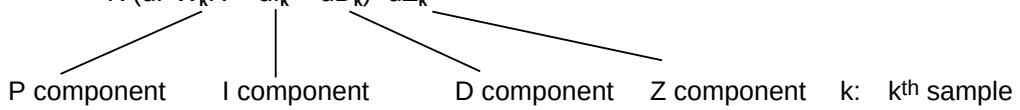
$$dY_k = K[(XW_k - XW_{k-1})R + TI \cdot XW_k + (TD(XW_k - 2XW_{k-1} + XW_{k-2}) + dD_{k-1})] + (Z_k - Z_{k-1})$$
$$= K(dPW_kR + dI_k + dD_k) + dZ_k$$

- Without feedforward injection of disturbance variable (D11.5 = 1) and application of XZ to the differentiator (D11.1 = 1)

$$dY_k = K[(XW_k - XW_{k-1})R + TI \cdot XW_k + (TD(XZ_k - 2XZ_{k-1} + XZ_{k-2}) + dD_{k-1})]$$
$$= K(dPW_kR + dI_k + dD_k)$$

- With feedforward injection of disturbance variable (D11.5 = 0) and application of XZ to the differentiator (D11.1 = 1)

$$dY_k = K[(XW_k - XW_{k-1})R + TI \cdot XW_k + (TD(XZ_k - 2XZ_{k-1} + XZ_{k-2}) + dD_{k-1})] + (Z_k - Z_{k-1})$$
$$= K(dPW_kR + dI_k + dD_k) + dZ_k$$



When XW_k is applied:

$XW_k = W_k - X_k$
$PW_k = XW_k - XW_{k-1}$
$QW_k = PW_k - PW_{k-1}$
$= XW_k - 2XW_{k-1} + XW_{k-2}$

When XZ is applied:

$PZ_k = XZ_k - XZ_{k-1}$
$QZ_k = PZ_k - PZ_{k-1}$
$= XZ_k - 2XZ_{k-1} + XZ_{k-2}$

The result is:

$dPW_k = (XW_k - XW_{k-1})R$
$dI_k = TI \cdot XW_k$
$dD_k = (TD \cdot QW_k + dD_{k-1})$ with XW application
$= (TD \cdot QZ_k + dD_{k-1})$ with XZ application
$dZ_k = Z_k - Z_{k-1}$

Correction Algorithm

The same algorithm is used for the correction algorithm as for the correction rate algorithm.
The difference compared with the correction rate algorithm is that, at the sampling point t_k, the sum of all control increments dY_k calculated up to this point is output (in DW 48) instead of the correcting increment calculated at this instant.

At the instant t_k, manipulated variable Y_k is calculated as follows:

$$Y_k = \sum_{m=0}^{m=k} dY_m$$

Assigning Parameters to the PID Algorithm

The controller DB is OB 251's interface to its environment.
All data required for calculating the next value is stored in the controller DB. Every controller requires its own controller DB.
Controller-specific data parameters are set in a data block, which must include at least 49 data words.
If no DB is open or if the DB is not long enough, the CPU will stop with a transfer error (TRAF).



Important

Make sure the relevant controller DB has been opened before calling the OB 251 control algorithm. There must be no safety-related data in the controller DB. Opening a DB with safety-related data can lead to dangerous states since OB 251 has only been accepted as a ”reaction-free” block and not as a ”failsafe” block.
The inspector checks the call of the controller DB during the acceptance test.

Table 6-3. Structure of the Data Transfer Block

Data Word	Name	Remarks
1	K	Proportional action coefficient (-32 768 to +32 767) in the case of controllers without D component Proportional action coefficient (-1500 to +1500) in the case of controllers with D component [1] K is greater than zero in positive control direction and less than zero in negative control direction; the value specified is multiplied by the factor 0.001.
3	R	R parameter (- 32 768 to + 32 767) in the case of controllers without D component R parameter (-1500 to +1500) in the case of controllers with D component [1] Usually equal to 1 in controllers with P component, the value specified is multiplied by the factor 0.001.
5	TI	Constant TI (0 to 9999) $TI = \frac{\text{Sampling interval TA}}{\text{Integral-action time n}}$ The value specified is multiplied by the factor 0.001.
7	TD	Constant TD (0 to 999) $TD = \frac{\text{Derivative-action time TV}}{\text{Sampling time TA}}$
9	W	Setpoint (- 2047 to +2047)
11	STEU	Control word (bit pattern) (See table 6-2 for assignments)
12	YH	Value for manual mode (- 2047 to +2047)
14	BGOG	Upper limit (- 2047 to +2047)
16	BGUG	Lower limit (- 2047 to +2047)
22	X	Actual value (- 2047 to +2047)
24	Z	Disturbance variable (- 2047 to +2047)
29	XZ	Applied D component (- 2047 to +2047)
48	YA	Output variable (- 2047 to +2047)

1 Greater amplifications are possible if jump-like changes in the control difference are sufficiently small. Large changes of the control difference should therefore be divided into several small changes; e.g. by applying the setpoint over a ramp function.

All parameters (with the exception of the control word STEU) must be specified as 16-bit fixed-point numbers.

Important

The data words not listed in the table are used by the PID algorithm as auxiliary words.

Initializing and calling the PID controller in the STEP 5 program

Different PID controllers can be implemented with multiple OB 251 calls. Data transfer via a relevant data block must be ensured before each call. These DBs are the data interface between the controllers and the user.

Note

The important controller information is stored in the high byte of control word DW 11 (DL 11). It is therefore important to note that when influencing the user bits in the control word, write accesses are only made with T DR 11/SU D 11.0 to D 11.7 or with RU D 11.0 to D 11.7.

Selecting the sampling interval

The sampling time in digital control loops must not be too long (cf. analog control loops).
Experience has shown that a sampling interval TA of approximately 1/10 of the time constant $T_{RK,\ dom}$* leads to a controller result comparable with the analog case. The time constant $T_{RK,\ dom}$ determines the jump response of the closed control circuit.

$$TA = 1/10 \cdot T_{RK,\ dom}$$

Note

To guarantee a constant sampling interval, OB 251 must be called in the time OB (OB 13).

* $T_{RK,\ dom}$ =dominant time constant of the closed control loop

Typical application of the PID control algorithm in the S5-115F

A furnace is to be maintained at a constant temperature by a PID controller.
The temperature setpoint is entered by a potentiometer.
The setpoints and actual values reach the controller via an analog input module. The calculated manipulated variable is then output via an analog output module. Neither the analog modules nor the controller itself are safety-related.
The operating mode of the controller is set in input byte 0 (see control word DW 11 in the controller DB).
The controller setting must be calculated for each controlled system by the usual controller design method.

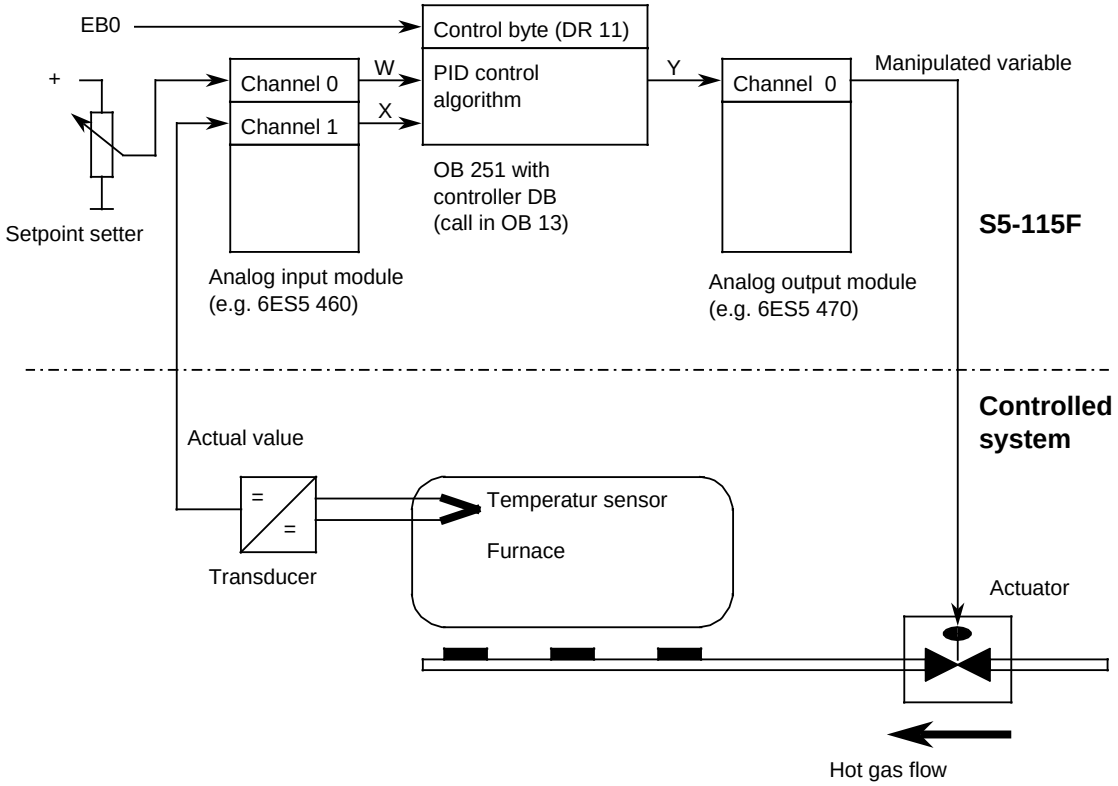


Figure 6-5. Process Schematic

The analog signals of the setpoints and actual values are converted into their corresponding digital values at each sampling instant (determined by time OB 13). From this, OB 251 calculates the new manipulated variable from which a corresponding analog signal will be generated with the analog output. This signal is then applied to the controlled system.

Calling the controller in the program:

OB 13 STL	Explanation
NETWORK 1 0000	
0000 :L FW 0	LPLZ SEQUENCE
0001 :L KF +1	
0003 :+F	
0004 :T FW 0	
0005 :	
0006 :JU FB 10	PROCESS CONTROLLER
0007 NAME :REGLER 1	
0008 :	THE CONTROLLER SAMPLING TIME IS
0009 :	DETERMINED BY THE OB 13 CALL TIME,
000A :	WHICH IS SET WITH COM 115F IN THE
000B :	”INITIALIZING THE OPERATING SYSTEM”
000C :	SCREEN FORM. THE CODING TIME
000D :	OF THE ANALOG INPUT MODULES
000E :	MUST BE TAKEN INTO ACCOUNT
000F :	WHEN SELECTING
0010 :	THE SAMPLING TIME.
0011	

FB 10 STL	Explanation
NETWORK 1 0000	
NAME :REGLER 1	
0005 :L FW 0	LPLZ-SEQUENCE
0006 :L KF +1	
0008 :+F	
0009 :T FW 0	
000A :	
000B :C DB 30	OPEN CONTROLLER DB
000C :	
000D :	**********
000E :	READ IN CONTROL BITS FOR CONTROLLER
000F :	**********
0010 :	
0011 :L PY 0	READ IN CONTROL INPUTS
0012 :T FY 10	FOR CONTROLLER
0013 :T DR 11	AND STORE IN DR 11
0014 :	CAUTION:
0015 :	IMPORTANT CONTROL INFORMATION
0016 :	FOR OB 251 IS STORED IN DL 11.
0017 :	FOR THIS REASON THE CONTROL BITS
0018 :	MUST BE TRANSFERRED WITH DR 11
0019 :	IN ORDER NOT TO
001A :	INFLUENCE DL 11.
001B :	
001C :	**********
001D :	READ IN ACTUAL VALUE AND SETPOINT
001E :	**********
001F :	
0020 :A F 12.0	ZERO FLAG
0021 :R F 12.0	(FOR UNUSED FUNCTIONS IN FB 250)
0022 :AN F 12.1	1 FLAG
0023 :S F 12.1	
0024 :	
0025 :JU FB 250	READ IN ACTUAL VALUE
0026 NAME :AGF:ANEI	
0027 BG : KF +128	MODULE ADDRESS
0028 KN : KF +0	CHANNEL NUMBER
0029 OGR : KF +2047	UPPER LIMIT ACTUAL VALUE
002A UGR : KF -2047	LOWER LIMIT ACTUAL VALUE
002B DRAK : F 12.0	NO WIRE-BREAK DETECTION
002C ODGR : KF +0	IRRELEVANT SINCE NO DRAK
002D UDGR : KF +0	IRRELEVANT SINCE NO DRAK
002E PASS : KF +0	PASSIVATION VALUE
002F EINZ : F 12.0	NO SELECTIVE SAMPLING
0030 DIR : F 12.1	READ IN VIA DIRECT ACCESS
0031 XA : DW 22	STORE SCALED ACTUAL VALUE IN CONTROL DB
0032 FB : F 12.2	ERROR BIT
0033 BU : F 12.3	OVERRANGE
0034 TBIT : F 12.4	ACTIVITY BIT
0035 TKON : F 12.5	TEST CONFLICT
0036 :	
0037 :	
0038 :JU FB 250	READ IN SETPOINT

FB 10 (Continued) STL	Explanation
0039 NAME :AGF:ANEI	
003A BG : KF +128	MODULE ADDRESS
003B KN : KF +1	CHANNEL NUMBER
003C OGR : KF +2047	UPPER LIMIT SETPOINT
003D UGR : KF -2047	LOWER LIMIT SETPOINT
003E DRAK : F 12.0	NO WIRE-BREAK DETECTION
003F ODGR : KF +0	IRRELEVANT
0040 UDGR : KF +0	IRRELEVANT
0041 PASS : KF +0	PASSIVATION VALUE
0042 EINZ : F 12.0	NO SELECTIVE SAMPLING
0043 DIR : F 12.1	READ IN VIA DIRECT ACCESS
0044 XA : DW 9	STORE SCALED SETPOINT IN CONTROL DB
0045 FB : F 13.1	ERROR BIT
0046 BU : F 13.2	OVERRANGE
0047 TBIT : F 13.3	ACTIVITY BIT
0048 TKON : F 13.4	TEST CONFLICT
0049 :	
004A :A F 10.0	IN MANUAL MODE, THE
004B :JC =WEIT	SETPOINT IS SET TO EQUAL
004C :L DW 22	THE ACTUAL VALUE SO THAT
004D :T DW 9	THE CONTROLLER WILL RESPOND
004E :	TO ANY CONTROL DEVIATION
004F :	WITH A P JUMP WHEN
0050 :	SWITCHING TO
0051 :	AUTOMATIC MODE
0052 WEIT :	
0053 :	******************************
0054 :JU OB 251	CALL CONTROLLER
0055 :	******************************
0056 :	
0057 :	******************************
0058 :	OUTPUT MANIPULATED VARIABLE Y
0059 :	******************************
005A :JU FB 251	
005B NAME :AGF:ANAU	
005C XE : DW 48	MANIPULATED VARIABLE Y TO ANALOG OUTPUT
005D BG : KF +176	MODULE ADDRESS
005E KNKT : KY 0,1	CHANNEL 0, FIXED-POINT BIPOLAR
005F OGR : KF +2047	UPPER LIMIT OF CORRECTING SIGNAL
0060 UGR : KF -2047	LOWER LIMIT OF CORRECTING SIGNAL
0061 FEH : F 13.5	ERROR BIT FOR LIMIT SPECIFICATION
0062 BU : F 13.6	OVERRANGE
0063 :BE	

DB 30	STL	Explanation
0:	KH = 0000;	
1:	KF = +01000;	K PARAMETER(HERE=1),FACTOR 0.001
2:	KH = 0000;	(VALUE RANGE: -32768 TO 32767)
3:	KF = +01000;	R PARAMETER(HERE=1),FACTOR 0.001
4:	KH = 0000;	(VALUE RANGE: -32768 TO 32767)
5:	KF = +00010;	TI=TA/TN(HERE=0.01),FACTOR 0.001
6:	KH = 0000;	(VALUE RANGE: 0 TO 9999)
7:	KF = +00010;	TD=TV/TA (HERE=10), FACTOR 1
8:	KH = 0000;	(VALUE RANGE: 0 TO 999)
9:	KF = +00000;	ACTUAL VALUE W, FACTOR 1
10:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
11:	KM = 00000000 00100000;	CONTROL WORD
12:	KF = +00500;	MANUAL VALUE YH, FACTOR 1
13:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
14:	KF = +02000;	UPPER LIMIT SETPOINT BGOG, FACTOR 1
15:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
16:	KF = -02000;	LOWER LIMIT SETPOINT BGUG, FACTOR 1
17:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
18:	KH = 0000;	
19:	KH = 0000;	
20:	KH = 0000;	
21:	KH = 0000;	
22:	KF = +00000;	ACTUAL VALUE X, FACTOR 1
23:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
24:	KF = +00000;	DISTURBANCE VARIABLE Z, FACTOR 1
25:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
26:	KH = 0000;	
27:	KH = 0000;	
28:	KH = 0000;	
29:	KF = +00000;	XZ FEEDFORWARD FOR DIFF.
30:	KH = 0000;	FACTOR 1, (-2047 TO 2047)
31:	KH = 0000;	
32:	KH = 0000;	
33:	KH = 0000;	
34:	KH = 0000;	
35:	KH = 0000;	
36:	KH = 0000;	
37:	KH = 0000;	
38:	KH = 0000;	
39:	KH = 0000;	
40:	KH = 0000;	
41:	KH = 0000;	
42:	KH = 0000;	
43:	KH = 0000;	
44:	KH = 0000;	
45:	KH = 0000;	
46:	KH = 0000;	
47:	KH = 0000;	
48:	KF = +00000;	CONTROLLER OUTPUT Y, FACTOR 1
49:	KH = 0000;	(VALUE RANGE: -2047 TO 2047)
50:		

6.3 Loadable Function Blocks

Frequently repeated or particularly complex program sections (such as event signalling functions and compute functions) are programmed in function blocks. These blocks have specifiable parameters and an extended operation set (e.g. Jump operations within a block).

The loadable function blocks are subdivided into

- User-written function blocks
- Standard function blocks

User-Written Function Blocks

As is the case for all other blocks, function blocks written by the user must be examined by the authorized inspector.

At the very least, such an examination comprises a verbal description of the function, functional tests and a code analysis.

Standard Function Blocks

In the S5-115F, standard function blocks may be used only when they have been prototype-tested. The standard FBs for the S5-115F implement the same functions as the standard FBs for the SIMATIC S5 U-range controllers, but have been adapted to state-of-the-art failsafe technology. An overview of the standard function blocks available for the S5-115F can be found in Catalog ST 57.

There are reaction-free and failsafe standard FBs. Only failsafe standard FBs may be used to form signals relevant to the system's failsafety. When initializing the block parameters, note that output parameters are failsafe only when all input parameters are failsafe.

All standard FBs are identified by a library number which, in conjunction with other measures, protect the blocks against falsifications caused by transfer errors or disk errors.

Note the following when using loadable standard FBs:

- In the safety mode, the operating system checks all standard FBs for integrity
- Standard FBs for the SIMATIC S5 U-range controllers are not allowed, and are rejected in the safety mode
- The authorized inspector's examination of the standard FBs limits itself to
  - a comparison of the library number with the one specified in the report
  - checking for proper use of the standard FBs
  - checking the FB calls for correctness and checking the parameter initialization routine for the FBs as per the block description
  - making sure that the provisions laid down in the report have been upheld

Note

When using standard FBs, flag words FW 200 to FW 254 may be used in the application program only when these flags are saved prior to invoking a standard FB and reloaded once that FB has executed.

7 Application

7.1 Hardware Installation . . . . . 7 - 1
7.1.1 Installing and Wiring the Hardware . . . . . 7 - 2

7.2 Entering the Subunit ID . . . . . 7 - 5

7.3 Initializing the Operating System . . . . . 7 - 5

7.4 Configuring the Hardware with COM 115F . . . . . 7 - 7

7.5 Transferring the Configuration Data to the PLC . . . . . 7 - 11

Figures

7-1. System Configuration for the Application 7.- 1
7-2. Switch and Jumper Settings for Parallel Link 7.- 2
7-3. Wiring the I/O Modules 7.- 3
7-4. Initialization Menu of the 115F Operating System for CPU 942-7UF13 . . . . . 7 - 5
7-5. Initialization Screen Form for I/O Type 1 . . . . . 7.- 7
7-6. Initialization Screen Form for I/O Type 3 . . . . . 7.- 8
7-7. Initialization Screen Form for I/O Type 8 . . . . . 7.- 9
7-8. Initialization Screen Form for I/O Type 10 . . . . . 7.-10

7 Application

This section contains a simple example of an S5-115F in test mode. The I/O modules used include the following:

- 32 reaction-free digital input modules, I/O type 1
- 32 failsafe digital input modules, I/O type 3
- 32 reaction-free digital output modules, I/O type 8
- 16 failsafe digital output modules, I/O type 10

The example contains, in a compact form, all the information necessary for mechanical installation of the central controllers, wiring of I/O modules and configuration with COM 115F.
If required, the example can be expanded to suit all needs.

Hardware
You require the following for installing the S5-115F:
- 2 CR 700-2F subracks
- 2 CPU 942-7UF13 central processing units
- 2 PS 951 power supply units
- 2 DI 430-7LA12 digital input modules
- 1 DI/DQ 482-7LF11 digital input/output module
- 1 DI/DQ 482-7LF21 digital input/output module
- 1 DQ 451-7LA11 digital output module
- 1 IM 304-3UB11 interface module with adapter casing
- 1 IM 324-3UA12 interface module with adapter casing
- 1 721 connector cable
- 2 IM 306 interface modules
- 1 220V/24V power supply for the I/O modules

Software
You require
- The COM 115F software package, Version 3.0
- The STEP 5 basic package

7.1 Hardware Installation

Hardware installation and wiring of the I/O modules are described in this section.

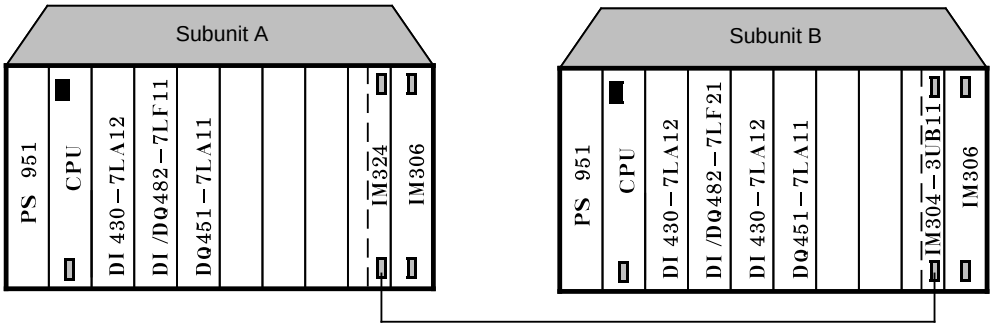


Figure 7-1. System Configuration for the Application

The following are implemented with this configuration

- Single-channel I/O modules
  32 digital input modules, I/O type 1, address 32.0 to 35.7, subunit B, slot 2
  32 digital output modules, I/O type 10, address 36.0 to 39.7, subunit B, slot 3
- Two-channel I/O modules
  32 digital input modules, I/O type 3, address 0.0 to 3.7, subunit A+B, slot 0 with 32 check DQs in subunit A, slot 2
  16 digital output modules, I/O type 8, address 4.0 to 5.7, subunit A+B, slot 1 with readback DI in subunit A+B

7.1.1 Installing and Wiring the Hardware

Secure both central racks
Plug the PS 951, the CPU and the digital modules into the central rack (Figure 7-1).
Plug an IM 306 into the central rack of subunit A and set the addressing for subunit A.
Plug an IM 306 into the central rack of subunit B and set the addressing for subunit B.
Check the settings for the parallel link on the IM 304 and IM 324 interface modules (Figure 7-2)

Plug the interface modules with the adapter casings into slot 7 (Figure 7-1)
Connect the IM 304 and IM 324 interface modules together with the 721 connecting cable.
Use the bottom connector on the IM 304 (connector X4)
Wire the I/O modules as shown in Figure 7-3.
Connect the PS 951F power supply.

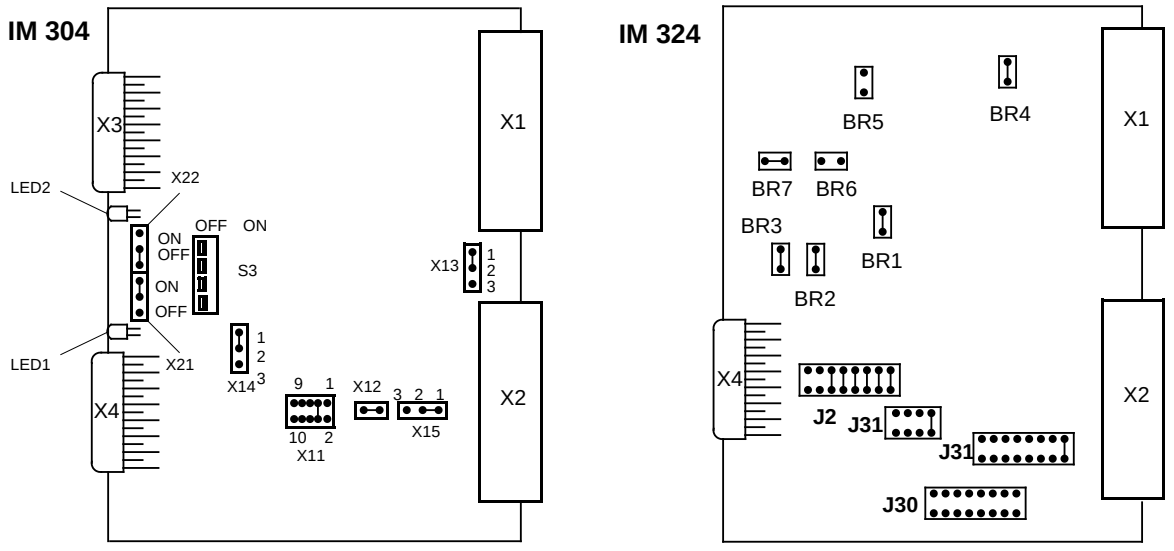
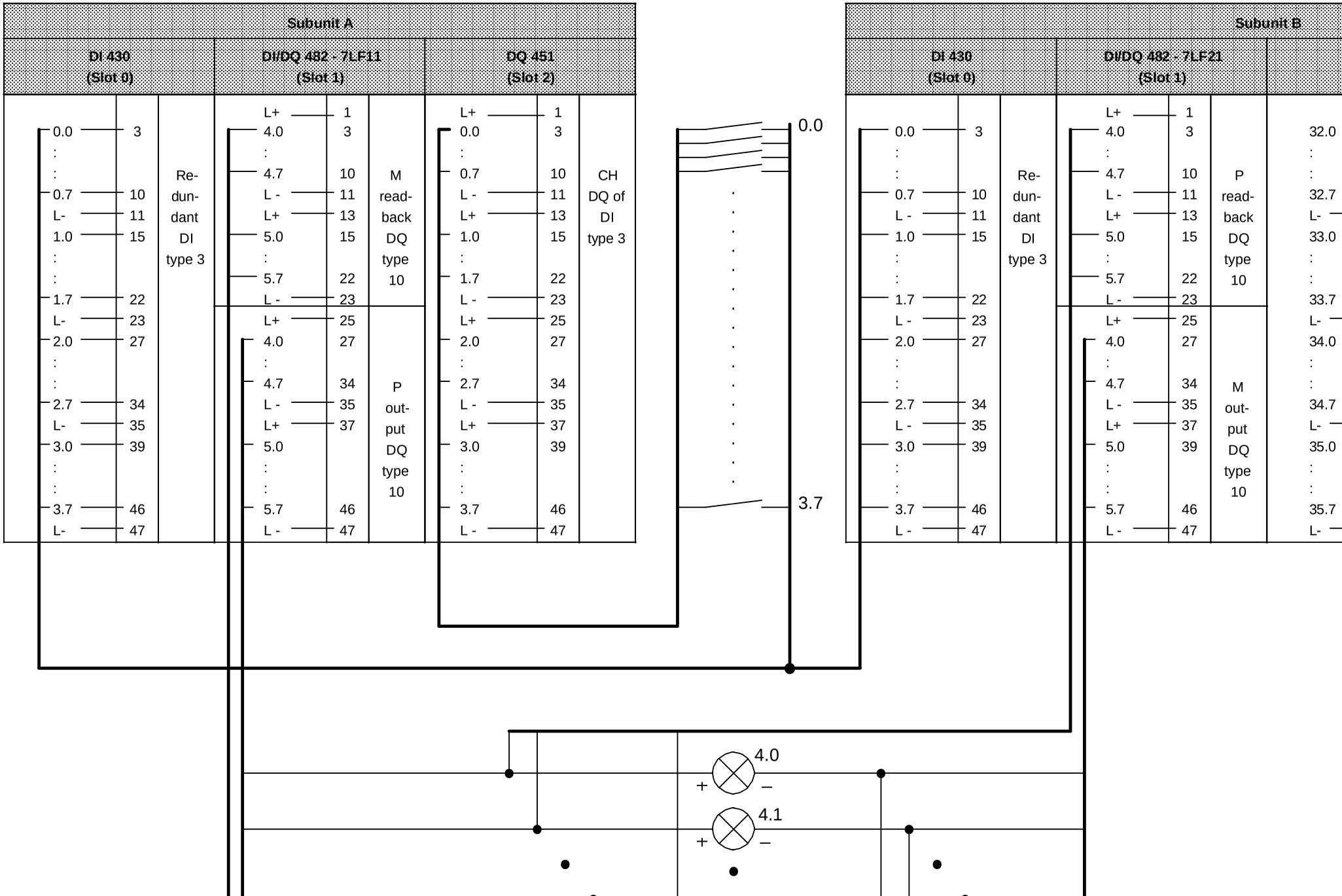


Figure 7-2. Switch and Jumper Settings for Parallel Link


Subunit A
DI 430 (Slot 0)
DI/DQ 482 - 7LF11 (Slot 1)
DQ 451 (Slot 2)
Re-dun-dant DI type 3
M read-back DQ type 10
P out-put DQ type 10
CH DQ of DI type 3
0.0
3.7
Subunit B
DI 430 (Slot 0)
DI/DQ 482 - 7LF21 (Slot 1)
Re-dun-dant DI type 3
P read-back DQ type 10
M out-put DQ type 10
4.0
4.1

7.2 Entering the Subunit ID

Connect the programmer to subunit A
Select the COM 115F Configure package and set the operating mode ”ON” in the Defaults screen form.
Select the ”Initializing the 115F Operating System” screen form
Press **F1** ”Subunit” and read the subunit ID entered there.
Change the subunit ID if required by entering the letter A.
Confirm the entry with the ”Enter” key.
Connect the programmer to the subunit which is to be subunit B.
Press **F1** ”Subunit” and read the subunit ID entered there.
Change the subunit ID if required by entering the letter **B**.
Confirm the entry with the ”Enter” key.
Switch both PS 951 power supplies off and on again.

After entering the subunit ID, connect the programmer to subunit A. Data interchange between the programmer and the S5-115F is only possible if both subunits have received correct subunit IDs.

> **Note**
>
> Every CPU is supplied from the factory with subunit ID B. If you have not changed this ID in the CPUs, you only need to enter ID A for subunit A.

7.3 Initializing the Operating System

Call the ”Initializing the 115F Operating System” form.
The system replies with the standard setting (Figure 7-4)

```
Initializing the 115F Operating System              COM 115F / PDC15

                     0 means not in use

User EPROM capacity             (0; 8; 16; 32):  16         KByte
Max. PLC cycle time              (10 to 16383):  25   *  10 msec
Second error occurrence time        (1 to 255):   9   *  10 min
Calculated test block time                   :  51         sec
Test cycle time                     (1 to 255):   9   *  10 min
Update user time: Max. interval  (2 to 16383):  10   *  10 msec
Interrupt proc.: Max. interval      (2 to 255):   5   *  10 msec
Time interval for OB 13      (0; 10 to 16383):   0   *  10 msec
Short discr. time for non-interr.DI (1 to 63):   3   *  10 msec
Short discr. time interr. DI    (0; 1 to 255):  20         msec
Short discr. time analog input   (0; 2 to 63):   0   *  10 msec
I/O error tolerance                  (1 to 4):   1
DB No for long discr. times      (0; 4 to 255):   4
Min. absolute AI deviation     (0; 16 to 255):  16
Global lower wire-break limit       (norm. KF):   0
Global upper wire-break limit       (norm. KF):   0

| F1      | F2 | F3 | F4 | F5 | F6   | F7 | F8     |
| SUBUNIT |    |    |    |    | SAVE |    | RETURN |
```

Figure 7-4. Initialization Menu of the 115F Operating System for CPU 942-7UF13

Change the following standard values:

```
Second error occurrence time      :  1  *  10 min
Update user time: Max. interval   :  16383  *  10 min (no monitoring)
Interrupt proc.: Max. interval    :  255  *  10 msec (no monitoring)
Short discr. time interr. DI      :  0        msec (not required)
Short discr. time analog input    :  0  *  10 msec (not required)
DB No. for long discr. times      :  0 (no long discrepancy times)
Min. absolute AI deviation        :  0 (not required)
```

Press F6 ”SAVE”. The parameters just set are then stored in the program file.
Terminate initialization of the operating system.

7.4 Configuring the Hardware with COM 115F

This section contains the configuration screen forms for the I/O types mentioned in the example. Configuring hardware with COM 115F means

- Selecting an I/O type for each input or output
- Defining characteristics at these inputs/outputs
- Defining signal groups, check addresses and data blocks for the configuration data.

Configuring the single-channel digital input modules

Call the configuration form for I/O type 1
Select DI word 32 with the cursor control keys
Enter type no. ”1” in the field for bit 32.0 and terminate the input with <Return>
The cursor jumps to the configuration field
Enter the following in the configuration field

```
DB number  (4 to 255):      11
SUBUNIT         (A/B): B
```

Repeat the entries for addresses 32.1 to 35.7
(You can copy the configuration data of bit 32.0 also to addresses 32.1 to 35.7 1.2.2).
Figure 7-5 shows the finished entries.

I/O type 1

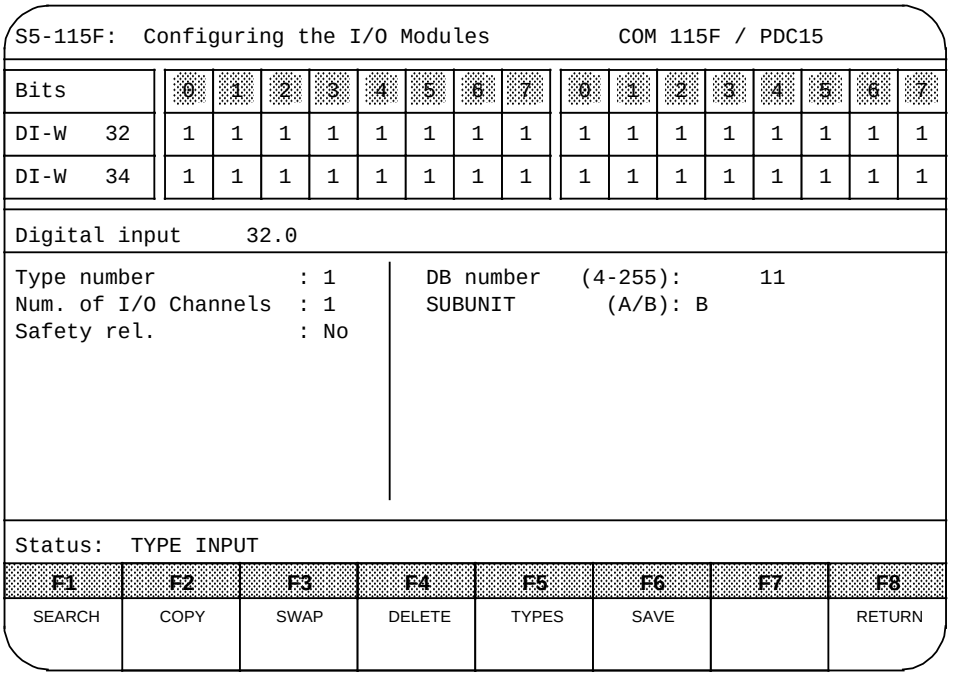


Figure 7-5. Initialization Screen Form for I/O Type 1

Save your configuration data on the default program file by pressing F6
Terminate the input by pressing F8

Configuring the two-channel digital input modules

Call the configuration form for I/O type 3
Select DI word 0 with the cursor control keys
Enter type no. ”3” in the field for bit 0.0 and terminate the input with <Return>
The cursor jumps to the configuration field.
Enter the following in the configuration field

```
DB number        (4 to 255):      13
Signal group      (0 to 28):       0
Interrupt generating (Y/N) : N
Inp. volt.         (24/220):      24  V
Discrepancy time      (S/L): S    30  msec
CH DQ Channel No.     (1/2):       1
CH DQ Byte addr. (0 to 125):       0 SUBUNIT:A
```

Repeat the entries for addresses...
(You can copy the configuration data of bits 0.0 also to addresses 0.1 to 3.7 (1.2.2)).
Figure 7-6 shows the finished entries.

I/O type 3

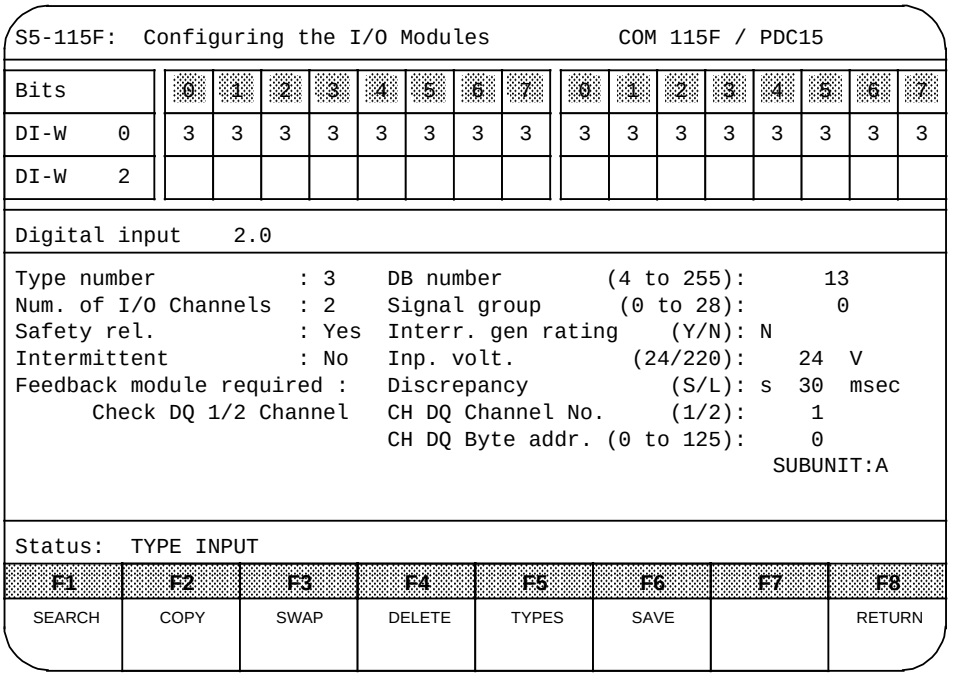


Figure 7-6. Initialization Screen Form for I/O Type 3

Save your configuration data on the default program file by pressing F6
Terminate the input by pressing F8

Configuring the single-channel digital output modules

Call the configuration form for I/O type 8
Select DQ word 36 with the cursor control keys
Enter type no. ”8” in the field for bit 36.0 and terminate the input with <Return>
The cursor jumps to the configuration field
Enter the following in the configuration field

```
DB number  (4 to 255):      18
SUBUNIT         (A/B): A
```

Repeat the entries for addresses...
(You can copy the configuration data of bit 36.0 also to addresses 36.1 to 39.7 1.2.2)

Figure 7-7 shows the finished entries.

I/O type 8

S5-115F: Configuring the I/O Modules COM 115F / PDC15

Bits	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
DQ-W 36	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
DQ-W 38	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8

Digital output 36.0

```
Type number           : 8      DB number  (4 to 255):      18
Num. of I/O channels  : 1      SUBUNIT         (A/B): A
Safety rel.           : No
```

Status: TYPE INPUT

F1	F2	F3	F4	F5	F6	F7	F8
SEARCH	COPY	SWAP	DELETE	TYPES	SAVE		RETURN

Figure 7-7. Initialization Screen Form for I/O Type 8

Save your configuration data on the program file by pressing F6
Terminate the input by pressing F8

Configuring the two-channel digital output modules

Call the configuration form for I/O type 10
Select DQ word 4 with the cursor control keys
Enter type no. ”10” in the field for bit 4.0 and terminate the input with <Return>
The cursor jumps to the configuration field
Enter the following in the configuration field

```
DB number       (4 to 255):      20
Signal group     (0 to 28):      12
R DI byte addr. (0 to 127):      4
```

Repeat the entries for addresses 36.1 to 39.7
(You can copy the configuration data of bit 36.0 also to addresses 36.1 to 39.7 1.2.2).

Figure 7-8 shows the finished entries.

I/O type 10

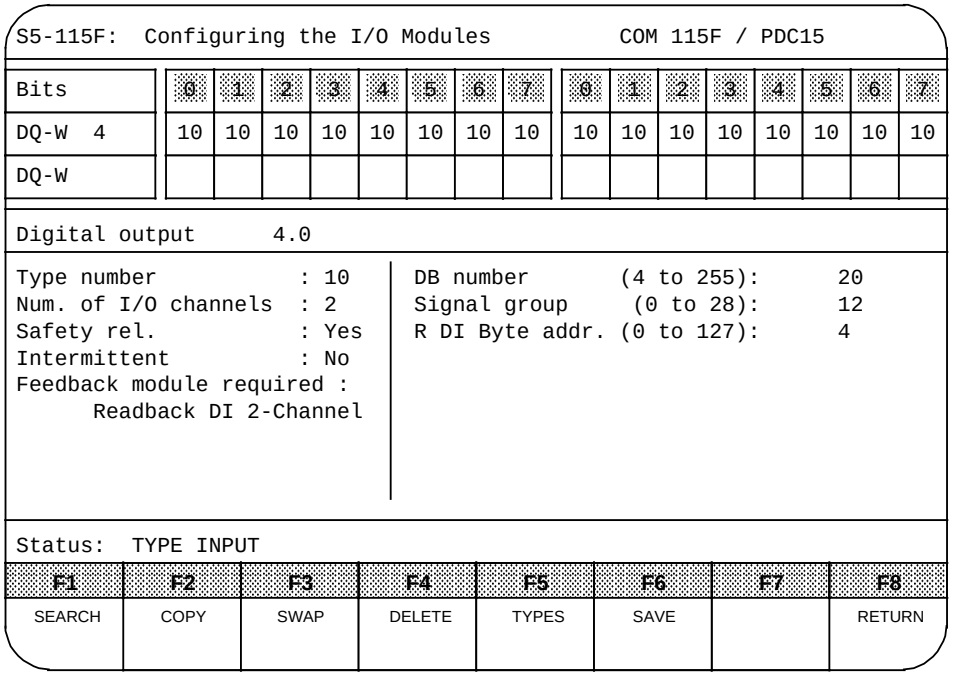


Figure 7-8. Initialization Screen Form for I/O Type 10

Save your configuration data on the default program file by pressing F6
Terminate the input by pressing F8

7.5 Transferring the Configuration Data to the PLC

In test mode, you can transfer the configuration data from the hard disk or the diskette to the PLC. Proceed as follows:

Connect the programmer to subunit A
Call the ”Defaults” screen form and enter the following:

- Name of the program file
- Operating mode ”ON”

Call the ”System handling” screen form and press F3 ”TRAN/LOAD”.
You are now in the ”Transfer/Load I/O” menu.
Press F3 ”LOAD FD”

The configuration is now loaded into the programmer memory from the hard disk or diskette.

Press F2 ”Tran PLC”
Press F1 ”Config. DB”
The configuration is now loaded into the PLC memory from the programmer memory.
You are now in the ”Transfer” menu
Press F8 ”Return” and
terminate the input.

The complete configuration is now in the PLC.
You can now load your control program into the PLC memory and try out the program in test mode.

Appendices

Appendix A . . List of Operations
Appendix B . . . PLC Error Messages
Appendix C . . . SIEMENS Addresses Worldwide

A List of Operations

A.1 Explanatory Notes on the Operands . . . . . A - 1
A.2 Basic Operation Set . . . . . A - 4
A.3 Supplementary Operations . . . . . A - 10
A.4 System Operations . . . . . A - 15
A.5 Evaluation of CC 1 and CC 0 . . . . . A - 16
A.6 Machine Code Listing . . . . . A - 17

A List of Operations

A.1 Explanatory Notes on the Operands

Abbreviations	Explanation
ACCUM 1	Accumulator 1 (during the loading of accumulator 1 any existing contents are shifted into accumulator 2).
ACCUM 2	Accumulator 2
CC 0 / CC 1	Condition code 0 / Condition code 1
CSF	Control system flowchart method of representation in STEP 5.
Formal operand	Expression of up to 4 characters. The first character must be a letter.
LAD	Ladder diagram method of representation in STEP 5.
OV	Overflow. This condition code bit is set if, for example, the numerical range is exceeded during arithmethic operations.
PII	Process input image
PIQ	Process output image
RLO	Result of logic operation
RLO-dependent Y Y / N	The statement is executed only if RLO = ”1”. The statement is executed only on the leading ( ) or trailing ( ) edge of the RLO. The statement is always executed.
RLO affected? Y / N	The RLO is affected / not affected by the operation.
RLO reloaded? Y / N	The RLO is reloaded / not reloaded with the signal state of the parameter used when the next logic operation (but not assignment) is performed.
STL	Statement list method of representation in STEP 5.

Abb.	Explanation	Permissible value range for operands for CPU 942F
C	Counter - with the "Bit test" and "Set" supplementary operations	0 to 127.7 0.0 to 127.7
BF	Byte constant (fixed-point number)	- 128 to + 127
D	Data word (1 bit)	0.0 to 255.15
DB	Data block	4 to 255
DL	Data word (left-hand byte)	0 to 255
DR	Data word (right-hand byte)	0 to 255
DW	Data word	0 to 255
F	Flag	0.0 to 255.7*
FY	Flag byte	0 to 255*
FW	Flag word	0 to 254*
FB	Function block	0 to 255
I	Input	0.0 to 127.7
IB	Input byte	0 to 127
IW	Input word	0 to 126 (digital) 128 to 254 (analog)
KB	Constant (1 byte)	0 to 255
KC	Constant (count)	0 to 999
KF	Constant (fixed-point number)	- 32768 to + 32767
KH	Constant (hexadecimal code)	0 to FFFF
KM	Constant (2-byte bit pattern)	Arbitrary bit pattern (16 bits)
KS	Constant (2 characters)	Any 2 ASCII characters
KT	Constant (time)	0.0 to 999.3
KY	Constant	0 to 255 (as 2 bytes)

Abb.	Explanation	Permissible value range for operands for CPU 942F
OB[2]	Organization block	1 to 255
PB	Program block (with block call and return operations)	0 to 125
PB/ PY[1]	Peripheral (input/output) byte	0 to 124
PW	Peripheral (input/output) word	0 to 126
Q	Output	0 to 125.7
QB	Output byte	0 to 125
QW	Output word	0 to 124 (digital) 128 to 254 (analog)
RS	System data area - in the case of Load operations (supplementary operations) and Transfer operations (system operations) - in the case of Bit Test and Set/Reset operations (system operations)	0 to 255 0.0 to 255.15
SB	Sequence block	0 to 255
T	Timer - with the ”Bit test” and ”Set ” supplementary operations	0 to 127 0.0 to 127.7

* FW 0 is reserved for the logical program counter
FW 2 to FW 198 (F 2.0 to F 199.7) permissible for the user porgram
FW 200 to FW 254 (F 200.0 to F 255.7) are only permissible in blocks as scratch flags. Scratch flags must be defined, set or reset at the start of block execution as well as after conditional and unconditional block calls. This prevents data exchange with other blocks.

1 In the case of programmers with S5-DOS

2 See 2.3.1 for an overview of the organization blocks and their functions.

A.2 Basic Operation Set

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Boolean Logic Operations										
A	•	•	•	•	•	N	Y	N	1.6	Scan operand for ”1” and combine with RLO through logic AND.
AN	•	•	•	•	•	N	Y	N	1.6	Scan operand for ”0” and combine with RLO through logic AND.
O	•	•	•	•	•	N	Y	N	1.6	Scan operand for ”1” and combine with RLO through logic OR.
ON	•	•	•	•	•	N	Y	N	1.6	Scan operand for ”0” and combine with RLO through logic OR.
O						N	Y	Y	1.6	Combine AND operations through logic OR.
A(						N	Y	Y	1.6	Combine expressions enclosed with parentheses (6 levels) through logic AND.
O(						N	Y	Y	1.6	Combine expressions enclosed with parentheses (6 levels) through logic OR.
)						N	Y	Y	1.6	Right parenthesis
Set/Reset Operations										
S	•	•	•			Y	N	Y	1.6	Set operand to ”1”
R	•	•	•			Y	N	Y	1.6	Reset operand to ”0”
=	•	•	•			Y	N	Y	1.6	Assign value of RLO to operand
Load Operations										
L	IB					N	N	N	1.6	Load an input byte from the PII into accumulator 1
L	QB					N	N	N	1.6	Load an input byte from the PIQ into accumulator 1

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected ? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Load Operations (cont.)										
L	IW					N	N	N	1.6	Load an input word from the PII into accumulator 1: byte n ACCUM 1(bits 8 to 15); byte n+1 ACCUM 1(bits 0 to 7)
L	QW					N	N	N	1.6	Load an output word from the PIQ into accumulator 1: byte n ACCUM 1 (bits 8 to 15); byte n+1 ACCUM 1 (bits 0 to 7)
L	PY					N	N	N	1330 *	Load an input byte of the digital/analog inputs from the interrupt/timer PII into accumulator 1
L	PW					N	N	N	1330 *	Load an input byte of the digital/analog inputs from the interrupt/timer PIQ into accumulator 1. Byte n ACCUM 1 (bits 8 to 15); byte n+1 ACCUM 1 (bits 0 to 7)
L	FB					N	N	N	1.6	Load a flag byte into accumulator 1
L	FW					N	N	N	1.6	Load a flag word into accumulator 1:byte n ACCUM 1 (bits 8 to 15); byte n+1 ACCUM 1 (bits 0 to 7)
L	DL					N	N	N	69	Load a data word (left-hand byte) of the current data block into accumulator 1
L	DR					N	N	N	71	Load a data word (right-hand byte) of the current data block into accumulator 1
L	DW					N	N	N	72	Load a data word of the current data block into ACCUM 1: byte n ACCUM 1 (bits 8 to 15); byte n+1 ACCUM 1 (bits 0 to 7)
L	KB					N	N	N	51	Load a constant (1-byte number) into accumulator 1
L	KC					N	N	N	1.6	Load a constant (2 characters in ASCII format) into accumulator 1
L	KF					N	N	N	1.6	Load a constant (fixed-point number) into accumulator 1
L	KH					N	N	N	1.6	Load a constant (hexadecimal code) into accumulator 1
L	KM					N	N	N	1.6	Load a constant (bit pattern) into accumulator 1
L	KY					N	N	N	1.6	Load a constant (2-byte number) into accumulator 1

* In the case of accesses to the 434 interrupt DI module, the time can increase by values up to the short discrepancy time (1.1.1).

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Load Operations (cont.)										
L	KT					N	N	N	1.6	Load a constant (time in BCD) into accumulator 1
L	KZ					N	N	N	1.6	Load a constant (time in BCD) into accumulator 1
L				•	•	N	N	N	1.6	Load a time or count (in binary code) into accumulator 1
LC				•		N	N	N	127	Load times (in BCD) into accumulator 1
LC					•	N	N	N	125	Load counts (in BCD) into accumulator 1
Transfer Operations										
T	IB					N	N	N	1.6	Transfer the contents of accumulator 1 to an input byte (into the PII)
T	QB					N	N	N	1.6	Transfer the contents of accumulator 1 to an output byte (into the PIQ)
T	IW					N	N	N	1.6	Transfer the contents of accumulator 1 to an input word (into the PII): ACCUM 1 (bits 8 to 15) Byte n; ACCUM 1 (bits 0 to 7) byte n+1
T	QW					N	N	N	1.6	Transfer the contents of accumulator 1 to an output word (into the PIQ): ACCUM 1 (bits 8 to 15) Byte n; ACCUM 1 (bits 0 to 7) byte n+1
T	PY					N	N	N	94	Transfer the contents of accumulator 1 to the digital/analog outputs with updating of the PIQ
T	PW					N	N	N	116	Transfer the contents of accumulator 1 to the interrupt/time PIQ of the digital/analog outputs with updating of the PIQ
T	FB					N	N	N	1.6	Transfer the contents of accumulator 1 to a flag byte
T	FW					N	N	N	1.6	Transfer the contents of accumulator 1 to a flag word (into the PIQ): ACCUM 1 (bits 8 to 15) byte n; ACCUM 1 (bits 0 to 7) byte n+1
T	DL					N	N	N	64	Transfer the contents of accumulator 1 to a data word (left-hand byte)
T	DR					N	N	N	66	Transfer the contents of accumulator 1 to a data word (right-hand byte)
T	DW					N	N	N	69	Transfer the contents of accumulator 1 to a data word

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Timer Operations										
SP				•		Y	N	Y	121	Start timer (in accumulator 1) as pulse (signal contracting) on leading edge of the RLO
SE				•		Y	N	Y	121	Start timer (in accumulator 1) as extended pulse (signal contracting and stretching) on leading edge of the RLO
SR				•		Y	N	Y	124	Start timer (in accumulator 1) as on-delay on leading edge of the RLO
SS				•		Y	N	Y	124	Start timer (in accumulator 1) as stored on-delay on leading edge of the RLO
SF				•		Y	N	Y	119	Start timer (in accumulator 1) as off-delay on falling edge of the RLO
R				•		Y	N	Y	81	Reset timer if RLO = ”1”
Counter Operations										
CU					•	Y	N	Y	159	Counter counts up 1 on leading edge of the RLO
CD					•	Y	N	Y	110	Counter counts down 1 on leading edge of the RLO
S					•	Y	N	Y	120	Set counter if RLO = ”1”
R					•	Y	N	Y	133	Reset counter if RLO = ”1”
Arithmetic Operations										
+F						N	N	N	1.6	Add two fixed-point numbers: ACCUM 1 + ACCUM 2; CC1/CC0 are affected
- F						N	N	N	1.6	Subtract two fixed-point numbers: ACCUM 2 - ACCUM 1; CC1/CC0 are affected

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Comparison Operations										
!=F						N	Y	N	1.6	Compare two fixed-point numbers if equal: if ACCUM 1=ACCUM 2, the RLO is ”1”. CC1/CC0 are affected.
><F						N	Y	N	1.6	Compare two fixed-point numbers if not equal: if ACCUM 1 ACCUM 2, the RLO is ”1”. CC1/CC0 are affected.
>F						N	Y	N	1.6	Compare two fixed-point numbers for greater than: if ACCUM 2 > ACCUM 1, the RLO is ”1”. CC1/CC0 are affected.
>=F						N	Y	N	1.6	Compare two fixed-point numbers for greater than or equal to: if ACCUM 2 ACCUM 1, the RLO is ”1”. CC1/CC0 are affected.
<F						N	Y	N	1.6	Compare two fixed-point numbers for less than: if ACCUM 2 < ACCUM 1, the RLO is ”1”. CC1/CC0 are affected.
<=F						N	Y	N	1.6	Compare two fixed-point numbers for less than or equal to: if ACCUM 2 ACCUM 1, the RLO is ”1”. CC1/CC0 are affected.
Block Call Operations										
JU	PY					N	N	Y	152	Unconditional jump to a program block
JU	FY					N	N	Y	154	Unconditional jump to a function block
JU	SB					N	N	Y	152	Unconditional jump to a sequence block
JC	PY					Y	Y [1]	Y	156	Conditional jump to a program block
JC	FY					Y	Y [1]	Y	161	Conditional jump to a function block
JC	SB					Y	Y [1]	Y	160	Conditional jump to a sequence block

1 RLO is set to ”1”

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Block Call Operations (cont.)										
C	DB					N	N	N	66	Call a data block
G	DB					N	N	N	190	Generate a data block
Return Operations										
BE						N	N	Y	99	Block end (termination of a block)
BEC						Y	Y[1]	Y	101	Block end, conditional
BEU						N	N	Y	99	Block end, unconditional (cannot be used in organization blocks)
”No” Operations										
NOP 0						N	N	N	1.6	No operation, all bits = ”0”
NOP 1						N	N	N	1.6	No operation, all bits = ”1”
Stop Operation										
STP						N	N	N	47	Stop: Scanning cycle is completed; STS error identifier in the ISTACK is set
Display Generation Operations										
BLD 130						N	N	N	1.6	Display generation operation for the programmer: Generate blank line by carriage return
BLD 131						N	N	N	1.6	Display generation operation for the programmer: Switch over to statement list
BLD 132						N	N	N	1.6	Display generation operation for the programmer: Switch over to control system flowchart

1 RLO is set to ”1”

☒ for organization blocks (OB)
☒ for program blocks (PB)
☒ for function blocks (FB)
☒ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Display Generation Operations (cont.)										
BLD 133						N	N	N	1.6	Display generation operation for the programmer: Switch over to ladder diagram
BLD 255						N	N	N	1.6	Display generation operation for the programmer: Terminate segment

A.3 Supplementary Operations

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Boolean Logic Operations										
A=	Formal operand					N	Y	N	129	AND operation: Scanning a formal operand for ”1”
	•	•	•	•	•					
AN=	Formal operand					N	Y	N	129	AND operation: Scanning a formal operand for ”0”
	•	•	•	•	•					
O=	Formal operand					N	Y	N	129	OR operation: Scanning a formal operand for ”1”
	•	•	•	•	•					
ON=	Formal operand					N	Y	N	129	OR operation: Scanning a formal operand for ”0”
	•	•	•	•	•					
AW						N	N	N	1.6	Combine contents of ACCUM 2 and ACCUM 1 (word operation) through logic AND: result is stored in ACCUM 1. CC1/CC0 are affected
OW						N	N	N	1.6	Combine contents of ACCUM 2 and ACCUM 1 (word operation) through logic OR: result is stored in ACCUM 1. CC1/CC0 are affected
XOW						N	N	N	1.6	Combine contents of ACCUM 2 and ACCUM 1 (word operation) through EXCLUSIVE-OR: result is stored in ACCUM 1. CC1/CC0 are affected

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Bit Test Operations										
TB				•	•	N	Y	N	154	Test a bit of a timer or counter word for ”1”
TB	D					N	Y	N	154	Test a bit of a data word for ”1”
TB	BS					N	Y	N	152	Test a bit of a data word in the system data area for ”1”
TBN				•	•	N	Y	N	154	Test a bit of a timer or counter word for ”0”
TBN	D					N	Y	N	154	Test a bit of a data word for ”0”
TBN	BS					N	Y	N	153	Test a bit of a data word in the system data area for ”0”
SU				•	•	N	N	Y	155	Set a timer or counter word bit unconditionally
SU	D					N	N	Y	155	Set a data word bit unconditionally
RU				•	•	N	N	Y	155	Reset a timer or counter word bit unconditionally
RU	D					N	N	Y	155	Reset a data word bit unconditionally
Set/Reset Operations										
S=	Formal operand					Y	N	Y	127	Set a formal operand (binary), (with RLO = 1)
	•	•	•							
RB=	Formal operand					Y	N	Y	127	Reset a formal operand (binary), (with RLO = 1)
	•	•	•							
RD=	Formal operand					Y	N	Y	121	Reset a formal operand (digital), (with RLO = 1)
				•	•					
==	Formal operand					Y	N	Y	127	The value of the RLO is assigned to the status of the formal operand (binary)
	•	•	•							

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Timer and Counter Operations										
FR				•	•	Y	N	Y	83	Enable timer/counter for cold restart. If RLO = 1, "FR T" restarts the timer, and "FR C" sets, increments or decrements the counter
FR=	Formal operand			•	•	Y	N	Y	121 *	Enable formal operand (timer/counter) for cold restart. (For detailed description see "FR" operation)
SP=	Formal operand			•		Y	N	Y	121 *	Start timer (formal operand) as pulse with the value stored in ACCUM 1.
SE=	Formal operand			•		Y	N	Y	121 *	Start timer (formal operand) as On-delay with the value stored in ACCUM 1.
SEC=	Formal operand			•	•	Y	N	Y	121 *	Start timer (formal operand) with the value stored in ACCUM 1 as extended pulse or set counter (formal operand) with the count specified.
SSU=	Formal operand			•	•	Y	N	Y	121 *	Start timer (formal operand) with the value stored in ACCUM 1 as stored On-delay or increment a counter (formal operand)
SFD=	Formal operand			•	•	Y	N	Y	121 *	Start timer (formal operand) with the value stored in ACCUM 1 as Off-delay or decrement a counter (formal operand)
Load and Transfer Operations										
L=	Formal operand					N	N	N	127 *	Load the value of the formal operand into ACCUM 1 (Data type: BY, W)
L	BS					N	N	N	66	Load a word from the system data area into ACCUM 1
LC=	Formal operand			•	•	N	N	N	121 *	Load the value of the formal operand in BCD into ACCUM 1
LW=	Formal operand					N	N	N	126	Load the bit pattern of a formal operand into ACCUM 1 (parameter type: D; data type: KF, FH, KM; KY, KS, KT, KC)
T=	Formal operand					N	N	N	128 *	Transfer the contents of ACCUM 1 to the formal operand (data type: BY, W)

***** Plus execution time of the substituted operation

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Conversion Operations										
CFW						N	N	N	1.6	Form the one's complement of ACCUM 1.
CSW						N	N	N	1.6	Form the two's complement of ACCUM 1. CC1/CC2 and OV are affected
Shift Operations										
SLW	Parameter n=0 to 15					N	N	N	1.6	Shift the contents of ACCUM 1 to the left by the value specified in the parameter. Positions becoming vacant are padded with zeros. CC1/CC0 are affected
SRW	Parameter n=0 to 15					N	N	N	1.6	Shift the contents of ACCUM 1 to the right by the value specified in the parameter. Positions becoming vacant are padded with zeros. CC1/CC0 are affected
Jump Operations										
SPA=	Symbolic address max. 4 characters					N	N	N	1.6	Unconditional jump to symbolic address
SPB=	Symbolic address max. 4 characters					Y	Y[1]	Y	1.6	Conditional jump to symbolic address (If the RLO = ”0”, it is set to ”1”)
SPZ=	Symbolic address max. 4 characters					N	N	N	1.6	Jump if the result is zero: the jump is only made if CC1 = 0 and CC0 = 0. The RLO is not changed.
SPN=	Symbolic address max. 4 characters					N	N	N	1.6	Jump if the result is not zero: the jump is only made if CC1 CC0. The RLO is not changed.
SPP=	Symbolic address max. 4 characters					N	N	N	1.6	Jump if the result is greater than zero: the jump is only made if CC1 = 1 and CC0 = 1. The RLO is not changed.
SPM=	Symbolic address max. 4 characters					N	N	N	1.6	Jump if the result is less than zero: the jump is only made if CC1 = 0 and CC0 = 1. The RLO is not changed.
SPO=	Symbolic address max. 4 characters					N	N	N	1.6	Jump on overflow: the jump is only made if the OVERFLOW bit is set. The RLO is not changed.

1 RLO is set to ”1”

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Other Operations										
IA						N	N	N	50	Disable interrupt: input/output interrupt or timer OB processing is disabled
RA						N	N	N	50	Enable interrupt: cancels the effect of IA
D						N	N	N	43	Decrement the low byte (bits 0 to 7) of ACCUM 1 by the value of n (n = 0 to 255)
I						N	N	N	43	Increment the low byte (bits 0 to 7) of ACCUM 1 by the value of n (n = 0 to 255)
DO=	Formal operand					N	N	N	122 *	Process block. (Only C DB, JU PB, JU FB, JU SB can be substituted)
	•	•	•	•	•					
DO	DW **					N	N	N	118	Process data word: the next operation is combined (ORed) with the parameter specified in the data word and executed **.
DO	FW **					N	N	N	102	Process flag word: the next operation is combined (ORed) with the parameter specified in the flag word and executed **.

***** Plus execution time of the substituted operation

****** Permissible operations:

A, AN, O, ON;	L, LC, T;
S, R, =;	JU, JC, JZ, JN, JP, JM, JO, SLW, SRW;
FR T, R T, SF T, S DT , SI T, SS T, SE T, D, I;	
FR C, RC, SC, CD C, CU C	C DB; T RS, TNB

A.4 System Operations

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Set Operations										
SU	BS					N	N	Y	154	Set bit in system data area unconditionally
RU	BS					N	N	Y	154	Reset bit in system data area unconditionally
Load and Transfer Operations										
LIR	0 2					N	N	N	108	Load the register (0: ACCUM 1; 2: ACCUM 2) indirectly with the contents of a memory word
TIR	0 2					N	N	N	92	Transfer the register contents (0: ACCUM 1; 2: ACCUM 2) indirectly into the memory word
TNB	Parameter n=0 to 255					N	N	N	123 +32 Number of bytes	Field transfer byte by byte (number of bytes 0 to 255)
T	RS					N	N	N	61	Transfer a word to the system data area
Block call operations										
JU	OB					N	N	Y	154	Call organization block unconditionally
JC	OB					Y	Y [1]	Y	160	Call organization block conditionally
Arithmetic Operations										
ADD	BF					N	N	N	49	Add byte constant (fixed point) to ACCUM 1
ADD	KF					N	N	N	87	Add fixed-point constant (word) to ACCUM 1

1 RLO is set to ”1”

☐ for organization blocks (OB)
☐ for program blocks (PB)
☒ for function blocks (FB)
☐ for sequence blocks (SB)

Operation (STL)	Operands					1 RLO-depend.? 2 RLO affected? 3 RLO reloaded?			Execution time in µsec.	Function
	I	Q	F	T	C	1	2	3	CPU 942F	
Other Operations										
DI *	Formal operand					N	N	N	128	Process via the formal operand (indirect). The number of the formal operand is in ACCUM 1.
	•	•	•	•	•					
STS						N	N	N	47	Stop operation: Program processing is interrupted immediately after this operation.
TAK						N	N	N	47	Swap the contents of ACCUM 1 and ACCUM 2

***** Do not use DI in the safety-related parts of programs.

A.5 Evaluation of CC 1 and CC 0

CC 1	CC 0	Arithmetic operations	Digital logic operations	Comparison operations	Shift operations	Conversion operations
0	0	Result = 0	Result = 0	ACCUM 2 = ACCUM 1	Shifted bit = 0	-
0	1	Result < 0	-	ACCUM 2 < ACCUM 1	-	Result < 0
1	0	Result > 0	Result 0	ACCUM 2 > ACCUM 1	Shifted bit = 1	Result > 0

A.6 Machine Code Listing

Machine Code								Operation	Operand
B0		B1		B2		B3			
L	R	L	R	L	R	L	R		
0	0	0	0					NOP 0	
0	1	0	0					CFW	
0	2	0_d	0_d					L	T
0	3	0_l	0_l					TNB	
0	4	0_d	0_d					FR	T
0	5	0	0					BEC	
0	6	0_c	0_c					FR=	
0	7	0_c	0_c					A=	
0	8	0	0					IA	
0	8	8	0					RA	
0	9	0	0					CSW	
0	A	0_a	0_a					L	FY
0	B	0_a	0_a					T	FY
0	C	0_d	0_d					LD	T
0	D	0_i	0_i					JO=	
0	E	0_c	0_c					LD=	
0	F	0_c	0_c					0=	
1	0	8	2					BLD	130
1	0	8	3					BLD	131
1	0	8	4					BLD	132
1	0	8	5					BLD	133
1	0	F	F					BLD	255
1	1	0_n	0_n					I	
1	2	0_a	0_a					L	FW
1	3	0_a	0_a					T	FW
1	4	0_d	0_d					SF	T
1	5	0_i	0_i					JP=	
1	6	0_c	0_c					SFD=	
1	7	0_c	0_c					S=	
1	9	0_n	0_n					D	
1	C	0_d	0_d					SE	T
1	D	0_f	0_f					JC	FB

Machine Code								Operation	Operand
B0		B1		B2		B3			
L	R	L	R	L	R	L	R		
1	E	0_c	0_c					=SEC	
1	F	0_c	0_c					==	
2	0	0_f	0_f					C	DB
2	1	2	0					>F	
2	1	4	0					<F	
2	1	6	0					><F	
2	1	8	0					!=F	
2	1	A	0					>=F	
2	1	C	0					<=F	
2	2	0_g	0_g					L	DL
2	3	0_d	0_d					T	DL
2	4	0_c	0_c					SR	T
2	5	0_i	0_i					JM=	
2	6	0_c	0_c					SR=	
2	7	0_c	0_c					AN=	
2	8	0_e	0_e					L	KB
2	A	0_g	0_g					L	DR
2	B	0_g	0_g					T	DR
2	C	0_d	0_d					SS	T
2	D	0_i	0_i					JU=	
2	E	0_c	0_c	0_e	0_e	0_e	0_e	SSU=	
2	F	0_c	0_c	0_e	0_e	0_e	0_e	ON=	
3	0	0	1	0_e	0_e	0_e	0_e	L	KZ
3	0	0	2	0_e	0_e	0_e	0_e	L	KT
3	0	0	4	0_e	0_e	0_e	0_e	L	KF
3	0	1	0	0_e	0_e	0_e	0_e	L	KC
3	0	2	0	0_e	0_e	0_e	0_e	L	KY
3	0	4	0					L	KH
3	0	8	0					L	KM
3	2	0_g	0_g					L	DW
3	3	0_g	0_g					T	DW
3	4	0_d	0_d					SP	T

Machine Code								Operation	Operand
B0		B1		B2		B3			
L	R	L	R	L	R	L	R		
3	5	0_i	0_i					JN=	
3	6	0_c	0_c					SP=	
3	7	0_c	0_c					RB=	
3	C	0_d	0_d					R	T
3	D	0_f	0_f					JU	FB
3	E	0_c	0_c					RD=	
3	F	0_c	0_c					LW=	
4	0	0	0_k					LIR	
4	1	0	0					AW	
4	2	0_o	0_o					L	C
4	4	0_o	0_o					FR	C
4	5	0_i	0_i					JZ=	
4	6	0_c	0_c					L=	
4	8	0	0_k					TIR	
4	9	0	0					OW	
4	A	0_a	0_a					L	IB
4	A	8_a	0_a					L	QB
4	B	0_a	0_a					T	IB
4	B	8_a	0_a					T	QB
4	C	0_o	0_o					LC	C
4	D	0_f	0_f					JC	OB
4	E	0_g	0_g					DO	FW
5	0	0_e	0_e					ADD	BN
5	1	0	0					XOW	
5	2	0_a	0_a					L	IW
5	2	8_a	0_a					L	QW
5	3	0_a	0_a					T	IW
5	3	8_a	0_a					T	QW
5	4	0_o	0_o					CD	C
5	5	0_f	0_f					JC	PY
5	8	0	0	0_e	0_e	0_e	0_e	ADD	KF
5	9	0	0					-F	

Machine Code								Operation	Operand
B0		B1		B2		B3			
L	R	L	R	L	R	L	R		
5	C	0_o	0_o					S	C
5	D	0_f	0_f					JC	SB
6	1	0_h	0_h					SLW	
6	2	0_g	0_g					L	RS
6	3	0_g	0_g					T	RS
6	5	0	0					BE	
6	5	0	1					BEU	
6	6	0_c	0_c					T=	
6	9	0_h	0_h					SRW	
6	C	0_o	0_o					CU	C
6	D	0_f	0_f					JU	OB
6	E	0_g	0_g					DO	DW
7	0	0	0					STS	
7	0	0	2					TAK	
7	0	0	3					STP	
7	0	0	B	0_m	0_m	0_m	0_m	JUR	
7	0	1	5	C	0	0_o	0_o	TB	C
7	0	1	5	8	0	0_o	0_o	TBN	C
7	0	1	5	4	0	0_o	0_o	SU	C
7	0	1	5	0	0	0_o	0_o	RU	C
7	0	2	5	C	0	0_d	0_d	TB	T
7	0	2	5	8	0	0_d	0_d	TBN	T
7	0	2	5	4	0	0_d	0_d	SU	T
7	0	2	5	0	0	0_d	0_d	RU	T
7	0	4	6	C	0_b	0_g	0_g	TB	D
7	0	4	6	8	0_b	0_g	0_g	TBN	D
7	0	4	6	4	0_b	0_g	0_g	SU	D
7	0	4	6	0	0_b	0_g	0_g	RU	D
7	0	5	7	C	0_b	0_g	0_g	TB	RS
7	0	5	7	8	0_b	0_g	0_g	TBN	RS
7	0	5	7	4	0_b	0_g	0_g	SU	RS
7	0	5	7	0	0_b	0_g	0_g	RU	RS

Machine Code								Operation	Operand
B0		B1		B2		B3			
L	R	L	R	L	R	L	R		
7	2	0_d	0_d					L	PY
7	3	0_d	0_d					T	PY
7	5	0_f	0_f					JU	PY
7	6	0_c	0_c					DO=	
7	8	0	5	0	0	0_f	0_f	G	DB
7	9	0	0					+F	
7	A	0_a	0_a					L	PW
7	B	0_a	0_a					T	PW
7	C	0_o	0_o					R	C
7	D	0_f	0_f					JU	SB
7	E	0	0					DI	
8	0_b	0_a	0_a					U	M
8	8_b	0_a	0_a					O	M
9	0_b	0_a	0_a					S	M
9	8_b	0_a	0_a					=	M
A	0_b	0_a	0_a					AN	M
A	8_b	0_a	0_a					ON	M
B	0_b	0_a	0_a					R	M
B	8	0_o	0_o					A	C
B	9	0_o	0_o					O	C
B	A	0	0					A(	
B	B	0	0					O(	
B	C	0_o	0_o					AN	C

Machine Code								Operation	Operand
B0		B1		B2		B3			
L	R	L	R	L	R	L	R		
B	D	0_o	0_o					ON	C
B	F	0	0					)	
C	0_b	0_a	0_a					A	I
C	0_b	8_a	0_a					A	Q
C	8_b	0_a	0_a					O	I
C	8_b	8_a	0_a					O	Q
D	0_b	0_a	0_a					S	I
D	0_b	8_a	0_a					S	Q
D	8_b	0_a	0_a					=	I
D	8_b	8_a	0_a					=	Q
E	0_b	0_a	0_a					AN	I
E	0_b	8_a	0_a					AN	Q
E	8_b	0_a	0_a					ON	I
E	8_b	8_a	0_a					ON	Q
F	0_b	0_a	0_a					R	I
F	0_b	8_a	0_a					R	Q
F	8	0_d	0_d					A	T
F	9	0_d	0_d					O	T
F	A	0_i	0_i					JC=	
F	B	0	0					O	
F	C	0_d	0_d					AN	T
F	D	0_d	0_d					ON	T
F	F	F	F					NOP	1

Explanation of Indices

a + byte address
b + bit address
c + parameter address
d + counter address
e + constant
f + block number
g + word address
h + number of shifts
i + relative jump destination address
k + register address
l + length in bytes
m + jump displacement (16 bits)
n + value
o + counter number

B PLC Error Messages

B.1 PLC Error Messages with COM 115F . . . . . B - 1
B.2 Evaluation of Error DBs (DB2 and DB3) without COM 115F . . . . . B - 88

B PLC Error Messages

B.1 PLC Error Messages with COM 115F

Appendix B gives an overview of all error messages and indicates the possible causes of error.

Proceed as follows for the OVERALL RESET remedy:
- Safety mode: OVERALL RESET
- Test mode: - OVERALL RESET
  - Load system
  - STOP
  - RUN

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
02	01	User times cannot be updated: PL or CPU error	Error group No. 20 - PL (Parallel link) defective - CPU defective (in A or B)	- Replace PL - Replace CPU
		Additional info. 1: User time decrement in A/10msec. Additional info. 2: User time decrement in B/10msec.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
02	02	Interval between 2 FB 254s for updating time > configured limit	Error group No. 14 Configured error: Time difference between 2 synchr. FB calls is too long (Parameter: SYNC KM 0000001)	Select intervals between 2 synchr. calls max. configured time difference
		Additional info. 1: Unit time difference/10msec. Additional info. 2: Configured limit value/10msec. Additional info 3: Localization parameter	Uniform actual time interval between 2 FB 254s in 10 msec., decimal display Max. configured time interval between 2 FB 254s in 10 msec., decimal display FB 254 call intervals with the relevant localization parameter too long Hex display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
02	03	Too high a quartz frequency difference between A and B: CPU error	Error group No. 20 - More than a 10 % difference between the CPU quartz frequencies in A and B	- Replace CPU
		Additional info. 1: Quartz monitoring counter A Additional info. 2: Quartz monitoring counter B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
02	04	User times cannot be updated: PL or CPU error	Error group No. 20 More than 30 msec. difference between the sum of the user time decrements of a PC cycle in A and B: - PL error - CPU error	- Replace PL - Replace CPU - Service
		Additional info. 1: Sum in A /10msec. Additional info. 2: Sum in B /10msec.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
04	01	OB 13 calls in A and B not synchronous: CPU HW error	Error group No. 01 Too great an OB13 timer location value difference between A and B - CPU HW error	- Replace CPU - Service
		Additional info. 1: OB13 timer location value A /10msec. Additional info. 2: OB13 timer location value B /10msec.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
04	02	Time difference between 2 FB 254s for interrupt synchronization > configured limit	Error group No. 14 Configuration error: Time difference between 2 synchronization calls for interrupt synchronization is greater than the max. configured time difference (Parameter: SYNC KM 00000010)	Reduce intervals between both calls for interrupt synchronization (< max. configured time difference)
		Additional info. 1: Actual time interval/10msec. Additional info. 2: Max. configured time interval/10 msec. Additional info. 3: Localization parameter	Uniform actual time interval between 2 FB 254 calls in user program in 10 msec., decimal output Configured time interval between 2 FB 254 calls in the user program in 10 msec., decimal output FB 254 call intervals with the relevant localization parameter too long Hex display	
				Warm restart: 1. Stop/Run
04	03	A+B differ in time interval between 2 FB 254s: CPU/PK error	Error group No. 20 - CPU defective - PL defective	- Replace CPU - Replace PL
		Additional info. 1: Time difference in A/10msec. Additional info. 2: Time difference in B/10msec.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
14	01	Operating modes in A and B not defined: CPU error	Error group No. 13 Operating modes of A and B (entered with CPU frontplate switches) wrongly entered: - CPU error	- Replace CPU - Service
		Additional info. 1: Operating mode requests A and B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
16	01	DI modules in A and B unequal, discrepancy time-out	Error group No. 09 I/O module single comparison error Error group No. 08 I/O individual error (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) - Configured discrepancy too short - Too great a difference between sensors switching - Sensor for relevant DI module defective in case of 2-channel sensors - I/O error DI/CH DQ module or DQ/RB DI module defective - Wiring fault	 - Increase configured discrepancy time - Check sensors - Replace sensors - Replace module; either I/O or feedback module defective; if no subunit message present, either A or B is defective - Check wiring
		e.g. Additional info. 1: DI byte No. Additional info. 2: CH DQ byte No. Additional info. 3: Bit No.	 Byte No. for I/O module, decimal display Byte No. for CH DQ module decimal display Bit No. of the discovered error, decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode): 1. Stop / Run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode 1. POWER OFF/ON 2. Stop / Run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
16	04	Long discrepancy time for current DI bit not found: Configuration/CPU error	Error group No. 13 - Configuration: * Long discrepancy time in discrep. DB not configured * Discrepancy DB not present in PLC - CPU error	- Check configuration - Replace CPU - Service
		Additional info. 1: DI byte No. Additional info. 2: DI bit No.	Byte address of the current DI bits, decimal display Bit No. of the current DI bits, decimal display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
16	05	QVZ during DI access: I/O module error	Error group No. 19 - DI module defective - IM306 defective - IM304/314 defective	- Replace module - Replace module - Replace module
		Additional info. 1: DI word No.	Address of the accessed DI module	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
17	01	QVZ during reading of the AI module in self-test: I/O module error	Error group No. 19 - AI channel or AI module defective - IM306 defective - IM304/314 defective	- Replace module - Replace module - Replace module
		Additional info. 1: AI word No.	Channel address of the AI module which was to be read, decimal display	
				Warm restart: 1. Stop/Run
17	02	Overflow in safety-related AI: I/O error	Error group No. 09 I/O module single comparison error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) AI comparison error: - Value read in for AI 463 50% more than upper limit of nominal range	- Check sensor and range selection
		e.g. Additional info 1: AI word No. Additional info 2: CH AQ word No. Additional info 3: CH DQ word No.	Decimal display Check AQ word No. only with subunit ID, decimal display Check DQ word No. only with subunit ID, decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode 1. POWER OFF/ON 2. Stop/Run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
17	04	Wire-break in safety-related AI: I/O error	Error group No. 09 I/O module single comparison error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) AI comparison error: - Cable to AI 463 module interrupted in the case of channel types 3 to 6 - V_{min} selected too high for AI 463 and channel types 5, 6	- Check cables - Reconfigure V_{min}
		e.g. Additional info 1: AI word No. Additional info 2: CH AQ word No. Additional info 3: CH DQ word No.	Decimal display Check AQ word No. only with subunit ID, decimal display Check DQ word No. only with subunit ID, decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode) 1. Stop/Run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode 1. Power OFF/ON 2. Stop/Run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Causes	Remedy
17	06	AI modules in A and B unequal: I/O error	Error group No. 09 I/O module single comparison error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) AI comparison error: - Two-channel sensor * 1 sensor defective * AI module defective * Window too small * Discrepancy time too short * Sensors respond at different times (max. differences >discrepancy time) - Single-channel sensor * AI module defective * Window too small * Discrepancy time too short	* Replace sensor * Replace module * Configure larger window * Increase discrepancy time * Synchronize sensor response (Difference Discrepancy time) * Replace module * Configure larger window * Increase discrepancy time
		e.g. Additional info 1: AI word No. Additional info 3: CH DQ word No.	Decimal display Check DQ word No. with subunit ID, decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode): 1. Stop/run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode 1. POWER ON/OFF 2. Stop/run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER ON/OFF 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
18	01	Process output image (PIQ) in A and B unequal: CPU or PL error	Error group No. 20 - CPU error (PIQ is overwritten by e.g. PG, SINEC) - PL HW error	- Replace CPU - Replace PL - Service
		Additional info. 1: DQ word No. Additional info. 2: DQ word in A Additional info. 3: DQ word in B	Number of the word in which different PIQ contents in A and B were found, decimal display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
18	02	Counter in A and B not identical: CPU or PL error	Error group No. 20 - CPU error (PIQ is overwritten by e.g. PG, SINEC) - PL HW error	- Replace CPU - Replace PL - Service
		Additional info. 1: Counter No. Additional info. 2: Counter in word A Additional info. 3: DQ word in B	Number of the counter in which different contents in A and B were found Decimal display Display of counter contents in hexadecimal form Display of counter contents in hexadecimal form	
				Warm restart 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
23	01	Inconsistencies during test read of SINEC L1 receive buffer: CPU error	Error group No. 01 - CPU error	- Replace CPU
		Additional info. 1: Message source No.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
23	02	Inconsistent definition of Send mailbox: CPU error	Error group No. 01 - Falsification of Send mailbox definition by CPU, HW or operating system error	- Service
		Additional info. 1 Message destination No.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
23	03	Receive mailbox could not be overwritten: CPU error	Error group No. 01 - Falsification of Receive mailbox definition by CPU, HW or operating system error	- Service
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
25	01	Wrong initial address RAM area: CPU error	Error group No. 13 - CPU error	- Replace CPU - Service
		Additional info. 1: SD 36	Initial address of RAM area from system data, hex display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
25	02	No. of loaded DB <> configured programmer operator DB No.: handling error	Error group No. 16	Change PG entry DB number or read in DB
		Additional info. 1: Operated DB No. Additional info. 2: DB address	DB number of the loaded PG entry DB, decimal display RAM address of the loaded PG entry DB, hex display	
				Warm restart: 1. Stop/RUN
25	03	Parameter operator DB configured and not loaded	Error group No. 16	Generate parameter entry DB and load into PC
		Additional info 1: DB No. Additional info 2: DB address	DB number of PG operator DB called, decimal display RAM address of the PG entry DB called, hex display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
26	01	Error during RAM comparison of A and B: CPU error	Error group No. 15 - Memory location defective - One-sided overwrite of memory location by the 115F operating system	- Replace CPU - Service
		Additional info. 1: RAM address Additional info. 2: RAM byte in A Additional info. 3: RAM byte in B	RAM address with different contents in A and B, hex display Contents of the above-named RAM location in A, hex display Contents of the above-named RAM location in B, hex display	
				Warm restart: 1. OVERALL RESET
26	02	Error during RAM test: CPU error	Error group No. 13 - Wrong initial address of the current RAM test signal	- Replace CPU - Service
		Additional info. 1: Initial address test area		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
26	03	Error during RAM test: CPU error	Error group No. 07 Falsification of inverse double data (non-inverse together) - CPU error - Overwriting of the original or inverse data item by the 115F operating system	- Replace CPU - Service
		Additional info. 1: RAM address inverse location Additional info. 2: Contents of original location Additional info. 3: Contents of inverse location		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
26	04	Error during comparison µP 8032 RAM and EPROM: CPU error	Error group No. 13 - SINEC L1 structure identifier FIR bus contradicts configured SINEC L1 structure	- Replace module
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
27	01	Error during PL self-test: PL error	Error group No. 04 PL error during read-after-write-check of test section of current check background: - PL defective - CPU HW error	- Replace PL - Replace CPU
		Additional info. 1: Page No. Additional info. 2: Memory address Additional info. 3: Current test background		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
27	02	Error during PL self-test: PL error	Error group No. 04 PL error during read-after-write-check of the page byte to be tested - PL defective - CPU HW error	- Replace PL - Replace CPU
		Additional info. 1: Page No. Additional info. 2: Memory address Additional info. 3: Current test background Additional info. 4: Current check pattern		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
27	03	Error during PL self-test: CPU error	Error group No. 13 Wrong initial address of current test section: - CPU error	- Replace CPU
		Additional info. 1: Initial address		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
28	01	Number of configuration DB for type 3 I/O or 10 < 4: Configuration error	Error group No. 16	Change No. of DB
		Additional info. 1: I/O type	I/O type of wrong config. DB, decimal display	
				Warm restart: 1. Stop/Run
28	02	Error during self-test of DI modules , type 3 I/O, in A and B: I/O module error	Error group No. 10 Self-test individual error Error groups No. 8 I/O error (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) DI module test error: - I or CH DQ modules defective - Wiring error	- Change module; either I/O or feedback module defective; if there is no subunit message, either A or B defective. - Check wiring
		Additional info. 1: DI or DQ byte No. Additional info. 2: CH DQ byte No. Additional info. 3: Bit No.	Byte No. for DI module: decimal display; Byte No. for check DQ module decimal display Decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode 1. POWER OFF/ON 2. Stop/Run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
28	03	Error during self-test of I/O type 3 DI modules in A: I/O module error	see consec. No. 02	see consec. No. 02
		Additional info: Compare consec. No. 02		
28	04	Error during self-test of I/O type 3 DI modules in B: I/O module error	see consec. No. 02	see consec. No. 02
		Additional info: Compare consec. No. 02		
28	05	Error during DQ test in A and B; restart. I/O type 9, 10; Cycle: type 10	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) DQ I/O test error: - DQ or readback DI module defective - Wiring fault	- Replace module; either DQ or readback DI module defective; if there is no subunit message, either A or B defective - Check wiring
		Additional info 1 DQ byte No. Additional info 2 R DI byte No. Additional info 3 Bit no.	Byte No. for DQ module: Decimal display Byte No. for readback DI module: Decimal display Decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error and signal group No.1 to 28 (passivation): a) Test mode 1. POWER OFF/ON 2. Stop/Run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
28	06	Error during DQ test in A; restart: I/O type 9, 10; Cycle: type 10	see consec. No. 05	see consec. No. 05
		Additional info: Compare consec. No. 05		
28	07	Error during DQ test in B; restart: I/O type 9, 10; Cycle: type 10	see consec. No. 05	see consec. No. 05
		Additional info: Compare consec. No. 05		
28	08	QVZ during self-test of I/O type 3 DI modules I/O module error	Error group No. 19 - DI module defective - DI module not available - IM306 defective - Module wrongly addressed with IM 306	- Replace module - Plug in module (configure if necessary) - Replace IM306 - Correct addressing in IM306
		Additional info. 1: DI word No.	Word No. of erroneous DI modules, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
28	09	QVZ during DQ test Restart: I/O type 9 and 10: Cycle: I/O type 10	Error group No. 19 - DQ module defective - DQ module not available - IM306 defective - DQ module wrongly addressed with IM 306	- Replace module - Plug in module change configuration if necessary) - Replace IM306 - Correct addressing in IM306
		Additional info. 1: DQ word No.	Byte address of wrong DQs, decimal display	
				Warm restart: 1. Stop/Run
28	10	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF11 and CPU 942-7UF12 with I/O ETV 3) Interrupt DI test error:: - Test bit not set in A: * DI 434 module defective in A * CH DQ module defective;	* Replace module * Replace module
		Additional info 1: DI byte No. Additional info 2: CH DQ byte No. Additional info 3: Bit No.	Byte address of the interrupt module, decimal display Byte address of the check DQ, decimal display Decimal display	
				Warm restart in the case of individual error and signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode 1. POWER OFF/ON 2. Stop/Run b) Safety mode 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
28	11	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: - Test bit not set in B: * DI 434 module defective in B * CH DQ module defective;	 * Replace module * Replace module
		Additional info: See consec No. 10		Warm restart: See consec. No. 10
28	12	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Test bit not set in A and B: - DI 434 module defective in A or B - CH DQ module defective - No interrupt register read in OB 2	 - Replace module - Replace module - Read interrupt register in OB 2
		Additional info: See consec. No. 10		Warm restart: See consec. No. 10
28	13	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Interrupt bit not set by ISR in A: - Process interrupts occur in sub-unit A during startup in safety mode - DI 434 module defective in A - CH DQ module defective	 - Do not permit process interrupts during startup (configuration rule) - Replace module - Replace module
		Additional info: See consec. No. 10		Warm restart: See consec. No. 10

Error Identifier Progr.	Consec. No..	Error Message Text	Possible Cause	Remedy
28	14	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 8 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Interrupt bit not set by ISR in B: - Process interrupts occur in sub-unit B during startup in safety mode - DI 434 module defective in B - CH DQ module defective	- Do not permit process interrupts during startup (configuration rule) - Replace module - Replace module
		Additional info: See consec. No. 10		Warm restart: See consec. No. 10
28	15	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Interrupt bit not set by ISR in A: - Process interrupts occur in sub-unit A during startup in safety mode - DI 434 module defective in A - CH DQ module defective	- Do not permit process interrupts during startup (configuration rule) - Replace module - Replace module
		Additional info: See consec. No. 10		Warm restart: See consec. No. 10
28	16	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Test bit in interupt register not set in A: - DI434 module defective - CH DQ module defective - Wiring fault	- Replace module - Replace module - Check wiring
		Additional info: See consec. No. 10		Warm restart: See consec. No. 10

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
28	17	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Test bit in interrupt register not set in B: - DI 434 module defective - CH DQ module defective - Wiring fault	 - Replace module - Replace module - Check wiring
		Additional info.: See consec. No. 10		Warm restart: See consec. No. 10
28	18	Error during interrupt module test	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI test error: Test bit in interrupt register not set in B: - DI 434 module defective - CH DQ module defective - Wiring fault	 - Replace module - Replace module - Check wiring
		Additional info. 1: DI byte No. Additional info. 2: CH DQ byte No. Additional info. 3: Bit No.	Byte address of the interrupt module, decimal display Byte address of the check DQ, decimal display Decimal display	
				Warm restart: See consec. No. 10

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
29	01	User operation sequence for LPLZ incrementation is missing: Program error	Error group No. 12 Operation sequence: L FWO L KF+1 +F T FWO is missing	Insert operation sequence in the relevant block
		Additional info. 1: Block type Additional info. 2: Block No. Additional info. 3: User memory initial address	Type of block in which the LPLZ incrementation is missing, character display Number of the block, decimal display Initial address of the block in the user memory, hex display	
				Warm restart: 1. Stop/Run
29	02	Error during function test of the SSP: CPU HW error	Error group No. 03 SAZ test has detected error in SSP: - CPU HW error	- Replace CPU
		Additional info. 1: Block type Additional info. 2: Block No. Additional info. 3: User memory initial address		
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
30	01	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test components (FXTEKO) in A and B not identical - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTEKO in A Additional info. 2: FXTEKO in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	02	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test pointer 0 (FXTE_ZEI0_H) is not identical in A and B. - CPU error (single-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTE_ZEI0_H in A Additional info. 2: FXTE_ZEI0_H in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	03	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test pointer 0 (FXTE_ZEI0_H) is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTE_ZEI0_L in A Additional info. 2: FXTE_ZEI0_L in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
30	04	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test pointer 0 (FXTE_ZEI0_H) is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTE_ZEI1_H in A Additional info. 2: FXTE_ZEI1_H in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	05	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test pointer 1 (FXTE_ZEI1_L) is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTE_ZEI1_L in A Additional info. 2: FXTE_ZEI1_L in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	06	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test pointer 2 (FXTE_ZEI2_H) is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTE_ZEI2_H in A Additional info. 2: FXTE_ZEI2_H in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Causes	Remedy
30	07	Self-test in A and B not synchronous: CPU error	Error group No. 13 Test pointer 2 (FXTE_ZEI2_L) is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXTE_ZEI2_L in A Additional info. 2: FXTE_ZEI2_L in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	08	Self-test in A and B not synchronous: CPU error	Error group No. 13 One subunit has completed processing of test components in the PC self-test (FITEKO_OK = 1), while the other subunit is still processing test components (FITEKO_OK = 0). - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FITEKO_OK in A Additional info. 2: FITEKO_OK in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	09	Self-test in A and B not synchronous: CPU error	Error group No. 13 The FXAE_BLOCK_ADR_H datum is not identical in A and B - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXAE_BLOCK_ADR_H in A Additional info. 2: FXAE_BLOCK_ADR_H in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
30	10	Self-test in A and B not synchronous: CPU error	Error group No. 13 The FXAE_BLOCK_ADR_L datum is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXAE_BLOCK_ADR_L in A Additional info. 2: FXAE_BLOCK_ADR_L in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	11	Self-test in A and B not synchronous: CPU error	Error group No. 13 The FXAE_DB_END_ADR_H datum is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXAE_DB_END_ADR_H in A Additional info. 2: FXAE_DB_END_ADR_H in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
30	12	Self-test in A and B not synchronous: CPU error	Error group No. 13 The FXAE_DB_END_ADR_L datum is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXAE_DB_END_ADR_L in A Additional info. 2: FXAE_DB_END_ADR_L in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
30	13	Self-test in A and B not synchronous: CPU error	Error group No. 13 The FXAE_TEST_ANZ datum is not identical in A and B. - CPU error (one-sided overwrite of memory location)	- Replace CPU - Service
		Additional info. 1: FXAE_TEST_ANZ in A Additional info. 2: FXAE_TEST_ANZ in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
31	01	Error in other subunit: Partner or CPU error	Error group No. 23	- Analyze error message in partner PLC
		Additional info. 1: Subunit of the original mess.	Subunit which discovered the error	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
31	02	Contradictory error message: CPU error	Error group No. 29 Error messages in A and B are different	
		Additional info. 1: Contents of error DB byte in A Additional info. 2: Contents of error DB byte in B Additional info. 3: Byte No. in error DB incl. header		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run If error not corrected after repeated POWER OFF/ON: - Service
31	03	Contradictory error message: CPU error	Error group No. 29 Signal group No. in error message does not match (not ident. 0 to 28)	
		Additional info. 1: Signal group No.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run If error not corrected after repeated POWER OFF/ON: - Service
31	04	Contradictory error message: CPU error	Error group No. 29 Error identifier byte does not match; Error cannot be evaluated	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run If error not corrected after repeated POWER OFF/ON: - Service

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
31	05	Stack too deep: Handling error	Error group No. 24 Execution times for OB 2 and OB 13 too long	- Check OB 2 and OB 13 - Replace CPU
		Additional info 1: Current stack Additional info 2: maximum permissible stack	Decimal display Decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
32	01	QVZ during writing of CH DQ module: I/O module error	Error group No. 19 Event-driven DI module test: - CH DQ module not available - CH DQ module defective - CH DQ module in IM 306 wrongly addressed - IM306 defective	- Plug in module (if necessary, change configuration) - Replace module - Check addressing - Replace IM306
		Additional info. 1: CH DQ No.	Word No. of the erroneous module, decimal display	
				Warm restart: 1. Stop/Run
32	02	QVZ during reading of DI module: I/O module error	Error group No. 19 Event-driven DI module test: - DI module not available - DI module defective - DI module in IM306 wrongly addressed - IM306 defective	- Plug in module (if necessary, change configuration) - Replace module - Check addressing - Replace IM306
		Additional info. 1: DI word No.	Word No. of the erroneous module, decimal display	
				Warm restart: 1. Stop/Run
32	03	CPU error	Error group No. 13 Event-driven DI module test: Wrong type identifier in the reference table - CPU error	- Replace CPU - Service
		Additional info. 1: DI word No.	Word No. with wrong type identifier, decimal display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
32	04	Error during DI module test in A and B	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) DI module test error Event-driven DI module test: - DI module defective - CH DQ module defective - Wiring error	 - Replace module - Replace module - Check wiring
		Additional info. 1: DI byte No. Additional info. 2: CH DQ byte No. Additional info. 3: Bit No.	Byte No. of the DI module which could contain an error, decimal display Byte No. of the relevant CH DQ byte which could contain an error, decimal display Decimal display	
				Warm restart in the case of individual error in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
32	05	Error during DI module test in A	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case CPU 942-7UF13 with I/O ETV 3 or 4) DI module test error: Event-driven DI module test in subunit A: - DI module defective - CH DQ module defective - Wiring error	- Replace module - Replace module - Check wiring
		Additional info. 1: DI byte No. Additional info. 2: CH DQ No. Additional info. 3: Bit No.	Byte No. of the DI module which could contain an error, decimal display Byte No. of the relevant CH DQ byte which could contain an error, decimal display Decimal display	
				Warm restart in the case of individual error in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
32	06	Error during DI module test in B	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) DI module test error: Event-driven DI module test in subunit B: - DI module defective - CH DQ module defective - Wiring error	- Replace module - Replace module - Check wiring
		Additional info. 1: DI byte No. Additional info. 2: CH DQ No. Additional info. 3: Bit No.	Byte No. of the DI module which could contain an error, decimal display Byte No. of the relevant CH DQ byte which could contain an error, decimal display Decimal display	
				Warm restart in the case of individual error in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
33	01	QVZ during reading of AI module: I/O module error	Error group No. 19 - AI module not available - AI module defective - IM306 defective - AI module with IM306 wrongly addressed	- Plug in module (if necessary, change configuration) - Replace module - Replace IM306 - Check addressing
		Additional info. 1: AI word No.	Address of the AI channel which contained QVZ, decimal display	
				Warm restart: 1. Stop/Run
33	02	QVZ during output of CH relay DQ: I/O module error	Error group No. 19 - Module not available - Module defective - IM306 defective - Module with IM306 wrongly addressed	- Plug in module (if necessary, change configuration) - Replace module - Replace IM306 - Check addressing
		Additional info. 1: CH relay DQ word No.	Address of CH relay DQ module which contained QVZ, decimal display	
				Warm restart: 1. Stop/Run
33	03	QVZ during output of CH AQ: I/O module error	Error group No. 19 - Module not available - Module defective - IM306 defective - Module with IM306 wrongly addressed	- Plug in module (if necessary, change configuration) - Replace module - Replace IM306 - Check addressing
		Additional info. 1: CH AQ word No.	Address of the CH AQ channel which contained the QVZ, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
33	04	Error during AI module test: I/O module error	Error group No. 10 I/O module test individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) DI module test error: Comparison error when injecting check value via CH AQ module: - AI module defective - AQ module defective - CH Rel DQ module defective - Wiring error	- Replace module - Replace module - Replace module - Check wiring
		Additional info. 1: AI word No. Additional info. 2: CH AQ word No. Additional info. 3: CH DQ word No.	Address of the AI channel in which the error occurred, decimal display Address of the relevant CH AQ channel, decimal display Address of the relevant CH relay DQ word, decimal display	
				Warm restart in the case of individual error in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above. Warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
45	01	Error during RAM test: CPU HW error	Error group No. 07 Error during read-after-write check of current test section on test background: - CPU HW error	- Replace CPU - Service
		Additional info. 1: RAM address Additional info. 2: Setpoint test background Additional info. 3: Actual test background		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
45	02	Error during RAM test: CPU HW error	Error group No. 07 Error during read-after-write check of current test section on test background: - CPU HW error	- Replace CPU - Service
		Additional info. 1: RAM address Additional info. 2: Test background Additional info. 3: Setpoint check pattern Additional info. 4: Actual check pattern		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
46	01	Error during processor function test: CPU HW error	Error group No. 02 Command wrongly interpreted by µP8032: - µP8032 defective	- Replace CPU
		Additional info. 1: Operation group No.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
47	01	Error during test of processor RAM: CPU HW error	Error group No. 02 Internal µP8032 RAM, Error during read-after-write check of test byte: - CPU HW error	- Replace CPU - Service
		Additional info. 1: Check byte address Additional info. 2: Setpoint contents check byte Additional info. 3: Actual contents check byte		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
47	02	Error during test of processor RAM: CPU HW error	Error group No. 02 Internal µP8032 RAM: Error during read-after-write check of environment byte - CPU HW error	- Replace CPU - Service
		Additional info. 1: Environment byte address Additional info. 2: Check byte address Additional info. 3: Setpoint contents environment byte Additional info. 4: Actual contents environment byte Additional info. 5: Contents check byte		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
48	01	Error during processor function test: CPU HW error	Error group No. 03 SSP function test: - CPU HW error	- Replace CPU - Service
		Additional info. 1: Operation group No.		
				Warm restart: 1. Stop/Run
48	02	Error during processor function test: CPU HW error	Error group No. 03 µP8032 SSP timing defective	- Replace CPU - Service
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
49	01	Operating system EPROM defective: CPU HW error	Error group No. 05 Generated signature of the operating system EPROM is different to signature read: - Operating system EPROM defective	- Replace CPU
		Additional info. 1: Actual signature value Additional info. 2: Setpoint signature	Generated signature, hex display Read signature, hex display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
50	01	EPROM module for user program defective	Error group No. 06 Generated signature of the operating system EPROM is different to signature read: - User EPROM module defective	- Write new user EPROM with COM 115F
		Additional info. 1: Actual signature value Additional info. 2: Setpoint signature	Generated signature, hex display Read signature, hex display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
51	01	PL not functioning: CPU error	Error group No. 13 Page cannot be assigned, Wrong page identifier: - CPU error	- Replace CPU - Service
		Additional info. 1: Page No. Additional info. 2: Page access type Additional info. 3: No. of program causing error		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
51	02	PL not functioning: CPU or PL error	Error group No. 23 Page has been unassignable for too long, subunits are running apart - CPU error - PL error	At first occurrence: OVERALL RESET If the error has not been removed after several OVERALL RESETs: - Replace CPU - Replace PL - Service
		Additional info. 1: Page No. Additional info. 2: No. of program causing error		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
51	03	PL not functioning: CPU or PL error	Error group No. 01 Page is not assignable due to HW error, subunits are running apart: - CPU error - PL error	- Replace CPU - Replace PL - Service
		Additional info. 1: Page No. Additional info. 2: Semaphore register Additional info. 3: No. of program causing error		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
51	04	PL not functioning: CPU error	Error group No. 13 Page is not assignable due to wrong access identifier: - CPU error	- Replace CPU - Service
		Additional info. 1: Page No. Additional info. 2: Page access type Additional info. 3: No. of program causing error		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
51	05	PL not functioning: CPU, PL or IM error	Error group No. 23 Page has been unassignable for too long due to wrong previous assignment (r-, w-): - CPU error - PL error - IM304/IM324 connection interrupted	At first occurrence: OVERALL RESET. If error has not been removed after several OVERALL RESETs: - Replace CPU - Replace PL - Check connection
		Additional info. 1: Page No. Additional info. 2: Page access type Additional info. 3: Semaphore register Additional info. 4: No. of program causing error		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
53	01	PL access not possible: PL, CPU HW or IM error	Error group No. 01 Continuous READY on PL access, recognized in job header: - PL HW error - CPU HW error - IM error	- Replace PL - Replace CPU - Replace IM
		Additional info. 1: Job No. Additional info. 2: Call location parameter		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
53	02	CPU error	Error group No. 13 Job number not defined - CPU error	- Replace CPU - Service
		Additional info. 1: Logical operating system program counter Additional info. 2: Call location parameter Additional info. 3: Job No.		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
53	03	Synchronization of A and B not possible: CPU HW or PL error	Error group No. 23 User LPLZ in A different to B - CPU error - PL error	On first occurrence of the error: OVERALL RESET If the error has not been removed after several OVERALL RESETs: - Replace CPU - Replace PL - Service
		Additional info. 1: Logical user program counter A Additional info. 2: Logical user program counter B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
53	04	Synchronization of A and B not possible: CPU HW or PL error	Error group No. 23 115F operating system LPLZ in A is different to B: - CPU HW error - PL error	On first occurrence of the error: OVERALL RESET If error has not been removed after several OVERALL RESETs: - Replace CPU - Replace PL - Service
		Additional info. 1: Logical operating system program counter A Additional info. 2: Logical operating system program counter B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
53	05	Synchronization of A and B not possible: CPU HW or PL error	Error group No. 23 Job No. or location parameter A is different to B: - CPU HW error - PL error	On first occurrence of the error: OVERALL RESET If error has not been removed after several OVERALL RESETs: - Replace CPU - Replace PL - Service
		Additional info. 1: Job No. of A Additional info. 2: Job No. of B Additional info. 3: Location parameter of A Additional info. 4: Location parameter of B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
54	01	Error message contradictory: CPU error	Error group No. 29 Error group No. of the current error message wrong: - cannot be evaluated	- Replace CPU - Service
		Add. info. 1: (FXFE_URS+0) Add. info. 2: (FXFE_URS+1) Add. info. 3: (FXFE_URS+2) Add. info. 4: (FXFE_URS+3)	Section of 8-byte buffer for error messages	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
54	02	Error message contradictory: CPU error	Error group No. 29 No I/O module identifier of the error message available:	- Replace CPU - Service
		Add. info 1: (FXFE_URS+0) Add. info 2: (FXFE_URS+1) Add. info 3: (FXFE_URS+2) Add. info 4: (FXFE_URS+3) I/O ID Add. info 5: (FXFE_URS+4)		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
54	03	Error message contradictory: CPU error	Error group No. 29 Wrong DQ module identifier:	
		Add. info: See consec. No. 02		See consec. No. 02
54	04	Error message contradictory: CPU error	Error group No. 29 Wrong I/O module identifier:	
		Add. info: See consec. No. 02		See consec. No. 02

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
54	05	Error message contradictory: CPU error	Error group No. 29 Wrong DI module identifier:	
		Additional info: See consec. No. 02		See consec. No. 02
54	06	Error message contradictory: CPU error	Error group No. 29 Wrong DQ module identifier:	
		Additional info: See consec. No. 02		See consec. No. 02
54	07	Error message contradictory: CPU error	Error group No. 29 Wrong AI module identifier:	
		Additional info: See consec. No. 02		See consec. No. 02
54	08	Error message contradictory: CPU error	Error group No. 29 Wrong AQ module identifier: - Error cannot be evaluated	
		Additional info: See consec. No. 02		See consec. No. 02

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
55	01	I/O modules cannot be accessed: I/O module or IM error	Error group No. 19 QVZ during access of I/O module (self-test): - Module defective - Wrong address on the IM306 - IM306 defective	- Replace module - Check addressing - Replace IM306
		Additional info. 1: I/O word No.	Word address containing the QVZ, decimal display	
				Warm restart: 1. Stop/Run
55	02	SINEC L1 LAN cannot be operated: CPU error	Error group No. 01 Source-slave-specific monitoring time counter of the SINEC L1 LAN does not correspond to the inverse time counter	- Replace module
		Additional info. 1: Subunit Additional info. 2: Source slave number		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
56	01	I/O modules cannot be accessed: I/O module or IM error	Error group No. 19 QVZ during access of the I/O module (short test): - Module defective - Wrong address on the IM306 - IM306 defective	- Replace module - Check addressing - Replace IM306
		Additional info. 1: I/O word No.	Word address at which the QVZ was detected, decimal display	
				Warm restart: 1. Stop/Run
58 59 60 61 62 63	01	PL cannot be accessed: PL or CPU HW error	Error group No. 01 QVZ during access of PL: - PL defective - CPU HW error	- Replace PL - Replace CPU
		Additional info. 1: No. before processed program Additional info. 2: Data pointer		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
74	01	Configured I/Os not plugged in or wrong IM306 slot address	Error group No. 16 Handling error: - Module not plugged in - Module defective - Wrong address set in IM306 - IM306 defective	- Plug in module - (if necessary, change configuration) - Replace module - Check addressing in IM306 - Replace IM306
		Additional info. 1: Subunit Additional info. 2: I/O module type Additional info. 3: Smallest I/O word No.	Subunit in which the error was discovered, character display Relevant I/O module (DI, DQ, AI, AQ module), character display Word address of the first erroneous I/O module found, decimal display	
				Warm restart: 1. Stop/Run
74	02	Non-configured I/Os are connected or DB1 is not available in the user program	Error group No. 16 Handling error: - Wrong I/O module plugged in - Connected I/O modules not fully configured; - DB1 not available	- Check I/O modules - Check I/O configuration - Check DB1
		Additional info. 1: Subunit Additional info. 2: I/O module type Additional info. 3: Smallest I/O word No.	Subunit in which the error was discovered, character display Relevant I/O module, character display Word address of the first erroneous I/O module found, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
74	03	DB1 not present in PLC: Handling error	Error group No. 16 - DB1 not in user EPROM (Safety mode) - DB1 not loaded in PLC (Test mode)	- Configure DB1 with COM 115F and transfer to PLC
				Warm restart: 1. Stop/Run
74	04	PLC cannot be activated: Handling error or CPU error	Error group No. 16 - Configuration DB missing - Configuration DB address wrongly calculated by operating system	- Check configuration DBs - Replace CPU
		Additional info. 1: I/O module type Additional info. 2: DB No. Additional info. 3: DB address		
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
90	01	Wrong module address Programming error	Error group No. 12 - Wrong configuration	- Check configuration
		Additional info 1: Module address		
				Warm restart: 1. Stop/Run
90	02	DB1 not present in PLC Handling error	Error group No. 16 - Configuration DB not loaded	- Load configuration DB into PLC
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
99	01	Synchronization error or wrong subunit ID for test mode	Error group No. 01 Page 0 has been unassignable for too long since timer 1 is out. - Different startup of subunits (difference > 30 s) - CPU, PL, PS HW error - Subunit ID identical in both CPUs	Re-enter subunit ID. If error still not removed after entering subunit ID: - Replace modules Warm restart: 1. POWER OFF/ON 2. Stop/Run
99	02	Synchronization error or wrong subunit ID for test mode	Error group No. 01 Page 0 has been unassignable for too long since timer 2 is out. See consec. No. 01	See consec. No. 01
99	03	Synchronization error or wrong subunit ID for test mode	Error group No. 01 Page 1 has been unassignable for too long since timer 1 is out. See consec. No. 01	See consec. No. 01
99	04	Synchronization error or wrong subunit ID for test mode	Error group No. 01 Page 1 has been unassignable for too long since timer 2 is out. See consec. No. 01	See consec. No. 01

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
100	01	User memory submodule for S5-115F not permissible: Handling error	Error group No. 30 - Wrong EPROM submodule plugged in	- Exchange EPROM submodule in PLC
		Additional info. 1: Submodule identifier	Submodule identifier of the submodule used, hex display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
100	02	Wrong response to power failure: PS or CPU HW error	Error group No. 01 Power failure did not trigger interrupt service routine for power failure with NAU signal - PS defective - CPU HW error	- Replace PS - Replace CPU
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
100	03	Subunit identifier for test mode with RAM is missing: Operating system awaits input	Error group No. 31 - Subunit identifier not yet entered	- Enter subunit identifier using COM 115F
		Additional info. 1: Subunit identifier found	Wrong subunit identifier found, hex display.	
				Warm restart: Not necessary
100	04	Subunit identifier in EPROM/ EEPROM submodule missing: Handling error or PG error	Error group No. 30 - Subunit identifier not present in user submodule or defective - Wrong user submodule used - COM 115F error	- Rewrite EPROM - Service
		Additional info. 1: Subunit identifier found	Wrong subunit identifier found, hex display.	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identi-fier Progr.	Con-sec. No	Error Message Text	Possible Cause	Remedy
100	05	Fault in system startup: operator error, OVERALL RESET required	Error group No. 17 Block address list construction not completed	- OVERALL RESET
		Additional info. 1: Address 1 undefined character		
100	06	Fault in system startup: OVERALL RESET required	Error group No. 13 Error in table interpreter: Maximum of 69 actions - CPU error	At first occurrence: OVERALL RESET If error has not been removed after several OVERALL RESETs: - Replace CPU - Service
		Additional info. 1: Code number	Action No. for table interpreter	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
101	01	Different user memory submodules in A and B: Handling error	Error group No. 24 - Mixed RAM, EPROM or EEPROM - Submodules of different lengths used - Memories of different internal organization used	- Use the same user memory submodule type for both subunits (memory word, memory length, memory design)
		Additional info. 1: Submodule identifier of A Additional info. 2: Submodule identifier of B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
101	02	Different operating system EPROM versions in A and B: Handling error	Error group No. 05 Different operating system signatures in the subunits: - Different operating system versions - OS EPROM changes	- Use the same operating system versions in both subunits (service)
		Additional info. 1: Signature in A Additional info. 2: Signature in B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
102	01	User memory submodule for S5-115F not permissible: Handling error	Error group No. 30 - Wrong user memory submodule plugged in	- Use permissible user memory submodule
		Additional info. 1: Submodule identifier	Submodule identifier of the submodule used; hex display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
102	02	Different user memory submodule types in A and B: Handling error	Error group No. 24	- Use the same user memory submodule type for both subunits
		Additional info. 1: Submodule identifier of A Additional info. 2: Submodule identifier of B	Submodule ID of EPROM in subunit A Submodule ID of EPROM in subunit B; Hex display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
102	03	Different user programs in A and B: Handling error	Error group No. 24 Different user program signatures in the subunits: - Different user program versions - EPROM defective	- Use COM 115F to write the same user program (same signature) in EPROM - Replace EPROM
		Additional info. 1: Signature of A Additional info. 2: Signature of B		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
102	04	Backup battery is empty or missing	Error group No. 25	Replace battery
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
102	05	Cold restart not possible: OVERALL RESET required	Error group No. 18 NINEU bit set: - NAU routine has interrupted operating system function	
				Warm restart: - OVERALL RESET
102	06	Power failure: OVERALL RESET required	Error group No. 18 NAU signal active: - NAU routine has interrupted operating system function NINEU bit not set	
				Warm restart: - OVERALL RESET
102	07	I/O module unclear: OVERALL RESET required	Error group No. 19 PEU signal active: - Power failure in expansion unit - Connection with expansion unit interrupted - Central controller connector is missing - Wrong IM304 jumper assignment	- Check power supply in expansion unit - Check connection - Plug terminating resistor into central controller - Check jumpers
				Warm restart: - OVERALL RESET
102	08	Cenral bus wrongly terminated: Component structure error	Error group No. 19 - ASF signal active (I/O)	- Check bus connections
				Warm restart: - OVERALL RESET
102	09	Block address list structure not completed: OVERALL RESET required	Error group No. 17 Operating system function interrupted	
		Additional info 1: Address 1, undefined character		
				Warm restart: - OVERALL RESET

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
102	10	I/O module error or component structure error	Error group No. 19 PEU signal active during structuring of the 9th track: - Power failure in I/O expansion unit - Connection with expansion unit interrupted - Connector in central processor missing	- Check power supply in expansion unit - Check connection - Plug terminating resistor into central controller
				Warm restart: 1. Stop/Run
102	11	Standard block too large: User EPROM error	Error group No. 6 Standard block defective	- Check standard block
		Additional info. 1: Block type Additional info. 2: Block No. Additional info. 3: Initial address		
				Warm restart: 1. Stop/Run
102	12	Signature error in standard block: user EPROM error	Error group No. 16 Generated standard block signature differs from signature read	- Load prototype-tested standard block
		Additional info. 1: Block type Additional info. 2: Block No. Additional info. 3: ACTUAL signature	Generated signature Hex display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
103	01	Error in cycle control (in accordance with ISTACK): CPU error	Error group No. 01	- Analyze error with ISTACK (UAW)
		Additional info. 1: UAW		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
103	02	Back up battery is empty or missing	Error group No. 25	- Replace battery
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
103	03	CPU error	Error group No. 13 Available OBs do not correspond to the OB list in the operating system	- Service - Replace CPU
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
103	04	I/O module interrupted or wrongly addressed in the user program	Error group No. 19 PEU: - Power failure in I/O expansion unit - Connection to expansion unit interrupted - Terminating resistor connector in the central controller missing QVZ: - A missing or defective module has been accessed in the user program	- Check power supply in the expansion unit - Check connection - Plug termination connector into central unit - Analyze error with ISTACK
		Additional info. 1: UAW		
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
103	05	Error on I/O bus	Error group No. 19 Permanent READY signal on I/O bus: - I/O module defective - IM306 defective	- Replace module - Replace IM306
				Warm restart: 1. Stop/Run
103	06	Processor HW error	Error group No. 01 Test 1 signal in CPU active:	- Replace CPU - Service
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
103	07	DQ module error	Error group No. 19 - DQ module has permanent READY signal	- Replace module
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
104	01	CPU HW error	Error group No. 03 - µP 8032 SSP timing defective	- Replace CPU
				Warm restart: 1. Stop/Run
104	02	CPU error (ISTACK for details)	Error group No. 13 Errors have appeared in word interpreter	- Analyze errors with ISTACK
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
105	01	Configured scan time exceeded	Error group No. 21 - Endless loop in user program - Configured scan time too short	- Remove program errors - Increase configured scan time (Vol. 1, 10.3.2 of the Manual) CAUTION: Value requires authorization
		Additional info. 1: Configured scan time/ 10 msec. Additional info. 2: Actual scan time/10 msec.	Desired scan time, decimal display Actual scan time, decimal display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
105	02	OB 13 call could not be processed: CPU error	Error group No. 01 - FIOBT REQ signal has not been deleted: - CPU HW defective - Interrupt disabled too long - 115F operating system in error	- Replace CPU - Check distribution of interrupt disable operations in user program - Service
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
105	03	Configured test scan time cannot be maintained	Error group No. 01 - Endless loop in restart OB - Configured test scan time too short	- Remove program errors - Remove program errors Caution: Value requires authorization
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
106	01	PG operator DB too long (>4068 bytes): Handling error	Error group No. 16	- Shorten PG operator DB (<= 4068 bytes)
		Additional info. 1: DB No. Additional info. 2: DB length	No. of the PG operator DB, decimal display Length of PG operator DB in byte, hex display	
				Warm restart: 1. Stop/Run
106	02	DB does not belong in user RAM: Handling error	Error group No. 16 - RAM requirement of user system greater than available RAM	- Reduce RAM requirement of user system - Enlarge RAM in test operation: max. size of RAM submodule equal to user EPROM in safety mode - Reduce RAM requirement of user system in safety mode
		Additional info. 1: DB No. Additional info. 2: DB length	No. of the PG operator DB, decimal display	
				Warm restart: 1. Stop/Run

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
106	03	PG interface in 115F operating system faulted: CPU HW error	Error group No. 01	- Replace CPU
		Additional info. 1: R2 (Register bank 3) Additional info. 2: R3 (Register bank 3) Additional info. 3: R4 (Register bank 3) Additional info. 4: R5 (Register bank 3) Additional info. 5: R6 (Register bank 3)		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
106	04	G DB operation on error DBs (DB 2, 3) not permissible: programming error	Error group No. 12	- Check user program
		Additional info 1: DB No. Additional info 2: DB length		
				Warm restart: 1. Stop/Run
107	01	SINEC L1 traffic disturbed: SINEC L1 HW or CPU HW error	Error group No. 11 - Transceivers defective - Bus cables to PC or CP530 interface not plugged in - Master CPU or CP530 defective - Slave CPU defective	- Replace transceivers - Check bus connections - Replace Master CPU or CP530 - Replace Slave CPU
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
108	01	Non-defined interrupt: CPU HW or handling error	Error group No. 1 Interrupt >< ZYK, NAU, INTERRUPT INTA-N - CPU HW error - Non-defined interrupt in CPU	- Replace CPU - Impermissible module connected (intelligent I/Os)
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
108	02	Power failure in central rack	Error group No. 18	- Check power supply
				Warm restart: 1. Stop/Run
108	03	Configured scan time exceeded	Error group No. 21 - Endless loop in user program - Configured scan time too short	- Remove program errors - Increase configured scan time (Vol. 1, 10.4 of the Manual) CAUTION: Value requires authorization
		Additional info. 1: Configured scan time/ 10 msec. Additional info. 2: Actual scan time/10 msec.	Setpoint scan time, decimal display Actual scan time, decimal display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
108	04	Stack overflow during interrupt processing: Handling error	Error group No. 16 - Interrupt signal frequency too high	- Check interrupt signal sensor
		Additional info. 1: Current stack pointer Additional info. 2: Maximum stack pointer		
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
109	01	LPB/LPW operation on non-configured module: Programming error	Error group No. 12	- Check address in user program
		Additional info. 1: Input byte/word No.	Word address of non-configured module, decimal display	
				Warm restart: 1. Stop/Run
109	02	Long discrepancy time for LPB/LPW operation not permissible: configuration error	Error group No. 14 - No direct access (L PB 0 to 127, L PW 0 to 126) is permitted for the specified DI byte, since a long discrepancy time has been configured	- Change configuration: Configure DI byte with shorter discrepancy time - Change user program: Program PII access (L IB 0 to 127, L IW 0 to 126)
		Additional info. 1: Input byte/word No.	Word address of the non-configured module, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
109	04	DI byte for LPB/LPW operation in A different to B, discrepancy time out	Error group No. 09 I/O module comparison individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) DI module comparison error - Configured short discrepancy time too short - Accessed DI module in one sub-unit - In the case of 2-channel sensors: * one sensor defective * one sensor signal is too far behind the second sensor signal (> short discrepancy time)	- Enlarge configured short discrepancy time - Replace module * Check sensor * Check sensor signals for equal time (max. sensor signal difference short discrepancy time)
		Additional info. 1: DI byte No. Additional info. 2: IF CH DQ present: CH DQ word No. Additional info. 3: Bit No.	Word address of non-configured module, decimal display	
				Warm restart in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET
109	05	Module does not give acknowledge signal in connection with LPB/ LPW operation: I/O module error	Error group No. 19 - Module defective - IM306 defective - Wrong address set in IM306	- Replace module - Replace IM306 - Check address in IM306
		Additional info. 1: Input byte/word No.	Word address of the module which does not give acknowledge signal, decimal display	
				Warm restart: 1. Stop/Run

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
109	06	Illegal access on I/O area with LIR, TIR order: Programming error	Error group No. 12 - The addresses given in LIR and TIR belong to an illegal area, < 4 Kbytes,> 60 Kbytes or TIR addr. > 49 Kbytes (operating system data)	- Remove programming errors
		Additional info. 1: DQ byte/word No.	Source or destination address used, hex dispaly	
				Warm restart: 1. Stop/Run
109	07	TPB/TPW operation on unconnected DQ module: Programming error	Error group No. 12 - Wrong address given for T PY or T BW operation or address not * configured * connected	- Change address * Configure address * Connect and configure module
		Additional info. 1: DQ byte/word No.	Word address of unconnected module, decimal display	
				Warm restart: 1. Stop/Run
109	08	Illegal TNB operation on I/O area: Programming error	Error group No. 12 Illegal range, < 4 Kbytes, >60 Kbytes	- Remove programming error (change address)
		Additional info. 1: Current source address Additional info. 2: Current destination address	Source or destination address used, hex display	
		Additional info. 3: Current residual length	At the beginning of execution, the residual length is equal to the parameter.	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
109	09	FB call with wrong parameter: Programming error	Error group No. 12 Substitution error (SUF) during FB call in user program	- Correct function block call (analyze ISTACK) Warm restart: 1. Stop/Run
109	10	Block stack overflow, Nesting depth > 16: Programming error	Error group No. 12 - Maximum block call nesting in user program exceeded	- Remove programming error (BSTACK, ISTACK) Warm restart: 1. Stop/Run
109	11	Undefined STEP 5 operation (NNN): Programming error	Error group No. 26 - Uncoded operation - Nesting levels exceeded - Parameters exceeded	- Remove programming error (BSTACK, ISTACK) Warm restart: 1. Stop/Run
109	12	STS operation in user program: Desired PLC stop	Error group No. 26 - SW stop by statement - SW request from PG or SINEC L1 master	Warm restart: 1. Stop/Run
109	13	Transfer error (TRAF), Parameter > number of DB words: Programming error	Error group No. 12 Transfer error: - Programmed DB operation: * DW No. > DB length * DB opening forgotten	- Remove programming error Warm restart: 1. Stop/Run

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
109	14	Interrupt DI module in A different to B, discrepancy time-out	Error group No. 09 I/O comparison, individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) Interrupt DI module comparison error: - Interrupt module defective - Process interrupt signals too frequent (Interrupt interval < 30msec.)	- Replace module - Reduce interrupt burst
		E.g. Additional info. 1: DI byte No. Additional info. 2: CH DQ byte No. Additional info. 3: Bit No.		
				Warm restart in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual error and signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. Power OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: Warm restart in the case of passivation.
109	15	Undefined interrupt: CPU HW error	Error group No. 1 Interrupt >< ZYK, NAU, ALARM INTA-N - CPU HW error - Undefined interrupt in CPU	- Replace CPU - Impermissible module connected (intelligent I/O)
				Warm restart: 1. POWER OFF/ON 2. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
109	16	Acces to interrupt register outside OB2: Programming error	Error group No. 12 Wrong access on DI 434 interrupt-generating module	- Remove programming error Warm restart: 1. Stop/Run
109	17	L PW operation on odd word address: Programming error Additional info. 1: Word No.	Error group No. 1 - 2 I/O words with different configuration not permitted - Configuring error	- Check address for L PW operation - Check configuration of I/O modules Warm restart: 1. Stop/Run
109	18	T PW operation on odd word address: Programming error Additional info. 1: Word No.	Error group No. 12 - 2 I/O words with different configuration not permitted - Configuring error	- Check address for T PW operation - Check configuration of I/O modules Warm restart: 1. Stop/Run
109	19	Nonconfigured interrupts programmed	Error group No. 12 - User program in OB 21/OB 22 enables nonconfigured interrupts for 434 interrupt module	- Check interrupt enable for 434 interrupt module in OB 21/OB 22 Warm restart: 1. Stop/Run
109	20	Interrupt with wrong edge programmed	Error group No. 12 - User program in OB 21/OB 22 defines rising edge for current enabled interrupt DI bits	- Check process interrupt edges of currently enabled interrupt DI bits for missing edge (=”0”) Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
109	21	Block called missing	Error group No. 12 - Missing block called in safety mode	- Delete block call - Load block into EPROM
		Additional info 1: Block type Additional info 2: Block No.	Display of block type (OB, FB, PW, SB) Display of block No. (decimal)	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No	Error Message Text	Possible Cause	Remedy
250	01	Initialized AI module address not connected: Programming error	Error group No. 12 - User program error - Module not plugged in - IM306 defective - Wrong address set in IM306 - Wrong address used in user program	- Check user program - Plug in and configure module - Replace module - Replace IM306 - Check addresses set
		Additional info. 1: AI word No.	Word address of the unconnected module or AI channel, decimal display	
				Warm restart: 1. Stop/Run
250	02	Wrong call parameter for FB ANEI: BG (module address) not defined	Error group No. 12 Programming error: - Module address specified is not configured	- Remove programming error
		Additional info. 1: Module address (”BG”) Additional info. 2: Channel No. (”KN”)	Initial module address, decimal display Channel No. used, decimal display	
				Warm restart: 1. Stop/Run
250	03	Wrong call parameter for FB ANEI: KN (channel No.) not defined	Error group No. 12 Programming error: - Channel number specified is not configured	- Remove programming error
		Additional info. 1: Module address (”BG”) Additional info. 2: Channel No. (”KN”)	Initial module address, decimal display Channel No. used, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
250	04	Programmed channel type (3 to 6) for AI module type 13 to 16 is wrong	Error group No. 14 Configuration error	Configuration error
		Additional info. 1: Module address (”BG”) Additional info. 2: Channel No. (”KN”) Additional info. 3: Channel type from configuration DB	Initial module address, decimal display Channel No. used, decimal display Channel type configured for this channel, decimal display	
				Warm restart: 1. Stop/Run
250	05	Wrong call parameter for FB ANEI: Individual scanning of AI 463 not permissible	Error group No. 12 Programming error: - Invalid call parameter for FB ANEI	- Change call parameter Set cyclical scanning: EINZ: KF 0
		Additional info. 1: Module address (”BG”) Additional info. 2: Channel No. (”KN”)	Initial module address, decimal display Channel No. used, decimal display	
				Warm restart: 1. Stop/Run
250	06	QVZ during access to analog value: I/O module error	Error group No. 19 - AI module defective - Channel defective - IM306 defective	- Replace module - Replace IM306
		Additional info. 1: Module address (”BG”) Additional info. 2: Channel No. (”KN”)	Initial module address, decimal display Channel No. used, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
250	07	Overflow in safety-relevant AI module	Error group No. 09 I/O comparison individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) - Value read into the AI463 > 50% more than the upper limit of the nominal range	- Check sensors and choice of range
		e.g. Additional info. 1: AI word No. Additional info. 2: CH AQ word No. Additional info. 3: CH DQ word No.	Decimal display; Check AQ word No. with subunit identification, decimal display; Check DW word No. with subunit identification, decimal display	
				Warm restart in the case of individual errors in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual errors in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
250	09	Wire-break in safety-relevant AI module	Error group No. 09 I/O comparison individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) - Connection of AI 463 module interrupted in the case of channel type 3 to 6 - V_{min} selected too high in the case of AI 463 and channel type 5, 6	- Check connections - Re-configure V_{min}
		E.g. Additional info. 1: AI word No. Additional info. 2: CH AQ word No. Additional info. 3: CH DQ word No	Decimal display CH AQ word No. with subunit identification, decimal display CH DQ word No. with subunit identification, decimal display	
				Warm restart in the case of individual errors in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual errors in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: warm restart in the case of passivation.

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
250	11	Analog value out of window range, discrepancy time-out	Error group No. 09 I/O module comparison individual error Error group No. 08 I/O error burst (only in the case of CPU 942-7UF13 with I/O ETV 3 or 4) - Two-channel sensors * 1 sensor defective * AI module defective * Window too small * Discrepancy time too short * Sensors respond at different instants (max. difference > discrepancy time) - Single-channel sensor * AI module defective * Window too small * Discrepancy time too short	* Replace sensors * Replace module * Configure larger window * Increase discrepancy time * Synchronize sensor response (Difference discrepancy time) * Replace module * Configure larger window * Increase discrepancy time
		E.g. Additional info. 1: AI word No. Additional info. 2: CH AQ word No. Additional info. 3: CH DQ word No.	Decimal display CH AQ word No. with subunit identifier, decimal display; CH DQ word No. with subunit identifier, decimal display	
				Warm restart in the case of individual errors in signal group No. 0 (Stop mode): 1. Stop/Run Warm restart in the case of individual errors in signal group No. 1 to 28 (passivation): a) Test mode: 1. POWER OFF/ON 2. Stop/Run b) Safety mode: 1. OVERALL RESET Warm restart in the case of error burst: 1. POWER OFF/ON 2. Stop/Run This does not cancel passivation. To do so, see above: warm restart in the case of passivation.

Error Identi-fier Progr.	Con-sec. No.	Error Message Text	Possible Cause	Remedy
250	15	FB250 call with wrong parameters: Programming error	Error group No. 12 Substitution error	Correct the actual FB250 call parameter (analyze ISTACK)
				Warm restart: 1. Stop/Run
250	16	Lower wire-break limit value > upper wire-break limit value: Programming/configuring error	Error group No. 12 - Programming error: FB ANEI wrongly initialized - Configuring error: global upper and lower wire-break limit wrongly configured	- Check FB ANEI parameters ”DOGR” and ”DUGR” - Check global wire-break limits
		Additional info. 1: Lower wire-break limit value Additional info. 2: Upper wire-break limit value		
				Warm restart: 1. Stop/Run
250	17	AI address configured as CP 523: Configuration error	Error group No. 12 Module address specified has been configured as CP 523	- Check FB 250 ANEI call - Check configuration
		Additional info 1: AI address		
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
251	01	Output image in A different to B: CPU or PL error	Error group No. 20	- Replace CPU - Replace PL - Service
		Additional info. 1: AQ word No. Additional info. 2: AQ value in A Additional info. 3: AQ value in B	Word address at which output images of A and B differ, decimal display Output value of the AQ module in subunit A, hex display Output value of the AQ module in subunit B, hex display	
				Warm restart: 1. POWER OFF/ON 2. Stop/Run
251	02	FB251 call with wrong parameters: Programming error	Error group No. 12 - Substitution error	- Correct the actual FB251 call parameter (analyze ISTACK)
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
252	01	FB 252 called with wrong parameters: Programming error	Error group No. 12	- Correct FB 252 call - Check parameters
				Warm restart: 1. Stop/Run
252	02	CP 523 module address wrong: Additional info 1: Module address	Error group No. 12 - FB 252 wrongly initialized	- Check parameters
				Warm restart: 1. Stop/Run
252	03	AI module not configured as CP 523: Configuration error	Error group No. 12 - Wrong configuration	- Check configuration
				Warm restart: 1. Stop/Run
252	04	Wrong character entered for function: Programming error	Error group No. 12 - FB 252 wrongly initialized	- Check parameters
		Additional info 1: Wrong character		
				Warm restart: 1. Stop/Run
252	05	Data block missing: Handling error	Error group No. 12 - Data block specified is outside permissible range (>3 und <256)	- Check FB 252 parameters
		Additional info 1 DB number	DB No. entered Decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
252	06	Wrong character entered for memory type: Programming error	Error group No. 12	- Check FB 252 parameters
		Additional info 1: Wrong character		
				Warm restart: 1. Stop/Run
252	07	Number of blocks <1 or >32: Programming error	Error group No. 12 - Illegal block length specified	- Check FB 252 parameters
				Warm restart: 1. Stop/Run
252	08	Data block missing: Handling error	Error group No. 12 - DB missing	- Provide more memory space
		Additional info 1: DB number	DB number entered Decimal display	
				Warm restart: 1. Stop/Run
252	09	Too few data words provided: Handling error	Error group No. 12 - Insufficient memory space	- Provide more memory space
				Warm restart: 1. Stop/Run
252	10	Too few flag words provided: Handling error	Error group No. 12 - Insufficient memory space	- Provide more memory space
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
252	11	QVZ on CP 523: I/O module error	Error group No. 19 - Module defective - Wrong module address set on CP 523	- Replace module - Check module address
		Additional info 1: Module address		
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
253	01	FB253 call with wrong parameter: Programming error	Error group No. 12 - Wrong parameter format	- Correct actual parameter
				Warm restart: 1. Stop/Run
253	02	FB253 call with wrong source slave No., high byte: Programming error	Error group No. 12 - Wrong slave No. in high byte; correct value = 0	- Correct actual parameter
		Additional info. 1: Source slave No.	Number of the wrong source slave, decimal display	
				Warm restart: 1. Stop/Run
253	03	FB253 call with wrong source slave No., low byte: Programming error	Error group No. 12 - Wrong slave No. in low byte; correct value: Slave No. > 0 and < 31	- Correct actual parameter
		Additional info. 1: Source slave No.	Number of the wrong source slave, decimal display	
				Warm restart: 1. Stop/Run
253	04	FB253 call with non-configured source slave No.: Programming or configuration error	Error group No. 12 - Wrong slave No. programmed - Source slave No. specified not configured	- Correct actual parameter - Correct configuration
		Additional info. 1: Source slave No.	Number of source slave not found, decimal display	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
253	05	FB253 call with non-redundant configured source slave No.: Programming/configuration error	Error group No. 12 - Wrong source slave No. programmed - No redundant source slave No. in configuration	- Correct actual parameter - Correct configuration
		Additional info. 1: Source slave No.	Number of source slave not found, decimal display	
				Warm restart: 1. Stop/Run
253	06	Both SINEC L1 LAN channels defective: SINEC L1 I/O error	Error group No. 19 - Fault in SINEC L1 LAN channels - Fault in SINEC L1 master	- Check SINEC L1 LAN channels - Check SINEC L1 Master CPU
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
254	01	FB254 call with wrong parameter: Programming error	Error group No. 12	- Correct parameter
		Additional info. 1: Parameter	Wrong actual parameter	
				Warm restart: 1. Stop/Run
254	02	FB254 call with wrong parameter: Programming error	Error group No. 12 Substitution error	Correct the actual FB254 call parameter (analyze ISTACK)
				Warm restart: 1. Stop/Run
254	03	FB254 call with parameter for non-configured SINEC LAN: Programming/configuring error	Error group No. 12 SINEC-L1 synchronisation initialized although no SINEC interface has been configured	- Correct initialization
		Additional info. 1: Parameter	Wrong actual parameter	
				Warm restart: 1. Stop/Run

Error Identifier Progr.	Consec. No.	Error Message Text	Possible Cause	Remedy
255	01	Contents of error DB or configuration DB wrong	Error group No. 13 Error DB has been overwritten: Operating system error	- Check control program - Replace CPU
		Additional info 1: Signal group Additional info 2: I/O Typ Additional info 3: Current error block No.		Warm restart: 1. Power on/power off 2. Stop/Run
255	02	FB 255 call with wrong parameters: Programming error	Error group No. 12 Substitution error	Correct actual parameter of the FB 255 call
				Warm restart: 1. Stop/Run

B.2 Evaluation of Error DBs (DB2 and DB3) without COM 115F

You will normally use COM 115F to display user-friendly error messages on your programmer screen.

You evaluate the error DBs

- if you have no CRT-based programmer
- if you want to write your own error messages.

The error DBs contain all the information required for error location.

You will find notes on the structure of the error DBs in Section 5.4.2 of the Manual.

Consult the following pages for the precise meaning of the eight bytes of an error message.

Meaning of the Eight Bytes of an Error Message

Byte		
0	=	Gr = Error group (5.5.2 of the Manual)
1	=	Error No. = (Progr. No.; Consec. No. 5.4.1)
2	=	Additional information
3 to 7		

The following assignments apply in the case of error messages with "I/O identifier" in byte 3:

Byte 3 =	I/O identifier:
Byte 4 =	Word No. (0 to 63) of DI module, DQ module or AI module;
Byte 5 =	Word No. (0 to 63) of CH DQ, R DI or CH AQ module but not the CH relay DQ module for connecting sensors or the CH AQ module in the case of AI modules;
Byte 6 =	Bit No.;
Byte 7 =	Word No. (0 to 62) of the CH relay DQ in AIs

I/O Identifier in DI Error Messages:

Bit 3 to 0 =	0001		
Bit 4 =	0	DI without	CH DQ
=	1	DI with	CH DQ
Bit 5 =	0	not used	
Bit 6 =	0	CH DQ	not in subunit A
=	1	CH DQ	in subunit A
Bit 7 =	0	H DQ	not in subunit B
=	1	CH DQ	in subunit B

I/O Identifier in DI Error Messages:

Bit 3 to 0 =	0010		
Bit 4 =	0	DQ without	R DI
=	1	DA with	R DI
Bit 5 =	0	not used	
Bit 6 =	0	R DI	not in subunit A
=	1	R DI	in subunit A
Bit 7 =	0	R DI	not in subunit B
=	1	R DI	in subunit B

I/O Identifier in DI Error Messages:

Bit 3 to 0 =	0100	
Bit 4 =	0	AI without CH AQ and CH Rel DQ
=	1	AI with CH AQ and CH Rel DQ
Bit 5 =	0	AI with 1 sensor, CH Rel DQ in the same subunit as CH AQ
	1	AI with 2 sensors CH Rel DQs in both subunits
Bit 6 =	0	CH AQ not in subunit A
=	1	CH AQ in subunit A
Bit 7 =	0	CH AQ not in subunit B
=	1	CH AQ in subunit B

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
2.1	20	3, 4	User time decr. in A /10 msec.		5, 6	User time decr. in B /10msec.	
2.2	14	3, 4	Unit time difference /10 msec.		5, 6 7	Configured limit val. /10 msec. FB 254 localization parameter	
2.3	20	3, 4	Quartz monitor counter A		5, 6	Quartz monitor counter B	
2.4	20	3, 4	Sum in A /10 msec.		5, 6	Sum in B /10 msec.	
4.1	1	3, 4	OB 13 timer val. A /10 msec.		5, 6	B 13 timer val. B /10 msec.	
4.2	14	3	Actual time interval /10 msec.		4 5	Config. max. time int. /10 msec. FB 254 location parameter	
4.3	20	3	Time difference in A /10 msec.		4	Time difference in B /10 msec.	
14.1	13	3	Desired mode A and B				
16.1	8*	3	I/O identifier $=01_H$ DI without CH DQ $=51_H$ DI with CH DQ in subunit A $=91_H$ DI with CH DQ in subunit B $=D1_H$ DI with CH DQ in both subunits $=D2_H$ Input is R DI		4	DI word No. or DQ word No.	0, 1 to 63 ** 0, 1 to 62 **
16.1	9				5	Word No. of CH DQ (if present) or R DI word No.	0, 1 to 62** 0, 1 to 63 **
					6	Bit No.	0, 1 to 15
16.4	13	3	DI byte No.	0, 1 to 127	4	DI bit No.	0, 1 to 7
16.5	19	3	DI word No.	0, 2 to 126			
17.1	19	3	AI word No.	128, 130 to 254			
17.2	8*	3	I/O identifier $=04_H$ AI without CH AQ and CH Rel DQ, with one sensor; $=24_H$ AI without CH AQ and CH Rel DQ, with two sensors; $=54_H$ AI with CH AQ and CH Rel DQ in subunit A, with one sensor; $=74_H$ AI with CH AQ in subunit A, with CH Rel DQ in both subunits, with two sensors; $=94_H$ AI with CH AQ and CH Rel DQ in subunit B, with one sensor; $=B4_H$ AI with CH AQ in subunit B, with CH Rel DQ in both subunits, with two sensors;		4	AI word No.	0, 1 to 63 **
17.2	9				5	CH AQ word No., if CH AQ present	0, 1 to 63 **
17.4	8*						
17.4	9				6	---	
17.6	8*				7	CH Rel DQ word No if CH Rel DQ present	0, 1 to 62 ** (CH Rel DQ for connecting sensors or CH DQ)
17.6	9						
18.1	20	3	Counter No.	0, 2 to 124	4, 5	Counter word in A	
		6, 7	Counter word in B				
18.2	20	3	Counter No.	0 to 127	4, 5	Counter word in A	
		6, 7	Counter word in B				

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
23.1	1	3	Message source No.	0, 1 to 30			
23.2	1	3	Message destination No.	0, 1 to 30			
23.3	1						
25.1	13	3, 4	SD 36				
25.2	16	3	No. of DB serviced		4, 5	DB address	
25.3	16	3	DB No.		4, 5	DB address	
26.1	15	3, 4 6	RAM address RAM byte in B		5	RAM byte in A	
26.2	13	3, 4	Initial address test area				
26.3	7	3, 4 6	RAM addr. inverse location Contents of inverse location		5	Contents of original location	
26.4	13						
27.1	4	3 6	Page No. Current test background	0, 1	4, 5	Memory address	
27.2	4	3, 4 6	Page No. Current test background	0, 1	4, 5 7	Memory address Current check pattern	
27.3	13	3, 4	Initial address				
28.1	16	3	I/O type	DI, DQ, AI, AQ			
28.2	8*	3	I/O identifier: $=51_H$ DI with CH DQ in subunit A $=91_H$ DI with CH DQ in subunit B $=D1_H$ DI with CH DQ in both subunits		4	DI word No.	0, 1 to 63 **
28.2	10				5	CH DQ word No.	0, 1 to 62 **
28.3	8*				6	Bit No.	0, 1 to 15
28.3	10						
28.4	8*						
28.4	10						
28.5	8*	3	I/O identifier: $=D2_H$ DQ with R DI in both subunits		4	DQ word No.	0, 1 to 62 **
28.5	10				5	R DI word No.	0, 1 to 63 **
28.6	8*				6	Bit No.	0, 1 to15
28.6	10						
28.7	8*						
28.7	10						

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte-Nos.	Additional Information	Remarks
28.8	19	3	DI word No.	0, 2 to126			
28.9	19	3	DQ word No.	0, 2 to124			
28.10	8*	3	I/O identifier:		4	DI word No.	0, 1 to 63 **
28.10	10		$=51_H$ DI with CH DQ in subunit A		5	CH DQ word No.	0, 1 to 62 **
28.11	8*		$=91_H$ DI with CH DQ in subunit B		6	Bit No.	0, 1 to 15
28.11	10		$=D1_H$ DI with CH DQ in both subunits				
28.12	8*						
28.12	10						
28.13	8*						
28.13	10						
28.14	8*						
28.14	10						
28.15	8*						
28.15	10						
28.16	8*						
28.16	10						
28.17	8*						
28.17	10						
28.18	8*						
28.18	10						
29.1	12	3, 4	Block type	FB, OB, PB, SB	5	Block No.	
		6, 7	User memory initial address				
29.2	3	3, 4	Block type	FB, OB, PB, SB	5	Block No.	
		6, 7	User memory initial address				
30.1	13	3	FXTEKO in A		4	FXTEKO in B	
30.2	13	3	FXTE_ZEI0_Hin A		4	FXTE_ZEI0_Hin B	
30.3	13	3	FXTE_ZEI0_L in A		4	FXTE_ZEI0_L in B	
30.4	13	3	FXTE_ZEI1_Hin A		4	FXTE_ZEI1_Hin B	
30.5	13	3	FXTE_ZEI1_L in A		4	FXTE_ZEI1_L in B	

Error No.	Gr	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
30.6	13	3	FXTE_ZEI2_Hin A		4	FXTE_ZEI2_Hin B	
30.7	13	3	FXTE_ZEI2_L in A		4	FXTE_ZEI2_L in B	
30.8	13	3	FXTEKO_OK in A		4	FXTEKO_OK in B	
30.9	13	3	FXEA_BLOCK_ADR_H in A		4	FXEA_BLOCK_ADR_H in B	
30.10	13	3	FXEA_BLOCK_ADR_L in A		4	FXEA_BLOCK_ADR_L in B	
30.11	13	3	FXEA_DB_END_ADR_H in A		4	FXEA_DB_END_ADR_H in B	
30.12	13	3	FXEA_DB_END_ADR_L in A		4	FXEA_DB_END_ADR_L in B	
30.13	13	3	FXEA_TEST_ANZ in A		4	FXEA_TEST_ANZ in B	
31.1	23	3	Subunit of original message	A, B			
31.2	29	3 5	Contents of err. DB byte in A Byte No. in err. DB incl. header		4	Contents of err. DB byte in B	
31.3	29	3	Signal group No.	0, 1 to 128			
31.4	29						
31.5		3	Current stack	> 230	4	Maximum permissible stack	230
32.1	19	3	CH DQ word No.	0, 2 to 124			
32.2	19	3	DI word No.	0, 2 to 126			
32.3	13	3	DI word No.	0, 2 to 126			
32.4	8*	3	I/O identifier: $=51_H$ DI with CH DQ in subunit A $=91_H$ DI with CH DQ in subunit B $=D1_H$ DI with CH DQ in both subunits		4	DI word No.	0, 1 to 63 **
32.4	10				5	CH DQ word No.	0, 1 to 62 **
32.5	8*				6	Bit No.	0, 1 to 15
32.5	10						
32.6	8*						
32.6	10						
33.1	19	3	AI word No.	128, 130 to 254			
33.2	19	3	CH relay DQ word No.	0, 2 to 124			
33.3	19	3	CH AQ word No.	128, 130 to 254			
33.4	8*	3	I/O identifier: $=54_H$ AI with CH AQ and CH DQ in subun. A , with one sensor for both subunits; $=74_H$ AI with CH AQ in subunit A, with CH DQ in both subunits, with two sensors; $=94_H$ AI with CH AQ and CH DQ in subun. B, with one sensor for both subunits; $=B4_H$ AI with CH AQ in subunit B, with CH DQ in both subunits, with two sensors;		4	AI word No.	0, 1 to 63 **
33.4	10				5	CH AQ word No.	0, 1 to 63 **
					6	-	
					7	CH relay DQ word No.	0, 1 to 62 ** (CH relay DQ for connecting sensors or CH AQ)

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
44.1	14	3	Configured safety time	10 to 16383 *10 msec.	4	Calculated SINEC cycle time	/10 msec.
45.1	7	3, 4	RAM address		5	Desired test background	
		6	Actual test background				
45.2	7	3, 4	RAM address		5	Test background	
		6	Desired check pattern		7	Actual check pattern	
46.1	2	3	Operation group No.				
47.1	2	3	Check byte address		4	Desired contents check byte	
		5	Actual contents check byte				
47.2	2	3	Adjacent byte address		4	Check byte address	
		5	Desired contents adjac. byte		6	Actual contents adjac. byte	
		7	Contents check byte				
48.1	3	3	Operation group No.				
48.2	3						
49.1	5	3, 4	Actual signature		5, 6	Desired signature	
50.1	6	3, 4	Actual signature		5, 6	Desired signature	
51.1	13	3	Page No.	0, 1	4	Page access method (from R 7) Bit 1=1 Read access requested Bit 2=1 Write access requested Bit 6=1 Reserve own page Bit 7=1 Reserve other page Bits 0, 3 to 5 are irrelelvant	
			Byte 4 = 42_H Read own page = 44_H Write to own page				
51.2	23	3	Page No.	0, 1	4	No. of program causing error	
51.3	1	3	Page No.	0, 1	4	Semaphore register	
		5	No. of program causing error				
51.4	13	3	Page No.	0, 1	4	Page access method	
		5	No. of program causing error				
51.5	23	3	Page No.	0, 1	4	Page access method	
		5	Semaphore register		6	No. of program causing error	

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
53.1	1	3	Job No.		4	Call localization parameter	
53.2	13	3, 4	Log. OS progr. counter		5	Call localization parameter	
		6	Job No.				
53.3	23	3, 4	Log. user progr. counter A		5, 6	Log. user progr. counter B	
53.4	23	3, 4	Log. OS progr. counter A		5, 6	Log. OS progr. counter B	
53.5	23	3	Job No. of A		4	Job No. of B	
		5	Localization parameter of A		6	Localization parameter of B	
54.1	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
54.2	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				
54.3	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				
54.4	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				
54.5	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				
54.6	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				
54.7	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
54.8	29	3	(FXFE_URS+0)		4	(FXFE_URS+1)	
		5	(FXFE_URS+2)		6	(FXFE_URS+3)	
		7	(FXFE_URS+4)				
55.1	19	3	I/O word No.	0, 2 to 254			
55.2	1	3	Subunit	A, B	4	Source slave No.	
56.1	19	3	I/O word No.	0, 2 to 254			
58.1	1	3	No. of prev. processed progr.		4, 5	Data pointer	
59.1	1	3	No. of prev. processed progr.		4, 5	Data pointer	
60.1	1	3	No. of prev. processed progr.		4, 5	Data pointer	
61.1	1	3	No. of prev. processed progr.		4, 5	Data pointer	
62.1	1	3	No. of prev. processed progr.		4, 5	Data pointer	
63.1	1	3	No. of prev. processed progr.		4, 5	Data pointer	
74.1	16	3	Subunit	A, B		I/O type	DI, DQ, AI, AQ
		5	Smallest I/O word No.	0, 2 to 254			
74.2	16	3	Subunit	A, B	4	I/O type	DI, DQ, AI, AQ
		5	Smallest I/O word No.	0, 2 to 254			
74.3	16						
74.4	16	3	I/O type	1, 2 to 18, gaps	4	DB No.	4 to 255
		5, 6	DB address				
90.1	12	3	Module address				
90.2	16						
99.1	1						
99.2	1						
99.3	1						
99.4	1						
100.1	30	3	EPROM identifier				
100.2	1						
100.3	31	3	Subunit identifier found				
100.4	30	3	Subunit identifier found				
100.5	17	3, 4	Address 1st undefined char.				

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
100.6	13	3	Code No.				
101.1	24	3	EPROM identifier in A		4	EPROM identifier in B	
101.2	5	3, 4	User EPROM signature in A		5, 6	User EPROM signature in B	
102.1	30	3	EPROM identifier				
102.2	24	3	EPROM identifier in A		4	EPROM identifier in B	
102.3	24	3, 4	User EPROM signature in A		5, 6	User EPROM signature in B	
102.4	25						
102.5	18						
102.6	18						
102.7	19						
102.8	19						
102.9	17	3, 4	Address 1st undefined char.				
102.10	19						
102.11	6	3	Block type: =DE_H Function block =DC_H Organization block =$E0_H$ Program block =$E2_H$ Sequence block		4	Block No.	
					5, 6	Initial address	
102.12	16	3	Block type: =DE_H Function block =DC_H Organization block =$E0_H$ Program block =$E2_H$ Sequence block		4	Block No.	
					5, 6	Actual signature	
103.1	1	3, 4	Interrupt condition code word				
103.2	25						
103.3	13						
103.4	19	3, 4	Interrupt condition code word				
103.5	19						
103.6	1						
103.7	19						
104.1	3						
104.2	13						

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
105.1	21	3, 4	Configured scan time /10 msec.		5, 6	Actual scan time /10 msec.	
105.2	1						
106.1	16	3	DB No.		4, 5	DB length	
106.2	16	3	DB No.		4, 5	DB length	
106.3	1	3	R2 (register bank 3)		4	R3 (register bank 3)	
		5	R4 (register bank 3)		6	R6 (register bank 3)	
		7	R6 (register bank 3)				
106.4	12	3	DB No.	4, 5 to 255	4, 5	DB length	
107.1	11						
108.1	1						
108.2	18						
108.3	21	3, 4	Configured scan time /10 msec.	10 to 16383 *10 msec.	5, 6	Actual scan time /10 msec.	
108.4	16	3	Current stack pointer	80H to FFH	4	Highest stack pointer	
109.1	12	3	Input byte/word No.	0, (1) to 254, (255)			
109.2	14	3	Input byte/word No.	0, (1) to 254, (255)			
109.4	8*	3	I/O identifier: =01_H DI without CH DQ =51_H DI with CH DQ in subunit A =91_H DI with CH DQ in subunit B =$D1_H$ DI with CH DQ in both subunits		4	DI word No.	0, 1 to 63 **
109.4	9				5	CH DQ word No. if CH DQ present	0, 1 to 62
					6	Bit No.	0, 1 to 15
109.5	19	3	Input byte No. or input word No.	0, 1 to 255 0, 1 to 254			
109.6	12	3	Source address or destination address	with LIR oper. with TIR oper.			
109.7	12	3	DQ byte No. or DQ word No.	with TPY oper. with TPW oper.			
109.8	12	3, 4	Current source address	Decrementing	5, 6	Current destination address	
		7	Current residual length	Init. value 0			
109.9	12						

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
109.10	12						
109.11	12						
109.12	26						
109.13	12						
109.14	8*	3	I/O identifier: $=51_H$ Interrupt DI with CH DQ in subunit A $=91_H$ Interrupt DI with CH DQ in subunit B $=D1_H$ Interrupt DI with CH DQ in both subunits		4	Interrupt DI word No.	0, 1 to 63 **
109.14	9				5	CH DQ word No. (CH DQ with interrupt DI always present in this version)	0, 1 to 62 **
					6	Bit No.	0, 1 to 15
109.15	1						
109.16	12						
109.17	12	3	Word No.	1, 3 to255			
109.18	12	3	Word No.	1, 3 to 253			
109.21	12	3	Address within the block address list The high byte indicates the block type. DC_H Organization block DD_H Organization block DE_H Function block DF_H Function block $E0_H$ Program block $E1_H$ Program block $E2_H$ Sequence block $E3_H$ Sequence block		4	Address within the block address list The low byte indicates the block No. The following applies: Block No. = $\frac{\text{Value of the low byte}}{2}$ If the high byte DC_H, DE_H, $E0_H$, $E2_H$ Block No. = $\frac{\text{Value of the low byte}}{2}$ +127 If the high byte DD_H, DF_H, $E1_H$, $E3_H$	
250.1	12	3	AI word No.	128, 130 to 254			
250.2	12	3	Module address (”BG”)	128, 130 to 248	4	Channel No. (”KN”)	0, 1 to 7
250.3	12	3	Module address (”BG”)	128, 130 to 248	4	Channel No. (”KN”)	0, 1 to 7
250.4	14	3	Module address (”BG”)	128, 130 to 248	4	Channel No. (”KN”)	0, 1 to 7
		5	Channel type from config. DB	3, 4, 5, 6			
250.5	12	3	Module address (”BG”)	128, 130 to 248	4	Channel No. (”KN”)	0, 1 to 7
250.6	19	3	Module address (”BG”)	128, 130 to 248	4	Channel No. (”KN”)	0, 1 to 7

Error No.	Gr.	Byte Nos.	Additional Information	Remarks	Byte Nos.	Additional Information	Remarks
250.7	8*	3	I/O identifier: =04_H AI without CH AQ and CH Rel DQ, with one sensor; =24_H AI without CH AQ and CH Rel DQ, with two sensors; =54_H AI with CH AQ and CH Rel DQ in subunit A, with one sensor; =74_H AI with CH AQ in subunit A, with CH Rel DQ in both subunits, with two sensors; =94_H AI with CH AQ and CH Rel DQ in subunit B, with one sensor; =$B4_H$ AI with CH AQ in subunit B, with CH Rel DQ in both subunits, with two sensors;		4	AI word No.	0, 1 to 63 **
250.7	9				5	CH AQ word No. if CH AQ present	0, 1 to 63 **
250.9	8*				6	---	
250.9	9				7	CH Rel DQ word No. if CH Rel DQ present	0, 1 to 62 ** (CH Rel DQ for connecting sensors or CH AQ)
250.11	8*						
250.11	9						
250.12	12	3, 4	Lower wire-break limit		5, 6	Upper wire-break limit	
250.13	19	3	CH relay DQ word No.	0, 2 to 124			
250.14	19	3	CH AQ word No.	128, 130 to 254			
250.15	12						
250.16	12	3, 4	Lower wire-break limit		5, 6	Upper wire-break limit	
250.17	12	3	AI address				
251.1	20	3	AQ word No.		4, 5	AQ value in A	
		6, 7	AQ value in B				
251.2	12						
252.1	12						
252.2	12	3	Module address				
252.3	12						
252.4	12	3	Wrong character				
252.5	12		DB No.				
252.6	12		Wrong character				
252.7	12						
252.8	12	3	DB No.				
252.9	12						

Error No.	Gr.	Byte No.	Additional Information	Remarks	Byte No.	Additional Information	Remarks
252.10	12						
252.11	19	3	Module address				
253.1	12						
253.2	12	3	Source slave No.				
253.3	12	3	Source slave No.				
253.4	12	3	Source slave No.				
253.5	12	3	Source slave No.				
253.6	19						
254.1	12	3	Parameter				
254.2	12						
254.3	12	3	Parameter				
255.1	20	3	Signal group		4	I/O type	
		5	Current error block No.				
255.2	12						

* Only in the case of CPU 942-7UF11 and CPU 942-7UF12 with I/O ETV 3

** Internal word count; multiply the specified value by the factor 2 to get the STEP 5 I/O address.

C SIEMENS Addresses Worldwide

SIEMENS Addresses Worldwide

European Companies and Representatives

Austria
Siemens AG Österreich
Vienna
Bregenz
Graz
Innsbruck
Klagenfurt
Linz
Salzburg

Belgium
Siemens S.A.
Brussels
Liège
Siemens N.V.
Brussels
Antwerp
Gent

Bulgaria
RUEN office of the INTERPRED corporation, agency of Siemens AG Sofia
Sofia

Czechoslovakia
EFEKTIM Engineering Consultants, Siemens AG
Prague

Denmark
Siemens A/S
Copenhagen, Ballerup
Hojbjerg

Federal Republic of Germany
Branch offices of Siemens AG
Berlin
Bremen
Dortmund
Düsseldorf
Essen
Frankfurt/Main
Hamburg
Hanover

Federal Republic of Germany (continued)
Cologne
Leipsic
Mannheim
Munich
Nuremberg
Saarbrücken
Stuttgart

Finland
Siemens Osakeyhtiö
Helsinki

France
Siemens S.A.
Paris, Saint-Denis
Lyon, Caluire-et-Cuire
Marseilles
Metz
Seclin (Lille)
Strasbourg

Great Britain
Siemens Ltd.
London, Sunbury-on-Thames
Birmingham
Bristol, Clevedon
Congleton
Edinburgh
Glasgow
Leeds
Liverpool
Newcastle

Greece
Siemens A.E.
Athens
Thessaloniki

Hungary
SICONTACT GmbH
Budapest

Iceland
Smith & Norland H/F
Reykjavik

Ireland
Siemens Ltd.
Dublin

Italy
Siemens S. p. A.
Milan
Bari
Bologna
Brescia
Casoria
Florence
Genoa
Macomer
Padua
Rome
Turin

Luxemburg
Siemens S.A.
Luxembourg

Malta
J.R. Darmanin & Co., Ltd.
Valletta

Netherlands
Siemens Nederland N.V.
The Hague

Norway
Siemens A/S
Oslo
Bergen
Stavanger
Trondheim

Poland
PHZ Transactor S.A.
Warsaw
Gda sk-Letnica
Katowice

Portugal
Siemens S.R.A.L.
Lisbon
Faro
Leiria
Porto

Romania
Siemens birou de consultaţii tehnice
Bukarest

Spain
Siemens S.A.
Madrid

Sweden
Siemens AB
Stockholm
Eskilstuna
Gothenborg
Jönköping
Luleå
Malmö
Sundsvall

Switzerland
Siemens-Albis AG
Zürich
Bern
Siemens-Albis S.A.
Lausanne, Renens

Turkey
ETMAŞ
Istanbul
Adana
Ankara
Bursa
Izmir
Samsun

USSR
Siemens AG Agency
Moscow

Yugoslavia
General Export
OOUR Zastupstvo
Belgrade
Ljubljana
Rijeka
Sarajewo
Skopje
Zagreb

Non-European Companies and Representatives

Africa

Algeria
Siemens Bureau
Alger
Algier

Angola
Tecnidata
Luanda

Burundi
SOGECOM
Bujumbara

Egypt
Siemens Resident Engineers
Cairo-Mohandessin
Alexandria
Centech
Zamalek-Cairo

Ethiopia
Addis Electrical Engineering Ltd.
Addis Abeba

Ivory Coast
Siemens AG
Succursale Côte d'Ivoire
Abidjan

Kenya
Achelis (Kenya) Ltd.
Nairobi

Libya
Siemens AG
Branch Office Libya
Tripoli

Mauritius
Rey & Lenferna Ltd.
Port Louis

Morocco
SETEL
Société Electrotechnique et de Télécommunications S.A.
Casablanca

Mozambique
Siemens Resident Engineer
Maputo

Namibia
Siemens Resident Engineer
Windhoek

Nigeria
Electro Technologies Nigeria Ltd. (Eltec)
Lagos

Rwanda
Etablissement Rwandais
Kigali

Simbabwe
Electro Technologies Corporation (Pvt.) Ltd.
Harare

South Africa
Siemens Ltd.
Johannesburg
Cape Town
Durban
Middleburg
Newcastle
Port Elizabeth
Pretoria

Sudan
National Electrical & Commercial Company (NECC)
Khartoum

Swaziland
Siemens (Pty.) Ltd.
Mbabane

Tanzania
Tanzania Electrical Services Ltd.
Dar-es-Salaam

Tunesia
Sitelec S.A.
Tunis

Zaire
SOFAMATEL S.P.R.L.
Kinshasa

Zambia
Electrical Maintenance Lusaka Ltd.
Lusaka
Mining Projects: General Mining Industries Ltd.
Kitwe

America

Argentina
Siemens S.A.
Buenos Aires
Bahía Blanca
Córdoba
Mendoza
Rosario

Bolivia
Sociedad Comercial e Industrial Hansa Ltd.
La Paz

Brazil
Siemens S.A.
São Paulo
Belém
Belo Horizonte
Brasília
Campinas
Curitiba
Florianópolis
Fortaleza
Porto Alegre
Recife
Rio de Janeiro
Salvador de Bahía
Vitoria

Canada
Siemens Electric Ltd.
Montreal, Quebec
Toronto, Ontario

Chile
INGELSAC
Santiago de Chile

Colombia
Siemens S.A.
Bogotá
Baranquilla
Cali
Medellín

Costa Rica
Siemens S.A.
San José

Ecuador
Siemens S.A.
Quito
OTESA
Guayaquil
Quito

El Salvador
Siemens S.A.
San Salvador

Guatemala
Siemens S.A.
Ciudad de Guatemala

Honduras
Representaciones Electro-industriales S. de R.L.
Tegucigalpa

Mexico
Siemens S.A.
Mexico City, D.F.
Culiacán
Gómez Palacio
Guadalajara
León
Monterrey
Puebla

Nicaragua
Siemens S.A.
Managua

Paraguay
Rieder & Cia., S.A.C.I.
Asunción

Peru
Siemsa
Lima

Uruguay
Conatel S.A.
Montevideo

Venezuela
Siemens S.A.
Caracas
Valencia

United States of America
Siemens Energy & Automation Inc.
Alpharetta, Georgia

Asia

Bahrain
Transitec Gulf
Manama
or
Siemens Resident Engineer
Abu Dhabi

Bangladesh
Siemens Bangladesh Ltd.
Dhaka

Hong Kong
Jebsen & Co., Ltd.
Hong Kong

India
Siemens India Ltd.
Bombay
Ahmedabad
Bangalore
Calcutta
Madras
New Dehli
Secundarabad

Indonesia
P.T.Siemens Indonesia
Jakarta
P.T. Dian-Graha Elektrika
Jakarta
Bandung
Medan
Surabaya

Iran
Siemens Sherkate
Sahami Khass
Teheran

Iraq
Samhiry Bros. Co. (W.L.L.)
Baghdad
or
Siemens AG (Iraq Branch)
Baghdad

Japan
Siemens K.K.
Tokyo

Jordan
Siemens AG (Jordan Branch)
Amman
or
A.R. Kevorkian Co.
Amman

Korea (Republic)
Siemens Electrical Engineering Co., Ltd.
Seoul
Pusan

Kuwait
National & German Electrical and Electronic Service Co. (INGEECO)
Kuwait, Arabia

Lebanon
Ets. F.A. Kettaneh S.A.
Beirut

Malaysia
Siemens AG
Malaysian Branch
Kuala Lumpur

Oman
Waleed Associates
Muscat
or
Siemens Resident Engineers
Dubai

Pakistan
Siemens Pakistan Engineering Co., Ltd.
Karachi
Islamabad
Lahore
Peshawer
Quetta
Rawalpindi

People's Republic of China
Siemens Representative Office
Beijing
Guangzhou
Shanghai

Philippine Islands
Maschinen & Technik Inc. (MATEC)
Manila

Qatar
Trags Electrical Engineering and
Air Conditioning Co.
Doha
or
Siemens Resident Engineer
Abu Dhabi

Saudi Arabia
Arabia Electric Ltd. (Equipment)
Jeddah
Damman
Riyadh

Sri Lanka
Dimo Limited
Colombo

Syria
Siemens AG
(Damascus Branch)
Damascus

Taiwan
Siemens Liaison Office
Taipei
TAI Engineering Co., Ltd.
Taipei

Thailand
B. Grimm & Co., R.O.P.
Bangkok

United Arab Emirates
Electro Mechanical Co.
Abu Dhabi
or
Siemens Resident Engineer
Abu Dhabi
Scientechnic
Dubai
or
Siemens Resident Engineer
Dubai

Asia (continued)

Yemen (Arab Republic)
Tihama Tractors &
Engineering Co.o., Ltd.
Sanaa
or
Siemens Resident Engineer
Sanaa

Australasia

Australia
Siemens Ltd.
Melbourne
Brisbane
Perth
Sydney

New Zealand
Siemens Liaison Office
Auckland

Abbreviations

Abbreviations

Abbreviation	Term
AI	Analog input module
AQ	Analog output module
BASP	Command output disable
BAU	Battery failure
BCD	Binary coded decimal
BE	Block end
BEC	Conditional block end
BSTACK	Block stack
CC	Central controller
CH AQ	Check analog output module
CH DQ	Check digital output module
CP	Communications processor
CPU	Central processing unit
CSF	Control system flowchart
DB	Data block
DI	Digital input module
DL	Lefthand byte of a data word
DQ	Digital output module
DR	Righthand byte of a data word
DW	Data word
EEPROM	Electrically erasable and programmable read-only memory
EPROM	Erasable (with UV light) programmable read-only memory
ET	Electronic terminator
EU	Expansion unit
FB	Function block
FW	Flag word
FY	Flag byte
IB	Input byte
IM	Interface module
I/O	Input/output (module)

Abbreviation	Term
I/O ETV	Input/output error tolerance variant
IP	Intelligent input/output module
ISR	Interrupt service routine
ISTACK	Interrupt stack
IW	Input word
JC	Conditional jump
JU	Unconditional jump
KC	Character
KF	Fixed-point number
KG	Floating-point number
KH	Hexadecimal number
KM	Bit pattern
KN/KT	Channel number/channel type
KT	Time
KY	Byte; 2 binary absolute numbers 0 to 255
KZ	Count
LAD	Ladder diagram
LPLZ	Logical program counter
NOP	No operation
OB	Organization block
OR	Overall reset
OS	Operating system
PB	Program block
PC	Programmable controller
PEU	I/Os not ready
PII	Process output image
PIQ	Process output image
PL	Parallel link
PROM	Programmable read-only memory
PS	Power supply
PT	Printer

Abbreviation	Term
PW	Peripheral word
PY	Peripheral byte
QB	Output byte
QVZ	Timeout
QW	Output word
R DI	Readback digital input module
RAM	Read/write memory
RLO	Result of logic operation
RN	RUN
ROM	Read-only memory
SAC	Step address counter (memory address)
SB	Sequence block
SINEC	Siemens Network Communication
ST	STOP
STL	Statement list
TRAF	Transfer error
UAW	Interrupt condition code word
ZYK	Scan (cycle) time exceeded

Index

Index

A

Acceptance test 10-68
Access 10-25
- in the AI address area 3-12
- operation 3-75
- permissible 10-11
- to I/O modules 10-11
- type 10-10, 1-38
Accessories 2-12, 8-42
ACCUM 1 3-10, 3-12, 3-31, 3-49
ACCUM 2 3-10, 3-12, 3-31, 3-49
Accumulator 2-4
- structure 3-10
Accuracy of user times 10-9
Acknowledgement delay 4-3
Activity bit 6-2, 6-5, 6-7
Actual operand 2-16, 2-17, 3-59 to 3-62, 3-64
Actual parameter 5-4
Actuators 1-3, 1-4, 3-27, 3-31, 3-34, 9-3, 10-2, 10-8, 10-15, 10-29, 10-32
- connecting 10-29
- fuses 4-19
Adapter casing 3-14, 8-40
- for printed circuit boards 3-3, 8-41, 8-42
Addition 3-30
Additional information 5-10
Address
- analog modules 5-1
- assignment in the I/O area 5-11
- counter 3-39, 5-5, 5-6
- grid 10-53, 10-55
- meaning 5-10
- range 2-10
  - analog inputs 2-10
  - digital inputs 2-10
  - digital outputs 2-10
Address (cont.)
- setting 4-22, 5-1
  - on the IM 306 7-36, 7-47, 10-54
- structure 5-1
- switches 5-3, 5-4
Addressing 4-19, 5-1
- communications module CP 523 7-36
- fixed slot addressing 5-2
- panel 5-2, 5-3
- slot 10-53
- variable 5-2, 10-54
AI
- absolute deviation 1-11
- access to address area 3-12
- comparison error 10-63
- intermittent signal 1-50
- module 460 1-43
- module 463 1-43
Algorithm 6-15
Analog channels, unused
- I/O type assignment 10-51
Analog input 2-5, 2-10, 3-27, 9-5, 10-37, 1-11, 6-5
Analog input module 6-1, 10-39, 10-40, 10-43, 10-46, 10-47
- 460-7LA12 8-26
- 463-4UA11 8-28
- AI 460 1-3, 10-37
  - function selector switch 6-10
  - startup 6-8
- AI 463 1-3, 10-37
  - address setting 6-14
  - front connector terminal assignment 6-11
  - switches 6-15
- channel type 6-34, 1-47
- checking 10-46
- safety-related use 6-1, 6-33
- type 13 10-39
- type 14 10-40
- type 15 10-43
- type 16 10-46
Analog input /output 3-75

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Analog input signal
- intermittent 10-38
- non-intermittent 10-37
Analog I/O modules
- configuring 6-34
Analog module Initialization form
Analog module 1-3, 10-52, 10-54, 1-30
- addresses 5-1
Analog output 2-10, 3-27
Analog output module 3-27, 6-28, 10-40, 10-43
- 470-7LA12 8-31
- 470-7LB12 8-33
- 470-7LC12 8-35
- connecting loads 6-29
- to current and voltage outputs 3-38
- type 18 10-50
Analog output signal 6-32
Analog signal
- intermittent 6-33
Analog test value 1-11
Analog value 5-8, 10-10, 10-38, 10-47, 10-50, 10-68, 1-11, 6-1, 6-4, 6-6, 6-7
- matching blocks 6-37, 6-4, 6-5
- outputting 6-42, 6-8
- reading in 6-37, 6-4
- representation 6-5, 6-8
- scaled 6-5
- scaling 6-37
- sensor 10-43, 10-44
Analysis 5-1
- ”BSTACK” 5-1
- ”ISTACK” 5-3
- ISTACK display 5-1
AND-before-OR operation 3-4
AND operation 3-3
Application 7-1
- requiring official approval 1-1
- regulations 1-1
Areas of application 1-1
Arithmetic blocks 6-3
Arithmetic operations 2-9
- overview 3-30
Assignment form 5-12, 10-25, 10-26
Assigning parameters to the PID algorithm 6-18
Automatic mode 6-16
Auxiliary function 1-23, 1-53, 1-56
Auxiliary programs 5-12
Availability 10-57
- of electronic control systems 9-3
- of the S5-115F programmable controller 9-4

B

Backup
- battery 2-12, 8-9, 8-37, 8-43, 10-1
- changing battery 4-2
- data 4-13
Basic menu 1-7, 1-13, 1-16
Basic operation 2-2, 2-3, 3-1, 3-38, 3-42, 3-62
Basic setting Default
Basic setting 1-7
Battery 4-13
- change 1-2
- compartment 2-2
- failure 1-2, 4-2, 4-8, 4-13, 5-3
- fault 5-18
- indicator 4-2
BERO 2-2, 8-9 to 8-13
Binary inputs 9-5
Binary measured value 6-16, 6-22
Binary number 2-20
Binary outputs 9-6
- safe 9-6
Binary scaler 3-73
Binary signal 6-31
Bit
- test operations 2-9, 3-44
- transfer 1-34
BLD 255 operation 2-8, 3-1, 3-39

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Block 6-1
- address 5-12
- address list 5-10, 2-6, 2-18, 3-32, 3-36
- body 2-7
- calls 10-13, 5-3, 5-4
  - conditional 3-34
  - operation, overview 3-32
  - unconditional 3-33
- code 2-14
- description 10-14, 6-25
- directory 1-24
- end 3-32, 3-37
  - conditional 3-32, 3-38
  - unconditional 3-32, 3-37
- header 2-4, 2-6, 2-7, 2-14, 5-3, 5-9
- integral 6-1
  - method of representation 2-3
  - overview 6-1
  - structure 2-7
- length 2-7
- name 3-40
- operation 3-1
- parameter 2-15, 3-64
- stack 2-5, 5-8
  - overflow 5-3, 5-4
  - pointer 5-3
- transfer 5-13
- transfer block 6-9
- types 2-5, 2-6
  - comparison 2-6

Boolean logic operations 2-9, 3-1, 3-2, 3-19, 3-26
Box terminal 3-29
Break key 1-7
Broadcasts 7-22
BSTACK 10-68, 4-5, 5-1, 5-7, 5-8
Burner control 9-4
Bus line 8-11
Bus terminal BT 777 7-1, 7-11
Byte
- high 3-10
- low 3-10

C

Cabinet installation 3-18
Cable
- adaptation of the length 3-25
- duct 3-42
- shielding 3-46

Call
- disable 2-11
- interval 10-8, 2-10, 2-11, 6-11, 6-15

Call statement 2-17
Cause of F error 1-22
Causes of interruption 4-8
CC 0 3-69
CC 1 3-69
Central controller 3-1, 10-53
- rack 3-4

Central functions 5-13
Central processing unit (CPU) 1-2, 1-3, 2-1, 2-9, 2-10, 4-3, 8-9
- description 2-9

Central rack 3-2
Certificate 1-1
Changing battery 4-2
Channel
- number 6-7, 6-8, 6-22, 6-23
- type 10-37, 10-41, 10-44, 1-44, 1-47, 1-49, 1-51, 6-5, 6-8
  - for analog input modules 6-34

Characteristics field 1-29, 1-32, 1-36
CH DQ channel number 1-40
Check
- analog outputs 10-40
- pulse 9-6
- value 9-5, 1-49

Check DQ Checking digital output
Checking 10-38
- analog output modules 10-43
- digital output modules 10-22, 10-24 10-37
- output module 1-47

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Checking (cont.)
- relay digital output 10-40
- modules 10-35, 10-43
Circuit diagram 2-2
- conversion 2-4
Circuit interruption voltage 8-14, 8-15
Clock 3-74
Clock-pulse generator 3-74
Code converter 6-2
Code switch 10-55
Coding
- element 3-13
- time 8-27, 8-30
Cold RESTART 4-8
- after power failure 2-12
- after power on 4-9
COM 115F
- configuration of error message 1-56, 1-57
- configure 1-1, 1-2
- document 1-2
- files 1-1
- main menu 1-7, 1-8
Command 7-32
- interpreter 1-6
- window 1-32
Communications 7-1, 7-10
- mode 7-34, 7-44
- interpretive 7-44
- transparent 7-44
- module 1-45
- CP 523 8-37
- setting 7-35
- parameter setting 1-13
- processor 1-1, 3-75, 6-10
- CP 523 1-4, 3-10, 3-75, 6-9
- CP 530 7-1, 10-69
- systems 1-5
Comparison
- error 10-63, 5-10
- operation 2-3, 3-29, 3-69
- overview 3-29
- processing 3-29
Compensating box 6-4
Component test 1-8, 5-12
Condition code generation 3-69
- for arithmetic operation 3-70
Condition code generation (cont.)
- for conversion operation 3-71
- for digital logic operation 3-70
- for shift operation 3-71
Conditions of inspection 10-73
Configuration I/O modules
Configuration
- block 1-27
- centralized 1-4, 3-4, 3-21
- data 10-67, 1-9, 1-19, 1-27, 1-33, 1-37
- transferring to the PLC 1-9
- data block
- delete 1-23
- display the directory 1-23
- load 1-23
- print out 1-23
- transfer 1-23
- DB 1-8, 1-27
- directory 1-24
- print 1-52
- distributed 1-4, 3-4, 3-22
- settings on the IM 304 3-24
- settings on the IM 314 3-26
- with IM 304/IM 314 3-23
- field 1-29, 1-30, 1-33
- I/O type 1-29, 1-30
- menu 1-4, 1-5, 1-30
- hierarchy 1-3
- printout 1-54
- rules 1-57
- screen for I/O type 1 1-37
Configured data 1-2
- printout 1-52
Configuring 7-19
- analog I/O modules 6-34
- I/O modules 1-29
Connecting
- cable 3-19, 3-21, 3-23
- resistance thermometers 6-6
- sensor 6-3, 6-11
- several S5-115Fs with S5 PLCs of the U range 7-22
- transducer 6-8

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Connection
- fault-tolerant 7-2
- line voltage 4-2
- nonsafety-related 7-2, 7-3
- safety-related 7-2
- to power supply for programmers 3-47
- types 7-11
- with IM 304/314 interface modules 3-23

Connector for PG or SINEC L1-LAN 4-3
Connector pin assignment
- of analog input/output modules B-3
- of the CPU B-2
- of the ER 701-3 subrack B-7
- of interface modules B-4
- power supply B-1

Construction of address lists 5-3
Contact
- positive-action 10-36

Control
- algorithm 6-18, 6-20
- bits 5-2
- circuit 3-31
- direct 10-29 to 10-31
- indirect 10-29, 10-32
- loop 10-50, 6-19
- panel 3-16, 3-17
- parameters 6-15
- processing unit
  - LEDs on the control panel 4-7
- program
  - structured 2-9
- pulse 10-34
- System Flowchart (CSF) 2-2, 2-3, 2-6

Controlled system 6-15, 6-20
Controller
- DB 6-18
- information 6-19
- setting 6-20
- structure 6-16

Conversion 6-1
- blocks 6-2
- function 3-69
- operation 2-4, 2-9, 3-1, 3-51, 3-69, 3-71
  - condition code generation 3-71
  - processing 3-51

Coordination
- byte 7-4, 7-5, 7-6, 7-19, 7-21, 1-3
  - structure 7-6

Copy
- command 1-35
- softkey commands 1-32

Correction
- algorithm 6-17
- increment 6-16, 6-17
- rate algorithm 6-17

Count 2-16, 3-27
- loading 3-25

Counter 2-4, 2-10
- count down 3-25
- count up 3-25
- operation 2-9, 3-25, 3-62
  - overview 3-62
- resetting 3-28
- scanning 3-26
- setting 3-27

Coupling relay 10-11, 10-31, 10-33, 10-34, 10-36

CP 523 Communications processor
CP 523 7-34, 1-43, 1-45, 3-12, 6-10
CP 530 7-12
CPU Central processing unit
CPU 1-6, 5-5
- address assignment 5-10
- connector pin assignment B-2
- operator functions 4-3, 4-4
- RAM 5-10
- schematic representation 2-11

Crimp-snap-in 3-29
- connections 8-42

Cross-reference lists 10-1
CSF Control System Flowchart
CSF 2-3, 2-6
Current sensor 10-41
Cursor keys Key assignment
Cutout ability 9-5
Cycle 10-27
- control 5-13
- time exceeded 4-3

Cyclic sampling 6-41
Cyclic scanning 6-28

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

D

Data
- correction 10-69
- flow charts 10-1
- to be initialized 1-31
- traffic between the master and the slave 7-4
- transmission 7-2

Data block (DB) 1-14, 1-23, 2-5, 2-10, 2-18, 3-32, 3-37, 5-3, 5-7
- address 5-3
- call 3-32
- calling 3-35
- DB 1
  - print 1-52
- delete 1-25, 3-32, 3-35, 3-36
- generate 3-32, 3-35, 3-36
- length 5-3
- load 1-26
- number 1-37
- program processing 2-19
- transfer 1-26
- used 10-13

Data communication equipment 7-47
DB Data block 1-54
Dead time of interposing relay 10-30
Decrement 3-52
Default 1-7, 1-22, 2-11
- screen form 1-4, 1-6, 1-7

Define test block time 1-12
Delete 1-35, 1-38
- menu 1-25

Design
- modular 2-1

Destination
- address 10-12, 1-33, 1-34
- slave 1-15

Destination slave number
- high-level protection 1-16

DI with interrupt capability
Discrepancy time
DI without interrupt capability
Discrepancy time
Diagnostics 1-21
- functions 9-3

Digital input 2-5, 2-7, 2-8, 2-10, 10-17, 1-28

Digital input module 7-10, 10-16, 10-25, 1-34
- 430-7LA12 8-10
- 434-7LA12 8-11
- 435-7LB11 8-12
- 436-7LC11 8-13
- 482-7LF31 8-21
- check 10-24
- non-intermittent 10-24
- type 1 10-18
- type 2 10-19
- type 3 10-21

Digital input/output module
- 482-7LA11 8-20
- 482-7LF11 8-21, 8-23
- 482-7LF21 8-22, 8-23
- 482-7LF31 8-24, 8-25

Digital input value 6-16
Digital logic operation 3-46, 3-70
- processing 3-46

Digital module 1-3, 4-18
- addresses 5-1
- connection 3-28
- initialization form 1-29

Digital output 2-5, 2-10, 10-29, 10-31, 10-40, 1-28
- 24 V, safety-related 10-31
- 220 V, safety-related 10-31

Digital output module 7-10, 10-16, 10-27, 10-28, 10-43
- 451-7LA11 8-14
- 454-7LB11 8-16

Digital output module (cont.)
- 456-7LB11 8-17
- check 10-37
- non-intermittent 10-29
- requirements 10-29
- safety-related 10-37
- type 8 10-28
- type 9 10-29
- type 10 10-29

Digital representation
- of an analog value 6-31
- of a measured value 6-16, 6-22

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Digital words unused
- I/O type assignment 10-51
Dimension drawing 3-15, 3-16, 3-17
- modules with adapter casing 3-17
- modules without adapter casing 3-16
- mounting racks 3-15
Diode switching 10-31, 10-32, 10-33
Direct access 5-8, 5-9, 9-5, 10-2, 10-5, 10-8, 10-24, 10-25, 10-63 1-38, 1-39, 3-12, 6-4, 6-7, 6-22, 6-23
Direct control
- in OB 2 10-2
- in OB 13 10-2
- intermittent 10-32
- in the user program 10-2
Direct read access
- to DI modules 10-24
Directory
- display 10-67, 4-5
- menu 1-24
DIR PLC 5-5
Disconnection facilities 4-18
Discrepancy 1-8, 2-5, 2-7, 2-8, 9-5, 3-12, 5-16, 6-7
- analysis 5-6, 9-5, 10-24 3-12, 6-5
- criterion 1-46, 1-47, 1-50
  - absolute 1-47
  - relative 1-47
- interval 2-7
  - long 2-8
  - short 2-7
Discrepancy (cont.)
- time 9-5, 10-7, 10-8, 10-10, 10-11, 10-25, 10-29, 1-8, 1-36, 1-38, 1-39, 1-56, 2-10, 5-10, 5-11, 6-1, 6-5
  - long 10-10, 1-10, 3-12
  - short 10-10, 3-12
  - short, for DIs without interrupt capability 1-10
  - short, for non-interrupt DIs 10-11
- value 1-46, 1-47, 1-50
Display 4-4
- cause of F error 1-21
- generation operation 3-38, 3-39
- signal states 4-2
Disturbance variable 6-15, 6-16, 6-18
Divider 6-4
Documentation 1-24
DO operation 3-35, 3-64
Down time 10-38
DQ address 126 10-27
Duplication of addresses 10-55

E

Early failures 9-2
Edge
- change 5-12
- evaluation 3-72
- initialization 10-26
- trigger flag 4-3
EEPROM 1-27
EEPROM submodule 2-12
Electrical shock 3-41
Electromagnetic interference 3-46
Emergency OFF
- commands 10-27
- equipment 4-18
Enable
- functions 9-5
- operation 3-42
Enable/disable initialization 10-25

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Entry
- fields 1-8
- function 4-5
- illegal 1-8
- in the Error DB 5-10
Entry DB 4-6
EPROM 1-27
- submodule 2-12
Equipotential bonding 3-44, 6-29
- conductor 3-44
- in the case of distributed configurations 3-45
Error
- accumulation 5-17
- address 5-5, 5-6
  - determining 5-5
- analysis 10-63
- bit 6-7, 6-22, 6-23
- bits/words 5-14
- buffer 5-15
- bursts 9-5, 10-58
- DB 10-13, A-1, 5-10
  - block 1 10-69
  - error entry 5-14
  - evaluation 5-16
  - memory area 0 10-59
  - transfer 7-11
- detection 5-10
- diagnostic 1-21, 5-1
- entry 4-6, 5-14
  - in error DBs 5-14
- group 7-39, 5-16
- ID 3-39
- information 4-13
- line 1-32
- message 5-9, 10-63, A-1
  - COM 115F 5-15
  - COM 115F configuration 1-56, 1-57
  - EPROM function 1-58
  - occurring when using diskettes 1-59
  - PLC interface 1-59
- message frame 7-11, 7-37
- message text 7-39

Error (cont.)
- response 1-8, 2-7, 4-6, 10-25, 10-27, 10-63, 10-65, 1-38, 1-49, 5-12, 5-14, 5-15, 5-16
- routine
  - central 5-15
- signalling
  - indicated by LEDs 5-5
- stack 10-58, 5-15
- tolerance variant 10-57, 10-58
  I/O modules error tolerance variant
Evaluation of error DB 5-16
Exec key 1-7
Execution
- of the operation 3-52
- time 2-9, 2-10, 10-7
- time component 10-8
Expansion
- capability 1-4
- unit (EU) 1-4, 2-2, 2-4, 3-4, 3-6, 3-8, 3-10, 5-2, 8-3, 8-4, 8-5, 8-38, 8-40, 10-54

F

Failure
- characteristics 9-2
- occurrences
  - distribution 9-3
Failsafe characteristics
- of the CP 523 7-47
False interrupts 10-25
Fastening screw 3-31
Fault-voltage-operated circuit breaker 3-35
FB
- integrated 5-13
FB 250 ANEI Function block
FB 252 BLUE 6-10
FB 253 7-32, 6-10
FB 253 MBXT 7-32
FB 254 6-11
FB 255 DEPA 6-14

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Feedback
- address 1-31, 1-33
- module 1-8, 9-5

Feedforward injection of disturbance variable 6-17
Field transfer 3-67
File name 1-6
Filter program 7-10, 10-67, 10-68, 4-4
Flag 2-10
- area 5-9
- word 2-16

Flip-flop 3-8, 3-9
Footer 1-6
FORCE 4-3
FORCE VAR 4-3
Formal operand 2-14, 2-17, 2-18, 3-59, 3-62, 3-64, 3-69
Four-wire transducer 6-8
Frame
- body 7-11
- header 7-11

Front connector 3-29, 8-27, 8-32, 8-34, 8-36, 8-42 10-49
- installation 3-30, 6-11
- pin assignment 6-2
- terminal assignment 3-35, 6-7

Front connector AE 463
- terminal assignment 6-12

Function
- BSTACK 5-7

Function block 5-8, 7-37, 7-41, 8-9, 10-14, 10-47, 2-5, 2-13, 2-15, 3-30, 3-64
- creating 2-14
- FB 250 ANEI 10-38, 1-12, 1-47, 6-4
- FB 251 6-8
- FB 252 7-44, 7-48
- FB 252 BLUE 6-9
- FB 253 6-11
- FB 253 MBXT 9-4
- FB 254 SYNC 10-9
- FB 255 DEPA 6-14

Function block (cont.)
- integral 6-1
- integrated 2-17, 5-13
- programming 2-17
- test 10-13, 10-38

Function selector switch
- AE 460 6-10

Fuses 8-44
- for sensors and actuators 4-19

G

Global wire-break limit 1-12
Grounding
- of the cable shielding 3-46
- point 3-35

H

Handling menu 1-8, 1-36, 1-53
Hardware defects 5-10
High availability 7-31, 7-32

I

Illegal access 3-75
IM 306 B-6
IM 306 interface module
- addressing 10-54

IM 324 Parallel interface
IM 324
- jumper settings for parallel interface 3-20

Image comparison 1-7
- cyclic 10-63

Increment 3-52, 3-53
Incrementing sequence 3-76
Individual
- acceptance test 10-68, 10-70
- error 5-17
- passivation 10-57, 10-58, 10-59
- test 10-66

Inductances
- installed 3-46

Initial address 10-54
Initial module address 10-53
Initialization
- form
  - analog modules 1-30
  - for CP 523 I/O type 13 1-45
  - for digital modules 1-29

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Initialization (cont.)
- form
  - I/O type 2 1-38
  - I/O type 8 1-40
  - I/O type 9 1-41
  - I/O type 10 1-42
  - I/O type 13 1-43
  - I/O type 14 1-46
  - I/O type 15 1-49
  - I/O type 18 1-51
- of the operating system 2-10, 7-5
- program 4-4
- screen form
  - for I/O type 3 1-39

Initialize Operating system
Initialize operating system 1-8
Input
- field 1-7
- frequency 3-73
- module 1-3, 2-2, 1-43, 1-44
- parameter 2-15
- signal 10-24, 10-37, 10-38
  - intermittent 10-19, 10-46
- value 5-6
  - digital 6-16

Input/output I/O
Input/output bus 2-4
Installation
- electrical 3-34
- of the front connector 3-30, 6-11
- guidelines 3-1
- mechanical 3-12
- of the modules 3-12

Installation in a 19-in cabinet
- dimensions 3-18

Integration time 8-27, 8-30
Intercall interval 10-26
Interconnecting the two subunits 3-19
Interface SINEC L1
Interface
- serial 2-4

Interface connector 1-3, 7-35
Interface module (IM) 1-2, 1-4, 2-2, 3-1, 3-16, 3-17, 3-21 to 3-23, 7-9, 8-38 to 8-40, 10-54, 10-56, 5-1

Interface module (cont.)
- connector pin assignment B-4
- functions 2-11
- IM 304 8-38
- IM 306 3-21, 5-2, 5-3, 8-39
- IM 314 8-40
- IM 324 8-41
- setting addresses 5-3

Interface relays 9-6
Intermittent 1-36, 1-39
- signal 1-38, 1-50

Internode
- communication 7-14, 7-15,
  - message structure 7-22
- traffic 5-3

Interposing relay 10-29, 10-30, 10-32, 10-46
Interrupt 5-14, 10-9, 10-25
- analysis 5-1
- block 2-12
- DI 434 Interrupt digital inputs
- digital inputs DIs 10-25
  - automatically initialized 10-26
  - module, non-matching 3-12
- disable 10-26, 2-10, 3-53, 5-3
- DI test 10-27
- edge byte 5-4
- enable 3-53, 5-3
- enable byte 5-4
- evaluation 8-4
- generation 2-8
- handling 1-10, 2-9, 3-54
- inhibit 10-7
- input 5-15
- input module 5-4
- list 10-68
- messages 7-22
- module 5-14
- processing 5-14, 10-9, 1-8, 1-12, 2-10
- register 5-12, 10-25
- register byte 5-4
- request 5-15, 5-3
- response time 2-10, 6-11

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Interrupt (cont.)
- stack 5-1, 5-2
- synchronization 5-3
Interslave data traffic 7-31
Interval monitoring 2-10, 3-54
I/O error 7-40, 10-6, 10-57, 10-58, 10-66, 1-10, 1-54, 5-10
I/O error tolerance variant (I/O ETV) 10-58
- I/O ETV 1 10-57
- I/O ETV 2 10-57
- I/O ETV 3 10-57
- I/O ETV 4 10-57
- I/O ETV 3 and 4
  - flowchart 10-65
- I/O ETV 3 with signal group 28 10-58
- I/O ETV 4 with signal groups 28 and 29 10-63
- process schematic 10-60
- signal group 10-58
- using I/O ETV 3 10-65
- using I/O ETV 4 10-65
- with signal group 28 10-58
I/O ETV 1, 2, 3
  I/O error tolerance variant
I/O module CP 523
I/O modules
- configuring 1-19
- error tolerance variant 1-10
- responding to errors 10-57, 1-30
- serial 7-34
I/O type
- assignment
  - unused analog channels 10-51
  - unused digital words 10-51
- characteristics 10-16
- characteristics field 1-29 to 1-31
- configuration field 1-29, 1-30
- mixes 10-52
- nonsafety-related 6-33
- print 1-52
- safety-related 6-33
I/O type 1 1-37
I/O type 2 1-38
- initialization form 1-38
I/O type 3 1-39
I/O type 8 1-40
I/O type 9 1-41
I/O type 10 1-42
I/O type 13 1-43
I/O type 14 1-46
I/O type 15 1-49
I/O type 16 1-50
I/O type 18 1-51
Isolation
- galvanic 3-32
ISTACK 1-21

J

Job
- header 5-12
- number 7-39
Jump
- conditionally 3-32, 3-57
- functions 3-49
- operations 10-12, 10-13, 2-5, 2-14, 3-33 to 3-35, 3-37, 3-57, 3-69 to 3-71, 5-7
  - processing 3-57
  - to missing blocks 3-33, 3-34
  - to unloaded blocks 10-13
- response 6-19
- statement 2-17
- unconditionally 3-32, 3-57
Jumper comb 8-42

K

Key assignment 1-7

L

Label strips 3-27
LAD Ladder diagram
Ladder diagram (LAD) 2-2, 2-3
LAN fault 9-4
Library number 2-14
Licensing authority 1-2
Limit value processing 9-5
Line voltage connection 4-2
LIR operations 10-12
Load circuit 3-33
- grounding 3-35
Load operation 3-10, 3-12, 3-42, 3-61, 3-66
- overview 3-61, 3-66

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Load power supply 10-59
- units 3-33, 3-34
Load resistance 8-32, 8-34, 8-36
Loads
- connecting to analog output modules 6-29
Local area network SINEC L1
Location 10-26
- parameter 6-11
Logic operation 1-8, 2-3
- binary 3-59
Logical program counter 10-1
Lower limit 6-15
LPLZ Logical program counter
LPLZ sequence 2-7, 3-39, 3-76

M

253 MBXT 6-10
Mailbox 7-3, 7-14, 7-15, 7-19 to 7-22, 7-30 to 7-32, 10-69, 1-3, 1-57
- address 1-17
- length 7-21, 1-17
- structure 7-18
- system 7-19
- table
  - print out 1-17
- transfer block FB 253
Main circuits 10-32
Main menu of the printout package 1-52
Main message 5-14 to 5-16
Main switch 4-19
Malfunction
- analysis 5-3
- report 5-1
Manipulated variable 6-16
Manual mode 6-16, 6-18, 6-23
Marking field 5-15
Master
- coordination byte 1-17
- traffic 7-14
Master-slave
- data traffic 7-31
- data transfer 7-5
- principle 7-1
Matching blocks 6-4
Mean time to repair 9-4
Measured value
- binary 6-16, 6-22
Measures against electromagnetic interference 3-46
Memory
- area
  - illegal 10-12
- available 2-12
- capacity 2-12
- cell 3-73
- internal
  - program memory 2-10
- map 5-10, 2-15
  - CPU 5-10
- programming number 1-26
- requirement 2-14
- submodule 2-2, 2-3, 2-11, 2-12, 4-3, 1-8, 1-26
- type 4-9, 4-10
Message
- mode 7-13
- module 7-37, 10-63, 1-45
- structure for internode communication 7-22
- text 7-34, 2-18
  - number 7-39, 7-40
  - printout 7-44
Methods of representation 2-1
Mode
- LEDs 4-4
- selector 4-4, 2-12, 3-39, 5-3, 5-4
Modifying the program 2-20
Module
- addressing 10-52
- delay 2-6, 10-2
- installation 3-12
- intelligent 1-4
- test 1-8
- with adapter casing
  - dimension drawing 3-17
- without adapter casing
  - dimension drawing 3-16
Monitoring
- program 2-13
- time 1-8, 2-5, 10-9, 10-65

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Mounting rack 2-2, 3-1 to 3-3, 3-5, 3-6, 3-8, 3-10, 3-12 to 3-15, 8-3 to 8-5
- CR 700-0
- for central controller 8-3
- possible configurations 3-3
- CR 700-2F
- for central controller 8-3
- possible configurations 3-2
- dimension drawing 3-15
- ER 701-1
- for expansion unit 1 8-4
- possible configurations 3-6
- ER 701-2
- for expansion unit 2 8-4
- possible configurations 3-8
- ER 701-3
- for expansion unit 3 8-5
- possible configurations 3-10
Multi-mailbox system 7-14, 7-30
Multiple connection CP
Multiple message 5-4
Multiplier 6-3
Multitest 10-63

N

Nesting 2-5
- depth 2-5, 2-11
Network SINEC L1
Network
- fault-tolerant 7-30
- interrupt 5-18
- via SINEC L1 LAN 10-2
Noise immunity 8-2
Nominal current Power supply
Nominal range 6-4, 6-5, 6-8
Non-interrupt DI 10-25
- direct access 10-63
Non-interrupt-generating DI module
- nonmatching 3-12
”NOP” operation 3-39
Number 10-26
- format 6-17, 3-31
- of data paths 10-5
- representation 2-20

O

OB Organization block
OB 2 5-14
ODGR Analog value matching blocks
Off-delay 3-24
- timer 3-16, 3-62
OGR Analog value matching blocks
On-delay 3-22, 3-23
- latching 3-23
- reset 3-23
- timer 3-16, 3-62
Operand 5-3
- areas 2-3, 3-10
- digital 3-44
Operating function 1-23
Operating mode 4-4
- changing 4-8
- settings 4-7
Operating system 1-8
- additional tasks 1-8
- central functions 5-13
- function 1-8
- initializing 7-5
- maximum execution time 10-7
- parameters 3-54
- run time 2-10
Operating voltage 1-2
Operation 1-2
- arithmetic 3-30, 3-31, 3-68
- condition code generation 3-70
- changing the memory 10-1
- DI 3-76
- JUR 3-76
- LIR 3-75
- L PW 3-75
- L PY 3-75
- mnemonics 2-21
- other 3-38, 3-69
- output inhibit 4-3
- register 5-3
- sequence 3-76
- set 2-10, 2-5, 2-14, 3-42
- supervised 10-65
- supplementary 2-3, 3-1, 3-42, 3-65

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Operation (cont.)
- TIR 3-75
- TNB 3-75
- T PW 3-13, 3-75
- T PY 3-13, 3-75
Operator
- entry function 10-67, 10-68
- functions
- of the CPU 4-3, 4-4
- overview 4-5
- programmer 4-4
- inputs 7-10
- panel 2-2
- parameter 4-4
OP mode 1-6
Optocoupler 3-28, 7-45, 8-10, 8-17, 8-20
OR-before-AND-operation 3-5
Ordinal number 6-11
Organization block 2-4, 2-5, 2-7 to 2-9, 3-1, 6-1
- integral 2-9
- OB 1 2-4
- OB 2 10-26
- OB 13 1-10, 2-11, 2-12, 6-13
- OB 21 2-12
- OB 22 10-69, 2-12
- OB 251 3-1, 6-1, 6-15
- overview 2-8
Output
- modules 1-2, 1-3, 2-2
- RESET 4-13, 10-61
- switch 4-4
Output signal
- analog 6-32
- intermittent 10-29
Overall reset 10-66
Overflow condition code 3-70, 3-71
Overvoltages 3-48

P

Package selection 1-54
Page area 3-75
Parallel configuration 3-2, 3-3
Parallel interface 1-7, 2-2 to 2-4, 3-19, 3-20, 5-11, 8-41, 5-12
- settings on the IM 304 3-19
- settings on the IM 324 3-20
Parallel link 5-11
Parameter
- entry DB 1-14, 4-4
- list 2-17, 2-18, 3-40
- setting 1-35
Parentheses 10-12
- level 3-76
Parenthesized structure 3-2
Partners 7-9
Passivation 4-6, 9-4, 10-57, 10-59, 10-63, 10-66, 1-38, 1-39, 1-42, 1-47, 5-15, 5-17, 6-7
- AI modules 10-59
- AQ modules 10-59
- DI modules with interrupt capability 10-59
- DQ modules 10-59
- error 4-6, 5-15
- value 10-59, 10-63, 6-7, 6-22
Path name 1-6
PLC
- cycle time 10-2
- defining 10-7
- function 1-20
- max. 1-9
- redundancy 9-5
- scan time 2-6
- STOP program
- RESTART 4-11
Peripherial access 2-5
Permanent enable 10-27
Permanent error 10-69
Permission
- to send 7-5
- to write 7-5

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Personal computer 1-2
PG Programmer
PID
- algorithm 1-3, 6-1, 6-15
- assigning parameters 6-18
- calling the controller 6-19
- control algorithm
- typical application 6-20
- DB 6-15
PII 5-8, 10-61
Pin assignment
- front connector 6-2
- socket connector of the CP 523 7-43
Plaintext
- error message 1-21
- line 1-31
- message 5-16
Point-to-point connection 7-9, 7-11, 7-34, 1-15
Polling list SINEC L1
Polling list 7-21, 10-6, 10-70
Polling time SINEC L1
Polling time 7-19, 7-21, 10-5, 10-6, 10-70
Possible configurations
Mounting rack CR 700-0
Mounting rack CR 700-2F
Mounting rack ER 701-1
Mounting rack ER 701-2
Potential difference 6-3
Power 4-2
- cable filter 3-48
- failure 1-2, 5-3, 5-4
- supply 1-2, 3-31, 9-4
- connector pin assignment B-1
- nominal current 3-33
POWER ON
- RESTART 4-4
Power supply module 2-1, 4-2, 8-6 8-8
- PS 951 connection 3-27
Print
- menu 1-28
- mode 7-34, 7-37
Printed circuit boards 3-14
Printer 2-13
Printout
- configuration 1-28, 1-54
- configured data 1-52
- package 1-52
Process
- access 5-13
- image 2-4, 4-4, 4-5, 5-5, 5-10, 3-14, 4-2, 4-3, 5-9
- input image 10-3, 10-10, 10-63
- interrupt 1-7, 5-12, 5-14, 5-15, 10-5, 10-26, 2-8, 2-10, 5-13, 6-1,
- interrupt scanning 6-10
- synchronization 2-10
- I/O image 10-11
- accessing 5-6, 5-7
- of the outputs 5-5
- output image 10-3
- parameters already set 1-7
- response time 10-26
- signals
- handling 5-5
- stand-alone 10-59
- variable 4-3
Processing
- an arithmetic operation 3-31
- comparison operation 3-29
- for servicing process 6-13
- of interrupts 2-5
- status 1-32
- time-controlled 6-15
Processing for time interrupts
Organization block (OB 13)
Processor 2-4, 2-11
Program
- block 9-4, 2-13, 2-16, 3-35, 3-37
- check 5-3
- counters 2-4, 3-2, 3-7, 3-39, 3-76

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Program (cont.)
- cyclical 10-38
- error-detecting 5-11
- errors 5-5
- file 1-7
- memory 2-3, 2-10
- operation 2-11
- parts 2-5
- processing 2-19
  - time-driven 2-11
  - with data block 2-19
- scanning 2-5, 2-10, 2-8, 2-10, 2-12, 4-1
  - cyclic 2-5, 2-9
- scan time 2-6
- structure 2-4
- structured 3-32
- test 7-32, 4-1
- time-driven 10-38
- trace 5-7

Programmer (PG) 1-6, 2-12, 2-13, 7-32, 1-1, 1-2, 1-6, 1-27, 3-38, 3-72
- connected to the serial interface of the CPU 7-32
- connecting 10-66
- connection to power supply 3-47
- connectors 4-3
- display 4-4
- function 4-6, 7-2, 5-5
- handling 10-66
- input data 7-10, 7-32
- number 1-58
- operation 4-6
- operator
  - entry 10-67
  - function 4-4
- port 7-1

Programmer function Programmer
Programmer operator function Programmer
Programmer (PG) (cont.)
Programming 2-7
- linear 2-4
- number 1-26
  - memory submodule 1-26
- structured 2-5

Proportional-action 6-15
Protection facilities
- outputting operations 10-27

Protective ground conductor 3-41
Prototype test certification C-1
Pulse 3-20
- extended 3-21

Pulse duty train 3-74
Pulse train 3-74
Pulse width 3-43

Q

Quiescent state
- safe 9-5

QVZ Time-out

R

RAM
- comparison-errors 4-6
- submodule 2-12

Range card 6-8, 10-39, 1-44, 1-47
- module 1-3

Reaction-free 1-5
Read
- accesses 10-10
- loops 10-21

Readback
- digital input 10-37
- function 10-31
- inputs 9-6, 10-29
- method 10-32
- result 10-32

Reading
- cyclical 10-2

Readout
- safety-related 1-52

Receive
- error 5-14
- frame 7-11
- mailbox 7-3, 7-6, 7-15, 7-16, 7-18, 7-32, 10-5, 10-65, 6-10
  - delete 10-70
- permission 5-3

Redundancy structure 1-7
Register contents 5-12
- loading 3-65
- transferring 3-65

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Regulations for applications
requiring official approval 1-1
Relay 8-17, 8-44
- digital output
- DQ 458-7LB11 10-27
- output 3-27, 10-24, 10-28, 10-36
- output module
- 458-7LA11 8-18
- 458-7LB11 8-19
Release 5-9
Reliability of SIMATIC S5 programmable controllers 9-2
Repair procedure 10-66
Replacement of modules 3-13
Representation of an analog value
- digital 6-31
Representation of a measured value
- digital 6-16, 6-22
Requirements
- for sensors with current outputs 10-37
- to safety class 1-1
Reset
- counter 3-25
Resistance thermometers 2-2, 3-52
- connecting 6-6
Resistance-type sensor 6-4
Response accuracy
- in the case of interrupts 10-9
Response time SINEC L1-LAN
Response time 2-6, 10-2, 10-5, 10-9
- in the case of cyclic reading 10-3
- in the case of direct access in the cyclic program 10-3
- in the case of direct access in the process interrupt OB (OB 2) 10-5
- in the case of direct access in the time interrupt OB (OB 13) 10-5
- in the case of SINEC L1-LAN link 10-5
- of the PC S5-115F 10-7
- SINEC L1 10-5, 10-6
Restart 4-4, 4-8, 5-6
- block 4-8, 2-12
- characteristics
- setting 2-12
- from POWER ON 4-4
- from STOP 4-4
- manual 2-12
RESTART test 4-4
Result of the logic operation 2-4
Return address
- absolute 5-7
- relative 5-7
RI specification 8-2
RS flip-flop with flags 3-9
”RUN” mode 4-5

S

S5 command interpreter 1-6
S5 DOS operating system 1-23
Safe inputs 9-5
Safety
- class requirements 1-1
- function 5-8, 5-12, 10-26
- interval of SINEC L1 7-22, 2-12
- mode 4-6, 4-9 to 4-13, 4-17, 4-24, 10-1, 10-13 to 10-15, 10-66, 10-69
1-8, 1-13, 1-14, 3-33, 3-34, 4-4, 4-5, 5-3, 5-17, 5-18, 6-25
- startup of an S5-115F 4-17
- operation 4-19
- requirements 10-70
- requirements made of the user program 10-64
- response 10-58, 10-63, 10-61, 10-70
- test 6-25
- time 7-19, 7-21, 7-31, 9-5, 10-2, 10-5, 10-7, 10-9, 10-70
- SINEC L1 7-30, 10-69
Safety-related use 6-22
- analog input module 6-33
Safety time SINEC L1
Sample programs 3-72
Sampling
- cyclical 6-41, 6-5
- instant 6-20

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Sampling (cont.)
- interval 6-15
- selecting 6-19
- point 6-17
- selective 6-41

Scaling
- of analog values 6-5
- schematic 6-5

Scan
- monitoring time 2-10, 2-11
- time 2-5
- exceeded 5-3 to 5-5
- monitoring 4-4, 8-9, 2-11
- trigger 2-5, 4-1

Scanning 6-26
- cyclic 6-28
- selective 6-28

Scratch flag 2-10, 2-12
Screw terminals 1-2
Screw-type
- connection 1-3, 3-29
- terminals 8-42

Search keys 4-4
Secondary message 1-22
Second error
- entry time 5-10
- occurrence 9-5
- occurrence time 10-15, 10-20, 10-58, 10-65, 10-66, 1-38

Segment 3-39
- boundaries 3-57

Segmentation 3-39
- operation 10-2, 2-8, 2-13

Segmenting 3-1
Selective sampling 6-40, 1-43, 6-5, 6-7, 6-22, 6-23 6-41
Selective scanning 6-28
Selftest 4-6, 1-9
Send
- buffer 7-4, 7-5
- mailbox 7-3, 7-15

Sensor 3-27, 3-28, 3-31, 3-34, 4-19, 5-12, 6-2, 9-3, 10-21, 10-23, 10-24, 10-40, 10-41, 10-43, 10-47, 1-39, 1-46, 1-47, 1-49
- connecting 6-3, 6-11
- deactivatable
- connection 10-21
- floating 6-3
- fuses 4-19
- implementation 10-17
- line 8-31, 8-33, 8-35
- non-deactivatable
- connecting 10-23
- nonfloating 6-3
- requirements with current outputs 10-37
- signal 10-17, 10-22, 10-25
- non-intermittent 10-22
- requirements 10-17
- supply 10-21, 10-22, 10-24
- two-channel connection 10-20

Sequence block 2-5, 2-13, 2-16, 3-1
Sequence of operations 10-1, 10-2
Set
- counter 3-25
- operation 3-65

Set/reset operation 3-7, 3-60
- overview 3-60

Setting
- addresses 5-4
- communication parameters 1-13

Sheet-metal barriers 3-47
Shielding 3-45
Shift operation 3-49, 3-69, 3-71
- condition code generation 3-71
- intermittent 1-38
- overview 3-49
- processing 3-49

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Short test at restart 10-27
Shutdown 10-29
Signal
- binary 6-31
- characteristics 10-25
- group 9-4, 10-25, 10-46, 10-57, 10-59, 1-36, 1-38, 1-39, 1-41, 1-42, 1-46, 1-49, 1-50, 1-54, 5-14, 5-15
- mix 10-60
- group number 10-38, 5-15
- of the AI and the relevant check AQ 10-38
- intermittent 10-15, 10-16, 10-25, 10-37, 10-38, 1-38
- non-intermittent 10-36
- output
- RS flip-flop 3-8
- safety-related 10-15
- sensors
- permanently failsafe 10-17
- status display 4-1
- ”STATUS” 4-2
- ”STATUS VAR” 4-2
Signalling functions 2-1
Signature 4-10, 4-11, 7-22, 7-29, 1-26, 1-27
- extension 7-22
SIGUT 3-29
SIMICONT 8-17
Simulator 3-31, 8-43
SINEC L1 4-3, 4-5, 7-1, 10-2, 10-5, 10-9, 10-63, 10-66, 10-68
- configuration 7-19
- data exchange 10-6
- data path 5-15
- interface 1-16, 1-17
- LAN 10-68, 6-10
- networking 10-2
- redundancy 9-4
- response time 1-16
- single-channel 7-17
SINEC L1 (cont.)
- LAN
- link 1-5
- master 7-33
- master coordination byte 1-17
- messages 7-11
- network 1-15
- polling list 10-69, 1-14
- polling time 10-69, 1-16
- processing 6-13
- programmer functions 7-2
- redundant 1-17
- response time 10-5, 1-16
- safety interval 7-22
- safety time 7-30, 10-5, 10-69, 1-16
- single LAN 1-17
- transmission time 10-9
Single coding 8-27, 8-30
Slave number 1-14, 1-15, 1-17, 7-40
Slot 2-2, 3-1, 3-3, 3-6, 3-8, 3-10
- addressing 3-21, 5-2, 10-53
- coding 3-13, 3-16, 3-17
- mechanical 3-13
- number 10-56, 6-9
Slot addressing Addressing
Softkey
- command line 1-32
- commands
- copy 1-32
- search 1-32
- operations
- delete 1-35
- save 1-36
- swap 1-35
- types 1-36
- routine 1-32, 1-36
Source
- addresses 10-12, 3-65
- slave 1-14, 5-14
Specification of type 1-6
Standard function block
Function block
Standard function block 6-25
- loadable 6-1
Standard value 3-12

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Standardization schematic 6-39
Starting COM 115F 1-6
”Start” operation 3-43
Startup
- analog input module 460 6-8
Statement list (STL) 2-1
Status
- byte 5-4, 5-13
- line 1-32
- processing 5-13
- register 5-12, 10-25
- scan 5-13
STATUS Signal status display
STATUS VAR Signal status display
STEP 5
- address counter 5-5
- illegal operations 10-12
- introduction 2-1
- limitations 10-11
STL Statement list
STL form 3-42
STOP 2-10
- ANLAUF 4-5
- loop 1-22
  - large 4-5
  - lesser 4-5
- mode 4-4, 4-5
  - operation 3-39
- program 4-10
Stray noise 3-35
Submodule
- ID 5-4
- identifiers 4-11
Substitution
- error 5-3, 5-4
- operations 2-9, 3-59
Subtraction 3-30
Subunit
- definition 1-37
- identifier 1-13, 1-27, 5-14, 7-5
- synchronization 1-8
Supply voltage 10-18
Symbols 1-6
SYNC 6-11
Synchronization 1-8, 7-30, 7-32, 10-9, 10-26
- block 3-17, 3-18, 3-20, 3-23, 6-10
  - FB 254 SYNC 6-10
- calls 10-5, 10-9, 10-26
- error 5-3
- event-driven 1-7
- FB 254 10-26
- intervals 6-11
- of process interrupt handling 2-10
- parameter 6-11
- pattern 2-7
SYS.OPS 4-3
System
- parameters 5-9
- response 10-57, 10-58
- startup 4-1, 10-27
  - checking a plant 4-19
  - configuration of the operating system and the I/O modules 4-16
  - enter subunit IDs 4-15
  - of an S5-115F in safety mode 4-17
- startup
  - overview 4-14
  - safety measures 4-18
  - steps 4-14
System components 1-2
- configuration
  - safety-oriented 2-1
- data 5-10, 2-11, 3-42, 3-65, 3-66
- data word 10-27
- handling 1-23
- operation 2-2, 2-3, 3-1, 3-65, 3-66, 3-69
System test 10-73

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

T

T BIT Analog value matching blocks
Technical specifications 8-1
Temperature sensor 6-20
Terminal
- assignment
  - front connector 3-35, 6-7
  - front connector AE 463 6-12
- connector 3-2, 3-3
Terminating connector 3-23, 5-2, 8-40, 10-53
Terminator 5-4
Test 5-10
- block 4-1
- block time 1-8, 1-39
  - calculated 1-9
- component 1-9
- conflict 10-63
- cycle 10-21, 10-29
- cycle time 1-8, 1-9, 5-10
- functions 4-6, 4-1 to 4-3
- mode 1-8
- operation 4-6, 4-19, 10-1, 3-44
- organization 4-6
- slices 10-7, 1-9
- supplementary 10-27
Thermocouple 6-4
Time
- accuracy 2-11
- base 3-17
- discrepancies 10-11
- interrupt 1-7, 10-9, 10-26, 10-68, 2-8, 3-53, 4-5
- interval 2-12
  - OB 13 1-10
- loading 3-14, 3-15, 3-17, 3-18
- out 5-3 to 5-5
- scan 3-19
- slices 4-5, 5-10, 5-11
- starting 3-19
- transferring 3-14
- update 1-11
Time-out (QVZ) 5-9
Timer 2-4, 2-10, 3-19
- operation 2-3, 3-16, 3-17, 3-19, 3-62
  - overview 3-16, 3-62
TNB operations 10-12
Tolerance variant 10-57, 10-58, 10-65, 1-11, 1-54
Total current consumption 3-32
Total nesting depth 2-5
Transceiver 10-68, 10-69
- BT 777 7-9, 10-68
Transducer 8-30, 6-20
- connecting 6-8
- four-wire 6-8
- two-wire 6-8
Transfer (cont.)
- block 7-32
- byte 10-5
- error 5-3, 5-4
- memory 7-45
- menu 1-26
- operations 2-9, 3-10, 3-13 to 3-15, 3-61, 3-65
  - overview 3-61, 3-65
Transferring an input/output value 3-13
Transitional-pulse relay 3-72
Transmission
- data byte 1-16
- time of the SINEC L1-LAN 10-9
Trick programming 10-12, 3-76
TTY interface 2-2
Two's complement 2-20, 3-71
Two-wire transducer 6-8
Type
- access to the analog value 6-4
- display 1-36
- entry 1-35, 1-36
- error 5-10, 5-17, 5-18
- input 1-29, 1-30, 1-37 to 1-39, 1-41, 1-42, 1-44, 1-46, 1-49
- matrix 1-32
- mixes 1-37
- number 1-30, 1-32, 1-35, 1-36, 1-53

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

U

UDGR Analog value matching blocks
Unified value 6-5
Uniform value 10-10, 10-24, 10-25, 10-63
- generation 10-63
Unit value 1-47
Updating
- cyclic 1-43
- user cycle time 1-10
Use
- safety-related 6-22
User
- interface 2-10
- locations 5-11
- memory 10-1
- message frame 7-11
- program check 2-2
- times 10-7, 6-1, 6-13
  - accuracy 10-9
  - updating 6-13
Using the programmer 1-14
- outside a cabinet 3-45

V

Visual checks 9-2
Visualization 7-10

W

Wear-out failures 9-2
Wire-break 6-17, 8-27, 8-30, 9-5, 9-6, 1-12, 6-6
- detection 6-26, 1-47
  - with the FB 250 ANEI 6-26
  - per hardware 6-26
  - per user program 6-26
- limit 1-8, 1-12, 6-7
- monitor 6-16, 6-7
- monitoring 10-37, 10-38
  - input ranges 10-38
- signalling 6-26
  - for resistance thermometers 6-27
Wiring 3-27
- arrangement
  - inside a cabinet 3-42
  - outside buildings 3-43
  - outside a cabinet 3-42
Word
- addressing 10-52
- operations 2-11, 10-12

X

XA Analog value matching blocks

Index S5-115F Manual, Volume 1
Index S5-115F Manual, Volume 2

Siemens AG
AUT 125 Doku
Postfach 1963

D-92209 Amberg
Federal Republic of Germany

From:

Your Name: _____

Your Title: _____

Company Name: _____

Street: _____

City, Zip Code: _____

Country: _____

Phone: _____

Please check any industry that applies to you:

- ☐ Automotive
- ☐ Chemical
- ☐ Electrical Machinery
- ☐ Food
- ☐ Instrument and Control
- ☐ Nonelectrical Machinery
- ☐ Petrochemical
- ☐ Pharmaceutical
- ☐ Plastic
- ☐ Pulp and Paper
- ☐ Textiles
- ☐ Transportation
- ☐ Other _____

Remarks Form

Your comments and recommendations will help us to improve the quality and usefulness of our publications. Please take the first available opportunity to fill out this questionnaire and return it to Siemens.

Title of Your Manual: _____

Order No. of Your Manual: _____

Edition: _____

Please give each of the following questions your own personal mark within the range from 1 (very good) to 5 (poor).

1. Do the contents meet your requirements? ☐
2. Is the information you need easy to find? ☐
3. Is the text easy to understand? ☐
4. Does the level of technical detail meet your requirements? ☐
5. Please rate the quality of the graphics/tables: ☐

Additional comments:

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____